UNIVERSIDADE FEDERAL DE SANTA CATARINA

CENTRO TECNOLÓGICO

CURSO DE SISTEMAS DE INFORMAÇÃO

DESENVOLVIMENTO DO MÓDULO DE PLANEJAMENTO E ACOMPANHAMENTO DE FROTA PARA A BIBLIOTECA DO PROJETO VIA DIGITAL

AUTOR(A): ALECINDRO STEINKE CASTILHO

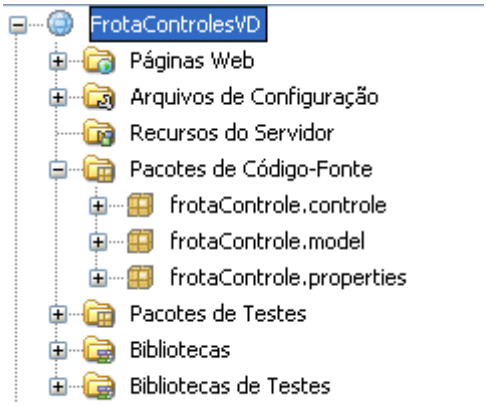
ANTEPROJETO DE PESQUISA PARA TRABALHO DE CONCLUSÃO DO CURSO DE SISTEMAS DE INFORMAÇÃO

ORIENTADOR(A): PROF. JOSÉ EDUARDO DE LUCCA
BANCA: PROF. JOÃO BOSCO DA MOTA ALVES
PROF. VITÓRIO BRUNO MAZZOLA

FLORIANÓPOLIS, 27 DE MAIO DE 2007

LISTA DE FIGURAS

SUMÁRIO



1 INTRODUÇÃO

O software, segundo Pressman (2002), assume um duplo papel, ou seja, ele é um produto e, ao mesmo tempo, o veículo para entrega do produto. Como produto ele disponibiliza o potencial de computação presente no computador ou, mais amplamente, numa rede de computadores local ou na Internet. Por outro lado, o software é um transformador de informação, quer resida em telefone celular, quer opere em um computador de grande porte.

Ele produz, gera, adquire, modifica, exibe ou transmite informação. Isto é, o software funciona como um veículo de entrega do produto mais importante da nossa época – a informação. Ou seja, o software é um programa de computador que permite escrever textos, planilhas, navegar e se comunicar pela Internet, entre tantas outras funções.

Segundo o professor Rogério Cid Bastos, a década de 90, em especial a sua segunda metade, foi marcada pela evolução do uso de softwares em todos os setores produtivos da sociedade. E o surgimento de uma nova tecnologia: a tecnologia *Web*. A *Web* surgiu com o objetivo de formar um repositório do conhecimento humano (BERNESS-LEE, 1994) e divulgação de informações. Mas ao longo do tempo essa tecnologia foi sendo modificada de forma a incorporar novos recursos e funções. Logo após a fase inicial, foi incorporado como um meio de marketing e propaganda para divulgação de empresas e seus produtos. Em seguida desenvolveu-se o comércio eletrônico e, por conseguinte sistemas de apoio e sistemas

internos, também chamados de *extranets* e *intranets*. Hoje, quase tudo que fazemos ou com que interagimos, seja entretenimento, educação, economia, segurança, transportes, saúde, etc passa pelo uso de informação e sistemas de informação que têm muitas vezes como um de seus elementos a tecnologia *Web*.

Antes do surgimento da *Web* o termo "Software Livre" era praticamente inédito fora do círculo da computação. Segundo Humberto Rossetti Baptista, com a explosão da Web foi dado um súbito destaque a uma nova categoria de programas que praticamente 'carregava' a *Internet* movimentando diversos serviços e em alguns casos indo até ao sistema operacional das máquinas que compõem a rede. A função do modelo de software livre é implementar e manter a "liberdade" relativa a um programa. A motivação disto é evitar que o conhecimento seja retido, permitir que as pessoas se ajudem e identifiquem os autores em seus trabalhos e derivados.

Mas como surgiu o termo "Software Livre"?

Em 1983 Richard Stallman, um brilhante programador do MIT, ficou muito aborrecido quando viu o resultado de um trabalho acadêmico em que participara ser vendido pelo MIT a uma empresa e ser "trancado" para sempre por trás de contratos de licença impenetráveis. Com isto Stallman pediu demissão e formalizou o conceito de software livre em um manifesto no qual apresentava e discutia a definição e a versão inicial da licença de uso de um programa livre: a licença GNU ou GPL. O termo GNU significa '*is not Unix*' e o objetivo do projeto é

criar um sistema operacional e aplicativos livres para as pessoas usarem. E Para defender a noção de software livre Stallman fundou a **Free Software Foundation** que tem como filosofia os seguintes pontos principais:

- Liberdade para conhecer (código fonte disponível);
- Liberdade para alterar (programa pode ser modificado);
- Liberdade para compartilhar (copiar, distribuir etc.);

Os benefícios de se usar programas livres são vários e entre eles temos: qualidade, bom suporte (apesar de não haver uma empresa responsável obtém-se respostas em pouco tempo, às vezes em horas, na internet) ou suporte contratado, participação nos destinos do programa muito mais direta (podendo chegar à implementação).

Além disso, não há gastos com o pagamento de licenças de uso nem *royalties*. Essa verba pode ser redirecionada para investimentos em Tecnologia de Informação (TI), treinamento de profissionais e aquisição de melhores equipamentos. E os programas podem ser adaptados de acordo com as necessidades específicas de cada usuário ou empresa. O usuário pode buscar as atualizações de código diretamente com a comunidade de desenvolvedores daquele aplicativo ou sistema, via internet, uma vez que as melhorias promovidas são compartilhadas e tornadas públicas. Também existem as empresas que customizam seus produtos, vendem suporte e treinamento para estas plataformas. Ou seja, flexibilidade é uma palavra-chave para quem trabalha com software livre.

Toda essa liberdade gera uma evolução constante e compartilhada, que faz do software livre uma opção cada vez mais adequada e eficiente para o governo federal, estadual e municipal. No entanto, a maioria dos municípios brasileiros sofre de carência de recursos. Seja carência de recursos financeiros, seja de carência de recursos humanos, não possuindo pessoas habilitadas a gerenciar ou até mesmo compor um corpo técnico especializado em informática. E temos o caso da Prefeitura de Rio das Ostras no estado do Rio de Janeiro, onde a Prefeitura tem como política a adoção de software livre, mas quando fizeram uma licitação para um Sistema de "Prefeitura Eletrônica", o vencedor da licitação foi a proposta que apresentou o menor preço, com a tecnologia .NET, da Microsoft. Caso houvesse algum software livre similar ao Sistema da Prefeitura Eletrônica, um sistema de tributos "*on line*" adotado pela prefeitura de Rio das Ostras, o custo da aquisição seria menos oneroso, pois só haveria necessidade de algumas customizações. E nesse foco é que trabalha o projeto Via Digital.

A proposta do projeto Via Digital é a implantação de um centro de referência virtual (portal) que permita o acesso e o desenvolvimento compartilhado de software livre para gestão municipal, propiciando a construção de um acervo público de soluções com foco na realidade das pequenas prefeituras. Tudo via Software Livre (SL).

E nesse trabalho iremos desenvolver o módulo de planejamento e acompanhamento de frota para a biblioteca do projeto Via Digital.

1.1 APRESENTAÇÃO

Segundo o site "http://www.viadigital.ufsc.br/" o projeto Via Digital pretende estimular uma nova dinâmica em torno da oferta de soluções de software livre para prefeituras. Viabilizar a informatização livre de pequenas prefeituras através de um centro de referência on-line de soluções em gestão municipal é seu objetivo principal. E para isso, precisamos disponibilizar um conjunto de módulos integráveis que atenda as atividades básicas de uma prefeitura. Entre esses módulos encontra-se o módulo de planejamento e acompanhamento de frota. Para desenvolver esse módulo, vamos nos basear na consultoria realizado pela empresa SWFactory Consultoria e Sistemas Ltda. para a Prefeitura Municipal de Amparo, onde foi feito o levantamento dos requisitos necessários para desenvolvimento desse sistema. Essa consultoria foi realizada em Abril de 2006 e após uma análise dos requisitos verificamos que o modelo proposto é aplicável no contexto do projeto Via Digital.

1.2 FORMULAÇÃO DO PROBLEMA

O projeto Via Digital não possui um módulo que abrange o planejamento e acompanhamento de frotas para prefeituras. E a maioria das prefeituras brasileiras é carente de recursos. Se não conseguem investir em modernização, em novas soluções de sistemas que venha a otimizar o uso dos recursos financeiros, imagine adotar sistemas que requeiram grandes recursos de manutenção e pessoal. Um sistema que possibilite gerenciar uma frota de

veículos, sabendo os custos gerados com cada veículo, manutenção e consumo de combustível, se existe possibilidade de alocação do veículo, motoristas capacitados a utilizar o veículo requerido, são necessidades existentes nas prefeituras.

1.3 JUSTIFICATIVAS

Como justificativa para desenvolvimento do projeto de desenvolvimento do módulo de planejamento e acompanhamento de frota temos:

- Desenvolvimento de um sistema que fará parte do projeto Via Digital, projeto esse que tem forte cunho social;

- Proporcionar as prefeituras que têm dificuldades de compra e implementação de softwares uma solução viável para controle de frotas de veículos;

- Proporcionar um modelo de sistema que pode ser modificado, distribuído e implementado de acordo com a necessidade de cada prefeitura;

- Proporcionar o acesso ao sistema de controle de frotas através do projeto Via Digital através da *Web*;

1.4 OBJETIVOS

1.4.1 Objetivo Geral

O objetivo geral do trabalho é realizar a validação e implementação do módulo de planejamento e acompanhamento de frota de prefeituras do projeto Via Digital (GENESS – UFSC, 2006).

1.4.2 Objetivos Específicos

Desenvolver um Sistema *Web* que irá auxiliar o administrador da frota a planejar as manutenções, assim como acompanhar os gastos com cada veículo. Estudar e desenvolver baseado na consultoria realizada pela empresa SWF Consultoria e Sistemas com base em estudos realizados na prefeitura de Amparo no Estado de São Paulo.

1.5 DELIMITAÇÃO DO ESCOPO

O foco do nosso trabalho é auxiliar o administrador da frota a planejar todas as manutenções, assim como acompanhar todos os gastos com cada veículo da frota (automóvel, ônibus, tratores, etc.). Não trataremos do rastreamento ou localização do veículo em trânsito.

2 O MERCADO

Analisando e pesquisando sistemas de controle de frota similares, no mesmo foco, que envolve software livre e tenha aplicação voltada a prefeituras encontramos duas alternativas ao nosso sistema.

Na cidade de Itajaí, no estado de Santa Catarina, encontramos a CTIMA - Centro Tecnológico de Informação e Modernização Administrativa (http://ctima.itajai.sc.gov.br/ctima.php) responsável pela estrutura tecnológica da prefeitura daquela cidade. A CTIMA desenvolveu diversos softwares livres voltados para a administração pública, todos disponíveis na sua página para download. Entre os softwares desenvolvidos encontra-se o i-Frotas. É um software ainda em desenvolvimento, sendo criado em na linguagem PHP e tem como principio o gerenciamento e o controle de itens como: abastecimento, revisões, viagens, licenciamentos, combustíveis, pagamento de seguros, reservas de veículos, serviços de troca de óleo, peças, pneus, recapagens, entre outros. Segundo o CTIMA, o I-Frotas permite ao gestor avaliar, de forma clara e precisa, todos os gastos com a frota municipal, buscando a melhor relação custo-benefício quanto ao uso dos veículos. No site do CTIMA a documentação ainda não existe, apenas com algumas fontes para download não permitindo nem saber qual o banco de dados utilizado. O grande diferencial do nosso sistema em relação ao do CTIMA, baseado no que está disponível para

análise, é por ser desenvolvido em Java, no qual permite uma maior robustez, escalabilidade e segurança.

Outro software encontrado foi o SIAMWEB (encontrado no site http://www.ccanet.com.br/index.php?pg=1012). Desenvolvido pela CCA Consultoria e Administração, empresa que oferece produtos totalmente orientados à *Web*, baseados exclusivamente em Softwares Livres. O SIAMWEB é um software desenvolvido especialmente para Prefeituras Municipais. Segundo o site da empresa, o software é capaz de controlar as receitas e despesas do município, fornecendo relatórios gerenciais para os mais diversos departamentos. Desenvolvido com ferramentas de software livre, o SIAMWEB também torna o custo de implementação uma alternativa mais econômica. Na verdade, o SIAMWEB é um sistema ERP para prefeituras municipais, no qual está inserido um subsistema de controle de frotas. Algumas prefeituras adotaram esse software, tais como a prefeitura de do Rio de Janeiro, no estado do Rio de Janeiro. Em relação à tecnologia adotada, nada conseguimos descobrir, por falta de documentação. Mas descobriu-se através de editais de compra e análises orçamentárias que o sistema chega a custar R$ 500.000,00 (quinhentos mil reais), o que foge do foco do nosso software, que é atender pequenas prefeituras carentes de recursos.

3 DESENVOLVIMENTO

Como nosso software tem como base a análise de requisitos do sistema feitas pela empresa SWFactory, iremos discorrer nesse tópico sobre quais tecnologias iremos aplicar ao nosso software, bem como alterações realizadas dessa análise.

3.1 Análise de Requisitos não-funcionais

Como dito anteriormente, as prefeituras brasileiras são carentes de recursos. Se não conseguem investir em modernização, em novas soluções de sistemas que venham a otimizar o uso dos recursos financeiros, imaginem adotar sistemas que requeiram grandes recursos de manutenção e pessoal. Com esse enfoque chegamos ao primeiro requisito: a aplicação deve apresentar uma interface web, para não haver necessidade de instalação e configuração de computadores clientes. Todo o serviço técnico reduz-se ao servidor, no qual pode ser terceirizado, inclusive o próprio servidor. Ainda, com o mesmo enfoque de simplicidade de manutenção, a aplicação inicialmente deve ser implantada em um container web. Mas é importante que a aplicação seja facilmente escalável, para que depois possa acomodar componentes de negócio em um container a parte (container *EJB*, por exemplo). E esse é o nosso terceiro requisito. Como quarto requisito, podemos afirmar que a aplicação deve suportar múltiplos usuários simultâneos. O quinto requisito é que a aplicação pode ser fornecida como um framework, para diversas prefeituras, que podem ou não utilizar um container de componentes de negócio em sua infra-estrutura. É importante também que a

aplicação seja flexível, para utilizar bancos de dados relacionais diversos, muitos já existentes nas prefeituras. Além disso, as aplicações têm de usar software livre, para que não haja custo de aquisição de software, bem como possuam portabilidade, ou seja, rode em diversos sistemas operacionais.

3.2 Requisitos do Projeto Via Digital

Além dos requisitos apresentados anteriormente, o projeto Via Digital define requisitos a serem pré-estabelecidos para construção de componentes. Esses requisitos pré-estabelecidos são para que possam rodar o software desenvolvido nos servidores do projeto. Os requisitos são os seguintes:

A implementação dos componentes devem estar baseados na arquitetura J2EE e a tecnologia empregada deve estar baseada na arquitetura apresentada na figura abaixo:

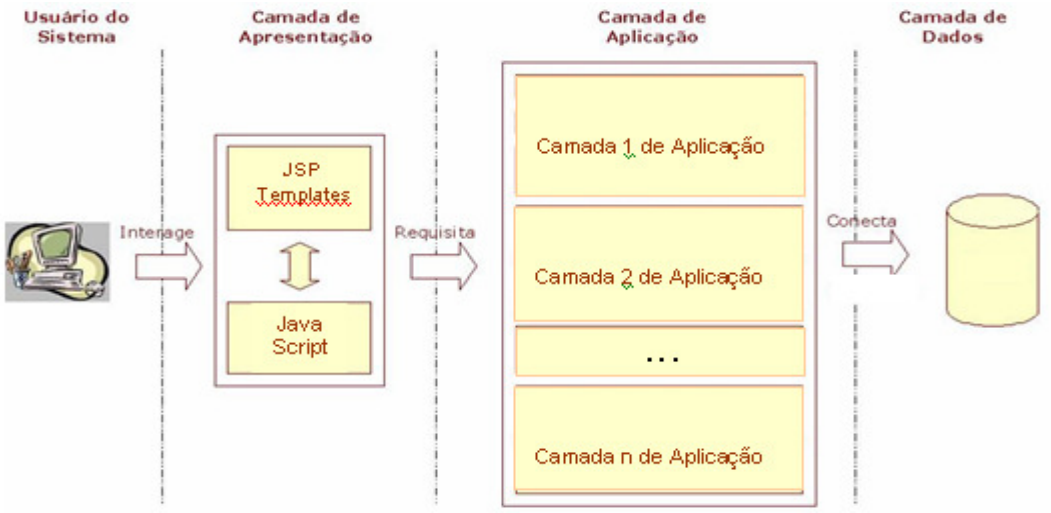


Figura 1 - Arcabouço Arquitetural

Fonte: Projeto Via Digital – Anexo_A4_tecnologia

Na camada de apresentação deverão ser empregadas as seguintes tecnologias: HTML na versão 4.0, *JavaScript* 1.1, JSP 1.1 (*Java Server Pages*) segundo a plataforma J2EE 1.4 e ter compatibilidade com os navegadores *Internet Explorer* (versão 5.0 e superiores) e *Mozilla Firefox* (versão 1.3 e superiores). Na camada de aplicação a linguagem deve ser Java com padrão de desenvolvimento J2SE SDK na versão 5.0 ou superior, *framework Struts* na versão 1.1 ou superior para implementação do padrão MVC – *Model*-2 e para realização do controle de fluxo entre as camadas da arquitetura dos componentes, conforme mostrado na figura 2. E o *container web* deverá ser o *Tomcat* na versão 5.0 ou posterior.

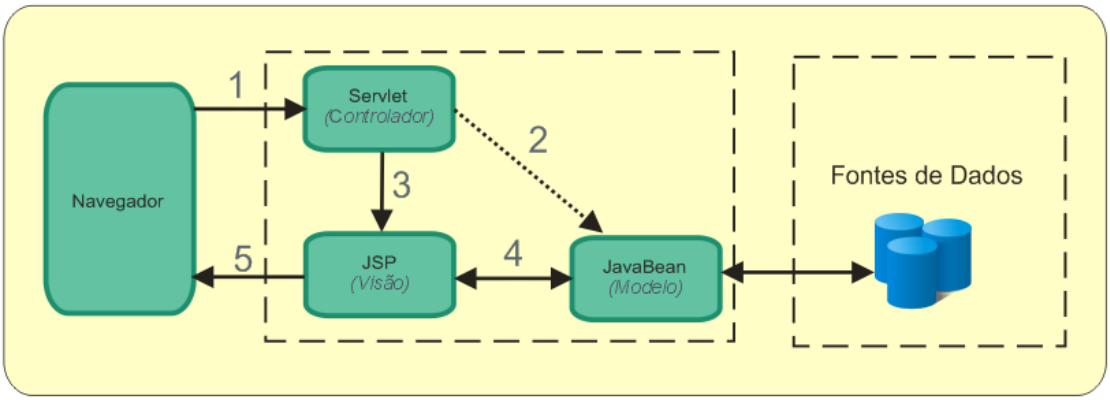


Figura 2 – MVC - Model-2

Fonte: Projeto Via Digital – ANEXO_A4_tecnologia

Na tecnologia de camada de dados é exigido para persistência o *framework hibernate* na versão 3.0 ou posterior e banco de dados o armazenamento deverá ser efetuado em *POSTGRESQL* na versão 8.0 ou superior.

3.3 Análise dos Requisitos

Interpretando os requisitos não funcionais, adequando aos requisitos exigidos pelo projeto Via Digital, e em consenso com os responsáveis pelo projeto, optamos por novas tecnologias, atualizações de tecnologias solicitadas pelo projeto, outras similares a tecnologias requeridas e algumas novas tecnologias que contemplam os requisitos e que não comprometem a compatibilidade com o servidor e outros módulos do projeto.

3.3.1 Padrão de Arquitetura

A plataforma Java J2EE 1.4 oferece uma base de desenvolvimento ampla, estável, bem testada e com alto desempenho final. No entanto essa versatilidade acarreta certo prejuízo no tempo de desenvolvimento do software. Comparativamente a plataforma .NET (Microsoft) oferece uma maior produtividade no desenvolvimento, no entanto é uma tecnologia nova e instável, apresentando vários problemas no desenvolvimento de software, além de ser proprietária.

A plataforma Java EE5 possui as mesmas funcionalidades da plataforma J2EE 1.4, mas com maior simplicidade. Por isso, iremos optar pelo padrão *Java Enterprise Edition* 5

(superior a tecnologia J2EE 1.4). É livre, funcional, robusto e contempla todos os requisitos. Apenas requer uma atualização do servidor do projeto Via Digital para o padrão Java EE 5, e é compatível com versões anteriores.

A plataforma Java EE 5 permite uma arquitetura flexível sendo que tanto o container *web* quanto o container EJB são opcionais. Como nosso trabalho tem como foco implementar um módulo de planejamento e acompanhamento da frota de prefeituras para *web*, não aplicaremos *EJB*. E o projeto Via Digital não especifica servidores de aplicações. Assim, aplicaremos padrões e estratégias de projeto que consideramos necessários à construção de aplicações J2EE profissionais não distribuídas. Como, não usaremos EJB, apenas o container *web*, que será o *Tomcat*, definido pelo projeto. Isso não impede que futuramente seja implementado o servidor de aplicações e EJB. Portanto, não ferimos nossos requisitos iniciais, mantendo a escalabilidade e simplicidade de manutenção.

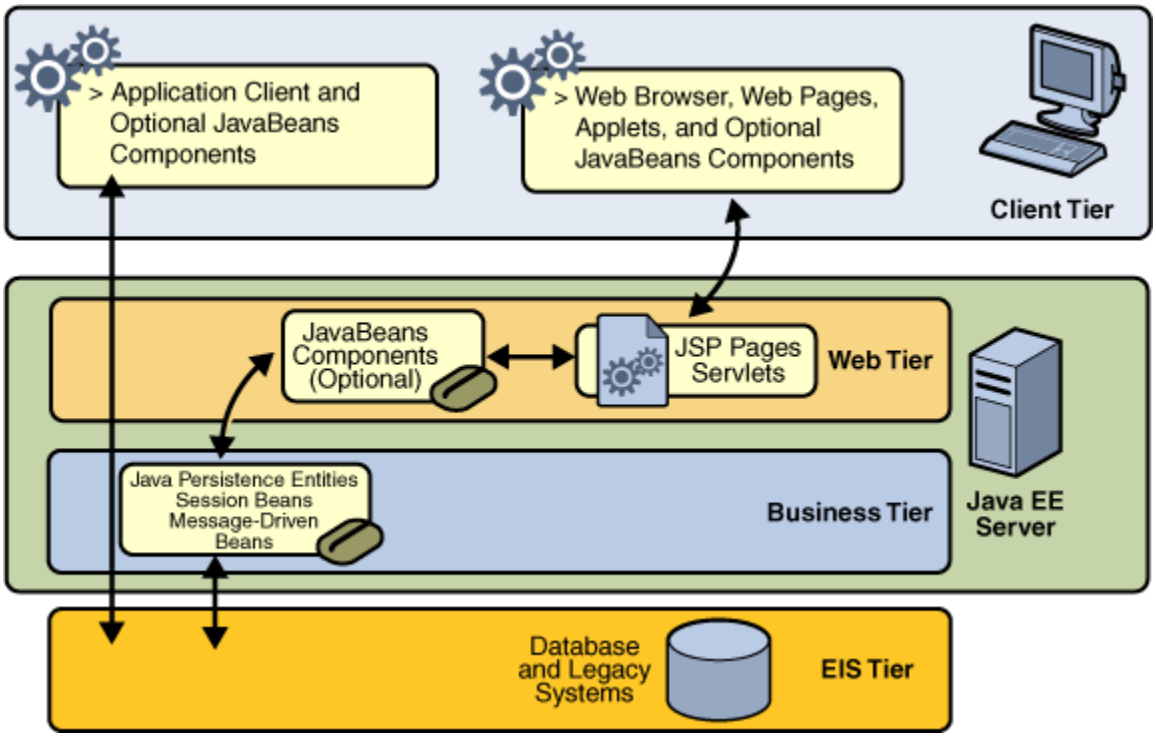


Figura 3 - Arquitetura Java EE 5

Fonte: Tutorial Java EE 5

3.3.2 Framework Adotado

O *Struts* oferece uma infra-estrutura de desenvolvimento web orientado pelos principais *designs patterns* sugeridos para MVC. Este é o *framework* MVC mais difundido no mercado, e sua robustez e popularidade o tornam extremamente competitivo ao escolher entre tecnologias web. Mesmo com as facilidades e sucesso do *Struts*, havia no mercado uma autêntica demanda por mais produtividade de desenvolvimento. E ainda havia espaço para uma especificação de padronização de *frameworks* MVC. Então, aparece a proposta do Java Server Faces.

O Java Server Faces não é apenas mais um *framework*, é uma especificação de um modelo de programação baseado na arquitetura MVC. Essa especificação define um padrão para desenvolvimento de aplicações com alta produtividade e componentização. É um *framework* para criação de interfaces gráficas para aplicações *web*. É composto por um conjunto de *APIS* para representação de componentes de interface com o usuário, gerenciamento do estado dos componentes, tratamento de eventos de interface gráfica, validação dos dados de entrada, e navegação. O *Java Server Faces* será utilizado em páginas *JSP* através de *taglibs* (bibliotecas), que associadas a componentes de interface de usuário no lado do servidor, que por sua vez renderizam em HTML. Também poderemos incorporar as páginas *JSP*, *Javascript* e *Ajax* (Asynchronous Javascript e XML).

Além da produtividade, o que nos faz adotar esta tecnologia é a definição de um padrão que está sendo adotado em massa pelo mercado, e que possibilita a portabilidade do software desenvolvido entre soluções de diferentes fornecedores.

Mas o que é MVC? Segundo a Wikipédia, Model-View-Controller (MVC) é um padrão de arquitetura de software. Em aplicações complexas, que enviam uma série de dados para o usuário, o desenvolvedor freqüentemente necessita separar os dados (Model) da interface (View). Desta forma, alterações feitas na interface não afetarão a manipulação dos dados, e estes poderão ser reorganizados sem alterar a interface do usuário. O MVC resolve este problema através da separação das tarefas de acesso aos dados e lógica do negócio da apresentação e interação com o usuário, introduzindo um componente entre os dois: o

Controller. MVC é usado em padrões de projeto de software, mas MVC abrange mais da arquitetura de uma aplicação do que é típico para um padrão de projeto.

3.3.3 Camada de Apresentação

No nosso projeto a função camada de apresentação é interação com o usuário. Será utilizado o ambiente web para apresentar os dados ao usuário através de código *HTML*, *Javascript* e folhas de estilo *CSS*. Vamos procurar explorar ao máximo o conceito de *Web* 2.0 com o intuito de fornecer ao usuário uma interface rica e semelhante a interfaces de softwares *Desktop*. Isso proporciona ao usuário um menor desconforto ao trabalhar com sistemas *Web*.

Em se tratando de sistemas *Web* a camada de apresentação é dividida em duas etapas: a executada pelo servidor, que é responsável por montar e encaminhar ao cliente o código gerado, e pelo lado cliente que recebe o código gerado pelo browser ou uma aplicação específica.

No nosso sistema foram utilizadas diversas tecnologias e frameworks com o intuito de aprimorar e gerar produtividade na programação, bem como auxiliar no enriquecimento da interface do usuário.

3.3.3.1 Tecnologias e aplicações da camada de apresentação:

Apache Tomcat: Segundo Wikipedia (http://pt.wikipedia.org/wiki/Apache_Tomcat) o Tomcat é um servidor de aplicações Java para web. Tecnicamente o Tomcat é um *container*

Web, cobrindo parte da especificação J2EE com tecnologias como Servlet e JSP, e tecnologias de apoio relacionadas como *Realms* e segurança, JNDI *Resources* e JDBC *DataSources*.

Java Servlet: Segundo Wikipédia (http://pt.wikipedia.org/wiki/Servlet), *Servlet* é um componente que disponibiliza ao programador da linguagem Java uma interface para o servidor web (ou servidor de aplicação), através de uma API. As aplicações baseadas no Servlet geram conteúdo dinâmico (normalmente HTML) e interagem com os clientes, utilizando o modelo *request/response*. Os *servlets* normalmente utilizam o protocolo HTTP, apesar de não serem restritos a ele. O Tomcat é responsável pela execução dos *Servlets*.

JavaServer Pages: *JavaServer Page* (JSP) é uma tecnologia utilizada no desenvolvimento de aplicações para *Web,* composta de componentes, que permite aos desenvolvedores e designers um rápido desenvolvimento e fácil manutenção. Uma página criada com a tecnologia JSP, depois de instalada em um servidor de aplicação compatível com a tecnologia *Java EE,* é transformada em um *Servlet*.

JavaServer Faces: Os componentes que fazem parte da do framework JSF fazem parte da nossa camada de apresentação. No nosso sistema a versão utilizada foi a *JavaServer Faces* 1.1.

AJAX - Asynchronous Javascript And XML: Segundo o Wikipédia (http://pt.wikipedia.org/wiki/AJAX_%28programa%C3%A7%C3%A3o%29) é o uso

sistemático de tecnologias providas por navegadores, como *Javascript* e XML, para tornar páginas mais interativas com o usuário, utilizando-se de solicitações assíncronas de informações.

Tomahawk: *MyFaces Tomahawk*, assim como o RichFaces, é uma biblioteca de componentes JSF. É open-source e atualmente está na versão 1.1.6. Faz parte do projeto *MyFaces* da Apache Foundation e a página do projeto é http://myfaces.apache.org/tomahawk/index.html.

Ajax4JSF: Ajax4jsf é um framework que adiciona suporte *AJAX* a componentes JSF sem a necessidade de codificar *JavaScrips*. O *Ajax4JSF* é open-source, está atualmente na versão 3.1.1 e é mantido e desenvolvido pela parceria realizada entre a Exadel e JBoss. Pode ser atualmente encontrado no site http://labs.jboss.com/jbossajax4jsf/.

***RichFaces*:** *RichFaces* é uma biblioteca de componentes JSF, um framework para fácil integração do AJAX dentro das aplicações *WEB*, proporcionando rápido desenvolvimento. O *RichFaces* é open-source, está atualmente na versão 3.1.1 e é mantido e desenvolvido pela parceria realizada entre a Exadel e JBoss. Pode ser atualmente encontrado no site http://labs.jboss.com/jbossrichfaces/.

Sandbox: *Sandbox*, assim como o *Tomahawk*, é uma biblioteca de componentes JSF e faz parte do projeto *MyFaces* da Apache Foundation. É open-source e atualmente está na versão 1.1.7 e a página do projeto é http://myfaces.apache.org/sandbox/index.html

CSS: *Cascading Style Sheets* (CSS) é uma linguagem de estilo utilizada para definir a apresentação de documentos (como fontes, cores, espaços) escritos em uma linguagem de marcação, como HTML ou XML. Seu principal benefício é prover a separação entre o formato e o conteúdo de um documento. As especificações do CSS são mantidas pelo *World Wide Web Consortium* (W3C).

JavaScript: Segundo o Wikipédia (http://pt.wikipedia.org/wiki/JavaScript) *JavaScript* é uma linguagem de programação criada pela Netscape em 1995, que a princípio se chamava LiveScript, para atender, principalmente, as seguintes necessidades: validação de formulários no lado cliente (programa navegador), interação com a página web, tipagem dinâmica - tipos de variáveis não são definidos, e é interpretada, ao invés de compilada.

Sua união com o CSS é conhecida como DHTML. Usando o Javascript, é possível modificar dinamicamente os estilos dos elementos da página em HTML.

***Prototype*:** O *Prototype* é um framework *JavaScript* que proporciona fácil desenvolvimento de aplicações web dinâmicas. É open-source, atualmente está na versão 1.6.0 e a página do projeto é http://www.prototypejs.org/.

Script.aculo.us: O *Script.aculo.us* é uma biblioteca de funções *JavaScript*, baseado no *Prototype*, que proporciona fácil desenvolvimento de aplicações web dinâmicas. É open-source, atualmente está na versão 1.7.0 e a página do projeto é http://script.aculo.us/.

***JasperReports*:** é uma poderosa ferramenta open-source para gerar relatórios, sendo nos formatos PDF, HTML, XLS, CSV e XML ou direto para impressão. É escrito em Java e pode ser usado para gerar conteúdo dinâmico em aplicações ou páginas *Web*.

3.3.4 Camada de Controle

No nosso sistema o controle é realizado pelo framework JSF através de um *servlet* denominado *FacesServlet*, por arquivos de configuração e por um conjunto de manipuladores de ações e observadores de eventos. O *FacesServlet* é responsável por receber requisições da camada de apresentação, redirecioná-las para camada de dados e então remeter uma resposta a camada de apresentação.

3.3.5 Camada de Modelo de Dados

Qual alternativa deve-se usar para a camada de persistência? O framework *JSF* nos proporciona diversas alternativas. As principais são JDBC, EJB/CMP (*Entity Beans*), *Hibernate* ou JPA. JDBC pode ser muito trabalhoso se considerarmos a independência do BD requerida. *Entity Beans*, que só funcionam dentro de *containers* EJB, estando assim

descartados pela exigência de ser possível rodar apenas em um container *web*. O *Hibernate* tem como principal problema a produtividade, com a necessidade de escrever arquivos de mapeamento longos e complexos, sujeitando-se a erros e manutenção.

Já a *Java Persistence API (*JPA*)* é a especificação padrão da *Sun* para o gerenciamento de persistência e mapeamento objeto relacional, surgida na plataforma Java EE 5.0, e que foi baseado no *Hibernate* proposto pelo projeto Via Digital. A JPA tem o intuito de simplificar o desenvolvimento de aplicações Java EE e Java SE que utilizam persistência de dados e possui uma completa especificação para realizar mapeamento objeto relacional, utilizando anotações da linguagem Java (Java SE 5.0 ou superior). Também dá suporte a uma rica linguagem de consulta, semelhante à SQL, permitindo consultas estáticas ou dinâmicas. Com isso, adotaremos a JPA (Java Persistence API) que atende nossos requisitos e vai facilitar muito a reutilização de software colocada anteriormente. Além disso, com a JPA é possível trocarmos o banco de dados alterando apenas o arquivo XML de persistência, atendendo nosso requisito de flexibilidade.

Quando se decide utilizar o JPA, é necessária a escolha de um provedor JPA. Como esta API é uma especificação para *frameworks* de persistência, existe a necessidade de provedores JPA. Por padrão, a implementação de referência é o *Oracle Toplink Essentials*. Também existem outros provedores JPA no mercado, como o *Hibernate Entity Manager*, *Bea Kodo* e o *ApacheJPA*. Utilizaremos *Oracle Toplink Essentials* devido à familiaridade do programador com este provedor.

Para nossa aplicação ser integralmente funcional deverá abordar algum banco de dados. Nessa abordagem, para o ambiente de desenvolvimento iremos utilizar o banco de dados *POSTGRESQL*, por ser requisito estabelecido pelo projeto Via Digital. É livre e robusto. Mas para que nossa aplicação fosse perfeitamente escalável, recomendaríamos o banco de dados Derby da Apache em parceria com a IBM, que é livre e funcional, além de ser um banco de dados escrito também na linguagem Java. O Derby é facilmente migrado para o banco de dados da IBM o DB2 Express-C Edition (sem a necessidade de reescrever o banco de dados) que também é gratuito, e comparado a banco de dados como o Oracle e SQL Server da Microsoft. E caso a aplicação justifique um maior processamento, existe a possibilidade ainda de migrar para a versão paga do DB2.

No entanto, caso usássemos o Derby, haveria necessidade de migrarmos todos os módulos do projeto Via Digital para haver uma uniformidade de desenvolvimento e manutenção. Por isso adotaremos o *POSTGRESQL* versão 8.2, a última versão estável até o momento.

3.4 O Sistema Planejamento e Acompanhamento de Frota

Para desenvolver nosso sistema começamos com a análise de requisitos realizado pela empresa SWFactory. Todos os requisitos funcionais e não funcionais estão apresentados no Anexo I. Ao fazer a análise deparamos com alguns possíveis problemas e/ou modificações que deveriam ser realizadas, já que os requisitos tinham sido feitos para a Prefeitura

Municipal de Amparo (SP) e nosso sistema deve ser mais abrangente, para qualquer prefeitura.

3.4.1 Alterações dos requisitos

Aqui iremos apresentar as alterações realizadas. Os requisitos estão descritos na íntegra no Anexo I.

No documento original da SWFactory, exigia que *"todo veículo da frota ao retornar para a garagem deverá realizar o abastecimento, completando o tanque e que só seria permitido encerrar a atividade com a nota de abastecimento do integrante, e com certificação de que o tanque de combustível está completo."*

Resolvemos extrair essa exigência, visto que muitos municípios brasileiros são extramente pequenos, não ultrapassando o raio limítrofe urbano de 5 Km (IBGE/2000). Como exigir de um carro que abasteça a cada saída da prefeitura, se muitas vezes não chegam a gastar um tanque de combustível por mês?

Mesmo assim, caso a prefeitura tenha interesse em manter essa exigência, poderá fazê-lo, cabendo instruir o operador do sistema a fazê-lo através de normas de trabalho desta.

Alterações no Cadastro de Integrantes (Veículos): Outra solicitação requerida pelos requisitos é o relatório de Quilometragem percorrida com cada Veículo por Período,

bem como o agendamento de manutenções. Para que fossem atendidas essas solicitações foram adicionados ao cadastro de Integrantes (Veículos) os seguintes campos:

- “kminicial”: no momento de cadastrar um veículo o usuário deverá digitar a kilometragem do veículo. Esse campo é de preenchimento obrigatório caso o veículo seja do subgrupo automóvel. Caso o veículo seja de outro subgrupo esse campo é setado nulo e não aparece no cadastro.

- “kmatual”: campo de atualização automática. A cada retorno do veículo é digitado na tabela de Finalizar a Atividade a kilometragem final do veículo. Esse campo é de preenchimento obrigatório caso o veículo seja do subgrupo automóvel. Caso o veículo seja de outro subgrupo esse campo é setado nulo e não aparece no momento de finalizar atividade.

- “ativointegrante”: Em nenhum momento os requisitos se referem ao veículo como parte integrante da frota. Em muitos casos o veículo pode ser retirado da frota por sinistro ou depreciação. Foi adicionado o campo “ativointegrante” definindo se o veículo ainda está disponível para trânsito ou se foi dado “baixa” do veículo. O veículo se manterá na base de dados para fins estatísticos. Ao cadastrar o veículo este campo virá automaticamente preenchido, mas permitindo ao usuário que modifique tal atributo. Ao editar um determinado veículo é possível mudar esse campo.

- “disponível”: para uma melhor performance do banco de dados e uma melhor programação orientada a objetos incluímos o campo “disponível” para sabermos se o veículo está a disposição de ser alocado ou não. Caso não esteja a disposição, é porque está alocado ou está em manutenção. Esse campo é preenchido automaticamente pelo sistema a cada atividade ou manutenção.

Alterações no Cadastro de Categoria de Habilitação: Foi inserido o campo descrição da categoria de habilitação, para um melhor entendimento do usuário.

Alterações no Cadastro de Motorista/Operador: Foram inseridos os campos “telefone”, “celular”, “ramal” para uma eventual necessidade de contato com o motorista, o sistema proporcione essas informações. Outro campo inserido foi “disponível”. Caso o motorista esteja realizando alguma atividade o campo é preenchido automaticamente. É possível o usuário indicar se o motorista está disponível ou não (férias, licença) através do cadastro de motoristas.

Alterações no Cadastro de Nota Fiscal de Gasto: a data de emissão da Nota Fiscal de Gasto não é preenchida automaticamente pelo sistema. É realizada pelo usuário e de forma obrigatória. Isso se deve pelo motivo que a Nota Fiscal de Gasto nem sempre é cadastrado no mesmo dia da emissão da Nota Fiscal. Por isso, incluímos o campo “datapreenchegasto”, preenchido automaticamente pelo sistema, a cada cadastro.

Alterações no Cadastro de Fornecedores: foram inseridos os campos “cep”, “page” e “email” para que o usuário possa ter um cadastro mais completo do fornecedor.

Alterações no Cadastro de Manutenção: foram inseridos os campos “kmprogramados”, “ ” (para automóveis - a kilometragem que o carro vai estar para fazer a manutenção), “horasPrevistas” (para tratores- número de horas que o trator deve percorrer para fazer a manutenção) e “horas”(número de horas do trator no momento da manutenção).

3.4.2 Diagrama de Classes

Nosso diagrama de classes foi definido com base no diagrama de casos de uso fornecido pela empresa SWFactory e com as alterações dos requisitos realizadas, e feito com o software JUDE Community (encontrado em http://jude.change-vision.com/jude-web/index.html). Uma imagem do diagrama de classes pode ser encontrada no anexo III.

Nosso diagrama de classes é composto por 17 classes sendo elas:

- Usuariosistema: classe que representa as informações dos usuários do Sistema;

- Perfilusuario: classe que representa as informações sobre os perfis de acesso ao sistema dos usuários. Serão criados automaticamente pelo sistema os perfis Administrador, Gestor e Responsável pela frota;

- Integrantes: classe que representa as informações sobre os veículos do sistema de controle de frotas;

- GrupoIntegrante: classe que representa os grupos de integrantes do sistema. Serão pré-cadastrados no sistema no momento da implantação os grupos automóveis e tratores;

- SubgrupoIntegrantes: classe que representa os subgrupos de integrantes do sistema. É uma especialização do GrupoIntegrante. Serão pré-cadastrados no sistema no momento da implantação os subgrupos: caminhão, caminhonete, microônibus, ônibus, utilitários, passeio, motocicletas (como especialização do grupo automóveis) e pá carregadeira, retro escavadeira, esteira, motoniveladora, rolocompactador (como especialização do grupo tratores);

- Tipocombustivel: classe que representa os combustíveis que cada integrante de frota poderá conter;

- Manutenção: classe que representa os agendamentos e manutenções dos integrantes do sistema;

- Atividade: classe que representa as atividades de entrada e saída dos integrantes de frota;

- Motoristaoperador: classe que representa os motoristas e/ou operadores dos integrantes de frota;

- Categoriahabilitacao: classe que representa a categoria de habilitação dos motoristas/operadores e dos integrantes de frota;

- Fornecedor: classe que representa os fornecedores de peças e serviços dos integrantes de frota;

- NotafiscalGasto: classe que representa as notas fiscais de compra realizada com fornecedores para os integrantes de frota;

- Itens: classe que representa itens que serão utilizados para manutenção, reparação ou gastos com os integrantes de frota

- Unidade: classe que representa as unidades dos itens dos integrantes de frota. No momento da implantação do sistema já deverá ser inserido as unidades unitário, litro e Homem/hora;

- NotafiscalgastoHasItensPK: classe que representa uma associação entre os itens cadastrados e a nota fiscal de gasto. Quando o mesmo item é comprado em quantidade essa classe faz a associação de cada item com a nota fiscal de gasto;

- NotafiscalgastoHasItens: classe que representa o corpo da nota fiscal de gasto. É uma associação entre a quantidade de itens gasto, a nota fiscal de gasto e seus respectivos itens;

- Notaservico: classe que representa os serviços que são realizados em um integrante de frota. Toda manutenção será considerada um serviço. E essa fará associação com o integrante de frota.

3.4.3 Modelagem do Banco de Dados

A partir do diagrama de classes, partimos para a modelagem e criação do nosso banco de dados. Usamos a ferramenta DbDesigner4 (que pode ser encontrado em http://fabforce.net/dbdesigner4/) para fazer nossa modelagem. No anexo IV apresentamos imagem da nossa modelagem.

3.4.4 O Código

Como dito anteriormente, nosso sistema foi todo implantado baseado no padrão de arquitetura MVC. O nosso model é composto por 17 classes “Pojo” de java e 1 arquivo XML que fazem a integração do nosso sistema com o banco de dados. O controller é composto por 37 classes java, sendo 17 classes “Pojo” que mapeam as ações do usuário na View e fazem a integração com o nosso model, 17 classes “Pojo” que mapeiam a conversão de componentes como objetos para String e vice-versa, 1 classe que é um servlet que controla a apresentação

de imagens, 1 classe que faz o tratamento de mensagens do sistema, 1 XML arquivo de mapeamento das classes java e controle de navegação entre páginas do browser. Em sistemas Web, todo e qualquer dado inserido na View é considerado String e é de responsabilidade das classes de conversão para que transformem data, números, etc em String para a View e em objetos para o banco e/ou sistema fazer a leitura dos dados adequada. E a nossa View é representada pelas classes JSP, JSF, HTML que será reinderizada na leitura pelos navegadores de internet.

4 REVISÃO BIBLIOGRÁFICA

Nossa revisão bibliográfica tem o intuito de apresentar a tecnologia Java para iniciantes ou leitores que desejam revisar e aprimorar os conceitos desta tecnologia. Grande parte da revisão foi um resumo e livre tradução do Tutorial *JavaEE5* da Sun MicroSystems.

4.1 A Plataforma Java

A plataforma Java é o nome dado ao ambiente computacional, ou plataforma, da empresa Sun Microsystems. O desenvolvedor de *software* cria programas para este ambiente através da linguagem de programação Java e de um conjunto de ferramentas de desenvolvimento[PTJAVA07]. Neste caso, a plataforma não se refere a um sistema operacional ou *hardware* específico, mas a um programa chamado de máquina virtual, e a um conjunto de bibliotecas que disponibilizam funcionalidades comuns. Por isso, um programa

escrito para a plataforma Java necessita desses dois componentes para ser executado: a máquina virtual Java, e um conjunto de bibliotecas de classe que disponibilizam um série de serviços para esse programa. O pacote de *software* que contém a máquina virtual e esta biblioteca de classes é conhecido como JRE (Java Runtime Environment).

A máquina virtual é o coração da plataforma Java. É o conceito de um processador "virtual", que executa os programas formados por *bytecodes* Java. Este *bytecode* é o mesmo independentemente do *hardware* ou sistema operacional do sistema em que o programa será executado. A plataforma Java disponibiliza um interpretador, a JVM, que traduz, em tempo de execução, o *bytecode* para instruções nativas do processador. Isto permite que uma mesma aplicação seja executada em qualquer plataforma computacional que possua uma implementação da máquina virtual.

Desde a versão 1.2 da JRE, a implementação da Sun da JVM inclui um compilador *just-in-time* (JIT). Com este compilador todo o *bytecode* de um programa é transformado em instruções nativas e carregado na máquina virtual em uma só operação, permitindo um ganho de desempenho muito grande em comparação com a implementação anterior, onde as instruções em *bytecode* eram interpretadas uma por vez. O compilador JIT pode ser projetado de acordo com a plataforma ou *hardware* de destino, e o código que ele gera pode ser otimizado com base na observação de padrões de comportamento dos programas.

A plataforma Java não é a primeira plataforma baseada em uma máquina virtual, mas é de longe a mais conhecida e a que alcançou maior sucesso. Anteriormente esta tecnologia era utilizada na criação de emuladores para auxílio ao projeto de *hardware* ou de sistemas operacionais. A plataforma Java foi desenhada para ser implementada inteiramente em *software*, enquanto permitindo a sua migração de maneira fácil para plataformas de *hardware* de todos os tipos.

Na maioria dos sistemas operacionais modernos, um corpo formado por código reusável é organizado e disponibilizado para simplificar o trabalho do programador. Este código encontra-se, normalmente, na forma de bibliotecas dinâmicas que a aplicação utiliza durante a sua execução. Como a plataforma Java não é dependente de qualquer sistema operacional, as aplicações não podem depender das bibliotecas destes sistemas. Ao contrário, a plataforma Java disponibiliza um grande conjunto padronizado de bibliotecas de classe, que contém praticamente o mesmo número de funções encontradas nos sistemas operacionais modernos. Em Java, as bibliotecas são chamadas de APIS - *Application Programming Interface* (ou Interface de Programação de Aplicativos).

As APIS servem a três propósitos dentro da plataforma Java. Como outras bibliotecas padrão, elas disponibilizam ao programador um conjunto de funções bem conhecidas que realizam tarefas comuns, como a manutenção de listas de elementos ou manipulação de *strings*. Em adição, a biblioteca contém uma interface para tarefas que dependem do *hardware* e do sistema operacional. Tarefas como acesso a rede e a arquivos são altamente dependentes

das capacidades nativas do ambiente. As APIS `java.net` e `java.io` implementam o código necessário internamente, e disponibilizam uma interface padrão para que as aplicações Java possam executar estas tarefas. Finalmente, se alguma plataforma não suportar alguma função que uma aplicação Java necessita, as APIS implementam esta funcionalidade usando os recursos disponíveis, ou disponibilizam um meio consistente para que a aplicação verifique a presença de determinada funcionalidade.

A plataforma Java é constituída de um grande número de tecnologias, cada uma provê uma porção distinta de todo o ambiente de desenvolvimento e execução de *software*. Os servidores finais, tipicamente, interagem com a máquina virtual Java (Java Virtual Machine, ou JVM) e um conjunto padrão de bibliotecas de classe. Existe um grande número de maneiras de se utilizar uma aplicação Java, incluíndo applets embutidas em páginas web, aplicativos de uso geral em *desktops*, aplicativos em aparelhos celulares e em servidores de aplicações para Internet.

As três plataformas principais que foram criadas para segmentos específicos de aplicações Java são:

- Java SE (Java Platform, Standard Edition). É a base da plataforma; inclui o ambiente de execução e as bibliotecas comuns.

- Java EE (Java Plataform, Enterprise Edition). A edição voltada para o desenvolvimento de aplicações corporativas.

- Java ME (Java Platform, Micro Edition). A edição para o desenvolvimento de aplicações para dispositivos móveis e embarcados.

Além disso, pode-se destacar outras duas plataformas Java mais específicas:

- Java Card: voltada para dispositivos embarcados com limitações de processamento e armazenamento, como smart cards e o Java Ring.

- JavaFX: plataforma para desenvolvimento de aplicações multimídia em desktop/web (JavaFX Script) e dispositivos móveis (JavaFX Mobile).

O nosso sistema irá comtemplar as plataformas Java EE na sua versão 5 (também chamado de J2EE 5) e J2SE na versão 1.6.

4.2 Java EE

O *Java EE* (Java 2 Enterprise Edition) ou J2EE é uma plataforma de programação de computadores que faz parte da plataforma Java. Ela é voltada para aplicações multi-camadas, baseadas em componentes que são executados em um servidor de aplicações.

A plataforma J2EE surgiu com o objetivo de padronizar e simplificar a criação de aplicações empresariais. Para isso, propõe um modelo onde componentes J2EE (páginas JSP, *Servlets*, EJB's, etc.) escritos pelos servidores da plataforma, podem fazer uso de serviços providos por esta, os quais simplificam sua implementação e possibilitam maior foco no negócio [SINGH02].

Esta plataforma é considerada um padrão de desenvolvimento já que o fornecedor de *software* nesta plataforma deve seguir determinadas regras se quiser declarar os seus produtos como compatíveis com *Java EE*. Ela contém bibliotecas desenvolvidas para o acesso a base de dados, RPC, CORBA, etc.

Os desenvolvedores reconhecem cada vez mais a necessidade para as aplicações distribuídas, transacional, e portáteis que influenciam a velocidade, a segurança, e a confiabilidade da tecnologia do lado do servidor. No mundo da tecnologia de informação, as aplicações da empresa devem ser projetadas, construídas, e produzidas para menor dispêndio financeiro, com velocidade maior, e com poucos recursos. E esse é o alvo da plataforma de *Java2EE*. Fornecer aos desenvolvedores um conjunto poderoso de *APIs* com o intuito de reduzir o tempo de desenvolvimento, reduzir a complexidade da aplicação, e melhorar o desempenho da aplicação.

Os principais serviços disponibilizados pela plataforma J2EE destinam-se a suprir as necessidades de aplicações empresariais distribuídas, isto é, aquelas que necessitam da

flexibilidade de disponibilizar acesso à sua lógica de negócio e dados para diferentes tipos de dispositivos clientes (navegadores, dispositivos móveis, aplicações desktop, etc.) e/ou para outras aplicações residentes na mesma empresa ou fora desta.

As aplicações distribuídas são comumente compostas de uma camada cliente, que implementam a interface com o servidor, uma ou mais camadas intermediárias, que processam a lógica do negócio e provêem serviços à camada cliente, e outra, chamada de *Enterprise Information System* ou *EIS*, formada por sistemas legados e bancos de dados. A infra-estrutura oferecida pela J2EE possibilita que estas camadas, possivelmente localizadas em máquinas diferentes, possam se comunicar remotamente e juntas comporem uma aplicação. Um componente criado numa aplicação J2EE deve ser instalado no container apropriado. Um container é um ambiente de execução padronizado que provê serviços específicos a um componente. Assim, um componente pode esperar que em qualquer plataforma J2EE implementada por qualquer fornecedor estes serviços estejam disponíveis.

Um container *web* destina-se a processar componentes *web* como Servlets, JSP's, HTML's e Java Beans. Estes são suficientes para criar aplicações completas que não necessitam ser acessadas por diferentes tipos de cliente nem tampouco tornar seus dados e lógica distribuídos. Já um container EJB destina-se a prover a infra-estrutura necessária para a execução de componentes de negócio distribuídos. Um EJB (*Enterprise Java Bean*) é um componente de software que estende as funcionalidades de um servidor permitindo encapsular lógica de negócio e dados específicos de uma aplicação. Tal componente pode ser

acessado de maneiras diferentes, por exemplo, através de RMI, CORBA ou SOAP, o que possibilita que este seja utilizado por qualquer tecnologia que provê suporte a um destes padrões de comunicação e que seja localizado virtualmente a partir de qualquer rede TCP/IP.

A figura 4 ilustra um ambiente típico de J2EE.

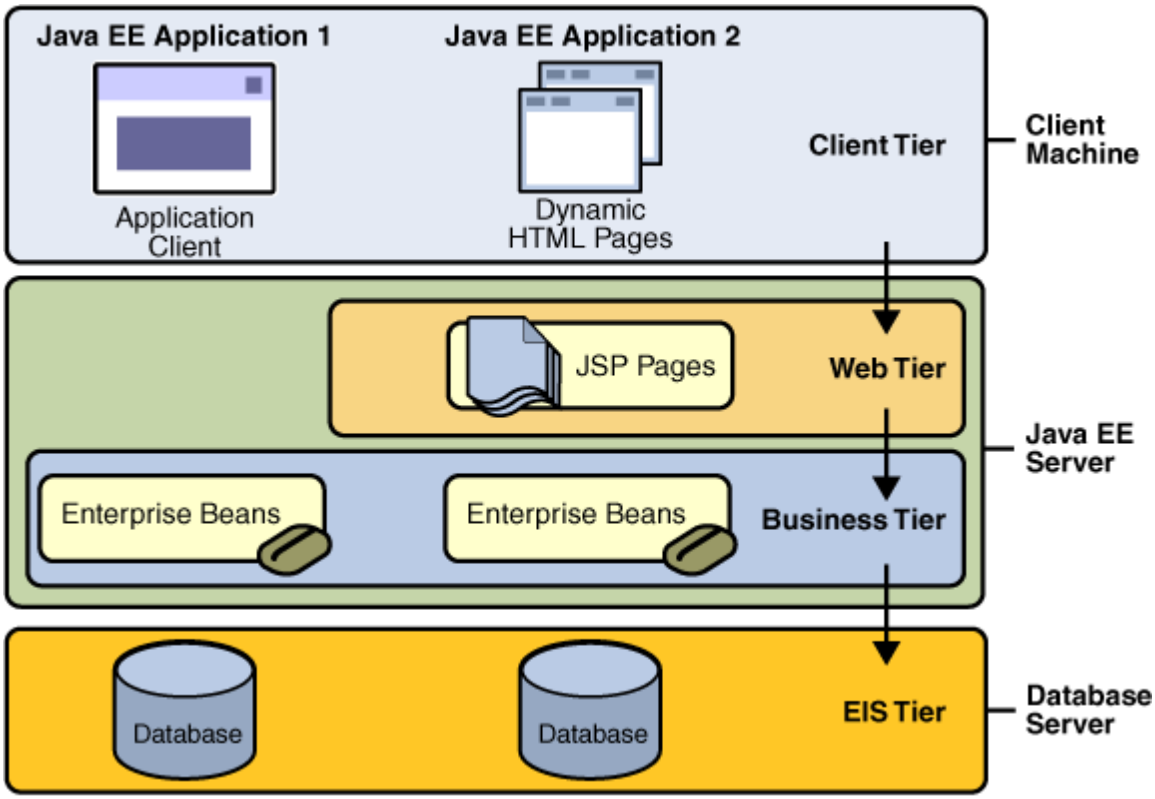


Figura 4 - aplicações de Multicamada

Fonte: Tutorial Java EE 5

A plataforma J2EE permite uma arquitetura flexível sendo que tanto o container *web* quanto o container EJB são opcionais. Como nosso trabalho tem como foco implementar um módulo de planejamento e acompanhamento da frota de prefeituras para Web, não estudaremos *EJB*. Assim, apresentamos padrões e estratégias de projeto que consideramos necessários à construção de aplicações J2EE profissionais não distribuídas. Como apresentado no livro [J2EETIP], uma aplicação web em três camadas é na maioria das vezes a escolha inicial para uma aplicação J2EE. Nesta configuração, o container web encarrega-se de tratar tanto a lógica de apresentação como a de negócios e organizaremos estas responsabilidades de forma a criarmos aplicações modulares, organizadas, com reaproveitamento de código e mais manuteníveis e extensíveis, ou seja, como criar aplicações profissionais neste cenário.

- Os componentes da camada cliente funcionam na máquina do cliente;

- Os componentes da camada Web funcionam no servidor de J2EE;

- Os componentes da camada de negócio funcionam no servidor de J2EE.

- O software da camada de sistema de informação da empresa (EIS) - funciona no servidor de EIS (servidor de banco de dados);

Na figura 5 apresentamos o cenário proposto na nossa aplicação:

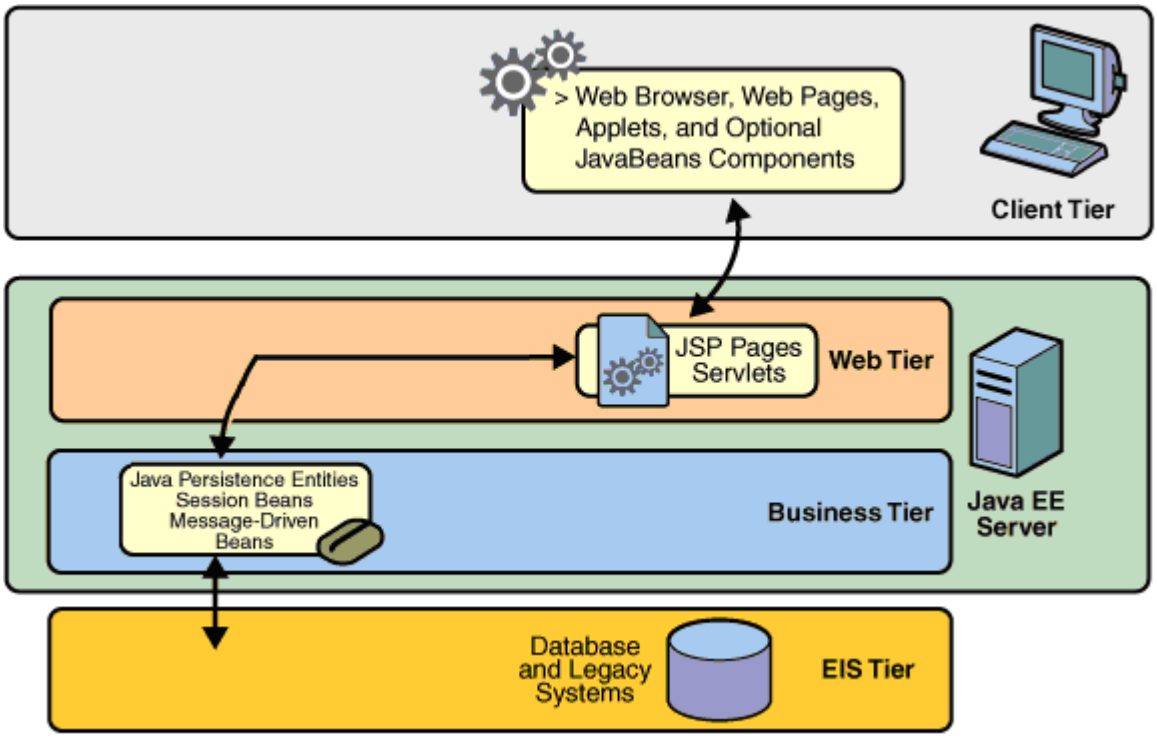


Figura 5 - Cenário de aplicação J2EE proposto.

Fonte : Tutorial *JavaEE5*

Atualmente, a plataforma J2EE está na sua 5ª versão, sendo assim também denominada de *JAVA EE 5* ou *Java EE*. Mas para fins de convenção usaremos a nomenclatura J2EE. Com a atualização, a edição da empresa (*Java EE*), o desenvolvimento de aplicações para empresa em Java nunca foi tão fácil ou tão rápido. A plataforma de *J2EE* introduz um modelo de programação simplificado. Com a tecnologia de J2EE, descrições em XML são agora opcionais. Em vez disso, um desenvolvedor pode simplesmente incorporar a informação como uma anotação diretamente em uma linha de código fonte Java, e o servidor de *Java EE* configurará o componente na distribuição e no *runtime (*em tempo de execução*)*.

Estas anotações são usadas geralmente mapear dados do programa ao invés de usar descrições em XML. Com anotações, a informação da especificação é posta diretamente em seu código fonte ao lado do elemento de que é mapeado.

Na plataforma de *Java EE*, a injeção da dependência pode ser aplicada a todos os recursos que um componente necessita, eficazmente escondendo a criação e o *lookup* dos recursos do código da aplicação. A injeção da dependência pode ser usada em contêineres *EJB*, em contêineres *Web Services*, e em clientes da aplicação. A injeção da dependência permite que o contêiner de *Java EE* introduza automaticamente referências a outros componentes ou recursos requeridos usando anotações.

Outro detalhe é quando falamos de componentes. As aplicações de J2EE são compostas de componentes. Um componente de J2EE é uma unidade funcional independente de software que são montadas em uma aplicação de J2EE com suas classes e arquivos relacionados e que se comunica com outros componentes. A especificação de J2EE define os seguintes componentes:

- Aplicações clientes e os *applets* são componentes que rodam no cliente (não aplicaremos *applets* em nosso sistema por necessitar de *plugins* no cliente);

- Java Servlets, JavaServer Faces e JavaServer Pages™ (JSP™);

- Os componentes Enterprise JavaBeans™ (EJB™), Session Beans, Java Persistence Beans (enterprise beans) são os componentes de negócio que rodam no servidor. Não aplicaremos EJB conforme descrito anteriormente.

Os componentes *web* de J2EE são *servlets* ou páginas criadas usando a tecnologia JSP (páginas de JSP) e/ou Java Server Faces. *Servlets* são classes Java que geram conteúdo dinâmico e interagem com os clientes através do modelo *request/response*. As páginas *JSP* permitem ao desenvolvedor de páginas para Internet produzir aplicações que, acessam o banco de dados, manipulam arquivos no formato texto, capturam informações a partir de formulários e captam informações sobre o visitante e sobre o servidor.

Uma página criada com a tecnologia JSP, depois de instalada em um container *web* compatível com a tecnologia J2EE, é transformada em um Servlet. A tecnologia *Java Server Faces* é construída sobre *servlets* ou páginas JSP e é um *framework* MVC para o desenvolvimento de aplicações Web.

Os *applets*, as páginas estáticas, como as páginas HTML, não são consideradas componentes *web* de J2EE.

