

UNIVERSIDADE FEDERAL DO PIAUÍ
CAMPUS SENADOR HELVÍDIO NUNES DE BARROS
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Marcelino Mendes da Silva Neto

SISTEMA *WEB* PARA CONTROLE DOS PROCESSOS DE ALOCAÇÃO DE
VEÍCULOS DO SETOR DE TRANSPORTE DA UNIVERSIDADE FEDERAL DO
PIAUÍ – *CAMPUS* SENADOR HELVÍDIO NUNES DE BARROS

PICOS

2015

MARCELINO MENDES DA SILVA NETO

**SISTEMA *WEB* PARA CONTROLE DOS PROCESSOS DE ALOCAÇÃO DE
VEÍCULOS DO SETOR DE TRANSPORTE DA UNIVERSIDADE FEDERAL DO
PIAUÍ – *CAMPUS* SENADOR HELVÍDIO NUNES DE BARROS**

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação do *Campus* Senador Helvídio Nunes de Barros da Universidade Federal do Piauí como parte dos requisitos para obtenção do Grau de Bacharel em Sistemas de Informação, sob orientação do Professor Ivenilton Alexandre de Souza Moura.

PICOS

2015

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

S586s Silva Neto, Marcelino Mendes da.
Sistema Web para controle dos processos de alocação de
veículos do setor de transporte da Universidade Federal do Piauí
– Campus Senador Helvídio Nunes de Barros / Marcelino
Mendes da Silva Neto. - 2014.
CD-ROM : il. ; 4 ¼ pol. (64 p.)

Monografia(Bacharelado em Sistemas de Informação) –
Universidade Federal do Piauí. Picos-PI, 2014.
Orientador(A): Prof. Esp. Ivenilton Alexandre de Sousa Moura

1. Sistema de Informação. 2. Ruby on Rails. 3. Engenharia
do Software. I. Título.

CDD 005.1

MARCELINO MENDES DA SILVA NETO

**SISTEMA *WEB* PARA CONTROLE DOS PROCESSOS DE ALOCAÇÃO DE
VEÍCULOS DO SETOR DE TRANSPORTE DA UNIVERSIDADE FEDERAL DO
PIAUI – *CAMPUS* SENADOR HELVÍDIO NUNES DE BARROS**

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação do *Campus* Senador Helvídio Nunes de Barros da Universidade Federal do Piauí como parte dos requisitos para obtenção do Grau de Bacharel em Sistemas de Informação, sob orientação do Professor Ivenilton Alexandre de Souza Moura.

Data de Aprovação:

06/01/2015

Ivenilton Alexandre de Souza Moura

UFPI

Alcilene Dalília de Sousa

UFPI

Allan Jheyson Ramos Gonçalves

UFPI

PICOS - PI

2015

Dedico este trabalho em primeiro lugar a Deus, que me trouxe até aqui com toda sua bondade e misericórdia. Em segundo lugar aos meus pais e meus avós, que sempre me amaram e me apoiaram nos momentos mais difíceis. Toda a minha família que sempre esteve comigo em todos os momentos. A vocês eu devo tudo isso.

AGRADECIMENTOS

Agradeço primeiramente a Deus. Agradeço à minha família por sempre acreditar e investir em mim. A minha mãe, por estar sempre do meu lado, cuidando e se esforçando para me dar esperança para seguir em frente. Ao meu pai, por mostrar segurança e certeza de que sempre que eu precisar ele estará ao meu lado.

“Tenho a impressão de ter sido uma criança brincando à beira-mar, divertindo-me em descobrir uma pedrinha mais lisa ou uma cocha mais bonita que as outras, enquanto o imenso oceano da verdade continua misterioso diante de meus olhos.”

Issac Newton

RESUMO

O presente trabalho demonstra o desenvolvimento de um Sistema de Informação Gerencial para a Universidade Federal do Piauí - *Campus* Senador Helvídio Nunes de Barros, devido à percepção da dificuldade encontrada no gerenciamento dos processos de alguns setores do *campus*. O objetivo é o desenvolvimento de um sistema *web* para o setor de transporte afim de automatizar os processos e o gerenciamento de recursos do setor referido, resolvendo assim as deficiências encontradas através de entrevistas realizadas com o chefe de setor. Com isso, foram utilizadas boas práticas de desenvolvimento de sistemas, sendo apresentada a modelagem do sistema utilizada para o desenvolvimento do projeto. Os resultados foram alcançados através da utilização de conceitos da engenharia de *software*, desde a fase de levantamento de requisitos até a fase de implantação. Para o desenvolvimento do projeto foi utilizada a linguagem de programação *Ruby*, e a *framework Rails*, também chamando de *Ruby on Rails*.

Palavras-chave: Sistema de Informação. *Ruby on Rails*. Engenharia de *Software*. Alocação de Transporte.

ABSTRACT

This work demonstrates the development of a Management Information System for the Universidade Federal do Piauí - Campus Senator Helvídios Nunes de Barros, because of perceived the difficulty in managing the processes of some sectors of the campus. The objective is to develop a web system for the transport sector in order to automate processes and resource management of this sector, thereby addressing the deficiencies found through interviews with the head of sector. With that, we used good systems development practices, and presented the modeling of the system used for the development of the project. The results were achieved through the use of software engineering concepts, since the requirements elicitation phase to the implementation phase. To develop the project we used the Ruby programming language and the Rails framework, also called Ruby on Rails.

Keywords: *Information System. Ruby on Rail. Software Engineering. Transportation allocation.*

LISTA DE FIGURAS

Figura 1- Diagrama de Caso de Uso (Usuários Solicitantes)	32
Figura 2- Diagrama de Caso de Uso (Usuário Administrador)	33
Figura 3- Representação da classe Usuário	33
Figura 4- Representação de associação entre classes	34
Figura 5- Diagrama de Classes	35
Figura 6 - Diagrama de Atividade (autenticação no sistema)	37
Figura 7 – Diagrama de Atividade (Processo de Solicitação)	39
Figura 8 - Diagrama de Atividade (Análise de Solicitação).....	41
Figura 9 – Diagrama de Atividade (Agrupamento de Solicitações).....	42
Figura 10 - Diagrama de Atividade (Gerar relatórios)	44
Figura 11 – MER	46
Figura 12- Tela de Login	47
Figura 13- Tela de Login (Administrador)	47
Figura 14- Tela de Solicitação de Transporte.....	48
Figura 15- Aprovação de Solicitação	48
Figura 16- Desaprovação de Solicitação	49
Figura 17- Justificativa de Desaprovação.....	49
Figura 18- Agrupamento de Solicitações	50
Figura 19- Geração de Relatórios	50

LISTA DE QUADROS

Quadro 1 - Requisitos Funcionais	29
Quadro 2 - Requisitos Não Funcionais.....	30
Quadro 3 - Notação de Relacionamento.....	34
Quadro 4 - Diagrama de atividades (Autenticação do Usuário) – Fluxo principal	38
Quadro 5 – Diagrama de atividades(Autenticação do Usuário) – Fluxo Alternativo	38
Quadro 6 - Diagrama de Atividade (Processo de Solicitação)	39
Quadro 7 - Diagrama de Atividade (Processo de Solicitação)	40
Quadro 8 – Diagrama de Atividade (Análise de Solicitação) – Fluxo Principal.....	41
Quadro 9 - Diagrama de Atividade (Análise de Solicitação) – Alternativo	42
Quadro 10 - Diagrama de Atividade (Agrupamento de Solicitações) – Fluxo Principal	43
Quadro 11 - Diagrama de Atividade(Agrupamento de Solicitações) – Fluxo Alternativo	43
Quadro 12 – Diagrama de Atividade (Gerar relatórios) – Fluxo Principal	44
Quadro 13 – Diagrama de Atividade (Gerar relatórios) – Fluxo Alternativo 1	45
Quadro 14 – Diagrama de Atividade (Gerar relatórios) – Fluxo Alternativo 2	45
Quadro 15 - Dicionário de dados – Cargo	59
Quadro 16 - Dicionário de dados – Confirmação.....	59
Quadro 17 – Dicionário de dados – Despesas	60
Quadro 18 - Dicionário de dados – Estado_vigencia	60
Quadro 19 - Dicionário de dados – Função	60
Quadro 20 – Dicionário de dados – Motorista	60
Quadro 21 – Dicionário de dados – Participantes	61
Quadro 22 - Dicionário de dados – Repasse.....	61
Quadro 23 - Dicionário de dados – Setor	62
Quadro 24 – Dicionário de dados – Solicitação	62
Quadro 25 - Dicionário de dados – Tipo_Usuário	62
Quadro 26 - Dicionário de dados – Usuário	63
Quadro 27 – Dicionário de dados – Veículo	63
Quadro 28 – Dicionário de dados – Viagem	64
Quadro 29 – Dicionário de dados – Viagem_has_motorista.....	64
Quadro 30 – Dicionário de dados – Viagem_has_solicitação.....	64
Quadro 31 – Dicionário de dados – Viagem_has_veiculo	64
Quadro 32 – Dicionário de dados – Vigência.....	65

LISTA DE ABREVIATURAS E SIGLAS

CAF	Coordenador Administrativo Financeiro
CRM	<i>Customer Relationship Management</i>
CSHNB	Campus Senador Helvídio Nunes de Barros
DRY	<i>Don't repeat yourself</i>
DW/DM	<i>Data Warehouse/Data mining</i>
MVC	<i>Model-View-Controller</i>
RF	Requisitos Funcionais
RNF	Requisitos Não Funcionais
SAD	Sistema de Apoio à Decisão
SGE	Sistema de Gestão Empresarial
SIG	Sistemas de Informação Gerencial
UFPI	Universidade Federal do Piauí
UML	<i>Unified Modeling Language</i>
XP	<i>eXtreme Programming</i>

SUMÁRIO

1	INTRODUÇÃO.....	14
1.1	Motivação	14
1.2	Objetivo.....	14
1.3	Organização do Documento.....	15
2	REFERENCIAL TEÓRICO	16
2.1	Sistemas de Informação.....	16
2.1.1	Sistemas	16
2.1.2	Sistemas de Informação	16
2.2	Processo decisório	19
2.3	Sistemas <i>WEB</i>.....	20
2.4	Linguagem de Programação <i>Ruby</i>	20
2.5	<i>Ruby on Rails</i>	21
2.6	Metodologias de Desenvolvimento	23
2.7	<i>Scrum</i>.....	25
2.8	Requisitos de <i>Software</i>	26
2.9	UML	27
3	O DESENVOLVIMENTO DO SISTEMA.....	28
3.1	Análise e Especificação de Requisitos	28
3.2	Requisitos Funcionais	29
3.2.1	Requisitos Não Funcionais	30
3.3	Análise e <i>Design</i>	31
3.3.1	Diagramas de Caso de Uso	32
3.3.2	Diagrama de Classe	33
3.3.3	Diagrama de Atividades.....	37
3.4	Modelagem de Dados.....	46
3.4.1	Modelo de Entidade-Relacionamento (MER)	46
3.5	Principais telas do sistema	46
	CONCLUSÕES E TRABALHOS FUTUROS.....	51
	REFERÊNCIAS.....	52
	APÊNDICE	55
	APÊNDICE A – Entrevistas	56

APÊNDICE B - Descrição E Conteúdo Das Tabelas Do Banco De Dados.....59

1 INTRODUÇÃO

1.1 Motivação

Na atualidade o mundo vive na era da informação, exigindo das organizações uma gestão estratégica eficiente, a qual pode ser facilitada pela utilização de recursos inteligentes oferecidos por sistemas de informações.