Os componentes de J2EE são escritos em Java e compilados da mesma maneira que qualquer outro programa da linguagem Java. A diferença entre componentes de J2EE e classes “padrão” (pojo) de Java é que os componentes de J2EE estão montados em uma

aplicação de J2EE, verificados se estão em conformidade com a especificação de J2EE, e são *deployed* para produção, onde são controlados pelo servidor de J2EE.

O processo da distribuição instala componentes Java EE nos containers como ilustrado na figura 8.

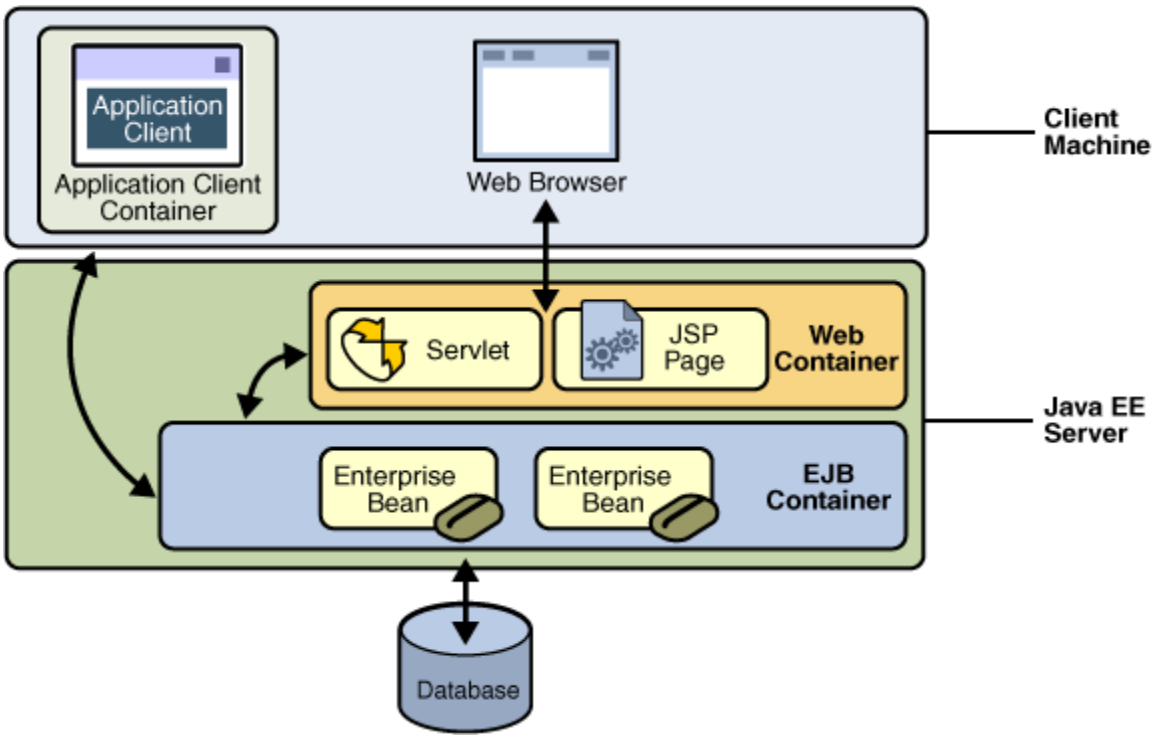


Figura 6 - servidor e recipientes de Java EE 5

Fonte: Tutorial Java EE 5

:

- **Servidor Java EE (Java EE Server)**: Executa os componentes de negócio de Java EE. Um servidor de Java EE é composto container *web* e /ou container EJB.

- **Container EJB**: Controla a execução de *enterprise beans* para aplicações de Java EE.

- **Container *Web***: Controla a execução de páginas JSP e Servlet's para aplicações Java EE. Os componentes *web* e seu container funcionam no servidor de Java EE.

- **Container da aplicação cliente**: Controla a execução de componentes da aplicação do cliente. No nosso sistema é o *web browser*.

- **Container de *applet***: Controla a execução dos applet. Consiste em um *browser web* e Java *Plugin* no lado do cliente. Não entra no contexto da nossa aplicação.

Um cliente de Java EE pode ser *web* ou um cliente da aplicação *desktop*. No nosso caso iremos apenas trabalhar com a interface *web*.

Um cliente *web* consiste de duas partes:

- Páginas dinâmicas que contêm vários tipos de linguagem de *markup* (HTML, XML, e outros), que são geradas pelos componentes *web* que rodam na camada *web;*

- Um *browser web*, que renderiza as páginas que recebe do servidor.

Algumas vezes, um cliente *web* é chamado de *thin client*. Os *thin clients* geralmente não fazem *querys* na base de dados, não executam regras de negócio complexas, ou não se conectam a sistemas legados. Quando você usa um *thin client*, tais operações são executadas no servidor J2EE, onde podem prover a segurança, a velocidade, os serviços, e a confiabilidade das tecnologias do servidor J2EE.

A figura 7 mostra os vários elementos que podem compor a camada do cliente. O cliente comunica-se com a camada do negócio que funciona no servidor de Java EE diretamente ou, na nossa aplicação onde o cliente funciona em um *browser*, atravessando as páginas JSP ou os Servlets que funcionam na camada *web*.

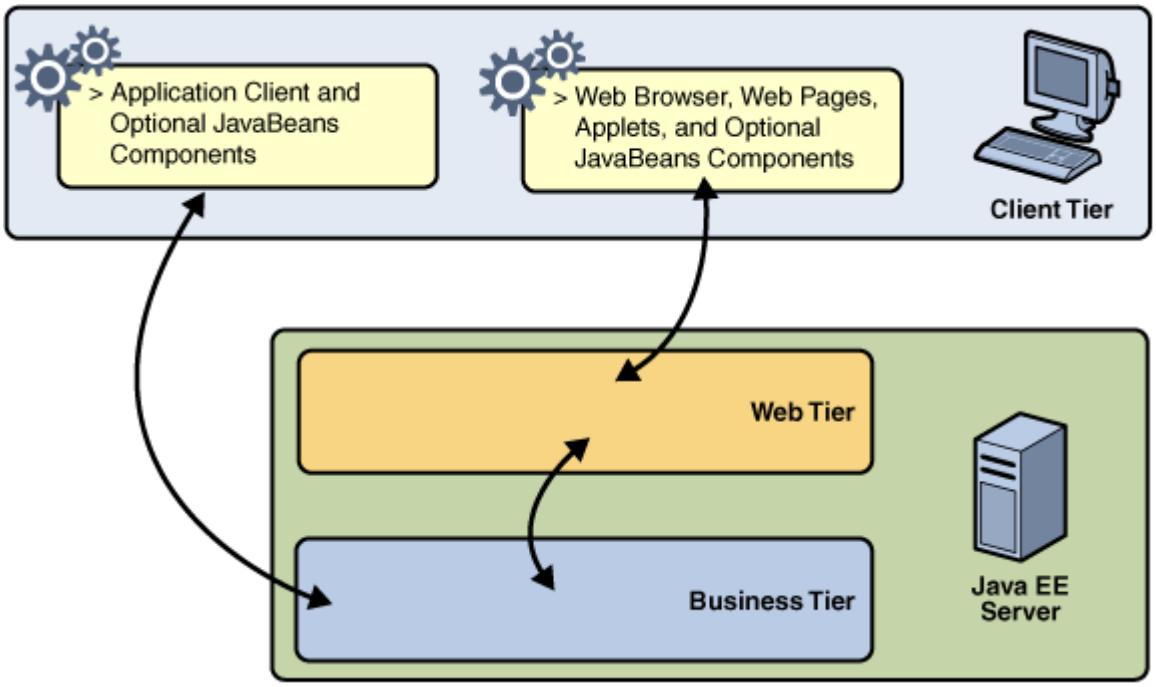


Figura 6: comunicação do servidor Java EE

Fonte: Tutorial Java EE 5

E nessa comunicação temos de ter preocupação com a segurança. E J2EE tem como destaque a segurança. Enquanto outros modelos de aplicações empresariais requerem plataforma específica de segurança em cada aplicação, o ambiente da segurança de J2EE permite restrições de segurança a serem definidos na hora do *deploy*. A plataforma de J2EE torna aplicações portáveis a uma grande variedade de implementações de segurança protegendo os desenvolvedores da aplicação da complexidade destas características de segurança.

A plataforma de *J2EE* fornece regras padrão de controle de acesso que são definidas pelo desenvolvedor e interpretadas quando a aplicação é feito o *deploy* no servidor. J2EE fornece também mecanismos padrão de *login* permitindo que os desenvolvedores de aplicação não tenham que programar tais mecanismos. A mesma aplicação trabalha em uma variedade de diferentes ambientes da segurança sem mudar o código de fonte. E isso é uma das muitas facilidades existentes em J2EE, fornecidas através de suas APIS. E J2EE é composto por diversas APIs. Nessa última versão foram adicionados novos APIS.

Na figura 6 apresentamos um mapa das APIS utilizadas em J2EE e seus respectivos *containers.*

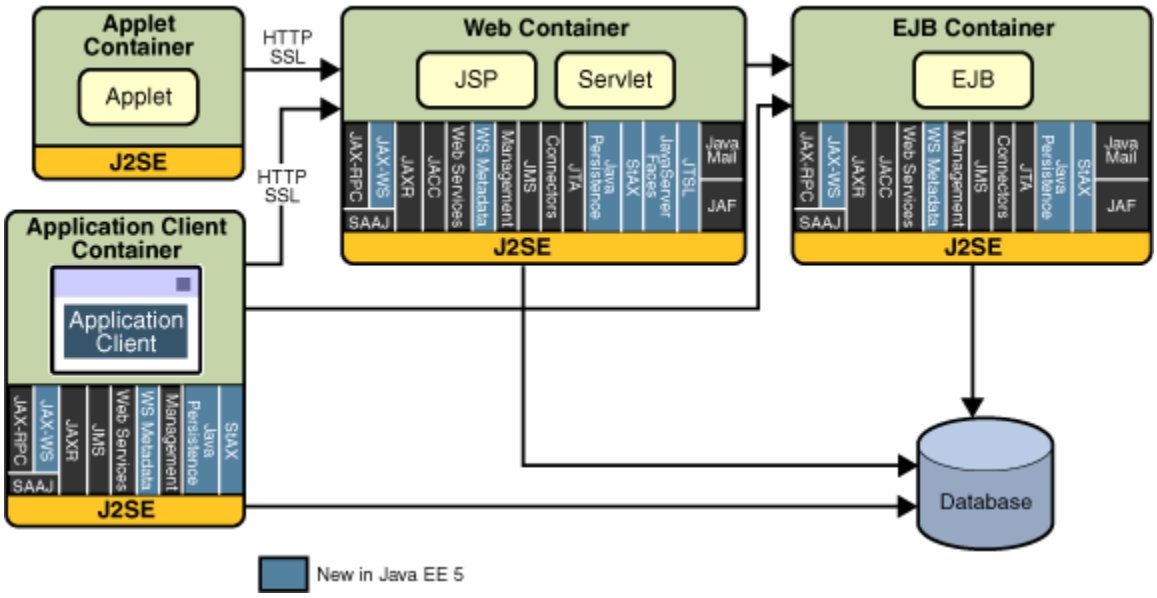


Figura 7 - Java EE 5 APIS

Fonte: Tutorial *Java EE 5*

Entre várias APIS vamos apresentar a novíssima JPA (*Java Persistence API*). JPA é a especificação padrão para o gerenciamento de persistência e mapeamento objeto relacional, surgida na plataforma J2EE na especificação do EJB 3.0 (*Enterprise Java Beans* 3.0). Introduzida no intuito de substituir os *Entity Beans* (que foram descontinuados) e simplificar o desenvolvimento de aplicações J2EE e Java SE que utilizam persistência de dados. JPA possui uma completa especificação para realizar mapeamento objeto relacional, utilizando anotações da linguagem Java (Java SE 5.0 ou superior). Também dá suporte a uma rica linguagem de consulta, semelhante à SQL, permitindo consultas estáticas ou dinâmicas.

Um dos principais conceitos relacionados à API JPA é o de entidade. Uma entidade corresponde a um objeto que pode ser gravado na base de dados a partir de um mecanismo de

persistência. A classe que implementa a entidade persistente é um POJO, que basicamente contém um construtor padrão sem argumentos e uma chave primária e também não precisa derivar de nenhuma interface ou classe, nem implementar qualquer método em especial.

Quando se decide utilizar o JPA, é necessária a escolha de um provedor JPA. Como esta API é uma especificação para *frameworks* de persistência, existe a necessidade de provedores JPA. Por padrão, a implementação de referência é o *Oracle Toplink Essentials*. Também existem outros provedores JPA no mercado, como o *Hibernate Entity Manager*, *Bea Kodo* e o *ApacheJPA*.

Como a JPA trabalha?

Nas diversas aplicações existentes, sempre que for necessário propagar o estado de um objeto que está em memória para o banco de dados ou vice-versa, há a necessidade de que a aplicação interaja com uma camada de persistência. Com a API JPA, isso é feito invocando o gerenciador de persistência, também conhecido como gerenciador de entidades (*EntityManager*), responsável por quase todas as operações de persistência de objetos.

Outro conceito também relacionado a esta especificação é o de contexto persistente (*PersistenceContext*), que é uma área de memória onde os objetos persistentes se encontram. Quando se interage com o mecanismo de persistência, é necessário para a aplicação ter conhecimento sobre os estados do ciclo de vida da persistência.

Em aplicações orientadas a objetos, a persistência permite que um objeto continue a existir mesmo após a destruição do processo que o criou. Na verdade, o que continua a existir é seu estado, já que pode ser armazenado em disco e então, no futuro, ser recriado em um novo objeto. O clico de vida de uma entidade pode ser constituído por quatro estados: *new*, *managed*, *removed* e *detached*, como mostrado na Figura 9.

Um objeto que se encontra no estado *new* pode ser definido como aquele que foi definido a partir do operador **new** e que não possui um valor para o seu identificador persistente, por exemplo, aquele com um ciclo de vida limitado ao tempo de vida do processo que o instanciou. Objetos neste estado também são conhecidos como objetos transientes. Em outras palavras, o objeto não está associado ao *EntityManager*.

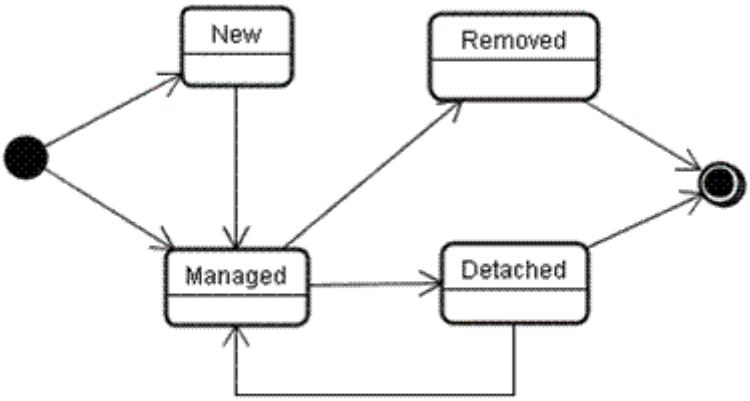


Figura 7 - Ciclo de Vida de Objetos Persistentes

Fonte: Revista Java Magazine Ed. 44 pág. 28

Já um objeto no estado *managed* é aquele que tem um valor definido para o seu identificador persistente e está associado a um contexto de persistência. Assim, o *EntityManager* irá fazer com que a base de dados seja atualizada com novos valores dos atributos persistentes no fim da transação.

Por outro lado, existem objetos também associados a um contexto de persistência, porém que estão "marcados" para serem excluídos da base de dados no final da transação, e assim permanecendo no estado *removed*.

Por fim, também existem aqueles objetos que se encontram em um estado denominado *detached*, que possuem registros correspondentes na base de dados, mas não estão associados a um contexto de persistência.

4.3 Banco de Dados Relacional

Iremos utilizar o banco de dados *PostgreSQL*, por ser robusto, fácil de instalar e ser livre. Mas antes de discorrermos sobre o PostgreSQL vamos estudar um pouco de história.

Em 1986, o professor Michael Stonebraker liderou o projeto Postgres contando com patrocínio da DARPA (Defense Advanced Research Projects Agency), ARO (Army Research Office) e NSF (National Science Foundation). Até 1993, o Postgres estava com

desenvolvimento na tutela da Universidade de Berkeley, sendo que em 1994 Andrew Yu e Jolly Chen acrescentam o dialeto SQL ao Postgres, rebatizando essa versão de Postgres95.

Em 1996, já com o seu desenvolvimento nas mãos de voluntários no mundo, o Postgres95 muda o nome para PostgreSQL para refletir o relacionamento com o Postgres original mais o suporte ao SQL, também retomam a numeração original abandonando o 95. Em 1996 é lançada a versão 6 do PostgreSQL. Em 2006 é lançada a versão 8.2 que nós utilizaremos no nosso software.

As principais características do PostgreSQL 8.2 são: suporte a Stored Procedures, Triggers, possibilidade de configuração de cursores, possui tipo de dados para moeda ("money"), uso de domains, possui recuperação no tempo (point-in-time), views atualizáveis, transações incluindo *savepoints*, portável (roda em *Linux, Unix (AIX, BSD, HP-UX, SGI, IRIX, Mac OS X, Solaris, Tru64*) e *Windows*), possui plataforma de administração e desenvolvimento (*pgAdmin*). Além disso, um recurso importante e de extrema necessidade é o tipo de bloqueio adotado. O bloqueio serve para impedir que duas transações simultâneas acessem o mesmo recurso ou dado ao mesmo tempo. O *PostgreSQL* utiliza o "*optimistic locking*". Resumidamente, permite que dados possam ler do banco e escrever sem utilizar explicitamente um bloqueio. Isto permite isolar a leitura da escrita. Caso o "*optimistic locking*" não funcione adequadamente, pode-se utilizar explicitamente o bloqueio de tabela ou linha.

5 CONCLUSÕES

O desenvolvimento de sistemas web é uma realidade do mercado atual. Com a adoção de tecnologias *WEB* 2.0, o ambiente "online" se tornou mais dinâmico e interativo. O foco do mercado de software é serviços e não software empacotado. Os sistemas tem de ter escabilidade de custo eficiente. As fontes de dados são únicas e que ficam mais ricas quanto mais pessoas a utilizarem. As interfaces de usuário se tornam mais leves. Tudo isso fazendo com que aplicações *desktop* também migrem para a *WEB*. É a evolução natural da internet. E todo esse conceito vem de encontro com o que nos propusemos a fazer: Um sistema de fácil uso, com baixo custo e simplicidade de manutenção, escalável e flexível, o Módulo de Planejamento e Acompanhamento de Frota.

Mas algumas dificuldades foram encontradas. Por se tratar de um módulo que faz parte de todo um projeto de sistemas para as prefeituras, sentimos a necessidade de alguns requisitos que já deveriam estar implementados. Com o intuito de proporcionar uma homogeneidade entre todos os módulos, sentimos a necessidade de uma política de controle e segurança única para todo o sistema. Com regras de acesso e controle permitiria um desenvolvimento mais conciso entre os módulos do projeto. Existe a necessidade também de uma interface padrão com recursos de navegação *WEB* 2.0, já implementadas, permitindo um desenvolvimento uniforme entre os módulos. E como solução, indicaríamos a necessidade de criação de um módulo de Recursos Humanos (RH), no qual constasse o cadastro de grupos de

segurança e seus papéis, grupos e cadastro de usuário com papéis pré-definidos, mas flexível que permitam alterações e atualizações. A premissa se basearia que qualquer usuário que queira acessar qualquer módulo deve estar cadastrado no RH. Com referência a interface padrão, existe a necessidade da contratação de um profissional de webdesigner, que em conjunto com os administradores do projeto Via Digital, definissem interfaces para diversas possibilidades de uso.

Além disso, prevendo necessidades futuras, é importante a contratação de profissionais que façam os mais diversos testes antes das disponibilização do sistema e seus módulos aos clientes finais. Analisando esses empecilhos aqui relatados, verificamos que são poucos, e de fácil solução.

Finalmente, gostaria de agradecer ao Professor José Eduardo Delucca, pela oportunidade gerada. Oportunidade de aprender, desenvolver e crescer junto com o sistema. Conceitos foram aprimorados. Novas tecnologias foram apreendidas. Foi de muita valia a implementação do módulo de Planejamento e Controle de Frotas para projeto Via Digital visando atender a demanda das prefeituras brasileiras.

6 BIBLIOGRAFIA

BERNESS-LEE, T. The World-Wide Web, Communications of the ACM, New York, v. 37, n.8, Agosto 1994.

PRESSMAN, Roger S. Engenharia de Software. 5. ed. Rio de Janeiro: McGraw-Hill, 2002. 843p.

[SINGH02] Singh, I.; Stearns, M. J., Enterprise Team. Designing Enterprise Applications with the J2EE Platform, Second Edition, Addison Wesley, 2002.

ROCHA, José Antonio Meira da. **Modelo de Ficha bibliográfica e de leitura.** Documento digital em formato PDF disponível em <http://www.meiradarocha.jor.br/uploads/1021/193/fichas-bibliograficas-e-de-leitura.pdf>. Acesso em: 17 Maio 2007.

DEITEL, H. M.; DEITEL, P. J.; **Java Como Programar**; tradução Carlos Arthur Lang Lisbôa. - 4ª Edição – Porto Alegre : Bookman, 2003 – Reimpressão 2004.

Humberto Rossetti Baptista, Software Livre - Definição e Direções, disponível em http://www.ime.usp.br/~cgmac/bccnews/softwarelivre.html Acesso em: 25 de Maio de 2007.

GeNESS - UFSC. Via Digital. Disponível em: < http://viadigital.org.br/>. Acesso em: 15 fev. 2007.

NETBEANS – Disponível em: < http://www.netbeans.org/>. Acesso em 27 maio 2007.

[PTJAVA2007] Plataforma Java - Disponível em: < http://pt.wikipedia.org/wiki/Plataforma_Java >. Acesso em 27 maio 2007.

Rick, C.; Inscore, J.; Enterprise Partners. J2EE Technology in Practice: Building Business Applications with the Java 2 Platform, Enterprise Edition, Sun Microsystems Press, 2001.

J2EE - Disponível em: <http://pt.wikipedia.org/wiki/J2EE >. Acesso em 27 maio 2007.

Tutorial JavaEE5 - Disponível em: <http://java.sun.com/javaee/5/docs/tutorial/doc/>. Acesso em 27 maio 2007.

Ball, J.; Carson, D. B.; Evans, I; Haase, K; Jendrock, E; The Java™ EE 5Tutorial.; February 18, 2006.; Sun; www.java.sun.com; Acesso em 30/04/2006 às 01:05hs.

Revista Java Magazine – Edição 42, 44, 45, 46 – Ano V.

Revista SQL Magazine – Edição 41, Ano 4.

Revista Mundo Java – Edição 21, 22, Ano IV.

Revista Mundo Java – Edição 16, Ano III.

Silva, Marcos Alberto Lopes da. Novos recursos para desenvolvimento na plataforma Java EE 5 com a especificação EJB 3.0. Disponível em: <http://www.linhadecodigo.com.br/artigos.asp?id_ac=1085&pag=4>. Acesso em 27 maio 2007.

Software Livre – Mudando para Melhor Disponível em: <http://www.softwarelivre.gov.br/documentos/cartilhaempdf> Acesso em 25 junho 2007.

CTIMA - <http://ctima.itajai.sc.gov.br/ctima.php> Acesso em 2 de julho 2007.

CCA - <http://www.ccanet.com.br/> Acesso em 1 de julho 2007.

Cartilha Amarela do Governo Federal – Software Livre Mudando para Melhor.

Wikipédia - <http://pt.wikipedia.org/wiki/MVC/> Acesso em 8 de agosto de 2007.

7 ANEXO I – ANÁLISE DE REQUISITOS – SWFACTORY COM MUDANÇAS REALIZADAS

SISTEMA DE PLANEJAMENTO E ACOMPANHAMENTO DE FROTA

Cliente: ***Prefeitura Municipal de Amparo (SP)***

SIPAF – Sistema de Planejamento e Acompanhamento de Frota

DOCUMENTO DE REQUISITOS

Versão 4.1

SWFactory Consultoria e Sistemas Ltda
Campus Histórico da UFLA CEP: 37200-000
Lavras – MG Tel: 35-3822-8148
http://www.swfactory.com.br

Revisões do Documento

Revisões são melhoramentos na estrutura do documento e também no seu conteúdo. O objetivo primário desta tabela é a fácil identificação da versão do documento. Toda modificação no documento deve constar nesta tabela.

Data	Versão	Descrição	Autor
03/04/2006	1.0	Criação do Documento	Roberto Damiani
09/05/2006	1.1	Considerações da Prefeitura de Amparo: Adição do Relatório: Relatório da Quantidade de Veículos por Tipos de Veículos.	Rafael Vargas
15/05/2006	2.0	Preenchimento da seção 3	Roberto Damiani
18/05/2006	2.1	Considerações da Prefeitura de Amparo: • Glauber ficou de enviar novos relatórios necessários • Ficou pendente como seria computado as horas trabalhas dos tratores, mas decidimos então resolver isso na próxima reunião.	Rafael Vargas
23/05/2006	3.0	Fechamento do documento após terceira reunião pelo skype	Roberto Damiani
22/06/2006	4.0	Não houveram alterações da versao 3.0 para a 4.0	Rafael Vargas
23/07/2007	4.1	Alterações nos requisitos: • Não há necessidade de se abastecer a cada retorno; • Manutenção recebeu kmprogramados, km (a kilometragem que o carro vai estar para fazer a manutenção), horasPrevistas e horas(quantas horas vai estar o trator para a manutenção); • Em integrante deverá conter km e/ou horas iniciais e rodados. Inserido atributos: kmInicial e horaInicial no inserir integrante, KmAtual e horaFinal na alteração e visualização do integrante;	Alecindro

		• Em integrante inserido atributo disponível para indicar se o integrante está disponível ou não para realizar atividade; • Em integrante inserido atributo ativointegrante para saber se o integrante está está em Atividade, ou seja, não foi depreciado; • Em categoria habilitação inserido descrição; • Nota Fiscal de Gasto: inserido data de preenchimento; • Inserido telefone, ramal e celular do motorista/operador; • Fornecedor – inserido CEP, email, homepage; • Motorista- inserido atributo disponível para indicar se o integrante está disponível ou não para realizar atividade.	

Auditorias do Documento

Auditorias são inspeções conduzidas pela equipe de PPQA – Product Process Quality Assurance (Garantia da qualidade do produto e processo) – do projeto, e tem por objetivo garantir uma qualidade mínima dos artefatos gerados durante o processo de desenvolvimento. Essa tabela pode ser utilizada também pelo GN – Gerente da Área de Negócio com o objetivo de documentar a viabilidade do mesmo.

Data	Versão	Descrição	Autor
23/07/2007	4.1	Alterações – Integrante	Alecindro
23/07/2007	4.1	Eliminação – Associar Item a NFGasto	Alecindro

Tipo de Licença do Documento

Este trabalho está licenciado sob uma Licença Creative Commons Atribuição-Compartilhamento pela mesma licença 2.5 Brazil. Para ver uma cópia desta licença, visite http://creativecommons.org/licenses/by-sa/2.5/br/ ou envie uma carta para Creative Commons, 559, Nathan Abbott Way, Stanford, California 94305, USA.

7.1 Introdução

Este documento especifica os requisitos do Sistema de Planejamento e Acompanhamento de Frota, fornecendo aos desenvolvedores as informações necessárias para a execução de seu projeto e implementação, assim como para a realização dos testes e homologação.

Esta introdução fornece as informações necessárias para fazer um bom uso deste documento, explicitando seus objetivos e as convenções que foram adotadas no texto. As demais seções apresentam a especificação do Sistema de Planejamento e Acompanhamento de Frota e estão organizadas como descrito abaixo:

- **Seção 2 - Descrição geral do produto/serviço**: apresenta uma visão geral do produto/serviço, caracterizando qual é o seu escopo e descrevendo seus usuários.
- **Seção 3 - Requisitos funcionais:** lista e descreve os requisitos funcionais do produto/serviço, especificando seus objetivos, funcionalidades, atores e prioridades.
- **Seção 4 - Requisitos não funcionais**: especifica todos os requisitos não funcionais do produto/serviço, divididos em requisitos de usabilidade, confiabilidade, desempenho, segurança, distribuição, adequação a padrões e requisitos de hardware e software.
- **Seção 5 - Rastreabilidade:** apresenta os relacionamentos entre os requisitos do produto/serviço.
- **Seção 6 - Referências:** contém uma lista de referências para outros documentos relacionados

7.1.1 Convenções, termos e abreviações

A correta interpretação deste documento exige o conhecimento de algumas convenções e termos específicos, que são descritos a seguir.

7.1.1.1 Identificação dos Requisitos

Por convenção, a referência a requisitos é feita através do identificador do requisito, de acordo com o esquema abaixo:

[identificador de tipo de requisito.identificador do requisito]

O identificador de tipo de requisito pode ser:

- RF – requisito funcional
- RNF – requisito não-funcional

Identificador do requisito é um número, criado seqüencialmente, que determina que aquele requisito é único para um determinado tipo de requisito.

Ex: RF001, RF002, RNF001, RNF002.

7.1.1.2 Prioridades dos Requisitos

Para estabelecer a prioridade dos requisitos foram adotadas as denominações "essencial", "importante" e "desejável".

- **Essencial** é o requisito sem o qual o sistema não entra em funcionamento. Requisitos essenciais são requisitos imprescindíveis, que têm que ser implementados impreterivelmente.
- **Importante** é o requisito sem o qual o sistema entra em funcionamento, mas de forma não satisfatória. Requisitos importantes devem ser implementados, mas, se não forem, o sistema poderá ser implantado e usado mesmo assim.
- **Desejável** é o requisito que não compromete as funcionalidades básicas do sistema, isto é, o sistema pode funcionar de forma satisfatória sem ele. Requisitos desejáveis são requisitos que podem ser deixados para versões posteriores do sistema, caso não haja tempo hábil para implementá-los na versão que está sendo especificada.

7.1.1.3 Especificação de Requisitos

As especificações dos requisitos deste documento obedecem às propriedades enumeradas a seguir:

- Não ambigüidade: Todas as especificações devem, idealmente, ter uma única interpretação.
- Completude: Descrever cada aspecto significante e relevante do sistema e deve incluir detalhes a respeito de todas as informações.
- Consistência: Não devem existir requisitos contraditórios na especificação.
- Verificabilidade: Quando o sistema for projetado e implementado, deverá ser possível verificar se o projeto de implementação satisfaz os requisitos originais.
- Validação: O usuário/cliente deve ser capaz de ler e entender a especificação de requisitos e, então, indicar se os requisitos refletem as suas idéias.
- Modificação: Alterações devem ser feitas facilmente, sem a necessidade de que tais modificações sejam realizadas em toda a especificação.
- Compreensão: Clientes, usuários, analista, projetistas e engenheiros devem ser capazes de entender os requisitos.
- Rastreamento: Devem ser feitas referencias entre os requisitos, aspectos de projetos e implementação.

7.1.1.4 Integrante

Um integrante da frota são todos os componentes da frota. Por exemplo, tratores, carros, caminhões, ônibus, etc.

7.1.1.5 Tabelas Relacionadas aos Requisitos de Relatórios

Em algumas tabelas de relatórios do sistema (como a tabela explicativa 1) existe o símbolo '>' na coluna "Nome do Campo". Esta foi a simbologia adotada para representar sub-níveis no relatório. Para o primeiro sub-nível '>'; para o segundo '>>' e assim sucessivamente.

Podem existir vários sub-níveis '>' para um determinado nível, assim como vários sub-níveis '>>' para um sub-nível '>'. As tabelas (Tabela Explicativa 1 e Tabela Explicativa 2) abaixo exemplificam o texto anteriormente descrito:

Nome do Campo	Descrição do Campo
Grupo	[RF42] – (Automóveis ou Tratores)
> Subgrupo	Subgrupo - [RF09]
>> Nome	Nome do Integrante
>> Placa	Placa do integrante (se existir)
>> Código	Código do Integrante

Tabela Explicativa 1 – Listagem de Integrantes por Grupos e Sub-Grupos

Campos apresentados no Relatório
Grupo: Automóveis
> Subgrupo: Caminhão
>> Nome: 715 C
>>> Placa: KQQ 0701
>>> Código: 1
>> Nome: 716 C
>>> Placa: KQQ 0709
>>> Código: 2
> Subgrupo: Ônibus
>> Nome: 715 T
>>> Placa: KQQ 0703
>>> Código: 3

Tabela Explicativa 2 – Forma de exibição do Relatório

7.1.2 Visão geral do Produto/serviço

O sistema aqui proposto busca prover uma solução que possibilite acompanhar a frota de veículos de uma prefeitura. Dentre outras funcionalidades o Sistema irá auxiliar o administrador da frota a planejar as manutenções, assim como acompanhar os gastos com cada veículo. Este possui controle de acesso ao sistema por usuário, permitindo que somente pessoas autorizadas possam utilizar o sistema de acordo com as permissões de acesso.

7.1.2.1 Abrangência e sistemas relacionados

O Sistema de Planejamento e Acompanhamento de Frota (SIPAF) tem como objetivo principal, auxiliar o administrador da frota a planejar todas as manutenções, assim como acompanhar todos os gastos com cada veículo da frota (automóvel, ônibus, tratores, etc.). Visto que há dois grupos bem distintos de veículos dentro da frota, iremos classificá-los em automóveis e tratores, sendo o acompanhamento dos automóveis realizado em quilômetros e dos tratores em horas.

Cada veículo possuirá um memorial descritivo, que conterá os dados referentes ao estado do veículo, como por exemplo, uma lanterna quebrada, retrovisor quebrado. Ao iniciar a atividade o motorista deverá realizar uma vistoria no veículo, para averiguar se o estado é equivalente ao memorial. Caso o memorial não seja compatível com o veículo, o responsável pela frota deverá ser comunicado.

Cada atividade de um integrante da frota deverá ser monitorada pelo responsável da frota. Este monitoramento consiste em informações relacionadas:

- Hora de saída e entrada;

- Informações sobre o Motorista;

- Quilometragem percorrida/ Horas trabalhadas;

- Ocorrência de multa(s);

- Ocorrência de acidente(s) (Classificar como baixo, médio e alto, de acordo com o Boletim de Ocorrência de Trânsito);

- Observação (Neste campo deverá ser informado qualquer incidente na atividade que possa justificar alguma anormalidade na atividade);

- Nota de abastecimento.

O sistema permitirá ao responsável pela frota agendar todas as manutenções de cada veículo da frota, desta forma o mesmo auxiliará ao responsável no processo de manutenção. Além do controle de manutenções será possível controlar todos os gastos com cada veículo da frota. Este controle será realizado através do cadastro de todos os acessórios, serviços, e qualquer outro tipo de gasto relacionado a um determinado veículo da frota.

Além do cadastro de veículos e gastos, ainda será cadastrados todos os motoristas e operadores de máquinas da frota. A partir deste cadastro será possível controlar as multas do motorista, veículos que ele utilizou, tipo e vencimento da Carteira Nacional de Trânsito.

Relatórios

- Relatório de Gastos com cada Veículo por Período
- Quilometragem percorrida com cada Veículo por Período
- Data da Próxima manutenção de cada Veículo
- Consumo de combustível de cada veículo
- Quilometragem percorrida por cada motorista (automóveis)
- Horas trabalhadas de cada motorista (tratores)
- Características de Cada Veículo
- Status de Cada Veículo (Disponível, Em manutenção, Alocado)

- Memorial Descritivo de cada Veículo.

- Previsão de produtos (Óleo, Filtros, Pneus) para manutenção.

7.1.2.2 Descrição do cliente

GeNESS (SC).

7.1.2.3 Descrição dos usuários

7.1.2.4 Usuário Administrador

Usuário capaz de executar todas as funcionalidades do sistema. Principalmente adicionar usuários no sistema e definir seus perfis.

7.1.2.5 Usuário Gestor

Usuário responsável por administrar a frota, podendo executar todas as funcionalidades relacionadas à regra de negócio.

7.1.2.6 Usuário Responsável pela Frota

Usuário responsável por acompanhar os integrantes da frota, registrando as atividades, manutenções e danificações no integrante.

7.1.3 Requisitos funcionais

7.1.3.1 Cadastros Gerais

Os requisitos funcionais agrupados nesta seção estão relacionados aos cadastros gerais do sistema, ou seja, cadastros básicos de usuários, perfis de usuários, e demais cadastros necessários para fornecer suporte para o bom funcionamento do sistema.

[RF01] Inserir Usuário do Sistema

Ator: Usuário Administrador.

Este requisito funcional começa quando o Ator deseja adicionar um novo usuário no sistema.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
* Nome	Nome do Usuário
* Perfil	Perfil do Usuário
Endereço	Endereço do Usuário
* Login	Login do Usuário

* Senha	Senha do Usuário
*CPF	CPF do Usuário

Tabela 1 - Campos para o Cadastro de Usuário

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Obs.1: Deverá existir, no momento da implantação do sistema, cadastrado um usuário de perfil Usuário Administrador, com a senha “123” e login ”admin”.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF02] Alterar Usuário do Sistema

Ator: Usuário Administrador

Este requisito funcional começa quando o Ator deseja alterar os dados de um Usuário no sistema. Primeiramente devem ser listados todos os Usuários do sistema (através do requisito [RF48]), e ao escolher um dos listados, será obtida a visualização de seus dados (através do requisito [RF04]). A partir disso, este mesmo Usuário visualizado poderá ter seus dados alterados.

Os campos da Tabela 2 não podem ser modificados, os demais campos presentes na Tabela 1 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 2 - Campos não Modificáveis

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF03] Remover Usuário do Sistema

Ator: Usuário Administrador.

Este requisito funcional começa quando o Ator deseja remover os dados de um Usuário no sistema. Primeiramente devem ser listados todos os Usuários do sistema (através do requisito [RF48]), e ao escolher um dos listados, será obtida a visualização de seus dados (através do requisito [RF04]). A partir disso, este mesmo Usuário visualizado poderá ter seus dados removidos.

Neste caso o usuário só poderá ser removido se ele não realizou nenhuma operação no sistema. Caso contrário ele deverá ser desativado.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF04] Visualizar Usuário do Sistema

Ator: Usuário Administrador, Usuário Responsável pela Frota e Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar os dados de um usuário no sistema. Primeiramente devem ser listados todos os Usuários do sistema (através do requisito [RF48]), e ao escolher um dos listados, será obtida a visualização dos seguintes dados:

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Nome	Nome do Usuário
Perfil	Perfil do Usuário
Endereço	Endereço do Usuário
Login	Login do Usuário
CPF	CPF do Usuário

Tabela 3 - Campos de visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF05] Inserir Perfil de Usuário do Sistema

Ator: Sistema e Usuário Administrador

Este requisito funcional começa quando o Ator deseja criar um novo perfil de usuário no sistema.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
* Nome	Nome do Perfil
* Funcionalidade	Funcionalidades do sistema
* Descrição	Descrição do Perfil do Usuário

Tabela 4 - Campos para Cadastro de Perfil de Usuário

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Obs.1: Durante a inserção de um novo perfil, devem ser mapeadas todas as funcionalidades que um usuário deste perfil poderá realizar.

Obs2: Entende-se por perfil, um grupo de funcionalidades que um usuário pode realizar.

Obs.3: Deverão ser criados automaticamente pelo sistema no momento da implantação os seguintes perfis:

- Administrador
- Gestor
- Responsável pela Frota

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF06] Alterar Perfil de Usuário do Sistema

Ator: Usuário Administrador

Este requisito funcional começa quando o Ator deseja alterar os dados de um Perfil de Usuário do sistema. Primeiramente devem ser listados todos os Perfis do sistema (através do requisito [RF49]), e ao escolher um dos listados, será obtida a visualização de seus dados (através do requisito 0). A partir disso, este mesmo Perfil visualizado poderá ter seus dados alterados.

Os campos da Tabela 5 não podem ser alterados, os demais campos presentes na Tabela 4 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 5 - Campos não Modificáveis

Obs.1: Não poderão ser alterados os perfis padrões do sistema (Obs3, [RF05]).

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF07] Remover Perfil de Usuário do Sistema

Ator: Usuário Administrador

Este requisito funcional começa quando o Ator deseja remover os dados de um Perfil de Usuário do sistema. Primeiramente devem ser listados todos os Perfis do sistema (através do requisito [RF49]), e ao escolher um dos listados, será obtida a visualização de seus dados (através do requisito 0). A partir disso, este mesmo Perfil visualizado poderá ter seus dados removidos.

O Perfil só poderá ser removido se nenhum usuário estiver utilizando-o.

Obs.1: Não poderão ser removidos os perfis padrões do sistema (Obs3, [RF05]).

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF08] Visualizar Perfil do Usuário do Sistema

Ator: Usuário Administrador, Usuário Gestor e Usuário Responsável pela Frota

Este requisito funcional começa quando o Ator deseja visualizar os dados de um perfil de usuário do sistema. Primeiramente devem ser listados todos os Perfis de Usuários do sistema (através do requisito [RF49]), e ao escolher um dos listados, será obtida a visualização dos seguintes dados:

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Nome	Nome do Perfil
Funcionalidade	Funcionalidades do sistema
Descrição	Descrição do Perfil do Usuário

Tabela 6 - Campos de visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.1.4 Cadastros Relacionados aos Integrantes da Frota

[RF09] Inserir Subgrupo de Integrantes

Ator: Usuário Administrador.

Este requisito funcional começa quando o ator deseja cadastrar um novo subgrupo de integrantes, sendo que este subgrupo deve ser relacionado a um dos grupos do requisito [RF42].

Deverão ser pré-cadastrados no sistema no momento da implantação os seguintes subgrupos:

AUTOMÓVEIS:

Caminhão;

Caminhonete;

Microônibus;

Ônibus;

Utilitários,

Passeio,

Motocicletas.

TRATORES:

Pá Carregadeira

Retro Escavadeira

Esteira

Motoniveladora

Rolocompactador

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Nome	Nome do Subgrupo
*Grupo	[RF42] – (Automóveis ou Tratores)

Tabela 7 - Campos para o cadastro de Subgrupo de Integrantes

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF10] Alterar Subgrupo de Integrantes

Ator: Usuário Administrador.

Este requisito funcional começa quando o ator deseja alterar um subgrupo de integrantes cadastrado anteriormente no sistema. Para isto ele deve primeiramente listar os subgrupos através do requisito [RF50], e ao escolher um dos listados será visualizado os dados através do requisito 0. A partir disso o subgrupo poderá ser alterado.

Os campos da Tabela 8 não podem ser modificados, os demais campos presentes na Tabela 7 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 8 - Campos não Modificáveis

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF11] Remover Subgrupo de Integrantes

Ator: Usuário Administrador.

Este requisito funcional começa quando o ator deseja excluir um subgrupo do sistema. Para isto ele deve primeiramente listar os subgrupos através de requisito [RF50], e ao escolher um dos listados ele será visualizado. A partir disso ele poderá remover o subgrupo, desde que não haja dependências a ele.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF12] Visualizar Subgrupo de Integrantes

Ator: Usuário Administrador.

Este requisito funcional começa quando o ator deseja visualizar um subgrupo de integrantes. Primeiramente deve ser listado os Subgrupos através do requisito [RF50], e ao escolher um dos listados ele será visualizado.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente

	pelo sistema
Nome	Nome do Subgrupo
Grupo	[RF42] – (Automóveis ou Tratores)

Tabela 9 - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF13] Inserir Integrante

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja cadastrar um novo integrante na frota.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Grupo	[RF42] – (Automóveis ou Tratores)
*Subgrupo	[RF09] – Nome do Subgrupo

*Nome	Nome do Integrante
*Placa	Placa do Integrante
Chassis	Chassi do Integrante
*Ano de Fabricação	Ano de Fabricação do Integrante
Ano do Modelo	Ano do Modelo do Integrante
*Combustível	[RF43] – Tipo de Combustível
Data da Compra	Data da compra do veículo
*Categoria de Habilitação	[RF40] – Categoria de Habilitação
Marca	Marca do Integrante
Potência do Motor	Em cilindradas
Km Inicial	Kilometragem do Integrante ao cadastrar caso automóveis/número de horas caso tratores

Km Atual	Kilometragem do Integrante caso automóveis/número de horas caso tratores - atualizada automaticamente
Ativointegrante[1]	Indica se o integrante está em Atividade, ou seja, não foi depreciado
Disponível[1]	Indica se o integrante está disponível para realizar atividade ou não

Tabela 10 - Campos para o Cadastro de Integrante

- Esses campos são considerados de preenchimento obrigatório
- [1] Ao inserir estará atuomaticamente setado como verdadeiro

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF14] Alterar Integrante

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja alterar um integrante cadastrado anteriormente no sistema. Para isto ele deve primeiramente listar os integrantes através do requisito [RF52], e ao escolher um dos listados será obtida a visualização dos dados através do requisito 0. A partir disso o integrante poderá ser alterado.

Os campos da Tabela 11 não poderão ser modificados, os demais campos presentes na Tabela 10 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 11 - Campos não Modificáveis

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF15] Remover Integrante

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja excluir um integrante do sistema. Para isto ele deve primeiramente listar os integrantes através de requisito [RF52], e ao escolher um dos listados ele será visualizado através do requisito 0. A partir disto ele poderá remover o integrante.

O integrante não será removido fisicamente do sistema, apenas desativado de todas as funcionalidades exceto das funcionalidades referente ao histórico.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF16] Visualizar Integrante

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja visualizar um integrante. Primeiramente deve ser listado os integrantes através do requisito [RF52], e ao escolher um dos listados ele poderá ser visualizado.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Grupo	[RF42] – (Automóveis ou Tratores)
Subgrupo	[RF09] – Nome do Subgrupo
Nome	Nome do Integrante
Placa	Placa do Integrante
Chassis	Chassi do Integrante
Ano de Fabricação	Ano de Fabricação do Integrante

Ano do Modelo	Ano do Modelo do Integrante
Combustível	[RF43] – Tipo de Combustível
Data da Compra	Data da compra do veículo
Categoria de Habilitação	[RF40] – Categoria de Habilitação
Marca	Marca do Integrante
Potência do Motor	Em cilindradas
Km Inicial	Kilometragem do Integrante ao cadastrar caso automóveis/número de horas caso tratores
Km Atual	Kilometragem do Integrante caso automóveis/número de horas caso tratores - atualizada automaticamente
Ativointegrante	Indica se o integrante está em Atividade, ou seja, não foi depreciado

Disponível	Indica se o integrante está disponível para realizar atividade ou não

Tabela 12 - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.1.5 Cadastros Relacionados aos Motoristas e Operadores da Frota

[RF17] Inserir Motorista/Operador

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja cadastrar um novo motorista/operador da frota.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Nome	Nome do Motorista/Operador
*CPF	Número do CPF

*Matrícula	Número de Matrícula na Prefeitura
*Categoria de Habilitação	[RF40] - Categoria de Habilitação
*Classificação	1 – Motorista; 2 - Operador de Tratores; 3 - Motorista e Operador de Tratores.
Validade da Carteira de Habilitação	Data de Vencimento da Carteira de Habilitação
Foto	Foto do Motorista (JPEG)
Telefone	Telefone do motorista
Ramal	Ramal do motorista
Celular	Celular do motorista

Tabela 13 - Campos para o Cadastro de Motorista/Operador

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF18] Alterar Motorista/Operador

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja alterar os dados de um motorista/operador cadastrado anteriormente no sistema. Para isto ele deve primeiramente listar os motoristas/operadores através do requisito [RF51], e ao escolher um dos listados será obtida a visualização dos dados através do requisito [RF20]. A partir disso o motorista/operador poderá ser alterado.

Os campos da Tabela 14 não poderão ser modificados, os demais campos presentes na Tabela 13 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 14 - Campos não Modificáveis

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF19] Remover Motorista/Operador

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja excluir um operador/motorista do sistema. Para isto ele deve primeiramente listar os motoristas/operadores através do requisito [RF51], e ao escolher um dos listados ele será visualizado. A parti disto ele poderá remover o motorista/operador.

O motorista/operador não será removido fisicamente do sistema, apenas desativado de todas as funcionalidades, exceto das funcionalidades referente ao histórico.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF20] Visualizar Motorista/Operador

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o ator deseja visualizar um motorista/operador. Primeiramente deve ser listado os motoristas/operadores através do requisito [RF51], e ao escolher um dos listados ele poderá ser visualizado.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Nome	Nome do Motorista/Operador
CPF	Número do CPF
Matrícula	Número de Matrícula na Prefeitura
Categoria de Habilitação	[RF40] - Categoria de Habilitação
Classificação	1 – Motorista; 2 - Operador de Tratores; 3 - Motorista e Operador de Tratores.
Validade da Carteira de Habilitação	Data de Vencimento da Carteira de Habilitação

Foto	Foto do Motorista (JPEG)
Telefone	Telefone do motorista
Ramal	Ramal do motorista
Celular	Celular do motorista

Tabela 15 - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.1.6 Requisitos Relacionados aos Gastos com a Frota

[RF21] Inserir Fornecedor

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja adicionar um novo fornecedor no sistema.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema

* Razão Social	Razão Social do Fornecedor
Descrição	Descrição do Fornecedor
*CNPJ	CNPJ do Fornecedor
Endereço	Endereço do Fornecedor
Telefone	Telefone do Fornecedor
Telefone-Fax	Telefone-Fax do Fornecedor
CEP	Cep do Fornecedor
Page	Homepage do Fornecedor
E-mail	E-mail do Fornecedor

Tabela 16 - Campos para o Cadastro de Fornecedor

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF22] Alterar Fornecedor

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja alterar os dados de um Fornecedor anteriormente cadastrado no sistema. Primeiramente devem ser listados os Fornecedores do sistema através do requisito [RF53], e ao escolher um dos listados, será obtida a visualização de seus dados através do requisito [RF24]. A partir disso, este mesmo Fornecedor visualizado poderá ter seus dados alterados.

Os campos da Tabela 17 não podem ser modificados, os demais campos presentes na Tabela 16 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 17 - Campos não modificáveis

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF23] Remover Fornecedor

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja remover os dados de um Fornecedor anteriormente cadastrado no sistema. Primeiramente devem ser listados os Fornecedores do sistema através do requisito [RF53], e ao escolher um dos listados, será obtida a visualização de seus dados através do requisito [RF24]. A partir disso, este mesmo Fornecedor visualizado poderá ter seus dados removidos.

Neste caso o fornecedor não é removido fisicamente do sistema, é apenas desativado de todas as funcionalidades, exceto nas funcionalidades referente a histórico.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF24] Visualizar Fornecedor

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar os dados de um Fornecedor do sistema. Primeiramente devem ser listados os Fornecedores do sistema através do requisito [RF53], e ao escolher um dos listados, será obtida a visualização dos seguintes dados:

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Razão Social	Razão Social do Fornecedor
Descrição	Descrição do Fornecedor
CNPJ	CNPJ do Fornecedor
Endereço	Endereço do Fornecedor
Telefone	Telefone do Fornecedor
Telefone-fax	Telefone-fax do Fornecedor
CEP	Cep do Fornecedor
Page	Homepage do Fornecedor
E-mail	E-mail do Fornecedor

Tabela 18 - Campos para Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF25] Inserir Nota Fiscal de Gasto

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja cadastrar uma nova nota fiscal de gasto com os integrantes da frota.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Número	Número da Nota Fiscal
datapreenchegasto	Data que cadastra esta nota Fiscal
# Data	Data de Emissão da Nota
*Valor	Valor da Nota
*Fornecedor	[RF21] – Fornecedor da Nota

> *Item	[RF44] - Todos os Itens da Nota
>> *Quantidade	Quantidade de Cada Item da Nota
>> *Valor Unitário	Valor Unitário da Cada Item da Nota
>> *Valor Total	Valor total de cada Item da Nota

Tabela 19 - Campos para o Cadastro de Nota Fiscal de Gasto

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF26] Remover Nota Fiscal de Gasto

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja excluir uma nota fiscal previamente cadastrada no sistema. Para isto ele deve primeiramente listar as notas através do requisito [RF54], posteriormente selecionar uma nota e visualizar a através do requisito [RF54]. Após a visualização o ator poderá excluir a nota visualizada. Uma nota só pode ser removida se não houver nenhum vínculo a ela no sistema

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

.

[RF27] Visualizar Nota Fiscal de Gasto

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar uma nota fiscal previamente cadastrada no sistema. Primeiramente ele deve listar as notas cadastradas no sistema através do requisito [RF54] e selecionar uma nota para ser visualizada.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Número	Número da Nota Fiscal
datapreenchegasto	Data que cadastra esta nota Fiscal
Data	Data de Emissão da Nota
Valor	Valor da Nota

Fornecedor	[RF21] – Fornecedor da Nota
> Itens	[RF44] - Todos os Itens da Nota
> Quantidade	Quantidade de Cada Item da Nota
> Valor Unitário	Valor Unitário da Cada Item da Nota
> Valor Total	Valor total de cada Item da Nota

Tabela 20 - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF28] Associar Item da Nota a Integrante da Frota – Todo Item será um Serviço

---

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja associar uma quantidade de um item de uma nota previamente cadastrada no sistema, a um integrante da frota. Por exemplo, uma nota de que contenha um item, 10 pneus. Este item poderá ser associado para 4 integrantes distintos da seguinte forma: 2 pneus para um integrante, 4 para outro e 4 para outro.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Fornecedor	[RF21] – Fornecedor da Nota
*Número	Número da Nota
> *Integrante	[RF13] – Integrante da frota
>> *Item	[RF44] – Item da Nota
>> *Quantidade	Quantidade do Item
>> *Natureza do Reparo	Deverá ser selecionada qual manutenção (Acidente/Desgaste/Danificação)-**Erro! Fonte de referência não encontrada.**, está sendo reparada por este item.
>> * Status da Manutenção	Completa ou Incompleta

Tabela 21 - Campos para a Associação de Item a Integrante da Frota

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Obs.1: Não é obrigatório que o Ator associe todos os itens de uma nota a algum integrante.

Ele poderá associar qualquer item da nota a qualquer integrante;

Obs.2: O usuário Ator poderá utilizar apenas uma nota para associar itens a mais de um integrante;

Obs.3: É obrigatório selecionar o Status da Manutenção informando se com a associação do item a Manutenção do Integrante associado será finalizada ou não. Com isso o usuário Ator poderá saber quais manutenções já foram finalizadas ou não no relatório de manutenções [RF60].

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF29] Dissociar Item da Nota de Integrante da Frota

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja desassociar uma quantidade de um item de uma nota previamente associado a um integrante. Para isto deve-se primeiramente listar as associações de uma nota através do requisito [RF56]. Após isto ele visualizará a associação de acordo com a **Erro! Fonte de referência não encontrada.**e realizará a dissociação. Sendo que o Ator poderá selecionar quais associações serão dissociadas:

Toda a associação;

Somente a associação a um integrante;

Somente alguns itens da associação a um integrante;

Somente algumas unidades da quantidade do item associado a um integrante.

Nome do Campo	Descrição do Campo
Código	Código da associação gerado no [RF28]
Fornecedor	[RF21] – Fornecedor da Nota

Número	Número da Nota
> Integrante	[RF13] - Integrante da frota
>> Item	[RF44] – Item da Nota
>> Quantidade	Quantidade do Item
>> Natureza do Reparo	Manutenção (Acidente/Desgaste/Danificação) - **Erro! Fonte de referência não encontrada.**, reparado por este item.
>> Status da Manutenção	Completa ou Incompleta

Tabela 22 - Campos de Visualização da Dissociação

Obs.1: Se a opção de dissociação utilizada pelo Ator for a de dissociar somente alguma unidade da quantidade do item associado a um integrante ele deverá preencher o número de unidades que serão dissociadas da quantidade associada

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

.

[RF30] Inserir Nota de Serviço

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja cadastrar uma nova nota de serviço para um integrante da frota

Nome do Campo	Descrição do Campo
# Código	Código gerado automaticamente pelo sistema
# Data	Data de Emissão da Nota (preenchido automaticamente pelo sistema)
*Integrante	[RF13] – Integrante da Frota
> *Item	[RF44] - Nome do Item
> *Quantidade	Quantidade do Item
> *Natureza do Reparo	Deverá ser informada qual manutenção (Acidente/Desgaste/Danificação) - **Erro! Fonte de referência não**

	encontrada., está sendo reparada por este serviço.
> Status da Manutenção	Completa ou Incompleta

Tabela 23 - Campos para o Cadastro de Nota de Serviço

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Obs1.: As manutenções reparadas por este serviço deverão ser informadas ao sistema, para que estas não constem no memorial do integrante

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável .

[RF31] Remover Nota de Serviço

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja excluir uma nota de serviço previamente cadastrado no sistema. Para isto ele deve primeiramente listar as notas de serviço através do requisito [RF58], e posteriormente selecionar uma nota e visualizar através do requisito [RF32]. A partir desta visualização o ator poderá excluir a nota. Uma nota só poderá ser removida se não houver nenhum vínculo a ela no sistema.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF32] Visualizar Nota de Serviço

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar uma nota de serviço previamente cadastrada no sistema. Primeiramente ele deve listar as notas cadastradas no sistema através do requisito [RF58]. A partir disto ele seleciona uma nota para ser visualizada.

Nome do Campo	Descrição do Campo
Código	Código gerado automaticamente pelo sistema
Data	Data de Emissão da Nota
Integrante	[RF13] – Integrante da Frota
> Item	[RF44] - Nome do item
> Quantidade	Quantidade de Cada Item da Nota
> Natureza do Reparo	Manutenção (Acidente/Desgaste/Danificação) - **Erro! Fonte de referência não encontrada.**,

	reparado por este serviço
> Status da Manutenção	Completa ou Incompleta

Tabela 24 - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.1.7 Registro de Atividades

[RF33] Iniciar Atividade

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja iniciar uma atividade de um integrante da frota.

Nome do Campo	Descrição do Campo
# Código	Código gerado automaticamente pelo sistema
# Data	Data da Atividade (Preenchida automaticamente pelo sistema)

* Integrante	[RF13] – Nome do Integrante da Frota
* Placa do Integrante	[RF13] – Placa do Integrante da Frota
*Motorista/Operador	[RF17] - Nome do motorista ou operador
* Hora Inicial	Hora do início da Atividade
Data de Término	Data Prevista de término da atividade
Hora de término	Hora Prevista de término da atividade
*Quilometragem Inicial	Quilometragem Inicial no caso do integrante ser do grupo automóveis
Observação	Observação

Tabela 24 - Campos para Iniciar uma Atividade

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Obs.1: O Motorista/Operador deve possuir carteira de habilitação compatível com a habilitação requerida pelo integrante que será utilizado na atividade.

Obs.2: O Campo Quilometragem Inicial deverá ser obrigatoriamente preenchido se o integrante for do grupo automóvel, senão o campo não aparecerá no formulário.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF34] Finalizar Atividade

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja finalizar uma atividade de um integrante da frota. Para isto ele dever listar todas as atividades iniciadas através do requisito [RF59], após isto ele visualizará a atividade através do requisito [RF36]. Posteriormente ele poderá finalizar a atividade escolhida preenchendo os campos de acordo com a tabela abaixo.

Nome do Campo	Descrição do Campo
# Código	Código gerado no [RF33]
# Data Inicial	Data do Início da Atividade
* Data Final	Data do Término da Atividade

# Integrante	[RF13] – Integrante da Frota
# Placa do Integrante	[RF13] – Integrante da Frota
# Motorista/Operador	[RF17] - Nome do motorista ou operador
# Hora Inicial	Hora do início da Atividade
* Hora Final	Hora do término da Atividade
# Quilometragem Inicial	Quilometragem Inicial no caso de integrante do grupo Automóveis
* Quilometragem Final/Horas Trabalhadas	Quilometragem final – No caso de automóveis Hora Trabalhadas – No caso de tratores
* Multa	1 – Sim, 2 – Não(default)
Gravidade da Multa	No caso de multa. É obrigatório

	informar a gravidade
* Acidente	1 – Sim, 2 – Não(default)
Gravidade do Acidente	No caso de acidente, é obrigatório informar a gravidade.
Observação sobre o percurso	Observação
Outras Observações	Outras Observações
* Nota de Abastecimento	O Ator deverá selecionar a nota de abastecimento que está cadastrada no sistema como Nota Fiscal de Gasto, para que se possa avaliar o consumo de combustível do integrante.

Tabela 25 - Campos para Finalizar Atividade

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Obs.1: O Ator somente poderá selecionar mais de uma nota de abastecimento entre as notas fiscais cadastradas no sistema, para o campo Nota de Abastecimento.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF35] Cancelar Atividade

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja cancelar uma atividade de um integrante da frota. Para isto ele deverá listar todas as atividades iniciadas através do requisito [RF59], após isto ele visualizará a atividade através do requisito [RF36]. Posteriormente ele poderá cancelar a atividade escolhida, mas será obrigatório preencher uma observação com o motivo do cancelamento.

Nome do Campo	Descrição do Campo
Observação	Informar uma observação com o motivo do cancelamento.

Tabela 26 . - Campos para Cancelar atividade

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF36] Visualizar Atividade

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar uma atividade de um integrante da frota. O Ator deverá selecionar para a visualização entre atividades em andamento e atividades finalizadas, para isto ele dever listar as atividades através do requisito [RF59], após isto ele visualizará a atividade.

Nome do Campo	Descrição do Campo
Código	Código gerado automaticamente pelo sistema
Data	Data da Atividade (Preenchida automaticamente pelo sistema)
Integrante	[RF13] – Nome do Integrante da Frota
Placa do Integrante	[RF13] – Placa do Integrante da Frota
Motorista/Operador	[RF17] - Nome do motorista ou operador

Hora Inicial	Hora do início da Atividade
Data de Término	Data Prevista de término da atividade
Hora de término	Hora Prevista de término da atividade
Quilometragem Inicial	Quilometragem Inicial no caso do integrante ser do grupo automóveis
Observação	Observação

Tabela 27 - Campos de visualização de Atividade em Andamento

Nome do Campo	Descrição do Campo
Código	Código gerado no [RF33]
Data	Data do Início da Atividade
Integrante	[RF13] – Integrante da Frota
Placa do Integrante	[RF13] – Integrante da Frota

Motorista/Operador	[RF17] - Nome do motorista ou operador
Hora Inicial	Hora do início da Atividade
Hora Final	Hora do término da Atividade
Quilometragem Inicial	Quilometragem Inicial no caso de integrante do grupo Automóveis
Quilometragem Final/Horas Trabalhadas	Quilometragem final – No caso de automóveis Hora Trabalhadas – No caso de tratores
Multa	1 – Sim, 2 – Não
Gravidade da Multa	Gravidade da multa, se esta ocorreu.
Acidente	1 – Sim, 2 – Não
Gravidade do Acidente	Gravidade do acidente, se este

	aconteceu.
Observação sobre o percurso	Observação
Outras Observações	Outras Observações
Nota de Abastecimento	Nota de abastecimento que informa a quantidade de combustível gasta na atividade.

Tabela 28 - Campos de Visualização de Atividade Finalizada

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF37] Inserir Manutenção por Acidente/Desgaste/Danificação

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja cadastrar um problema que deva ser reparado em um integrante da frota

Nome do Campo	Descrição do Campo
# Código	Código gerado automaticamente pelo sistema

*Descrição	Descrição detalhada da manutenção
Atividade	[RF33] - Se for proveniente de alguma atividade, esta deverá ser informada.
*Data da Manutenção	Data em que o registro foi inserido
Data Prevista para o Reparo	Data Prevista para o Reparo
Observação	Observação
*Urgência	A (Urgente), B, C, D e E (Não Urgente)
* Categoria	1 – Manutenção Preventiva (neste caso a data prevista é obrigatória) 2 – Manutenção por Desgaste 3 – Manutenção por Acidente.
*Integrante	[RF13] – Nome do Integrante da Frota

Tabela 29 . - Campos para Cadastro de Manutenção por Acidente/Desgaste/Danificação

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF38] Remover Manutenção por Acidente/Desgaste/ Danificação

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja remover uma manutenção previamente cadastrada no sistema. Primeiramente ele deve listar as manutenções através do requisito [RF60]. Posteriormente ele seleciona uma e visualiza através do requisito [RF39], após isto ele poderá removê-la.

A manutenção só poderá ser removida se ela não possuir nenhum vínculo no sistema.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF39] Visualizar Manutenção por Acidente/Desgaste/Danificação

---

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar uma manutenção cadastrada previamente no sistema. Para isto ele dever listar as manutenções através do requisito [RF60], após isto ele visualizará a manutenção.

Nome do Campo	Descrição do Campo
Código	Código gerado automaticamente pelo sistema
Descrição	Descrição detalhada da manutenção
Atividade	Se for proveniente de alguma atividade, esta deverá ser informada.
Data da Manutenção	Data em que o registro foi inserido
Data Prevista para o Reparo	Data Prevista para o Reparo
Observação	Observação
Urgência	A (Urgente), B, C, D e E(Não Urgente)
Categoria	1 – Manutenção Preventiva (neste caso a data prevista é obrigatória)

	2 – Manutenção por Desgaste 3 – Manutenção por Acidente.
Integrante	[RF13] – Integrante da Frota
kmprogramados	Com quantos Km rodados o automóvel vai repetir a manutenção
km	Com quantos Km rodados o automóvel fez ou vai fazer a manutenção
horasPrevistas	Com quantas horas rodadas o trator vai repetir a manutenção
horas	Com quantas horas o trator fez ou vai fazer a manutenção

Tabela 30 . - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.1.8 Requisitos de Sistema

[RF40] Inserir Categoria de Habilitação

Ator: Sistema e Usuário Administrador.

Este requisito funcional começa quando o ator deseja cadastrar uma nova categoria de habilitação no sistema.

No momento da implantação do sistema já deverá ser inserido no sistema as seguintes categorias:

A (1);

B (2);

C (3);

D (4);

E (5);

AB (6);

AC (7);

AD (8);

AE (9).

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Categoria	Nome da Categoria.
Descrição	Descrição da Categoria

Tabela 31 - Campos para o cadastro de Categoria de Habilitação.

*Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF41] Inserir Unidade

Ator: Sistema e Usuário Administrador.

Este requisito funcional começa quando o ator deseja cadastrar uma nova unidade no sistema.

No momento da implantação do sistema já deverá ser inserido as seguintes unidades:

Unitário (1);

Litro (2);

Homem/hora (3);

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Nome	Nome da Unidade.
* Tipo	Real ou Inteiro

Tabela 32 - Campos para o cadastro de unidade.

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF42] Inserir Grupo de Integrante

Ator: Sistema.

Este requisito funcional começa na implantação do sistema, onde deverão ser inseridos no sistema os dois grupos básicos de integrantes da frota. Sendo eles:

Automóveis, com o código 1(um);

Tratores, com o código 2(dois).

Nome do Campo	Descrição do Campo
*Código	Código do Grupo
*Nome do Grupo	Nome (Automóveis ou Tratores)

Tabela 33 - Campos para o Cadastro de Grupo de Integrante

* Esses campos são considerados de preenchimento obrigatório

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF43] Inserir Tipo de Combustível

Ator: Sistema e Usuário Administrador.

Este requisito funcional começa quando o ator deseja cadastrar um novo tipo de combustível.

No momento da implantação do sistema já deverá ser inserido os seguintes tipos:

Álcool (1);

Biodiesel (2);

Diesel (3);

Flex (4);

Gás Natural (5);

Gasolina (6).

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
*Nome	Nome do Tipo (Álcool, Gasolina, Flex, Diesel, etc)

Tabela 34 Campos para o cadastro de tipo de combustível

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF44] Inserir Itens

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja adicionar um novo item no sistema.

Exemplo: Pneu, Gasolina, Mão-de-obra de Eletricista, etc.

Nome do Campo	Descrição do Campo
# Código	Código único gerado automaticamente pelo sistema
* Descrição	Descrição do Item
* Marca	Marca do Item
Valor	Valor do Item
* Unidade	[RF41] - Unidade do Item
* Tipo	Tipo do Item: 1 – Produto 2 – Serviço

Tabela 35 - Campos para o Cadastro de Itens

* Esses campos são considerados de preenchimento obrigatório

Esses campos somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF45] Alterar Item

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja alterar os dados de um Item anteriormente cadastrado no sistema. Primeiramente devem ser listados os itens do sistema através do requisito [RF57], e ao escolher um dos listados, será obtida a visualização de seus dados através do requisito [RF47]. A partir disso, este mesmo Item visualizado poderá ter seus dados alterados.

Os campos da Tabela 36 não podem ser modificados, os demais campos presentes na Tabela 35 poderão ser alterados.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema

Tabela 36 - Campos não modificáveis

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF46] Remover Item

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja remover um item anteriormente cadastrado no sistema. Primeiramente devem ser listados os Itens do sistema (através do requisito [RF57]), e ao escolher um dos listados, será obtida a visualização de seus dados (através do requisito [RF47]). A partir disso, este mesmo Item visualizado poderá ter seus dados removidos.

Neste caso o item não é removido fisicamente do sistema, é apenas desativado de todas as funcionalidades, exceto nas funcionalidades referente a histórico.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF47] Visualizar Item

Ator: Usuário Administrador ou Usuário Gestor.

Este requisito funcional começa quando o Ator deseja visualizar um item previamente cadastrado no sistema. Primeiramente ele deve listar os itens cadastrados através do requisito [RF57]. A partir disto ele selecionará um item para ser visualizado.

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
Descrição	Descrição do Item
Marca	Marca do Item
Valor	Valor do Item
Unidade	[RF41] - Unidade do Item
Tipo	Tipo do Item: 1 – Produto 2 – Serviço

Tabela 37 - Campos para a Visualização

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.1.9 Relatórios/Listagens Gerais

Todos os relatórios relacionados a essa seção estão vinculados diretamente ao ambiente de administração do sistema.

Estes relatórios poderão ser visualizados através do sistema, impressos em uma impressora instalada no computador local, bem como exportados para arquivo no formato PDF.

Obs.1: Os relatórios impressos e exportados para arquivo PDF deverão ser formatos em folha de tamanho A4.

[RF48] Relatório/Listagem de Todos os Usuários do Sistema

Ator: Usuário Administrador.

Listagem de todos os usuários do sistema, contendo os seguintes campos:

Nome do Campo	Descrição do Campo
& Nome	Nome do Usuário
Perfil	Perfil do Usuário
Login	Login *do Usuário*

Tabela 38 - Campos apresentados no relatório de Usuários do Sistema, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar usuário do sistema

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF49] Relatório/Listagem de Todos os Perfis

Ator: Usuário Administrador.

Listagem de todos os perfis do sistema contendo os seguintes campos:

Nome do Campo	Descrição do Campo
& Nome	Nome do Perfil
Descrição	Descrição do Perfil do Usuário

Tabela 39 - Campos apresentados no relatório de Perfis, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar perfil de usuário do sistema

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF50] Relatório/Listagem de Todos os Subgrupos de Integrantes

Ator: Usuário Administrador.

Listagem de todos os subgrupos do sistema, contendo os seguintes campos:

Nome do Campo	Descrição do Campo
Código	Código gerado automaticamente pelo sistema
& Nome	Nome do Subgrupo
Grupo	[RF42] – (Automóveis ou Tratores)

Tabela 40 - Campos apresentados no relatório de subgrupos, que somente poderão ser visualizados.

& Este campo será um hiperlink para a funcionalidade visualizar subgrupo de integrante

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF51] Relatório/Listagem dos Motoristas/Operadores (Por Classificação)

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de todos os Motoristas/ Operadores agrupados por classificação (Motorista, Operador ou Motorista e Operador).

Caso nenhum campo seja informado para a listagem, todos os motoristas/operadores da frota deverão ser exibidos, agrupados por grupo e subgrupo.

Nome do Campo	Descrição do Campo
Classificação	Classificação do Motorista: Motorista, Operador ou Motorista e Operador

Tabela 41 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Grupo	Nome do Grupo
> Subgrupo	Nome do Subgrupo

>> Código	Código do Motorista/Operador
>> & Nome	Nome do Motorista/Operador
>> Matrícula	Número de Matrícula do Motorista/Operador
>> CPF	CPF do Motorista/Operador

Tabela 42 - Campos apresentados no relatório de motorista/operador, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar motorista/operador

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF52] . Relatório/Listagem dos Integrantes da Frota (Por Grupo e Subgrupo)

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de todos os integrantes da frota separados por Grupo e Subgrupo.

Caso nenhum campo seja informado para a listagem, todos os integrantes da frota deverão ser exibidos, agrupados por grupo e subgrupo.

Nome do Campo	Descrição do Campo
Grupo	[RF42] – (Automóveis ou Tratores)
Subgrupo	Subgrupo - [RF09]
Placa	Placa do integrante

Tabela 43 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Grupo	[RF42] – (Automóveis ou Tratores)
> Subgrupo	Subgrupo - [RF09]
>> & Integrante	Nome do Integrantre
>> Placa do Integrante	Placa do integrante
>> Código	Código do Integrante

Tabela 44 - Campos apresentados no relatório de Integrantes da Frota, que somente poderão ser visualizados.

& Este campo será um hiperlink para a funcionalidade visualizar integrantes

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF53] Relatório/Listagem dos Fornecedores

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de todos os fornecedores do sistema, contendo os seguintes campos:

Nome do Campo	Descrição do Campo
& Razão Social	Razão Social do Fornecedor
CNPJ	CNPJ do Fornecedor

Tabela 45 - Campos apresentados no relatório de Fornecedores, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar fornecedor

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF54] Relatório/Listagem de Nota Fiscal de Gasto

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de Nota Fiscais de Gasto com a Frota.

Caso nenhum campo seja informado para a listagem, todas as notas serão exibidas, agrupadas por fornecedor.

Nome do Campo	Descrição do Campo
Número	Número da Nota
Fornecedor	Nome do fornecedor

Tabela 46 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Fornecedor	[RF21] – Fornecedor da Nota
> & Número	Número da Nota Fiscal
> Data	Data de Emissão da Nota
> Valor	Valor da Nota

Tabela 47 - Campos apresentados no relatório de Notas Fiscais, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar nota fiscal de gasto

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF55] Relatório/Listagem de Gastos com Integrantes

Ator: Usuário Administrador ou Usuário Gestor.

Listagem dos gastos com integrantes da frota.

Caso nenhum campo seja informado para a listagem, todos os integrante serão exibidos agrupadas por grupos e subgrupos.

Nome do Campo	Descrição do Campo
Grupo	[RF42] – Grupo do Integrante
Subgrupo	[RF09] – Subgrupo do Integrante
Integrante	[RF13] – Integrante da frota

Tabela 48 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Grupo	[RF42] – Grupo do Integrante
> Subgrupo	[RF09] – Subgrupo do Integrante
>> Integrante	Nome do Integrante
>> Placa do Integrante	Placa do Integrante
>>> Número da Nota Fiscal	Número da Nota
>>> Valor	Valor gasto com o Integrante na nota indicada
>> Gasto Total	Valor de todos os gastos com o integrante

Tabela 49 - Campos apresentados no relatório de Gasto com Integrantes, que somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF56] Relatório/Listagem de Associação de Item da Nota Fiscal a Integrante

Ator: Usuário Administrador ou Usuário Gestor.

Listagem da associação dos itens da nota fiscal com os integrantes da frota.

Caso nenhum campo seja informado para a listagem, todas as associações serão exibidas, agrupadas por fornecedor.

Nome do Campo	Descrição do Campo
Fornecedor	[RF21] – Fornecedor da Nota
Número da Nota	Número da Nota

Tabela 50 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Fornecedor	[RF21] – Fornecedor da Nota
> Número da Nota	Número da Nota

>> Integrante	[RF13] - Integrante da Frota
>>> Item	Item da Nota
>>> Quantidade	Quantidade do Item alocado para o Integrante

Tabela 51 - Campos apresentados no relatório de Associação de Item da Nota com Integrante, que somente poderá ser visualizado

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF57] Relatório/Listagem de Itens

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de todos os itens do sistema, contendo os seguintes campos:

Nome do Campo	Descrição do Campo
Código	Código do Item
& Nome	Nome do Item

Unidade	Unidade do Item
Valor	Valor do Item
Tipo	Tipo do Item – Produto ou Serviço

Tabela 52 - Campos apresentados no relatório de itens, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar item

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF58] Relatório/Listagem de Notas de Serviço

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de todas as notas de Serviço do sistema.

Caso nenhum campo seja informado para a listagem, todas as notas de serviço serão exibidas, agrupadas por integrante.

Nome do Campo	Descrição do Campo
Integrante	Nome do Integrante

Data de Emissão (Início)	Notas a partir desta data
Data de Emissão (Término)	Notas anteriores a esta data

Tabela 53 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Integrante	Nome do Integrante
> & Número da Nota	Número da Nota
> Data	Data de Emissão da Nota
>> Itens	[RF44] - Todos os Itens da Nota
>> Quantidade	Quantidade de Cada Item da Nota

Tabela 54 - Campos apresentados no relatório de Notas de Serviço, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar notas de serviço

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF59] Relatório/Listagem de Atividades

Ator: Usuário Administrador, Usuário Responsável pela Frota ou Usuário Gestor.

Listagem de todas as atividades finalizadas e não finalizadas.

O Ator deverá selecionar o Status da atividade cadastrada no sistema.

Nome do Campo	Descrição do Campo
Status	Finalizadas ou Em Andamento

Tabela 55 - Campos para realizar a pesquisa

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
& Atividade	Nome da Atividade
Integrante	[RF42] – (Automóveis ou Tratores)
Motorista/Operador	[RF17] – Motorista/Operador

Data/Hora	Data e Hora do início da Atividade

Tabela 56 Campos apresentados no relatório de Atividades em Andamento, que somente poderão ser visualizados. (a)

& Este campo será um hiperlink para a funcionalidade visualizar atividades

Nome do Campo	Descrição do Campo
Código	Código único gerado automaticamente pelo sistema
& Atividade	Nome da Atividade
Integrante	[RF42] – (Automóveis ou Tratores)
Motorista/Operador	[RF17] – Motorista/Operador

Tabela 57 - Campos apresentados no relatório de Atividades Finalizadas, que somente poderão ser visualizados. (b)

& Este campo será um hiperlink para a funcionalidade visualizar atividades

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF60] Relatório/Listagem de Manutenções (Entre um Intervalo de Datas)

Ator: Usuário Administrador ou Usuário Gestor.

Listagem de manutenções não finalizadas.

Caso nenhum campo seja informado para a listagem, todas as manutenções não finalizadas serão exibidas, agrupadas por grupos.

Nome do Campo	Descrição do Campo
* Data de Início	Data de Início do período a ser pesquisado
* Data de Fim	Data de Término do período a ser pesquisado
Urgência	A (Urgente), B, C, D e E (Não Urgente)
Categoria	1 – Manutenção Preventiva (neste caso a data prevista é obrigatória) 2 – Manutenção por Desgaste

	3 – Manutenção por Acidente.

Tabela 58 - Campos para realizar a pesquisa

* Esses campos são considerados de preenchimento obrigatório

Nome do Campo	Descrição do Campo
Categoria	1 – Manutenção Preventiva (neste caso a data prevista é obrigatória) 2 – Manutenção por Desgaste 3 – Manutenção por Acidente.
> & Código	Código único gerado automaticamente pelo sistema
> Integrante	[RF13] – Integrante da Frota
> Urgência	Descrição da Urgência
> Data	Data do cadastro da manutenção

Tabela 59 - Campos apresentados no relatório de Manutenção, que somente poderão ser visualizados

& Este campo será um hiperlink para a funcionalidade visualizar Manutenção

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

[RF61] Relatório/Listagem de Memorial de Integrante

Ator: Usuário Administrador ou Usuário Gestor.

Gerar o memorial, o qual deverá conter todas as manutenções de um integrante do sistema, previamente cadastrada.

Nome do Campo	Descrição do Campo
Data de Início	Data de Início do período a ser pesquisado
Data de Fim	Data de Término do período a ser pesquisado
Status	Completa ou Incompleta
Urgência	A (Urgente), B, C, D e E (Não Urgente)

Categoria	1 – Manutenção Preventiva (neste caso a data prevista é obrigatória) 2 – Manutenção por Desgaste 3 – Manutenção por Acidente.
* Integrante	[RF13] – Integrante da frota

Tabela 60 - Campos para realizar a pesquisa

* Esses campos são considerados de preenchimento obrigatório

Nome do Campo	Descrição do Campo
Status	Completa ou Incompleta
> Categoria	1 – Manutenção Preventiva (neste caso a data prevista é obrigatória) 2 – Manutenção por Desgaste 3 – Manutenção por Acidente.

>> Urgência	A (Urgente), B, C, D e E (Não Urgente)
>>> Manutenção	[RF37] – Descrição da Manutenção

Tabela 61 - Campos apresentados no Memorial, que somente poderão ser visualizados

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.2 Requisitos não funcionais

Esta seção contém os requisitos não funcionais do sistema. Para uma melhor organização deste documento, utilizamos as subseções abaixo para agrupar os requisitos não funcionais relacionados.

7.2.1 Descrição do Componente

Descrição do componente é um documento expondo as propriedades do componente, com o principal objetivo de auxiliar os potenciais adquirentes, sejam usuários ou integradores, na avaliação da adequação do componente antes de sua aquisição. Deve seguir os requisitos de qualidade da descrição de produto da norma NBR ISO/IEC 12119.

7.2.1.1 Existência

Disponibilidade da Descrição do Componente

O componente Via Digital deve disponibilizar no portal um documento (ou página) de Descrição do Componente.

Prioridade: ☒ Essencial ☐ Importante ☐ Desejável

7.2.1.2 Conteúdo

[RNF01] Clareza

O texto da descrição do componente deve ser claro não dando margem à interpretações ambíguas.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

7.2.1.3 Identificações e Indicações

[RNF02] Identificação do Componente

A Descrição do Componente deve conter as seguintes informações de identificação:

- Identificação do nome do componente;
- Identificação da versão ou a data de criação do componente;
- Identificação do produtor;
- Identificação do nome do produtor/fornecedor do componente;
- Identificação de uma forma de contato.

Prioridade: ☒Essencial ☐ Importante ☐ Desejável

[RNF03] Identificação do Uso e Especificidade

A Descrição do Componente deve conter a classificação do mesmo acordo com seu uso e especificidade. Isto é, se ele é:

- Componente Genérico: de uso comum em muitos sistemas (Por exemplo, componentes de interface com os usuários tais como botões, janelas, entre outros);

- Componente de Serviços: fornecem serviços especializados, mas que não são específicos do ponto de vista de domínio de aplicação (Por exemplo, componentes para tratamento de erros em comunicação de dados, criptografia, segurança, geração de gráficos, entre outros); ou

- Componente de Domínio: específico para domínio definido, que implementam regras de negócios simples ou complexas (Por exemplo, regras do setor financeiro ou de construção civil)

Prioridade: ☐ Essencial ☒Importante ☐ Desejável .

[RNF04] Identificação de Tarefas

A Descrição do Componente deve identificar as tarefas que podem ser realizadas com a utilização do componente.

Prioridade: ☐ Essencial ☐ Importante ☒Desejável

[RNF05] Identificação de Documentos de Requisitos

Convém que a Descrição do Componente identifique os documentos de requisitos com o qual o componente está em conformidade.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF06] dentificação de Requisitos de Hardware

A Descrição do Componente deve identificar os requisitos de hardware.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF07] Identificação de Requisitos de Software

A Descrição do Componente deve identificar os requisitos de software.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF08] Identificação de Interfaces com Outros Produtos

Se o componente tiver interfaces com outros produtos esses devem estar identificados com nome e versão ou data de criação na Descrição do Componente.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF09] Identificação dos Artefatos

A Descrição do Componente deve identificar todos os artefatos que o acompanham.

Prioridade: ☐ Essencial ☒Importante ☐ Desejável

[RNF10] Indicação de Instalação ou Integração

A Descrição do Componente deve conter uma indicação a respeito de quem pode instalar e ou integrar o componente (Por exemplo, o usuário ou a equipe de suporte).

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável

[RNF11] Declaração sobre Fornecimento de Suporte

A Descrição do Componente deve conter uma declaração sobre fornecimento de suporte.

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável

[RNF12] Declaração sobre Fornecimento de Manutenção

A Descrição do Componente deve conter uma declaração sobre fornecimento de manutenção.

Prioridade: ☐ Essencial ☐ Importante ☒Desejável

7.2.1.4 Declarações

[RNF13] Declaração sobre as Funções

A Descrição do componente deve conter uma visão geral das funções chamadas pelo usuário, os dados necessários e as facilidades oferecidas.

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável

[RNF14] Declaração sobre os Valores-limite

A Descrição do componente deve conter declarações de valores-limite, caso o uso do componente seja limitado por eles.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF15] Declaração sobre Recursos de Segurança de Acesso

Caso o componente apresente recursos de segurança de acesso, a descrição do componente deve descrever esses recursos para evitar o acesso não autorizado (acidental ou intencional) aos serviços e dados.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF16] Declaração sobre Confiabilidade

A descrição do componente deve conter uma declaração sobre procedimentos ou propriedades para preservação de dados (Por exemplo: backup pelo sistema operativo, verificação das entradas, recuperação após erros).

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF17] Declaração sobre Usabilidade

A descrição do componente deve conter declarações sobre:

1) O tipo de interface com o usuário;

2) Os conhecimentos requeridos pelo usuário do componente;

3) Adaptações que podem ser realizadas pelo usuário. Por exemplo: parâmetros, algoritmos, teclas de função.

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável .

[RNF18] Declaração sobre Eficiência

A Descrição do Componente pode conter uma declaração sobre o comportamento do componente em relação ao tempo e em relação aos recursos de hardware ou software.

Prioridade: ☐ Essencial ☐ Importante ☒Desejável

[RNF19] Declaração sobre Manutenibilidade

A Descrição do Componente pode conter uma declaração sobre a manutenibilidade do componente, ou seja, sobre a facilidade de ser modificado e testado.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF20] Declaração sobre Portabilidade

A Descrição do Componente pode conter uma declaração sobre a portabilidade do componente.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.2 Documentação para o Usuário

O componente admissível para o Via Digital deverá disponibilizar uma documentação para o usuário, contendo todas as informações necessárias para a correta utilização do componente Via Digital.

A documentação pode ser apresentada na forma impressa, em mídia ou on-line.

Essa documentação deve seguir os requisitos de qualidade das normas NBR ISO/IEC 12119 e IEEE 1063, como especificados nas subseções abaixo:

7.2.2.1 Completitude

A Documentação para o usuário deve conter todas as informações necessárias para a utilização do mesmo. No mínimo deve conter as informações especificadas nos requisitos desta subseção.

[RNF21] Identificação do Componente

A Documentação para o Usuário deve identificar o componente, incluindo: o nome , o número da versão e/ou a data da criação da versão.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF22] Identificação do Produtor

A Documentação para o Usuário deve identificar o produtor, incluindo: nome, endereço e meio de contato (telefone, fax e e-mail).

Prioridade: ☐ Essencial ☒Importante ☐ Desejável

[RNF23] Descrição das Tarefas

A Documentação para o Usuário deve conter uma descrição das tarefas que podem ser realizadas com a utilização do componente.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF24] Descrição das Funções

A Documentação para o Usuário deve conter uma descrição geral do componente e uma descrição completa de cada uma das funções chamadas pelo usuário através da interface do componente.

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável

[RNF25] Especificação dos Valores Limite

A Documentação para o Usuário deve conter a especificação de todos os valores limite, incluindo todos os declarados na Descrição do Componente. Por exemplo, valores máximos ou mínimos de determinados campos, quantidade máxima de registros em um arquivo, quantidade máxima de tentativas com uso de senha errada, quantidade máxima de usuários simultâneos, etc. No caso de não ser possível fornecer limites fixos (Por exemplo, quando eles dependem do tipo de aplicação ou do tipo de dados de entrada), as limitações devem ser especificadas. Combinações de valores, se permitidas, devem ser fornecidas.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF26] Manual de Instalação ou Integração

Caso na Descrição do Componente estiver indicado que o usuário poderá instalar ou integrar o componente, a Documentação para o Usuário deverá apresentar um manual de instalação ou de integração.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF27] Conhecimento Requerido

A Documentação para o Usuário deve conter informações relativas ao conhecimento específico requerido para a utilização do componente. Por exemplo, conhecimento de uma área técnica, conhecimento que possa ser adquirido num treinamento especial, etc.

Prioridade: ☐ Essencial ☐ Importante ☒Desejável

[RNF28] Recursos de Proteção

A Documentação para o Usuário deve conter informações relativas aos recursos de proteção do componente com vistas a evitar acesso não autorizado (acidental ou intencional). Por exemplo, criptografia dos dados, senha de acesso, etc.

Prioridade: ☐ Essencial ☒Importante ☐ Desejável

[RNF29] Interfaces com outros Produtos de Software

A Documentação para o Usuário deve conter informações relativas à utilização das interfaces do componente com outros produtos de software.

Prioridade: ☐ Essencial ☒Importante ☐ Desejável

[RNF30] Procedimentos para Preservação de Dados

A Documentação para o Usuário deve conter informações relativas a procedimentos para preservação de dados. Descrevendo propriedades adicionais destinadas a assegurar a capacidade funcional do componente. P. ex., verificação de dados de entrada aceitáveis, proteção contra conseqüências danosas decorrentes de erro de usuário, recuperação de erro, etc.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF31] Procedimentos para Adaptação ou Manutenção

Caso o componente possa ser adaptado ou mantido pelo usuário, a Documentação para o Usuário deve conter informações relativas a procedimentos para adaptação ou manutenção e as condições para seu uso. (como é isso em software livre?) Exemplos de tipo de adaptação: mudanças de parâmetros do componente, geração ou alteração de algoritmos computacionais, adaptação de teclas de função, adaptação de teclas de atalho, geração ou alteração de leiautes de relatórios, personalização de objetos da interface, entre outros.

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável

7.2.2.2 Usabilidade da Documentação para o Usuário

Visando facilitar a utilização da Documentação para o Usuário, a mesma deve seguir os requisitos recomendados nesta seção.

[RNF32] Descrição da Organização

A Documentação para o Usuário deve conter:

Uma visão geral de sua estrutura;

A descrição dos símbolos, convenções estilísticas e convenções da sintaxe de comandos utilizados;

Quando necessário, uma referência sobre o conhecimento de área específica ou outras características do público alvo.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF33] Índice Geral

A Documentação para o Usuário deve apresentar índice geral que respeite a seqüência dos capítulos e indique corretamente a página inicial de cada entrada no índice. Deverá possuir entradas, pelo menos, até o terceiro nível da hierarquia estrutural do documento. Por exemplo, considera-se como índice geral da documentação on-line, a pasta "Conteúdo" do "Tópicos de Ajuda" do Office.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF34] Glossário

A Documentação para o Usuário deve apresentar Glossário, listado em ordem alfabética as definições de todos os termos, siglas e abreviações, usadas no documento, que não sejam familiares ao público alvo.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF35] Estrutura e Coerência

A Documentação para o Usuário deve apresentar boa organização no que diz respeito à estrutura das informações de forma coerente. Isto é:

Possuir relação lógica e consistente entre os assuntos que se deseja transmitir e as informações apresentadas na introdução, tópicos ou capítulos, sub-capítulos, seções, ilustrações e exemplos;

A documentação impressa deverá possuir capítulos e sub-capítulos ordenados corretamente;

A documentação on-line deverá possuir estrutura em tópicos e links coerentes com as informações.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF36] Destaque das Informações

A Documentação para o Usuário deve apresentar boa Organização, destacando as informações relevantes através do uso de negrito, itálico, etc.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

[RNF37] Consulta Sensível ao Contexto

A Documentação para o Usuário on-line deve apresentar consulta sensível ao contexto. Por exemplo, ao acionar a função de ajuda, p. ex., F1, deverá apresentar explicações referentes ao assunto ou campo selecionado na interface ou à tarefa em execução.

Prioridade: ☐ Essencial ☐ Importante ☒ Desejável

[RNF38] Clareza

A Documentação para o Usuário deve ser clara, isto é:

Livre de erros gramaticais;

Livre de erros ortográficos;

Utilizar linguagem compatível com o público a que se destina o documento;

Utilizar termos com o mesmo significado em toda a documentação;

Destacar palavras em outro idioma; e

Explicar mensagens de erro quando necessário.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF39] Consistência

A Documentação para o Usuário deve apresentar consistência, sendo:

Livre de contradições internas;

Livre de contradições com os programas e os dados.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF40] Exemplos

A Documentação para o Usuário deve apresentar exemplos, que auxiliem o usuário na compreensão dos assuntos tratados.

Prioridade: ☐ Essencial ☒Importante ☐ Desejável

[RNF41] Ilustrações

A Documentação para o Usuário deve apresentar ilustrações, quando necessário, que auxiliem o usuário na compreensão dos assuntos tratados. Por exemplo, apresentar ilustrações dos: menus, janelas, teclas de função, teclas de atalho, ícones e botão, etc.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

[RNF42] Padronização

A Documentação para o Usuário deve apresentar Padronização, nos seguintes itens:

Formatos de títulos de capítulos,

Formatos de sub-capítulos,

Formatos de seções e subseções;

Formatos de funções,

Formatos de ilustrações,

Formatos de exemplos e formatos de termos utilizados.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3 Componente Via Digital

O componente da Via Digital deve seguir os requisitos de qualidade de um produto de software alinhando-se aos requisitos da Norma NBR ISO/IEC 9126-1 e os requisitos para pacote de software da norma NBR ISO/IEC 12119.

7.2.3.1 Funcionalidade

[RNF43] Adequação

As funções devem:

Ser adequadas para executar as tarefas especificadas;

Estar implementadas de acordo como descrito na documentação ou descrição do componente;

A instalação ou integração deve ser possível seguindo o manual de instalação ou integração.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF44] Acurácia

Os resultados das operações retornados pelo componente devem ser corretos ou de acordo com o grau de precisão esperado.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF45] Segurança de Acesso

É desejável que o componente possua controle de acesso às suas funcionalidades de acordo com o perfil do usuário.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF46] Criptografia dos Dados

É desejável que o componente possua criptografia para proteger os dados que manipula.

Prioridade: ☐Essencial ☒ Importante ☐ Desejável

[RNF47] Interoperabilidade

O componente deve utilizar uma representação de dados de entrada e saída que esteja em conformidade com algum padrão ou convenção internacional.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

[RNF48] Conformidade

É desejável que as funcionalidades estejam em conformidade à regulamentações, padrões e convenções aplicáveis ao domínio de aplicação do componente.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3.2 Confiabilidade

[RNF49] Tolerância à Falhas - Prevenção de operações incorretas

É desejável que o componente possua funções que são implementadas com a capacidade de prevenir operações incorretas. Obs.: padrões de operações incorretas são: parâmetros incorretos, seqüência incorreta de entrada de dados, e seqüência incorreta de operações.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF50] Recuperabilidade

É desejável que o componente possua a habilidade de controlar situações que possam levar a erros.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3.3 Usabilidade

Aqui são apresentadas recomendações destinadas a incrementar a facilidade de interação do usuário com o Componente. Em geral, a usabilidade de um produto qualquer é vista na interface do mesmo.

[RNF51] Adaptação da Teclas de Função

A Interface deve permitir renomear teclas de função, segundo a necessidade ou preferência do usuário.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

[RNF52] Visualização de Campos de Entrada de Dados

A interface deve apresentar campos para entrada de dados com visualização diferenciada. Por exemplo, utilização de fundo branco para campos que podem receber dados (campos ativados) e fundo cinza para os demais (desativados).

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF53] Alinhamento de Campos

A interface deve alinhar campos numéricos e alfanuméricos de acordo ao seu tipo e uso. Por exemplo, campos de conteúdo alfanumérico estão alinhados à esquerda, campos de conteúdo numérico estão alinhados à direita, campos de conteúdo numérico têm seus pontos decimais e vírgulas alinhados na mesma linha vertical nas apresentações em janela.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF54] Texto

A interface deve ser livre de erros gramaticais e ortográficos.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF55] Controle de Tempo de Exposição de Mensagens

A interface deve permitir que o usuário controle o tempo de exposição das mensagens na janela, utilizando, p.ex., botões “Cancela” ou “OK” nas mensagens exibidas.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF56] Apresentação de Ícones

A interface deve apresentar ícones que representem claramente a função a ele associada. Por exemplo, ícone de impressora, cesto de papel, relógio, gráficos entre outros.

Prioridade: ☐ Essencial ☐ Importante ☐ Desejável ☒

[RNF57] Padronização Usual

A interface deve respeitar a padronização usual em relação a formatos e cores, utilizando padrões de data no formato “dd/mm/aaaa” e unidades de medida do Componente. Por exemplo, sistema métrico, decimal.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF58] Apreensibilidade

É desejável que o componente possua:

Um sistema de ajuda que facilite no aprendizado de suas funcionalidades;

Algum tipo de treinamento para a utilização.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF59] Atratividade

É desejável que a interface do componente seja atrativa para o usuário final. Por exemplo, o uso de cores, tamanho de fontes, diálogos, mensagens audiovisuais, entre outros.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

7.2.3.4 Eficiência

[RNF60] Comportamento em relação ao tempo

É desejável que o tempo de resposta gasto de um serviço a partir de um pedido recebido, até que uma resposta seja emitida, seja mínimo.

Prioridade: ☐ Essencial ☒Importante ☐ Desejável

[RNF61] Utilização de Recursos - Memória requerida

É desejável que a quantidade de memória necessária para um componente operar seja conforme a declaração de requisitos de hardware.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3.5 Manutenibilidade

[RNF62] Analisabilidade

É desejável que o componente seja projetado de tal forma que facilite a análise de falhas.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF63] Modificabilidade

É desejável que o componente seja de fácil modificação, de acordo com o declarado na Descrição do Componente.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3.6 Portabilidade

[RNF64] Adaptabilidade

É desejável que o componente oferece parâmetros de configuração para que o usuário (desenvolvedor) consiga adaptá-lo facilmente para um novo ambiente.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF65] Capacidade para ser instalado - Facilidade de instalação

---

É desejável que o componente oferece os arquivos necessários para fazer a instalação do componente ou um arquivo para realizar uma auto-instalação.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF66] Capacidade para substituir - Compatibilidade de versões

---

É desejável que o componente atual seja compatível com outras versões que estão indicadas em sua especificação. Isto é, deve:

Possuir as funcionalidades das versões anteriores, de modo que seja possível realizar todas as tarefas realizáveis com as versões anteriores;

Ser possível trabalhar com os dados das versões anteriores com um mínimo de esforço para adaptação dos mesmos.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3.7 Suporte

[RNF67] Listas de Discussão na Web

É desejável que o componente possua alguma lista de discussão onde os usuários possam obter ajuda e tirar dúvidas.

Prioridade: ☐ Essencial ☒ Importante ☐ Desejável

7.2.3.8 Tecnologia

[RNF68] Ferramental Público

É desejável que o projeto do componente seja desenvolvido com APIs, Plugins e/ou Linguagens de desenvolvimento públicas para que futuras extensões e também o seu uso seja possível por outras comunidades.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF69] Espaço Comum

É desejável que o projeto do componente possua algum local onde desenvolvedores interessados possam contribuir com o projeto.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.3.9 Maturidade

[RNF70] Equipe Profissional

É desejável que o projeto do componente possua uma equipe de profissionais qualificados.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF71] Processo de Desenvolvimento

É desejável que o projeto do componente esteja sendo desenvolvido num processo bem definido e maduro.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

[RNF72] Contribuições

É desejável que o projeto do componente prevê a possibilidade de assimilar contribuições de comunidades externas.

Prioridade:☒ Essencial ☐ Importante ☐ Desejável

7.2.4 Requisitos de Instalação

Os requisitos não funcionais que tratam das considerações necessárias para a construção do pacote de instalação do componente de infra-estrutura estão descritos nesta seção.

[RNF73] Padrão para o Componente

Esta funcionalidade descreve o padrão da estrutura de arquivos que se deve seguir para construir um pacote de componente.

O pacote deve ser disponibilizado no formato ZIP de compactação e estar em conformidade segundo informações abaixo.

Descrição dos diretórios:

- **dist**: Usado para a tarefa dist do ANT (caso seja utilizado no projeto), onde se colocar o arquivo *.war* referente ao projeto.
- **conf**: Usado para se colocar arquivos e/ou diretórios necessários à configuração do componente para sua montagem e/ou execução. Inclusive o diretório META-INF/ e seus arquivos de configuração respectivos.
- docs**: Usado para se colocar documentação referente ao projeto (modelos, HOWTOs, etc.)**
- **lib**: Usado para se colocar bibliotecas externas utilizadas pelo projeto.
- **setup**: Contém arquivos necessários ao ambiente da aplicação, tais como: scripts SQL, arquivos XML, etc.
- **src:** Colocar todos os arquivos fontes (.java), pacotes. Enfim, tudo o que for relativo à codificação do projeto.
- **test:** Colocar os arquivos referentes a testes do componente (unidade, aceitação...) neste diretório.
- **web**: Devem ser colocadas páginas JSP, HTML ou qualquer outro arquivo referente a camada de apresentação do componente.

- Subdiretórios de web:
  - **WEB-INF**: Obrigatório em qualquer aplicação web. Em sua raiz devem estar os arquivos necessários para a configuração da aplicação (*web.xml*, *faces-config.xml*, etc.).
  - **WEB-INF/jspf**: Deve conter arquivos *.jspf* caso sejam utilizados no projeto.
  - **WEB-INF/lib**: Deve conter bibliotecas externas utilizadas pelo projeto.
  - **WEB-INF/tlds**: Deve conter os arquivos *.tlds* (Descritores de *tags*).
  - **WEB-INF/tags**: Deve conter as *tags* personalizadas usadas no componente.
  - **WEB-INF/classes**: Deve conter todos o arquivos *.class* criados a partir da compilação do código fonte.

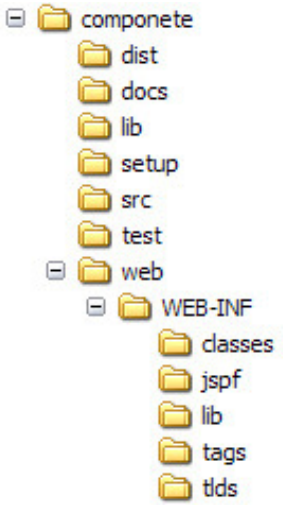


Tabela 62 - Estrutura de arquivos do componente

Prioridade:☒ Essencial ☐ Importante ☐ Desejável ☐

[RNF74] Plataformas Suportadas

Esta funcionalidade diz respeito às plataformas que devem ser suportadas pelo sistema.

O sistema deve disponibilizar instalações para as plataformas Windows e Linux. As versões dos SOs que devem suportar o sistema são:

- Windows 2000 ou superior;
- Qualquer distribuição Linux;

Prioridade:☒ Essencial ☐ Importante ☐ Desejável ☐

[RNF75] Representação do Componente em Arquivo

Esta funcionalidade diz respeito à forma como este sistema será disponibilizado ao repositório de componentes do projeto Via Digital.

Todos os arquivos do componente de planejamento e acompanhamento de frotas (SIPAF) devem ser disponibilizados em um único arquivo compactado. A extensão deste arquivo deve ser .jar.

O componente de Intra-Estrutura (SINFRA – Sistema de Infra-Estrutura) necessitará trabalhar com apenas um arquivo para efetuar a instalação do sistema especificado neste documento.

8 ANEXOII – DIAGRAMA DE CASOS DE USO – SWFACTORY

8.1 Diagrama de Caso de Uso - Cadastro de Integrantes de Frota

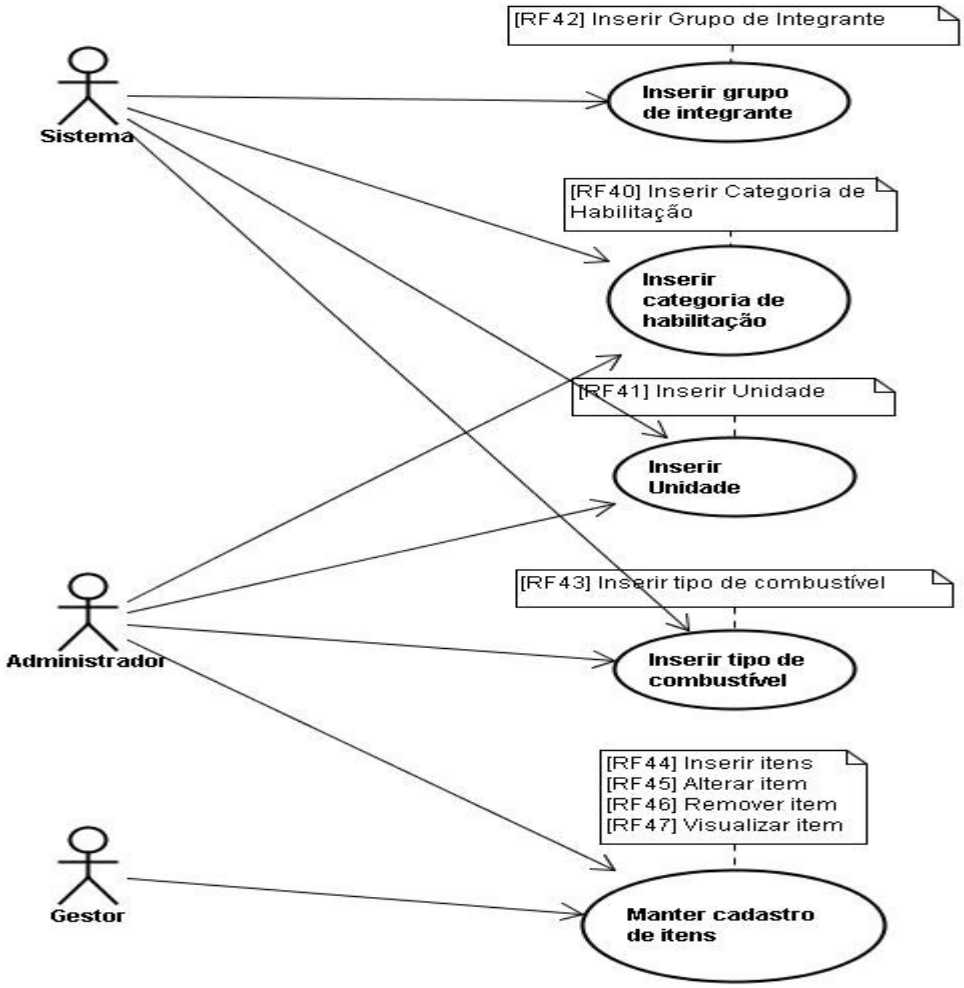


Figura 8 - Cadastro de Integrantes de Frota

8.2 Diagrama de Caso de Uso - Cadastro de Motoristas e Operadores de Frota

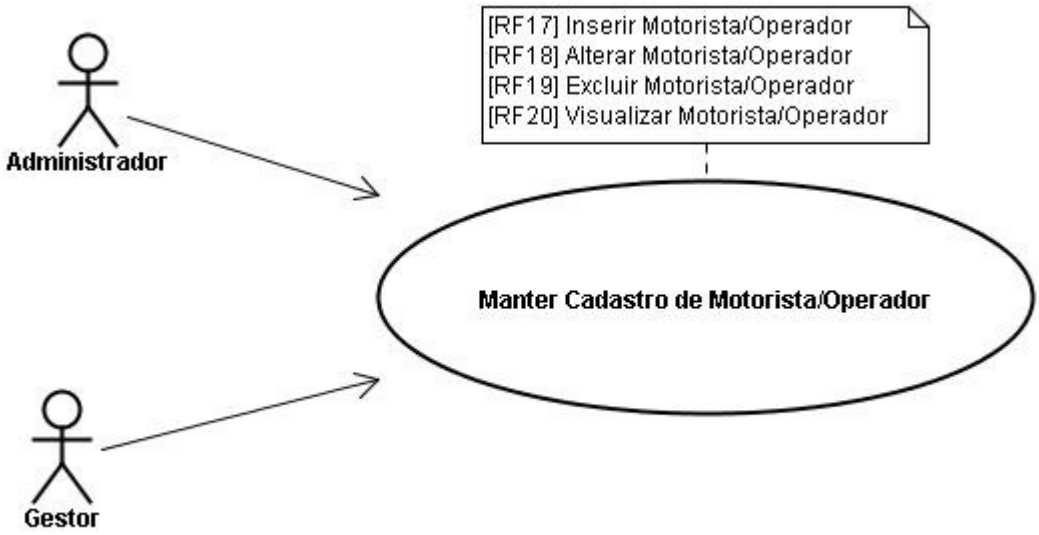


Figura 9 - Cadastro de Motoristas e Operadores de Frota

8.3 Diagrama de Caso de Uso - Cadastros Gerais

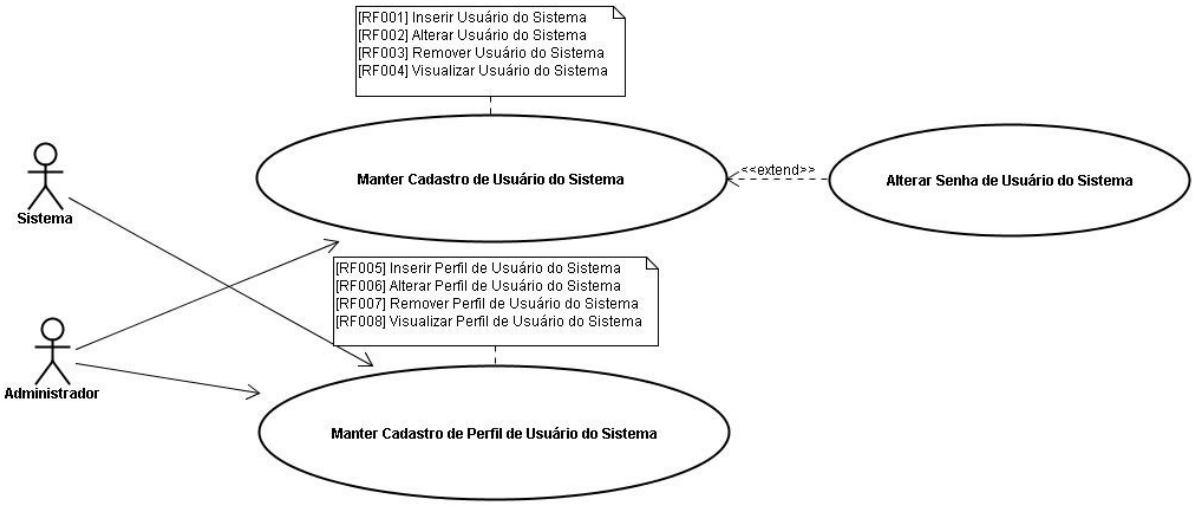


Figura 10 - Cadastros Gerais

8.4 Diagrama de Caso de Uso - Funcionalidades Gerais

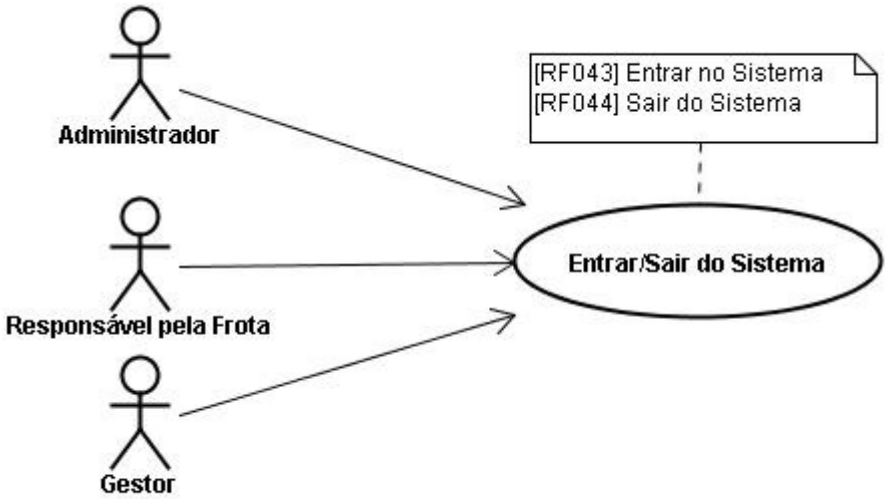


Figura 11 - Funcionalidades Gerais

8.5 Diagrama de Caso de Uso - Funcionalidades relacionadas aos gastos com frota

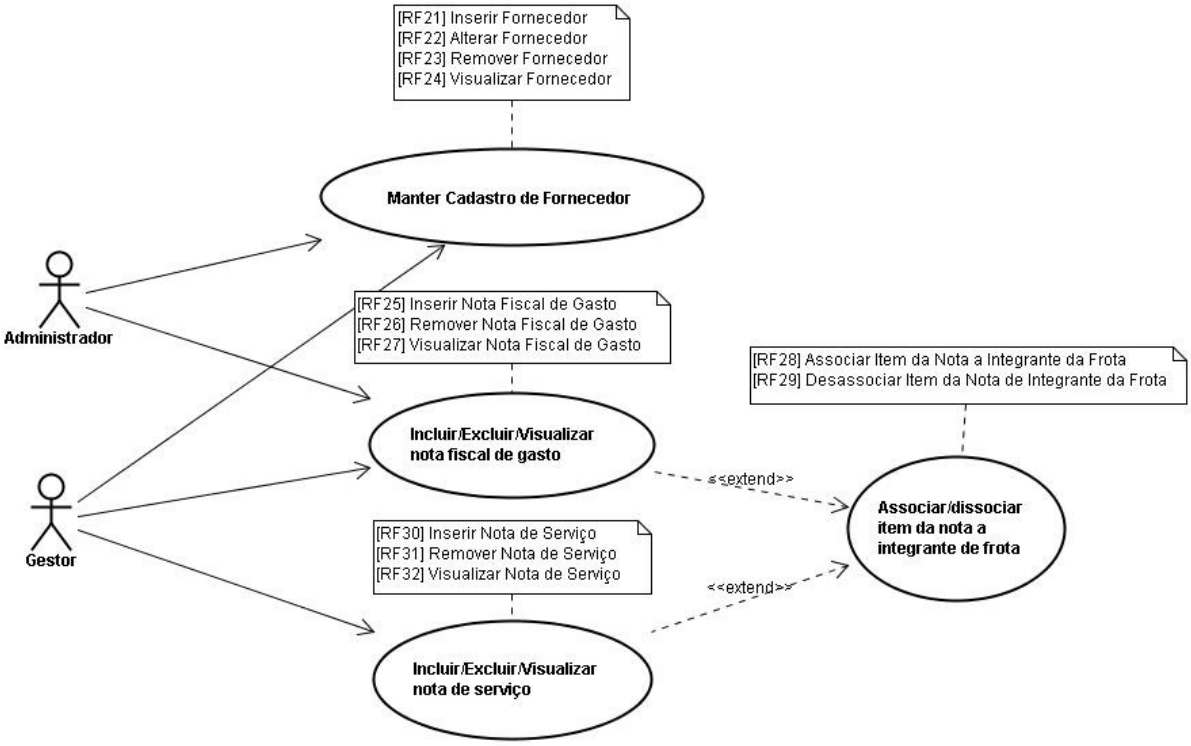


Figura 12 - Funcionalidades relacionadas aos gastos com frota

8.6 Diagrama de Caso de Uso - Registro de Atividades

Figura 13 - Registro de Atividades

8.7 Diagrama de Caso de Uso - Relatórios relacionados aos cadastros gerais, integrantes da frota e motoristas operadores da frota

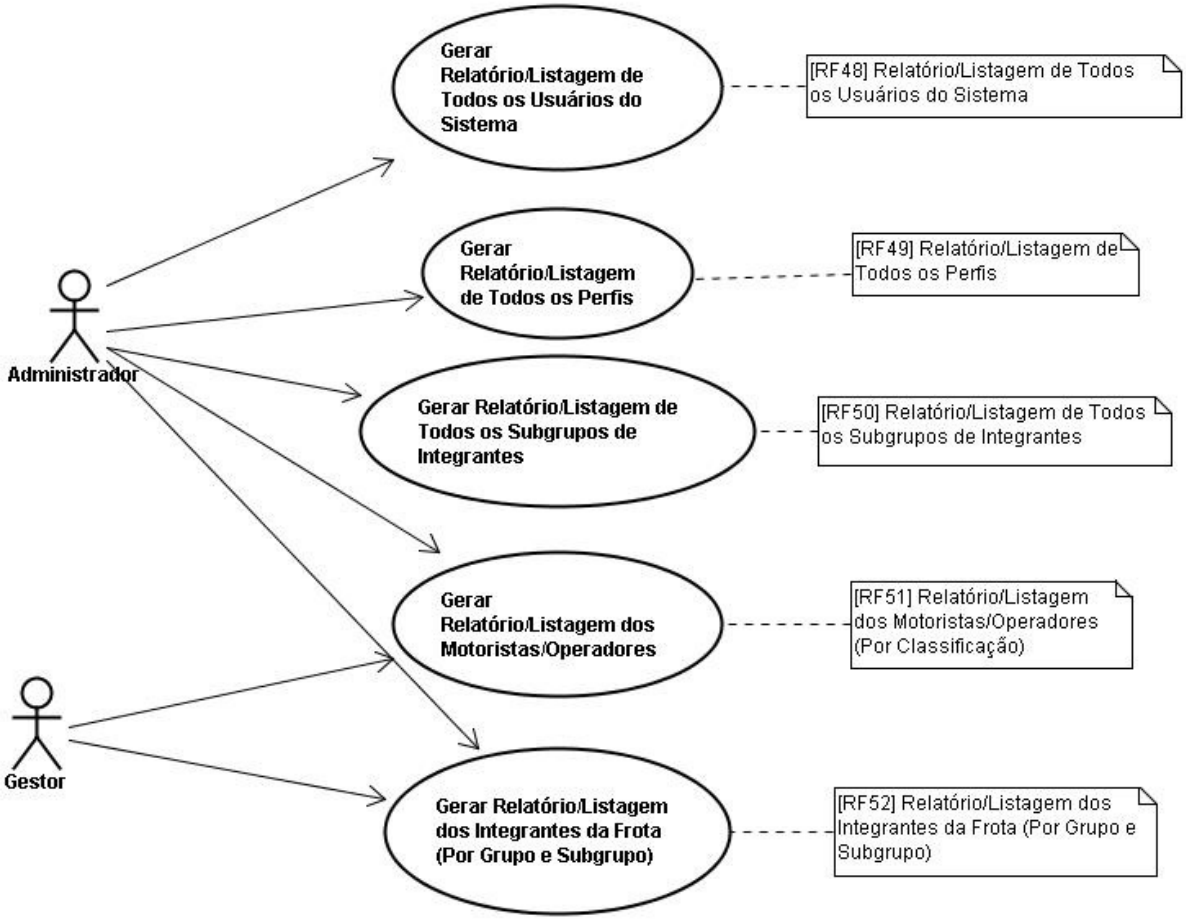


Figura 14 - Relatórios relacionados aos cadastros gerais, integrantes da frota e motoristas operadores da frota