Sistemas de informação é um conjunto de processos de transformação de dados em informação visando a sua utilização na estrutura decisória da empresa para proporcionar uma sustentabilidade administrativa, podendo assim aperfeiçoar os resultados esperados pela organização. (REZENDE E ABREU, 2006).

Partindo dessa ideia de sistema de informação, o papel não é somente coletar e armazenar dados, mas também de organizá-los de maneira que palavras e números possam ser inseridos num contexto, tornando-os informações úteis para a tomada de decisão de uma empresa no momento que for necessário.

Na Universidade Federal do Piauí (UFPI), *campus* Senador Helvídio Nunes de Barros - CSHNB, especificamente no setor de transporte, todos os processos de alocação de veículos e gerenciamento de recursos são realizados através do uso de formulários impressos sem controle eletrônico, o que torna esses processos complexos e desgastantes.

Após análise das necessidades e problemas diagnosticados na organização, houve consenso quanto à aprovação da proposta para desenvolvimento de um sistema *web* de automatização para oferecer suporte às atividades operacionais e de gestão do setor de transporte. O sistema automatizado foi desenvolvido em módulos, integrado a outros dois sistemas, de alocação de espaços físicos e o sistema de controle de diárias, compondo assim um sistema único que será utilizado por todos os setores do *campus*.

1.2 Objetivo

O objetivo desse trabalho foi desenvolver um sistema de informação gerencial (SIG) *web* para Universidade Federal do Piauí - *Campus* Senador Helvídio Nunes de Barros para automatizar os processos de solicitação de veículos e gerenciamento de informações do mesmo.

1.3 Organização do Documento

Após as sessões 1.1 e 1.2 que correspondem à motivação, onde é relatado o problema a ser resolvido e o objetivo do trabalho. As etapas de desenvolvimento do projeto serão melhor descritos nos capítulos seguintes, sendo organizados em:

- Capítulo 2 – corresponde ao embasamento teórico necessário para o desenvolvimento do projeto, contendo estudos e trabalhos realizados por outros autores e estudiosos das áreas que esse trabalho engloba.
- Capítulo 3 – tem como objetivo concentrar e organizar os requisitos identificados para o Sistema de Alocação de Veículos da Universidade Federal do Piauí, *campus* Picos, fornecendo as informações necessárias para o desenvolvimento, bem como para a realização dos testes do sistema. E também a modelagem visual através dos diagramas da *Unified Modeling Language* – UML.
- Capítulo 4 – apresenta a conclusão do sistema e sugestões de trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 Sistemas de Informação

2.1.1 Sistemas

De acordo com Oliveira (2002) e Batista (2004), sistema é um conjunto de partes interligadas e interdependentes que, conjuntamente, formam um todo unitário com a finalidade de executar uma ou mais atividades ou, ainda, um conjunto de eventos para realizar tarefas predefinidas.

Segundo Rezende e Abreu (2006), sistemas em geral procuram atuar como:

- Ferramentas para exercer o funcionamento das empresas e de sua intrincada abrangência e complexidade;
- Instrumentos que possibilitam uma avaliação analítica e, quando necessária, sintética das empresas;
- Facilitadores dos processos internos e externos com suas respectivas intensidades e relações;
- Meios para suportar a qualidade, produtividade e inovação tecnológica organizacional;
- Geradores de modelos de informações para auxiliar os processos decisórios empresariais;
- Produtores de informações oportunas e geradores de conhecimento;
- Valores agregados e complementares à modernidade, perenidade, lucratividade e competitividade empresarial.

Sistemas podem ser constituídos de partes unitárias e independentes, cada uma com finalidade de executar uma ou mais atividades, para solucionar uma tarefa específica. Sistema de informação é um tipo de sistema, normalmente automatizado, com a finalidade de manipular dados e gerar informação.

2.1.2 Sistemas de Informação

Segundo Mosimann e Fisch (2009), sistemas de informação é uma rede de informações cujos fluxos alimentam o processo de tomada de decisões; não apenas da empresa como um todo, mas também de cada área de responsabilidade. O conjunto de recursos humanos, físicos e tecnológicos que o compõe, transforma os dados captados em

informações, com a observância dos limites impostos pelo usuário quanto ao tipo de informação necessária a suas decisões, condicionando, portanto, a relação dos dados de entrada.

O'brien (2010) incrementa seu conceito dizendo que um sistema de informação é um sistema que recebe recursos de dados como entrada e os processa em produtos de informação com saída e que esse sistema depende dos recursos humanos (usuários finais e especialistas em TI), *hardware* (máquinas), *software* (programas), dados (banco de dados e base de conhecimento) e redes (meios de comunicações e apoio de redes) para executar atividades de entrada, processamento, produção, armazenamento e controle que convertem recursos de dados em informações, estes são os cinco recursos básicos de um sistema de informação.

Como pode ser visto, sistema de informação é um sistema que tem como entrada um fluxo de dados, que serão processados e que proporcionam como saída, as informações. Essas são úteis e necessárias e tem como finalidade suportar as atividades da organização, agregando valor ao produto, aumentando assim o poder competitivo da empresa.

De acordo com Pereira e Fonseca (1997), os sistemas de informação para serem efetivos precisam corresponder as seguintes expectativas:

- Atender as reais necessidades dos usuários;
- Estar centrados no usuário (cliente) e não no profissional que o criou;
- Atender ao usuário com presteza;
- Apresentar custos compatíveis;
- Adaptar-se constantemente às novas tecnologias de informação;
- Estar alinhados com as estratégias de negócios da empresa.

Segundo Faoro *et al.* (2010), os sistemas de informação são classificados em:

1. Sistema de Apoio à Decisão (SAD) - foco no suporte às decisões através de simulações, com a utilização modelos, que normalmente dão suporte às decisões de problemas semiestruturados e não-estruturados. Segundo Turban, McLean e Wetherbe(1996), as decisões para problemas semiestruturados, envolvem a combinação de soluções e procedimentos que não mudam e são baseadas nas experiências individuais. Já os não estruturados, são processos vagos e problemas complexos, onde a intuição humana é frequentemente utilizada para tomar tais decisões.
2. Sistema de Gestão Empresarial (ERP ou SGE) - foco na integração das informações em uma organização. Os sistemas ERP têm a finalidade de administrar partes

importantes da empresa, tais como o planejamento do produto, compra de componentes, manutenção de estoques, interação com fornecedores, entre outros. Fornecendo assim, informações importantes para os negócios *on-line* e o intercâmbio automático;

3. *Data Warehouse/Data mining* (DW/DM) - foco na exploração dos dados gerados pela empresa. Que segundo Harrison (1998), *data mining* “é a exploração e análise, por meio automático ou semiautomático, das grandes quantidades de dados para descobrir padrões e regras significativas.”. Este permite a empresa ter uma melhor compreensão dos clientes, um aumento na proporção das vendas, e apoio ao mesmo;
4. *Customer Relationship Management* (CRM) - foco no relacionamento com o cliente, de forma individual. Esse sistema tem como objetivo guardar informações sobre o cliente e principais preferências, para melhorar o atendimento, superando assim suas expectativas iniciais. Por isso, para Peppers (2000), é necessário que siga algumas dicas antes da implantação de um CRM, sendo elas:
 - Real conhecimento do cliente;
 - Saber o que ele deseja;
 - Fabricar exatamente o que ele deseja e entregar no prazo combinado;
 - E ter certeza da qualidade do seu serviço ou produto, pensando em seguida, em como torna-lo mais personalizado.
5. Sistema de Informação Gerencial (SIG) - foco em fornecer informações associadas aos subsistemas funcionais para a tomada de decisões, ou seja, sistemas que fornecem relatórios. Segundo Oliveira (2002), o SIG é um processo de transformação de dados em informações que são utilizadas na estrutura decisória da empresa, proporcionando, ainda, a sustentação administrativa para aperfeiçoar os resultados esperados. Os relatórios SIG podem ser classificados em:
 - Programados: são os relatórios gerados periodicamente, onde as datas para a emissão de relatórios são definidas pelo usuário;
 - Relatórios indicados por pontos críticos: é um tipo especial de relatório programado, emitido no começo de cada dia, resumindo as atividades do dia anterior;
 - Sob solicitação: são produzidos quando o usuário deseja informação a respeito de um item específico;

- De exceção: são parametrizados para informar automaticamente critérios preestabelecidos pela empresa, exemplo: quando o estoque de algum produto está abaixo do ideal.

Possuir um sistema bem estruturado, que proporcione informações necessárias para a tomada de decisão é o diferencial das organizações no mercado competitivo, pois as informações geradas pelos recursos que o compõem são os bens mais preciosos de qualquer empresa.

2.2 Processo decisório

Segundo Oliveira (2010), processo decisório é a conversão das informações analisadas em ação. Os desafios impostos levam os administradores a buscar informações que espelhem fielmente a real situação das organizações, para que o processo decisório seja efetuado de forma eficaz, alcançando assim os resultados pretendidos.

Oliveira (2010) afirma que, no processo decisório, deve ser estabelecida uma orientação em relação à opção escolhida, requerendo dos administradores uma racionalidade objetiva, o processo decisório é dividido nas seguintes fases:

- Identificação do problema, ou seja, identificar o cenário atual da empresa;
- Análise do problema, iniciando assim que forem reunidas as informações sobre o problema, identificando assim oportunidades e ameaças;
- Propor possíveis soluções e alternativas para a resolução do problema;
- Analisar e comparar as soluções que foram levantadas, considerando as vantagens e desvantagens de cada solução;
- Selecionar a opção mais viável por meio do conhecimento das vantagens e desvantagens dessas alternativas;
- Implantação da opção escolhida, com o treinamento necessário das pessoas envolvidas na resolução do problema;
- Avaliar a opção escolhida por meio dos resultados obtidos;

A análise das informações deve ser feita em todos os momentos, para ser possível identificar problemas, e a partir de então, propor uma ou mais soluções. Com o uso de sistemas *web* o responsável pela tomada de decisões não ficará restrito ao seu local de trabalho, pois ele poderá analisar as informações em qualquer lugar que permita acesso à internet.

2.3 Sistemas WEB

A Internet é uma rede que conecta computadores do mundo inteiro. Teve sua origem na década de 1960 (na época, Arpanet), com o objetivo de o governo americano trocar informações de uma base militar para outra. Com o passar do tempo, foi utilizada para comunicação nas universidades e institutos de pesquisas.