8.8 Diagrama de Caso de Uso - Relatórios

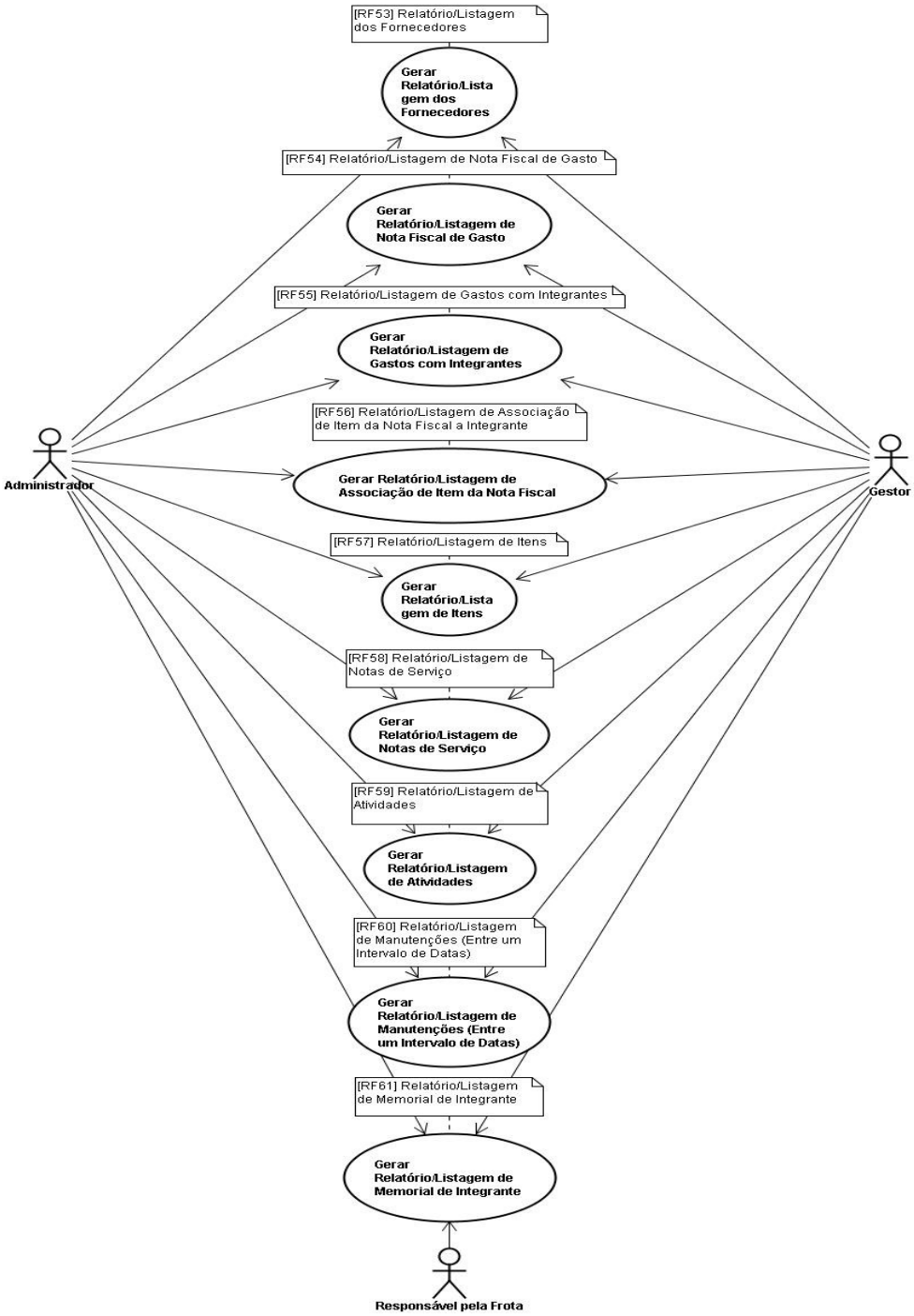


Figura 15 - Relatórios

8.9 Diagrama de Caso de Uso - Requisitos de Sistema

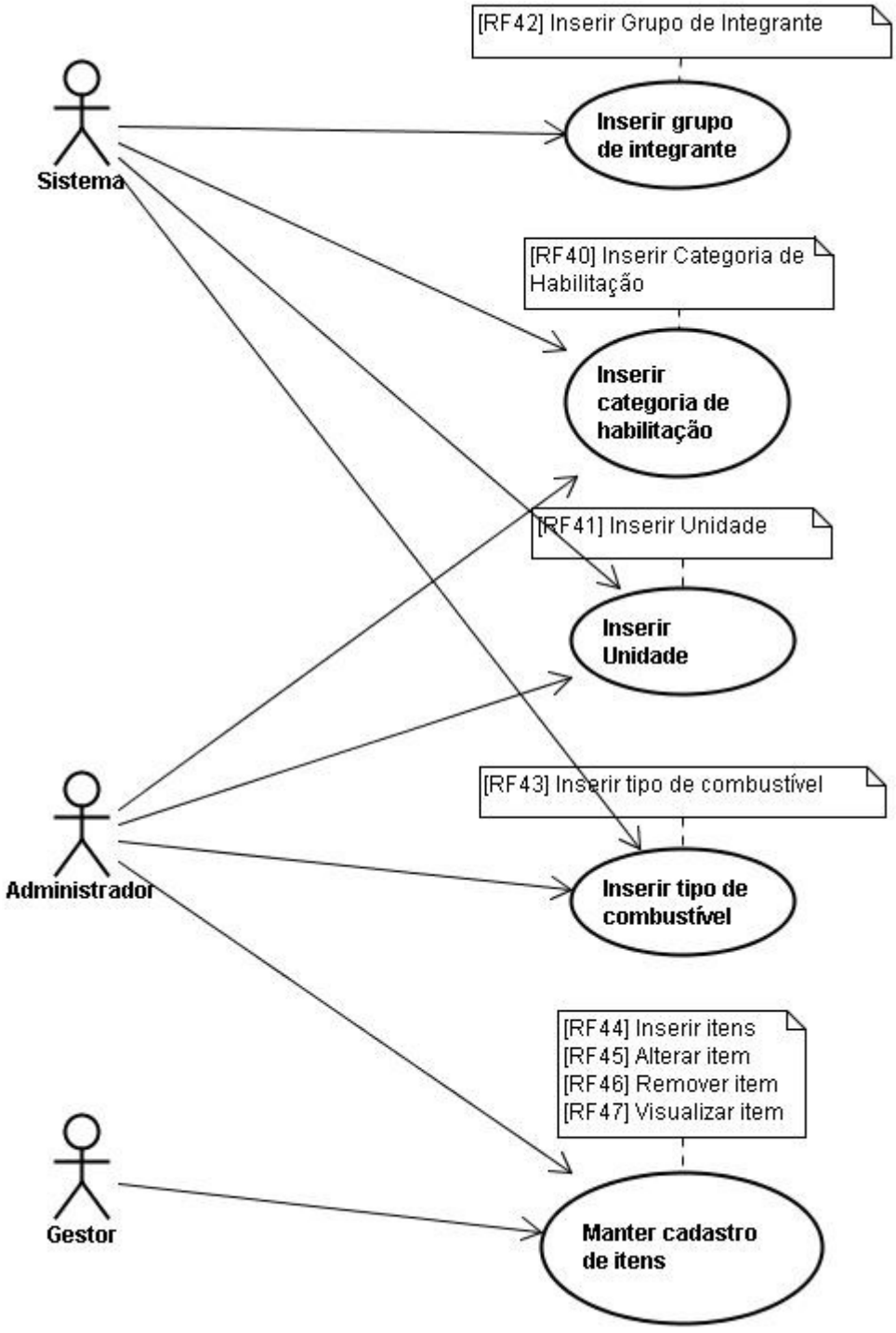


Figura 16 - Requisitos de Sistema

9 ANEXOIII – DIAGRAMA DE CLASSES

9.1 Diagrama de Classes 1

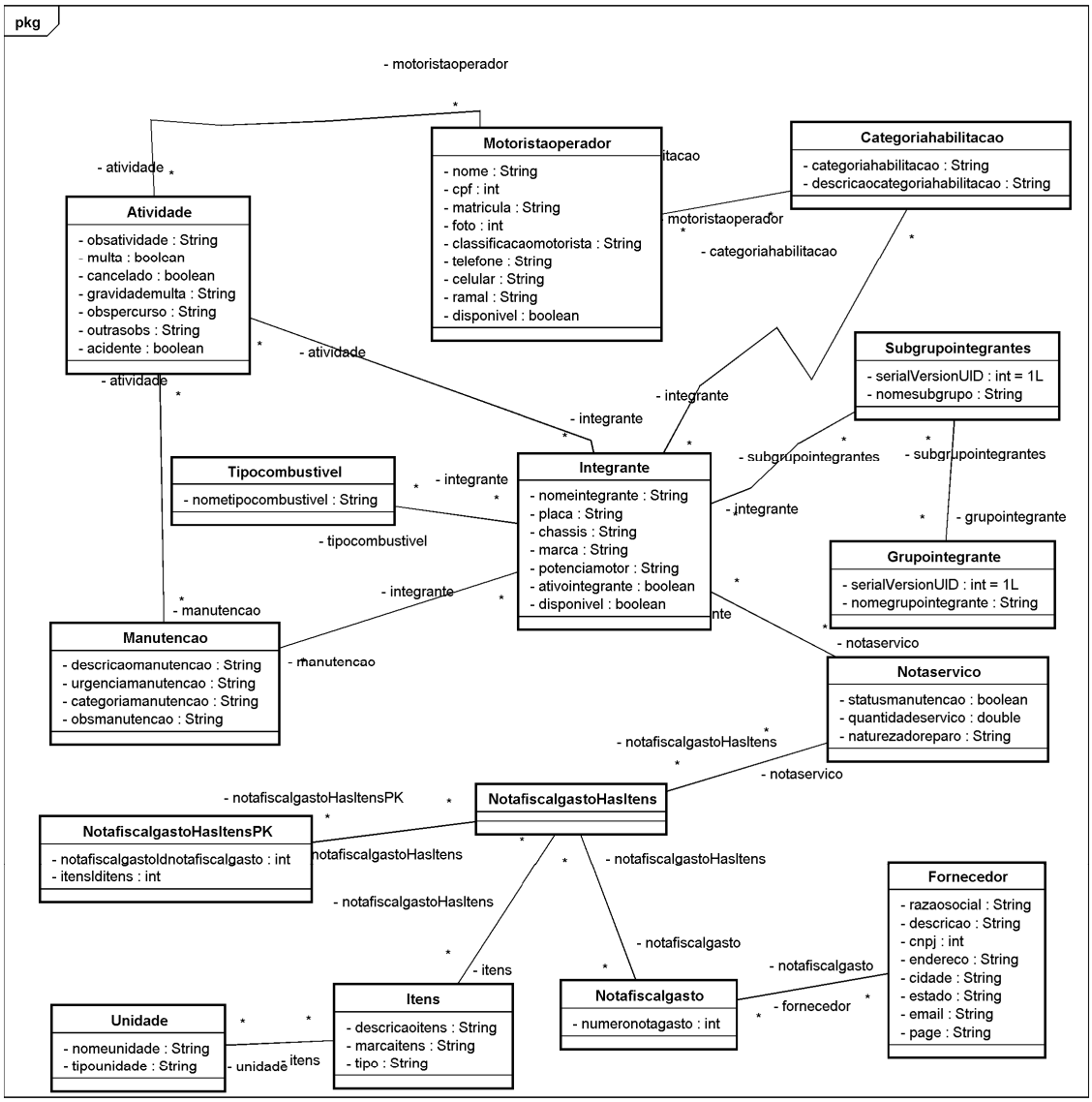


Figura 17 - Diagrama de Classes 1

9.2 Diagrama de Classes 2 – Usuário do Sistema e Perfil do Usuário

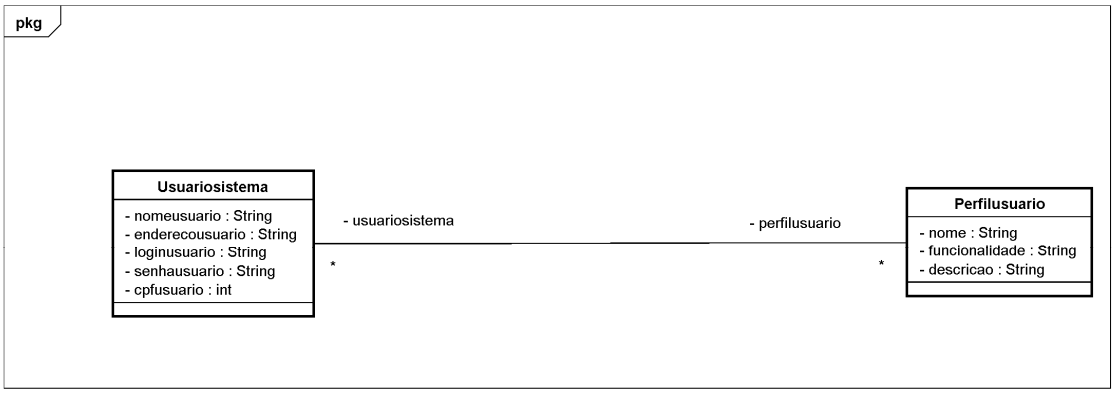


Figura 18- Diagrama de classes - Usuário do Sistema e Perfil do Usuário

10 ANEXO IV – MODELAGEM DO BANCO DE DADOS

10.1 Usuário do Sistema – Perfil do Usuário

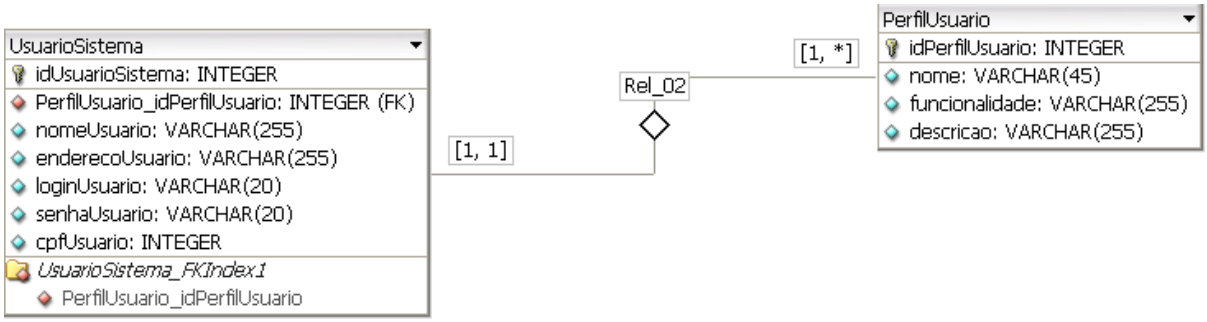


Figura 19 Modelagem do Banco de Dados: Usuário Sistema – Perfil Usuário

10.2 Fornecedor, Nota Fiscal de Gasto, Unidade, Itens, Associação Itens x Nota Fiscal

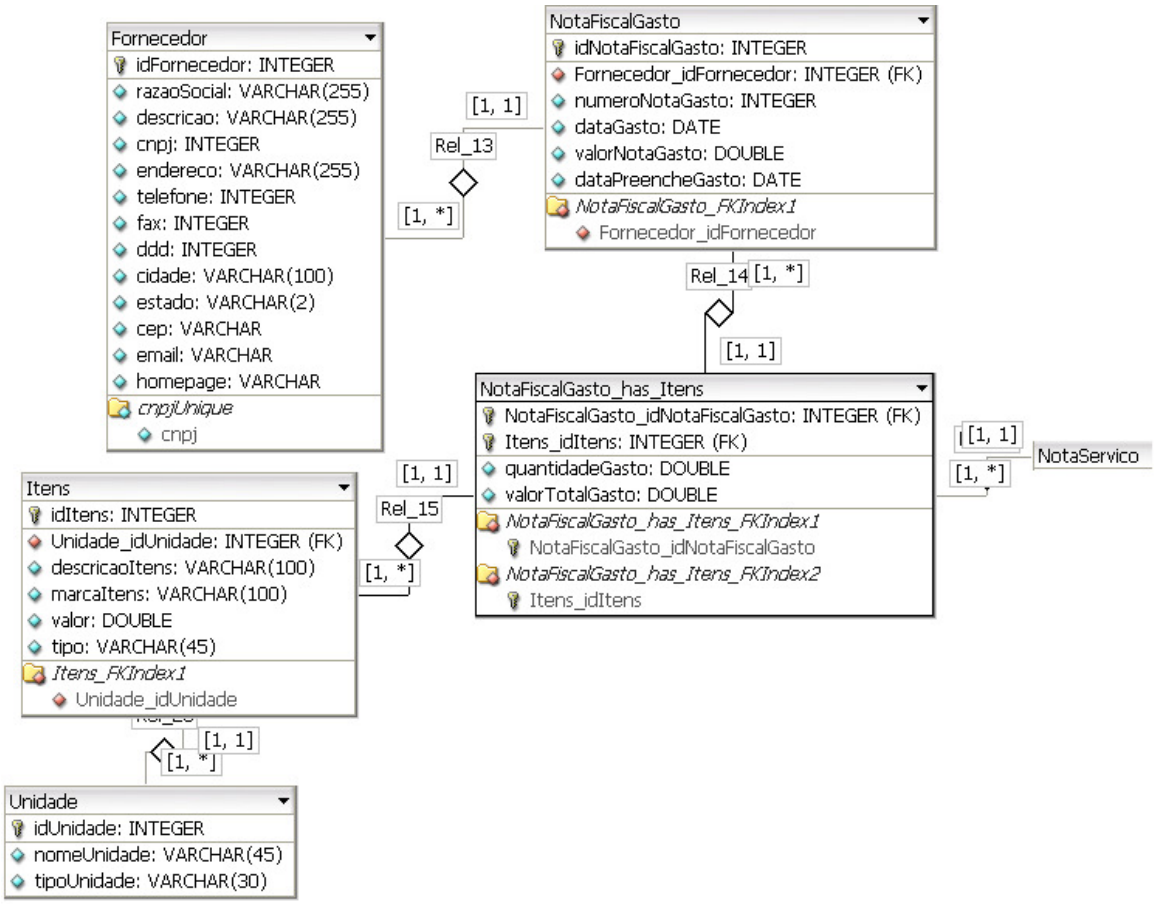


Figura 20 Modelagem do Banco de Dados: Fornecedor, Nota Fiscal de Gasto, Unidade, Itens, Associação Itens x Nota Fiscal

10.3 Nota Serviço

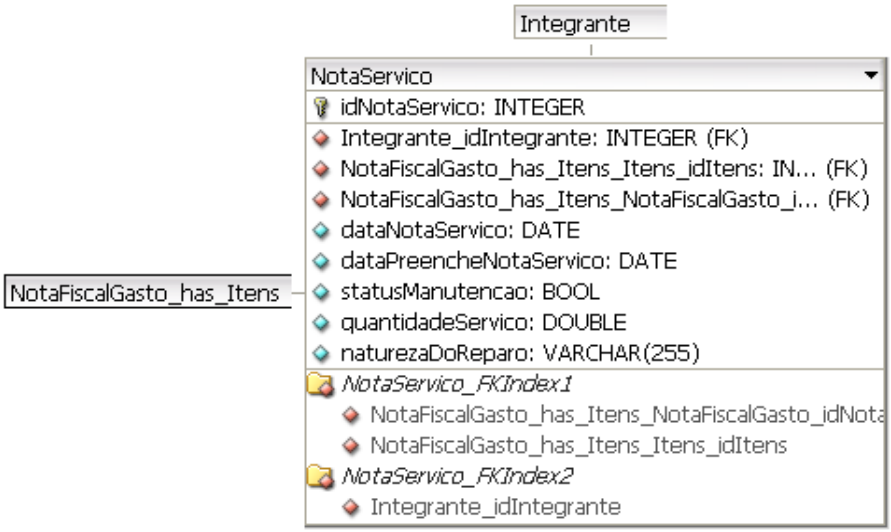


Figura 21 Modelagem do Banco de Dados: Nota Serviço

10.4 SubGrupo Integrantes, Grupo Integrantes e Integrantes

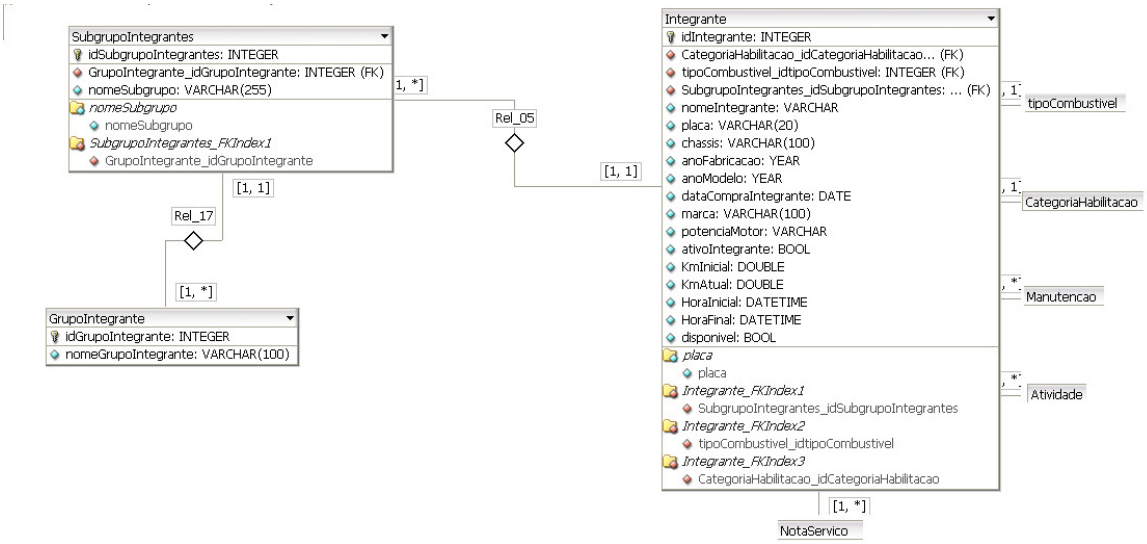


Figura 22 - Modelagem do Banco de Dados: SubGrupo Integrantes, Grupo Integrantes e Integrantes

10.5 Motorista/Operador, tipo de combustível, Categoria de Habilitação e Manutenção

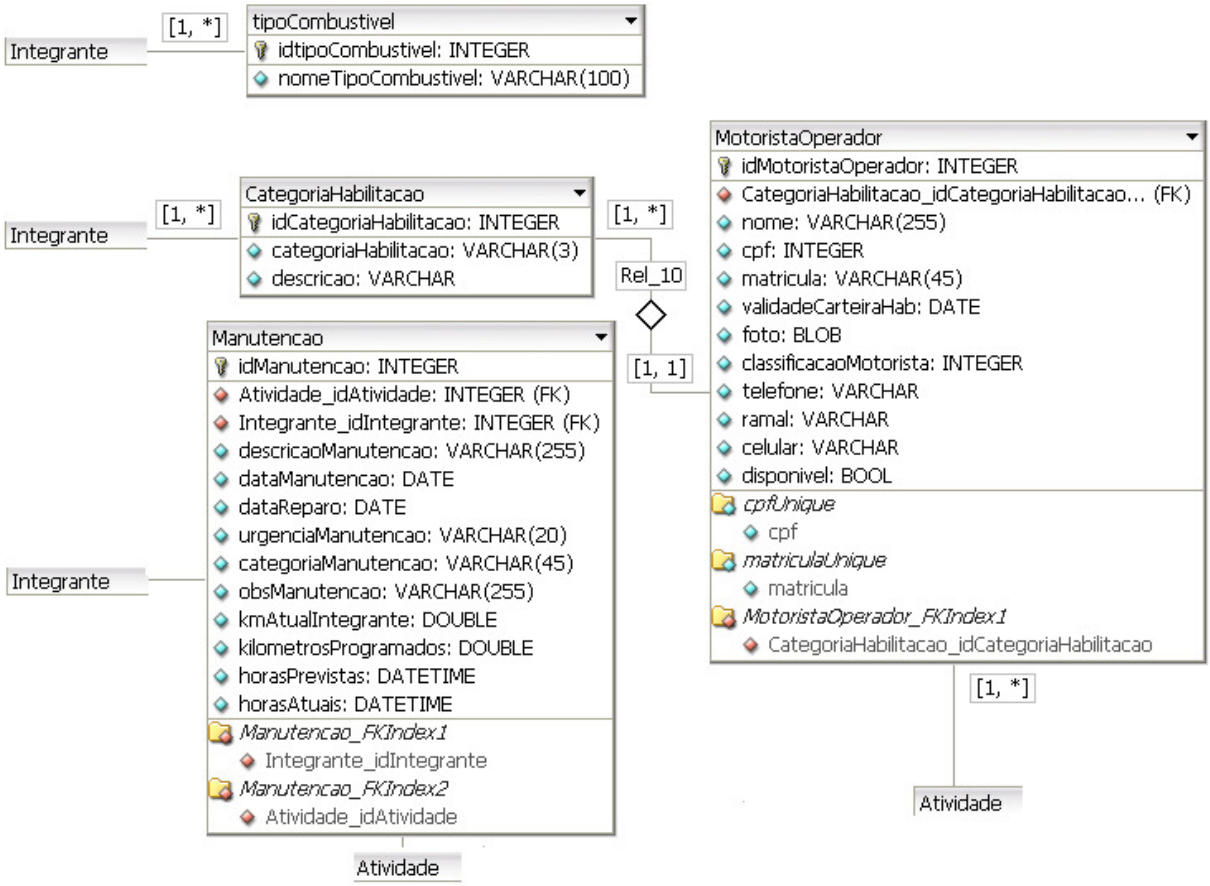


Figura 23 Modelagem do Banco de Dados: Motorista/Operador, tipo de combustível, Categoria de Habilitação e Manutenção

10.6 Atividade

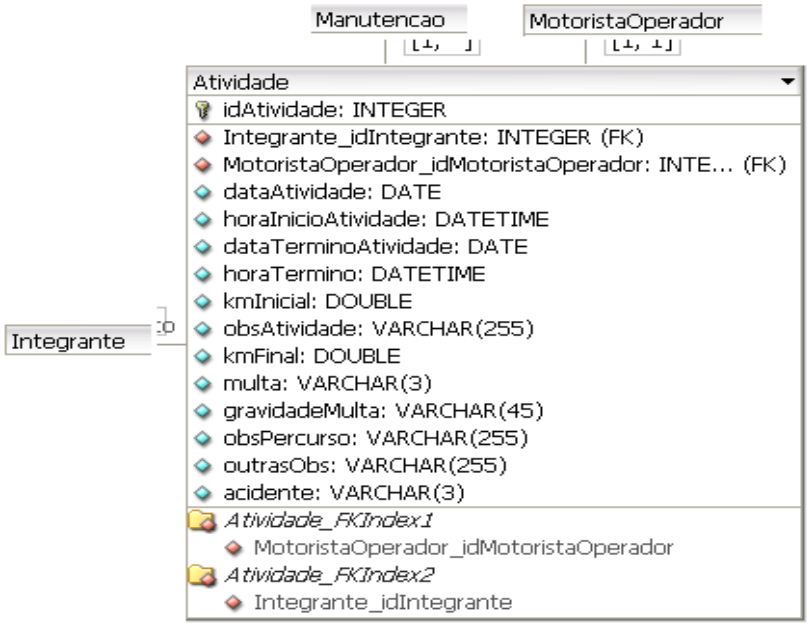


Figura 24 - Modelagem do Banco de Dados: Atividade

11 ANEXO V – CÓDIGO FONTE

11.1 Estrutura do Código Fonte:

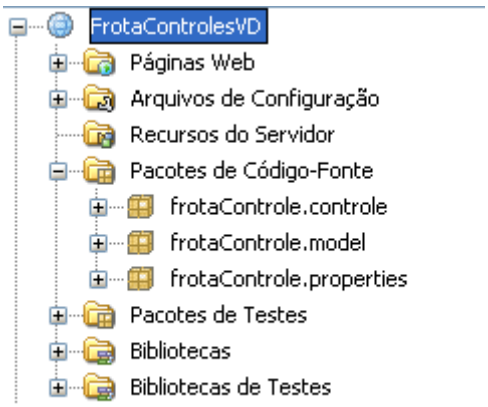


Figura 25 - Estrutura do Código Fonte

Nosso código fonte está organizado em uma estrutura MVC, onde temos a área "VIEW" definida nas Páginas Web, a área de "Controller" definida no pacote frotaControle.controle e a área "Model" definida no pacote frotaControle.model. No pacote frotaControle.properties definidos os arquivos de internacionalização e mensagens do sistema. Na pasta arquivos de configuração, temos arquivos que definem arquivos XML de acesso a banco de dados e configuração do servidor de aplicações. A pasta biblioteca define as bibliotecas utilizadas pelo sistema.

11.2 Páginas WEB

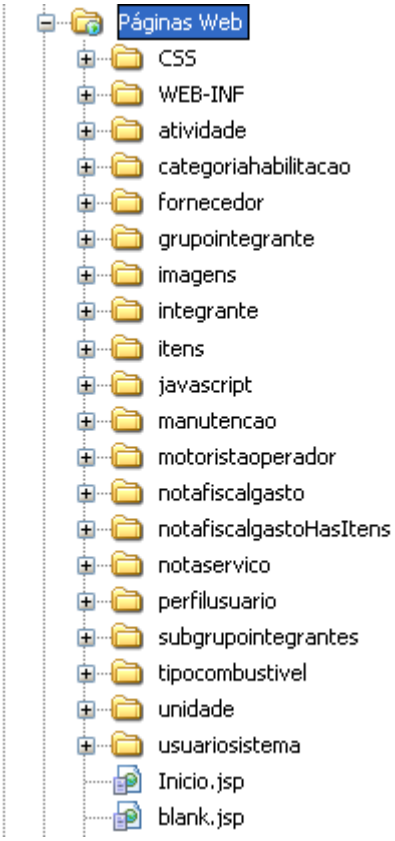


Figura 26 - Páginas Web

As páginas Web foram divididas em pastas, onde cada pasta corresponde a interface de apresentação das funcionalidades de cada classe. Temos como exceção as pastas CSS, que define os estilos da nossa página e WEB-INF que define arquivos de configuração da nossa aplicação.

11.2.1 Inicio.jsp – Arquivo de início do sistema

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@ taglib uri="http://myfaces.apache.org/tomahawk" prefix="t" %>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">

        <script type="text/javascript" src="javascript/scriptaculous.js"
></script>
        <script type="text/javascript"
src="javascript/prototype.js"></script>
        <script type="text/javascript"
src="javascript/effects.js"></script>
        <script type="text/javascript"
src="javascript/validacao.js"></script>
        <script type="text/javascript"
src="faces/javascript/paginacao.js"></script>
        <script type="text/javascript"
src="faces/javascript/modal.js"></script>
        <title>Controle de Frotas</title>
        <%@include file="WEB-INF/jspf/cabecalho.jspf"%>
    </head>
    <body>
        <f:view>
            <div id="corpo">


                <h:form>
                    <a4j:region id="tabela" renderRegionOnly="true" >
                <div id="textocomponente" ></div>
                <div id="logoprefeitura" ><p>logo da prefeitura</p></div>
                <div id="parceirosprefeitura" ><p>ÃƒÂ□rea destinada a
parceiros da prefeitura </p></div>
                <div id="parceiros" ></div>
                 <div id="manipulacaodadoscomponente">

                    <div id="teste" align="center" >
                        <a4j:outputPanel id="panel">

                        </a4j:outputPanel>
                    </div>
                    <div id="teste2" align="center" ></div>
                    <div id="mod" align="center"></div>
                </div>
                <div id="menusuperior" >
```

```
                        <rich:toolBar id="barraPrincipal">

                            <rich:dropDownMenu value="Sistema"
rendered="#{usuariosistema.estaLogado}"  id="sistema">

                                <rich:menuItem
ignoreDupResponses="true" value="Usuarios do Sistema"
action="usuariosistema_list" reRender="_id0:teste">
                                </rich:menuItem>
                                <rich:menuItem
ignoreDupResponses="true" value="Perfil do Usuario"
onclick="navega('faces/perfilusuario/List.jsp')">
                                </rich:menuItem>
                            </rich:dropDownMenu>
                            <rich:dropDownMenu value="Cadastro"
rendered="#{usuariosistema.estaLogado}" id="cadastro">
                                <rich:menuGroup value="Frota">
                                    <rich:menuItem
ignoreDupResponses="true" value="Veiculo"
onclick="navega('faces/integrante/List.jsp')">
                                    </rich:menuItem>
                                    <rich:menuItem
ignoreDupResponses="true" value="Tipo de Veiculo"
onclick="navega('faces/subgrupointegrantes/List.jsp')">
                                    </rich:menuItem>
                                    <rich:menuItem
ignoreDupResponses="true" value="Grupo do Veiculo"
onclick="navega('faces/grupointegrante/List.jsp')">
                                    </rich:menuItem>
                                    <rich:menuItem
ignoreDupResponses="true" value="Tipo de CombustÃvel"
onclick="navega('faces/tipocombustivel/List.jsp')">
                                    </rich:menuItem>

                                </rich:menuGroup>

                                <rich:menuItem
ignoreDupResponses="true" value="Motorista"
onclick="navega('faces/motoristaoperador/List.jsp')" >
                                </rich:menuItem>
                                <rich:menuItem
ignoreDupResponses="true" value="Habilitacao"
onclick="navega('faces/categoriahabilitacao/List.jsp')">
                                </rich:menuItem>
                                <rich:menuGroup value="Compras">
                                    <rich:menuItem
ignoreDupResponses="true" value="Fornecedor"
onclick="navega('faces/fornecedor/List.jsp')" >
                                    </rich:menuItem>
                                    <rich:menuItem
ignoreDupResponses="true" value="Itens de Compra"
onclick="navega('faces/itens/List.jsp')" >
                                    </rich:menuItem>
```

```
                                    </rich:menuGroup>
                                </rich:dropDownMenu>
                                <rich:dropDownMenu value="Atividade"
rendered="#{usuariosistema.estaLogado}" id="atividade">
                                    <rich:menuItem
ignoreDupResponses="true" value="Iniciar Atividade"
onclick="navega('faces/atividade/iniciarAtividade.jsp')" >
                                    </rich:menuItem>
                                    <rich:menuItem
ignoreDupResponses="true" value="Finalizar Atividade"
onclick="navega('faces/atividade/List.jsp')">
                                    </rich:menuItem>
                                </rich:dropDownMenu>
                                <rich:dropDownMenu
value="Manutencao"rendered="#{usuariosistema.estaLogado}" id="manutencao">

                                    <rich:menuItem
ignoreDupResponses="true" value="Manutencao"
onclick="navega('faces/manutencao/List.jsp')" >
                                    </rich:menuItem>
                                    <rich:menuItem
ignoreDupResponses="true" value="Nota Fiscal"
onclick="navega('faces/notafiscalgasto/List.jsp')" >
                                    </rich:menuItem>

                                </rich:dropDownMenu>


                                <h:panelGrid id="login" columns="6"
rendered="#{not usuariosistema.estaLogado}">

                                    <h:outputText value="UsuÃ¡rio: "/>
                                    <h:inputText id="usuario"
onfocus="login()" value="#{usuariosistema.login}"/>
                                    <h:outputText value="Senha: "/>
                                    <h:inputSecret id="senha"
onkeyup="login()" value="#{usuariosistema.senha}"/>

                                    <h:commandButton id="btnEntrar"
title="Fazer o Login" action="#{usuariosistema.loginAction}"
value="Entrar"/>


                                </h:panelGrid>
                                <h:panelGrid columns="5" id="logoff"
rendered="#{usuariosistema.estaLogado}" style="margin-left:250px">
                                    <h:outputText id="textLogin"
value="UsuÃ¡rio:    "/>
                                    <h:outputText
value="#{usuariosistema.login}"/>
                                    <a4j:commandLink value="Sair"
reRender="teste,barraPrincipal" action="#{usuariosistema.sair}"/>
                                </h:panelGrid>

                                <a4j:outputPanel ajaxRendered="true">
```

```
                                    <h:messages id='error' tooltip="true"
errorStyle="color: red" layout="table"/>
                                </a4j:outputPanel>
                            </rich:toolBar>
                            <a4j:status for="tabela">
                                <f:facet name="start">
                                    <h:graphicImage style="margin-left:
400px;margin-top: 100px" value="faces/imagens/seta_001.gif"/>
                                </f:facet>
                            </a4j:status>

                        </a4j:region>

                    </h:form>
                </div>

            </div>
        </f:view>
    </body>
</html>
```

11.2.2 Categoria Habilitação/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
    </head>
    <body>
        <f:view>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

            <h:form>

                <h3>Cadastro da Categoria de HabilitaÃ§Ã£o</h3>

                <rich:dataTable id='tabelaPerfil'
value='#{categoriahabilitacao.categoriahabilitacaos}' rows="10"
rowKeyVar="rk"  var='item' border="1" cellpadding="2" cellspacing="0"
onRowMouseOver="this.style.backgroundColor='#FFFFFF';this.style.fontWeight=
'bold'"
onRowMouseOut="this.style.backgroundColor='#{a4jSkin.tableBackgroundColor}'
;this.style.fontWeight='normal';">
            <h:message for="tabelaPerfil" tooltip="true" errorStyle="color:
red" infoStyle="color: green" />
```

```
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Categoriahabilitacao" />
                        </f:facet>
                        <h:outputText
value="#{item.categoriahabilitacao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="Descricaocategoriahabilitacao"/>
                        </f:facet>
                        <h:outputText
value="#{item.descricaocategoriahabilitacao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{categoriahabilitacao.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                        </f:facet>

                        <a4j:commandLink value="#{bundle.excluir}"
immediate="true" actionListener="#{categoriahabilitacao.editAction}"
bypassUpdates="true" reRender="modalExcluir" oncomplete="modalExcluir()">
                        </a4j:commandLink>

                    </h:column>
                    <h:column>
                        <f:facet name="header">
                           <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{categoriahabilitacao.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">

                        </a4j:commandLink>
                    </h:column>
                </rich:dataTable>
                <rich:datascroller id="scroller" for="tabelaPerfil">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>
```

```
            </h:form>

            <rich:modalPanel id='modalAjuda' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="faces/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasperfildousuario}"
/>
                </f:facet>
                <h:graphicImage value="/faces//imagens/interrogacao.jpg" />
            </rich:modalPanel>


            <rich:modalPanel id='modalSalvar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2">
                        <h:outputText value="Categoria"/>
                        <h:inputText
value="#{categoriahabilitacao.categoriahabilitacao.categoriahabilitacao}"
tabindex="1"/>
                        <h:outputText value="Descricao"/>
                        <h:inputText
value="#{categoriahabilitacao.categoriahabilitacao.descricaocategoriahabili
tacao}" tabindex="2"/>
                        <a4j:commandButton id="saveEdit"
action="#{categoriahabilitacao.create}"
reRender="tabelaPerfil,scroller"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idcategoriahabilitacao"
value="#{categoriahabilitacao.categoriahabilitacao.idcategoriahabilitacao}"
/>
                        </a4j:commandButton>
                        <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>

            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
```

```
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2">
                        <h:outputText value="Categoria"/>
                        <h:inputText
value="#{categoriahabilitacao.categoriahabilitacao.categoriahabilitacao}"
tabindex="1" />
                        <h:outputText value="Descricao"/>
                        <h:inputText
value="#{categoriahabilitacao.categoriahabilitacao.descricaocategoriahabili
tacao}" tabindex="2" />
                        <a4j:commandButton id="saveEdit"
action="#{categoriahabilitacao.edit}"
reRender="tabelaPerfil,scroller"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="idcategoriahabilitacao"
value="#{categoriahabilitacao.categoriahabilitacao.idcategoriahabilitacao}"
/>
                        </a4j:commandButton>
                        <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalEditar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>


            <rich:modalPanel id='modalExcluir' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalExcluir" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2">
                        <h:outputText value="Categoria"/>
                        <h:outputText
value="#{categoriahabilitacao.categoriahabilitacao.categoriahabilitacao}"/>
                        <h:outputText value="Descricao"/>
                        <h:outputText
value="#{categoriahabilitacao.categoriahabilitacao.descricaocategoriahabili
tacao}"/>
                        <a4j:commandButton id="dbExcluir"
action="#{categoriahabilitacao.destroy}"
```

```
reRender="tabelaPerfil,scroller"value="#{bundle.excluir}"
oncomplete="Richfaces.hideModalPanel('modalExcluir')">
                              <a4j:actionparam name="idcategoriahabilitacao"
value="#{categoriahabilitacao.categoriahabilitacao.idcategoriahabilitacao}"
/>
                         </a4j:commandButton>
                         <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalExcluir')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>


        </f:view>
    </body>


      </html>
```

11.2.3 Fornecedor/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <SCRIPT LANGUAGE="Javascript">
function modal()
{
Richfaces.showModalPanel('mp',{width:450, top:200});

}
        </SCRIPT>
    </head>
    <body style="text-align:center">
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
             <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
         <h:form id="formInicial">

             <h3>Cadastro de Fornecedores</h3>

             <rich:dataTable value='#{fornecedor.fornecedors}'
id="tabelaPerfil" rows="10"  rowKeyVar="rk" var='item' border="1"
cellpadding="2" cellspacing="0">
                    <h:column>
```

```
                        <f:facet name="header">
                            <h:outputText value="#{bundle.razaosocial}"/>
                        </f:facet>
                        <a4j:commandLink value="#{item.razaosocial}"
immediate="true" actionListener="#{fornecedor.detailAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.descricao}"/>
                        </f:facet>
                        <h:outputText value="#{item.descricao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.cnpj}"/>
                        </f:facet>
                        <h:outputText value="#{item.cnpj}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.endereco}"/>
                        </f:facet>
                        <h:outputText value="#{item.endereco}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.telefone}"/>
                        </f:facet>
                        <h:outputText value="#{item.telefone}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.fax}"/>
                        </f:facet>
                        <h:outputText value="#{item.fax}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.ddd}"/>
                        </f:facet>
                        <h:outputText value="#{item.ddd}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.cidade}"/>
                        </f:facet>
                        <h:outputText value="#{item.cidade}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.estado}"/>
                        </f:facet>
                        <h:outputText value="#{item.estado}"/>
                    </h:column>
                    <h:column>
```

```
                        <f:facet name="header">
                            <h:outputText value="#{bundle.email}"/>
                        </f:facet>
                        <h:outputText value="#{item.email}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.homepage}"/>
                        </f:facet>
                        <h:outputText value="#{item.page}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{fornecedor.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                        </f:facet>
                        <h:commandLink value="#{bundle.excluir}"
action="#{fornecedor.destroy}">
                            <f:param name="idperfilusuario"
value="#{item.idfornecedor}"/>
                            </h:commandLink>

                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{fornecedor.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">

                        </a4j:commandLink>
                    </h:column>
                </rich:dataTable>

                <rich:datascroller id="scroller" for="tabelaPerfil">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>
            </h:form>

            <rich:modalPanel id='modalAjuda' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
```

```
                <f:facet name="controls">
                    <h:graphicImage value="/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasperfildousuario}"
/>
                </f:facet>
                <h:graphicImage value="/imagens/interrogacao.jpg" />
            </rich:modalPanel>


            <rich:modalPanel id='modalSalvar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
                <f:facet name="header">
                    <h:outputText value="#{bundle.cadastrofornecedor}" />
                </f:facet>

                <h:form>
                    <h:panelGrid columns="2">
                        <h:outputText value="#{bundle.razaosocial}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.razaosocial}"
readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.descricao}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.descricao}"
readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.cnpj}"/>
                        <h:inputText  value="#{fornecedor.fornecedor.cnpj}"
readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.ddd}"/>
                        <h:inputText  value="#{fornecedor.fornecedor.ddd}"
readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.telefone}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.telefone}"readonly="#{usuariosistema.input}"
/>
                        <h:outputText value="#{bundle.fax}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.fax}"readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.endereco}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.endereco}"
readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.cidade}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.cidade}" readonly="#{usuariosistema.input}"
/>
                        <h:outputText value="#{bundle.estado}"/>
```

```
                        <h:inputText
value="#{fornecedor.fornecedor.estado}" readonly="#{usuariosistema.input}"
/>
                        <a4j:commandButton id="saveEdit"
action="#{fornecedor.create}" reRender="tabelaPerfil"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idperfilusuario"
value="#{usuariosistema.usuariosistema.idusuariosistema}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>


            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
                <f:facet name="header">
                    <h:outputText value="#{bundle.cadastrofornecedor}" />
                </f:facet>

                <h:form>
                    <h:panelGrid columns="2">
                        <h:outputText value="#{bundle.razaosocial}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.razaosocial}"/>
                        <h:outputText value="#{bundle.descricao}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.descricao}"/>
                        <h:outputText value="#{bundle.cnpj}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.cnpj}"/>
                        <h:outputText value="#{bundle.ddd}"/>
                        <h:inputText  value="#{fornecedor.fornecedor.ddd}"
/>
                        <h:outputText value="#{bundle.telefone}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.telefone}"/>
                        <h:outputText value="#{bundle.fax}"/>
                        <h:inputText value="#{fornecedor.fornecedor.fax}"/>
                        <h:outputText value="#{bundle.endereco}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.endereco}"/>
                        <h:outputText value="#{bundle.cidade}"/>
                        <h:inputText
value="#{fornecedor.fornecedor.cidade}"/>
                        <h:outputText value="#{bundle.estado}"/>
```

```
                        <h:inputText
value="#{fornecedor.fornecedor.estado}"/>
                        <a4j:commandButton id="saveEdit"
action="#{fornecedor.create}" reRender="tabelaPerfil"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="idperfilusuario"
value="#{usuariosistema.usuariosistema.idusuariosistema}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalEditar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>

        </f:view>
    </body>
</html>
```

11.2.4 Grupo Integrante/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
    <body style="text-align:center">
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
                        <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <a4j:loadBundle var="mensagem"
basename="frotaControle.properties.mensagens"/>

            <h:form>
                                    <h3>Cadastro de Grupos de VeÃculos</h3>

                  <rich:dataTable id='tabelaPerfil' rendered="true"
value='#{grupointegrante.grupointegrantes}' rows="10" rowKeyVar="rk"
var='item' border="1" cellpadding="2" cellspacing="0">
                    <h:message for="tabelaPerfil" tooltip="true"
errorStyle="color: red" infoStyle="color: green" />
                      <rich:column>
                        <f:facet name="header">
                        <h:outputText
value="#{bundle.nomegrupointegrante}"/>
```

```
                            </f:facet>
                        <h:outputText value="#{item.nomegrupointegrante}"
/>

                    </rich:column>

                        <rich:column>
                            <f:facet name="header">
                                <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{grupointegrante.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                            </f:facet>

                            <h:commandLink value="#{bundle.excluir}"
action="#{grupointegrante.destroy}">
                                <f:param name="idgrupointegrante"
value="#{item.idgrupointegrante}"/>
                            </h:commandLink>

                        </rich:column>
                        <rich:column>
                            <f:facet name="header">
                                <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                            </f:facet>
                            <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{grupointegrante.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">

                            </a4j:commandLink>
                        </rich:column>
                    </rich:dataTable>
                    <rich:datascroller id="scroller" for="tabelaPerfil"
align="center">
                        <f:facet name="first">
                            <h:outputText value="#{bundle.primeiro}"/>
                        </f:facet>
                        <f:facet name="last">
                            <h:outputText value="#{bundle.ultimo}"/>
                        </f:facet>
                        <f:facet name="next">
                            <h:outputText value="#{bundle.proximo}"/>
                        </f:facet>
                        <f:facet name="previous">
                            <h:outputText value="#{bundle.anterior}"/>
                        </f:facet>
                    </rich:datascroller>

            </h:form>

                        <rich:modalPanel id='modalAjuda' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
```

```
                    <h:graphicImage value="faces/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasusuario}" />
                </f:facet>
                <h:graphicImage value="faces/imagens/interrogacao.jpg" />
            </rich:modalPanel>

            <rich:modalPanel id='modalSalvar' height="300" minWidth="450"
                             zindex="2000" resizeable="true">
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.usuario}" />
                </f:facet>
                <h:form  style="text-align:center">
                    <h:panelGrid columns="2" >
                        <h:outputText
value="#{bundle.nomegrupointegrante}"/>

                        <h:inputText
value="#{grupointegrante.grupointegrante.nomegrupointegrante}" />

                        <a4j:commandButton id="saveEdit"
action="#{grupointegrante.create}" reRender="tabelaPerfil"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idgrupointegrante"
value="#{grupointegrante.grupointegrante.idgrupointegrante}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>

                </h:form>
            </rich:modalPanel>
            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2" >
```

```
                        <h:outputText
value="#{bundle.nomegrupointegrante}"/>

                        <h:inputText
value="#{grupointegrante.grupointegrante.nomegrupointegrante}" />


                        <a4j:commandButton id="saveEdit"
action="#{grupointegrante.edit}" reRender="tabelaPerfil"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="idgrupointegrante"
value="#{grupointegrante.grupointegrante.idgrupointegrante}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalEditar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>

        </f:view>
    </body>
</html>
```

11.2.5 Integrante/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>List Integrante</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Listing Integrantes</h1>
            <h:form>
                <h:commandLink action="#{integrante.createSetup}"
value="New Integrante"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
                <br>
                <h:outputText value="Item #{integrante.firstItem +
1}..#{integrante.lastItem} of #{integrante.itemCount}"/> 
                <h:commandLink action="#{integrante.prev}" value="Previous
#{integrante.batchSize}" rendered="#{integrante.firstItem >=
integrante.batchSize}"/> 
```

```
                <h:commandLink action="#{integrante.next}" value="Next
#{integrante.batchSize}" rendered="#{integrante.lastItem +
integrante.batchSize <= integrante.itemCount}"/> 
                <h:commandLink action="#{integrante.next}" value="Remaining
#{integrante.itemCount - integrante.lastItem}"
                               rendered="#{integrante.lastItem <
integrante.itemCount && integrante.lastItem + integrante.batchSize >
integrante.itemCount}"/>
                <h:dataTable value='#{integrante.integrantes}' var='item'
border="1" cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Idintegrante"/>
                        </f:facet>
                        <h:commandLink action="#{integrante.detailSetup}"
value="#{item.idintegrante}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Nomeintegrante"/>
                        </f:facet>
                        <h:outputText value="#{item.nomeintegrante}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Placa"/>
                        </f:facet>
                        <h:outputText value="#{item.placa}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Chassis"/>
                        </f:facet>
                        <h:outputText value="#{item.chassis}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Anofabricacao"/>
                        </f:facet>
                        <h:outputText value="#{item.anofabricacao}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Anomodelo"/>
                        </f:facet>
                        <h:outputText value="#{item.anomodelo}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datacompraintegrante"/>
```

```
                        </f:facet>
                        <h:outputText value="#{item.datacompraintegrante}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Marca"/>
                        </f:facet>
                        <h:outputText value="#{item.marca}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Potenciamotor"/>
                        </f:facet>
                        <h:outputText value="#{item.potenciamotor}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Ativointegrante"/>
                        </f:facet>
                        <h:outputText value="#{item.ativointegrante}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Kminicial"/>
                        </f:facet>
                        <h:outputText value="#{item.kminicial}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Kmatual"/>
                        </f:facet>
                        <h:outputText value="#{item.kmatual}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Horainicial"/>
                        </f:facet>
                        <h:outputText value="#{item.horainicial}">
                            <f:convertDateTime type="TIME"
pattern="hh:mm:ss" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Horafinal"/>
                        </f:facet>
                        <h:outputText value="#{item.horafinal}">
                            <f:convertDateTime type="TIME"
pattern="hh:mm:ss" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
```

```
                            <h:outputText
value="CategoriahabilitacaoIdcategoriahabilitacao"/>
                        </f:facet>
                        <h:outputText
value="#{item.categoriahabilitacaoIdcategoriahabilitacao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="SubgrupointegrantesIdsubgrupointegrantes"/>
                        </f:facet>
                        <h:outputText
value="#{item.subgrupointegrantesIdsubgrupointegrantes}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="TipocombustivelIdtipocombustivel"/>
                        </f:facet>
                        <h:outputText
value="#{item.tipocombustivelIdtipocombustivel}"/>
                    </h:column>
                    <h:column>
                        <h:commandLink value="Destroy"
action="#{integrante.destroy}">
                            <f:param name="idintegrante"
value="#{item.idintegrante}"/>
                        </h:commandLink>
                        <h:outputText value=" "/>
                        <h:commandLink value="Edit"
action="#{integrante.editSetup}">
                            <f:param name="idintegrante"
value="#{item.idintegrante}"/>
                        </h:commandLink>
                    </h:column>
                </h:dataTable>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.6 Integrante/Detail.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Detail of Integrante</title>
    </head>
    <body>
        <f:view>
```

```
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Detail of integrante</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idintegrante:"/>
                    <h:outputText
value="#{integrante.integrante.idintegrante}" title="Idintegrante" />
                    <h:outputText value="Nomeintegrante:"/>
                    <h:outputText
value="#{integrante.integrante.nomeintegrante}" title="Nomeintegrante" />
                    <h:outputText value="Placa:"/>
                    <h:outputText value="#{integrante.integrante.placa}"
title="Placa" />
                    <h:outputText value="Chassis:"/>
                    <h:outputText value="#{integrante.integrante.chassis}"
title="Chassis" />
                    <h:outputText value="Anofabricacao:"/>
                    <h:outputText
value="#{integrante.integrante.anofabricacao}" title="Anofabricacao" />
                    <h:outputText value="Anomodelo:"/>
                    <h:outputText
value="#{integrante.integrante.anomodelo}" title="Anomodelo" />
                    <h:outputText value="Datacompraintegrante:"/>
                    <h:outputText
value="#{integrante.integrante.datacompraintegrante}"
title="Datacompraintegrante" />
                    <h:outputText value="Marca:"/>
                    <h:outputText value="#{integrante.integrante.marca}"
title="Marca" />
                    <h:outputText value="Potenciamotor:"/>
                    <h:outputText
value="#{integrante.integrante.potenciamotor}" title="Potenciamotor" />
                    <h:outputText value="Ativointegrante:"/>
                    <h:outputText
value="#{integrante.integrante.ativointegrante}" title="Ativointegrante" />
                    <h:outputText value="Kminicial:"/>
                    <h:outputText
value="#{integrante.integrante.kminicial}" title="Kminicial" />
                    <h:outputText value="Kmatual:"/>
                    <h:outputText value="#{integrante.integrante.kmatual}"
title="Kmatual" />
                    <h:outputText value="Horainicial:"/>
                    <h:outputText
value="#{integrante.integrante.horainicial}" title="Horainicial" />
                    <h:outputText value="Horafinal:"/>
                    <h:outputText
value="#{integrante.integrante.horafinal}" title="Horafinal" />
                    <h:outputText
value="CategoriahabilitacaoIdcategoriahabilitacao:"/>
                    <h:outputText
value="#{integrante.integrante.categoriahabilitacaoIdcategoriahabilitacao}"
title="CategoriahabilitacaoIdcategoriahabilitacao" />
                    <h:outputText
value="SubgrupointegrantesIdsubgrupointegrantes:"/>
```

```
                    <h:outputText
value="#{integrante.integrante.subgrupointegrantesIdsubgrupointegrantes}"
title="SubgrupointegrantesIdsubgrupointegrantes" />
                    <h:outputText
value="TipocombustivelIdtipocombustivel:"/>
                    <h:outputText
value="#{integrante.integrante.tipocombustivelIdtipocombustivel}"
title="TipocombustivelIdtipocombustivel" />
                </h:panelGrid>
                <h:commandLink action="integrante_edit" value="Edit" />
                <br>
                <h:commandLink action="integrante_list" value="Show All
Integrante"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.7 Integrante/Edit.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Edit Integrante</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Edit integrante</h1>
            <h:form>
                <h:inputHidden value="#{integrante.integrante}"
immediate="true"/>
                <h:panelGrid columns="2">
                    <h:outputText value="Idintegrante:"/>
                    <h:outputText
value="#{integrante.integrante.idintegrante}" title="Idintegrante" />
                    <h:outputText value="Nomeintegrante:"/>
                    <h:inputText id="nomeintegrante"
value="#{integrante.integrante.nomeintegrante}" title="Nomeintegrante" />
                    <h:outputText value="Placa:"/>
                    <h:inputText id="placa"
value="#{integrante.integrante.placa}" title="Placa" />
                    <h:outputText value="Chassis:"/>
                    <h:inputText id="chassis"
value="#{integrante.integrante.chassis}" title="Chassis" />
                    <h:outputText value="Anofabricacao (MM/dd/yyyy):"/>
```

```
                    <h:inputText id="anofabricacao"
value="#{integrante.integrante.anofabricacao}" title="Anofabricacao" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Anomodelo (MM/dd/yyyy):"/>
                    <h:inputText id="anomodelo"
value="#{integrante.integrante.anomodelo}" title="Anomodelo" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Datacompraintegrante
(MM/dd/yyyy):"/>
                    <h:inputText id="datacompraintegrante"
value="#{integrante.integrante.datacompraintegrante}"
title="Datacompraintegrante" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Marca:"/>
                    <h:inputText id="marca"
value="#{integrante.integrante.marca}" title="Marca" />
                    <h:outputText value="Potenciamotor:"/>
                    <h:inputText id="potenciamotor"
value="#{integrante.integrante.potenciamotor}" title="Potenciamotor" />
                    <h:outputText value="Ativointegrante:"/>
                    <h:inputText id="ativointegrante"
value="#{integrante.integrante.ativointegrante}" title="Ativointegrante" />
                    <h:outputText value="Kminicial:"/>
                    <h:inputText id="kminicial"
value="#{integrante.integrante.kminicial}" title="Kminicial" />
                    <h:outputText value="Kmatual:"/>
                    <h:inputText id="kmatual"
value="#{integrante.integrante.kmatual}" title="Kmatual" />
                    <h:outputText value="Horainicial (hh:mm:ss):"/>
                    <h:inputText id="horainicial"
value="#{integrante.integrante.horainicial}" title="Horainicial" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText value="Horafinal (hh:mm:ss):"/>
                    <h:inputText id="horafinal"
value="#{integrante.integrante.horafinal}" title="Horafinal" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText
value="CategoriahabilitacaoIdcategoriahabilitacao:"/>
                    <h:selectOneMenu
id="categoriahabilitacaoIdcategoriahabilitacao"
value="#{integrante.integrante.categoriahabilitacaoIdcategoriahabilitacao}"
title="CategoriahabilitacaoIdcategoriahabilitacao">
                        <f:selectItems
value="#{integrante.categoriahabilitacaoIdcategoriahabilitacaos}"/>
                    </h:selectOneMenu>
```

```
                    <h:outputText
value="SubgrupointegrantesIdsubgrupointegrantes:"/>
                    <h:selectOneMenu
id="subgrupointegrantesIdsubgrupointegrantes"
value="#{integrante.integrante.subgrupointegrantesIdsubgrupointegrantes}"
title="SubgrupointegrantesIdsubgrupointegrantes">
                        <f:selectItems
value="#{integrante.subgrupointegrantesIdsubgrupointegrantess}"/>
                    </h:selectOneMenu>
                    <h:outputText
value="TipocombustivelIdtipocombustivel:"/>
                    <h:selectOneMenu id="tipocombustivelIdtipocombustivel"
value="#{integrante.integrante.tipocombustivelIdtipocombustivel}"
title="TipocombustivelIdtipocombustivel">
                        <f:selectItems
value="#{integrante.tipocombustivelIdtipocombustivels}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{integrante.edit}" value="Save"/>
                <br>
                <h:commandLink action="integrante_list" value="Show All
Integrante"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.8 Integrante/New.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>New Integrante</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>New integrante</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idintegrante:"/>
                    <h:inputText id="idintegrante"
value="#{integrante.integrante.idintegrante}" title="Idintegrante" />
                    <h:outputText value="Nomeintegrante:"/>
                    <h:inputText id="nomeintegrante"
value="#{integrante.integrante.nomeintegrante}" title="Nomeintegrante" />
                    <h:outputText value="Placa:"/>
```

```
                    <h:inputText id="placa"
value="#{integrante.integrante.placa}" title="Placa" />
                    <h:outputText value="Chassis:"/>
                    <h:inputText id="chassis"
value="#{integrante.integrante.chassis}" title="Chassis" />
                    <h:outputText value="Anofabricacao (MM/dd/yyyy):"/>
                    <h:inputText id="anofabricacao"
value="#{integrante.integrante.anofabricacao}" title="Anofabricacao" >
                        <!--f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                    </h:inputText>
                    <h:outputText value="Anomodelo (MM/dd/yyyy):"/>
                    <h:inputText id="anomodelo"
value="#{integrante.integrante.anomodelo}" title="Anomodelo" >
                        <!--f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                    </h:inputText>
                    <h:outputText value="Datacompraintegrante
(MM/dd/yyyy):"/>
                    <h:inputText id="datacompraintegrante"
value="#{integrante.integrante.datacompraintegrante}"
title="Datacompraintegrante" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Marca:"/>
                    <h:inputText id="marca"
value="#{integrante.integrante.marca}" title="Marca" />
                    <h:outputText value="Potenciamotor:"/>
                    <h:inputText id="potenciamotor"
value="#{integrante.integrante.potenciamotor}" title="Potenciamotor" />
                    <h:outputText value="Ativointegrante:"/>
                    <h:inputText id="ativointegrante"
value="#{integrante.integrante.ativointegrante}" title="Ativointegrante" />
                    <h:outputText value="Kminicial:"/>
                    <h:inputText id="kminicial"
value="#{integrante.integrante.kminicial}" title="Kminicial" />
                    <h:outputText value="Kmatual:"/>
                    <h:inputText id="kmatual"
value="#{integrante.integrante.kmatual}" title="Kmatual" />
                    <h:outputText value="Horainicial (hh:mm:ss):"/>
                    <h:inputText id="horainicial"
value="#{integrante.integrante.horainicial}" title="Horainicial" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText value="Horafinal (hh:mm:ss):"/>
                    <h:inputText id="horafinal"
value="#{integrante.integrante.horafinal}" title="Horafinal" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText
value="CategoriahabilitacaoIdcategoriahabilitacao:"
rendered="#{integrante.integrante.categoriahabilitacaoIdcategoriahabilitaca
o == null}"/>
```

```
                    <h:selectOneMenu
id="categoriahabilitacaoIdcategoriahabilitacao"
value="#{integrante.integrante.categoriahabilitacaoIdcategoriahabilitacao}"
title="CategoriahabilitacaoIdcategoriahabilitacao"
rendered="#{integrante.integrante.categoriahabilitacaoIdcategoriahabilitaca
o == null}">
                        <f:selectItems
value="#{integrante.categoriahabilitacaoIdcategoriahabilitacaos}"/>
                    </h:selectOneMenu>
                    <h:outputText
value="SubgrupointegrantesIdsubgrupointegrantes:"
rendered="#{integrante.integrante.subgrupointegrantesIdsubgrupointegrantes
== null}"/>
                    <h:selectOneMenu
id="subgrupointegrantesIdsubgrupointegrantes"
value="#{integrante.integrante.subgrupointegrantesIdsubgrupointegrantes}"
title="SubgrupointegrantesIdsubgrupointegrantes"
rendered="#{integrante.integrante.subgrupointegrantesIdsubgrupointegrantes
== null}">
                        <f:selectItems
value="#{integrante.subgrupointegrantesIdsubgrupointegrantess}"/>
                    </h:selectOneMenu>
                    <h:outputText value="TipocombustivelIdtipocombustivel:"
rendered="#{integrante.integrante.tipocombustivelIdtipocombustivel ==
null}"/>
                    <h:selectOneMenu id="tipocombustivelIdtipocombustivel"
value="#{integrante.integrante.tipocombustivelIdtipocombustivel}"
title="TipocombustivelIdtipocombustivel"
rendered="#{integrante.integrante.tipocombustivelIdtipocombustivel ==
null}">
                        <f:selectItems
value="#{integrante.tipocombustivelIdtipocombustivels}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{integrante.create}"
value="Create"/>
                <br>
                <h:commandLink action="integrante_list" value="Show All
Integrante"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.9 Itens/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>
```

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
      </head>
    <body style="text-align:center">

            <f:view>
                <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>

                <h:messages errorStyle="color: red" infoStyle="color:
green" layout="table"/>
                <h:form id="formInicial">

                    <h3>Cadastro de Itens de Compra</h3>

                    <rich:dataTable value='#{itens.itenss}'
id="tabelaPerfil" rows="10"  rowKeyVar="rk" var='item' border="1"
cellpadding="2" cellspacing="0">
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.descricao}"/>
                            </f:facet>
                            <a4j:commandLink value="#{item.descricaoitens}"
id="nome" immediate="true" actionListener="#{itens.detailAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.marca}"/>
                            </f:facet>
                            <h:outputText value="#{item.marcaitens}"/>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.valor}"/>
                            </f:facet>
                            <h:outputText value="#{item.valor}"/>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.tipo}"/>
                            </f:facet>
                            <h:outputText value="#{item.tipo}"/>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.unidade}"/>
                            </f:facet>
                            <h:selectOneMenu
value="#{item.unidadeIdunidade}" disabled="true" id="unidade"
readonly="true">
                                <f:selectItems
value="#{itens.unidadeIdunidades}"/>
```

```
                            </h:selectOneMenu>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{itens.newAction}" bypassUpdates="true"
reRender="modalSalvar" oncomplete="modalSalvar()"/>
                            </f:facet>
                        </h:column>
                        <h:column>
                            <h:commandLink value="#{bundle.excluir}"
action="#{itens.destroy}">
                        </h:commandLink>

                        </h:column>
                        <h:column>
                            <f:facet name="header">
                             <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                            </f:facet>
                            <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{itens.editAction}" bypassUpdates="true"
reRender="modalEditar" oncomplete="modalEditar()">
                            </a4j:commandLink>
                        </h:column>
                    </rich:dataTable>
                    <rich:datascroller id="scroller" for="tabelaPerfil">
                        <f:facet name="first">
                            <h:outputText value="#{bundle.primeiro}"/>
                        </f:facet>
                        <f:facet name="last">
                            <h:outputText value="#{bundle.ultimo}"/>
                        </f:facet>
                        <f:facet name="next">
                            <h:outputText value="#{bundle.proximo}"/>
                        </f:facet>
                        <f:facet name="previous">
                            <h:outputText value="#{bundle.anterior}"/>
                        </f:facet>
                    </rich:datascroller>

                </h:form>
                <rich:modalPanel id='modalAjuda' minHeight="200"
minWidth="450"
                                 zindex="2000" resizeable="true" >
                    <f:facet name="controls">
                        <h:graphicImage value="/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                    </f:facet>
                    <f:facet name="header">
                        <h:outputText
value="#{bundle.duvidasperfildousuario}" />
                    </f:facet>
                    <h:graphicImage value="/imagens/interrogacao.jpg" />
                </rich:modalPanel>
```

```
                <rich:modalPanel id='modalSalvar' minHeight="200"
minWidth="450"
                                 zindex="2000" resizeable="true" >
                    <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                    <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
                    <f:facet name="header">
                        <h:outputText value="#{bundle.itens}" />
                    </f:facet>

                    <h:form>
                        <h:panelGrid columns="2" id="panel">
                            <h:outputText value="#{bundle.descricao}" />
                            <h:inputText
value="#{itens.itens.descricaoitens}" readonly="#{itens.input}"
tabindex="1"/>
                            <h:outputText value="#{bundle.marca}"/>
                            <h:inputText value="#{itens.itens.marcaitens}"
readonly="#{itens.input}"  tabindex="2"/>
                            <h:outputText value="#{bundle.valor}"/>
                            <h:inputText value="#{itens.itens.valor}"
readonly="#{itens.input}"  tabindex="3"/>
                            <h:outputText value="#{bundle.tipo}"/>
                            <h:selectOneMenu value="#{itens.itens.tipo}"
readonly="#{itens.input}"  tabindex="4">
                                <f:selectItem itemLabel="Produto"
itemValue="PRODUTO"/>
                                <f:selectItem itemLabel="ServiÃ§o"
itemValue="SERVICO"/>
                            </h:selectOneMenu>
                            <h:outputText value="#{bundle.unidade}"/>
                            <h:selectOneMenu
value="#{itens.itens.unidadeIdunidade}" readonly="#{itens.input}"
tabindex="5">
                                <f:selectItems
value="#{itens.unidadeIdunidades}"/>
                            </h:selectOneMenu>
                            <a4j:commandButton id="saveEdit"
action="#{itens.create}" reRender="tabelaPerfil,scroller"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="iditens"
value="#{itens.itens.iditens}"/>
                            </a4j:commandButton>
                            <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalSalvar')"/>
                        </h:panelGrid>
                    </h:form>
                </rich:modalPanel>

                <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                                 zindex="2000" resizeable="true" >
```

```
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                        <h:panelGrid columns="2" id="panel">
                            <h:outputText value="#{bundle.descricao}" />
                            <h:inputText
value="#{itens.itens.descricaoitens}" readonly="#{itens.input}"
tabindex="1"/>
                            <h:outputText value="#{bundle.marca}"/>
                            <h:inputText value="#{itens.itens.marcaitens}"
readonly="#{itens.input}"  tabindex="2"/>
                            <h:outputText value="#{bundle.valor}"/>
                            <h:inputText value="#{itens.itens.valor}"
readonly="#{itens.input}"  tabindex="3"/>
                            <h:outputText value="#{bundle.tipo}"/>
                            <h:selectOneMenu value="#{itens.itens.tipo}"
readonly="#{itens.input}"  tabindex="4">
                                <f:selectItem itemLabel="Produto"
itemValue="PRODUTO"/>
                                <f:selectItem itemLabel="ServiÃ§o"
itemValue="SERVICO"/>
                            </h:selectOneMenu>
                            <h:outputText value="#{bundle.unidade}"/>
                            <h:selectOneMenu
value="#{itens.itens.unidadeIdunidade}" readonly="#{itens.input}"
tabindex="5">
                                <f:selectItems
value="#{itens.unidadeIdunidades}"/>
                            </h:selectOneMenu>
                            <a4j:commandButton id="saveEdit"
action="#{itens.edit}" reRender="tabelaPerfil,scroller"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="iditens"
value="#{itens.itens.iditens}"/>
                            </a4j:commandButton>
                            <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalEditar')"/>
                        </h:panelGrid>
                </h:form>
            </rich:modalPanel>


            </f:view>

    </body>
</html>
```

11.2.10 Motorista Operador/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>
<%@ taglib uri="http://myfaces.apache.org/tomahawk" prefix="t"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
   <link rel="stylesheet" type="text/css" media="screen"
href="<%=request.getContextPath()%>/CSS/viadigital.css" />
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <h:form id="formInicial">

                <h3>Cadastro Motoristas</h3>

                <rich:dataTable
value='#{motoristaoperador.motoristaoperadors}' id="tabelaPerfil" rows="10"
rowKeyVar="rk" var='item' border="1" cellpadding="2" cellspacing="0"
onRowMouseOver="this.style.backgroundColor='#FFFFFF';this.style.fontWeight=
'bold'"
onRowMouseOut="this.style.backgroundColor='#{a4jSkin.tableBackgroundColor}'
;this.style.fontWeight='normal';">
                    <rich:column >
                        <f:facet name="header">
                            <h:outputText value="#{bundle.nomemotorista}"/>
                        </f:facet>
                        <a4j:commandLink value="#{item.nome}"
immediate="true" actionListener="#{motoristaoperador.detailAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.cpfmotorista}"/>
                        </f:facet>
                        <h:outputText value="#{item.cpf}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.matriculamotorista}"/>
                        </f:facet>
                        <h:outputText value="#{item.matricula}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
```

```
                            <h:outputText
value="#{bundle.validadecarteirahab}"/>
                        </f:facet>
                        <h:outputText value="#{item.validadecarteirahab}">
                            <f:convertDateTime type="DATE"
pattern="dd/MM/yyyy" />
                        </h:outputText>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.classificacaomotorista}"/>
                        </f:facet>
                        <h:outputText
value="#{item.classificacaomotorista}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.categoriahabilitacao}"/>
                        </f:facet>
                        <h:outputText
value="#{item.categoriahabilitacaoIdcategoriahabilitacao}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.telefone}"/>
                        </f:facet>
                        <h:outputText  value="#{item.telefone}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.ramal}"/>
                        </f:facet>
                        <h:outputText  value="#{item.ramal}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.celular}"/>
                        </f:facet>
                        <h:outputText  value="#{item.celular}"/>
                    </rich:column>

                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.foto}"/>
                        </f:facet>
                        <h:graphicImage width="50" height="50"
url="/servlet/DisplayImage?imageid=#{item.idmotoristaoperador}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
styleClass="botao" immediate="false"
actionListener="#{motoristaoperador.newAction}" bypassUpdates="true"
reRender="modalSalvar" oncomplete="modalSalvar()"/>
```