De acordo com Kessler (2002), entre a década de 80 e o início dos anos 90, a rede foi aperfeiçoada e começaram a aparecer os serviços que deram à *Internet* sua forma atual. O principal deles é a *World Wide Web* (WWW), lançado em 1991, que viabilizou a transmissão de imagens, som e vídeo pela rede, pois, até então, a maior parte do material disponível era composta de documentos hipertexto.

Segundo Rodrigues (2010), o desenvolvimento *web* é o termo utilizado para descrever o desenvolvimento de *sites*, na *internet* ou em uma *intranet*, a rede interna das empresas, estando normalmente associado à programação, à marcação e também pode ser usado para se referir ao projeto visual das páginas. Na década de 90 foi uma das indústrias de maior crescimento no mundo, pois trouxe novos hábitos de consumo e criou inovadoras formas de negócio.

Uma página *web* torna o acesso aos serviços disponíveis na *Internet* totalmente transparentes para o usuário, tornando possível a manipulação de uma quantidade enorme de informações, em qualquer lugar, disponível 24 horas e sete dias por semana.

Lobo (2007) afirma que o interesse dos programadores pelo desenvolvimento para Internet já atinge um nível muito alto desde que o uso da *web* se popularizou. Isso ocorre, em grande parte, pela complexidade desses sistemas e pelo envolvimento cada vez maior de tecnologias variadas. Entre essas tecnologias, estão as linguagens de programação para *web*, por exemplo, *php*, *javascript*, *python* e *Ruby*. Podendo ser integradas em *frameworks*, como *Django* (*python*) e *Rails* (*ruby*).

2.4 Linguagem de Programação Ruby

Ruby é uma linguagem de programação interpretada multiparadigma de tipagem dinâmica e forte, com gerenciamento de memória automática, originalmente projetada e desenvolvida no Japão em 1995, por Yukihiro “Matz” Matsumoto, para ser usada com linguagem de *script*. Matz queria uma linguagem de *script* que fosse mais poderosa que o

perl e ainda mais orientada a objeto do que o *python*. Foi inspirada principalmente pelas linguagens de programação *python*, *perl*, *smalltalk*, *eiffel*, *ada* e *lisp*, sendo muito similar em vários aspectos a *python*.

Ruby possui algumas características que a torna muito poderosa e flexível, como o fato de ser multiplataforma e multiparadigma, possuindo programação funcional, orientada a objeto, imperativa e reflexiva, tais características, torna o *Ruby* uma linguagem extremamente funcional, por poder ajudar o programador em diversas situações. E pelas características apresentadas anteriormente, possibilitam que os códigos *Ruby* sejam mais curtos, por não ser uma linguagem burocrática como o Java e C, e fáceis de ser compreendidos.

Segundo Souza (2014), os principais fatores que levaram ao *Ruby* a ser uma das linguagens mais usadas é pelo fato da curva de aprendizagem da linguagem ser muito alta, e por causa da disseminação de sua principal *framework* MVC, o *Ruby on Rails*.

2.5 *Ruby on Rails*

Rails é um *framework*, conjunto de classes que incorpora um projeto abstrato de soluções para um a família de problemas relacionados, de desenvolvimento *web* gratuito e de código aberto, escrito na linguagem *Ruby*. Surgiu dentro da empresa 37signals e foi criado por David Heinemeier Hansson que resolveu extrair parte da lógica *web* de um projeto chamado Basecamp¹, sendo lançado a público pela primeira vez em 2003.

Segue o padrão MVC (*MODEL-VIEW-CONTROLLER*) e o *Converntion over configuration*, ou seja, que acaba com a enorme quantidade de XMLs que existem na maioria dos sites existentes no mercado, dificultando a manutenção.

De acordo com a RailsGuides (2014), a Filosofia *Rails* e seus princípios orientadores são:

- DRY - “*Don't repeat yourself*” - sugere que escrever o mesmo código várias vezes é algo ruim.
- *Converntion over configuration* - significa que o Rails faz suposições sobre o que você quer fazer, e como você está indo para fazê-lo, em vez de exigir que você especifique cada pequena coisa através de arquivos de configuração sem fim.
- *REST* – é o melhor padrão para aplicações web – organiza a sua aplicação em torno de recursos e verbos HTTP, ações que podem ser executadas em um recurso identificado

¹ Projeto Basecamp pode ser visto no *site* <http://basecamp.com>

por uma URL, sua implementação fica a critério do servidor, o *Rails* usa os seguintes verbos:

- GET - para retornar um recurso específico ou uma coleção;
- POST - para enviar um novo elemento à uma coleção;
- PUT - para alterar um elemento existente;
- DELETE - para remover um elemento.

Segundo Fluentes (2012), o *Ruby on Rails* é um *framework* que adota o padrão de arquitetura chamado *Model-View-Controller* (MVC), ou Modelo, Apresentação e Controle.

Modelos (*Models*): possuem duas responsabilidades: eles são os dados que normalmente ficam persistidos em um ou mais banco de dados (seu perfil de usuário, por exemplo). Eles também fazem parte da regra de negócio, ou seja, cálculos e outros procedimentos, como verificar se uma senha é válida.

Controle (*Controller*) é a camada intermediária entre a *web* e o seu sistema. Ele pega os dados que vem de parâmetros na URL e/ou de um formulário e repassa para os modelos, que vão fazer o trabalho pesado. Em seguida, pega o resultado e transforma da maneira adequada para a Apresentação.

Apresentação (*View*) é como o aplicativo mostra o resultado das operações e os dados. Normalmente podem ser uma bela página usando as novas tecnologias como as linguagens de folha de estilo CSS (*Cascading Style Sheets*)³ e a linguagem de marcação de texto HTML (*Hyper Text Markup Language*)⁵ a até pequenas representações de objetos em JSON (*JavaScript Object Notation*), um formato mais leve para troca de dados computacionais.

O *Rails* é uma meta-framework, ou seja, uma *framework de framework*, segundo a RailsGuides (2014) é composta por:

- *Active Record*: é a base para os *models* em uma aplicação *Rails*. Ele fornece independência entre o banco de dados e aplicação, responsável pela abstração dos dados.
- *Action Pack*: é uma única *gem*, uma biblioteca, um conjunto de arquivos *Ruby* reusáveis, etiquetada com um nome e versão, que contém o *action controller*, *action view* e *action dispatch*, o “VC” do “MVC”.
- *Action Controller* é o componente que gerencia os *controlles* de uma aplicação. Ele processa as solicitações de entrada para uma aplicação *Rails*, extrai parâmetros, e os

despacha para a ação pretendida. Serviços prestados pelo *action controller* incluem gerenciamento da sessão, renderizações de templates.

- *Action View* é o componente que gerencia as *view* da aplicação. Ele cria os HTMLs e XMLs de saída padronizados.
- *Action Dispatch* lida com o encaminhamento de solicitações *web* e despacha-os como quiser, seja para sua aplicação ou outra qualquer.
- *Action Model*: fornece uma interface entre os serviços do *Action pack* e as ferramentas de mapeamento de objetos, como o *active record*. Permite utilizar outras frameworks no lugar do *action record*, caso sua aplicação precise.
- *Action mailer*: é um *framework* para a criação de serviços de e-mail. No qual pode ser utilizado para receber e processar e-mails recebidos, assim como enviar.

Mesmo com o uso da arquitetura MVC, facilitando a localização e organização do código e de erros, é importante e necessário o uso de metodologias de desenvolvimento. Estas últimas podem dar suporte na obtenção de produtos de software com mais robustos de maneira econômica.

2.6 Metodologias de Desenvolvimento

O desenvolvimento de *Software*, na década de 70, ficou conhecido pela Crise do Software (PRESSMAN, 2006), pois nesse período a produção de Software era feita de forma desorganizada, desestruturada, sem planejamento, sem produção de documentação e a análise do projeto não utilizava métodos no seu desenvolvimento. Com isso, prazo e custo não correspondiam a real necessidade.

“A Engenharia de *Software* utiliza o Processo de Desenvolvimento, que consiste na criação de documentos, artefatos e marcos, capazes de representar o contexto do *software*, levando em consideração recursos, ferramentas, prazos, restrições, e outros aspectos que envolvem o desenvolvimento de um produto, para no final produzir software de qualidade.” (PRESSMAN, 2006).

Ainda no contexto de engenharia de *software*, temos as metodologias tradicionais e as ágeis. Segundo Pressman (2006), as metodologias tradicionais surgiram em um cenário de construção de *software* muito diferente do atual. A sequência de tarefas para desenvolvimento era realizada em terminais burros e *mainframes*, onde o custo para correção de erros ou qualquer outra alteração era muito elevado. O acesso aos computadores era muito limitado e

não haviam ferramentas de apoio à construção de *software*, com isso para minimizar os problemas durante o desenvolvimento, necessitava-se planejar e projetar antes. Esse foi um dos fatores primordiais para a existência de um processo de gerenciamento de *software*, baseado em documentação detalhada, planejamento extenso e etapas bem delimitadas.

Exemplos: Cascata, espiral:

- **Cascata:** Conforme Pressman (2006), o modelo em cascata sugere uma abordagem linear e sequencial de todas as atividades envolvidas no desenvolvimento, e por se tratar de uma sequência de eventos, esse modelo também ficou conhecido como “ciclo de vida”.
- **Espiral:** Segundo Pádua (2000), no modelo espiral o produto é desenvolvido em uma série de iterações. Cada nova iteração corresponde a uma volta na espiral. Isto permite construir produtos em prazos curtos, com novas características e recursos que são agregados na medida em que a experiência descobre sua necessidade.

Metodologias ágeis de gerenciamento de projetos apareceram com contraposto às chamadas metodologias pesadas – metodologias tradicionalistas – como uma alternativa para se adaptar ao contexto do mercado atual, que exige resultados cada vez mais rápidos para atender às necessidades dos clientes, sob condições de constantes mudanças e incertezas. Pesquisas realizadas na Pontifícia Universidade Católica do Rio Grande do Sul em 2005 demonstram que somente 16,2% dos quase 8380 projetos usados por base da pesquisa, foram entregues dentro dos prazos, custos e com todas as funcionalidades especificadas (SOUZA, 2008).

De acordo com Cockburn (2002), os métodos ágeis são adaptativos e não previsíveis, como é o caso das metodologias tradicionais. Eles se adaptam aos novos fatores que surgem durante o desenvolvimento do projeto ao invés de verificar antecipadamente todos os impedimentos que podem acontecer durante a construção do produto. O problema em si não está nas mudanças, pois estas estão sempre sujeitas a ocorrer em quaisquer dos âmbitos, o problema está na maneira que cada uma das metodologias trata essas mudanças. Como recebem, avaliam e respondem às mesmas. Exemplos: *Scrum*, *eXtreme Programming (XP)*, *Cristal*:

- **Cristal:** Segundo Pressman (2006), a metodologia cristal foi criada por Alistair Cockburn e Jim Highsmith, visando conseguir elaborar uma abordagem de desenvolvimento que priorizasse a adaptabilidade durante o que Cockburn (2002) caracteriza como um jogo de comunicação cooperativa e com recursos limitados,

tendo como primeiro objetivo entregar *software* útil em funcionamento e como segundo preparar-se para o jogo seguinte.

- *eXtreme Programming* (XP): Segundo Teles (2004), a XP é um processo de desenvolvimento de *software* apropriado para os seguintes projetos: requisitos vagos que e mudam com frequência; Desenvolvimento de sistemas orientados a objeto; Equipes pequenas; e desenvolvimento incremental.
- *Scrum*: Schwaber (2002) define o Scrum como “um processo Ágil ou ainda um *framework* para gerenciamento de projetos ágeis”.

2.7 *Scrum*

A metodologia *Scrum* foi desenvolvida por Ken Schwaber e Jeff Sutherland. Nasceu da necessidade de encontrar uma metodologia que abordasse o problema do desenvolvimento de *software* de uma forma não tradicional, como num jogo de *Rugby*, a equipe age como um todo para atingir os seus objetivos (“*scrum*” denomina, no *Rugby*, uma equipa aglomerada, em que toda a gente age em conjunto para transportar a bola pelo campo).

Scrum é um *framework* estrutural que está sendo usado para gerenciar o desenvolvimento de produtos complexos desde o início de 1990, de forma que as pessoas possam tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregando produtos com o mais alto valor possível.

De acordo com Schwaber e Sutherland (2009), *scrum* não é um processo ou técnica para construir produtos, em vez disso é um *framework*, na qual podem ser empregados vários processos e técnicas, deixando a eficácia relativa das práticas de gerenciamento e desenvolvimento de produtos, possibilitando melhorá-lo. O *scrum* é fundamentado nas teorias empíricas de controle de processos, ou empirismo. O empirismo afirma que o conhecimento vem da experiência e da tomada de decisões baseadas no que é conhecido.

Segundo Ferreira (2005), as principais características do *scrum* são:

- É um processo ágil para gerenciar e controlar o desenvolvimento de projetos;
- É um processo que controla o caos resultante de necessidades e interesses conflitantes;
- É uma forma de aumentar a comunicação e maximizar a cooperação;
- É uma forma de detectar e remover qualquer impedimento que atrapalhe o desenvolvimento de um produto;
- É escalável desde projetos pequenos até grandes projetos em toda empresa.

2.8 Requisitos de *Software*

Zave (1997) define Engenharia de Requisitos como:

“[...] é o ramo da engenharia de software que está preocupado com os objetivos do mundo real para as funções e restrições aplicáveis a sistemas de software. Está também preocupado com o relacionamento destes fatores para especificações precisas do comportamento do software e com sua evolução no tempo e através de famílias de produtos.”

Segundo Pressman (2006) requisito de *software* é uma condição ou uma capacidade com o qual o sistema deve estar de acordo, expressando as necessidades do cliente. Os requisitos de software podem ser divididos em:

- Os requisitos funcionais (RF) são as funcionalidades ou serviços que se espera que o sistema disponibilize em benefício dos usuários (PÁDUA, 2000).
- Os requisitos não funcionais (RNF) são aqueles que não estão ligados diretamente às funcionalidades fornecidas pelo sistema, ou seja, estão relacionadas aos atributos de qualidade de *software* (PÁDUA, 2000).

Thayer e Dorfman (1997) apresentam cinco etapas bem definidas para a Engenharia de Requisitos:

- Levantamento de requisitos de software – Processo por meio do qual os clientes e o desenvolvedor do sistema, possam descobrir e compreender a necessidade do usuário e as restrições que o software deverá apresentar.
- Análise de Requisitos de Software – Processo de análise das necessidades dos usuários e clientes para chegar a uma definição dos requisitos do software.
- Especificação de requisitos de software – Desenvolvimento de um documento que de forma clara cada requisito do sistema de software, para que tanto o cliente quanto o desenvolvedor possam entendê-los.
- Verificação de Requisitos Software – Processo que valida às especificações do software estão de acordo com os requisitos de sistema
- Gerência de requisitos de software – Planejamento e controle do levantamento, análise, especificação e verificação das atividades de requisitos.

Após essas etapas é possível desenvolver os diagramas da UML de forma clara e segura, pois com esses diagrama é possível modelar, especificar e documentar as funcionalidades e atores do sistema, de forma visual, deixando claro o desenvolvimento.

2.9 UML

Segundo Nunes e O'Neill (2004) a UML é uma linguagem que utiliza uma notação padronizada para modelar, especificar, construir, visualizar e documentar os artefatos envolvidos em sistemas de informação por meio do paradigma da orientação a objetos.

Seguindo a ideia do autor, a UML tem como objetivo a especificação, documentação e estruturação dos artefatos do sistema para a maior visualização lógica do desenvolvimento de um sistema de informação. Segundo Page-Jones (2001), a UML tem como principais vantagens:

- Padronização: a UML utiliza uma notação padronizada para que todos os envolvidos possam entender as descrições do sistema.
- Independência de linguagem de programação e metodologia de desenvolvimento. Com isso a UML pode dar suporte a todas as linguagens de programação e aos vários processos de desenvolvimento de *software*.
- Varias visões, ou seja, a UML permite moldar o sistema de forma que tanto o desenvolvedor quanto o cliente possa entender o que vai ser desenvolvido.

A UML disponibiliza diversos diagramas, sendo que cada diagrama possui uma visão diferente para moldar o sistema para que ambos, o desenvolvedor e o cliente possam entender o que será desenvolvido. Entre os diagramas oferecidos pela UML, podemos citar:

- Diagrama de Caso de Uso tem um papel central para a modelagem do comportamento de um sistema, de um subsistema ou de uma classe. Cada um mostra um conjunto de casos de uso e atores e seus relacionamentos. (BOOCH, JACOBSON, RUMBAUGH, 2005).
- Diagrama de Classe, segundo Furlan (1998) representa uma estrutura estática do sistema mostrando os objetos, as relações entre eles, os atributos e as operações que o caracterizam, por meio de uma representação gráfica.
- Diagrama de Atividades, que representa o detalhamento de tarefas e o fluxo de uma atividade para outra de um sistema.

3 O DESENVOLVIMENTO DO SISTEMA

O objetivo do capítulo é descrever o desenvolvimento do sistema proposto, contendo os requisitos de *software*, os estudos de caso de uso e o funcionamento do sistema, para que seja possível identificar os usuários e as suas funcionalidades.

O sistema de alocação de transporte foi desenvolvido a partir da necessidade encontrada de automatizar os processos de solicitação de veículos e gerenciamento de recursos, que são realizadas através do uso de formulários impressos sem controle eletrônico. Com isso, houveram reuniões com a direção do *campus* para que fosse possível analisar a proposta do desenvolvimento de um sistema *web*, para oferecer suporte às atividades operacionais e de gestão do setor de transporte, além de outros dois sistemas, o de controle de diárias e alocação de espaços físicos.

O sistema foi desenvolvido utilizando a linguagem de programação *Ruby* e a *framework Rails*. Uma linguagem caracterizada por ser bastante simples, e uma *framework* com diversos recursos que podem ser utilizados para várias finalidades. Também foram aplicados conceitos de engenharia de *software*, facilitando assim o desenvolvimento de um produto robusto e que atenda as reais necessidades do cliente.

O desenvolvimento do projeto passou por diversas etapas. Iniciando pela análise e especificação necessária para levantar as reais necessidades do cliente em relação às funcionalidades do sistema. Seguido pela Análise e *Design* utilizados para proporcionar uma visão geral da arquitetura do sistema, através da modelagem da solução, com o uso de diagramas UML. Após essas fases, foi realizada a modelagem de dados, que contém o diagrama de entidade relacionamento. Esse diagrama tem como finalidade descrever, de maneira conceitual, os dados utilizados no desenvolvimento do sistema. E por fim os formulários do sistema, mostrando algumas telas das principais funcionalidades desenvolvidas.

3.1 Análise e Especificação de Requisitos

Segundo Pressman (2006) requisito de *software* é uma condição ou uma capacidade com o qual o sistema deve estar de acordo, expressando as necessidades do cliente. Esses requisitos foram divididos em duas categorias, requisitos funcionais e não funcionais.

O processo de especificação de requisitos foi realizado por meio de entrevistas (apêndice A) junto ao chefe do setor transporte da UFPI - CSHNB. Durante essas entrevistas

foram expostos os processos do referido setor, identificando assim suas dificuldades e necessidades.

3.2 Requisitos Funcionais

Os requisitos funcionais (RF) são as funcionalidades ou serviços que se espera que o sistema possua. Eles podem variar de acordo com o tipo de *software* que será desenvolvido, podem ser expressos de diversas maneiras e em diferentes níveis de detalhamento. Os requisitos funcionais levantados podem ser vistos no Quadro 1.

Quadro 1 - Requisitos Funcionais

Requisitos Funcionais	Descrição
RF01 - O sistema deverá possuir controle de usuários.	<p>O controle de usuários deve ser feito por meio de <i>login</i> e senha. O sistema diferencia os usuários em:</p> <ul style="list-style-type: none"> • Os usuários não cadastrados no sistema poderão visualizar apenas as solicitações aprovadas realizadas por outras pessoas. • Os usuários comuns cadastrados no sistema poderão solicitar transporte, visualizar suas solicitações, sendo possível editar e excluir suas solicitações que ainda não foram analisadas. • O usuário chefe do setor poderá gerenciar as solicitações, aprovar ou desaprovar, gerenciar os motoristas e veículos, agrupar solicitações (regra que será descrita no decorrer do capítulo), gerar relatórios de repasse de diárias dos motoristas. • Usuário Coordenador Administrativo Financeiro (CAF) e Diretoria, que possuem as mesmas funcionalidades dos outros usuários, mas com uma diferença, eles poderão modificar o que foi feito pelo chefe do setor.
RF02 - O sistema deve oferecer uma área de solicitação de transporte.	<p>A interface de solicitação deve conter os seguintes campos:</p> <ul style="list-style-type: none"> • Data de partida (deve ser maior ou igual à data corrente); • Data de retorno (deve ser maior ou igual à data de partida); • Destino; • Objetivo; • Justificativa; • Distância (único campo não obrigatório).
RF03 - Aprovação e Desaprovação de Solicitação.	<p>O sistema deve se comportar da seguinte forma:</p> <ul style="list-style-type: none"> • Caso a solicitação seja aprovada, deve aparecer uma

	<p>barra verde sobre a solicitação, tanto para o chefe do setor, quanto para o solicitante, indicando que a solicitação foi aprovada;</p> <ul style="list-style-type: none"> • Caso a solicitação for negada, deve ser apresentada uma justificativa de desaprovação. Por fim, deve aparecer uma barra vermelha sobre a solicitação, indicando que mesma foi negada; • As solicitações em espera manterão a cor padrão do sistema, cinza claro;
RF04 – Cadastro e gerenciamento de Motoristas	O cadastro de motorista necessita apenas do nome. O sistema deve gerenciar o número de diárias do motorista, pois só poderá acumular quatro (04) diárias ao mês. Caso a distancia até o destino da viagem for maior do que seiscentos quilômetros, mais de um motorista deve ser escalado.
RF05 – Cadastro e gerenciamento de veículo	O cadastro de veículo necessita dos seguintes dados: o número de passageiros, placa e tipo. Caso seja necessário mais de um veículo pode ser escalado para atender a uma solicitação.
RF06 – Agrupamento de Solicitações.	O sistema deve agrupar mais de uma solicitação, caso estas possuam o mesmo destino. As solicitações que possuem o mesmo destino devem possuir as mesmas datas de partida e retorno.
RF07 – Requisitar relatórios	O sistema deve emitir relatórios ao administrador de acordo como solicitado. Os relatórios disponíveis no sistema são: <ul style="list-style-type: none"> • Relatório com as atividades de cada motorista; • Relatório geral com as atividades de todos os motoristas.