```
                        </f:facet>
                        <h:commandLink value="#{bundle.excluir}"
action="#{motoristaoperador.destroy}">
                            <f:param name="idperfilusuario"
value="#{motoristaoperador.motoristaoperador.idmotoristaoperador}"/>
                        </h:commandLink>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{motoristaoperador.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">
                            <a4j:actionparam
value="#{motoristaoperador.motoristaoperador.idmotoristaoperador}"/>
                        </a4j:commandLink>
                    </rich:column>
                </rich:dataTable>

                <rich:datascroller id="scroller" for="tabelaPerfil">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>
             </h:form>


            <rich:modalPanel id='modalAjuda'
headerClass="headerModal"minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText
value="#{bundle.duvidasmotoristaOperador}" />
                </f:facet>
                <h:graphicImage value="/imagens/interrogacao.jpg" />
            </rich:modalPanel>


            <rich:modalPanel id='modalSalvar' minHeight="350"
minWidth="450"
```

```
                            zindex="2000" resizeable="true" >
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center" enctype="multipart/form-
data">
                    <h:panelGrid columns="2" >
                        <h:outputText value="#{bundle.nomemotorista}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.nome}"/>
                        <h:outputText value="#{bundle.cpfmotorista}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.cpf}"/>
                        <h:outputText
value="#{bundle.matriculamotorista}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.matricula}"/>
                        <h:outputText
value="#{bundle.validadecarteirahab}"/>
                        <t:inputCalendar
value="#{motoristaoperador.motoristaoperador.validadecarteirahab}"
popupDateFormat="dd/MM/yyyy" renderAsPopup="true"/>
                        <h:outputText value="#{bundle.telefone}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.telefone}"/>
                        <h:outputText value="#{bundle.ramal}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.ramal}"/>
                        <h:outputText value="#{bundle.celular}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.celular}"/>
                        <h:outputText
value="#{bundle.classificacaomotorista}"/>
                        <h:selectOneMenu
value="#{motoristaoperador.motoristaoperador.classificacaomotorista}">
                            <f:selectItem itemLabel="#{bundle.motorista}"
itemValue="1"/>
                            <f:selectItem
itemLabel="#{bundle.operadormaquinas}" itemValue="2"/>
                            <f:selectItem itemLabel="#{bundle.ambos}"
itemValue="3"/>
                        </h:selectOneMenu>
                        <h:outputText
value="#{bundle.categoriahabilitacao}"/>
                        <h:panelGrid columns="2">
                            <h:selectOneMenu
value="#{motoristaoperador.motoristaoperador.categoriahabilitacaoIdcategori
ahabilitacao}">
                                <f:selectItems
value="#{motoristaoperador.categoriahabilitacaoIdcategoriahabilitacaos}"/>
                            </h:selectOneMenu>
```

```
                        </h:panelGrid>

                        <h:outputText value="#{bundle.foto}"/>
                        <h:outputText value=""/>
                    <h:outputText value="Carregar foto"/>
                    <t:inputFileUpload id="fileuploadSalvar"

                                       value="#{motoristaoperador.arquivo}"
                                       immediate="true"
                                       storage="file"
                                       required="true">

                         </t:inputFileUpload>
                     <h:inputHidden/>
                    <h:message for="fileuploadSalvar" showDetail="true" />
                       <h:commandButton id="save"
action="#{motoristaoperador.create}" styleClass="botao"
value="#{bundle.salvar}">
                            <a4j:support
oncomplete="Richfaces.hideModalPanel('modalSalvar')" />
                        </h:commandButton>

                        <h:commandButton style="button" styleClass="botao"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>

                <rich:modalPanel id='modalEditar' minHeight="350"
minWidth="450"
                                 zindex="2000" resizeable="true" >
                    <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                    <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                    <f:facet name="header">
                        <h:outputText
value="#{bundle.categoriahabilitacao}" />
                    </f:facet>
                    <h:form style="text-align:center"
enctype="multipart/form-data">
                        <h:panelGrid columns="2" >
                            <h:outputText value="#{bundle.nomemotorista}"/>
                            <h:inputText
value="#{motoristaoperador.motoristaoperador.nome}"
readonly="#{usuariosistema.input}"   />
                            <h:outputText value="#{bundle.cpfmotorista}"/>
                            <h:inputText
value="#{motoristaoperador.motoristaoperador.cpf}"
readonly="#{usuariosistema.input}"   />
                            <h:outputText
value="#{bundle.matriculamotorista}"/>
```

```
                            <h:inputText
value="#{motoristaoperador.motoristaoperador.matricula}"
readonly="#{usuariosistema.input}"   />
                            <h:outputText
value="#{bundle.validadecarteirahab}"/>
                            <t:inputCalendar
value="#{motoristaoperador.motoristaoperador.validadecarteirahab}"
immediate="true" forceId="true" popupDateFormat="dd/MM/yyyy"
renderAsPopup="true">

                            </t:inputCalendar>
                             <h:outputText value="#{bundle.telefone}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.telefone}"/>
                        <h:outputText value="#{bundle.ramal}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.ramal}"/>
                        <h:outputText value="#{bundle.celular}"/>
                        <h:inputText
value="#{motoristaoperador.motoristaoperador.celular}"/>

                            <h:outputText
value="#{bundle.classificacaomotorista}"/>
                            <h:selectOneMenu
value="#{motoristaoperador.motoristaoperador.classificacaomotorista}"
readonly="#{usuariosistema.input}"   >
                                <f:selectItem
itemLabel="#{bundle.motorista}" itemValue="1"/>
                                <f:selectItem
itemLabel="#{bundle.operadormaquinas}" itemValue="2"/>
                                <f:selectItem itemLabel="#{bundle.ambos}"
itemValue="3"/>
                            </h:selectOneMenu>
                            <h:outputText
value="#{bundle.categoriahabilitacao}"/>
                            <h:panelGrid columns="2">
                                <h:selectOneMenu
value="#{motoristaoperador.motoristaoperador.categoriahabilitacaoIdcategori
ahabilitacao}" readonly="#{usuariosistema.input}"   >
                                    <f:selectItems
value="#{motoristaoperador.categoriahabilitacaoIdcategoriahabilitacaos}"/>
                                </h:selectOneMenu>
                            </h:panelGrid>

                            <h:outputText value="#{bundle.foto}"/>
                            <h:outputText value=""/>
                            <h:outputText value="Carregar foto"/>
                            <t:inputFileUpload id="fileupload"


value="#{motoristaoperador.arquivo}"
                                                    immediate="true"
                                                    storage="file"
                                                    required="false">

                            </t:inputFileUpload>
```

```
                            <h:inputHidden/>
                            <h:message for="fileupload" showDetail="true"
/>

                            <h:commandButton id="saveEdit"
styleClass="botao" action="#{motoristaoperador.edit}"
value="#{bundle.salvar}">
                                <a4j:support
oncomplete="Richfaces.hideModalPanel('modalEditar')" />
                                <a4j:actionparam name="idmotoristaoperador"
value="#{motoristaoperador.motoristaoperador.idmotoristaoperador}"/>
                            </h:commandButton>


                            <h:commandButton styleClass="botao"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalEditar')"/>
                        </h:panelGrid>
                    </h:form>
                </rich:modalPanel>

        </f:view>
    </body>
</html>
```

11.2.11 Motorista Operador/fileupload_showimg.jsp

```
<%@ page import="java.io.File,
                 java.io.InputStream,
                 java.io.FileInputStream,
                 java.io.OutputStream"%><%@ page session="true" %><%
    String contentType =
(String)application.getAttribute("fileupload_type");
    String fileName = (String)application.getAttribute("fileupload_name");

    String allowCache = request.getParameter("allowCache");
    String openDirectly = request.getParameter("openDirectly");

    if(allowCache == null || allowCache.equalsIgnoreCase("false"))
    {
        response.setHeader("pragma", "no-cache");
        response.setHeader("Cache-control", "no-cache, no-store, must-
revalidate");
        response.setHeader("Expires", "01 Apr 1995 01:10:10 GMT");
    }

    if(contentType!=null)
    {
        response.setContentType(contentType);
    }

    if(fileName != null)
    {
        fileName = fileName.substring(fileName.lastIndexOf('\\')+1);
        fileName = fileName.substring(fileName.lastIndexOf('/')+1);
```

```
        StringBuffer contentDisposition = new StringBuffer();

        if(openDirectly==null || openDirectly.equalsIgnoreCase("false"))
        {
            contentDisposition.append("attachment;");
        }

        contentDisposition.append("filename=\"");
        contentDisposition.append(fileName);
        contentDisposition.append("\"");

        response.setHeader ("Content-Disposition",
contentDisposition.toString());
    }

    byte[] bytes = (byte[])application.getAttribute("fileupload_bytes");
    if (bytes != null)
    {
        response.getOutputStream().write(bytes);

    }
%>
```

11.2.12 Perfil Usuario/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
    </head>
    <body>
        <f:view>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

            <h:form>

                <h3>Cadastro de Perfil do UsuÃ¡rio - Controle de
Acesso</h3>

                <rich:dataTable id='tabelaPerfil'
value='#{perfilusuario.perfilusuarios}' rows="10" rowKeyVar="rk"
var='item' border="1" cellpadding="2" cellspacing="0">
                <h:message for="tabelaPerfil" tooltip="true"
errorStyle="color: red" infoStyle="color: green" />
```

```
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.nomeperfilusuario}" />
                        </f:facet>
                        <a4j:commandLink value="#{item.nome}"
id="nomePerfil" immediate="true"
actionListener="#{perfilusuario.detailAction}" bypassUpdates="true"
reRender="modalSalvar"
oncomplete="Richfaces.showModalPanel('modalSalvar',{width:450, top:200})"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.funcionalidade}"/>
                        </f:facet>
                        <h:outputText value="#{item.funcionalidade}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.descricao}"/>
                        </f:facet>
                        <h:outputText value="#{item.descricao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{perfilusuario.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                        </f:facet>
                        <h:commandButton value="#{bundle.excluir}">
                           <a4j:support event="onclick"
reRender="tabelaPerfil"
                        onsubmit="if(!confirm('Are you sure to change the
country ?'))
                        {return true; return false;}"
                        oncomplete="alert('Value succesfully stored')"/>

                            <a4j:actionparam name="idperfilusuario"
value="#{item.idperfilusuario}"/>

                        </h:commandButton>

                    </h:column>
                    <h:column>
                        <f:facet name="header">
                           <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{perfilusuario.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">
                        </a4j:commandLink>
                    </h:column>
                </rich:dataTable>
```

```
                <rich:datascroller id="scroller" for="tabelaPerfil">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>

            </h:form>

            <rich:modalPanel id='modalAjuda' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="faces/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasperfildousuario}"
/>
                </f:facet>
                <h:graphicImage value="/faces//imagens/interrogacao.jpg" />
            </rich:modalPanel>


            <rich:modalPanel id='modalSalvar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.perfildousuario}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2">
                        <h:outputText value="#{bundle.nomeperfilusuario}"/>
                        <h:inputText
value="#{perfilusuario.perfilusuario.nome}"
readonly="#{perfilusuario.input}"  tabindex="1" id="modalNome"
title="Nomeperfil" />
                        <h:outputText value="#{bundle.funcionalidade}"/>
                        <h:inputText
value="#{perfilusuario.perfilusuario.funcionalidade}"
readonly="#{perfilusuario.input}"   id="modalFunc" tabindex="2"
title="Funcionalidade" />
                        <h:outputText value="#{bundle.descricao}"/>
```

```
                        <h:inputText
value="#{perfilusuario.perfilusuario.descricao}"
readonly="#{perfilusuario.input}"  id="modalDesc" tabindex="3"
title="Descricao" />
                        <a4j:commandButton id="saveEdit"
action="#{perfilusuario.create}"
reRender="tabelaPerfil,scroller"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idperfilusuario"
value="#{perfilusuario.perfilusuario.idperfilusuario}"/>
                        </a4j:commandButton>
                        <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>
            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2">
                        <h:outputText value="#{bundle.nomeperfilusuario}"/>
                        <h:inputText
value="#{perfilusuario.perfilusuario.nome}"
readonly="#{perfilusuario.input}"  tabindex="1"/>
                        <h:outputText value="#{bundle.funcionalidade}"/>
                        <h:inputText
value="#{perfilusuario.perfilusuario.funcionalidade}"
readonly="#{perfilusuario.input}"   tabindex="2"/>
                        <h:outputText value="#{bundle.descricao}"/>
                        <h:inputText
value="#{perfilusuario.perfilusuario.descricao}"
readonly="#{perfilusuario.input}"  tabindex="3"/>
                        <a4j:commandButton id="saveEdit"
action="#{perfilusuario.edit}"
reRender="tabelaPerfil,scroller"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="idperfilusuario"
value="#{perfilusuario.perfilusuario.idperfilusuario}"/>
                        </a4j:commandButton>
                        <h:commandButton value="#{bundle.cancelar}"
type="reset" onclick="Richfaces.hideModalPanel('modalEditar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>
```

```
        </f:view>
    </body>
</html>
```

11.2.13 SubGrupo Integrantes/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>List Subgrupointegrantes</title>
    </head>
    <body>
        <f:view>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <a4j:loadBundle var="mensagem"
basename="frotaControle.properties.mensagens"/>
            <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
            <h:form>

                <h3>Cadastro do Tipo de VeÃculo</h3>
                <rich:dataTable id='tabelaPerfil' rendered="true"
value='#{subgrupointegrantes.subgrupointegrantess}' rows="10"
rowKeyVar="rk"  var='item'border="1" cellpadding="2" cellspacing="0"
onRowMouseOver="this.style.backgroundColor='#FFFFFF';this.style.fontWeight=
'bold'"
onRowMouseOut="this.style.backgroundColor='#{a4jSkin.tableBackgroundColor}'
;this.style.fontWeight='normal'">
                    <h:message for="tabelaPerfil" tooltip="true"
errorStyle="color: red" infoStyle="color: green" />

                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="Nomesubgrupo"/>
                        </f:facet>
                        <h:outputText value="#{item.nomesubgrupo}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <h:outputText
value="GrupointegranteIdgrupointegrante"/>
                        </f:facet>
                        <%--h:outputText
value="#{item.grupointegranteIdgrupointegrante}"/--%>
```

```
                        <h:selectOneMenu
value="#{item.grupointegranteIdgrupointegrante}" disabled="true"
readonly="true">
                            <f:selectItems
value="#{subgrupointegrantes.grupointegranteIdgrupointegrantes}"/>
                        </h:selectOneMenu>
                    </rich:column>

                    <rich:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{subgrupointegrantes.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                        </f:facet>

                        <h:commandLink value="#{bundle.excluir}"
action="#{subgrupointegrantes.destroy}">
                            <f:param name="idusuariosistema"
value="#{item.idsubgrupointegrantes}"/>
                        </h:commandLink>

                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{subgrupointegrantes.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">
                        </a4j:commandLink>
                    </rich:column>
                </rich:dataTable>
                <rich:datascroller id="scroller" for="tabelaPerfil"
align="center">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>
            </h:form>

             <rich:modalPanel id='modalAjuda' minHeight="200"
minWidth="450"
                              zindex="2000" resizeable="true" >
                <f:facet name="controls">
```

```
                    <h:graphicImage value="faces/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasusuario}" />
                </f:facet>
                <h:graphicImage value="faces/imagens/interrogacao.jpg" />
            </rich:modalPanel>


            <rich:modalPanel id='modalSalvar' height="300" minWidth="450"
                             zindex="2000" resizeable="true">
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.usuario}" />
                </f:facet>
                <h:form  style="text-align:center">
                    <h:panelGrid columns="2" >
                        <h:outputText value="Nomesubgrupo"/>
                        <h:inputText
value="#{subgrupointegrantes.subgrupointegrantes.nomesubgrupo}"/>
                        <h:outputText
value="GrupointegranteIdgrupointegrante"/>
                        <h:selectOneMenu
value="#{subgrupointegrantes.subgrupointegrantes.grupointegranteIdgrupointe
grante}">
                        <f:selectItems
value="#{subgrupointegrantes.grupointegranteIdgrupointegrantes}"/>
                        </h:selectOneMenu>
                        <a4j:commandButton id="saveEdit"
action="#{subgrupointegrantes.create}" reRender="tabelaPerfil"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idsubgrupointegrantes"
value="#{subgrupointegrantes.subgrupointegrantes.idsubgrupointegrantes}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>
                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>

            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
```

```
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2" >
                        <h:outputText value="Nomesubgrupo"/>
                        <h:outputText
value="#{subgrupointegrantes.subgrupointegrantes.nomesubgrupo}"/>
                        <h:outputText
value="GrupointegranteIdgrupointegrante"/>
                        <h:selectOneMenu
value="#{subgrupointegrantes.subgrupointegrantes.grupointegranteIdgrupointe
grante}">
                        <f:selectItems
value="#{subgrupointegrantes.grupointegranteIdgrupointegrantes}"/>
                        </h:selectOneMenu>
                        <a4j:commandButton id="saveEdit"
action="#{subgrupointegrantes.edit}" reRender="tabelaPerfil"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="idsubgrupointegrantes"
value="#{subgrupointegrantes.subgrupointegrantes.idsubgrupointegrantes}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalEditar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>

        </f:view>
    </body>
</html>
```

11.2.14 Tipo Combustível/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <SCRIPT LANGUAGE="Javascript">
function modal()
{
Richfaces.showModalPanel('mp',{width:450, top:200});

}
        </SCRIPT>
```

```
    </head>
    <body style="text-align:center">
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <a4j:loadBundle var="mensagem"
basename="frotaControle.properties.mensagens"/>

            <h:form >
                                    <h3>Cadastro de Tipos de CombustÃ-
vel</h3>


                <rich:dataTable id='tabelaPerfil' rendered="true"
value='#{tipocombustivel.tipocombustivels}' rows="10" rowKeyVar="rk"
var='item'border="1" cellpadding="2" cellspacing="0">
                    <h:message for="tabelaPerfil" tooltip="true"
errorStyle="color: red" infoStyle="color: green" />

                    <rich:column>
                        <f:facet name="header">
                            <h:outputText value="Nometipocombustivel"/>
                        </f:facet>
                        <h:outputText value="#{item.nometipocombustivel}"/>
                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{tipocombustivel.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                        </f:facet>

                        <h:commandLink value="#{bundle.excluir}"
action="#{tipocombustivel.destroy}">
                            <f:param name="idusuariosistema"
value="#{item.idtipocombustivel}"/>
                        </h:commandLink>

                    </rich:column>
                    <rich:column>
                        <f:facet name="header">
                            <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{tipocombustivel.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">

                        </a4j:commandLink>
                    </rich:column>
                </rich:dataTable>
                <rich:datascroller id="scroller" for="tabelaPerfil"
align="center">
```

```
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>
            </h:form>


            <rich:modalPanel id='modalAjuda' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="faces/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasusuario}" />
                </f:facet>
                <h:graphicImage value="faces/imagens/interrogacao.jpg" />
            </rich:modalPanel>

            <rich:modalPanel id='modalSalvar' height="300" minWidth="450"
                             zindex="2000" resizeable="true">
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
                <f:facet name="header">
                    <h:outputText value="#{bundle.usuario}" />
                </f:facet>
                <h:form  style="text-align:center">
                    <h:panelGrid columns="2" >

                        <h:outputText value="Nometipocombustivel"/>
                        <h:inputText
value="#{tipocombustivel.tipocombustivel.nometipocombustivel}"/>
                        <a4j:commandButton id="saveEdit"
action="#{tipocombustivel.create}"
reRender="tabelaPerfil"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idtipocombustivel"
value="#{tipocombustivel.tipocombustivel.idtipocombustivel}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>
```

```
                </h:form>

            </rich:modalPanel>

            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2" >
                        <h:outputText value="Nometipocombustivel"/>
                        <h:inputText
value="#{tipocombustivel.tipocombustivel.nometipocombustivel}"/>
                        <a4j:commandButton id="saveEdit"
action="#{tipocombustivel.edit}"
reRender="tabelaPerfil"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idtipocombustivel"
value="#{tipocombustivel.tipocombustivel.idtipocombustivel}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>

                </h:form>

            </rich:modalPanel>


        </f:view>
    </body>
</html>
```

11.2.15 Unidade/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
```

```
        <SCRIPT LANGUAGE="Javascript">
function modal()
{
Richfaces.showModalPanel('mp',{width:450, top:200});

}

        </SCRIPT>
    </head>
    <body style="text-align:center">
        <f:view>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>

            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h:form id="formInicial">
                <br>
                <a href="/FrotaControles/index.jsp">Back to index</a>
                <br>
                <rich:dataTable value='#{unidade.unidades}'
id="tabelaPerfil" rows="10"  rowKeyVar="rk" var='item' border="1"
cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.descricao}"/>
                        </f:facet>
                        <a4j:commandLink value="#{item.nomeunidade}"
id="nome" immediate="true" actionListener="#{unidade.detailAction}"
bypassUpdates="true" reRender="mp" oncomplete="modal()"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.tipo}"/>
                        </f:facet>
                        <h:outputText value="#{item.tipounidade}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{unidade.newAction}" bypassUpdates="true"
reRender="mp" oncomplete="modal()"/>
                        </f:facet>
                        <h:commandLink value="#{bundle.excluir}"
action="#{unidade.destroy}">
                            <f:param name="idperfilusuario"
value="#{item.idunidade}"/>
                            </h:commandLink>

                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="help"
oncomplete="help()"/>
                        </f:facet>
```

```
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{unidade.editAction}"
bypassUpdates="true" reRender="mp" oncomplete="modal()">

                        </a4j:commandLink>
                    </h:column>
                </rich:dataTable>

                <rich:datascroller for="tabelaPerfil">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>

            </h:form>


            <rich:modalPanel id='help' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('help')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasperfildousuario}"
/>
                </f:facet>
                <h:graphicImage value="/imagens/interrogacao.jpg" />
            </rich:modalPanel>

            <rich:modalPanel id='mp' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="mp" tooltip="true" errorStyle="color: red"
infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
                <f:facet name="header">
                    <h:outputText value="#{bundle.nomeunidade}" />
                </f:facet>

                <h:form id="formModal">
                    <h:panelGrid columns="2" id="panel">
                        <h:outputText value="#{bundle.descricao}"/>
                        <h:inputText value="#{unidade.unidade.nomeunidade}"
readonly="#{unidade.input}"/>
                        <h:outputText value="#{bundle.tipo}"/>
```

```
                        <h:inputText value="#{unidade.unidade.tipounidade}"
readonly="#{unidade.input}"/>
                        <h:commandButton id="saveEdit"
action="#{unidade.edit}" title="salvar" disabled="#{unidade.input}"
value="#{bundle.salvar}">
                            <f:param name="idperfilusuario"
value="#{unidade.unidade.idunidade}"/>
                            <%--  onsubmit="if(confirm('Are you sure to
change the option ?'))

{Richfaces.hideModalPanel('modalUser'); return true;}"
                                        oncomplete="alert('Value
succesfully stored')"--%>
                        </h:commandButton>
                        <h:inputHidden/>
                        <h:commandButton value="#{bundle.cancelar}"
accesskey="c" type="reset" onclick="Richfaces.hideModalPanel('mp')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>
        </f:view>
    </body>
</html>
```

11.2.16 Usuário Sistema/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@ taglib uri="http://myfaces.apache.org/tomahawk" prefix="t" %>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>

    </head>
    <body>

        <f:view>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <a4j:loadBundle var="mensagem"
basename="frotaControle.properties.mensagens"/>
            <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
            <h:form id="formInicial">

                    <h3>Cadastro de UsuÃ¡rios do Sistema</h3>
```

```
                    <rich:dataTable id='tabelaUsuario' rendered="true"
value='#{usuariosistema.usuariosistemas}' rows="10" rowKeyVar="rk"
var='item'border="1" cellpadding="2" cellspacing="0">
                        <h:message for="tabelaUsuario" tooltip="true"
errorStyle="color: red" infoStyle="color: green" />

                        <rich:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.nomeusuario}"
/>
                            </f:facet>
                            <a4j:commandLink value="#{item.nomeusuario}"
id="nome" immediate="true" actionListener="#{usuariosistema.detailAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()"/>

                        </rich:column>
                        <rich:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.endereco}"/>
                            </f:facet>
                            <h:outputText value="#{item.enderecousuario}"/>
                        </rich:column>
                        <rich:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.login}"/>
                            </f:facet>
                            <h:outputText value="#{item.loginusuario}"/>
                        </rich:column>
                        <rich:column>
                            <f:facet name="header">
                                <h:outputText value="#{bundle.cpf}"/>
                            </f:facet>
                            <h:outputText value="#{item.cpfusuario}"/>
                        </rich:column>
                        <rich:column>
                            <f:facet name="header">
                                <h:outputText
value="#{bundle.perfildousuario}"/>
                            </f:facet>
                            <h:outputText
value="#{item.perfilusuarioIdperfilusuario.nome}"/>
                        </rich:column>

                        <rich:column>
                            <f:facet name="header">
                                <a4j:commandButton value="#{bundle.novo}"
immediate="true" actionListener="#{usuariosistema.newAction}"
bypassUpdates="true" reRender="modalSalvar" oncomplete="modalSalvar()"/>
                            </f:facet>

                            <h:commandLink value="#{bundle.excluir}"
action="#{usuariosistema.destroy}">
                                <f:param name="idusuariosistema"
value="#{item.idusuariosistema}"/>
                            </h:commandLink>
```

```
                        </rich:column>
                        <rich:column>
                            <f:facet name="header">
                                <a4j:commandButton
image="faces/imagens/inter2.bmp" immediate="true" reRender="modalAjuda"
oncomplete="modalAjuda()"/>
                            </f:facet>
                            <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{usuariosistema.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">

                            </a4j:commandLink>
                        </rich:column>
                    </rich:dataTable>
                    <rich:datascroller id="scroller" for="tabelaUsuario"
align="center">
                        <f:facet name="first">
                            <h:outputText value="#{bundle.primeiro}"/>
                        </f:facet>
                        <f:facet name="last">
                            <h:outputText value="#{bundle.ultimo}"/>
                        </f:facet>
                        <f:facet name="next">
                            <h:outputText value="#{bundle.proximo}"/>
                        </f:facet>
                        <f:facet name="previous">
                            <h:outputText value="#{bundle.anterior}"/>
                        </f:facet>
                    </rich:datascroller>

            </h:form>

            <rich:modalPanel id='modalAjuda' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="faces/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasusuario}" />
                </f:facet>
                <h:graphicImage value="faces/imagens/interrogacao.jpg" />
            </rich:modalPanel>

            <rich:modalPanel id='modalSalvar' height="300" minWidth="450"
                             zindex="2000" resizeable="true">
                <h:message for="modalSalvar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.usuario}" />
                </f:facet>
                <h:form  style="text-align:center">
                    <h:panelGrid columns="2" >
```

```
                        <h:outputText value="#{bundle.nomeusuario}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.nomeusuario}" />
                        <h:outputText value="#{bundle.endereco}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.enderecousuario}"  />
                        <h:outputText value="#{bundle.login}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.loginusuario}"  />
                        <h:outputText value="#{bundle.senha}"/>
                        <h:inputSecret
value="#{usuariosistema.usuariosistema.senhausuario}"  />
                        <h:outputText value="#{bundle.cpf}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.cpfusuario}"  />
                        <h:outputText value="#{bundle.perfilusuario}"/>
                        <h:selectOneMenu
value="#{usuariosistema.usuariosistema.perfilusuarioIdperfilusuario}" >
                            <f:selectItems
value="#{usuariosistema.perfilusuarioIdperfilusuarios}" />
                        </h:selectOneMenu>
                        <a4j:commandButton id="saveEdit"
action="#{usuariosistema.create}"
reRender="tabelaUsuario,scroller"value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalSalvar')">
                            <a4j:actionparam name="idusuariosistema"
value="#{usuariosistema.usuariosistema.idusuariosistema}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalSalvar')"/>

                    </h:panelGrid>

                </h:form>
            </rich:modalPanel>
            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>

                <f:facet name="header">
                    <h:outputText value="#{bundle.categoriahabilitacao}" />
                </f:facet>
                <h:form style="text-align:center">
                    <h:panelGrid columns="2" >
                        <h:outputText value="#{bundle.nomeusuario}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.nomeusuario}"
readonly="#{usuariosistema.input}"   />
                        <h:outputText value="#{bundle.endereco}"/>
```

```
                        <h:inputText
value="#{usuariosistema.usuariosistema.enderecousuario}"
readonly="#{usuariosistema.input}" />
                        <h:outputText value="#{bundle.login}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.loginusuario}"
readonly="#{usuariosistema.input}" />
                        <h:outputText value="#{bundle.senha}"/>
                        <h:inputSecret
value="#{usuariosistema.usuariosistema.senhausuario}"
readonly="#{usuariosistema.input}" />
                        <h:outputText value="#{bundle.cpf}"/>
                        <h:inputText
value="#{usuariosistema.usuariosistema.cpfusuario}">

                        </h:inputText>
                        <h:outputText value="#{bundle.perfilusuario}"/>
                        <h:selectOneMenu
value="#{usuariosistema.usuariosistema.perfilusuarioIdperfilusuario}" >
                            <f:selectItems
value="#{usuariosistema.perfilusuarioIdperfilusuarios}" />
                        </h:selectOneMenu>
                        <a4j:commandButton id="saveEdit"
action="#{usuariosistema.edit}" reRender="tabelaUsuario,scroller"
value="#{bundle.salvar}"
oncomplete="Richfaces.hideModalPanel('modalEditar')">
                            <a4j:actionparam name="idusuariosistema"
value="#{usuariosistema.usuariosistema.idusuariosistema}"/>
                        </a4j:commandButton>
                        <h:commandButton style="button"
value="#{bundle.cancelar}" type="reset"
onclick="Richfaces.hideModalPanel('modalEditar')"/>

                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>


        </f:view>
    </body>
</html>
```

11.2.17 Atividade/Detail.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Detail of Atividade</title>
    </head>
    <body>
```

```
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Detail of atividade</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idatividade:"/>
                    <h:outputText
value="#{atividade.atividade.idatividade}" title="Idatividade" />
                    <h:outputText value="Dataatividade:"/>
                    <h:outputText
value="#{atividade.atividade.dataatividade}" title="Dataatividade" />
                    <h:outputText value="Horainicioatividade:"/>
                    <h:outputText
value="#{atividade.atividade.horainicioatividade}"
title="Horainicioatividade" />
                    <h:outputText value="Dataterminoatividade:"/>
                    <h:outputText
value="#{atividade.atividade.dataterminoatividade}"
title="Dataterminoatividade" />
                    <h:outputText value="Horatermino:"/>
                    <h:outputText
value="#{atividade.atividade.horatermino}" title="Horatermino" />
                    <h:outputText value="Kminicial:"/>
                    <h:outputText value="#{atividade.atividade.kminicial}"
title="Kminicial" />
                    <h:outputText value="Obsatividade:"/>
                    <h:outputText
value="#{atividade.atividade.obsatividade}" title="Obsatividade" />
                    <h:outputText value="Kmfinal:"/>
                    <h:outputText value="#{atividade.atividade.kmfinal}"
title="Kmfinal" />
                    <h:outputText value="Multa:"/>
                    <h:outputText value="#{atividade.atividade.multa}"
title="Multa" />
                    <h:outputText value="Gravidademulta:"/>
                    <h:outputText
value="#{atividade.atividade.gravidademulta}" title="Gravidademulta" />
                    <h:outputText value="Obspercurso:"/>
                    <h:outputText
value="#{atividade.atividade.obspercurso}" title="Obspercurso" />
                    <h:outputText value="Outrasobs:"/>
                    <h:outputText value="#{atividade.atividade.outrasobs}"
title="Outrasobs" />
                    <h:outputText value="Acidente:"/>
                    <h:outputText value="#{atividade.atividade.acidente}"
title="Acidente" />
                    <h:outputText value="IntegranteIdintegrante:"/>
                    <h:outputText
value="#{atividade.atividade.integranteIdintegrante}"
title="IntegranteIdintegrante" />
                    <h:outputText
value="MotoristaoperadorIdmotoristaoperador:"/>
                    <h:outputText
value="#{atividade.atividade.motoristaoperadorIdmotoristaoperador}"
title="MotoristaoperadorIdmotoristaoperador" />
```

```
                </h:panelGrid>
                <h:commandLink action="atividade_edit" value="Edit" />
                <br>
                <h:commandLink action="atividade_list" value="Show All
Atividade"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.18 Atividade/Edit.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Edit Atividade</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Edit atividade</h1>
            <h:form>
                <h:inputHidden value="#{atividade.atividade}"
immediate="true"/>
                <h:panelGrid columns="2">
                    <h:outputText value="Idatividade:"/>
                    <h:outputText
value="#{atividade.atividade.idatividade}" title="Idatividade" />
                    <h:outputText value="Dataatividade (MM/dd/yyyy):"/>
                    <h:inputText id="dataatividade"
value="#{atividade.atividade.dataatividade}" title="Dataatividade" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Horainicioatividade (hh:mm:ss):"/>
                    <h:inputText id="horainicioatividade"
value="#{atividade.atividade.horainicioatividade}"
title="Horainicioatividade" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText value="Dataterminoatividade
(MM/dd/yyyy):"/>
                    <h:inputText id="dataterminoatividade"
value="#{atividade.atividade.dataterminoatividade}"
title="Dataterminoatividade" >
```

```
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Horatermino (hh:mm:ss):"/>
                    <h:inputText id="horatermino"
value="#{atividade.atividade.horatermino}" title="Horatermino" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText value="Kminicial:"/>
                    <h:inputText id="kminicial"
value="#{atividade.atividade.kminicial}" title="Kminicial" />
                    <h:outputText value="Obsatividade:"/>
                    <h:inputText id="obsatividade"
value="#{atividade.atividade.obsatividade}" title="Obsatividade" />
                    <h:outputText value="Kmfinal:"/>
                    <h:inputText id="kmfinal"
value="#{atividade.atividade.kmfinal}" title="Kmfinal" />
                    <h:outputText value="Multa:"/>
                    <h:inputText id="multa"
value="#{atividade.atividade.multa}" title="Multa" />
                    <h:outputText value="Gravidademulta:"/>
                    <h:inputText id="gravidademulta"
value="#{atividade.atividade.gravidademulta}" title="Gravidademulta" />
                    <h:outputText value="Obspercurso:"/>
                    <h:inputText id="obspercurso"
value="#{atividade.atividade.obspercurso}" title="Obspercurso" />
                    <h:outputText value="Outrasobs:"/>
                    <h:inputText id="outrasobs"
value="#{atividade.atividade.outrasobs}" title="Outrasobs" />
                    <h:outputText value="Acidente:"/>
                    <h:inputText id="acidente"
value="#{atividade.atividade.acidente}" title="Acidente" />
                    <h:outputText value="IntegranteIdintegrante:"/>
                    <h:selectOneMenu id="integranteIdintegrante"
value="#{atividade.atividade.integranteIdintegrante}"
title="IntegranteIdintegrante">
                        <f:selectItems
value="#{atividade.integranteIdintegrantes}"/>
                    </h:selectOneMenu>
                    <h:outputText
value="MotoristaoperadorIdmotoristaoperador:"/>
                    <h:selectOneMenu
id="motoristaoperadorIdmotoristaoperador"
value="#{atividade.atividade.motoristaoperadorIdmotoristaoperador}"
title="MotoristaoperadorIdmotoristaoperador">
                        <f:selectItems
value="#{atividade.motoristaoperadorIdmotoristaoperadors}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{atividade.edit}" value="Save"/>
                <br>
                <h:commandLink action="atividade_list" value="Show All
Atividade"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
```

```
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.19 Atividade/IniciarAtividade.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>
<%@ taglib uri="http://myfaces.apache.org/tomahawk" prefix="t"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

    </head>
    <body>

        <f:view>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <h:form id="iniciar">

            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>

                <a4j:outputPanel ajaxRendered="true"   >
                        <h:panelGrid columns="2" style="align:left">
                            <h:outputText value="#{bundle.grupo}"/>
                            <h:selectOneMenu id="grupo"
value="#{atividade.grupo}">
                                <f:selectItems
value="#{atividade.grupoIntegranteIdGrupointegrantes}"/>
                                <a4j:support ajaxSingle="true"
event="onchange" reRender="sub,integrante,motorista"/>
                            </h:selectOneMenu>

                        </h:panelGrid>
                    </a4j:outputPanel>
                    <a4j:outputPanel id="sub" ajaxRendered="true">
                        <h:panelGrid columns="2">
                            <h:outputText value="#{bundle.tipo}"/>
                            <h:selectOneMenu  value="#{atividade.sub}">
                                <f:selectItems
value="#{atividade.subGrupoIntegrante}"/>
                                <a4j:support ajaxSingle="true"
event="onchange" reRender="integrante,motorista"/>
                            </h:selectOneMenu>
```

```
                        </h:panelGrid>
                    </a4j:outputPanel>
                    <a4j:outputPanel id="integrante" ajaxRendered="true">
                        <h:panelGrid columns="2">
                            <h:outputText value="#{bundle.veiculo}"/>
                            <h:selectOneMenu id= "veiculo"
value="#{atividade.integrante}">
                                <f:selectItems
value="#{atividade.integranteIdintegrantes}"/>
                                <a4j:support ajaxSingle="true"
event="onchange" reRender="placa"/>
                            </h:selectOneMenu>
                            <h:message for="veiculo" infoStyle="color:
green"/>
                        </h:panelGrid>
                    </a4j:outputPanel>
                    <a4j:outputPanel id="placa" ajaxRendered="true">
                        <h:panelGrid columns="2">
                            <h:outputText value="#{bundle.placa}"/>
                            <h:selectOneMenu
value="#{atividade.integranteReal}">
                                <f:selectItems
value="#{atividade.placas}"/>
                                <a4j:support ajaxSingle="true"
event="onchange" reRender="km"/>
                            </h:selectOneMenu>
                        </h:panelGrid>
                    </a4j:outputPanel>
                    <a4j:outputPanel id="outros" ajaxRendered="true">
                        <h:outputText id="km"
value="#{bundle.QuilometragemInicial}"/>
                        <h:inputText readonly="true"
value="#{atividade.kmAtual}">
                            <a4j:actionparam value="#{atividade.kmAtual}"/>
                        </h:inputText>

                    </a4j:outputPanel>

                    <a4j:outputPanel id="motorista" ajaxRendered="true">
                        <h:panelGrid columns="2">
                            <h:outputText value="#{bundle.motorista}"/>
                            <h:selectOneMenu
value="#{atividade.motorista}">
                                <f:selectItems
value="#{atividade.motoristaoperadorIdmotoristaoperadors}"/>
                                <a4j:support ajaxSingle="true"
event="onchange" reRender="km"/>
                            </h:selectOneMenu>
                        </h:panelGrid>
                    </a4j:outputPanel>
                    <h:panelGrid columns="2">
                        <h:outputText value="#{bundle.horaInicial}"/>
                        <t:inputDate id="date2"
value="#{atividade.atividade.horainicioatividade}" type="short_time"/>
                        <h:message for="date2"/>
                        <h:outputText value="#{bundle.dataTermino}"/>
```

```
                        <t:inputCalendar
value="#{atividade.atividade.dataterminoatividade}"
                                         popupTodayString="'hoje'"
immediate="true"
                                         popupDateFormat="dd/MM/yyyy"
renderAsPopup="true"
                                         helpText="DD/MM/YYYY"
                                         forceId="true"/>
                        <h:outputText value="#{bundle.horaTermino}"/>
                        <t:inputDate id="horaFinal"
value="#{atividade.atividade.horatermino}" type="short_time"/>

                        <h:message for="horaFinal"/>
                        <h:outputText value="#{bundle.obs}"/>

                        <h:inputTextarea cols="20" rows="5"
value="#{atividade.atividade.obsatividade}">
                            <f:validateLength maximum="255"/>
                        </h:inputTextarea>
                    </h:panelGrid>



               <h:commandButton id="saveEdit"  action="#{atividade.create}"
value="#{bundle.salvar}" >
                            <f:param name="idatividade"
value="#{atividade.atividade.idatividade}"/>
                        </h:commandButton>
                        <h:commandButton value="#{bundle.cancelar}"
type="submit"action="view_inicio"/>

</h:form>

        </f:view>

    </body>
</html>
```

11.2.20 Atividade/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://richfaces.ajax4jsf.org/rich" prefix="rich"%>
<%@ taglib uri="https://ajax4jsf.dev.java.net/ajax" prefix="a4j"%>
<%@ taglib uri="http://myfaces.apache.org/tomahawk" prefix="t"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
```

```
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <a4j:loadBundle var="bundle"
basename="frotaControle.properties.internacional"/>
            <h:form id="formInicial">
                <h:commandLink action="#{atividade.createSetup}" value="New
Atividade"/>
                <rich:dataTable value='#{atividade.atividades}'
id="tabelaPerfil" rows="10"  rowKeyVar="rk" var='item' border="1"
cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.dataInicio}"/>
                        </f:facet>
                        <h:outputText value="#{item.dataatividade}">
                            <f:convertDateTime type="DATE"
pattern="dd/MM/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.horaInicio}"/>
                        </f:facet>
                        <h:outputText value="#{item.horainicioatividade}">
                            <f:convertDateTime type="TIME" pattern="hh:mm"
/>
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.dataPrevistaTermino}"/>
                        </f:facet>
                        <h:outputText value="#{item.dataterminoatividade}">
                            <f:convertDateTime type="DATE"
pattern="dd/MM/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.horaPrevistaTermino}"/>
                        </f:facet>
                        <h:outputText value="#{item.horatermino}">
                            <f:convertDateTime type="TIME"
pattern="hh:mm:ss" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.kmAtual}"/>
                        </f:facet>
```

```
                        <h:outputText value="#{item.kminicial}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.obsAtividade}"/>
                        </f:facet>
                        <h:outputText value="#{item.obsatividade}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.veiculo}"/>
                        </f:facet>
                        <h:outputText
value="#{item.integranteIdintegrante}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="#{bundle.motorista}"/>
                        </f:facet>
                        <h:outputText
value="#{item.motoristaoperadorIdmotoristaoperador}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="#{bundle.entradaAtividade}"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.entrada}"
immediate="true" actionListener="#{atividade.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()"/>

                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <a4j:commandButton  image="/imagens/inter2.bmp"
immediate="true" reRender="modalAjuda" oncomplete="modalAjuda()"/>
                        </f:facet>
                        <a4j:commandLink value="#{bundle.editar}"
immediate="true" actionListener="#{atividade.editAction}"
bypassUpdates="true" reRender="modalEditar" oncomplete="modalEditar()">

                        </a4j:commandLink>
                    </h:column>

                </rich:dataTable>
                <rich:datascroller id="scroller" for="tabelaPerfil">
                    <f:facet name="first">
                        <h:outputText value="#{bundle.primeiro}"/>
                    </f:facet>
                    <f:facet name="last">
                        <h:outputText value="#{bundle.ultimo}"/>
                    </f:facet>
                    <f:facet name="next">
                        <h:outputText value="#{bundle.proximo}"/>
                    </f:facet>
                    <f:facet name="previous">
```

```
                        <h:outputText value="#{bundle.anterior}"/>
                    </f:facet>
                </rich:datascroller>

            </h:form>


            <rich:modalPanel id='modalAjuda' minHeight="200" minWidth="450"
                             zindex="2000" resizeable="true" >
                <f:facet name="controls">
                    <h:graphicImage value="/imagens/close.png"
style="cursor:pointer" onclick="Richfaces.hideModalPanel('modalAjuda')" />
                </f:facet>
                <f:facet name="header">
                    <h:outputText value="#{bundle.duvidasperfildousuario}"
/>
                </f:facet>
                <h:graphicImage value="/imagens/interrogacao.jpg" />
            </rich:modalPanel>


            <rich:modalPanel id='modalEditar' minHeight="200"
minWidth="450"
                             zindex="2000" resizeable="true" >
                <h:message for="modalEditar" tooltip="true"
errorStyle="color: red" infoStyle="color: green"/>
                <h:messages tooltip="true" errorStyle="color: red"
infoStyle="color: green" layout="table"/>
                <f:facet name="header">
                    <h:outputText value="#{bundle.cadastrofornecedor}" />
                </f:facet>

                <h:form>
                    <h:panelGrid columns="2">
                        <h:outputText value="#{bundle.dataInicio}"/>
                        <h:inputText
value="#{atividade.atividade.dataatividade}" readonly="true">
                            <f:convertDateTime type="DATE"
pattern="dd/MM/yyyy" />
                        </h:inputText>
                        <h:outputText value="#{bundle.dataFim}"/>
                        <h:inputText
value="#{atividade.atividade.dataterminoatividade}">
                            <f:convertDateTime type="DATE"
pattern="dd/MM/yyyy"/>
                        </h:inputText>
                        <h:outputText value="#{bundle.integrante}"/>
                        <h:inputText
value="#{atividade.atividade.integranteIdintegrante.nomeintegrante}"
readonly="true">
                            </h:inputText>
                        <h:outputText value="#{bundle.placa}"/>
                        <h:inputText
value="#{atividade.atividade.integranteIdintegrante.placa}"
readonly="true">
                        </h:inputText>
```

```
                        <h:outputText value="#{bundle.motorista}"/>
                        <h:inputText
value="#{atividade.atividade.motoristaoperadorIdmotoristaoperador.nome}"
readonly="true">
                        </h:inputText>
                            <h:outputText
value="#{bundle.horaPrevistaTermino}"/>
                        <h:inputText value="#{item.horatermino}"
readonly="true">
                            <f:convertDateTime type="TIME"
pattern="hh:mm:ss" />
                        </h:inputText>
                        <h:outputText value="#{bundle.kmInicial}"/>
                        <h:inputText
value="#{atividade.atividade.kmInicial}" readonly="true">
                        </h:inputText>
                        <h:panelGrid rendered="#{not empty
atividade.atividade.integranteIdintegrante.kmatual}"/>
                        <h:outputText value="#{bundle.kmInicial}"/>
                        <h:inputText
value="#{atividade.atividade.kmInicial}" readonly="true">
                        </h:inputText>



                    </h:panelGrid>
                </h:form>
            </rich:modalPanel>
        </f:view>
    </body>
</html>
```

11.2.21 Atividade/New.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>New Atividade</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>New atividade</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idatividade:"/>
                    <h:inputText id="idatividade"
value="#{atividade.atividade.idatividade}" title="Idatividade" />
                    <h:outputText value="Dataatividade (MM/dd/yyyy):"/>
```

```
                    <h:inputText id="dataatividade"
value="#{atividade.atividade.dataatividade}" title="Dataatividade" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Horainicioatividade (hh:mm:ss):"/>
                    <h:inputText id="horainicioatividade"
value="#{atividade.atividade.horainicioatividade}"
title="Horainicioatividade" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText value="Dataterminoatividade
(MM/dd/yyyy):"/>
                    <h:inputText id="dataterminoatividade"
value="#{atividade.atividade.dataterminoatividade}"
title="Dataterminoatividade" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Horatermino (hh:mm:ss):"/>
                    <h:inputText id="horatermino"
value="#{atividade.atividade.horatermino}" title="Horatermino" >
                        <f:convertDateTime type="TIME" pattern="hh:mm:ss"
/>
                    </h:inputText>
                    <h:outputText value="Kminicial:"/>
                    <h:inputText id="kminicial"
value="#{atividade.atividade.kminicial}" title="Kminicial" />
                    <h:outputText value="Obsatividade:"/>
                    <h:inputText id="obsatividade"
value="#{atividade.atividade.obsatividade}" title="Obsatividade" />
                    <h:outputText value="Kmfinal:"/>
                    <h:inputText id="kmfinal"
value="#{atividade.atividade.kmfinal}" title="Kmfinal" />
                    <h:outputText value="Multa:"/>
                    <h:inputText id="multa"
value="#{atividade.atividade.multa}" title="Multa" />
                    <h:outputText value="Gravidademulta:"/>
                    <h:inputText id="gravidademulta"
value="#{atividade.atividade.gravidademulta}" title="Gravidademulta" />
                    <h:outputText value="Obspercurso:"/>
                    <h:inputText id="obspercurso"
value="#{atividade.atividade.obspercurso}" title="Obspercurso" />
                    <h:outputText value="Outrasobs:"/>
                    <h:inputText id="outrasobs"
value="#{atividade.atividade.outrasobs}" title="Outrasobs" />
                    <h:outputText value="Acidente:"/>
                    <h:inputText id="acidente"
value="#{atividade.atividade.acidente}" title="Acidente" />
                    <h:outputText value="IntegranteIdintegrante:"
rendered="#{atividade.atividade.integranteIdintegrante == null}"/>
                    <h:selectOneMenu id="integranteIdintegrante"
value="#{atividade.atividade.integranteIdintegrante}"
title="IntegranteIdintegrante"
rendered="#{atividade.atividade.integranteIdintegrante == null}">
```

```
                        <f:selectItems
value="#{atividade.integranteIdintegrantes}"/>
                    </h:selectOneMenu>
                    <h:outputText
value="MotoristaoperadorIdmotoristaoperador:"
rendered="#{atividade.atividade.motoristaoperadorIdmotoristaoperador ==
null}"/>
                    <h:selectOneMenu
id="motoristaoperadorIdmotoristaoperador"
value="#{atividade.atividade.motoristaoperadorIdmotoristaoperador}"
title="MotoristaoperadorIdmotoristaoperador"
rendered="#{atividade.atividade.motoristaoperadorIdmotoristaoperador ==
null}">
                        <f:selectItems
value="#{atividade.motoristaoperadorIdmotoristaoperadors}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{atividade.create}"
value="Create"/>
                <br>
                <h:commandLink action="atividade_list" value="Show All
Atividade"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.22 Manutenção/Detail.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Detail of Manutencao</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Detail of manutencao</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idmanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.idmanutencao}" title="Idmanutencao" />
                    <h:outputText value="Descricaomanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.descricaomanutencao}"
title="Descricaomanutencao" />
```

```
                    <h:outputText value="Datamanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.datamanutencao}" title="Datamanutencao" />
                    <h:outputText value="Datareparo:"/>
                    <h:outputText
value="#{manutencao.manutencao.datareparo}" title="Datareparo" />
                    <h:outputText value="Urgenciamanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.urgenciamanutencao}"
title="Urgenciamanutencao" />
                    <h:outputText value="Categoriamanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.categoriamanutencao}"
title="Categoriamanutencao" />
                    <h:outputText value="Obsmanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.obsmanutencao}" title="Obsmanutencao" />
                    <h:outputText value="Kmatualintegrante:"/>
                    <h:outputText
value="#{manutencao.manutencao.kmatualintegrante}"
title="Kmatualintegrante" />
                    <h:outputText value="AtividadeIdatividade:"/>
                    <h:outputText
value="#{manutencao.manutencao.atividadeIdatividade}"
title="AtividadeIdatividade" />
                    <h:outputText value="IntegranteIdintegrante:"/>
                    <h:outputText
value="#{manutencao.manutencao.integranteIdintegrante}"
title="IntegranteIdintegrante" />
                </h:panelGrid>
                <h:commandLink action="manutencao_edit" value="Edit" />
                <br>
                <h:commandLink action="manutencao_list" value="Show All
Manutencao"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.23 Manutenção/Edit.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Edit Manutencao</title>
    </head>
    <body>
        <f:view>
```

```
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Edit manutencao</h1>
            <h:form>
                <h:inputHidden value="#{manutencao.manutencao}"
immediate="true"/>
                <h:panelGrid columns="2">
                    <h:outputText value="Idmanutencao:"/>
                    <h:outputText
value="#{manutencao.manutencao.idmanutencao}" title="Idmanutencao" />
                    <h:outputText value="Descricaomanutencao:"/>
                    <h:inputText id="descricaomanutencao"
value="#{manutencao.manutencao.descricaomanutencao}"
title="Descricaomanutencao" />
                    <h:outputText value="Datamanutencao (MM/dd/yyyy):"/>
                    <h:inputText id="datamanutencao"
value="#{manutencao.manutencao.datamanutencao}" title="Datamanutencao" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Datareparo (MM/dd/yyyy):"/>
                    <h:inputText id="datareparo"
value="#{manutencao.manutencao.datareparo}" title="Datareparo" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Urgenciamanutencao:"/>
                    <h:inputText id="urgenciamanutencao"
value="#{manutencao.manutencao.urgenciamanutencao}"
title="Urgenciamanutencao" />
                    <h:outputText value="Categoriamanutencao:"/>
                    <h:inputText id="categoriamanutencao"
value="#{manutencao.manutencao.categoriamanutencao}"
title="Categoriamanutencao" />
                    <h:outputText value="Obsmanutencao:"/>
                    <h:inputText id="obsmanutencao"
value="#{manutencao.manutencao.obsmanutencao}" title="Obsmanutencao" />
                    <h:outputText value="Kmatualintegrante:"/>
                    <h:inputText id="kmatualintegrante"
value="#{manutencao.manutencao.kmatualintegrante}"
title="Kmatualintegrante" />
                    <h:outputText value="AtividadeIdatividade:"/>
                    <h:selectOneMenu id="atividadeIdatividade"
value="#{manutencao.manutencao.atividadeIdatividade}"
title="AtividadeIdatividade">
                        <f:selectItems
value="#{manutencao.atividadeIdatividades}"/>
                    </h:selectOneMenu>
                    <h:outputText value="IntegranteIdintegrante:"/>
                    <h:selectOneMenu id="integranteIdintegrante"
value="#{manutencao.manutencao.integranteIdintegrante}"
title="IntegranteIdintegrante">
                        <f:selectItems
value="#{manutencao.integranteIdintegrantes}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
```

```
                <h:commandLink action="#{manutencao.edit}" value="Save"/>
                <br>
                <h:commandLink action="manutencao_list" value="Show All
Manutencao"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.24 Manutenção/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <link rel="stylesheet" type="text/css" media="screen"
href="<%=request.getContextPath()%>/CSS/viadigital.css" />
   </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Listing Manutencaos</h1>
            <h:form>
                <h:commandLink action="#{manutencao.createSetup}"
value="New Manutencao"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
                <br>
                <h:outputText value="Item #{manutencao.firstItem +
1}..#{manutencao.lastItem} of #{manutencao.itemCount}"/> 
                <h:commandLink action="#{manutencao.prev}" value="Previous
#{manutencao.batchSize}" rendered="#{manutencao.firstItem >=
manutencao.batchSize}"/> 
                <h:commandLink action="#{manutencao.next}" value="Next
#{manutencao.batchSize}" rendered="#{manutencao.lastItem +
manutencao.batchSize <= manutencao.itemCount}"/> 
                <h:commandLink action="#{manutencao.next}" value="Remaining
#{manutencao.itemCount - manutencao.lastItem}"
                               rendered="#{manutencao.lastItem <
manutencao.itemCount && manutencao.lastItem + manutencao.batchSize >
manutencao.itemCount}"/>
                <h:dataTable value='#{manutencao.manutencaos}' var='item'
border="1" cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Idmanutencao"/>
                        </f:facet>
                        <h:commandLink action="#{manutencao.detailSetup}"
value="#{item.idmanutencao}"/>
```

```
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Descricaomanutencao"/>
                        </f:facet>
                        <h:outputText value="#{item.descricaomanutencao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datamanutencao"/>
                        </f:facet>
                        <h:outputText value="#{item.datamanutencao}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datareparo"/>
                        </f:facet>
                        <h:outputText value="#{item.datareparo}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Urgenciamanutencao"/>
                        </f:facet>
                        <h:outputText value="#{item.urgenciamanutencao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Categoriamanutencao"/>
                        </f:facet>
                        <h:outputText value="#{item.categoriamanutencao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Obsmanutencao"/>
                        </f:facet>
                        <h:outputText value="#{item.obsmanutencao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Kmatualintegrante"/>
                        </f:facet>
                        <h:outputText value="#{item.kmatualintegrante}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="AtividadeIdatividade"/>
                        </f:facet>
                        <h:outputText
value="#{item.atividadeIdatividade}"/>
                    </h:column>
```

```
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="IntegranteIdintegrante"/>
                        </f:facet>
                        <h:outputText
value="#{item.integranteIdintegrante}"/>
                    </h:column>
                    <h:column>
                        <h:commandLink value="Destroy"
action="#{manutencao.destroy}">
                            <f:param name="idmanutencao"
value="#{item.idmanutencao}"/>
                        </h:commandLink>
                        <h:outputText value=" "/>
                        <h:commandLink value="Edit"
action="#{manutencao.editSetup}">
                            <f:param name="idmanutencao"
value="#{item.idmanutencao}"/>
                        </h:commandLink>
                    </h:column>
                </h:dataTable>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.25 Manutenção/New.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>New Manutencao</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>New manutencao</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idmanutencao:"/>
                    <h:inputText id="idmanutencao"
value="#{manutencao.manutencao.idmanutencao}" title="Idmanutencao" />
                    <h:outputText value="Descricaomanutencao:"/>
                    <h:inputText id="descricaomanutencao"
value="#{manutencao.manutencao.descricaomanutencao}"
title="Descricaomanutencao" />
                    <h:outputText value="Datamanutencao (MM/dd/yyyy):"/>
                    <h:inputText id="datamanutencao"
value="#{manutencao.manutencao.datamanutencao}" title="Datamanutencao" >
```

```
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Datareparo (MM/dd/yyyy):"/>
                    <h:inputText id="datareparo"
value="#{manutencao.manutencao.datareparo}" title="Datareparo" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Urgenciamanutencao:"/>
                    <h:inputText id="urgenciamanutencao"
value="#{manutencao.manutencao.urgenciamanutencao}"
title="Urgenciamanutencao" />
                    <h:outputText value="Categoriamanutencao:"/>
                    <h:inputText id="categoriamanutencao"
value="#{manutencao.manutencao.categoriamanutencao}"
title="Categoriamanutencao" />
                    <h:outputText value="Obsmanutencao:"/>
                    <h:inputText id="obsmanutencao"
value="#{manutencao.manutencao.obsmanutencao}" title="Obsmanutencao" />
                    <h:outputText value="Kmatualintegrante:"/>
                    <h:inputText id="kmatualintegrante"
value="#{manutencao.manutencao.kmatualintegrante}"
title="Kmatualintegrante" />
                    <h:outputText value="AtividadeIdatividade:"
rendered="#{manutencao.manutencao.atividadeIdatividade == null}"/>
                    <h:selectOneMenu id="atividadeIdatividade"
value="#{manutencao.manutencao.atividadeIdatividade}"
title="AtividadeIdatividade"
rendered="#{manutencao.manutencao.atividadeIdatividade == null}">
                        <f:selectItems
value="#{manutencao.atividadeIdatividades}"/>
                    </h:selectOneMenu>
                    <h:outputText value="IntegranteIdintegrante:"
rendered="#{manutencao.manutencao.integranteIdintegrante == null}"/>
                    <h:selectOneMenu id="integranteIdintegrante"
value="#{manutencao.manutencao.integranteIdintegrante}"
title="IntegranteIdintegrante"
rendered="#{manutencao.manutencao.integranteIdintegrante == null}">
                        <f:selectItems
value="#{manutencao.integranteIdintegrantes}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{manutencao.create}"
value="Create"/>
                <br>
                <h:commandLink action="manutencao_list" value="Show All
Manutencao"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.26 Nota Fiscal de Gasto/Detail.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Detail of Notafiscalgasto</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Detail of notafiscalgasto</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idnotafiscalgasto:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.idnotafiscalgasto}"
title="Idnotafiscalgasto" />
                    <h:outputText value="Numeronotagasto:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.numeronotagasto}"
title="Numeronotagasto" />
                    <h:outputText value="Datagasto:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.datagasto}" title="Datagasto" />
                    <h:outputText value="Valornotagasto:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.valornotagasto}"
title="Valornotagasto" />
                    <h:outputText value="Datapreenchegasto:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.datapreenchegasto}"
title="Datapreenchegasto" />
                    <h:outputText value="FornecedorIdfornecedor:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.fornecedorIdfornecedor}"
title="FornecedorIdfornecedor" />
                </h:panelGrid>
                <h:commandLink action="notafiscalgasto_edit" value="Edit"
/>
                <br>
                <h:commandLink action="notafiscalgasto_list" value="Show
All Notafiscalgasto"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.27 Nota Fiscal Gasto/Edit.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Edit Notafiscalgasto</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Edit notafiscalgasto</h1>
            <h:form>
                <h:inputHidden value="#{notafiscalgasto.notafiscalgasto}"
immediate="true"/>
                <h:panelGrid columns="2">
                    <h:outputText value="Idnotafiscalgasto:"/>
                    <h:outputText
value="#{notafiscalgasto.notafiscalgasto.idnotafiscalgasto}"
title="Idnotafiscalgasto" />
                    <h:outputText value="Numeronotagasto:"/>
                    <h:inputText id="numeronotagasto"
value="#{notafiscalgasto.notafiscalgasto.numeronotagasto}"
title="Numeronotagasto" />
                    <h:outputText value="Datagasto (MM/dd/yyyy):"/>
                    <h:inputText id="datagasto"
value="#{notafiscalgasto.notafiscalgasto.datagasto}" title="Datagasto" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Valornotagasto:"/>
                    <h:inputText id="valornotagasto"
value="#{notafiscalgasto.notafiscalgasto.valornotagasto}"
title="Valornotagasto" />
                    <h:outputText value="Datapreenchegasto (MM/dd/yyyy):"/>
                    <h:inputText id="datapreenchegasto"
value="#{notafiscalgasto.notafiscalgasto.datapreenchegasto}"
title="Datapreenchegasto" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="FornecedorIdfornecedor:"/>
                    <h:selectOneMenu id="fornecedorIdfornecedor"
value="#{notafiscalgasto.notafiscalgasto.fornecedorIdfornecedor}"
title="FornecedorIdfornecedor">
                        <f:selectItems
value="#{notafiscalgasto.fornecedorIdfornecedors}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{notafiscalgasto.edit}"
value="Save"/>
                <br>
```

```
                <h:commandLink action="notafiscalgasto_list" value="Show
All Notafiscalgasto"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.28 Nota Fiscal Gasto/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>List Notafiscalgasto</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Listing Notafiscalgastos</h1>
            <h:form>
                <h:commandLink action="#{notafiscalgasto.createSetup}"
value="New Notafiscalgasto"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
                <br>
                <h:outputText value="Item #{notafiscalgasto.firstItem +
1}..#{notafiscalgasto.lastItem} of #{notafiscalgasto.itemCount}"/> 
                <h:commandLink action="#{notafiscalgasto.prev}"
value="Previous #{notafiscalgasto.batchSize}"
rendered="#{notafiscalgasto.firstItem >=
notafiscalgasto.batchSize}"/> 
                <h:commandLink action="#{notafiscalgasto.next}" value="Next
#{notafiscalgasto.batchSize}" rendered="#{notafiscalgasto.lastItem +
notafiscalgasto.batchSize <= notafiscalgasto.itemCount}"/> 
                <h:commandLink action="#{notafiscalgasto.next}"
value="Remaining #{notafiscalgasto.itemCount - notafiscalgasto.lastItem}"
                               rendered="#{notafiscalgasto.lastItem <
notafiscalgasto.itemCount && notafiscalgasto.lastItem +
notafiscalgasto.batchSize > notafiscalgasto.itemCount}"/>
                <h:dataTable value='#{notafiscalgasto.notafiscalgastos}'
var='item' border="1" cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Idnotafiscalgasto"/>
                        </f:facet>
                        <h:commandLink
action="#{notafiscalgasto.detailSetup}" value="#{item.idnotafiscalgasto}"/>
                    </h:column>
```