Fonte: O Autor (2014)

3.2.1 Requisitos Não Funcionais

Os requisitos não funcionais (RNF) são aqueles que estão relacionadas às propriedades que o sistema deve ter, como confiabilidade, segurança, usabilidade, desempenho e outros atributos encontrados na qualidade de *software*. Os requisitos não funcionais levantados podem ser vistos no Quadro 2.

Quadro 2 - Requisitos Não Funcionais

Requisitos não funcionais	Descrição
RNF01 – Usabilidade (Interface)	A <i>interface</i> do sistema deve ser amigável e objetiva, ou seja, suas funções devem estar bem visíveis e possuir uma padronização de cores.
RNF02 – Usabilidade (Mensagem)	As mensagens de erro do sistema deverão ser de fácil entendimento pelo usuário, ou seja, devem ser claras e deve possuir uma

	indicação de como resolver um problema ocorrido.
RNF03 – Usabilidade (treinamento)	Os usuários devem ser treinados ou uma orientação inicial para facilitar o aprendizado quanto ao uso do sistema.
RNF04 – Eficiência	O sistema deverá possuir uma boa agilidade em seus processos.
RNF05 – Confiabilidade	Durante algum erro o sistema deverá informar ao usuário e em seguida o sistema deverá retornar a um estágio consistente.
RNF06 – Segurança (<i>login</i>)	O usuário deverá possuir um <i>login</i> e senha para que possa acessar as funcionalidades do sistema.
RNF07 – Segurança (dados)	O sistema deverá possuir um banco de dados confiável, ou seja, manipular as principais operações do banco de forma segura e precisa.
RNF08 – Segurança (classificação de usuários)	O sistema deverá classificar os usuários por níveis de acordo com cada funcionalidade do sistema.
RNF09 – Performance	O banco de dados deve manipular uma quantidade considerável de dados sem perda ou uma leve queda de desempenho
RNF10 – Manutenibilidade	Deve ter uma equipe treinada na manutenção do sistema para atender as solicitações do usuário quando necessário.

Fonte: O Autor (2014)

3.3 Análise e Design

Esta sessão tem como objetivo analisar e detalhar a solução do sistema de acordo com os requisitos levantados e validados no sessão 3.1. Para isso, deve-se ter uma visão geral da arquitetura do sistema e a modelagem da solução através de diagramas UML.

Para o desenvolvimento do projeto foi utilizado o diagrama de caso de uso, que expressa o comportamento em alto nível que o sistema deve executar para cada ator; O diagrama de classe que representa as classes do sistema, assim como os seus relacionamentos; O diagrama de atividades, que apresenta o fluxo de tarefas que podem ser executadas pelo sistema e pelos atores, e o modelo de entidade e relacionamento, cuja finalidade é descrever os dados que foram utilizados no sistema proposto.

3.3.1 Diagramas de Caso de Uso

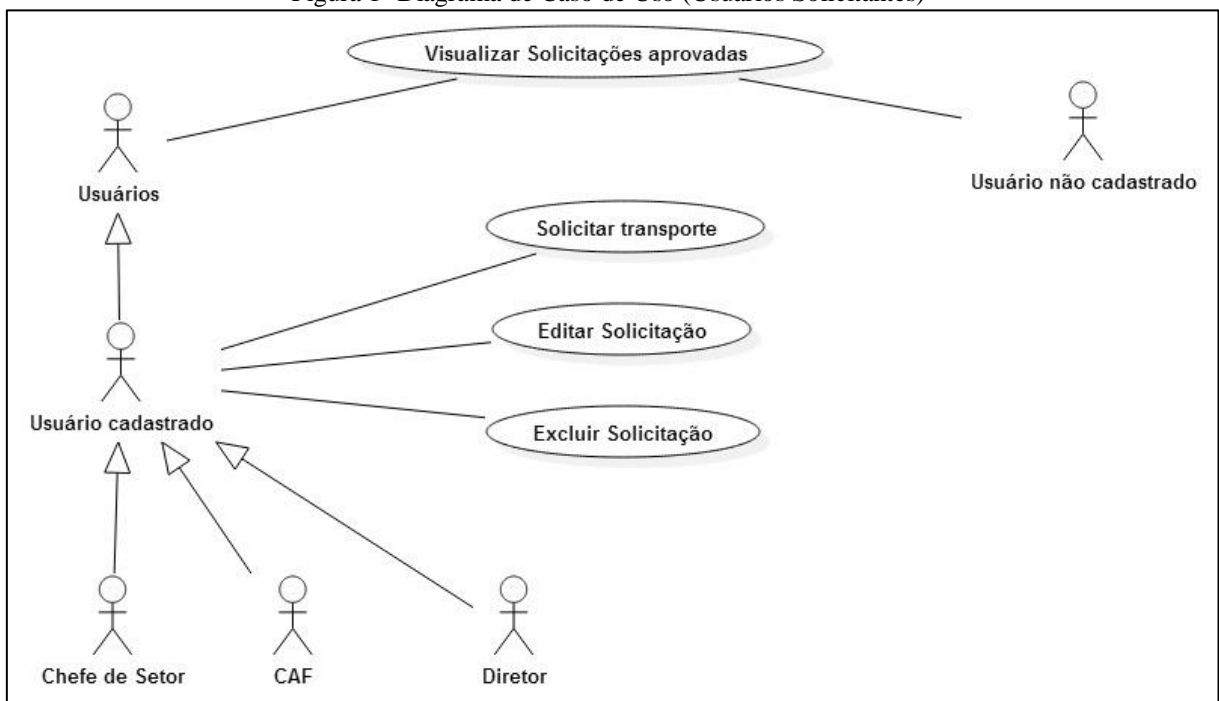
Os diagramas de Caso de Uso são utilizados para serem entendidos por qualquer pessoa da área de desenvolvimento de sistemas ou não, proporcionando assim que tanto o cliente quanto o desenvolvedor possam entender o que será desenvolvido.

Os elementos que compõem um diagrama de caso de uso são os atores, o próprio caso de uso e o sistema. O ator interage com sistema, requisitando uma ação e recebendo uma reação do mesmo. Os atores são identificados por quem inicializa o fluxo de atividades do sistema, fornecendo dados e recebendo informações.

As Figuras 1 e 2 mostram os casos de uso, assim como os seus atores, que podem ser o diretor, o CAF, o chefe do setor de transporte, o usuário cadastrado e usuário não cadastrado, e suas ações e restrições no sistema, como descritos no subcapítulo 3.1.1.

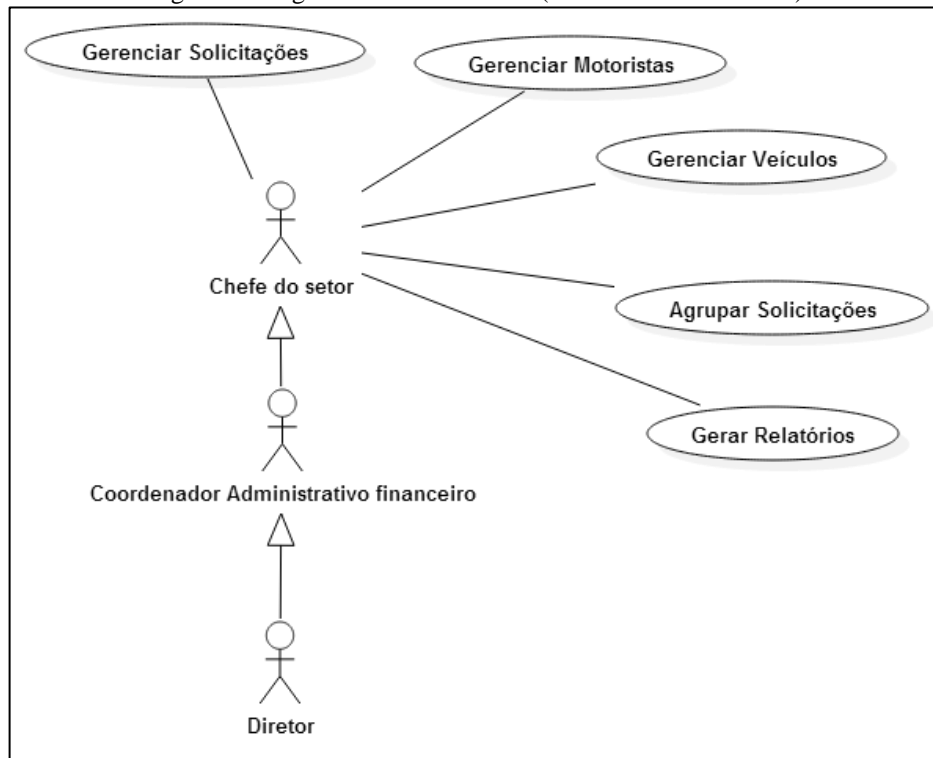
As setas significam que o usuário herda as ações do usuário apontado, como pode ser visto na Figura 1. O usuário chefe de setor herda as atividades do usuário cadastrado, assim como os usuários CAF e diretor herdam as atividades do usuário cadastrado. Essa regra também se aplica na Figura 2.

Figura 1- Diagrama de Caso de Uso (Usuários Solicitantes)



Fonte: O Autor (2014)

Figura 2- Diagrama de Caso de Uso (Usuário Administrador)



Fonte: O autor (2014)

3.3.2 Diagrama de Classe

No diagrama de Classe, uma classe é representada por um retângulo, que possui duas linhas que separam três partes. A primeira parte representa o nome da classe, a segunda os atributos da classe e por fim os métodos como podem ser visto na Figura 3, que representa a classe Usuário.

Figura 3- Representação da classe Usuário

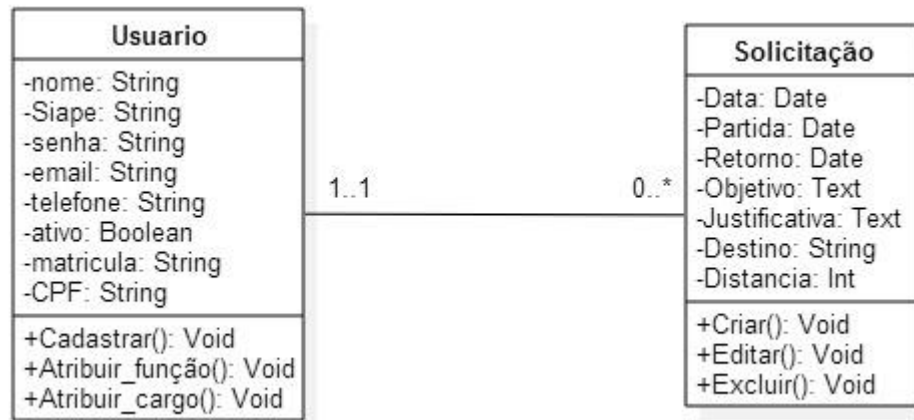


Fonte: O Autor (2014)

Além disso, o diagrama de classe possui associação entre classe, que permite especificar quais classes estão se relacionando com outras classes. As associações são representadas por uma reta utilizada para unir as duas classes, como visto na Figura 4. Os

valores nas extremidades da reta indicam a multiplicidade das associações, seguindo a notação, que pode ser observado no Quadro 3.