```
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Numeronotagasto"/>
                        </f:facet>
                        <h:outputText value="#{item.numeronotagasto}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datagasto"/>
                        </f:facet>
                        <h:outputText value="#{item.datagasto}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Valornotagasto"/>
                        </f:facet>
                        <h:outputText value="#{item.valornotagasto}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datapreenchegasto"/>
                        </f:facet>
                        <h:outputText value="#{item.datapreenchegasto}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="FornecedorIdfornecedor"/>
                        </f:facet>
                        <h:outputText
value="#{item.fornecedorIdfornecedor}"/>
                    </h:column>
                    <h:column>
                        <h:commandLink value="Destroy"
action="#{notafiscalgasto.destroy}">
                            <f:param name="idnotafiscalgasto"
value="#{item.idnotafiscalgasto}"/>
                        </h:commandLink>
                        <h:outputText value=" "/>
                        <h:commandLink value="Edit"
action="#{notafiscalgasto.editSetup}">
                            <f:param name="idnotafiscalgasto"
value="#{item.idnotafiscalgasto}"/>
                        </h:commandLink>
                    </h:column>
                </h:dataTable>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.29 Nota Fiscal Gasto/New.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>New Notafiscalgasto</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>New notafiscalgasto</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idnotafiscalgasto:"/>
                    <h:inputText id="idnotafiscalgasto"
value="#{notafiscalgasto.notafiscalgasto.idnotafiscalgasto}"
title="Idnotafiscalgasto" />
                    <h:outputText value="Numeronotagasto:"/>
                    <h:inputText id="numeronotagasto"
value="#{notafiscalgasto.notafiscalgasto.numeronotagasto}"
title="Numeronotagasto" />
                    <h:outputText value="Datagasto (MM/dd/yyyy):"/>
                    <h:inputText id="datagasto"
value="#{notafiscalgasto.notafiscalgasto.datagasto}" title="Datagasto" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Valornotagasto:"/>
                    <h:inputText id="valornotagasto"
value="#{notafiscalgasto.notafiscalgasto.valornotagasto}"
title="Valornotagasto" />
                    <h:outputText value="Datapreenchegasto (MM/dd/yyyy):"/>
                    <h:inputText id="datapreenchegasto"
value="#{notafiscalgasto.notafiscalgasto.datapreenchegasto}"
title="Datapreenchegasto" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="FornecedorIdfornecedor:"
rendered="#{notafiscalgasto.notafiscalgasto.fornecedorIdfornecedor ==
null}"/>
                    <h:selectOneMenu id="fornecedorIdfornecedor"
value="#{notafiscalgasto.notafiscalgasto.fornecedorIdfornecedor}"
title="FornecedorIdfornecedor"
rendered="#{notafiscalgasto.notafiscalgasto.fornecedorIdfornecedor ==
null}">
                        <f:selectItems
value="#{notafiscalgasto.fornecedorIdfornecedors}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
```

```
                <h:commandLink action="#{notafiscalgasto.create}"
value="Create"/>
                <br>
                <h:commandLink action="notafiscalgasto_list" value="Show
All Notafiscalgasto"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.30 Nota Fiscal Gasto Has Itens/Detail.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Detail of NotafiscalgastoHasItens</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Detail of notafiscalgastoHasItens</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="NotafiscalgastoHasItensPK:"/>
                    <h:outputText
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgastoHas
ItensPK}" title="NotafiscalgastoHasItensPK" />
                    <h:outputText value="Quantidadegasto:"/>
                    <h:outputText
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.quantidadegasto}"
title="Quantidadegasto" />
                    <h:outputText value="Valortotalgasto:"/>
                    <h:outputText
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.valortotalgasto}"
title="Valortotalgasto" />
                    <h:outputText value="Itens:"/>
                    <h:outputText
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.itens}"
title="Itens" />
                    <h:outputText value="Notafiscalgasto:"/>
                    <h:outputText
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgasto}"
title="Notafiscalgasto" />
                </h:panelGrid>
                <h:commandLink action="notafiscalgastoHasItens_edit"
value="Edit" />
                <br>
```

```
                <h:commandLink action="notafiscalgastoHasItens_list"
value="Show All NotafiscalgastoHasItens"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.31 Nota Fiscal Has Itens/Edit.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Edit NotafiscalgastoHasItens</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Edit notafiscalgastoHasItens</h1>
            <h:form>
                <h:inputHidden
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens}"
immediate="true"/>
                <h:panelGrid columns="2">
                    <h:outputText value="NotafiscalgastoHasItensPK:"/>
                    <h:outputText
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgastoHas
ItensPK}" title="NotafiscalgastoHasItensPK" />
                    <h:outputText value="Quantidadegasto:"/>
                    <h:inputText id="quantidadegasto"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.quantidadegasto}"
title="Quantidadegasto" />
                    <h:outputText value="Valortotalgasto:"/>
                    <h:inputText id="valortotalgasto"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.valortotalgasto}"
title="Valortotalgasto" />
                    <h:outputText value="Itens:"/>
                    <h:selectOneMenu id="itens"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.itens}"
title="Itens">
                        <f:selectItems
value="#{notafiscalgastoHasItens.itenss}"/>
                    </h:selectOneMenu>
                    <h:outputText value="Notafiscalgasto:"/>
                    <h:selectOneMenu id="notafiscalgasto"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgasto}"
title="Notafiscalgasto">
```

```
                        <f:selectItems
value="#{notafiscalgastoHasItens.notafiscalgastos}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{notafiscalgastoHasItens.edit}"
value="Save"/>
                <br>
                <h:commandLink action="notafiscalgastoHasItens_list"
value="Show All NotafiscalgastoHasItens"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.32 Nota Fiscal Gasto has Itens/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>List NotafiscalgastoHasItens</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Listing NotafiscalgastoHasItenss</h1>
            <h:form>
                <h:commandLink
action="#{notafiscalgastoHasItens.createSetup}" value="New
NotafiscalgastoHasItens"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
                <br>
                <h:outputText value="Item
#{notafiscalgastoHasItens.firstItem +
1}..#{notafiscalgastoHasItens.lastItem} of
#{notafiscalgastoHasItens.itemCount}"/> 
                <h:commandLink action="#{notafiscalgastoHasItens.prev}"
value="Previous #{notafiscalgastoHasItens.batchSize}"
rendered="#{notafiscalgastoHasItens.firstItem >=
notafiscalgastoHasItens.batchSize}"/> 
                <h:commandLink action="#{notafiscalgastoHasItens.next}"
value="Next #{notafiscalgastoHasItens.batchSize}"
rendered="#{notafiscalgastoHasItens.lastItem +
notafiscalgastoHasItens.batchSize <=
notafiscalgastoHasItens.itemCount}"/> 
```

```
                <h:commandLink action="#{notafiscalgastoHasItens.next}"
value="Remaining #{notafiscalgastoHasItens.itemCount -
notafiscalgastoHasItens.lastItem}"
                               rendered="#{notafiscalgastoHasItens.lastItem
< notafiscalgastoHasItens.itemCount && notafiscalgastoHasItens.lastItem +
notafiscalgastoHasItens.batchSize > notafiscalgastoHasItens.itemCount}"/>
                <h:dataTable
value='#{notafiscalgastoHasItens.notafiscalgastoHasItenss}' var='item'
border="1" cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText
value="NotafiscalgastoHasItensPK"/>
                        </f:facet>
                        <h:commandLink
action="#{notafiscalgastoHasItens.detailSetup}"
value="#{item.notafiscalgastoHasItensPK}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Quantidadegasto"/>
                        </f:facet>
                        <h:outputText value="#{item.quantidadegasto}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Valortotalgasto"/>
                        </f:facet>
                        <h:outputText value="#{item.valortotalgasto}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Itens"/>
                        </f:facet>
                        <h:outputText value="#{item.itens}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Notafiscalgasto"/>
                        </f:facet>
                        <h:outputText value="#{item.notafiscalgasto}"/>
                    </h:column>
                    <h:column>
                        <h:commandLink value="Destroy"
action="#{notafiscalgastoHasItens.destroy}">
                            <f:param name="notafiscalgastoHasItensPK"
value="#{item.notafiscalgastoHasItensPK}"/>
                        </h:commandLink>
                        <h:outputText value=" "/>
                        <h:commandLink value="Edit"
action="#{notafiscalgastoHasItens.editSetup}">
                            <f:param name="notafiscalgastoHasItensPK"
value="#{item.notafiscalgastoHasItensPK}"/>
                        </h:commandLink>
                    </h:column>
                </h:dataTable>
```

```
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.33 Nota Fiscal Gasto Has Itens/New.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>New NotafiscalgastoHasItens</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>New notafiscalgastoHasItens</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="NotafiscalgastoHasItensPK:"/>
                    <h:inputText id="notafiscalgastoHasItensPK"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgastoHas
ItensPK}" title="NotafiscalgastoHasItensPK" />
                    <h:outputText value="Quantidadegasto:"/>
                    <h:inputText id="quantidadegasto"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.quantidadegasto}"
title="Quantidadegasto" />
                    <h:outputText value="Valortotalgasto:"/>
                    <h:inputText id="valortotalgasto"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.valortotalgasto}"
title="Valortotalgasto" />
                    <h:outputText value="Itens:"
rendered="#{notafiscalgastoHasItens.notafiscalgastoHasItens.itens ==
null}"/>
                    <h:selectOneMenu id="itens"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.itens}"
title="Itens"
rendered="#{notafiscalgastoHasItens.notafiscalgastoHasItens.itens ==
null}">
                        <f:selectItems
value="#{notafiscalgastoHasItens.itenss}"/>
                    </h:selectOneMenu>
                    <h:outputText value="Notafiscalgasto:"
rendered="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgasto
== null}"/>
                    <h:selectOneMenu id="notafiscalgasto"
value="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgasto}"
title="Notafiscalgasto"
rendered="#{notafiscalgastoHasItens.notafiscalgastoHasItens.notafiscalgasto
== null}">
```

```
                        <f:selectItems
value="#{notafiscalgastoHasItens.notafiscalgastos}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{notafiscalgastoHasItens.create}"
value="Create"/>
                <br>
                <h:commandLink action="notafiscalgastoHasItens_list"
value="Show All NotafiscalgastoHasItens"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.34 Nota Serviço/Detail.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Detail of Notaservico</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Detail of notaservico</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idnotaservico:"/>
                    <h:outputText
value="#{notaservico.notaservico.idnotaservico}" title="Idnotaservico" />
                    <h:outputText value="Datanotaservico:"/>
                    <h:outputText
value="#{notaservico.notaservico.datanotaservico}" title="Datanotaservico"
/>
                    <h:outputText value="Datapreenchenotaservico:"/>
                    <h:outputText
value="#{notaservico.notaservico.datapreenchenotaservico}"
title="Datapreenchenotaservico" />
                    <h:outputText value="Statusmanutencao:"/>
                    <h:outputText
value="#{notaservico.notaservico.statusmanutencao}"
title="Statusmanutencao" />
                    <h:outputText value="Quantidadeservico:"/>
                    <h:outputText
value="#{notaservico.notaservico.quantidadeservico}"
title="Quantidadeservico" />
                    <h:outputText value="Naturezadoreparo:"/>
```

```
                    <h:outputText
value="#{notaservico.notaservico.naturezadoreparo}"
title="Naturezadoreparo" />
                    <h:outputText value="IntegranteIdintegrante:"/>
                    <h:outputText
value="#{notaservico.notaservico.integranteIdintegrante}"
title="IntegranteIdintegrante" />
                    <h:outputText value="NotafiscalgastoHasItens:"/>
                    <h:outputText
value="#{notaservico.notaservico.notafiscalgastoHasItens}"
title="NotafiscalgastoHasItens" />
                </h:panelGrid>
                <h:commandLink action="notaservico_edit" value="Edit" />
                <br>
                <h:commandLink action="notaservico_list" value="Show All
Notaservico"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.35 Nota Serviço/Edit.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>Edit Notaservico</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Edit notaservico</h1>
            <h:form>
                <h:inputHidden value="#{notaservico.notaservico}"
immediate="true"/>
                <h:panelGrid columns="2">
                    <h:outputText value="Idnotaservico:"/>
                    <h:outputText
value="#{notaservico.notaservico.idnotaservico}" title="Idnotaservico" />
                    <h:outputText value="Datanotaservico (MM/dd/yyyy):"/>
                    <h:inputText id="datanotaservico"
value="#{notaservico.notaservico.datanotaservico}" title="Datanotaservico"
>
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
```

```
                    <h:outputText value="Datapreenchenotaservico
(MM/dd/yyyy):"/>
                    <h:inputText id="datapreenchenotaservico"
value="#{notaservico.notaservico.datapreenchenotaservico}"
title="Datapreenchenotaservico" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Statusmanutencao:"/>
                    <h:inputText id="statusmanutencao"
value="#{notaservico.notaservico.statusmanutencao}"
title="Statusmanutencao" />
                    <h:outputText value="Quantidadeservico:"/>
                    <h:inputText id="quantidadeservico"
value="#{notaservico.notaservico.quantidadeservico}"
title="Quantidadeservico" />
                    <h:outputText value="Naturezadoreparo:"/>
                    <h:inputText id="naturezadoreparo"
value="#{notaservico.notaservico.naturezadoreparo}"
title="Naturezadoreparo" />
                    <h:outputText value="IntegranteIdintegrante:"/>
                    <h:selectOneMenu id="integranteIdintegrante"
value="#{notaservico.notaservico.integranteIdintegrante}"
title="IntegranteIdintegrante">
                        <f:selectItems
value="#{notaservico.integranteIdintegrantes}"/>
                    </h:selectOneMenu>
                    <h:outputText value="NotafiscalgastoHasItens:"/>
                    <h:selectOneMenu id="notafiscalgastoHasItens"
value="#{notaservico.notaservico.notafiscalgastoHasItens}"
title="NotafiscalgastoHasItens">
                        <f:selectItems
value="#{notaservico.notafiscalgastoHasItenss}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{notaservico.edit}" value="Save"/>
                <br>
                <h:commandLink action="notaservico_list" value="Show All
Notaservico"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.36 Nota Serviço/List.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
```

```
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>List Notaservico</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>Listing Notaservicos</h1>
            <h:form>
                <h:commandLink action="#{notaservico.createSetup}"
value="New Notaservico"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
                <br>
                <h:outputText value="Item #{notaservico.firstItem +
1}..#{notaservico.lastItem} of #{notaservico.itemCount}"/> 
                <h:commandLink action="#{notaservico.prev}" value="Previous
#{notaservico.batchSize}" rendered="#{notaservico.firstItem >=
notaservico.batchSize}"/> 
                <h:commandLink action="#{notaservico.next}" value="Next
#{notaservico.batchSize}" rendered="#{notaservico.lastItem +
notaservico.batchSize <= notaservico.itemCount}"/> 
                <h:commandLink action="#{notaservico.next}"
value="Remaining #{notaservico.itemCount - notaservico.lastItem}"
                               rendered="#{notaservico.lastItem <
notaservico.itemCount && notaservico.lastItem + notaservico.batchSize >
notaservico.itemCount}"/>
                <h:dataTable value='#{notaservico.notaservicos}' var='item'
border="1" cellpadding="2" cellspacing="0">
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Idnotaservico"/>
                        </f:facet>
                        <h:commandLink action="#{notaservico.detailSetup}"
value="#{item.idnotaservico}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datanotaservico"/>
                        </f:facet>
                        <h:outputText value="#{item.datanotaservico}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Datapreenchenotaservico"/>
                        </f:facet>
                        <h:outputText
value="#{item.datapreenchenotaservico}">
                            <f:convertDateTime type="DATE"
pattern="MM/dd/yyyy" />
                        </h:outputText>
                    </h:column>
```

```
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Statusmanutencao"/>
                        </f:facet>
                        <h:outputText value="#{item.statusmanutencao}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Quantidadeservico"/>
                        </f:facet>
                        <h:outputText value="#{item.quantidadeservico}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="Naturezadoreparo"/>
                        </f:facet>
                        <h:outputText value="#{item.naturezadoreparo}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="IntegranteIdintegrante"/>
                        </f:facet>
                        <h:outputText
value="#{item.integranteIdintegrante}"/>
                    </h:column>
                    <h:column>
                        <f:facet name="header">
                            <h:outputText value="NotafiscalgastoHasItens"/>
                        </f:facet>
                        <h:outputText
value="#{item.notafiscalgastoHasItens}"/>
                    </h:column>
                    <h:column>
                        <h:commandLink value="Destroy"
action="#{notaservico.destroy}">
                            <f:param name="idnotaservico"
value="#{item.idnotaservico}"/>
                        </h:commandLink>
                        <h:outputText value=" "/>
                        <h:commandLink value="Edit"
action="#{notaservico.editSetup}">
                            <f:param name="idnotaservico"
value="#{item.idnotaservico}"/>
                        </h:commandLink>
                    </h:column>
                </h:dataTable>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.37 Nota Serviço/New.jsp

```
<%@page contentType="text/html"%>
<%@page pageEncoding="UTF-8"%>
```

```
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/>
        <title>New Notaservico</title>
    </head>
    <body>
        <f:view>
            <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            <h1>New notaservico</h1>
            <h:form>
                <h:panelGrid columns="2">
                    <h:outputText value="Idnotaservico:"/>
                    <h:inputText id="idnotaservico"
value="#{notaservico.notaservico.idnotaservico}" title="Idnotaservico" />
                    <h:outputText value="Datanotaservico (MM/dd/yyyy):"/>
                    <h:inputText id="datanotaservico"
value="#{notaservico.notaservico.datanotaservico}" title="Datanotaservico"
>
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Datapreenchenotaservico
(MM/dd/yyyy):"/>
                    <h:inputText id="datapreenchenotaservico"
value="#{notaservico.notaservico.datapreenchenotaservico}"
title="Datapreenchenotaservico" >
                        <f:convertDateTime type="DATE" pattern="MM/dd/yyyy"
/>
                    </h:inputText>
                    <h:outputText value="Statusmanutencao:"/>
                    <h:inputText id="statusmanutencao"
value="#{notaservico.notaservico.statusmanutencao}"
title="Statusmanutencao" />
                    <h:outputText value="Quantidadeservico:"/>
                    <h:inputText id="quantidadeservico"
value="#{notaservico.notaservico.quantidadeservico}"
title="Quantidadeservico" />
                    <h:outputText value="Naturezadoreparo:"/>
                    <h:inputText id="naturezadoreparo"
value="#{notaservico.notaservico.naturezadoreparo}"
title="Naturezadoreparo" />
                    <h:outputText value="IntegranteIdintegrante:"
rendered="#{notaservico.notaservico.integranteIdintegrante == null}"/>
                    <h:selectOneMenu id="integranteIdintegrante"
value="#{notaservico.notaservico.integranteIdintegrante}"
title="IntegranteIdintegrante"
rendered="#{notaservico.notaservico.integranteIdintegrante == null}">
                        <f:selectItems
value="#{notaservico.integranteIdintegrantes}"/>
                    </h:selectOneMenu>
                    <h:outputText value="NotafiscalgastoHasItens:"
rendered="#{notaservico.notaservico.notafiscalgastoHasItens == null}"/>
```

```
                    <h:selectOneMenu id="notafiscalgastoHasItens"
value="#{notaservico.notaservico.notafiscalgastoHasItens}"
title="NotafiscalgastoHasItens"
rendered="#{notaservico.notaservico.notafiscalgastoHasItens == null}">
                        <f:selectItems
value="#{notaservico.notafiscalgastoHasItenss}"/>
                    </h:selectOneMenu>
                </h:panelGrid>
                <h:commandLink action="#{notaservico.create}"
value="Create"/>
                <br>
                <h:commandLink action="notaservico_list" value="Show All
Notaservico"/>
                <br>
                <a href="/FrotaControlesVD/index.jsp">Back to index</a>
            </h:form>
        </f:view>
    </body>
</html>
```

11.2.38 CSS/viadigital.css

```
/*

 * Via Digital - http://viadigital.org.br/

 *

 * Arquivo de CSS padrÃ£o dos componentes do Via Digital.

 * Gerenciado pelo projeto VIA-CSS do Via Gigital -
http://repositorio.viadigital.org.br/projects/via-css/

 *
```

```
*  Developer: Carlos Henrique Pereira

*  Email:floripahenrique@gmail.com

*  Institution: GeNESS - http://www.geness.ufsc.br/

*

*  Date: quinta, 09 de agosto de 2007

*

*  License: GNU General Public License - http://www.gnu.org/copyleft/gpl.html/

*

*/




/*

 * todos os div's do site
```

```
 */



#global{

}



#barratitulo{

        position: relative;

        top: 10px;

        left: 0px;

        width: 955px;

        height: 68px;

        background-color: #edb00a;

      background-repeat: no-repeat;
```

```
        background-image: url(../imagens/icone.PNG) ;

        background-position: left;



 margin-bottom: 15px;

}



#corpo{

        position: relative;

}



#menulateral{

        position:absolute;

        top: 35px;
```

```
        left: 10px;

        width: 220px;

        height: 563px;

        border: 0px solid #000;

        background-color: #f1f1f1;

        padding-top: 30px;

        padding-left: 5px;

}



#menusuperior{

        position:absolute;

        top: 10px;
```

```
        left: 0px;

        width: 795px;

        height: 25px;

        padding-top: 5px;



}



#naologin{

        padding-top: 5px;

        padding-left: 3px;

        position:absolute;

        top: 36px;

        left: 240px;
```

```
        width: 650px;

        height: 30px;

        background-color: #b7673c;

}



#login{

        padding-top: 5px;

        padding-left: 3px;

        position:absolute;

        top: 36px;

        left: 240px;

        width: 650px;

        height: 30px;
```

```
        background-color: #eeeeee;
}

#textoviadigital{
        position:absolute;
        top: 67px;
        left: 240px;
        width: 325px;
        height: 450px;
}

#textocomponente{
        position:absolute;
```

```
        top: 67px;

        left: 565px;

        width: 325px;

        height: 450px;

}



#logoprefeitura{

        padding-left: 5px;

        position:absolute;

        top: 10px;

        left: 800px;

        width: 150px;

        height: 56px;
```

```
}



#parceirosprefeitura{

        position:absolute;

        top: 67px;

        left: 800px;

        width: 145px;

        height: 450px;

        background-color: #f1f1f1;

        padding: 5px 5px 5px 5px;

}



#parceiros{
```

```
        position:absolute;

        top: 523px;

        left: 0px;

    background-position:center;

        width: 780px;

        height: 80px;

        background-image: url(../imagens/parceiros.png) ;

        background-repeat: no-repeat;

}



#manipulacaodadoscomponente{



     margin:0 auto;
```

```
        text-align:left;

            position: absolute;

            top: 67px;

            left: 0px;

            width: 780px;

            height: 450px;

            background-color: #f1f1f1;

            padding: 0 5px 5px 10px;

}



#tituloaplicacao{

            position: relative;

            background-position:center;
```

```
        width: 775px;

        height: 35px;

        padding-top: 9px;

        padding-left: 20px;

        background-color:#b2b2b2;

        background-position: left top;

    background-repeat:repeat-x;

        margin-left: -10px;



}



/*

 * titulos do div barratitulo
```

```
 */



#sitetitulo{

        position: absolute;

        font-size: 30px;

      color: #fff;

        text-transform: uppercase;

        display: inline;

        margin-left: 250px;

        margin-top: 16px;

        font-family: Verdana, Arial, Helvetica,       sans-serif;

      background-repeat:repeat-x;
```

```
}

#sitesubtitulo{
        position: absolute;
        font-size: 15px;
        color: #fff;
        text-transform: uppercase;
        display: inline;
        margin-left: 400px;
        margin-top: 26px;
        font-family: Verdana, Arial, Helvetica,        sans-serif;
      background-repeat:repeat-x;
```

```
}



/*

 * menu lateral do div menulateral

 */



#menu2{

        display: block;

        font-size: 20px;

        margin-left: 5px;

        background-image: url(../imagens/menu2-block.png) ;

        background-position: -10px 0px;

        background-repeat: no-repeat;
```

```
        color: #aaa;

        font-family: Verdana, Arial, Helvetica,         sans-serif;

        font-size: 18px;

        font-weight: bold;

        font-style: bold;



}



#menu2 a{

        display: block;

        font-size: 20px;

        margin-left: 5px;
```

```
        text-decoration: none;




        color: #aaa;

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        font-size: 18px;

        font-weight: bold;

}



#menu2:hover, #menu2 a:hover {

        color: #666666;

}



#menu2 #menu2 {
```

```
        display: none;

        margin-left: 15px;

        text-decoration: none;

        background-image: url() ;

        background-position: top;

        background-repeat: no-repeat;

}



#menu2:hover #menu2  {

        display: block;

}
```

```
/*

 * menu superior do div menusuperior

 */



#menu{

        padding: 5px 5px 5px 10px;

        position: relative;

        display: block;

        color: #FF6600;

        display: inline;

        border-left: 0px solid #c0c0c0;

        background-image: url(../imagens/menu_superior.png) ;

        background-position: left;
```

```
        background-repeat: no-repeat;

}



#menu:hover, #menu #menu li:hover {

        color: #edb001;

}



#menu #menu {

        border: 0px;

        position: absolute;

        left: 3px;

        top: 3px;

        *left: -40px;
```

```
        *top: 20px;

        display: none;

        padding-left: 0;

        background-image: url() ;

}



#menu #menu:hover {

        color: #FF6600;

}



#menu #menu li{

        padding: 5px 3px 3px 5px;

        list-style-type: none;
```

```
        background-color: #fff;

        border: 1px solid #bbb;

        table-layout: inherit;

}



#menu:hover #menu  {

        display: block;

}



/*

 * todos os link do site

 */
```

```
a{
        color: #ff6600;
}

a:focus{
        color: #edb001;
}

a:hover{
        color: #edb001;
}

/*
```

```
 * todos os botoes do site

 */



#botao2, .botao2{

        background-image: url(../imagens/botao1.png) ;

        color: #b2b2b2;

        font-family: Verdana, Arial, Helvetica,         sans-serif;

        font-size: 13px;

        display:list-item;

}

#botao, .botao{

background: #edb001;

color:#FFFFFF;
```

```
font-weight: bold;

}

#botao1, .botao1 {

        background: #edb001;

        border:0px solid #ffffff;

        color:#ffffff;

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        font-size: 13px;

}



/*

 * os titulos e os textos do site

 */
```

```
h1, h2, h3, h4, p{

        font-family: Verdana, Arial, Helvetica,         sans-serif;

}



#tituloaplicacao, .tituloaplicacao{

        margin-top: 0px;

        color: #FFFFFF;

        font-family: Verdana, Arial, Helvetica, sans-serif;

        font-size: 14px; font-weight: bold;

        background-position:center;

}
```

```
#titulo1, #titulooperacao, .titulo1, .titulooperacao{

        color: #edb00a;

        font-family: Verdana, Arial, Helvetica,         sans-serif;

        font-size: 20px;

        display: inline;

        display:list-item;

        list-style-type: none;

        margin: 20px 10px 15px 20px;



}

#titulo2, #titulooperacao2, .titulo2, .titulooperacao2{

        color: #ff6600;

        font-family: Verdana, Arial, Helvetica,         sans-serif;
```

```
        font-size: 20px;

        display: inline;

        display:list-item;

        list-style-type: none;

        margin: 20px 10px 15px 20px;

        font-style: italic;

}



#texto1, .texto1{

        color: #b2b2b2;

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        font-size: 13px;
```

```
        display: inline;

        display:list-item;

        list-style-type: none;

        margin: 10px 10px 5px 20px;

}



#texto2, #textonaologin, .texto2, .textonaologin{

        padding: 5px 5px 5px 7px;

        color: #fff;

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        font-size: 13px;

}
```

```
#textologin, .textologin{

        padding: 5px 5px 5px 7px;

        color: #666666;

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        font-size: 13px;

}



#texto3, .texto3{

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        color: #666666;

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        font-size: 13px;

        display: inline;
```

```
        display:list-item;

        list-style-type: none;

}

#texto4, .texto4{

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        color: #edb001;

}

#texto5, .texto5{

        font-family: Verdana, Arial, Helvetica,        sans-serif;

        color: #ff6600;

}

#texto6, .texto6{

        font-family: Verdana, Arial, Helvetica,        sans-serif;
```

```
        color: #b2b2b2;

}




/*

 * texto do div naologin

 */



#textonaologin{

        padding: 5px 5px 5px 7px;

        color: #fff;

        font-family: Verdana, Arial, Helvetica,         sans-serif;

        font-size: 13px;
```

```
}

/*

.dr-toolbar-ext {

background-color:#edb00a;

background-image:url(../imagens/barramenu.png);

background-position:left top;

background-repeat: repeat-x;

border:1px solid #C0C0C0;

padding:0px;

height: 25px;

width: 795px;
```

```
}

.dr-toolbar-int {

color:#FFFFFF;

font-family:Arial,Verdana,sans-serif;

font-size:11px;

font-weight:bold;

padding:2px 10px;

white-space:nowrap;

}



.dr-menu-list-bg {

background-image:url(../imagens/menu_list.png);

border-left-color:#FFFFFF;
```

```
border-right-color:#FFFFFF;

border-top-color:#FFFFFF;

}

.dr-menu-list-border {

background-color:#FFFFFF;

border:1px solid #FFFFFF;

float:left;

position:absolute;

}

.dr-table-subheadercell {

background-color:#fffe96;

}
```

```
.dr-table-subheader {

background-color:#fffe96;

}

.dr-dscr {

}

.dr-dscr-t {

background-color:#FFFFFF;

border:1px solid #C0C0C0;

}

.dr-dscr-button {

background-color:#fffe96;

border-color:#C0C0C0;

font-family:Arial,Verdana,sans-serif;
```

```
font-size:11px;

}

.dr-dscr-act {

border-top:2px solid #FF6600;

color:#FF6600;

font-family:verdana;

font-size:11px;

width:25px;

}



.dr-dscr-inact {

border-top:2px solid #000000;

color:#000000;
```

```
cursor:pointer;

font-family:verdana;

font-size:11px;

width:25px;

}

.dr-stglpnl {

background-color:#FFFFFF;

border-color:#C0C0C0;

}

.dr-mpnl-header

{
```

```
        background-color:#edb00a;




        }


        #headerModal


        {




        background-color:#edb00a;




        }*/
```

11.2.39 CSS/CSScabecalho.css

```
/*
    Document   : ControleFrotas
    Created on : 25/07/2007, 19:44
    Author     : Alecindro
    Description:
        Purpose of the stylesheet follows.
*/

/*
   TODO customize this sample style
```

```
   Syntax recommendation http://www.w3.org/TR/REC-CSS2/
*/

root {
    display: block;
}

.header {
color: #ffffff;
      font-size: 14px;
        font-weight: bold;
      padding-left: 15px;
      padding-top: 20px;
}
.blog {
      text-align:justify;
}

form {
      display:inline;
      margin: 0px;
      padding: 0px;
}

.clr {
      clear: both;

}

.title {
      color: #ffffff;
      font-size: 40px;
      padding-left: 15px;
      padding-top: 20px;
}

.error {
  font-style: italic;
  text-transform: uppercase;
  padding: 5px;
  color: #cccccc;
  font-size: 14px;
  font-weight: bold;
}


.pagenav {
      align: center;
  font-size: 11px;
  font-weight: bold;
  border: 1px solid #cccccc;
  width: auto;
  padding: 4px;
  margin: 1px;
}
```

```
.pagenavbar {
	padding-right: 5px;
	float: right;
}

ul
{
margin: 0;
padding: 0;
list-style: none;
}

li
{
line-height: 15px;
padding-left: 10px;
padding-top: 0px;
background-image: url(../../../images/M_images/arrow.png) ;
background-repeat: no-repeat;
background-position: 0px 3px;
}

.latestnews
{
	font-family: Verdana, Helvetica, Sans Serif;
	font-size: 10px;
	color: #333333;
	  font-weight: normal;
}

td {
	font-size: 11px;
}



body {
	margin: 15px;
	height: 100%;
	padding: 0px;
	font-family: Verdana, Helvetica, Sans Serif;
	font-size: 10px;
	color: #333333;
	background: #dadada
url(../../../images/stories/viadigital_imagens/fundo.jpg) top repeat-x;
}

/* mambo core stuff */
a:link, a:visited {
	color: #333333; text-decoration: none;
	font-weight: bold;
}

a:hover {
	color: #000000;   text-decoration: none;
	font-weight: bold;
```

```
}

.button {
	padding: 2px 5px 2px 5px;
	color: #333333;
	font-family: Arial, Helvetica, Sans Serif;
	text-align: center;
	font-size: 11px;
	font-weight: bold;
	margin-top: 5px;
}

.inputbox {
	padding: 2px;
	border:solid 1px #cccccc;
	background-color: #ffffff;
}

.searchbox {
	border: 0px solid #4B4B4B;
	background-color: transparent;
	color: #3E3E3E;
}

.contentpane {
}

.contentpaneopen {
width: 100%;
}

.contentheading {
	display: block;
	background: #ebebeb;
	width: 100%;
	color: #666666;
	text-align: left;
	font-weight: bold;
	font-size: 12px;
	white-space: nowrap;
	border-bottom: px solid #ebebeb;
	padding: 3px;
	text-transform: uppercase;
	letter-spacing: 2px;
}

.contentpagetitle {
	font-size: 13px;
	font-weight: bold;
	color: #CCCCCC;
	text-align:left;
}

.componentheading{
	font-weight: bold;
	color: #000000;
```

```
      font-size: 15px;
      width: 100%;
      text-align: center;
}


table.moduletable {
      width: 100%;
      margin-bottom: 5px;
      padding: 0px;
      border-spacing: 0px;
      border-collapse: collapse;
}

table.moduletable th {
      color: #ffffff;
      text-align: center;
      padding-top: 4px;
      padding-left: 0px;
      height: 21px;
      font-weight: bolder;
      font-size: 11px;
        text-transform: uppercase;
        text-align: left;
        padding-left: 6px;
        background: #757575;
      border: 0px solid #FFFFFF;
}

table.moduletable td {
      font-size: 11px;
      padding: 0px;
      margin: 0px;
      font-weight: normal;
}

table.moduletable-header {
  width: 100%;
  border: 1px solid #999999;
}

table.pollstableborder td {
  padding: 2px;
}

.sectiontableheader {
  font-weight: bold;
  background: #f0f0f0;
  padding: 4px;
}

.sectiontablefooter {
}

.sectiontableentry1 {
      background-color : #ffffff;
```

```
}

.sectiontableentry2 {
      background-color : #e6e6e6;
}

.small {
color: #999999;
      font-size: 11px;
      font-weight: bold;
        letter-spacing: -1px;
}

.small2 {
      color: #999999;
      font-size: 11px;
        letter-spacing: -1px;
}

.createdate {
      height: 15px;
      padding-bottom: 10px;
      color: #999999;
      font-size: 11px;
      font-weight: bold;
}

.modifydate {
      height: 15px;
      padding-top: 10px;
      color: #999999;
      font-size: 11px;
      font-weight: bold;
}

table.contenttoc {
  border: 1px solid #cccccc;
  padding: 2px;
  margin-left: 2px;
  margin-bottom: 2px;
}

table.contenttoc th {
  background: url(../images/subhead_bg.png) repeat-x;
  color: #666666;
      text-align: left;
      padding-top: 2px;
      padding-left: 4px;
      height: 21px;
      font-weight: bold;
      font-size: 10px;
      text-transform: uppercase;
}

a.int {
      display: block;
```

```
      vertical-align: middle;
      font-size: 11px;
      color: #ffffff;
      font-weight: bolder;
      text-align: left;
      line-height: 20px;
      width: 100%;
      text-decoration: none;
      background: #9d9d9d;
      border-top: 1px solid White;
      border-left: 1px solid White;
      border-bottom: 1px solid #acacac;
      border-right: 1px solid #acacac;
      text-indent: 3px;
      height: 20px;
}

a.int:hover {
      display: block;
      vertical-align: middle;
      font-size: 11px;
      font-weight: bolder;
      color: #9d9d9d;
      text-align: left;
      padding-top: 0px;
      width: 100%;
      text-decoration: none;
      background: #efefef;
      border-top: 1px solid #CCCCCC;
      border-left: 1px solid #CCCCCC;
      border-bottom: 1px solid #CCCCCC;
      border-right: 1px solid #CCCCCC;
      height: 20px;
      text-indent: 3px;
}

a.mainlevel:link, a.mainlevel:visited {
      display: block;
      vertical-align: middle;
      font-size: 10px;
      font-weight: bold;
      color: #808080;
      text-align: left;
      line-height: 20px;
  width: 100%;
      text-decoration: none;
      background: #efefef;
      border-top: 1px solid White;
  border-left: 1px solid White;
  border-bottom: 1px solid #acacac;
  border-right: 1px solid #acacac;
      text-indent: 3px;
      height: 20px;
}

a.mainlevel:hover {
```

```
      display: block;
      vertical-align: middle;
      font-size: 10px;
      font-weight: bold;
      color: White;
      text-align: left;
      padding-top: 0px;
      width: 100%;
      text-decoration: none;
      background: #333333;
      border-top: 1px solid #CCCCCC;
  border-left: 1px solid #CCCCCC;
  border-bottom: 1px solid #CCCCCC;
  border-right: 1px solid #CCCCCC;
      height: 20px;
      text-indent: 3px;
      line-height: 20px;
}

a.sublevel:link, a.sublevel:visited {
      padding-left: 1px;
      vertical-align: middle;
      font-size: 11px;
      font-weight: bold;
      color: #ff6600;
      text-align: left;
}

a.sublevel:hover {
      color: #ffcc00;
      text-decoration: none;
}
```

11.2.40 frota.skin.properties – configuração do Skin dos componentes RichFaces

headerBackgroundColor #BED6F8

headerGradientColor #F2F7FF

headTextColor #000000

headerWeightFont bold

generalBackgroundColor #FFFFFF

generalTextColor= #000000

generalSizeFont= 11px

generalFamilyFont= Arial, Verdana, sans-serif

controlTextColor= #000000

controlBackgroundColor= #ffffff

additionalBackgroundColor= #ECF4FE

shadowBackgroundColor= #000000

shadowOpacity= 1

panelBorderColor= #BED6F8

subBorderColor= #ffffff

tabBackgroundColor= #C6DEFF

tabDisabledTextColor= #8DB7F3

trimColor= #D6E6FB

tipBackgroundColor= #FAE6B0

tipBorderColor= #E5973E

selectControlColor =#E79A00

generalLinkColor =#0078D0

hoverLinkColor =#0090FF

visitedLinkColor= #0090FF

headerSizeFont =11px

headerFamilyFont= Arial, Verdana, sans-serif

tabSizeFont =11px

tabFamilyFont =Arial, Verdana, sans-serif

buttonSizeFont =11px

buttonFamilyFont= Arial, Verdana, sans-serif

tableBackgroundColor= #FFFFFF

tableFooterBackgroundColor =#cccccc

tableSubfooterBackgroundColor =#f1f1f1

tableBorderColor =#C0C0C0

11.2.41 WEB-INF/faces-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE faces-config PUBLIC "-//Sun Microsystems, Inc.//DTD JavaServer
Faces Config 1.1//EN" "http://java.sun.com/dtd/web-facesconfig_1_1.dtd">
<!-- ========== FULL CONFIGURATION FILE ================================
-->
<faces-config xmlns="http://java.sun.com/JSF/Configuration">
  <converter>
    <converter-for-class>frotaControle.model.Atividade</converter-for-
class>
    <converter-class>frotaControle.controle.AtividadeConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-
class>frotaControle.model.Categoriahabilitacao</converter-for-class>
    <converter-
class>frotaControle.controle.CategoriahabilitacaoConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Fornecedor</converter-for-
class>
    <converter-class>frotaControle.controle.FornecedorConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Grupointegrante</converter-
for-class>
    <converter-
class>frotaControle.controle.GrupointegranteConverter</converter-class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Integrante</converter-for-
class>
    <converter-class>frotaControle.controle.IntegranteConverter</converter-
class>
  </converter>
```

```
  <converter>
    <converter-for-class>frotaControle.model.Itens</converter-for-class>
    <converter-class>frotaControle.controle.ItensConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Manutencao</converter-for-
class>
    <converter-class>frotaControle.controle.ManutencaoConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Motoristaoperador</converter-
for-class>
    <converter-
class>frotaControle.controle.MotoristaoperadorConverter</converter-class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Notafiscalgasto</converter-
for-class>
    <converter-
class>frotaControle.controle.NotafiscalgastoConverter</converter-class>
  </converter>
  <converter>
    <converter-for-
class>frotaControle.model.NotafiscalgastoHasItens</converter-for-class>
    <converter-
class>frotaControle.controle.NotafiscalgastoHasItensConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-
class>frotaControle.model.NotafiscalgastoHasItensPK</converter-for-class>
    <converter-
class>frotaControle.controle.NotafiscalgastoHasItensPKConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Notaservico</converter-for-
class>
    <converter-
class>frotaControle.controle.NotaservicoConverter</converter-class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Perfilusuario</converter-for-
class>
    <converter-
class>frotaControle.controle.PerfilusuarioConverter</converter-class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Tipocombustivel</converter-
for-class>
    <converter-
class>frotaControle.controle.TipocombustivelConverter</converter-class>
  </converter>
  <converter>
```

```
    <converter-for-class>frotaControle.model.Unidade</converter-for-class>
    <converter-class>frotaControle.controle.UnidadeConverter</converter-
class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Usuariosistema</converter-for-
class>
    <converter-
class>frotaControle.controle.UsuariosistemaConverter</converter-class>
  </converter>
  <converter>
    <converter-for-
class>frotaControle.model.Subgrupointegrantes</converter-for-class>
    <converter-
class>frotaControle.controle.SubgrupointegrantesConverter</converter-class>
  </converter>
  <converter>
    <converter-for-class>frotaControle.model.Integrante</converter-for-
class>
    <converter-class>frotaControle.controle.IntegranteConverter</converter-
class>
  </converter>
   <managed-bean>
    <managed-bean-name>navegacao</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.NavegacaoController</managed-bean-class>
    <managed-bean-scope>aplication</managed-bean-scope>
  </managed-bean>

  <managed-bean>
    <managed-bean-name>atividade</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.AtividadeController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>categoriahabilitacao</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.CategoriahabilitacaoController</managed-bean-
class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>fornecedor</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.FornecedorController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>grupointegrante</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.GrupointegranteController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>integrante</managed-bean-name>
```

```
    <managed-bean-
class>frotaControle.controle.IntegranteController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>itens</managed-bean-name>
    <managed-bean-class>frotaControle.controle.ItensController</managed-
bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>manutencao</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.ManutencaoController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>motoristaoperador</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.MotoristaoperadorController</managed-bean-
class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>notafiscalgasto</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.NotafiscalgastoController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>notafiscalgastoHasItens</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.NotafiscalgastoHasItensController</managed-
bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>notaservico</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.NotaservicoController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>perfilusuario</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.PerfilusuarioController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>tipocombustivel</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.TipocombustivelController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>unidade</managed-bean-name>
```

```
    <managed-bean-class>frotaControle.controle.UnidadeController</managed-
bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>uploadlistener</managed-bean-name>
    <managed-bean-class>frotaControle.controle.UploadListener</managed-
bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>usuariosistema</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.UsuariosistemaController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>subgrupointegrantes</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.SubgrupointegrantesController</managed-bean-
class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <managed-bean>
    <managed-bean-name>integrante</managed-bean-name>
    <managed-bean-
class>frotaControle.controle.IntegranteController</managed-bean-class>
    <managed-bean-scope>session</managed-bean-scope>
  </managed-bean>
  <navigation-rule>
    <navigation-case>
      <from-outcome>atividade_create</from-outcome>
      <to-view-id>/atividade/New.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>atividade_list</from-outcome>
      <to-view-id>/atividade/List.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>atividade_edit</from-outcome>
      <to-view-id>/atividade/Edit.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>atividade_detail</from-outcome>
      <to-view-id>/atividade/Detail.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>categoriahabilitacao_list</from-outcome>
```

```
    <to-view-id>/categoriahabilitacao/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>fornecedor_list</from-outcome>
    <to-view-id>/fornecedor/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>grupointegrante_list</from-outcome>
    <to-view-id>/grupointegrante/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>integrante_create</from-outcome>
    <to-view-id>/integrante/New.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>integrante_list</from-outcome>
    <to-view-id>/integrante/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>integrante_edit</from-outcome>
    <to-view-id>/integrante/Edit.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>integrante_detail</from-outcome>
    <to-view-id>/integrante/Detail.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>itens_list</from-outcome>
    <to-view-id>/itens/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>manutencao_create</from-outcome>
    <to-view-id>/manutencao/New.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>manutencao_list</from-outcome>
    <to-view-id>/manutencao/List.jsp</to-view-id>
  </navigation-case>
```

```
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>manutencao_edit</from-outcome>
    <to-view-id>/manutencao/Edit.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>manutencao_detail</from-outcome>
    <to-view-id>/manutencao/Detail.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>motoristaoperador_list</from-outcome>
    <to-view-id>/motoristaoperador/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgasto_create</from-outcome>
    <to-view-id>/notafiscalgasto/New.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgasto_list</from-outcome>
    <to-view-id>/notafiscalgasto/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgasto_edit</from-outcome>
    <to-view-id>/notafiscalgasto/Edit.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgasto_detail</from-outcome>
    <to-view-id>/notafiscalgasto/Detail.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgastoHasItens_create</from-outcome>
    <to-view-id>/notafiscalgastoHasItens/New.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgastoHasItens_list</from-outcome>
    <to-view-id>/notafiscalgastoHasItens/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
```

```
  <navigation-case>
    <from-outcome>notafiscalgastoHasItens_edit</from-outcome>
    <to-view-id>/notafiscalgastoHasItens/Edit.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notafiscalgastoHasItens_detail</from-outcome>
    <to-view-id>/notafiscalgastoHasItens/Detail.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notaservico_create</from-outcome>
    <to-view-id>/notaservico/New.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notaservico_list</from-outcome>
    <to-view-id>/notaservico/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notaservico_edit</from-outcome>
    <to-view-id>/notaservico/Edit.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>notaservico_detail</from-outcome>
    <to-view-id>/notaservico/Detail.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>perfilusuario_list</from-outcome>
    <to-view-id>/perfilusuario/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>tipocombustivel_list</from-outcome>
    <to-view-id>/tipocombustivel/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>unidade_list</from-outcome>
    <to-view-id>/unidade/List.jsp</to-view-id>
  </navigation-case>
</navigation-rule>
<navigation-rule>
  <navigation-case>
    <from-outcome>usuariosistema_list</from-outcome>
```

```
      <to-view-id>/usuariosistema/List.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>inicio</from-outcome>
      <to-view-id>/faces/Inicio.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>subgrupointegrantes_list</from-outcome>
      <to-view-id>/subgrupointegrantes/List.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>integrante_create</from-outcome>
      <to-view-id>/integrante/New.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>integrante_list</from-outcome>
      <to-view-id>/integrante/List.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>integrante_edit</from-outcome>
      <to-view-id>/integrante/Edit.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  <navigation-rule>
    <navigation-case>
      <from-outcome>integrante_detail</from-outcome>
      <to-view-id>/integrante/Detail.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
   <navigation-rule>
         <navigation-case>
      <from-outcome>view_inicio</from-outcome>
      <to-view-id>/Inicio.jsp</to-view-id>
    </navigation-case>
  </navigation-rule>
  </faces-config>
```

11.2.42 WEB-INF/web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">

    <context-param>
```

```
        <param-name>com.sun.faces.verifyObjects</param-name>
        <param-value>false</param-value>
    </context-param>
    <context-param>
        <param-name>com.sun.faces.validateXml</param-name>
        <param-value>true</param-value>
    </context-param>
    <context-param>
        <param-name>javax.faces.STATE_SAVING_METHOD</param-name>
        <param-value>client</param-value>
    </context-param>
    <context-param>
    <param-name>org.ajax4jsf.SKIN</param-name>
  <param-value>frota</param-value>
 </context-param>
  <context-param>
        <description>This parameter tells MyFaces if javascript code should
be allowed in the
            rendered HTML output.
            If javascript is allowed, command_link anchors will have
javascript code
            that submits the corresponding form.
            If javascript is not allowed, the state saving info and nested
parameters
            will be added as url parameters.
        Default: "true"</description>
        <param-name>org.apache.myfaces.ALLOW_JAVASCRIPT</param-name>
        <param-value>true</param-value>
    </context-param>
    <context-param>
        <param-name>org.apache.myfaces.DETECT_JAVASCRIPT</param-name>
        <param-value>false</param-value>
    </context-param>
    <context-param>
        <description>If true, rendered HTML code will be formatted, so that
it is "human readable".
            i.e. additional line separators and whitespace will be written,
that do not
            influence the HTML code.
        Default: "true"</description>
        <param-name>org.apache.myfaces.PRETTY_HTML</param-name>
        <param-value>true</param-value>
    </context-param>
    <context-param>
        <description>If true, a javascript function will be rendered that
is able to restore the
            former vertical scroll on every request. Convenient feature if
you have pages
            with long lists and you do not want the browser page to always
jump to the top
            if you trigger a link or button action that stays on the same
page.
        Default: "false"</description>
        <param-name>org.apache.myfaces.AUTO_SCROLL</param-name>
        <param-value>true</param-value>
    </context-param>
```

```
    <context-param>
        <description>
            Validate managed beans, navigation rules and ensure that forms
are not nested.
        </description>
        <param-name>org.apache.myfaces.VALIDATE</param-name>
        <param-value>true</param-value>
    </context-param>

    <context-param>
        <description>A class implementing the
            org.apache.myfaces.shared.renderkit.html.util.AddResource
            interface. It is responsible to
            place scripts and css on the right position in your HTML
document.
            Default:
"org.apache.myfaces.shared.renderkit.html.util.DefaultAddResource"
            Follow the description on the MyFaces-Wiki-Performance page to
enable
            StreamingAddResource instead of DefaultAddResource if you want
to
            gain performance.
        </description>
        <param-name>org.apache.myfaces.ADD_RESOURCE_CLASS</param-name>
        <param-
value>org.apache.myfaces.renderkit.html.util.DefaultAddResource</param-
value>
        <!--param-
value>org.apache.myfaces.component.html.util.StreamingAddResource</param-
value-->
    </context-param>

    <context-param>
        <description>
            A very common problem in configuring MyFaces-web-applications
            is that the Extensions-Filter is not configured at all
            or improperly configured. This parameter will check for a
properly
            configured Extensions-Filter if it is needed by the web-app.
            In most cases this check will work just fine, there might be
cases
            where an internal forward will bypass the Extensions-Filter and
the check
            will not work. If this is the case, you can disable the check
by setting
            this parameter to false.
        </description>
        <param-name>org.apache.myfaces.CHECK_EXTENSIONS_FILTER</param-name>
        <param-value>true</param-value>
    </context-param>

    <context-param>
        <description>
            Change the url-pattern from the ExtensionsFilter
            Default is "/faces/myFacesExtensionResource"
```

```
            Note: The filter-mapping for ExtensionsFilter, the url-pattern
is
            this value + "/*", else there comes a exception
        </description>
        <param-name>org.apache.myfaces.RESOURCE_VIRTUAL_PATH</param-name>
        <param-value>/faces/extensionResource</param-value>
    </context-param>

    <context-param>
        <description>
            This parameter enables partial state saving.
        </description>
        <param-name>javax.faces.PARTIAL_STATE_SAVING_METHOD</param-name>
        <param-value>false</param-value>
    </context-param>

    <context-param>
        <description>
            If true every time a page is rendered, the corresponding JSP is
dispatched also.
            This is very usefull if Scriptlets are used inside the JSP.
        </description>
        <param-
name>javax.faces.PARTIAL_STATE_SAVING_DISPATCH_EVERY_TIME</param-name>
        <param-value>true</param-value>
    </context-param>

  <filter>
        <filter-name>extensionsFilter</filter-name>
        <filter-class>
                org.apache.myfaces.component.html.util.ExtensionsFilter
        </filter-class>
        <init-param>
                <description>
                        Set the size limit for uploaded files. Format: 10 -
10
                        bytes 10k - 10 KB 10m - 10 MB 1g - 1 GB
                </description>
                <param-name>uploadMaxFileSize</param-name>
                <param-value>100m</param-value>
        </init-param>
        <init-param>
                <description>
                        Set the threshold size - files below this limit are
                        stored in memory, files above this limit are stored
on
                        disk.
                        Format: 10 - 10 bytes 10k - 10 KB 10m - 10 MB 1g -
1 GB
                </description>
                <param-name>uploadThresholdSize</param-name>
                <param-value>100k</param-value>
        </init-param>

        <init-param>
                <param-name>uploadRepositoryPath</param-name>
```

```
                <param-value>/imagens/temp</param-value>

        </init-param>

</filter>
 <filter-mapping>
        <filter-name>extensionsFilter</filter-name>
        <url-pattern>*.shtml</url-pattern>
    </filter-mapping>
 <filter>
  <display-name>Ajax4jsf Filter</display-name>
  <filter-name>ajax4jsf</filter-name>
  <filter-class>org.ajax4jsf.Filter</filter-class>
  <init-param>
   <param-name>forceparser</param-name>
   <param-value>false</param-value>
  </init-param>
 </filter>
 <filter-mapping>
  <filter-name>ajax4jsf</filter-name>
   <servlet-name>Faces Servlet</servlet-name>
   <dispatcher>REQUEST</dispatcher>
   <dispatcher>FORWARD</dispatcher>
   <dispatcher>INCLUDE</dispatcher>
 </filter-mapping>

 <filter-mapping>
    <filter-name>extensionsFilter</filter-name>
    <url-pattern>*.faces</url-pattern>
  </filter-mapping>
  <filter-mapping>
        <filter-name>extensionsFilter</filter-name>
        <url-pattern>*.jsf</url-pattern>
    </filter-mapping>
  <filter-mapping>
        <filter-name>extensionsFilter</filter-name>
        <url-pattern>/faces/*</url-pattern>
</filter-mapping>
    <servlet>
        <servlet-name>Faces Servlet</servlet-name>
        <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
        </servlet>
         <servlet>
        <servlet-name>SourceCodeServlet</servlet-name>
        <servlet-
class>org.apache.myfaces.shared_tomahawk.util.servlet.SourceCodeServlet</se
rvlet-class>
    </servlet>
    <servlet>
    <servlet-name>DisplayImage</servlet-name>
    <servlet-class>frotaControle.controle.DisplayImage</servlet-class>
  </servlet>

        <servlet-mapping>
    <servlet-name>Faces Servlet</servlet-name>
```

```
    <url-pattern>*.faces</url-pattern>
  </servlet-mapping>

<servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>*.jsf</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>SourceCodeServlet</servlet-name>
        <url-pattern>*.source</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>Faces Servlet</servlet-name>
        <url-pattern>/faces/*</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
    <servlet-name>DisplayImage</servlet-name>
    <url-pattern>/servlet/DisplayImage</url-pattern>
  </servlet-mapping>

    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
      <welcome-file>
            Inicio.jsp
        </welcome-file>
    </welcome-file-list>
    <filter>
    <filter-name>MyFacesExtensionsFilter</filter-name>
    <filter-
class>org.apache.myfaces.webapp.filter.ExtensionsFilter</filter-class>
    <init-param>
      <param-name>maxFileSize</param-name>
      <param-value>20m</param-value>
    </init-param>
  </filter>
  <filter-mapping>
    <filter-name>MyFacesExtensionsFilter</filter-name>
    <servlet-name>Faces Servlet</servlet-name>
  </filter-mapping>
  <filter-mapping>
    <filter-name>MyFacesExtensionsFilter</filter-name>
    <url-pattern>/faces/myFacesExtensionResource/*</url-pattern>
  </filter-mapping>
</web-app>
```

11.2.43 WEB-INF/jspf/cabecalho.jspf

```
<meta name="description" content="Via Digital - FLO-Pref - Free Livre Open
Software para Prefeituras - GeNESS UFCG CenPRA Softex OpenS" />
<meta name="keywords" content="software livre, componentes, prefeituras" />
<meta name="robots" content="index, follow" />
```

```
<link rel="stylesheet" type="text/css" media="screen"
href="<%=request.getContextPath()%>/CSS/viadigital.css" />
      <link rel="shortcut icon"
href="<%=request.getContextPath()%>/faces/imagens/viadigital.ico" />
                  <%--link rel="stylesheet" type="text/css"
href="<%=request.getContextPath()%>/CSS/CSSControleFrota.css"/--%>



<script type="text/javascript">
_uacct = "UA-270981-2";
_uacct = "UA-1496692-4";

</script>

    <TR>
    <TD>
    <!--TABLE cellSpacing=0  cellPadding=0 width="100%"
background="#ff6600" border=0-->
        <TBODY>

        <TD background="">          <!--table cellpadding="0"
style="margin:0 auto;text-align:left;"cellspacing="0" class="moduletable"--
>

                <!--td>
                <a target="_self" href="inicio.jsp"><img border="0"
src="faces/imagens/icone.PNG" /></a>
                </t>


            <!--/table>
        </TD-->
            <div id="barratitulo"><div id="sitetitulo" >Controle de
Frotas</div><p id="sitesubtitulo"></p></div>
        </TBODY>
    <!--/TABLE-->
    </TD>
    </TR>

    <hr></hr>
```

11.3 Classes de Controle

As classes de controle são divididas em duas partes: as classes Controller que definem as funções e o acesso das páginas WEB ao model, e as classes Converter que fazem a conversão de tipos primitos para String, para leitura dos dados pelas páginas WEB.

11.3.1 frotaControle/controle/ AtividadeController

```
package frotaControle.controle;

import frotaControle.model.*;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class AtividadeController
{

    public AtividadeController()
    {
        openIniciarAtividade = true;
        openFinalizarAtividade = false;
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Atividade getAtividade()
    {
        return atividade;
    }

    public void setAtividade(Atividade atividade)
    {
        this.atividade = atividade;
    }

    public DataModel getDetailAtividades()
    {
        return model;
    }

    public void setDetailAtividades(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
```

```
    }

    public String create()
    {
        EntityManager em;
        updateIntegrante();
        updateMotorista();
        em = getEntityManager();
        atividade.setIntegranteIdintegrante(integrante);
        atividade.setMotoristaoperadorIdmotoristaoperador(motorista);
        atividade.setKminicial(integrante.getKmatual());
        atividade.setDataatividade(new Date());
        em.getTransaction().begin();
        em.persist(atividade);
        em.getTransaction().commit();
        addSuccessMessage("Atividade was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_159;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_159;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return "view_inicio";
    }

    public void updateMotorista()
    {
        EntityManager em;
        motorista.setDisponivel(false);
        em = getEntityManager();
        Motoristaoperador l = (Motoristaoperador)em.createQuery("SELECT  i FROM
Motoristaoperador i WHERE (i.idmotoristaoperador=
```

```
:idmotoristaoperador)").setParameter("idmotoristaoperador",
motorista.getIdmotoristaoperador()).getSingleResult();
        l.setDisponivel(false);
        em.getTransaction().begin();
        em.merge(l);
        em.getTransaction().commit();
        addSuccessMessage("Motorista was successfully update.");
        em.close();
        break MISSING_BLOCK_LABEL_143;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_143;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void updateIntegrante()
    {
        EntityManager em;
        integrante.setDisponivel(false);
        em = getEntityManager();
        Integrante l = (Integrante)em.createQuery("SELECT  i FROM Integrante i
WHERE (i.idintegrante= :idintegrante)").setParameter("idintegrante",
integrante.getIdintegrante()).getSingleResult();
        l = integrante;
        em.getTransaction().begin();
        em.merge(l);
        em.getTransaction().commit();
        addSuccessMessage("Integrante was successfully update.");
        em.close();
        break MISSING_BLOCK_LABEL_143;
        Exception ex;
        ex;
        try
```

```
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_143;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        atividade = (Atividade)em.merge(atividade);
        em.getTransaction().commit();
        addSuccessMessage("Atividade was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String destroy()
```

```
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Atividade atividade = getAtividadeFromRequestParam();
    atividade = (Atividade)em.merge(atividade);
    em.remove(atividade);
    em.getTransaction().commit();
    addSuccessMessage("Atividade was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Atividade getAtividadeFromRequestParam()
{
    EntityManager em = getEntityManager();
    Atividade atividade1;
    Atividade o = (Atividade)model.getRowData();
    o = (Atividade)em.merge(o);
    atividade1 = o;
    em.close();
    return atividade1;
    Exception exception;
    exception;
    em.close();
    throw exception;
}

public void setAtividadeFromRequestParam()
```

```
    {
        Atividade atividade = getAtividadeFromRequestParam();
        setAtividade(atividade);
    }

    public DataModel getAtividades()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Atividade as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Atividade findAtividade(Integer id)
    {
        EntityManager em = getEntityManager();
        Atividade atividade1;
        Atividade o = (Atividade)em.find(frotaControle/model/Atividade, id);
        atividade1 = o;
        em.close();
        return atividade1;
```

```
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getGrupoIntegranteIdGrupointegrantes()
    {
        EntityManager em;
        em = getEntityManager();
        SelectItem select[] = {
            new SelectItem(new Integer(0), "...")
        };
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Grupointegrante as o ORDER BY
o.nomegrupointegrante").getResultList();
        SelectItem select[] = new SelectItem[l.size() + 1];
        int i = 1;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Grupointegrante x = (Grupointegrante)i$.next();
            select[i++] = new SelectItem(x, x.getNomegrupointegrante());
        }

        select[0] = new SelectItem(new Integer(0), "...");
        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getSubGrupoIntegrante()
    {
        SelectItem subIntegra[];
label0:
        {
            EntityManager em = getEntityManager();
            subIntegra = (new SelectItem[] {
                new SelectItem(new Integer(0), "...")
            });
            SelectItem aselectitem[];
            try
            {
```

```
            List l = em.createQuery("SELECT Distinct i FROM Subgrupointegrantes i
WHERE i.grupointegrante_idgrupointegrante= :grupointegrante_idgrupointegrante
ORDER BY i.nomesubgrupo").setParameter("grupointegrante_idgrupointegrante",
grupo).getResultList();
            subIntegra = new SelectItem[l.size() + 1];
            int i = 1;
            for(Iterator i$ = l.iterator(); i$.hasNext();)
            {
                Subgrupointegrantes x = (Subgrupointegrantes)i$.next();
                subIntegra[i++] = new SelectItem(x, x.getNomesubgrupo());
            }

            subIntegra[0] = new SelectItem(new Integer(0), "...");
            aselectitem = subIntegra;
        }
        catch(NullPointerException e)
        {
            em.close();
            break label0;
        }
        finally
        {
            em.close();
            throw exception;
        }
        em.close();
        return aselectitem;
    }
    return subIntegra;
}

public SelectItem[] getIntegranteIdintegrantes()
{
    SelectItem integra[];
label0:
    {
        EntityManager em = getEntityManager();
        integra = (new SelectItem[] {
            new SelectItem(new Integer(0), "...")
        });
        SelectItem aselectitem[];
        try
        {
            List l = em.createQuery("SELECT Distinct i FROM Integrante i WHERE
(i.disponivel = 'true') AND (i.ativointegrante = 'true') AND
(i.subgrupointegrantes_idsubgrupointegrantes =
```

```
:subgrupointegrantes_idsubgrupointegrantes) ORDER BY
i.nomeintegrante").setParameter("subgrupointegrantes_idsubgrupointegrantes",
sub).getResultList();
                if(l.isEmpty())
                {
                    this;
                    addSuccessMessage("n\343o existem ve\355culos dispon\355veis!!");
                }
                integra = new SelectItem[l.size() + 1];
                int i = 1;
                for(Iterator i$ = l.iterator(); i$.hasNext();)
                {
                    Integrante x = (Integrante)i$.next();
                    integra[i++] = new SelectItem(x, x.getNomeintegrante());
                }

                integra[0] = new SelectItem(new Integer(0), "...");
                aselectitem = integra;
            }
            catch(NullPointerException e)
            {
                em.close();
                break label0;
            }
            finally
            {
                em.close();
                throw exception;
            }
            em.close();
            return aselectitem;
        }
        return integra;
    }

    public SelectItem[] getPlacas()
    {
        SelectItem integra[];
label0:
        {
            EntityManager em = getEntityManager();
            integra = (new SelectItem[] {
                new SelectItem(new Integer(0), "...")
            });
            SelectItem aselectitem[];
            try
```

```
            {
                List l = em.createQuery("SELECT i FROM Integrante i WHERE
(i.disponivel = 'true') AND (i.nomeintegrante =
:nomeintegrante)").setParameter("nomeintegrante",
integrante.getNomeintegrante()).getResultList();
                integra = new SelectItem[l.size() + 1];
                int i = 1;
                for(Iterator i$ = l.iterator(); i$.hasNext();)
                {
                    Integrante x = (Integrante)i$.next();
                    integra[i++] = new SelectItem(x, x.getPlaca());
                }

                integra[0] = new SelectItem(new Integer(0), "...");
                aselectitem = integra;
            }
            catch(NullPointerException e)
            {
                em.close();
                break label0;
            }
            finally
            {
                em.close();
                throw exception;
            }
            em.close();
            return aselectitem;
        }
        return integra;
    }

    public SelectItem[] getMotoristaoperadorIdmotoristaoperadors()
    {
        SelectItem select[];
label0:
        {
            EntityManager em = getEntityManager();
            select = (new SelectItem[] {
                new SelectItem(new Integer(0), "...")
            });
            SelectItem aselectitem[];
            try
            {
                List l = em.createQuery("SELECT m FROM Motoristaoperador m WHERE
(m.disponivel = 'true') AND ( m.categoriahabilitacao_idcategoriahabilitacao =
```

```
:categoriahabilitacao_idcategoriahabilitacao)").setParameter("categoriahabilitacao_id
categoriahabilitacao",
integrante.getCategoriahabilitacaoIdcategoriahabilitacao()).getResultList();
                select = new SelectItem[l.size() + 1];
                int i = 1;
                for(Iterator i$ = l.iterator(); i$.hasNext();)
                {
                    Motoristaoperador x = (Motoristaoperador)i$.next();
                    select[i++] = new SelectItem(x, x.getNome());
                }

                select[0] = new SelectItem(new Integer(0), "...");
                aselectitem = select;
            }
            catch(NullPointerException e)
            {
                em.close();
                break label0;
            }
            finally
            {
                em.close();
                throw exception;
            }
            em.close();
            return aselectitem;
        }
        return select;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Atividade as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
```

```
{
    atividade = new Atividade();
}

public void createAction(ActionEvent a0)
{
    create();
}

public void editAction(ActionEvent a0)
{
    setAtividadeFromRequestParam();
}

public void detailAction(ActionEvent a0)
{
    setAtividadeFromRequestParam();
}

public boolean isOpenIniciarAtividade()
{
    if(openIniciarAtividade)
        atividade = new Atividade();
    openIniciarAtividade = !openIniciarAtividade;
    return true;
}

public boolean isopenFinalizarAtividade()
{
    openFinalizarAtividade = !openFinalizarAtividade;
    return openFinalizarAtividade;
}

public Grupointegrante getGrupo()
{
    return grupo;
}

public void setGrupo(Grupointegrante grupo)
{
    this.grupo = grupo;
}

public Subgrupointegrantes getSub()
{
    return sub;
```

```
}

public void setSub(Subgrupointegrantes sub)
{
    this.sub = sub;
}

public Motoristaoperador getMotorista()
{
    return motorista;
}

public void setMotorista(Motoristaoperador motorista)
{
    this.motorista = motorista;
}

public Integrante getIntegrante()
{
    return integrante;
}

public void setIntegrante(Integrante integrante)
{
    this.integrante = integrante;
}

public Double getKmAtual()
{
    if(integranteReal != null)
        return integranteReal.getKmatual();
    else
        return null;
}

public void setKmAtual(Double kmAtual)
{
    this.kmAtual = kmAtual;
}

public Integrante getIntegranteReal()
{
    return integranteReal;
}

public void setIntegranteReal(Integrante integranteReal)
```

```
    {
        this.integranteReal = integranteReal;
    }

    private Atividade atividade;
    private Subgrupointegrantes sub;
    private Motoristaoperador motorista;
    private Double kmAtual;
    private Grupointegrante grupo;
    private Integrante integrante;
    private Integrante integranteReal;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean openIniciarAtividade;
    private boolean openFinalizarAtividade;
    private int batchSize;
    private int firstItem;
}
```

11.3.2 frotaControle/controle/ AtividadeConverter

```
package frotaControle.controle;

import frotaControle.model.Atividade;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            AtividadeController

public class AtividadeConverter
    implements Converter
{

    public AtividadeConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
```

```
        } else
        {
            Integer id = new Integer(string);
            AtividadeController controller =
(AtividadeController)facesContext.getApplication().getVariableResolver().resolveVari
able(facesContext, "atividade");
            return controller.findAtividade(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Atividade)
        {
            Atividade o = (Atividade)object;
            return (new StringBuilder()).append("").append(o.getIdatividade()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Atividade").toString());
        }
    }
}
```

11.3.3 frotaControle/controle/CategoriahabilitacaoController

```
package frotaControle.controle;

import frotaControle.model.Categoriahabilitacao;
import java.util.ArrayList;
import java.util.Collection;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.persistence.*;

public class CategoriahabilitacaoController
{

    public CategoriahabilitacaoController()
```

```
{
    batchSize = 20;
    firstItem = 0;
    emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
}

private EntityManager getEntityManager()
{
    return emf.createEntityManager();
}

public Categoriahabilitacao getCategoriahabilitacao()
{
    return categoriahabilitacao;
}

public void setCategoriahabilitacao(Categoriahabilitacao categoriahabilitacao)
{
    this.categoriahabilitacao = categoriahabilitacao;
}

public DataModel getDetailCategoriahabilitacaos()
{
    return model;
}

public void setDetailCategoriahabilitacaos(Collection m)
{
    model = new ListDataModel(new ArrayList(m));
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(categoriahabilitacao);
    em.getTransaction().commit();
    addSuccessMessage("Categoriahabilitacao was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
```

```
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String edit()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    categoriahabilitacao = (Categoriahabilitacao)em.merge(categoriahabilitacao);
    em.getTransaction().commit();
    addSuccessMessage("Categoriahabilitacao was successfully updated.");
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
```

```
        em.getTransaction().begin();
        Categoriahabilitacao categoriahabilitacao =
getCategoriahabilitacaoFromRequestParam();
        categoriahabilitacao = (Categoriahabilitacao)em.merge(categoriahabilitacao);
        em.remove(categoriahabilitacao);
        em.getTransaction().commit();
        addSuccessMessage("Categoriahabilitacao was successfully deleted.");
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public Categoriahabilitacao getCategoriahabilitacaoFromRequestParam()
    {
        EntityManager em = getEntityManager();
        Categoriahabilitacao categoriahabilitacao1;
        Categoriahabilitacao o = (Categoriahabilitacao)model.getRowData();
        o = (Categoriahabilitacao)em.merge(o);
        categoriahabilitacao1 = o;
        em.close();
        return categoriahabilitacao1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setCategoriahabilitacaoFromRequestParam()
    {
```

```
        Categoriahabilitacao categoriahabilitacao =
getCategoriahabilitacaoFromRequestParam();
        setCategoriahabilitacao(categoriahabilitacao);
    }

    public DataModel getCategoriahabilitacaos()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Categoriahabilitacao as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Categoriahabilitacao findCategoriahabilitacao(Integer id)
    {
        EntityManager em = getEntityManager();
        Categoriahabilitacao categoriahabilitacao1;
        Categoriahabilitacao o =
(Categoriahabilitacao)em.find(frotaControle/model/Categoriahabilitacao, id);
        categoriahabilitacao1 = o;
        em.close();
```

```
        return categoriahabilitacao1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Categoriahabilitacao as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        categoriahabilitacao = new Categoriahabilitacao();
    }

    public void editAction(ActionEvent a0)
    {
        setCategoriahabilitacaoFromRequestParam();
    }

    public void destroyAction(ActionEvent a0)
    {
        destroy();
    }

    public void detailAction(ActionEvent a0)
    {
        setCategoriahabilitacaoFromRequestParam();
    }

    private Categoriahabilitacao categoriahabilitacao;
    private DataModel model;
    private EntityManagerFactory emf;
    private int batchSize;
```

```
    private int firstItem;
}
```

11.3.4 frotaControle/controle/ CategoriahabilitacaoConverter

```
package frotaControle.controle;

import frotaControle.model.Categoriahabilitacao;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            CategoriahabilitacaoController

public class CategoriahabilitacaoConverter
    implements Converter
{

    public CategoriahabilitacaoConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            CategoriahabilitacaoController controller =
(CategoriahabilitacaoController)facesContext.getApplication().getVariableResolver().r
esolveVariable(facesContext, "categoriahabilitacao");
            return controller.findCategoriahabilitacao(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
```

```
            return null;
        if(object instanceof Categoriahabilitacao)
        {
            Categoriahabilitacao o = (Categoriahabilitacao)object;
            return (new
StringBuilder()).append("").append(o.getIdcategoriahabilitacao()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Categoriahabilitacao").toString());
        }
    }
}
```

11.3.5 frotaControle/controle/ DisplayImage

```
package frotaControle.controle;

import frotaControle.model.Motoristaoperador;
import java.io.IOException;
import java.io.PrintStream;
import javax.persistence.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class DisplayImage extends HttpServlet
{

    public DisplayImage()
    {
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public void init(ServletConfig config)
        throws ServletException
    {
        super.init(config);
    }
```

```
    public void destroy()
    {
    }

    protected void processRequest(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException
    {
        String id = request.getParameter("imageid");
        try
        {
            ServletOutputStream out = response.getOutputStream();
            out.write(getImage(id));
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
            e.printStackTrace();
        }
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException
    {
        processRequest(request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException
    {
        processRequest(request, response);
    }

    public String getServletInfo()
    {
        return "Displays a picture from the database identified by a parameter
IMAGEID";
    }

    private byte[] getImage(String id)
        throws IOException
    {
        byte result[] = null;
        if(id == null)
            return null;
```

```
        EntityManager em = getEntityManager();
        try
        {
            Query q = em.createQuery("SELECT m FROM Motoristaoperador m WHERE
m.idmotoristaoperador = :idmotoristaoperador").setParameter("idmotoristaoperador",
Integer.valueOf(Integer.parseInt(id)));
            Motoristaoperador motorista = (Motoristaoperador)q.getSingleResult();
            result = motorista.getFoto();
        }
        catch(Exception e)
        {
            System.out.println(e.getMessage());
            e.printStackTrace();
        }
        return result;
    }

    private EntityManagerFactory emf;
}
```

11.3.6 frotaControle/controle/ FornecedorController

```
package frotaControle.controle;

import frotaControle.model.Fornecedor;
import java.util.ArrayList;
import java.util.Collection;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.persistence.*;

public class FornecedorController
{

    public FornecedorController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
```

```
}

public Fornecedor getFornecedor()
{
    return fornecedor;
}

public void setFornecedor(Fornecedor fornecedor)
{
    this.fornecedor = fornecedor;
}

public DataModel getDetailFornecedors()
{
    return model;
}

public void setDetailFornecedors(Collection m)
{
    model = new ListDataModel(new ArrayList(m));
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(fornecedor);
    em.getTransaction().commit();
    addSuccessMessage("Fornecedor was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
```

```
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        fornecedor = (Fornecedor)em.merge(fornecedor);
        em.getTransaction().commit();
        addSuccessMessage("Fornecedor was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String destroy()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        Fornecedor fornecedor = getFornecedorFromRequestParam();
        fornecedor = (Fornecedor)em.merge(fornecedor);
        em.remove(fornecedor);
        em.getTransaction().commit();
        addSuccessMessage("Fornecedor was successfully deleted.");
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception ex;
```

```
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Fornecedor getFornecedorFromRequestParam()
{
    EntityManager em = getEntityManager();
    Fornecedor fornecedor1;
    Fornecedor o = (Fornecedor)model.getRowData();
    o = (Fornecedor)em.merge(o);
    fornecedor1 = o;
    em.close();
    return fornecedor1;
    Exception exception;
    exception;
    em.close();
    throw exception;
}

public void setFornecedorFromRequestParam()
{
    Fornecedor fornecedor = getFornecedorFromRequestParam();
    setFornecedor(fornecedor);
}

public DataModel getFornecedors()
{
    EntityManager em = getEntityManager();
    DataModel datamodel;
    Query q = em.createQuery("select object(o) from Fornecedor as o");
    q.setMaxResults(batchSize);
```

```
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Fornecedor findFornecedor(Integer id)
    {
        EntityManager em = getEntityManager();
        Fornecedor fornecedor1;
        Fornecedor o = (Fornecedor)em.find(frotaControle/model/Fornecedor, id);
        fornecedor1 = o;
        em.close();
        return fornecedor1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
```

```
        int count = ((Long)em.createQuery("select count(o) from Fornecedor as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        fornecedor = new Fornecedor();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setFornecedorFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setFornecedorFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Fornecedor fornecedor;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.7 frotaControle/controle/ FornecedorConverter

```
package frotaControle.controle;

import frotaControle.model.Fornecedor;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            FornecedorController

public class FornecedorConverter
    implements Converter
{

    public FornecedorConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            FornecedorController controller =
(FornecedorController)facesContext.getApplication().getVariableResolver().resolveV
ariable(facesContext, "fornecedor");
            return controller.findFornecedor(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Fornecedor)
        {
            Fornecedor o = (Fornecedor)object;
```

```
            return (new
StringBuilder()).append("").append(o.getIdfornecedor()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Fornecedor").toString());
        }
    }
}
```

11.3.8 frotaControle/controle/ GrupointegranteController

```
package frotaControle.controle;

import frotaControle.model.Grupointegrante;
import java.util.ArrayList;
import java.util.Collection;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.persistence.*;

public class GrupointegranteController
{

    public GrupointegranteController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Grupointegrante getGrupointegrante()
    {
        return grupointegrante;
    }

    public void setGrupointegrante(Grupointegrante grupointegrante)
```

```
{
    this.grupointegrante = grupointegrante;
}

public DataModel getDetailGrupointegrantes()
{
    return model;
}

public void setDetailGrupointegrantes(Collection m)
{
    model = new ListDataModel(new ArrayList(m));
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(grupointegrante);
    em.getTransaction().commit();
    addSuccessMessage("Grupointegrante was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String edit()
{
    EntityManager em = getEntityManager();
```

```
        em.getTransaction().begin();
        grupointegrante = (Grupointegrante)em.merge(grupointegrante);
        em.getTransaction().commit();
        addSuccessMessage("Grupointegrante was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String destroy()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        Grupointegrante grupointegrante = getGrupointegranteFromRequestParam();
        grupointegrante = (Grupointegrante)em.merge(grupointegrante);
        em.remove(grupointegrante);
        em.getTransaction().commit();
        addSuccessMessage("Grupointegrante was successfully deleted.");
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
```

```
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public Grupointegrante getGrupointegranteFromRequestParam()
    {
        EntityManager em = getEntityManager();
        Grupointegrante grupointegrante1;
        Grupointegrante o = (Grupointegrante)model.getRowData();
        o = (Grupointegrante)em.merge(o);
        grupointegrante1 = o;
        em.close();
        return grupointegrante1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setGrupointegranteFromRequestParam()
    {
        Grupointegrante grupointegrante = getGrupointegranteFromRequestParam();
        setGrupointegrante(grupointegrante);
    }

    public DataModel getGrupointegrantes()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Grupointegrante as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
```

```
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Grupointegrante findGrupointegrante(Integer id)
    {
        EntityManager em = getEntityManager();
        Grupointegrante grupointegrante1;
        Grupointegrante o =
(Grupointegrante)em.find(frotaControle/model/Grupointegrante, id);
        grupointegrante1 = o;
        em.close();
        return grupointegrante1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Grupointegrante as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
```

```
            throw exception;
        }

        public void newAction(ActionEvent a0)
        {
            grupointegrante = new Grupointegrante();
            setInput(false);
        }

        public void editAction(ActionEvent a0)
        {
            setGrupointegranteFromRequestParam();
            setInput(false);
        }

        public void detailAction(ActionEvent a0)
        {
            setGrupointegranteFromRequestParam();
            setInput(true);
        }

        public boolean getInput()
        {
            return x;
        }

        public void setInput(boolean x)
        {
            this.x = x;
        }

        private Grupointegrante grupointegrante;
        private DataModel model;
        private EntityManagerFactory emf;
        private boolean x;
        private int batchSize;
        private int firstItem;
    }
```

11.3.9 frotaControle/controle/ GrupointegranteConverter

```
package frotaControle.controle;

import frotaControle.model.Grupointegrante;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
```

```
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            GrupointegranteController

public class GrupointegranteConverter
    implements Converter
{

    public GrupointegranteConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            GrupointegranteController controller =
(GrupointegranteController)facesContext.getApplication().getVariableResolver().resol
veVariable(facesContext, "grupointegrante");
            return controller.findGrupointegrante(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Grupointegrante)
        {
            Grupointegrante o = (Grupointegrante)object;
            return (new
StringBuilder()).append("").append(o.getIdgrupointegrante()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Grupointegrante").toString());
```

```
        }
    }
}
```

11.3.10 frotaControle/controle/IntegranteController

```
package frotaControle.controle;

import frotaControle.model.*;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.model.*;
import javax.persistence.*;

public class IntegranteController
{

    public IntegranteController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Integrante getIntegrante()
    {
        return integrante;
    }

    public void setIntegrante(Integrante integrante)
    {
        this.integrante = integrante;
    }

    public DataModel getDetailIntegrantes()
    {
        return model;
    }

    public void setDetailIntegrantes(Collection m)
    {
```

```
    model = new ListDataModel(new ArrayList(m));
}

public String createSetup()
{
    integrante = new Integrante();
    return "integrante_create";
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(integrante);
    em.getTransaction().commit();
    addSuccessMessage("Integrante was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return "integrante_list";
}

public String detailSetup()
{
    setIntegranteFromRequestParam();
    return "integrante_detail";
}

public String editSetup()
{
```

```
    setIntegranteFromRequestParam();
    return "integrante_edit";
}

public String edit()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    integrante = (Integrante)em.merge(integrante);
    em.getTransaction().commit();
    addSuccessMessage("Integrante was successfully updated.");
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return "integrante_list";
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Integrante integrante = getIntegranteFromRequestParam();
    integrante = (Integrante)em.merge(integrante);
    em.remove(integrante);
    em.getTransaction().commit();
    addSuccessMessage("Integrante was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
```

```
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return "integrante_list";
    }

    public Integrante getIntegranteFromRequestParam()
    {
        EntityManager em = getEntityManager();
        Integrante integrante1;
        Integrante o = (Integrante)model.getRowData();
        o = (Integrante)em.merge(o);
        integrante1 = o;
        em.close();
        return integrante1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setIntegranteFromRequestParam()
    {
        Integrante integrante = getIntegranteFromRequestParam();
        setIntegrante(integrante);
    }

    public DataModel getIntegrantes()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Integrante as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
```

```
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Integrante findIntegrante(Integer id)
    {
        EntityManager em = getEntityManager();
        Integrante integrante1;
        Integrante o = (Integrante)em.find(frotaControle/model/Integrante, id);
        integrante1 = o;
        em.close();
        return integrante1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getCategoriahabilitacaoIdcategoriahabilitacaos()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Categoriahabilitacao as
o").getResultList();
```

```
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Categoriahabilitacao x = (Categoriahabilitacao)i$.next();
            select[i++] = new SelectItem(x);
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getSubgrupointegrantesIdsubgrupointegrantess()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Subgrupointegrantes as
o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Subgrupointegrantes x = (Subgrupointegrantes)i$.next();
            select[i++] = new SelectItem(x);
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getTipocombustiveIdtipocombustivels()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Tipocombustivel as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
```

```
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Tipocombustivel x = (Tipocombustivel)i$.next();
            select[i++] = new SelectItem(x);
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Integrante as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getFirstItem()
    {
        return firstItem;
    }

    public int getLastItem()
    {
        int size = getItemCount();
        return firstItem + batchSize <= size ? firstItem + batchSize : size;
    }

    public int getBatchSize()
    {
        return batchSize;
    }
```

```
    public String next()
    {
        if(firstItem + batchSize < getItemCount())
            firstItem += batchSize;
        return "integrante_list";
    }

    public String prev()
    {
        firstItem -= batchSize;
        if(firstItem < 0)
            firstItem = 0;
        return "integrante_list";
    }

    private Integrante integrante;
    private DataModel model;
    private EntityManagerFactory emf;
    private int batchSize;
    private int firstItem;
}
```

11.3.11 frotaControle/controle/IntegranteConverter

```
package frotaControle.controle;

import frotaControle.model.Integrante;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            IntegranteController

public class IntegranteConverter
    implements Converter
{

    public IntegranteConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
```

```
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            IntegranteController controller =
(IntegranteController)facesContext.getApplication().getVariableResolver().resolveVar
iable(facesContext, "integrante");
            return controller.findIntegrante(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Integrante)
        {
            Integrante o = (Integrante)object;
            return (new StringBuilder()).append("").append(o.getIdintegrante()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Integrante").toString());
        }
    }
}
```

11.3.12 frotaControle/controle/ItensController

```
package frotaControle.controle;

import frotaControle.model.Itens;
import frotaControle.model.Unidade;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class ItensController
```

```
{

    public ItensController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Itens getItens()
    {
        return itens;
    }

    public void setItens(Itens itens)
    {
        this.itens = itens;
    }

    public DataModel getDetailItenss()
    {
        return model;
    }

    public void setDetailItenss(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }

    public String create()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        em.persist(itens);
        em.getTransaction().commit();
        addSuccessMessage("Itens was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception ex;
        ex;
        try
```

```
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        itens = (Itens)em.merge(itens);
        em.getTransaction().commit();
        addSuccessMessage("Itens was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }
```

```
public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Itens itens = getItensFromRequestParam();
    itens = (Itens)em.merge(itens);
    em.remove(itens);
    em.getTransaction().commit();
    addSuccessMessage("Itens was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Itens getItensFromRequestParam()
{
    EntityManager em = getEntityManager();
    Itens itens1;
    Itens o = (Itens)model.getRowData();
    o = (Itens)em.merge(o);
    itens1 = o;
    em.close();
    return itens1;
    Exception exception;
    exception;
    em.close();
    throw exception;
}
```

```
    public void setItensFromRequestParam()
    {
        Itens itens = getItensFromRequestParam();
        setItens(itens);
    }

    public DataModel getItenss()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Itens as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Itens findItens(Integer id)
    {
        EntityManager em = getEntityManager();
        Itens itens1;
        Itens o = (Itens)em.find(frotaControle/model/Itens, id);
        itens1 = o;
        em.close();
```

```
        return itens1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getUnidadeIdunidades()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Unidade as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Unidade x = (Unidade)i$.next();
            select[i++] = new SelectItem(x);
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Itens as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
```

```
        itens = new Itens();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setItensFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setItensFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Itens itens;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.13 frotaControle/controle/ItensConverter

```
package frotaControle.controle;

import frotaControle.model.Itens;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
```

```
//          ItensController

public class ItensConverter
    implements Converter
{

    public ItensConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            ItensController controller =
(ItensController)facesContext.getApplication().getVariableResolver().resolveVariable(
facesContext, "itens");
            return controller.findItens(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Itens)
        {
            Itens o = (Itens)object;
            return (new StringBuilder()).append("").append(o.getIditens()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Itens").toString());
        }
    }
}
```

11.3.14 frotaControle/controle/ManutencaoController

```
package frotaControle.controle;

import frotaControle.model.*;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class ManutencaoController
{

    public ManutencaoController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Manutencao getManutencao()
    {
        return manutencao;
    }

    public void setManutencao(Manutencao manutencao)
    {
        this.manutencao = manutencao;
    }

    public DataModel getDetailManutencaos()
    {
        return model;
    }

    public void setDetailManutencaos(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }
```

```
public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(manutencao);
    em.getTransaction().commit();
    addSuccessMessage("Manutencao was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String edit()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    manutencao = (Manutencao)em.merge(manutencao);
    em.getTransaction().commit();
    addSuccessMessage("Manutencao was successfully updated.");
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
```

```
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Manutencao manutencao = getManutencaoFromRequestParam();
    manutencao = (Manutencao)em.merge(manutencao);
    em.remove(manutencao);
    em.getTransaction().commit();
    addSuccessMessage("Manutencao was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Manutencao getManutencaoFromRequestParam()
{
    EntityManager em = getEntityManager();
```

```
        Manutencao manutencao1;
        Manutencao o = (Manutencao)model.getRowData();
        o = (Manutencao)em.merge(o);
        manutencao1 = o;
        em.close();
        return manutencao1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setManutencaoFromRequestParam()
    {
        Manutencao manutencao = getManutencaoFromRequestParam();
        setManutencao(manutencao);
    }

    public DataModel getManutencaos()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Manutencao as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
```

```
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Manutencao findManutencao(Integer id)
    {
        EntityManager em = getEntityManager();
        Manutencao manutencao1;
        Manutencao o = (Manutencao)em.find(frotaControle/model/Manutencao, id);
        manutencao1 = o;
        em.close();
        return manutencao1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getAtividadeIdatividades()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Atividade as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Atividade x = (Atividade)i$.next();
            select[i++] = new SelectItem(x, x.getIdatividade().toString());
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getIntegranteIdintegrantes()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
```

```
        List l = em.createQuery("select o from Integrante as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Integrante x = (Integrante)i$.next();
            select[i++] = new SelectItem(x, x.getNomeintegrante());
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Manutencao as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        manutencao = new Manutencao();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setManutencaoFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
```

```
    {
        setManutencaoFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Manutencao manutencao;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.15 frotaControle/controle/ManutencaoConverter

```
package frotaControle.controle;

import frotaControle.model.Manutencao;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            ManutencaoController

public class ManutencaoConverter
    implements Converter
{

    public ManutencaoConverter()
    {
```

```
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            ManutencaoController controller =
(ManutencaoController)facesContext.getApplication().getVariableResolver().resolveV
ariable(facesContext, "manutencao");
            return controller.findManutencao(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Manutencao)
        {
            Manutencao o = (Manutencao)object;
            return (new
StringBuilder()).append("").append(o.getIdmanutencao()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Manutencao").toString());
        }
    }
}
```

11.3.16 frotaControle/controle/MotoristaoperadorController

```
package frotaControle.controle;

import frotaControle.model.Categoriahabilitacao;
```

```
import frotaControle.model.Motoristaoperador;
import java.io.*;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;
import javax.servlet.ServletContext;
import org.apache.myfaces.custom.fileupload.UploadedFile;

public class MotoristaoperadorController
{

    public MotoristaoperadorController()
    {
        nameFoto = "";
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Motoristaoperador getMotoristaoperador()
    {
        return motoristaoperador;
    }

    public void setMotoristaoperador(Motoristaoperador motoristaoperador)
    {
        this.motoristaoperador = motoristaoperador;
    }

    public DataModel getDetailMotoristaoperadors()
    {
        return model;
    }

    public void setDetailMotoristaoperadors(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
```

```
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    if(arquivo != null)
        upload();
    motoristaoperador.setDisponivel(true);
    em.persist(motoristaoperador);
    em.getTransaction().commit();
    addSuccessMessage("Motoristaoperador was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_121;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_121;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String edit()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    if(arquivo != null)
        upload();
    motoristaoperador = (Motoristaoperador)em.merge(motoristaoperador);
    em.getTransaction().commit();
    addSuccessMessage("Motoristaoperador was successfully updated.");
    em.close();
    break MISSING_BLOCK_LABEL_120;
    Exception ex;
```

```
            ex;
            try
            {
                addErrorMessage(ex.getLocalizedMessage());
                em.getTransaction().rollback();
            }
            catch(Exception e)
            {
                addErrorMessage(e.getLocalizedMessage());
            }
            em.close();
            break MISSING_BLOCK_LABEL_120;
            Exception exception;
            exception;
            em.close();
            throw exception;
            return null;
        }

        public String destroy()
        {
            EntityManager em = getEntityManager();
            em.getTransaction().begin();
            Motoristaoperador motoristaoperador =
    getMotoristaoperadorFromRequestParam();
            motoristaoperador = (Motoristaoperador)em.merge(motoristaoperador);
            em.remove(motoristaoperador);
            em.getTransaction().commit();
            addSuccessMessage("Motoristaoperador was successfully deleted.");
            em.close();
            break MISSING_BLOCK_LABEL_114;
            Exception ex;
            ex;
            try
            {
                addErrorMessage(ex.getLocalizedMessage());
                em.getTransaction().rollback();
            }
            catch(Exception e)
            {
                addErrorMessage(e.getLocalizedMessage());
            }
            em.close();
            break MISSING_BLOCK_LABEL_114;
            Exception exception;
            exception;
```

```
            em.close();
            throw exception;
            return null;
        }

        public Motoristaoperador getMotoristaoperadorFromRequestParam()
        {
            EntityManager em = getEntityManager();
            Motoristaoperador motoristaoperador1;
            Motoristaoperador o = (Motoristaoperador)model.getRowData();
            o = (Motoristaoperador)em.merge(o);
            motoristaoperador1 = o;
            em.close();
            return motoristaoperador1;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void setMotoristaoperadorFromRequestParam()
        {
            Motoristaoperador motoristaoperador =
    getMotoristaoperadorFromRequestParam();
            setMotoristaoperador(motoristaoperador);
        }

        public DataModel getMotoristaoperadors()
        {
            EntityManager em = getEntityManager();
            DataModel datamodel;
            Query q = em.createQuery("select object(o) from Motoristaoperador as o");
            q.setMaxResults(batchSize);
            q.setFirstResult(firstItem);
            model = new ListDataModel(q.getResultList());
            datamodel = model;
            em.close();
            return datamodel;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public static void addErrorMessage(String msg)
        {
```

```
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Motoristaoperador findMotoristaoperador(Integer id)
    {
        EntityManager em = getEntityManager();
        Motoristaoperador motoristaoperador1;
        Motoristaoperador o =
(Motoristaoperador)em.find(frotaControle/model/Motoristaoperador, id);
        motoristaoperador1 = o;
        em.close();
        return motoristaoperador1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getCategoriahabilitacaoIdcategoriahabilitacaos()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Categoriahabilitacao as
o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Categoriahabilitacao x = (Categoriahabilitacao)i$.next();
            select[i++] = new SelectItem(x, x.getCategoriahabilitacao());
        }

        aselectitem = select;
        em.close();
        return aselectitem;
```

```
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public int getItemCount()
        {
            EntityManager em = getEntityManager();
            int i;
            int count = ((Long)em.createQuery("select count(o) from Motoristaoperador as
o").getSingleResult()).intValue();
            i = count;
            em.close();
            return i;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void newAction(ActionEvent aO)
        {
            motoristaoperador = new Motoristaoperador();
            setInput(false);
        }

        public void editAction(ActionEvent aO)
        {
            setMotoristaoperadorFromRequestParam();
            setInput(false);
        }

        public void detailAction(ActionEvent a0)
        {
            setMotoristaoperadorFromRequestParam();
            setInput(true);
        }

        public boolean getInput()
        {
            return x;
        }

        public void setInput(boolean x)
        {
```

```
        this.x = x;
    }

    public UploadedFile getarquivo()
    {
        return arquivo;
    }

    public void setarquivo(UploadedFile arquivo)
    {
        this.arquivo = arquivo;
    }

    public String upload()
        throws IOException
    {
        FacesContext facesContext = FacesContext.getCurrentInstance();
        try
        {

facesContext.getExternalContext().getApplicationMap().put("fileupload_bytes",
arquivo.getBytes());
            facesContext.getExternalContext().getApplicationMap().put("fileupload_type",
arquivo.getContentType());

facesContext.getExternalContext().getApplicationMap().put("fileupload_name",
arquivo.getName());
        }
        catch(IOException ex)
        {
            addErrorMessage("erro ao carregar foto");
            ex.printStackTrace();
        }
        try
        {
            BufferedInputStream stream = new
BufferedInputStream(arquivo.getInputStream());
            ServletContext theApplicationsServletContext =
(ServletContext)facesContext.getExternalContext().getContext();
            String realImageFilePath =
theApplicationsServletContext.getRealPath("/imagens/temp");
            File fileOnServer = new File((new
StringBuilder()).append(realImageFilePath).append("/teste").toString());
            BufferedOutputStream os = new BufferedOutputStream(new
FileOutputStream(fileOnServer));
            byte buffer[] = new byte[(int)arquivo.getSize()];
```

```
            for(int count = 0; (count = stream.read(buffer)) != -1;)
                os.write(buffer, 0, count);

            os.close();
            stream.close();
            motoristaoperador.setFoto(buffer);
            fileOnServer.delete();
        }
        catch(Exception ioe)
        {
            ioe.printStackTrace();
        }
        return null;
    }

    public boolean isUploaded()
    {
        FacesContext facesContext = FacesContext.getCurrentInstance();
        boolean x =
facesContext.getExternalContext().getApplicationMap().get("fileupload_bytes") !=
null;
        return
facesContext.getExternalContext().getApplicationMap().get("fileupload_bytes") !=
null;
    }

    public String getNameFoto()
    {
        return nameFoto;
    }

    public void setNameFoto(String nameFoto)
    {
        this.nameFoto = nameFoto;
    }

    public Object getIcon()
    {
        return icon;
    }

    public void setIcon(Object icon)
    {
        this.icon = icon;
    }
```

```
    public void editSaveAction(ActionEvent aO)
    {
        edit();
    }

    private Motoristaoperador motoristaoperador;
    private UploadedFile arquivo;
    private Object icon;
    private String nameFoto;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.17 frotaControle/controle/MotoristaoperadorConverter

```
package frotaControle.controle;

import frotaControle.model.Motoristaoperador;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            MotoristaoperadorController

public class MotoristaoperadorConverter
    implements Converter
{

    public MotoristaoperadorConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
```

```
            return null;
        } else
        {
            Integer id = new Integer(string);
            MotoristaoperadorController controller =
(MotoristaoperadorController)facesContext.getApplication().getVariableResolver().re
solveVariable(facesContext, "motoristaoperador");
            return controller.findMotoristaoperador(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Motoristaoperador)
        {
            Motoristaoperador o = (Motoristaoperador)object;
            return (new
StringBuilder()).append("").append(o.getIdmotoristaoperador()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Motoristaoperador").toString());
        }
    }
}
```

11.3.18 frotaControle/controle/ NotafiscalgastoController

```
import frotaControle.model.Fornecedor;
import frotaControle.model.Notafiscalgasto;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class NotafiscalgastoController
```

```
{

    public NotafiscalgastoController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Notafiscalgasto getNotafiscalgasto()
    {
        return notafiscalgasto;
    }

    public void setNotafiscalgasto(Notafiscalgasto notafiscalgasto)
    {
        this.notafiscalgasto = notafiscalgasto;
    }

    public DataModel getDetailNotafiscalgastos()
    {
        return model;
    }

    public void setDetailNotafiscalgastos(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }

    public String create()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        em.persist(notafiscalgasto);
        em.getTransaction().commit();
        addSuccessMessage("Notafiscalgasto was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception ex;
        ex;
        try
```

```
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        notafiscalgasto = (Notafiscalgasto)em.merge(notafiscalgasto);
        em.getTransaction().commit();
        addSuccessMessage("Notafiscalgasto was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }
```

```
public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Notafiscalgasto notafiscalgasto = getNotafiscalgastoFromRequestParam();
    notafiscalgasto = (Notafiscalgasto)em.merge(notafiscalgasto);
    em.remove(notafiscalgasto);
    em.getTransaction().commit();
    addSuccessMessage("Notafiscalgasto was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Notafiscalgasto getNotafiscalgastoFromRequestParam()
{
    EntityManager em = getEntityManager();
    Notafiscalgasto notafiscalgasto1;
    Notafiscalgasto o = (Notafiscalgasto)model.getRowData();
    o = (Notafiscalgasto)em.merge(o);
    notafiscalgasto1 = o;
    em.close();
    return notafiscalgasto1;
    Exception exception;
    exception;
    em.close();
    throw exception;
}
```

```
    public void setNotafiscalgastoFromRequestParam()
    {
        Notafiscalgasto notafiscalgasto = getNotafiscalgastoFromRequestParam();
        setNotafiscalgasto(notafiscalgasto);
    }

    public DataModel getNotafiscalgastos()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Notafiscalgasto as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Notafiscalgasto findNotafiscalgasto(Integer id)
    {
        EntityManager em = getEntityManager();
        Notafiscalgasto notafiscalgasto1;
        Notafiscalgasto o =
(Notafiscalgasto)em.find(frotaControle/model/Notafiscalgasto, id);
        notafiscalgasto1 = o;
```

```
            em.close();
            return notafiscalgasto1;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public SelectItem[] getFornecedorIdfornecedors()
        {
            EntityManager em = getEntityManager();
            SelectItem aselectitem[];
            List l = em.createQuery("select o from Fornecedor as o").getResultList();
            SelectItem select[] = new SelectItem[l.size()];
            int i = 0;
            for(Iterator i$ = l.iterator(); i$.hasNext();)
            {
                Fornecedor x = (Fornecedor)i$.next();
                select[i++] = new SelectItem(x, x.getRazaosocial());
            }

            aselectitem = select;
            em.close();
            return aselectitem;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public int getItemCount()
        {
            EntityManager em = getEntityManager();
            int i;
            int count = ((Long)em.createQuery("select count(o) from Notafiscalgasto as
o").getSingleResult()).intValue();
            i = count;
            em.close();
            return i;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void newAction(ActionEvent a0)
```

```
    {
        notafiscalgasto = new Notafiscalgasto();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setNotafiscalgastoFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setNotafiscalgastoFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Notafiscalgasto notafiscalgasto;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.19 frotaControle/controle/ NotafiscalgastoConverter

```
package frotaControle.controle;

import frotaControle.model.Notafiscalgasto;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
```

```
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            NotafiscalgastoController

public class NotafiscalgastoConverter
    implements Converter
{

    public NotafiscalgastoConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            NotafiscalgastoController controller =
(NotafiscalgastoController)facesContext.getApplication().getVariableResolver().resol
veVariable(facesContext, "notafiscalgasto");
            return controller.findNotafiscalgasto(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Notafiscalgasto)
        {
            Notafiscalgasto o = (Notafiscalgasto)object;
            return (new
StringBuilder()).append("").append(o.getIdnotafiscalgasto()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Notafiscalgasto").toString());
```

```
        }
    }
}
```

11.3.20 frotaControle/controle/NotafiscalgastoHasItensController

```
package frotaControle.controle;

import frotaControle.model.*;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class NotafiscalgastoHasItensController
{

    public NotafiscalgastoHasItensController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public NotafiscalgastoHasItens getNotafiscalgastoHasItens()
    {
        return notafiscalgastoHasItens;
    }

    public void setNotafiscalgastoHasItens(NotafiscalgastoHasItens
notafiscalgastoHasItens)
    {
        this.notafiscalgastoHasItens = notafiscalgastoHasItens;
    }

    public DataModel getDetailNotafiscalgastoHasItenss()
```

```
    {
        return model;
    }

    public void setDetailNotafiscalgastoHasItenss(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }

    public String create()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        em.persist(notafiscalgastoHasItens);
        em.getTransaction().commit();
        addSuccessMessage("NotafiscalgastoHasItens was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        notafiscalgastoHasItens =
(NotafiscalgastoHasItens)em.merge(notafiscalgastoHasItens);
        em.getTransaction().commit();
        addSuccessMessage("NotafiscalgastoHasItens was successfully updated.");
```

```
            em.close();
            break MISSING_BLOCK_LABEL_108;
            Exception ex;
            ex;
            try
            {
                addErrorMessage(ex.getLocalizedMessage());
                em.getTransaction().rollback();
            }
            catch(Exception e)
            {
                addErrorMessage(e.getLocalizedMessage());
            }
            em.close();
            break MISSING_BLOCK_LABEL_108;
            Exception exception;
            exception;
            em.close();
            throw exception;
            return null;
        }

        public String destroy()
        {
            EntityManager em = getEntityManager();
            em.getTransaction().begin();
            NotafiscalgastoHasItens notafiscalgastoHasItens =
getNotafiscalgastoHasItensFromRequestParam();
            notafiscalgastoHasItens =
(NotafiscalgastoHasItens)em.merge(notafiscalgastoHasItens);
            em.remove(notafiscalgastoHasItens);
            em.getTransaction().commit();
            addSuccessMessage("NotafiscalgastoHasItens was successfully deleted.");
            em.close();
            break MISSING_BLOCK_LABEL_114;
            Exception ex;
            ex;
            try
            {
                addErrorMessage(ex.getLocalizedMessage());
                em.getTransaction().rollback();
            }
            catch(Exception e)
            {
                addErrorMessage(e.getLocalizedMessage());
            }
```

```
            em.close();
            break MISSING_BLOCK_LABEL_114;
            Exception exception;
            exception;
            em.close();
            throw exception;
            return null;
        }

        public NotafiscalgastoHasItens getNotafiscalgastoHasItensFromRequestParam()
        {
            EntityManager em = getEntityManager();
            NotafiscalgastoHasItens notafiscalgastohasitens;
            NotafiscalgastoHasItens o = (NotafiscalgastoHasItens)model.getRowData();
            o = (NotafiscalgastoHasItens)em.merge(o);
            notafiscalgastohasitens = o;
            em.close();
            return notafiscalgastohasitens;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void setNotafiscalgastoHasItensFromRequestParam()
        {
            NotafiscalgastoHasItens notafiscalgastoHasItens =
getNotafiscalgastoHasItensFromRequestParam();
            setNotafiscalgastoHasItens(notafiscalgastoHasItens);
        }

        public DataModel getNotafiscalgastoHasItenss()
        {
            EntityManager em = getEntityManager();
            DataModel datamodel;
            Query q = em.createQuery("select object(o) from NotafiscalgastoHasItens as
o");
            q.setMaxResults(batchSize);
            q.setFirstResult(firstItem);
            model = new ListDataModel(q.getResultList());
            datamodel = model;
            em.close();
            return datamodel;
            Exception exception;
            exception;
            em.close();
```

```
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public NotafiscalgastoHasItens
findNotafiscalgastoHasItens(NotafiscalgastoHasItensPK id)
    {
        EntityManager em = getEntityManager();
        NotafiscalgastoHasItens notafiscalgastohasitens;
        NotafiscalgastoHasItens o =
(NotafiscalgastoHasItens)em.find(frotaControle/model/NotafiscalgastoHasItens, id);
        notafiscalgastohasitens = o;
        em.close();
        return notafiscalgastohasitens;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getItenss()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Itens as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Itens x = (Itens)i$.next();
            select[i++] = new SelectItem(x);
```

```
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getNotafiscalgastos()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Notafiscalgasto as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Notafiscalgasto x = (Notafiscalgasto)i$.next();
            select[i++] = new SelectItem(x);
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from
NotafiscalgastoHasItens as o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
```

```
    }

    public void newAction(ActionEvent a0)
    {
        notafiscalgastoHasItens = new NotafiscalgastoHasItens();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setNotafiscalgastoHasItensFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setNotafiscalgastoHasItensFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private NotafiscalgastoHasItens notafiscalgastoHasItens;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.21 frotaControle/controle/ NotafiscalgastoHasItensConverter

```
package frotaControle.controle;

import frotaControle.model.NotafiscalgastoHasItens;
```

```
import frotaControle.model.NotafiscalgastoHasItensPK;
import java.util.StringTokenizer;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            NotafiscalgastoHasItensController

public class NotafiscalgastoHasItensConverter
    implements Converter
{

    public NotafiscalgastoHasItensConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
            return null;
        NotafiscalgastoHasItensPK id = new NotafiscalgastoHasItensPK();
        StringTokenizer idTokens = new StringTokenizer(string, ";");
        String params[] = new String[2];
        int i = 0;
        while(idTokens.hasMoreTokens())
            params[i++] = idTokens.nextToken();
        if(i != 2)
        {
            throw new IllegalArgumentException("Expected format of parameter string is
a set of 2 IDs delimited by ;");
        } else
        {
            id.setNotafiscalgastoIdnotafiscalgasto(Integer.parseInt(params[0]));
            id.setItensIditens(Integer.parseInt(params[1]));
            NotafiscalgastoHasItensController controller =
(NotafiscalgastoHasItensController)facesContext.getApplication().getVariableResolv
er().resolveVariable(facesContext, "notafiscalgastoHasItens");
            return controller.findNotafiscalgastoHasItens(id);
        }
    }
```

```
    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof NotafiscalgastoHasItens)
        {
            NotafiscalgastoHasItens o = (NotafiscalgastoHasItens)object;
            return (new
StringBuilder()).append(o.getNotafiscalgastoHasItensPK().getNotafiscalgastoIdnotafi
scalgasto()).append(";").append(o.getNotafiscalgastoHasItensPK().getItensIditens()).t
oString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.NotafiscalgastoHasItens").toString());
        }
    }
}
```

11.3.22 frotaControle/controle/ NotaservicoController

```
package frotaControle.controle;

import frotaControle.model.*;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class NotaservicoController
{

    public NotaservicoController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
```

```
}

private EntityManager getEntityManager()
{
    return emf.createEntityManager();
}

public Notaservico getNotaservico()
{
    return notaservico;
}

public void setNotaservico(Notaservico notaservico)
{
    this.notaservico = notaservico;
}

public DataModel getDetailNotaservicos()
{
    return model;
}

public void setDetailNotaservicos(Collection m)
{
    model = new ListDataModel(new ArrayList(m));
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(notaservico);
    em.getTransaction().commit();
    addSuccessMessage("Notaservico was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
```

```
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String edit()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    notaservico = (Notaservico)em.merge(notaservico);
    em.getTransaction().commit();
    addSuccessMessage("Notaservico was successfully updated.");
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Notaservico notaservico = getNotaservicoFromRequestParam();
    notaservico = (Notaservico)em.merge(notaservico);
    em.remove(notaservico);
```

```
        em.getTransaction().commit();
        addSuccessMessage("Notaservico was successfully deleted.");
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public Notaservico getNotaservicoFromRequestParam()
    {
        EntityManager em = getEntityManager();
        Notaservico notaservico1;
        Notaservico o = (Notaservico)model.getRowData();
        o = (Notaservico)em.merge(o);
        notaservico1 = o;
        em.close();
        return notaservico1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setNotaservicoFromRequestParam()
    {
        Notaservico notaservico = getNotaservicoFromRequestParam();
        setNotaservico(notaservico);
    }

    public DataModel getNotaservicos()
```

```
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Notaservico as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Notaservico findNotaservico(Integer id)
    {
        EntityManager em = getEntityManager();
        Notaservico notaservico1;
        Notaservico o = (Notaservico)em.find(frotaControle/model/Notaservico, id);
        notaservico1 = o;
        em.close();
        return notaservico1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }
```

```
    public SelectItem[] getIntegranteIdintegrantes()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Integrante as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Integrante x = (Integrante)i$.next();
            select[i++] = new SelectItem(x, x.getNomeintegrante());
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getNotafiscalgastoHasItenss()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from NotafiscalgastoHasItens as
o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            NotafiscalgastoHasItens x = (NotafiscalgastoHasItens)i$.next();
            select[i++] = new SelectItem(x);
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
```

```
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Notaservico as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        notaservico = new Notaservico();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setNotaservicoFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setNotaservicoFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Notaservico notaservico;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
```

```
    private int batchSize;
    private int firstItem;
}
```

11.3.23 frotaControle/controle/ NotaservicoConverter

```
package frotaControle.controle;

import frotaControle.model.Notaservico;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            NotaservicoController

public class NotaservicoConverter
    implements Converter
{

    public NotaservicoConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            NotaservicoController controller =
(NotaservicoController)facesContext.getApplication().getVariableResolver().resolveV
ariable(facesContext, "notaservico");
            return controller.findNotaservico(id);
        }
    }
```

```
    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Notaservico)
        {
            Notaservico o = (Notaservico)object;
            return (new
StringBuilder()).append("").append(o.getIdnotaservico()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Notaservico").toString());
        }
    }
}
```

11.3.24 frotaControle/controle/ PerfilusuarioController

```
package frotaControle.controle;

import frotaControle.model.Perfilusuario;
import java.util.ArrayList;
import java.util.Collection;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.persistence.*;

public class PerfilusuarioController
{

    public PerfilusuarioController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }
```

```
private EntityManager getEntityManager()
{
    return emf.createEntityManager();
}

public Perfilusuario getPerfilusuario()
{
    return perfilusuario;
}

public void setPerfilusuario(Perfilusuario perfilusuario)
{
    this.perfilusuario = perfilusuario;
}

public DataModel getDetailPerfilusuarios()
{
    return model;
}

public void setDetailPerfilusuarios(Collection m)
{
    model = new ListDataModel(new ArrayList(m));
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(perfilusuario);
    em.getTransaction().commit();
    addSuccessMessage("Perfilusuario was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
```

```
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        perfilusuario = (Perfilusuario)em.merge(perfilusuario);
        em.getTransaction().commit();
        addSuccessMessage("Perfilusuario was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String destroy()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        Perfilusuario perfilusuario = getPerfilusuarioFromRequestParam();
        perfilusuario = (Perfilusuario)em.merge(perfilusuario);
        em.remove(perfilusuario);
        em.getTransaction().commit();
```

```
        addSuccessMessage("Perfilusuario was successfully deleted.");
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public Perfilusuario getPerfilusuarioFromRequestParam()
    {
        EntityManager em = getEntityManager();
        Perfilusuario perfilusuario1;
        Perfilusuario o = (Perfilusuario)model.getRowData();
        o = (Perfilusuario)em.merge(o);
        perfilusuario1 = o;
        em.close();
        return perfilusuario1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setPerfilusuarioFromRequestParam()
    {
        Perfilusuario perfilusuario = getPerfilusuarioFromRequestParam();
        setPerfilusuario(perfilusuario);
    }

    public DataModel getPerfilusuarios()
    {
```

```
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Perfilusuario as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Perfilusuario findPerfilusuario(Integer id)
    {
        EntityManager em = getEntityManager();
        Perfilusuario perfilusuario1;
        Perfilusuario o = (Perfilusuario)em.find(frotaControle/model/Perfilusuario, id);
        perfilusuario1 = o;
        em.close();
        return perfilusuario1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
```

```
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Perfilusuario as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        perfilusuario = new Perfilusuario();
        setInput(false);
    }

    public void destroyAction(ActionEvent a0)
    {
        destroy();
    }

    public void editAction(ActionEvent a0)
    {
        setPerfilusuarioFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setPerfilusuarioFromRequestParam();
        setInput(true);
    }

    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }
```

```
    private Perfilusuario perfilusuario;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.25 frotaControle/controle/ PerfilusuarioConverter

```
package frotaControle.controle;

import frotaControle.model.Perfilusuario;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            PerfilusuarioController

public class PerfilusuarioConverter
    implements Converter
{

    public PerfilusuarioConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            PerfilusuarioController controller =
(PerfilusuarioController)facesContext.getApplication().getVariableResolver().resolveV
ariable(facesContext, "perfilusuario");
```

```
            return controller.findPerfilusuario(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Perfilusuario)
        {
            Perfilusuario o = (Perfilusuario)object;
            return (new
StringBuilder()).append("").append(o.getIdperfilusuario()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Perfilusuario").toString());
        }
    }
}
```

11.3.26 frotaControle/controle/ SubgrupointegrantesController

```
package frotaControle.controle;

import frotaControle.model.Grupointegrante;
import frotaControle.model.Subgrupointegrantes;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
import javax.persistence.*;

public class SubgrupointegrantesController
{

    public SubgrupointegrantesController()
    {
        batchSize = 20;
```

```
    firstItem = 0;
    emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
}

private EntityManager getEntityManager()
{
    return emf.createEntityManager();
}

public Subgrupointegrantes getSubgrupointegrantes()
{
    return subgrupointegrantes;
}

public void setSubgrupointegrantes(Subgrupointegrantes subgrupointegrantes)
{
    this.subgrupointegrantes = subgrupointegrantes;
}

public DataModel getDetailSubgrupointegrantess()
{
    return model;
}

public void setDetailSubgrupointegrantess(Collection m)
{
    model = new ListDataModel(new ArrayList(m));
}

public String create()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    em.persist(subgrupointegrantes);
    em.getTransaction().commit();
    addSuccessMessage("Subgrupointegrantes was successfully created.");
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
```

```
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_101;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String edit()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    subgrupointegrantes = (Subgrupointegrantes)em.merge(subgrupointegrantes);
    em.getTransaction().commit();
    addSuccessMessage("Subgrupointegrantes was successfully updated.");
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
```

```
        Subgrupointegrantes subgrupointegrantes =
getSubgrupointegrantesFromRequestParam();
        subgrupointegrantes = (Subgrupointegrantes)em.merge(subgrupointegrantes);
        em.remove(subgrupointegrantes);
        em.getTransaction().commit();
        addSuccessMessage("Subgrupointegrantes was successfully deleted.");
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_114;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public Subgrupointegrantes getSubgrupointegrantesFromRequestParam()
    {
        EntityManager em = getEntityManager();
        Subgrupointegrantes subgrupointegrantes1;
        Subgrupointegrantes o = (Subgrupointegrantes)model.getRowData();
        o = (Subgrupointegrantes)em.merge(o);
        subgrupointegrantes1 = o;
        em.close();
        return subgrupointegrantes1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void setSubgrupointegrantesFromRequestParam()
    {
```

```
        Subgrupointegrantes subgrupointegrantes =
getSubgrupointegrantesFromRequestParam();
        setSubgrupointegrantes(subgrupointegrantes);
    }

    public DataModel getSubgrupointegrantess()
    {
        EntityManager em = getEntityManager();
        DataModel datamodel;
        Query q = em.createQuery("select object(o) from Subgrupointegrantes as o");
        q.setMaxResults(batchSize);
        q.setFirstResult(firstItem);
        model = new ListDataModel(q.getResultList());
        datamodel = model;
        em.close();
        return datamodel;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public static void addErrorMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage(null, facesMsg);
    }

    public static void addSuccessMessage(String msg)
    {
        FacesMessage facesMsg = new
FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
        FacesContext fc = FacesContext.getCurrentInstance();
        fc.addMessage("successInfo", facesMsg);
    }

    public Subgrupointegrantes findSubgrupointegrantes(Integer id)
    {
        EntityManager em = getEntityManager();
        Subgrupointegrantes subgrupointegrantes1;
        Subgrupointegrantes o =
(Subgrupointegrantes)em.find(frotaControle/model/Subgrupointegrantes, id);
        subgrupointegrantes1 = o;
        em.close();
```

```
        return subgrupointegrantes1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getGrupointegranteIdgrupointegrantes()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Grupointegrante as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Grupointegrante x = (Grupointegrante)i$.next();
            select[i++] = new SelectItem(x, x.getNomegrupointegrante());
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Subgrupointegrantes
as o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getFirstItem()
    {
```

```
    return firstItem;
}

public int getLastItem()
{
    int size = getItemCount();
    return firstItem + batchSize <= size ? firstItem + batchSize : size;
}

public int getBatchSize()
{
    return batchSize;
}

public String next()
{
    if(firstItem + batchSize < getItemCount())
        firstItem += batchSize;
    return "subgrupointegrantes_list";
}

public String prev()
{
    firstItem -= batchSize;
    if(firstItem < 0)
        firstItem = 0;
    return "subgrupointegrantes_list";
}

public void newAction(ActionEvent a0)
{
    subgrupointegrantes = new Subgrupointegrantes();
    setInput(false);
}

public void editAction(ActionEvent a0)
{
    setSubgrupointegrantesFromRequestParam();
    setInput(false);
}

public void detailAction(ActionEvent a0)
{
    setSubgrupointegrantesFromRequestParam();
    setInput(true);
}
```

```
    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Subgrupointegrantes subgrupointegrantes;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.27 frotaControle/controle/ SubgrupointegrantesConverter

```
package frotaControle.controle;

import frotaControle.model.Subgrupointegrantes;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            SubgrupointegrantesController

public class SubgrupointegrantesConverter
    implements Converter
{

    public SubgrupointegrantesConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
```

```
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            SubgrupointegrantesController controller =
(SubgrupointegrantesController)facesContext.getApplication().getVariableResolver().
resolveVariable(facesContext, "subgrupointegrantes");
            return controller.findSubgrupointegrantes(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Subgrupointegrantes)
        {
            Subgrupointegrantes o = (Subgrupointegrantes)object;
            return (new
StringBuilder()).append("").append(o.getIdsubgrupointegrantes()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Subgrupointegrantes").toString());
        }
    }
}
```

11.3.28 frotaControle/controle/ TipocombustivelController

```
package frotaControle.controle;

import frotaControle.model.Tipocombustivel;
import java.util.ArrayList;
import java.util.Collection;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
```

```
import javax.faces.event.ActionEvent;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.persistence.*;

public class TipocombustivelController
{

    public TipocombustivelController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Tipocombustivel getTipocombustivel()
    {
        return tipocombustivel;
    }

    public void setTipocombustivel(Tipocombustivel tipocombustivel)
    {
        this.tipocombustivel = tipocombustivel;
    }

    public DataModel getDetailTipocombustivels()
    {
        return model;
    }

    public void setDetailTipocombustivels(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }

    public String create()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        em.persist(tipocombustivel);
        em.getTransaction().commit();
```

```
        addSuccessMessage("Tipocombustivel was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        tipocombustivel = (Tipocombustivel)em.merge(tipocombustivel);
        em.getTransaction().commit();
        addSuccessMessage("Tipocombustivel was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
```

```
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Tipocombustivel tipocombustivel = getTipocombustivelFromRequestParam();
    tipocombustivel = (Tipocombustivel)em.merge(tipocombustivel);
    em.remove(tipocombustivel);
    em.getTransaction().commit();
    addSuccessMessage("Tipocombustivel was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Tipocombustivel getTipocombustivelFromRequestParam()
{
    EntityManager em = getEntityManager();
    Tipocombustivel tipocombustivel1;
    Tipocombustivel o = (Tipocombustivel)model.getRowData();
    o = (Tipocombustivel)em.merge(o);
    tipocombustivel1 = o;
    em.close();
    return tipocombustivel1;
```

```
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void setTipocombustivelFromRequestParam()
        {
            Tipocombustivel tipocombustivel = getTipocombustivelFromRequestParam();
            setTipocombustivel(tipocombustivel);
        }

        public DataModel getTipocombustivels()
        {
            EntityManager em = getEntityManager();
            DataModel datamodel;
            Query q = em.createQuery("select object(o) from Tipocombustivel as o");
            q.setMaxResults(batchSize);
            q.setFirstResult(firstItem);
            model = new ListDataModel(q.getResultList());
            datamodel = model;
            em.close();
            return datamodel;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public static void addErrorMessage(String msg)
        {
            FacesMessage facesMsg = new
    FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
            FacesContext fc = FacesContext.getCurrentInstance();
            fc.addMessage(null, facesMsg);
        }

        public static void addSuccessMessage(String msg)
        {
            FacesMessage facesMsg = new
    FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
            FacesContext fc = FacesContext.getCurrentInstance();
            fc.addMessage("successInfo", facesMsg);
        }

        public Tipocombustivel findTipocombustivel(Integer id)
```

```
    {
        EntityManager em = getEntityManager();
        Tipocombustivel tipocombustivel1;
        Tipocombustivel o =
(Tipocombustivel)em.find(frotaControle/model/Tipocombustivel, id);
        tipocombustivel1 = o;
        em.close();
        return tipocombustivel1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Tipocombustivel as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        tipocombustivel = new Tipocombustivel();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setTipocombustivelFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setTipocombustivelFromRequestParam();
        setInput(true);
    }
```

```
    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Tipocombustivel tipocombustivel;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.29 frotaControle/controle/ TipocombustivelConverter

```
package frotaControle.controle;

import frotaControle.model.Tipocombustivel;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            TipocombustivelController

public class TipocombustivelConverter
    implements Converter
{

    public TipocombustivelConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
```

```
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            TipocombustivelController controller =
(TipocombustivelController)facesContext.getApplication().getVariableResolver().resol
veVariable(facesContext, "tipocombustivel");
            return controller.findTipocombustivel(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Tipocombustivel)
        {
            Tipocombustivel o = (Tipocombustivel)object;
            return (new
StringBuilder()).append("").append(o.getIdtipocombustivel()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Tipocombustivel").toString());
        }
    }
}
```

11.3.30 frotaControle/controle/ UnidadeController

```
package frotaControle.controle;

import frotaControle.model.Unidade;
import java.util.ArrayList;
import java.util.Collection;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
```

```
import javax.faces.event.ActionEvent;
import javax.faces.model.DataModel;
import javax.faces.model.ListDataModel;
import javax.persistence.*;

public class UnidadeController
{

    public UnidadeController()
    {
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Unidade getUnidade()
    {
        return unidade;
    }

    public void setUnidade(Unidade unidade)
    {
        this.unidade = unidade;
    }

    public DataModel getDetailUnidades()
    {
        return model;
    }

    public void setDetailUnidades(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }

    public String create()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        em.persist(unidade);
        em.getTransaction().commit();
```

```
        addSuccessMessage("Unidade was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        unidade = (Unidade)em.merge(unidade);
        em.getTransaction().commit();
        addSuccessMessage("Unidade was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception exception;
```

```
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Unidade unidade = getUnidadeFromRequestParam();
    unidade = (Unidade)em.merge(unidade);
    em.remove(unidade);
    em.getTransaction().commit();
    addSuccessMessage("Unidade was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Unidade getUnidadeFromRequestParam()
{
    EntityManager em = getEntityManager();
    Unidade unidade1;
    Unidade o = (Unidade)model.getRowData();
    o = (Unidade)em.merge(o);
    unidade1 = o;
    em.close();
    return unidade1;
```

```
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void setUnidadeFromRequestParam()
        {
            Unidade unidade = getUnidadeFromRequestParam();
            setUnidade(unidade);
        }

        public DataModel getUnidades()
        {
            EntityManager em = getEntityManager();
            DataModel datamodel;
            Query q = em.createQuery("select object(o) from Unidade as o");
            q.setMaxResults(batchSize);
            q.setFirstResult(firstItem);
            model = new ListDataModel(q.getResultList());
            datamodel = model;
            em.close();
            return datamodel;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public static void addErrorMessage(String msg)
        {
            FacesMessage facesMsg = new
    FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
            FacesContext fc = FacesContext.getCurrentInstance();
            fc.addMessage(null, facesMsg);
        }

        public static void addSuccessMessage(String msg)
        {
            FacesMessage facesMsg = new
    FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
            FacesContext fc = FacesContext.getCurrentInstance();
            fc.addMessage("successInfo", facesMsg);
        }

        public Unidade findUnidade(Integer id)
```

```
    {
        EntityManager em = getEntityManager();
        Unidade unidade1;
        Unidade o = (Unidade)em.find(frotaControle/model/Unidade, id);
        unidade1 = o;
        em.close();
        return unidade1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Unidade as
o").getSingleResult()).intValue();
        i = count;
        em.close();
        return i;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public void newAction(ActionEvent a0)
    {
        unidade = new Unidade();
        setInput(false);
    }

    public void editAction(ActionEvent a0)
    {
        setUnidadeFromRequestParam();
        setInput(false);
    }

    public void detailAction(ActionEvent a0)
    {
        setUnidadeFromRequestParam();
        setInput(true);
    }
```

```
    public boolean getInput()
    {
        return x;
    }

    public void setInput(boolean x)
    {
        this.x = x;
    }

    private Unidade unidade;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.31 frotaControle/controle/ UnidadeConverter

```
package frotaControle.controle;

import frotaControle.model.Unidade;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            UnidadeController

public class UnidadeConverter
    implements Converter
{

    public UnidadeConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
```

```
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            UnidadeController controller =
(UnidadeController)facesContext.getApplication().getVariableResolver().resolveVaria
ble(facesContext, "unidade");
            return controller.findUnidade(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Unidade)
        {
            Unidade o = (Unidade)object;
            return (new StringBuilder()).append("").append(o.getIdunidade()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Unidade").toString());
        }
    }
}
```