Figura 4- Representação de associação entre classes



Fonte: O Autor (2014)

Um usuário pode estar associado a várias solicitações, ou a nenhuma. Uma solicitação necessariamente tem que estar associada a um usuário e a não mais do que uma, conforme mostra a figura 4.

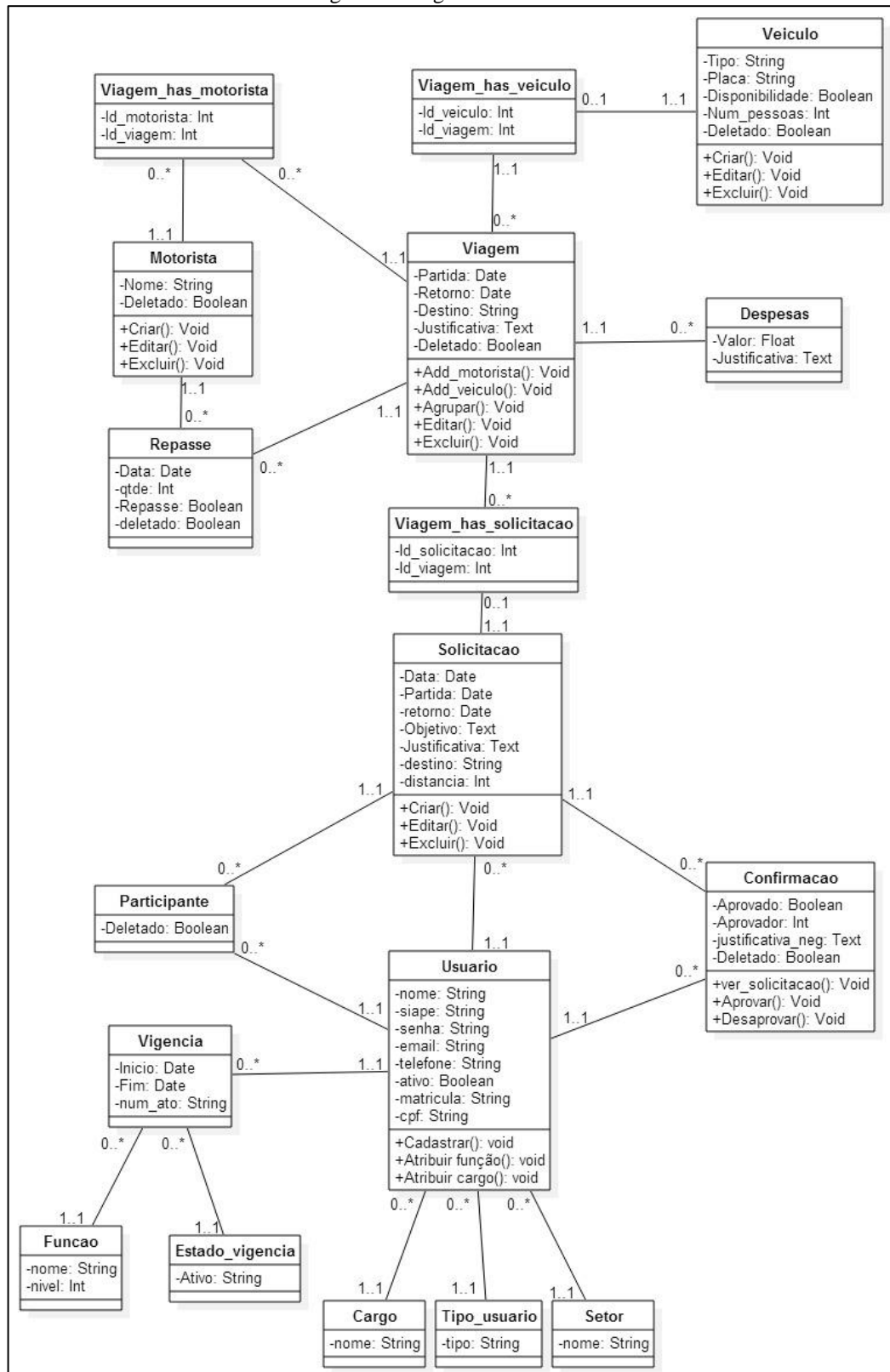
Quadro 3 - Notação de Relacionamento

Tipo	Significado
0..1	Zero ou um registro.
0..*	Não existe limite para o número de registros.
1..1	Exatamente um registro.

Fonte: O Autor (2014)

O diagrama de classes modelado para o sistema proposto. Nesse diagrama foram utilizados os tipos de associações apresentadas anteriormente, assim como a representação de uma classe por meio de um retângulo. Ele foi dividido em três partes, sendo apenas a primeira parte referente ao nome da classe, e tem que ser preenchido obrigatoriamente, como pode ser visto na Figura 5.

Figura 5- Diagrama de Classes



Fonte: O Autor (2014)

3.3.2.1 Descrição Textual do Diagrama de Classes

Para um melhor entendimento do diagrama de classes criado, é apresentada a descrição das classes de forma textual.

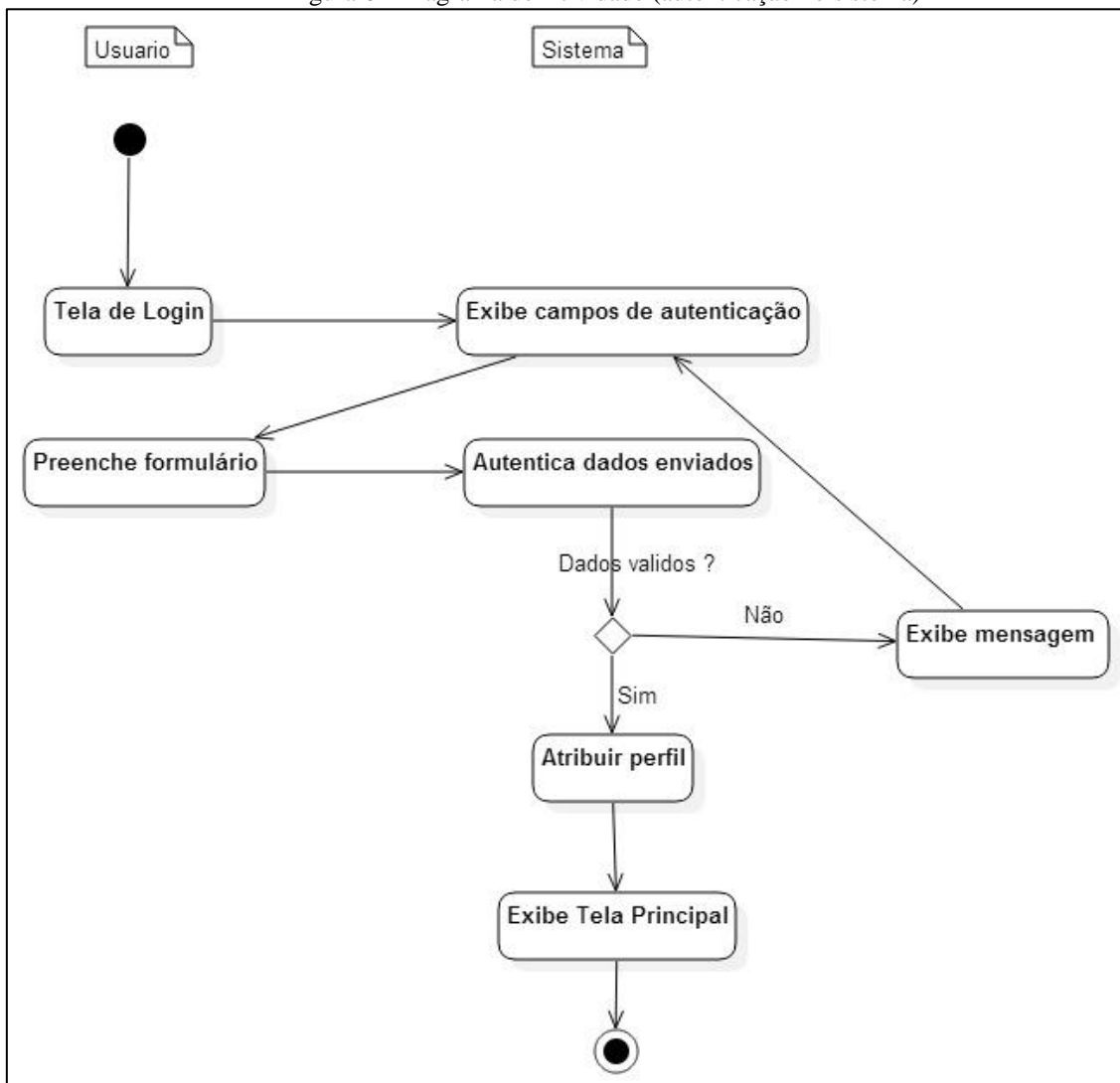
- a) As classes Usuário (Quadro 26) e Tipo_usuario (Quadro 25) estão relacionadas, sendo que a classe Usuários armazena as informações dos usuários. A classe Tipo_usuario é utilizada para diferenciar os tipos de usuários (aluno, funcionário público e externo).
- b) Cargo (Quadro 15), Estado_vigência (Quadro 18), Função (Quadro 19), Setor (Quadro 23), Vigência (Quadro 32): são as classes que armazenam as informações para controlar as funções dos usuários. Essas classes estão relacionadas com a classe Usuário.
- c) Confirmação: classe utilizada pelo administrador para realizar a análise das solicitações. Classe relacionada com Usuários e Solicitação. (Quadro 16).
- d) Despesas: classe que armazena as informações sobre as despesas da viagem. Classe relacionada com Viagem. (Quadro 17).
- e) Motorista: classe utilizada pelo administrador para armazenar as informações sobre motorista, podendo criar, editar e excluir motoristas. (Quadro 20).
- f) Participantes: classe utilizada para adicionar os membros da viagem, estando relacionada com Usuário e Solicitação. (Quadro 21).
- g) Repasse: classe responsável por guardar as diárias dos motoristas, possui um campo para informar se a diária foi repassada ou não. (Quadro 22).
- h) Solicitação: classe utilizada para realizar o processo de solicitação de transporte, podendo ser acessada por todos os tipos de usuário, exceto o usuário externo e não cadastrado no sistema. Sendo possível, solicitar, editar, excluir e adicionar participante a solicitação. (Quadro 24).
- i) Veiculo: classe utilizada pelo administrador para criar, editar e excluir motoristas. (Quadro 27).
- j) Viagem: classe utilizada para agrupar os recursos necessários para realização da viagem, como motorista e veículo. (Quadro 28).
- k) Viagem_has_motorista: classe utilizada para atender ao RF04. (Quadro 29).
- l) Viagem_has_solicitação: classe utilizada para agrupar solicitação, como descrito no RF06. (Quadro 30).
- m) Viagem_has_veiculo: classe utilizada para atender ao RF05. (Quadro 31).

3.3.3 Diagrama de Atividades

Nessa sessão será apresentado o diagrama de atividades, que representa o detalhamento de tarefas e o fluxo de uma atividade para outra de um sistema. Nem todos os sistemas necessitam da elaboração do diagrama de atividades, pois nem todas as tarefas do sistema necessitam de um detalhamento.

“Os diagramas de atividade não são importantes somente para a modelagem de aspectos dinâmicos de um sistema ou um fluxograma, mas também para a construção de sistemas executáveis por meio de engenharia reversa.” (GILLEANES, 2004).

Figura 6 - Diagrama de Atividade (autenticação no sistema)



Fonte: O Autor (2014)

O diagrama de atividade que pode ser visto na Figura 6, representa o RF01, referente à autenticação do usuário no sistema, como já visto anteriormente podem ser: usuários

cadastrados, chefe do setor de transporte, CAF e diretor. O processo de autenticação possui os seguintes fluxos (Quadros 4 e 5):

Quadro 4 - Diagrama de atividades (Autenticação do Usuário) – Fluxo principal

Fluxo Principal	
Ações do Ator (usuário)	Ações do sistema
1. O fluxo inicia quando o ator acessa alguma área do sistema que necessite de autenticação;	
	2. O sistema exibe o formulário com os campos <i>e-mail</i> e senha, necessários para autenticação;
3. O ator preenche o formulário e acessa a opção de entrar;	
	4. O sistema autentica os dados;
	5. O sistema atribui o perfil necessário ao ator e o redireciona para a tela principal de seu perfil;
	6. Fluxo principal encerrado.

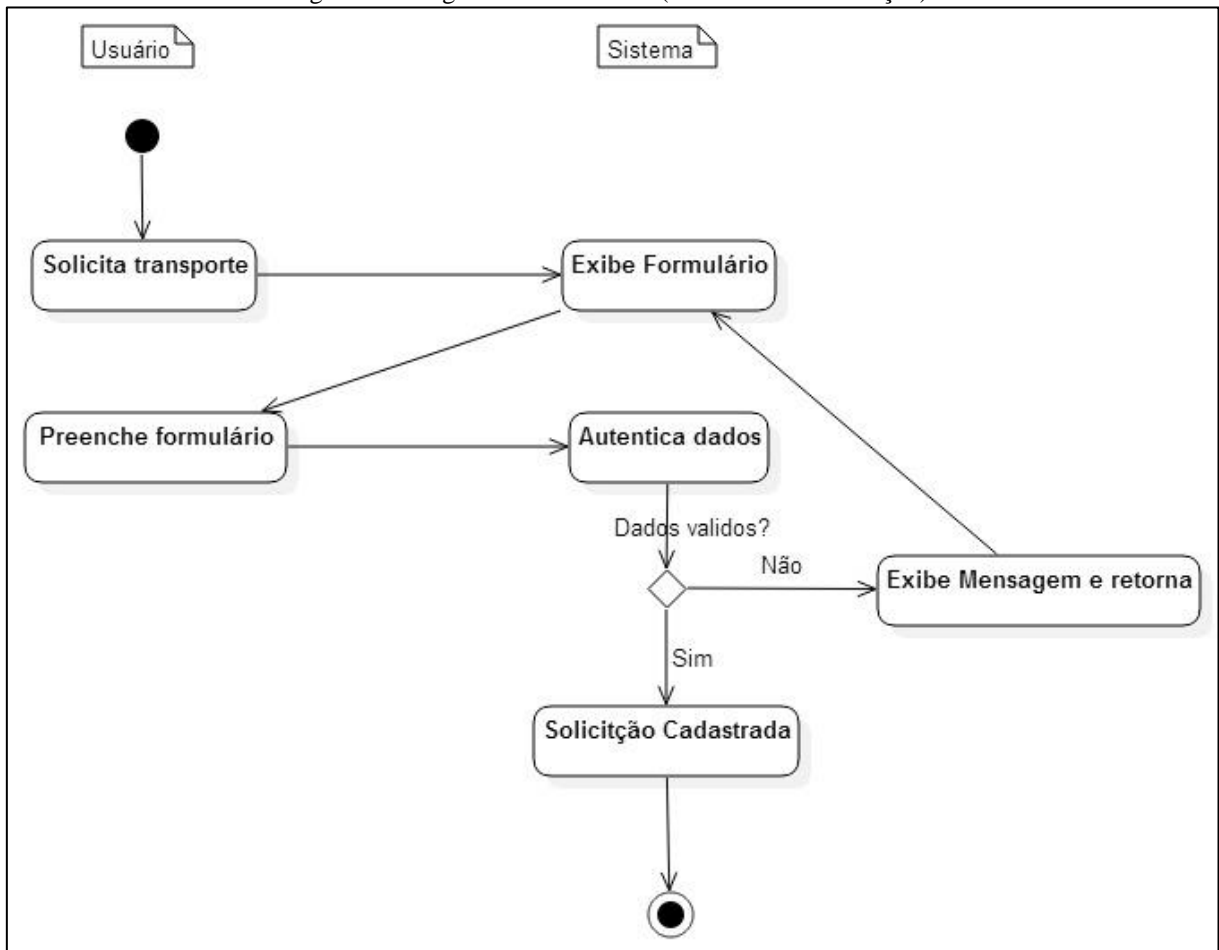
Fonte: O Autor (2014)

Quadro 5 – Diagrama de atividades(Autenticação do Usuário) – Fluxo Alternativo

Fluxo Alternativo	
Ações do Ator (usuário)	Ações do sistema
1. O fluxo inicia quando o ator acessa alguma área do sistema que necessite de autenticação;	
	2. O sistema exibe o formulário com os campos <i>e-mail</i> e senha, necessários para autenticação;
3. O ator preenche o formulário e acessa a opção de entrar;	
	4. O sistema encontra erro na validação dos dados;
	5. O sistema exibe uma mensagem de erro e retorna ao formulário de autenticação;
	6. Fluxo alternativo encerrado.

Fonte: O Autor (2014)

Figura 7 – Diagrama de Atividade (Processo de Solicitação)



Fonte: O autor (2014)

O diagrama de atividades na Figura 7 representa o processo de solicitação de veículo. Este diagrama corresponde ao RF02. E possui os seguintes fluxos (Quadros 6 e 7):

Quadro 6 - Diagrama de Atividade (Processo de Solicitação)
– Fluxo Principal

Fluxo Principal	
Ações do ator (usuário)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de solicitações;	
	2. O sistema exibe o formulário com os campos necessários para esta ação no sistema;
3. O ator usuário preenche os campos e acesso a opção salvar solicitação;	
	4. O sistema autentica os dados;
	5. Sistema cria um registro no banco de dados;
	6. Fluxo encerrado.

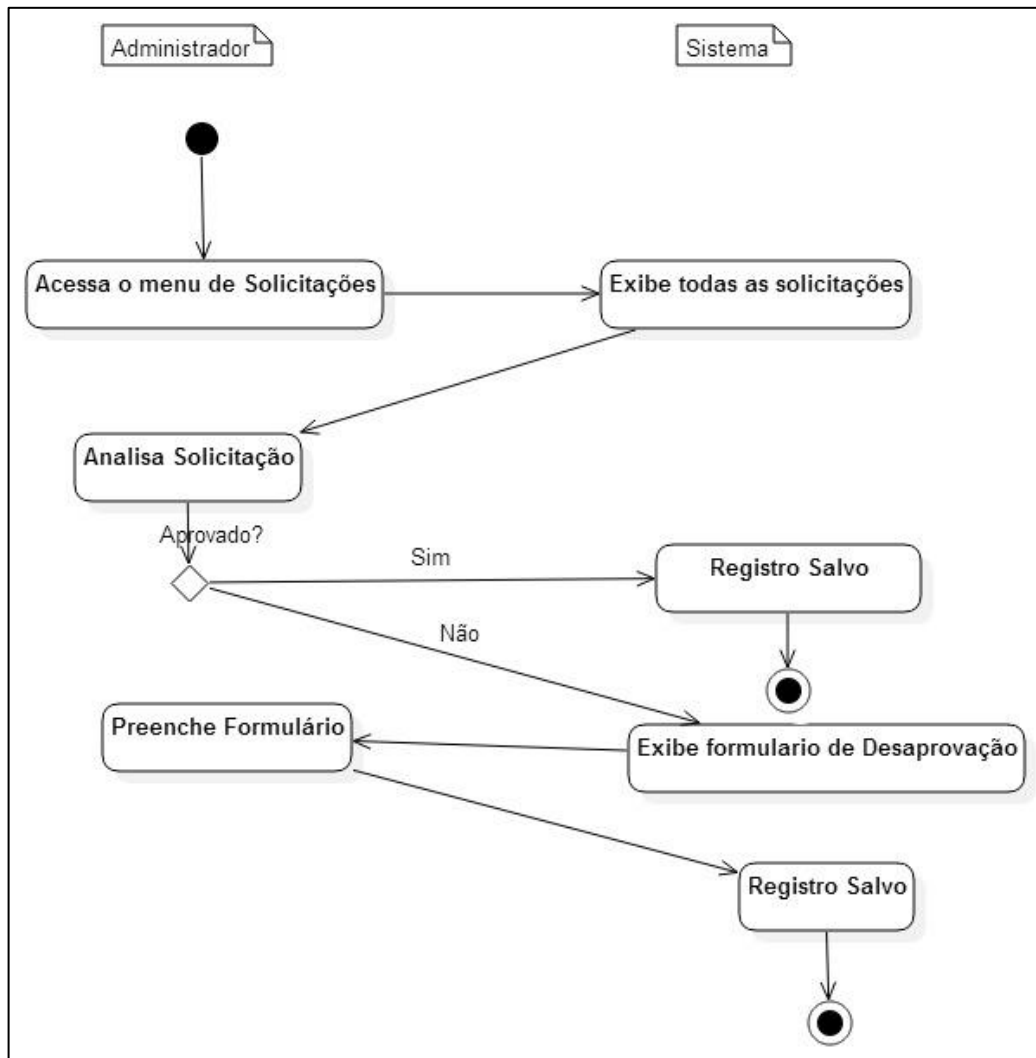
Fonte: O Autor (2014)

Quadro 7 - Diagrama de Atividade (Processo de Solicitação)
– Fluxo Alternativo

Fluxo Alternativo	
Ações do ator (usuário)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de solicitações;	
	2. O sistema exibe o formulário com os campos necessários para esta ação no sistema;
3. O ator usuário preenche os campos e acessa a opção salvar solicitação;	
	4. O sistema encontra erro na validação dos dados;
	5. Sistema exibe uma mensagem de erro e retorna ao formulário de solicitação;
	6. Fluxo encerrado.