11.3.32 frotaControle/controle/ UploadListener

```
package frotaControle.controle;

import java.io.IOException;
import javax.faces.event.*;
import org.apache.myfaces.custom.fileupload.UploadedFile;

public class UploadListener
    implements ValueChangeListener
{
```

```
    public void processValueChange(ValueChangeEvent event)
        throws AbortProcessingException
    {
        UploadedFile uploadedFile = (UploadedFile)event.getNewValue();
        try
        {
            setFoto(uploadedFile.getBytes());
        }
        catch(IOException ex)
        {
            ex.printStackTrace();
        }
    }

    public UploadListener()
    {
    }

    public byte[] getFoto()
    {
        return foto;
    }

    public void setFoto(byte foto[])
    {
        this.foto = foto;
    }

    private byte foto[];
}
```

11.3.33 frotaControle/controle/ UsuariosistemaController

```
package frotaControle.controle;

import frotaControle.model.Perfilusuario;
import frotaControle.model.Usuariosistema;
import java.util.*;
import javax.faces.application.FacesMessage;
import javax.faces.context.FacesContext;
import javax.faces.event.ActionEvent;
import javax.faces.model.*;
```

```
import javax.persistence.*;

public class UsuariosistemaController
{

    public UsuariosistemaController()
    {
        estaLogado = false;
        batchSize = 20;
        firstItem = 0;
        emf = Persistence.createEntityManagerFactory("FrotaControlesVDPU");
    }

    private EntityManager getEntityManager()
    {
        return emf.createEntityManager();
    }

    public Usuariosistema getUsuariosistema()
    {
        return usuariosistema;
    }

    public void setUsuariosistema(Usuariosistema usuariosistema)
    {
        this.usuariosistema = usuariosistema;
    }

    public DataModel getDetailUsuariosistemas()
    {
        return model;
    }

    public void setDetailUsuariosistemas(Collection m)
    {
        model = new ListDataModel(new ArrayList(m));
    }

    public String createSetup()
    {
        usuariosistema = new Usuariosistema();
        return null;
    }

    public String create()
    {
```

```
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        em.persist(usuariosistema);
        em.getTransaction().commit();
        addSuccessMessage("Usuariosistema was successfully created.");
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
        }
        em.close();
        break MISSING_BLOCK_LABEL_101;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String edit()
    {
        EntityManager em = getEntityManager();
        em.getTransaction().begin();
        usuariosistema = (Usuariosistema)em.merge(usuariosistema);
        em.getTransaction().commit();
        addSuccessMessage("Usuariosistema was successfully updated.");
        em.close();
        break MISSING_BLOCK_LABEL_108;
        Exception ex;
        ex;
        try
        {
            addErrorMessage(ex.getLocalizedMessage());
            em.getTransaction().rollback();
        }
        catch(Exception e)
        {
            addErrorMessage(e.getLocalizedMessage());
```

```
    }
    em.close();
    break MISSING_BLOCK_LABEL_108;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public String destroy()
{
    EntityManager em = getEntityManager();
    em.getTransaction().begin();
    Usuariosistema usuariosistema = getUsuariosistemaFromRequestParam();
    usuariosistema = (Usuariosistema)em.merge(usuariosistema);
    em.remove(usuariosistema);
    em.getTransaction().commit();
    addSuccessMessage("Usuariosistema was successfully deleted.");
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception ex;
    ex;
    try
    {
        addErrorMessage(ex.getLocalizedMessage());
        em.getTransaction().rollback();
    }
    catch(Exception e)
    {
        addErrorMessage(e.getLocalizedMessage());
    }
    em.close();
    break MISSING_BLOCK_LABEL_114;
    Exception exception;
    exception;
    em.close();
    throw exception;
    return null;
}

public Usuariosistema getUsuariosistemaFromRequestParam()
{
    EntityManager em = getEntityManager();
    Usuariosistema usuariosistema1;
    Usuariosistema o = (Usuariosistema)model.getRowData();
```

```
            o = (Usuariosistema)em.merge(o);
            usuariosistema1 = o;
            em.close();
            return usuariosistema1;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void setUsuariosistemaFromRequestParam()
        {
            Usuariosistema usuariosistema = getUsuariosistemaFromRequestParam();
            setUsuariosistema(usuariosistema);
        }

        public DataModel getUsuariosistemas()
        {
            EntityManager em = getEntityManager();
            DataModel datamodel;
            Query q = em.createQuery("select object(o) from Usuariosistema as o");
            q.setMaxResults(batchSize);
            q.setFirstResult(firstItem);
            model = new ListDataModel(q.getResultList());
            datamodel = model;
            em.close();
            return datamodel;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public static void addErrorMessage(String msg)
        {
            FacesMessage facesMsg = new
    FacesMessage(FacesMessage.SEVERITY_ERROR, msg, msg);
            FacesContext fc = FacesContext.getCurrentInstance();
            fc.addMessage(null, facesMsg);
        }

        public static void addSuccessMessage(String msg)
        {
            FacesMessage facesMsg = new
    FacesMessage(FacesMessage.SEVERITY_INFO, msg, msg);
            FacesContext fc = FacesContext.getCurrentInstance();
```

```
        fc.addMessage("successInfo", facesMsg);
    }

    public Usuariosistema findUsuariosistema(Integer id)
    {
        EntityManager em = getEntityManager();
        Usuariosistema usuariosistema1;
        Usuariosistema o =
(Usuariosistema)em.find(frotaControle/model/Usuariosistema, id);
        usuariosistema1 = o;
        em.close();
        return usuariosistema1;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public SelectItem[] getPerfilusuarioIdperfilusuarios()
    {
        EntityManager em = getEntityManager();
        SelectItem aselectitem[];
        List l = em.createQuery("select o from Perfilusuario as o").getResultList();
        SelectItem select[] = new SelectItem[l.size()];
        int i = 0;
        for(Iterator i$ = l.iterator(); i$.hasNext();)
        {
            Perfilusuario x = (Perfilusuario)i$.next();
            select[i++] = new SelectItem(x, x.getNome());
        }

        aselectitem = select;
        em.close();
        return aselectitem;
        Exception exception;
        exception;
        em.close();
        throw exception;
    }

    public int getItemCount()
    {
        EntityManager em = getEntityManager();
        int i;
        int count = ((Long)em.createQuery("select count(o) from Usuariosistema as
o").getSingleResult()).intValue();
```

```
            i = count;
            em.close();
            return i;
            Exception exception;
            exception;
            em.close();
            throw exception;
        }

        public void newAction(ActionEvent a0)
        {
            usuariosistema = new Usuariosistema();
            setInput(false);
        }

        public void editAction(ActionEvent a0)
        {
            setUsuariosistemaFromRequestParam();
            setInput(false);
        }

        public void detailAction(ActionEvent a0)
        {
            setUsuariosistemaFromRequestParam();
        }

        public boolean getInput()
        {
            return x;
        }

        public void setInput(boolean x)
        {
            this.x = x;
        }

        public String loginAction()
        {
            EntityManager em = getEntityManager();
            Usuariosistema o = (Usuariosistema)em.createQuery("SELECT u FROM
Usuariosistema u WHERE u.loginusuario =
:loginusuario").setParameter("loginusuario", login).getSingleResult();
            if(!o.getSenhausuario().equals(senha))
            {
                this;
                addErrorMessage("Senha Incorreta");
```

```
            estaLogado = false;
        } else
        {
            estaLogado = true;
        }
        em.close();
        break MISSING_BLOCK_LABEL_129;
        NoResultException e;
        e;
        this;
        addErrorMessage("Usu\341rio Inexistente");
        estaLogado = false;
        em.close();
        break MISSING_BLOCK_LABEL_129;
        e;
        this;
        addErrorMessage("Digitar nome do usu\341rio e senha");
        estaLogado = false;
        em.close();
        break MISSING_BLOCK_LABEL_129;
        Exception exception;
        exception;
        em.close();
        throw exception;
        return null;
    }

    public String sair()
    {
        estaLogado = false;
        return "view_inicio";
    }

    public boolean isEstaLogado()
    {
        return estaLogado;
    }

    public void setEstaLogado(boolean estaLogado)
    {
        this.estaLogado = estaLogado;
    }

    public String getLogin()
    {
        return login;
```

```
    }

    public void setLogin(String login)
    {
        this.login = login;
    }

    public String getSenha()
    {
        return senha;
    }

    public void setSenha(String senha)
    {
        this.senha = senha;
    }

    private Usuariosistema usuariosistema;
    private boolean estaLogado;
    private String login;
    private String senha;
    private DataModel model;
    private EntityManagerFactory emf;
    private boolean x;
    private int batchSize;
    private int firstItem;
}
```

11.3.34 frotaControle/controle/ UsuariosistemaConverter

```
package frotaControle.controle;

import frotaControle.model.Usuariosistema;
import javax.faces.application.Application;
import javax.faces.component.UIComponent;
import javax.faces.context.FacesContext;
import javax.faces.convert.Converter;
import javax.faces.el.VariableResolver;

// Referenced classes of package frotaControle.controle:
//            UsuariosistemaController

public class UsuariosistemaConverter
```

```
    implements Converter
{

    public UsuariosistemaConverter()
    {
    }

    public Object getAsObject(FacesContext facesContext, UIComponent
uIComponent, String string)
    {
        if(string == null)
        {
            return null;
        } else
        {
            Integer id = new Integer(string);
            UsuariosistemaController controller =
(UsuariosistemaController)facesContext.getApplication().getVariableResolver().resol
veVariable(facesContext, "usuariosistema");
            return controller.findUsuariosistema(id);
        }
    }

    public String getAsString(FacesContext facesContext, UIComponent
uIComponent, Object object)
    {
        if(object == null)
            return null;
        if(object instanceof Usuariosistema)
        {
            Usuariosistema o = (Usuariosistema)object;
            return (new
StringBuilder()).append("").append(o.getIdusuariosistema()).toString();
        } else
        {
            throw new IllegalArgumentException((new
StringBuilder()).append("object:").append(object).append(" of
type:").append(object.getClass().getName()).append("; expected type:
frotaControle.model.Usuariosistema").toString());
        }
    }
}
```

11.4 Classes Model

Classes que definem o acesso ao sistema.

11.4.1 frotaControle/model/Atividade

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;
import java.util.Date;

// Referenced classes of package frotaControle.model:
//            Integrante, Motoristaoperador

public class Atividade
    implements Serializable
{

    public Atividade()
    {
    }

    public Atividade(Integer idatividade)
    {
        this.idatividade = idatividade;
    }

    public Atividade(Integer idatividade, Date dataatividade, Date horainicioatividade)
    {
        this.idatividade = idatividade;
        this.dataatividade = dataatividade;
        this.horainicioatividade = horainicioatividade;
    }

    public Integer getIdatividade()
    {
        return idatividade;
    }

    public void setIdatividade(Integer idatividade)
    {
        this.idatividade = idatividade;
```

```
}

public Date getDataatividade()
{
    return dataatividade;
}

public void setDataatividade(Date dataatividade)
{
    this.dataatividade = dataatividade;
}

public Date getHorainicioatividade()
{
    return horainicioatividade;
}

public void setHorainicioatividade(Date horainicioatividade)
{
    this.horainicioatividade = horainicioatividade;
}

public Date getDataterminoatividade()
{
    return dataterminoatividade;
}

public void setDataterminoatividade(Date dataterminoatividade)
{
    this.dataterminoatividade = dataterminoatividade;
}

public Date getHoratermino()
{
    return horatermino;
}

public void setHoratermino(Date horatermino)
{
    this.horatermino = horatermino;
}

public Double getKminicial()
{
    return kminicial;
}
```

```
public void setKminicial(Double kminicial)
{
    this.kminicial = kminicial;
}

public String getObsatividade()
{
    return obsatividade;
}

public void setObsatividade(String obsatividade)
{
    this.obsatividade = obsatividade;
}

public Double getKmfinal()
{
    return kmfinal;
}

public void setKmfinal(Double kmfinal)
{
    this.kmfinal = kmfinal;
}

public boolean getMulta()
{
    return multa;
}

public void setMulta(boolean multa)
{
    this.multa = multa;
}

public String getGravidademulta()
{
    return gravidademulta;
}

public void setGravidademulta(String gravidademulta)
{
    this.gravidademulta = gravidademulta;
}
```

```
public String getObspercurso()
{
    return obspercurso;
}

public void setObspercurso(String obspercurso)
{
    this.obspercurso = obspercurso;
}

public String getOutrasobs()
{
    return outrasobs;
}

public void setOutrasobs(String outrasobs)
{
    this.outrasobs = outrasobs;
}

public String getAcidente()
{
    return acidente;
}

public void setAcidente(String acidente)
{
    this.acidente = acidente;
}

public Collection getManutencaoCollection()
{
    return manutencaoCollection;
}

public void setManutencaoCollection(Collection manutencaoCollection)
{
    this.manutencaoCollection = manutencaoCollection;
}

public Integrante getIntegranteIdintegrante()
{
    return integranteIdintegrante;
}

public void setIntegranteIdintegrante(Integrante integranteIdintegrante)
```

```
    {
        this.integranteIdintegrante = integranteIdintegrante;
    }

    public Motoristaoperador getMotoristaoperadorIdmotoristaoperador()
    {
        return motoristaoperadorIdmotoristaoperador;
    }

    public void setMotoristaoperadorIdmotoristaoperador(Motoristaoperador
motoristaoperadorIdmotoristaoperador)
    {
        this.motoristaoperadorIdmotoristaoperador =
motoristaoperadorIdmotoristaoperador;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idatividade == null ? 0 : idatividade.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Atividade))
            return false;
        Atividade other = (Atividade)object;
        return idatividade == other.idatividade || idatividade != null &&
idatividade.equals(other.idatividade);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Atividade[idatividade=").append(idatividade).append(
"]").toString();
    }

    public boolean isCancelado()
    {
        return cancelado;
    }

    public void setCancelado(boolean cancelado)
    {
```

```
        this.cancelado = cancelado;
    }

    private Integer idatividade;
    private Date dataatividade;
    private Date horainicioatividade;
    private Date dataterminoatividade;
    private Date horatermino;
    private Double kminicial;
    private String obsatividade;
    private Double kmfinal;
    private boolean multa;
    private boolean cancelado;
    private String gravidademulta;
    private String obspercurso;
    private String outrasobs;
    private String acidente;
    private Collection manutencaoCollection;
    private Integrante integranteIdintegrante;
    private Motoristaoperador motoristaoperadorIdmotoristaoperador;
}
```

11.4.2 frotaControle/model/Categoriahabilitacao

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

public class Categoriahabilitacao
    implements Serializable
{

    public Categoriahabilitacao()
    {
    }

    public Categoriahabilitacao(Integer idcategoriahabilitacao)
    {
        this.idcategoriahabilitacao = idcategoriahabilitacao;
    }
```

```
    public Categoriahabilitacao(Integer idcategoriahabilitacao, String
categoriahabilitacao, String descricaocategoriahabilitacao)
    {
        this.idcategoriahabilitacao = idcategoriahabilitacao;
        this.categoriahabilitacao = categoriahabilitacao;
        this.descricaocategoriahabilitacao = descricaocategoriahabilitacao;
    }

    public Integer getIdcategoriahabilitacao()
    {
        return idcategoriahabilitacao;
    }

    public void setIdcategoriahabilitacao(Integer idcategoriahabilitacao)
    {
        this.idcategoriahabilitacao = idcategoriahabilitacao;
    }

    public String getCategoriahabilitacao()
    {
        return categoriahabilitacao;
    }

    public void setCategoriahabilitacao(String categoriahabilitacao)
    {
        this.categoriahabilitacao = categoriahabilitacao;
    }

    public String getDescricaocategoriahabilitacao()
    {
        return descricaocategoriahabilitacao;
    }

    public void setDescricaocategoriahabilitacao(String descricaocategoriahabilitacao)
    {
        this.descricaocategoriahabilitacao = descricaocategoriahabilitacao;
    }

    public Collection getIntegranteCollection()
    {
        return integranteCollection;
    }

    public void setIntegranteCollection(Collection integranteCollection)
    {
        this.integranteCollection = integranteCollection;
```

```
    }

    public Collection getMotoristaoperadorCollection()
    {
        return motoristaoperadorCollection;
    }

    public void setMotoristaoperadorCollection(Collection
motoristaoperadorCollection)
    {
        this.motoristaoperadorCollection = motoristaoperadorCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idcategoriahabilitacao == null ? 0 : idcategoriahabilitacao.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Categoriahabilitacao))
            return false;
        Categoriahabilitacao other = (Categoriahabilitacao)object;
        return idcategoriahabilitacao == other.idcategoriahabilitacao ||
idcategoriahabilitacao != null &&
idcategoriahabilitacao.equals(other.idcategoriahabilitacao);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Categoriahabilitacao[idcategoriahabilitacao=").appen
d(idcategoriahabilitacao).append("]").toString();
    }

    private Integer idcategoriahabilitacao;
    private String categoriahabilitacao;
    private String descricaocategoriahabilitacao;
    private Collection integranteCollection;
    private Collection motoristaoperadorCollection;
}
```

11.4.3 frotaControle/model/Fornecedor

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

public class Fornecedor
    implements Serializable
{

    public Fornecedor()
    {
    }

    public Fornecedor(Integer idfornecedor)
    {
        this.idfornecedor = idfornecedor;
    }

    public Fornecedor(Integer idfornecedor, String razaosocial, int cnpj)
    {
        this.idfornecedor = idfornecedor;
        this.razaosocial = razaosocial;
        this.cnpj = cnpj;
    }

    public Integer getIdfornecedor()
    {
        return idfornecedor;
    }

    public void setIdfornecedor(Integer idfornecedor)
    {
        this.idfornecedor = idfornecedor;
    }

    public String getRazaosocial()
    {
        return razaosocial;
    }

    public void setRazaosocial(String razaosocial)
    {
        this.razaosocial = razaosocial;
    }
```

```
public String getDescricao()
{
    return descricao;
}

public void setDescricao(String descricao)
{
    this.descricao = descricao;
}

public int getCnpj()
{
    return cnpj;
}

public void setCnpj(int cnpj)
{
    this.cnpj = cnpj;
}

public String getEndereco()
{
    return endereco;
}

public void setEndereco(String endereco)
{
    this.endereco = endereco;
}

public Integer getTelefone()
{
    return telefone;
}

public void setTelefone(Integer telefone)
{
    this.telefone = telefone;
}

public Integer getFax()
{
    return fax;
}

public void setFax(Integer fax)
```

```
{
    this.fax = fax;
}

public Integer getDdd()
{
    return ddd;
}

public void setDdd(Integer ddd)
{
    this.ddd = ddd;
}

public String getCidade()
{
    return cidade;
}

public void setCidade(String cidade)
{
    this.cidade = cidade;
}

public String getEstado()
{
    return estado;
}

public void setEstado(String estado)
{
    this.estado = estado;
}

public Integer getCep()
{
    return cep;
}

public void setCep(Integer cep)
{
    this.cep = cep;
}

public String getEmail()
{
```

```
        return email;
    }

    public void setEmail(String email)
    {
        this.email = email;
    }

    public String getPage()
    {
        return page;
    }

    public void setPage(String page)
    {
        this.page = page;
    }

    public Collection getNotafiscalgastoCollection()
    {
        return notafiscalgastoCollection;
    }

    public void setNotafiscalgastoCollection(Collection notafiscalgastoCollection)
    {
        this.notafiscalgastoCollection = notafiscalgastoCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idfornecedor == null ? 0 : idfornecedor.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Fornecedor))
            return false;
        Fornecedor other = (Fornecedor)object;
        return idfornecedor == other.idfornecedor || idfornecedor != null &&
idfornecedor.equals(other.idfornecedor);
    }

    public String toString()
    {
```

```
        return (new
StringBuilder()).append("frotaControle.model.Fornecedor[idfornecedor=").append(idf
ornecedor).append("]").toString();
    }

    private Integer idfornecedor;
    private String razaosocial;
    private String descricao;
    private int cnpj;
    private String endereco;
    private Integer telefone;
    private Integer fax;
    private Integer ddd;
    private String cidade;
    private String estado;
    private Integer cep;
    private String email;
    private String page;
    private Collection notafiscalgastoCollection;
}
```

11.4.4 frotaControle/model/Grupointegrante

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

public class Grupointegrante
    implements Serializable
{

    public Grupointegrante()
    {
    }

    public Grupointegrante(Integer idgrupointegrante)
    {
        this.idgrupointegrante = idgrupointegrante;
    }

    public Grupointegrante(Integer idgrupointegrante, String nomegrupointegrante)
    {
```

```
        this.idgrupointegrante = idgrupointegrante;
        this.nomegrupointegrante = nomegrupointegrante;
    }

    public Integer getIdgrupointegrante()
    {
        return idgrupointegrante;
    }

    public void setIdgrupointegrante(Integer idgrupointegrante)
    {
        this.idgrupointegrante = idgrupointegrante;
    }

    public String getNomegrupointegrante()
    {
        return nomegrupointegrante;
    }

    public void setNomegrupointegrante(String nomegrupointegrante)
    {
        this.nomegrupointegrante = nomegrupointegrante;
    }

    public Collection getSubgrupointegrantesCollection()
    {
        return subgrupointegrantesCollection;
    }

    public void setSubgrupointegrantesCollection(Collection
subgrupointegrantesCollection)
    {
        this.subgrupointegrantesCollection = subgrupointegrantesCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idgrupointegrante == null ? 0 : idgrupointegrante.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Grupointegrante))
            return false;
```

```
        Grupointegrante other = (Grupointegrante)object;
        return idgrupointegrante == other.idgrupointegrante || idgrupointegrante != null
&& idgrupointegrante.equals(other.idgrupointegrante);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("frotaControle.model.Grupointegrante[idgrupointegrante=").a
ppend(idgrupointegrante).append("]").toString();
    }

    private Integer idgrupointegrante;
    private static final long serialVersionUID = 1L;
    private String nomegrupointegrante;
    private Collection subgrupointegrantesCollection;
}
```

11.4.5 frotaControle/model/Integrante

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;
import java.util.Date;

// Referenced classes of package frotaControle.model:
//          Categoriahabilitacao, Subgrupointegrantes, Tipocombustivel

public class Integrante
    implements Serializable
{

    public Integrante()
    {
    }

    public Integrante(Integer idintegrante)
    {
        this.idintegrante = idintegrante;
    }
```

```
    public Integrante(Integer idintegrante, String nomeintegrante, String placa, Integer
anofabricacao, boolean ativointegrante)
    {
        this.idintegrante = idintegrante;
        this.nomeintegrante = nomeintegrante;
        this.placa = placa;
        this.anofabricacao = anofabricacao;
        this.ativointegrante = ativointegrante;
    }

    public Integer getIdintegrante()
    {
        return idintegrante;
    }

    public void setIdintegrante(Integer idintegrante)
    {
        this.idintegrante = idintegrante;
    }

    public String getNomeintegrante()
    {
        return nomeintegrante;
    }

    public void setNomeintegrante(String nomeintegrante)
    {
        this.nomeintegrante = nomeintegrante;
    }

    public String getPlaca()
    {
        return placa;
    }

    public void setPlaca(String placa)
    {
        this.placa = placa;
    }

    public String getChassis()
    {
        return chassis;
    }

    public void setChassis(String chassis)
```

```
{
    this.chassis = chassis;
}

public Integer getAnofabricacao()
{
    return anofabricacao;
}

public void setAnofabricacao(Integer anofabricacao)
{
    this.anofabricacao = anofabricacao;
}

public Integer getAnomodelo()
{
    return anomodelo;
}

public void setAnomodelo(Integer anomodelo)
{
    this.anomodelo = anomodelo;
}

public Date getDatacompraintegrante()
{
    return datacompraintegrante;
}

public void setDatacompraintegrante(Date datacompraintegrante)
{
    this.datacompraintegrante = datacompraintegrante;
}

public String getMarca()
{
    return marca;
}

public void setMarca(String marca)
{
    this.marca = marca;
}

public String getPotenciamotor()
{
```

```
    return potenciamotor;
}

public void setPotenciamotor(String potenciamotor)
{
    this.potenciamotor = potenciamotor;
}

public boolean getAtivointegrante()
{
    return ativointegrante;
}

public void setAtivointegrante(boolean ativointegrante)
{
    this.ativointegrante = ativointegrante;
}

public Double getKminicial()
{
    return kminicial;
}

public void setKminicial(Double kminicial)
{
    this.kminicial = kminicial;
}

public Double getKmatual()
{
    return kmatual;
}

public void setKmatual(Double kmatual)
{
    this.kmatual = kmatual;
}

public Date getHorainicial()
{
    return horainicial;
}

public void setHorainicial(Date horainicial)
{
    this.horainicial = horainicial;
```

```
    }

    public Date getHorafinal()
    {
        return horafinal;
    }

    public void setHorafinal(Date horafinal)
    {
        this.horafinal = horafinal;
    }

    public Collection getManutencaoCollection()
    {
        return manutencaoCollection;
    }

    public void setManutencaoCollection(Collection manutencaoCollection)
    {
        this.manutencaoCollection = manutencaoCollection;
    }

    public Categoriahabilitacao getCategoriahabilitacaoIdcategoriahabilitacao()
    {
        return categoriahabilitacao_idcategoriahabilitacao;
    }

    public void setCategoriahabilitacaoIdcategoriahabilitacao(Categoriahabilitacao
categoriahabilitacaoIdcategoriahabilitacao)
    {
        categoriahabilitacao_idcategoriahabilitacao =
categoriahabilitacaoIdcategoriahabilitacao;
    }

    public Subgrupointegrantes getSubgrupointegrantesIdsubgrupointegrantes()
    {
        return subgrupointegrantes_idsubgrupointegrantes;
    }

    public void setSubgrupointegrantesIdsubgrupointegrantes(Subgrupointegrantes
subgrupointegrantes_idsubgrupointegrantes)
    {
        this.subgrupointegrantes_idsubgrupointegrantes =
subgrupointegrantes_idsubgrupointegrantes;
    }
```

```
    public Tipocombustivel getTipocombustivelIdtipocombustivel()
    {
        return tipocombustivelIdtipocombustivel;
    }

    public void setTipocombustivelIdtipocombustivel(Tipocombustivel
tipocombustivelIdtipocombustivel)
    {
        this.tipocombustivelIdtipocombustivel = tipocombustivelIdtipocombustivel;
    }

    public Collection getNotaservicoCollection()
    {
        return notaservicoCollection;
    }

    public void setNotaservicoCollection(Collection notaservicoCollection)
    {
        this.notaservicoCollection = notaservicoCollection;
    }

    public Collection getAtividadeCollection()
    {
        return atividadeCollection;
    }

    public void setAtividadeCollection(Collection atividadeCollection)
    {
        this.atividadeCollection = atividadeCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idintegrante == null ? 0 : idintegrante.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Integrante))
            return false;
        Integrante other = (Integrante)object;
        return idintegrante == other.idintegrante || idintegrante != null &&
idintegrante.equals(other.idintegrante);
    }
```

```
    public String toString()
    {
        return (new
StringBuilder()).append("frotaControle.model.Integrante[idintegrante=").append(idinte
grante).append("]").toString();
    }

    public boolean isDisponivel()
    {
        return disponivel;
    }

    public void setDisponivel(boolean disponivel)
    {
        this.disponivel = disponivel;
    }

    private Integer idintegrante;
    private String nomeintegrante;
    private String placa;
    private String chassis;
    private Integer anofabricacao;
    private Integer anomodelo;
    private Date datacompraintegrante;
    private String marca;
    private String potenciamotor;
    private boolean ativointegrante;
    private Double kminicial;
    private Double kmatual;
    private Date horainicial;
    private Date horafinal;
    private boolean disponivel;
    private Collection manutencaoCollection;
    private Categoriahabilitacao categoriahabilitacao_idcategoriahabilitacao;
    private Subgrupointegrantes subgrupointegrantes_idsubgrupointegrantes;
    private Tipocombustivel tipocombustivelIdtipocombustivel;
    private Collection notaservicoCollection;
    private Collection atividadeCollection;
}
```

11.4.6 frotaControle/model/Itens

```
package frotaControle.model;

import java.io.Serializable;
import java.math.BigDecimal;
import java.util.Collection;

// Referenced classes of package frotaControle.model:
//            Unidade

public class Itens
    implements Serializable
{

    public Itens()
    {
    }

    public Itens(Integer iditens)
    {
        this.iditens = iditens;
    }

    public Itens(Integer iditens, String descricaoitens, String marcaitens, String tipo)
    {
        this.iditens = iditens;
        this.descricaoitens = descricaoitens;
        this.marcaitens = marcaitens;
        this.tipo = tipo;
    }

    public Integer getIditens()
    {
        return iditens;
    }

    public void setIditens(Integer iditens)
    {
        this.iditens = iditens;
    }

    public String getDescricaoitens()
    {
        return descricaoitens;
    }
```

```
public void setDescricaoitens(String descricaoitens)
{
    this.descricaoitens = descricaoitens;
}

public String getMarcaitens()
{
    return marcaitens;
}

public void setMarcaitens(String marcaitens)
{
    this.marcaitens = marcaitens;
}

public BigDecimal getValor()
{
    return valor;
}

public void setValor(BigDecimal valor)
{
    this.valor = valor;
}

public String getTipo()
{
    return tipo;
}

public void setTipo(String tipo)
{
    this.tipo = tipo;
}

public Unidade getUnidadeIdunidade()
{
    return unidadeIdunidade;
}

public void setUnidadeIdunidade(Unidade unidadeIdunidade)
{
    this.unidadeIdunidade = unidadeIdunidade;
}

public Collection getNotafiscalgastoHasItensCollection()
```

```
    {
        return notafiscalgastoHasItensCollection;
    }

    public void setNotafiscalgastoHasItensCollection(Collection
notafiscalgastoHasItensCollection)
    {
        this.notafiscalgastoHasItensCollection = notafiscalgastoHasItensCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += iditens == null ? 0 : iditens.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Itens))
            return false;
        Itens other = (Itens)object;
        return iditens == other.iditens || iditens != null && iditens.equals(other.iditens);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Itens[iditens=").append(iditens).append("]").toString();
    }

    private Integer iditens;
    private String descricaoitens;
    private String marcaitens;
    private BigDecimal valor;
    private String tipo;
    private Unidade unidadeIdunidade;
    private Collection notafiscalgastoHasItensCollection;
}
```

11.4.7 frotaControle/model/Manutencao

```
package frotaControle.model;
```

```
import java.io.Serializable;
import java.util.Date;

// Referenced classes of package frotaControle.model:
//            Atividade, Integrante

public class Manutencao
    implements Serializable
{

    public Manutencao()
    {
    }

    public Manutencao(Integer idmanutencao)
    {
        this.idmanutencao = idmanutencao;
    }

    public Manutencao(Integer idmanutencao, String descricaomanutencao, Date
datamanutencao, String urgenciamanutencao, String categoriamanutencao)
    {
        this.idmanutencao = idmanutencao;
        this.descricaomanutencao = descricaomanutencao;
        this.datamanutencao = datamanutencao;
        this.urgenciamanutencao = urgenciamanutencao;
        this.categoriamanutencao = categoriamanutencao;
    }

    public Integer getIdmanutencao()
    {
        return idmanutencao;
    }

    public void setIdmanutencao(Integer idmanutencao)
    {
        this.idmanutencao = idmanutencao;
    }

    public String getDescricaomanutencao()
    {
        return descricaomanutencao;
    }

    public void setDescricaomanutencao(String descricaomanutencao)
```

```
{
    this.descricaomanutencao = descricaomanutencao;
}

public Date getDatamanutencao()
{
    return datamanutencao;
}

public void setDatamanutencao(Date datamanutencao)
{
    this.datamanutencao = datamanutencao;
}

public Date getDatareparo()
{
    return datareparo;
}

public void setDatareparo(Date datareparo)
{
    this.datareparo = datareparo;
}

public String getUrgenciamanutencao()
{
    return urgenciamanutencao;
}

public void setUrgenciamanutencao(String urgenciamanutencao)
{
    this.urgenciamanutencao = urgenciamanutencao;
}

public String getCategoriamanutencao()
{
    return categoriamanutencao;
}

public void setCategoriamanutencao(String categoriamanutencao)
{
    this.categoriamanutencao = categoriamanutencao;
}

public String getObsmanutencao()
{
```

```
    return obsmanutencao;
}

public void setObsmanutencao(String obsmanutencao)
{
    this.obsmanutencao = obsmanutencao;
}

public Double getKm()
{
    return km;
}

public void setKm(Double km)
{
    this.km = km;
}

public Double getKilometrosProgramados()
{
    return kilometrosProgramados;
}

public void setKilometrosProgramados(Double kilometrosProgramados)
{
    this.kilometrosProgramados = kilometrosProgramados;
}

public Date getHorasPrevistas()
{
    return horasPrevistas;
}

public void setHorasPrevistas(Date horasPrevistas)
{
    this.horasPrevistas = horasPrevistas;
}

public Date getHoras()
{
    return horas;
}

public void setHoras(Date horas)
{
    this.horas = horas;
```

```
    }

    public Atividade getAtividadeIdatividade()
    {
        return atividadeIdatividade;
    }

    public void setAtividadeIdatividade(Atividade atividadeIdatividade)
    {
        this.atividadeIdatividade = atividadeIdatividade;
    }

    public Integrante getIntegranteIdintegrante()
    {
        return integranteIdintegrante;
    }

    public void setIntegranteIdintegrante(Integrante integranteIdintegrante)
    {
        this.integranteIdintegrante = integranteIdintegrante;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idmanutencao == null ? 0 : idmanutencao.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Manutencao))
            return false;
        Manutencao other = (Manutencao)object;
        return idmanutencao == other.idmanutencao || idmanutencao != null &&
idmanutencao.equals(other.idmanutencao);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("frotaControle.model.Manutencao[idmanutencao=").append(i
dmanutencao).append("]").toString();
    }

    private Integer idmanutencao;
```

```
    private String descricaomanutencao;
    private Date datamanutencao;
    private Date datareparo;
    private String urgenciamanutencao;
    private String categoriamanutencao;
    private String obsmanutencao;
    private Double km;
    private Double kilometrosProgramados;
    private Date horasPrevistas;
    private Date horas;
    private Atividade atividadeIdatividade;
    private Integrante integranteIdintegrante;
}
```

11.4.8 frotaControle/model/Motoristaoperador

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;
import java.util.Date;

// Referenced classes of package frotaControle.model:
//            Categoriahabilitacao

public class Motoristaoperador
    implements Serializable
{

    public Motoristaoperador()
    {
    }

    public Motoristaoperador(Integer idmotoristaoperador)
    {
        this.idmotoristaoperador = idmotoristaoperador;
    }

    public Motoristaoperador(Integer idmotoristaoperador, String nome, int cpf, String
matricula, String classificacaomotorista)
    {
        this.idmotoristaoperador = idmotoristaoperador;
        this.nome = nome;
```

```
    this.cpf = cpf;
    this.matricula = matricula;
    this.classificacaomotorista = classificacaomotorista;
}

public Integer getIdmotoristaoperador()
{
    return idmotoristaoperador;
}

public void setIdmotoristaoperador(Integer idmotoristaoperador)
{
    this.idmotoristaoperador = idmotoristaoperador;
}

public String getNome()
{
    return nome;
}

public void setNome(String nome)
{
    this.nome = nome;
}

public int getCpf()
{
    return cpf;
}

public void setCpf(int cpf)
{
    this.cpf = cpf;
}

public String getMatricula()
{
    return matricula;
}

public void setMatricula(String matricula)
{
    this.matricula = matricula;
}

public Date getValidadecarteirahab()
```

```
    {
        return validadecarteirahab;
    }

    public void setValidadecarteirahab(Date validadecarteirahab)
    {
        this.validadecarteirahab = validadecarteirahab;
    }

    public byte[] getFoto()
    {
        return foto;
    }

    public void setFoto(byte foto[])
    {
        this.foto = foto;
    }

    public String getClassificacaomotorista()
    {
        return classificacaomotorista;
    }

    public void setClassificacaomotorista(String classificacaomotorista)
    {
        this.classificacaomotorista = classificacaomotorista;
    }

    public Categoriahabilitacao getCategoriahabilitacaoIdcategoriahabilitacao()
    {
        return categoriahabilitacao_idcategoriahabilitacao;
    }

    public void setCategoriahabilitacaoIdcategoriahabilitacao(Categoriahabilitacao
categoriahabilitacaoIdcategoriahabilitacao)
    {
        categoriahabilitacao_idcategoriahabilitacao =
categoriahabilitacaoIdcategoriahabilitacao;
    }

    public Collection getAtividadeCollection()
    {
        return atividadeCollection;
    }
```

```
    public void setAtividadeCollection(Collection atividadeCollection)
    {
        this.atividadeCollection = atividadeCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idmotoristaoperador == null ? 0 : idmotoristaoperador.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Motoristaoperador))
            return false;
        Motoristaoperador other = (Motoristaoperador)object;
        return idmotoristaoperador == other.idmotoristaoperador || idmotoristaoperador
!= null && idmotoristaoperador.equals(other.idmotoristaoperador);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("frotaControle.model.Motoristaoperador[idmotoristaoperador
=").append(idmotoristaoperador).append("]").toString();
    }

    public String getTelefone()
    {
        return telefone;
    }

    public void setTelefone(String telefone)
    {
        this.telefone = telefone;
    }

    public String getCelular()
    {
        return celular;
    }

    public void setCelular(String celular)
    {
        this.celular = celular;
```

```
    }

    public String getRamal()
    {
        return ramal;
    }

    public void setRamal(String ramal)
    {
        this.ramal = ramal;
    }

    public boolean isDisponivel()
    {
        return disponivel;
    }

    public void setDisponivel(boolean disponivel)
    {
        this.disponivel = disponivel;
    }

    private Integer idmotoristaoperador;
    private String nome;
    private int cpf;
    private String matricula;
    private Date validadecarteirahab;
    private byte foto[];
    private String classificacaomotorista;
    private String telefone;
    private String celular;
    private String ramal;
    private boolean disponivel;
    private Categoriahabilitacao categoriahabilitacao_idcategoriahabilitacao;
    private Collection atividadeCollection;
}
```

11.4.9 frotaControle/model/Notafiscalgasto

```
package frotaControle.model;

import java.io.Serializable;
import java.math.BigDecimal;
```

```
import java.util.Collection;
import java.util.Date;

// Referenced classes of package frotaControle.model:
//            Fornecedor

public class Notafiscalgasto
    implements Serializable
{

    public Notafiscalgasto()
    {
    }

    public Notafiscalgasto(Integer idnotafiscalgasto)
    {
        this.idnotafiscalgasto = idnotafiscalgasto;
    }

    public Notafiscalgasto(Integer idnotafiscalgasto, int numeronotagasto, Date
datagasto, BigDecimal valornotagasto, Date datapreenchegasto)
    {
        this.idnotafiscalgasto = idnotafiscalgasto;
        this.numeronotagasto = numeronotagasto;
        this.datagasto = datagasto;
        this.valornotagasto = valornotagasto;
        this.datapreenchegasto = datapreenchegasto;
    }

    public Integer getIdnotafiscalgasto()
    {
        return idnotafiscalgasto;
    }

    public void setIdnotafiscalgasto(Integer idnotafiscalgasto)
    {
        this.idnotafiscalgasto = idnotafiscalgasto;
    }

    public int getNumeronotagasto()
    {
        return numeronotagasto;
    }

    public void setNumeronotagasto(int numeronotagasto)
    {
```

```
        this.numeronotagasto = numeronotagasto;
    }

    public Date getDatagasto()
    {
        return datagasto;
    }

    public void setDatagasto(Date datagasto)
    {
        this.datagasto = datagasto;
    }

    public BigDecimal getValornotagasto()
    {
        return valornotagasto;
    }

    public void setValornotagasto(BigDecimal valornotagasto)
    {
        this.valornotagasto = valornotagasto;
    }

    public Date getDatapreenchegasto()
    {
        return datapreenchegasto;
    }

    public void setDatapreenchegasto(Date datapreenchegasto)
    {
        this.datapreenchegasto = datapreenchegasto;
    }

    public Collection getNotafiscalgastoHasItensCollection()
    {
        return notafiscalgastoHasItensCollection;
    }

    public void setNotafiscalgastoHasItensCollection(Collection
notafiscalgastoHasItensCollection)
    {
        this.notafiscalgastoHasItensCollection = notafiscalgastoHasItensCollection;
    }

    public Fornecedor getFornecedorIdfornecedor()
    {
```

```
        return fornecedorIdfornecedor;
    }

    public void setFornecedorIdfornecedor(Fornecedor fornecedorIdfornecedor)
    {
        this.fornecedorIdfornecedor = fornecedorIdfornecedor;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idnotafiscalgasto == null ? 0 : idnotafiscalgasto.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Notafiscalgasto))
            return false;
        Notafiscalgasto other = (Notafiscalgasto)object;
        return idnotafiscalgasto == other.idnotafiscalgasto || idnotafiscalgasto != null &&
idnotafiscalgasto.equals(other.idnotafiscalgasto);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Notafiscalgasto[idnotafiscalgasto=").append(idnotafis
calgasto).append("]").toString();
    }

    private Integer idnotafiscalgasto;
    private int numeronotagasto;
    private Date datagasto;
    private BigDecimal valornotagasto;
    private Date datapreenchegasto;
    private Collection notafiscalgastoHasItensCollection;
    private Fornecedor fornecedorIdfornecedor;
}
```

11.4.10 frotaControle/model/NotafiscalgastoHasItens

```
package frotaControle.model;
```

```
import java.io.Serializable;
import java.math.BigDecimal;
import java.util.Collection;

// Referenced classes of package frotaControle.model:
//            NotafiscalgastoHasItensPK, Itens, Notafiscalgasto

public class NotafiscalgastoHasItens
    implements Serializable
{

    public NotafiscalgastoHasItens()
    {
    }

    public NotafiscalgastoHasItens(NotafiscalgastoHasItensPK
notafiscalgastoHasItensPK)
    {
        this.notafiscalgastoHasItensPK = notafiscalgastoHasItensPK;
    }

    public NotafiscalgastoHasItens(int itensIditens, int
notafiscalgastoIdnotafiscalgasto)
    {
        notafiscalgastoHasItensPK = new NotafiscalgastoHasItensPK(itensIditens,
notafiscalgastoIdnotafiscalgasto);
    }

    public NotafiscalgastoHasItensPK getNotafiscalgastoHasItensPK()
    {
        return notafiscalgastoHasItensPK;
    }

    public void setNotafiscalgastoHasItensPK(NotafiscalgastoHasItensPK
notafiscalgastoHasItensPK)
    {
        this.notafiscalgastoHasItensPK = notafiscalgastoHasItensPK;
    }

    public Double getQuantidadegasto()
    {
        return quantidadegasto;
    }

    public void setQuantidadegasto(Double quantidadegasto)
```

```
{
    this.quantidadegasto = quantidadegasto;
}

public BigDecimal getValortotalgasto()
{
    return valortotalgasto;
}

public void setValortotalgasto(BigDecimal valortotalgasto)
{
    this.valortotalgasto = valortotalgasto;
}

public Collection getNotaservicoCollection()
{
    return notaservicoCollection;
}

public void setNotaservicoCollection(Collection notaservicoCollection)
{
    this.notaservicoCollection = notaservicoCollection;
}

public Itens getItens()
{
    return itens;
}

public void setItens(Itens itens)
{
    this.itens = itens;
}

public Notafiscalgasto getNotafiscalgasto()
{
    return notafiscalgasto;
}

public void setNotafiscalgasto(Notafiscalgasto notafiscalgasto)
{
    this.notafiscalgasto = notafiscalgasto;
}

public int hashCode()
{
```

```
        int hash = 0;
        hash += notafiscalgastoHasItensPK == null ? 0 :
notafiscalgastoHasItensPK.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof NotafiscalgastoHasItens))
            return false;
        NotafiscalgastoHasItens other = (NotafiscalgastoHasItens)object;
        return notafiscalgastoHasItensPK == other.notafiscalgastoHasItensPK ||
notafiscalgastoHasItensPK != null &&
notafiscalgastoHasItensPK.equals(other.notafiscalgastoHasItensPK);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.NotafiscalgastoHasItens[notafiscalgastoHasItensPK=
").append(notafiscalgastoHasItensPK).append("]").toString();
    }

    protected NotafiscalgastoHasItensPK notafiscalgastoHasItensPK;
    private Double quantidadegasto;
    private BigDecimal valortotalgasto;
    private Collection notaservicoCollection;
    private Itens itens;
    private Notafiscalgasto notafiscalgasto;
}
```

11.4.11 frotaControle/model/NotafiscalgastoHasItensPK

```
package frotaControle.model;

import java.io.Serializable;

public class NotafiscalgastoHasItensPK
    implements Serializable
{

    public NotafiscalgastoHasItensPK()
    {
```

```
    }

    public NotafiscalgastoHasItensPK(int itensIditens, int
notafiscalgastoIdnotafiscalgasto)
    {
        this.itensIditens = itensIditens;
        this.notafiscalgastoIdnotafiscalgasto = notafiscalgastoIdnotafiscalgasto;
    }

    public int getNotafiscalgastoIdnotafiscalgasto()
    {
        return notafiscalgastoIdnotafiscalgasto;
    }

    public void setNotafiscalgastoIdnotafiscalgasto(int
notafiscalgastoIdnotafiscalgasto)
    {
        this.notafiscalgastoIdnotafiscalgasto = notafiscalgastoIdnotafiscalgasto;
    }

    public int getItensIditens()
    {
        return itensIditens;
    }

    public void setItensIditens(int itensIditens)
    {
        this.itensIditens = itensIditens;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += itensIditens;
        hash += notafiscalgastoIdnotafiscalgasto;
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof NotafiscalgastoHasItensPK))
            return false;
        NotafiscalgastoHasItensPK other = (NotafiscalgastoHasItensPK)object;
        if(itensIditens != other.itensIditens)
            return false;
```

```
        return notafiscalgastoIdnotafiscalgasto ==
other.notafiscalgastoIdnotafiscalgasto;
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.NotafiscalgastoHasItensPK[itensIditens=").append(ite
nsIditens).append(",
notafiscalgastoIdnotafiscalgasto=").append(notafiscalgastoIdnotafiscalgasto).append
("]").toString();
    }

    private int notafiscalgastoIdnotafiscalgasto;
    private int itensIditens;
}
```

11.4.12 frotaControle/model/Notaservico

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Date;

// Referenced classes of package frotaControle.model:
//            Integrante, NotafiscalgastoHasItens

public class Notaservico
    implements Serializable
{

    public Notaservico()
    {
    }

    public Notaservico(Integer idnotaservico)
    {
        this.idnotaservico = idnotaservico;
    }

    public Notaservico(Integer idnotaservico, Date datanotaservico, Date
datapreenchenotaservico, boolean statusmanutencao, double quantidadeservico,
String naturezadoreparo)
```

```
{
    this.idnotaservico = idnotaservico;
    this.datanotaservico = datanotaservico;
    this.datapreenchenotaservico = datapreenchenotaservico;
    this.statusmanutencao = statusmanutencao;
    this.quantidadeservico = quantidadeservico;
    this.naturezadoreparo = naturezadoreparo;
}

public Integer getIdnotaservico()
{
    return idnotaservico;
}

public void setIdnotaservico(Integer idnotaservico)
{
    this.idnotaservico = idnotaservico;
}

public Date getDatanotaservico()
{
    return datanotaservico;
}

public void setDatanotaservico(Date datanotaservico)
{
    this.datanotaservico = datanotaservico;
}

public Date getDatapreenchenotaservico()
{
    return datapreenchenotaservico;
}

public void setDatapreenchenotaservico(Date datapreenchenotaservico)
{
    this.datapreenchenotaservico = datapreenchenotaservico;
}

public boolean getStatusmanutencao()
{
    return statusmanutencao;
}

public void setStatusmanutencao(boolean statusmanutencao)
{
```

```
        this.statusmanutencao = statusmanutencao;
    }

    public double getQuantidadeservico()
    {
        return quantidadeservico;
    }

    public void setQuantidadeservico(double quantidadeservico)
    {
        this.quantidadeservico = quantidadeservico;
    }

    public String getNaturezadoreparo()
    {
        return naturezadoreparo;
    }

    public void setNaturezadoreparo(String naturezadoreparo)
    {
        this.naturezadoreparo = naturezadoreparo;
    }

    public Integrante getIntegranteIdintegrante()
    {
        return integranteIdintegrante;
    }

    public void setIntegranteIdintegrante(Integrante integranteIdintegrante)
    {
        this.integranteIdintegrante = integranteIdintegrante;
    }

    public NotafiscalgastoHasItens getNotafiscalgastoHasItens()
    {
        return notafiscalgastoHasItens;
    }

    public void setNotafiscalgastoHasItens(NotafiscalgastoHasItens
notafiscalgastoHasItens)
    {
        this.notafiscalgastoHasItens = notafiscalgastoHasItens;
    }

    public int hashCode()
    {
```

```
        int hash = 0;
        hash += idnotaservico == null ? 0 : idnotaservico.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Notaservico))
            return false;
        Notaservico other = (Notaservico)object;
        return idnotaservico == other.idnotaservico || idnotaservico != null &&
idnotaservico.equals(other.idnotaservico);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Notaservico[idnotaservico=").append(idnotaservico).a
ppend("]").toString();
    }

    private Integer idnotaservico;
    private Date datanotaservico;
    private Date datapreenchenotaservico;
    private boolean statusmanutencao;
    private double quantidadeservico;
    private String naturezadoreparo;
    private Integrante integranteIdintegrante;
    private NotafiscalgastoHasItens notafiscalgastoHasItens;
}
```

11.4.13 frotaControle/model/Perfilusuario

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

public class Perfilusuario
    implements Serializable
{

    public Perfilusuario()
```

```
    {
    }

    public Perfilusuario(Integer idperfilusuario)
    {
        this.idperfilusuario = idperfilusuario;
    }

    public Perfilusuario(Integer idperfilusuario, String nome, String funcionalidade,
String descricao)
    {
        this.idperfilusuario = idperfilusuario;
        this.nome = nome;
        this.funcionalidade = funcionalidade;
        this.descricao = descricao;
    }

    public Integer getIdperfilusuario()
    {
        return idperfilusuario;
    }

    public void setIdperfilusuario(Integer idperfilusuario)
    {
        this.idperfilusuario = idperfilusuario;
    }

    public String getNome()
    {
        return nome;
    }

    public void setNome(String nome)
    {
        this.nome = nome;
    }

    public String getFuncionalidade()
    {
        return funcionalidade;
    }

    public void setFuncionalidade(String funcionalidade)
    {
        this.funcionalidade = funcionalidade;
    }
```

```
    public String getDescricao()
    {
        return descricao;
    }

    public void setDescricao(String descricao)
    {
        this.descricao = descricao;
    }

    public Collection getUsuariosistemaCollection()
    {
        return usuariosistemaCollection;
    }

    public void setUsuariosistemaCollection(Collection usuariosistemaCollection)
    {
        this.usuariosistemaCollection = usuariosistemaCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idperfilusuario == null ? 0 : idperfilusuario.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Perfilusuario))
            return false;
        Perfilusuario other = (Perfilusuario)object;
        return idperfilusuario == other.idperfilusuario || idperfilusuario != null &&
idperfilusuario.equals(other.idperfilusuario);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Perfilusuario[idperfilusuario=").append(idperfilusuario
).append("]").toString();
    }

    private Integer idperfilusuario;
    private String nome;
```

```
    private String funcionalidade;
    private String descricao;
    private Collection usuariosistemaCollection;
}
```

11.4.14 frotaControle/model/Subgrupointegrantes

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

// Referenced classes of package frotaControle.model:
//            Grupointegrante

public class Subgrupointegrantes
    implements Serializable
{

    public Subgrupointegrantes()
    {
    }

    public Subgrupointegrantes(Integer idsubgrupointegrantes)
    {
        this.idsubgrupointegrantes = idsubgrupointegrantes;
    }

    public Subgrupointegrantes(Integer idsubgrupointegrantes, String nomesubgrupo)
    {
        this.idsubgrupointegrantes = idsubgrupointegrantes;
        this.nomesubgrupo = nomesubgrupo;
    }

    public Integer getIdsubgrupointegrantes()
    {
        return idsubgrupointegrantes;
    }

    public void setIdsubgrupointegrantes(Integer idsubgrupointegrantes)
    {
        this.idsubgrupointegrantes = idsubgrupointegrantes;
    }
```

```
    public String getNomesubgrupo()
    {
        return nomesubgrupo;
    }

    public void setNomesubgrupo(String nomesubgrupo)
    {
        this.nomesubgrupo = nomesubgrupo;
    }

    public Grupointegrante getGrupointegranteIdgrupointegrante()
    {
        return grupointegrante_idgrupointegrante;
    }

    public void setGrupointegranteIdgrupointegrante(Grupointegrante
grupointegrante_idgrupointegrante)
    {
        this.grupointegrante_idgrupointegrante = grupointegrante_idgrupointegrante;
    }

    public Collection getIntegranteCollection()
    {
        return integranteCollection;
    }

    public void setIntegranteCollection(Collection integranteCollection)
    {
        this.integranteCollection = integranteCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idsubgrupointegrantes == null ? 0 : idsubgrupointegrantes.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Subgrupointegrantes))
            return false;
        Subgrupointegrantes other = (Subgrupointegrantes)object;
```

```
        return idsubgrupointegrantes == other.idsubgrupointegrantes ||
idsubgrupointegrantes != null &&
idsubgrupointegrantes.equals(other.idsubgrupointegrantes);
    }

    public String toString()
    {
        return (new
StringBuilder()).append("frotaControle.model.Subgrupointegrantes[idsubgrupointegra
ntes=").append(idsubgrupointegrantes).append("]").toString();
    }

    private Integer idsubgrupointegrantes;
    private static final long serialVersionUID = 1L;
    private String nomesubgrupo;
    private Grupointegrante grupointegrante_idgrupointegrante;
    private Collection integranteCollection;
}
```

11.4.15 frotaControle/model/Tipocombustivel

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

public class Tipocombustivel
    implements Serializable
{

    public Tipocombustivel()
    {
    }

    public Tipocombustivel(Integer idtipocombustivel)
    {
        this.idtipocombustivel = idtipocombustivel;
    }

    public Tipocombustivel(Integer idtipocombustivel, String nometipocombustivel)
    {
        this.idtipocombustivel = idtipocombustivel;
        this.nometipocombustivel = nometipocombustivel;
```

```
    }

    public Integer getIdtipocombustivel()
    {
        return idtipocombustivel;
    }

    public void setIdtipocombustivel(Integer idtipocombustivel)
    {
        this.idtipocombustivel = idtipocombustivel;
    }

    public String getNometipocombustivel()
    {
        return nometipocombustivel;
    }

    public void setNometipocombustivel(String nometipocombustivel)
    {
        this.nometipocombustivel = nometipocombustivel;
    }

    public Collection getIntegranteCollection()
    {
        return integranteCollection;
    }

    public void setIntegranteCollection(Collection integranteCollection)
    {
        this.integranteCollection = integranteCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idtipocombustivel == null ? 0 : idtipocombustivel.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Tipocombustivel))
            return false;
        Tipocombustivel other = (Tipocombustivel)object;
        return idtipocombustivel == other.idtipocombustivel || idtipocombustivel != null
&& idtipocombustivel.equals(other.idtipocombustivel);
```

```
    }

    public String toString()
    {
        return (new
StringBuilder()).append("model.Tipocombustivel[idtipocombustivel=").append(idtipoc
ombustivel).append("]").toString();
    }

    private Integer idtipocombustivel;
    private String nometipocombustivel;
    private Collection integranteCollection;
}
```

11.4.16 frotaControle/model/Unidade

```
package frotaControle.model;

import java.io.Serializable;
import java.util.Collection;

public class Unidade
    implements Serializable
{

    public Unidade()
    {
    }

    public Unidade(Integer idunidade)
    {
        this.idunidade = idunidade;
    }

    public Integer getIdunidade()
    {
        return idunidade;
    }

    public void setIdunidade(Integer idunidade)
    {
        this.idunidade = idunidade;
    }
```

```
    public String getNomeunidade()
    {
        return nomeunidade;
    }

    public void setNomeunidade(String nomeunidade)
    {
        this.nomeunidade = nomeunidade;
    }

    public String getTipounidade()
    {
        return tipounidade;
    }

    public void setTipounidade(String tipounidade)
    {
        this.tipounidade = tipounidade;
    }

    public Collection getItensCollection()
    {
        return itensCollection;
    }

    public void setItensCollection(Collection itensCollection)
    {
        this.itensCollection = itensCollection;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idunidade == null ? 0 : idunidade.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Unidade))
            return false;
        Unidade other = (Unidade)object;
        return idunidade == other.idunidade || idunidade != null &&
idunidade.equals(other.idunidade);
    }
```

```
    public String toString()
    {
        return (new
StringBuilder()).append("model.Unidade[idunidade=").append(idunidade).append("]")
.toString();
    }

    private Integer idunidade;
    private String nomeunidade;
    private String tipounidade;
    private Collection itensCollection;
}
```

11.4.17 frotaControle/model/Usuariosistema

```
package frotaControle.model;

import java.io.Serializable;

// Referenced classes of package frotaControle.model:
//            Perfilusuario

public class Usuariosistema
    implements Serializable
{

    public Usuariosistema()
    {
    }

    public Usuariosistema(Integer idusuariosistema)
    {
        this.idusuariosistema = idusuariosistema;
    }

    public Usuariosistema(Integer idusuariosistema, String nomeusuario, String
loginusuario, String senhausuario, int cpfusuario)
    {
        this.idusuariosistema = idusuariosistema;
        this.nomeusuario = nomeusuario;
        this.loginusuario = loginusuario;
        this.senhausuario = senhausuario;
```

```
    this.cpfusuario = cpfusuario;
}

public Integer getIdusuariosistema()
{
    return idusuariosistema;
}

public void setIdusuariosistema(Integer idusuariosistema)
{
    this.idusuariosistema = idusuariosistema;
}

public String getNomeusuario()
{
    return nomeusuario;
}

public void setNomeusuario(String nomeusuario)
{
    this.nomeusuario = nomeusuario;
}

public String getEnderecousuario()
{
    return enderecousuario;
}

public void setEnderecousuario(String enderecousuario)
{
    this.enderecousuario = enderecousuario;
}

public String getLoginusuario()
{
    return loginusuario;
}

public void setLoginusuario(String loginusuario)
{
    this.loginusuario = loginusuario;
}

public String getSenhausuario()
{
    return senhausuario;
```

```
    }

    public void setSenhausuario(String senhausuario)
    {
        this.senhausuario = senhausuario;
    }

    public int getCpfusuario()
    {
        return cpfusuario;
    }

    public void setCpfusuario(int cpfusuario)
    {
        this.cpfusuario = cpfusuario;
    }

    public Perfilusuario getPerfilusuarioIdperfilusuario()
    {
        return perfilusuarioIdperfilusuario;
    }

    public void setPerfilusuarioIdperfilusuario(Perfilusuario perfilusuarioIdperfilusuario)
    {
        this.perfilusuarioIdperfilusuario = perfilusuarioIdperfilusuario;
    }

    public int hashCode()
    {
        int hash = 0;
        hash += idusuariosistema == null ? 0 : idusuariosistema.hashCode();
        return hash;
    }

    public boolean equals(Object object)
    {
        if(!(object instanceof Usuariosistema))
            return false;
        Usuariosistema other = (Usuariosistema)object;
        return idusuariosistema == other.idusuariosistema || idusuariosistema != null &&
idusuariosistema.equals(other.idusuariosistema);
    }

    public String toString()
    {
```

```
        return (new
StringBuilder()).append("model.Usuariosistema[idusuariosistema=").append(idusuari
osistema).append("]").toString();
    }

    private Integer idusuariosistema;
    private String nomeusuario;
    private String enderecousuario;
    private String loginusuario;
    private String senhausuario;
    private int cpfusuario;
    private Perfilusuario perfilusuarioIdperfilusuario;
}
```

11.5 Classes de Internacionalização e Mensagens do Sistema

11.5.1 frotaControle/properties/internacional.properties

```
#novo item novo=Novo


#editar item editar=Editar


#excluir Item excluir=Excluir


#perfil usuario – nome nomeperfilusuario=Nome Perfil Usu\u00E1rio


#Funcionalidade – Perfil funcionalidade=Funcionalidade


#Descri\u00E7\u00E3o – item descricao=Descri\u00E7\u00E3o


#nome – perfil nome=nome
```

#Primeiro – tabela primeiro=Primeiro

#proximo – tabela proximo=Pr\u00F3ximo

#anterior – tabela anterior=Anterior

#ultimo Tabela ultimo=\u00DAltimo

#D\u00FAvidas - Perfil do Usuario duvidasperfildousuario=D\u00FAvidas - Perfil do Usuario

#Perfil do Usuario perfildousuario=Perfil do Usu\u00E1rio

#Salvar – tabela salvar=Salvar

#Nome do Usu\u00E1rio nomeusuario=Nome do Usu\u00E1rio

#endere\u00E7o endereco=Endere\u00E7o

#login usuario login=Login

#senha usuario senha=Senha

#cpf usuario cpf=N\u00BA.CPF

#D\u00FAvidas Usu\u00E1rio duvidasusuario=Ajuda

#usuario usuario=Usu\u00E1rio

#Data da Atividade datadaatividade=Data da In\u00EDcio

#Hora Inicio Atividade horainicioatividade=Hora In\u00EDcio

#dataterminoatividade dataterminoatividade=Data de T\u00E9rmino

#horaterminoAtividade horatermino=Hora T\u00E9rmino

#kminicial kminicial=Km Inicial

#observacao observacaoatividade=Observa\u00E7\u00E3o da Atividade

#kmfinal kmfinal=Km Final

#multa multa=Multa

#gravidademulta gravidademulta=Gravidade da multa

#obspercurso obspercurso=Observa\u00E7\u00E3o Percurso

#outrasobs outrasobs=Outras Observa\u00E7\u00F5es

#integrante integrante=Integrante

#motorista motorista=Motorista

#atividade atividade=Atividade

#Nomeintegrante nomeintegrante=Ve\u00EDculo

#placa placa=Placa

#marca marca=Marca

#kmatualiniciante kmatualiniciante=Km Atual

#categoriahabilitacaoIdcategoriahabilitacao categoriahabilitacao=Catgoria C.N.H.

#SubgrupointegrantesIdsubgrupointegrantes subgrupointegrantes=Tipo de Ve\u00EDculo

#tipocombustivel tipocombustivel=Tipo Combust\u00EDvel

#grupointegrantes grupointegrantes=Grupo de Ve\u00EDculos

#nomeunidade nomeunidade=Descricao Unidade

#tipo tipo=Tipo

#unidade unidade=Unidade

#ok okay=Ok

#cancelar cancelar=Cancelar

#nomemotorista nomemotorista=Nome do Motorista

#cpfmotorista cpfmotorista=CPF do Motorista

#matriculamotorista matriculamotorista=N\u00BA. Matricula

#validadecarteirahab validadecarteirahab=Validade da C.N.H.

#classificacaomotorista classificacaomotorista=Classifica\u00E7\u00E3o

#foto foto=Foto

#operadormaquinas operadormaquinas=Operador de M\u00E1quinas

#ambos ambos=Ambos

#razaosocial razaosocial=Raz\u00E3o Social

#cnpj cnpj=Cnpj

#telefone telefone=Telefone

#fax fax=Fax

#ddd ddd=DDD

#cidade cidade=Cidade

#estado estado=Estado

#cadastrofornecedor cadastrofornecedor=Cadastro Fornecedor

#valor valor=Valor

#itens itens=Item

11.5.2 frotaControle/properties/mensagens.properties

#Atualiza\u00E7\u00E3o realizada com sucesso

atualizacaoOK=Atualiza\u00E7\u00E3o realizada com sucesso

#Atualiza\u00E7\u00E3o n\u00E3o foi realizada - Erro\!\!

atualizacaoError=Atualiza\u00E7\u00E3o n\u00E3o foi realizada - Erro\!\!

#Exclus\u00E3o realizada com sucesso

exclusaoOK=Exclus\u00E3o realizada com sucesso

#N\u00E3o foi poss\u00EDvel realizar Exclus\u00E3o

exclusaorError=N\u00E3o foi poss\u00EDvel realizar Exclus\u00E3o