Fonte: O Autor (2014)

Figura 8 - Diagrama de Atividade (Análise de Solicitação)



Fonte: O autor (2014)

O diagrama de atividades representado pela Figura 8 corresponde ao processo de aprovação e desaprovação de solicitações de veículo (RF03). Esse processo possui os seguintes fluxos (Quadros 8 e 9):

Quadro 8 – Diagrama de Atividade (Análise de Solicitação) – Fluxo Principal

Fluxo Principal	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de solicitações;	
	2. O sistema exibe uma lista com todas as solicitações;
3. O ator analisa a solicitação e a aprova;	
	4. O sistema salva o registro de aprovação;
	5. Fluxo encerrado.

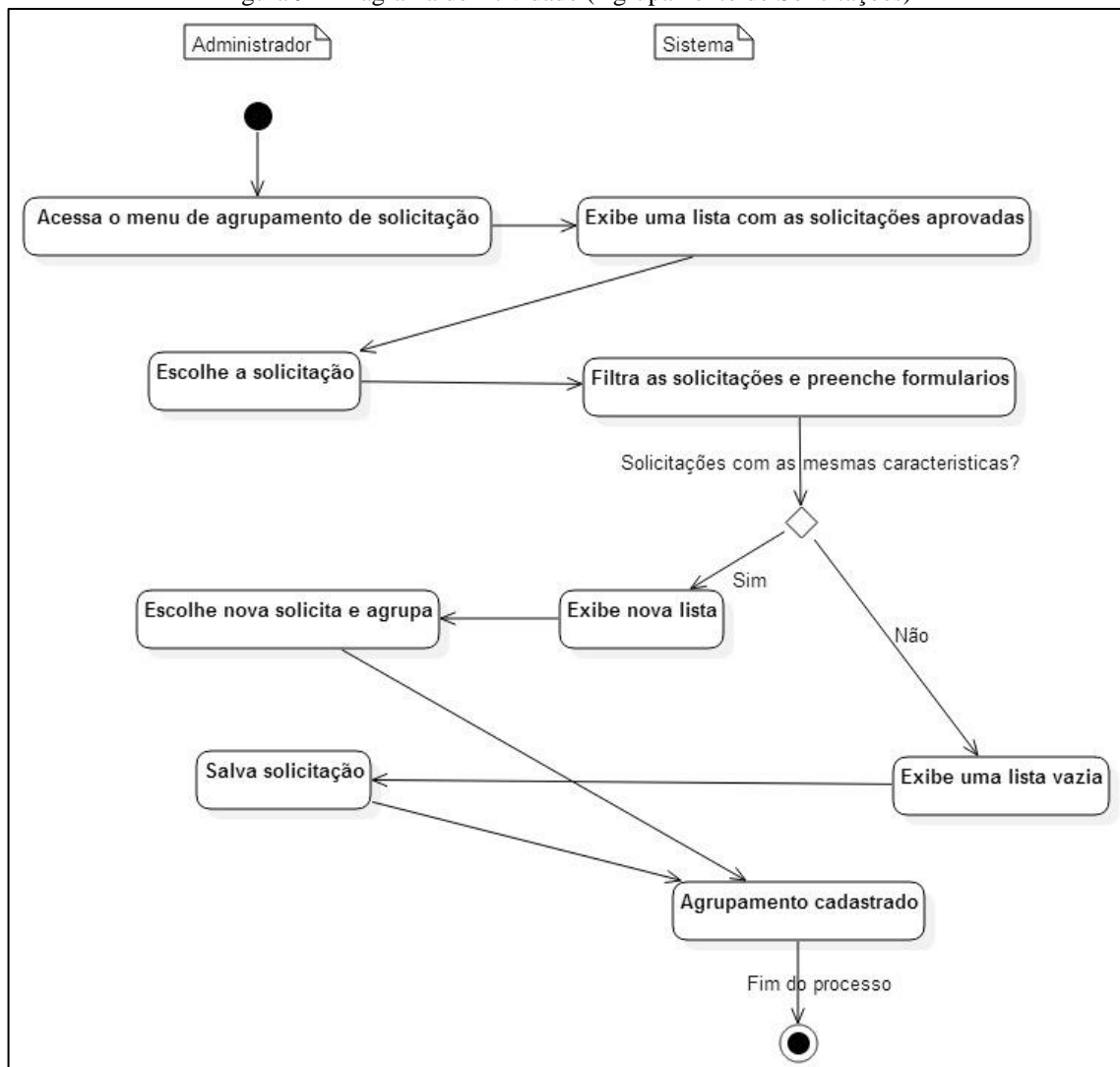
Fonte: O Autor (2014)

Quadro 9 - Diagrama de Atividade (Análise de Solicitação) – Alternativo

Fluxo Alternativo	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de solicitações;	
	2. O sistema exibe uma lista com todas as solicitações;
3. O ator analisa a solicitação e a desaprova;	
	4. O sistema exibe o formulário de desaprovação;
5. O ator preenche o formulário e acessa botão de salvar;	
	6. O sistema salva o registro de desaprovação;
	7. Fluxo encerrado.

Fonte: O Autor (2014)

Figura 9 – Diagrama de Atividade (Agrupamento de Solicitações)



Fonte: O autor (2014)

O diagrama de atividade, como mostrado na Figura 9, representa o processo de agrupamento de solicitações, referente ao RF06. Com os seguintes fluxos (Quadros 10 e 11):

Quadro 10 - Diagrama de Atividade (Agrupamento de Solicitações) – Fluxo Principal

Fluxo Principal	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de agrupamento de solicitações;	
	2. O sistema exibe uma lista com as solicitações aprovadas;
3. O ator escolhe a solicitação;	
	4. O sistema filtra as solicitações de acordo com o RF06;
	5. Exibe uma nova lista de solicitações;
6. O ator escolhe as solicitações e acessa o botão de salvar;	
	7. Salva o registro de agrupamento;
	8. Fluxo encerrado.

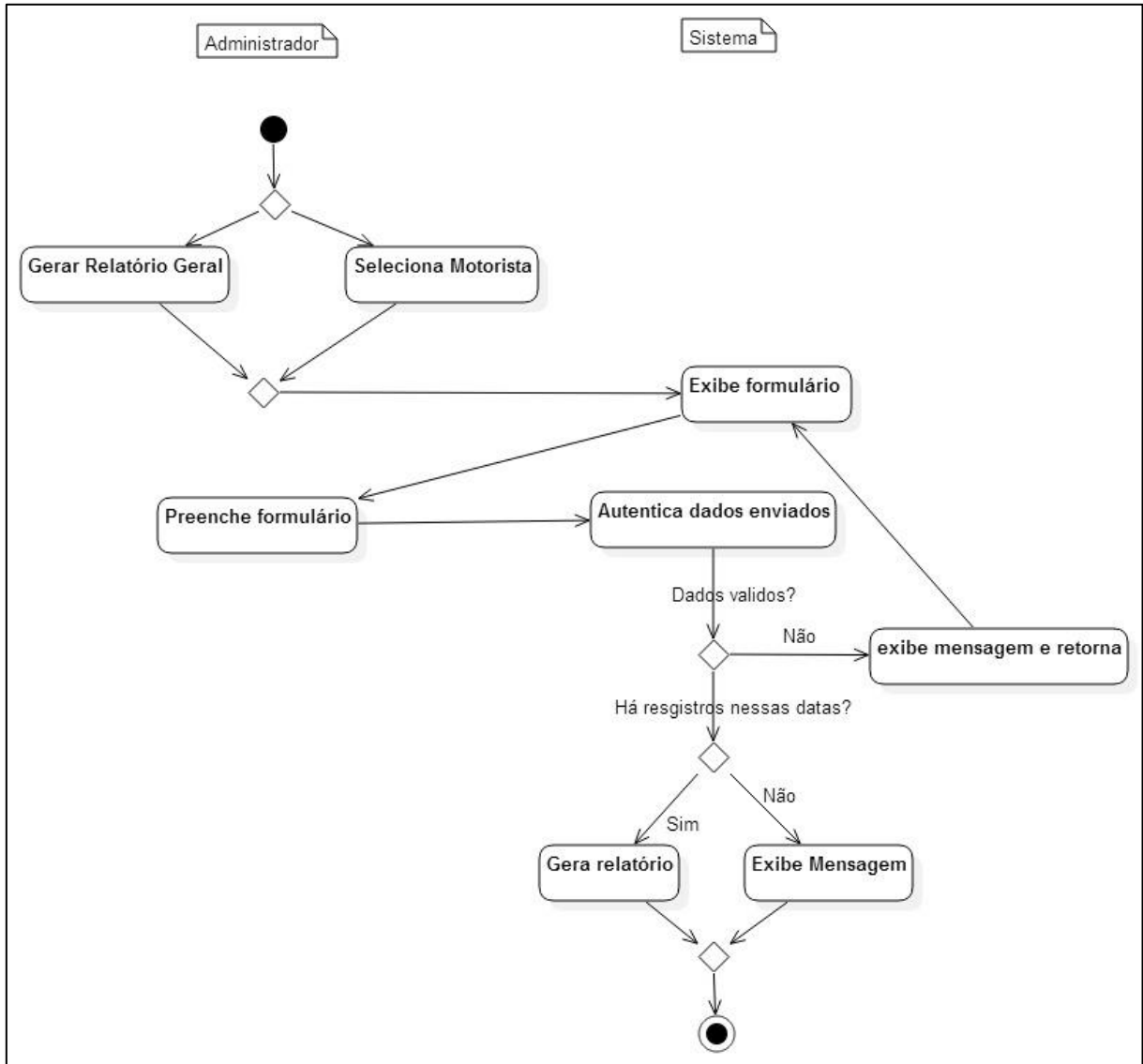
Fonte: O autor (2014)

Quadro 11 - Diagrama de Atividade(Agrupamento de Solicitações) – Fluxo Alternativo

Fluxo Alternativo	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de agrupamento de solicitações;	
	2. O sistema exibe uma lista com as solicitações aprovadas;
3. O ator escolhe a solicitação;	
	4. O sistema filtra as solicitações de acordo com o RF06;
	5. Exibe uma nova lista vazia;
6. O ator acessa o botão de salvar;	
	7. Salva o registro de agrupamento;
	8. Fluxo encerrado.

Fonte: O autor (2014)

Figura 10 - Diagrama de Atividade (Gerar relatórios)



Fonte: O autor (2014)

O processo de geração de relatório (Figura 10) demonstra o fluxo de ações para emissão de relatório geral e específico de cada motorista (RF07), apresentando os seguintes fluxos (Quadros 12, 13, 14):

Quadro 12 – Diagrama de Atividade (Gerar relatórios) – Fluxo Principal

Fluxo Principal	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de motorista, podendo escolher gerar relatório geral ou relatório por motorista;	
	2. O sistema exibe o formulário com os campos de datas;
3. O ator preenche o formulário;	

	4. O sistema autentica os dados;
	5. Caso tenha registros entre as datas, o relatório é gerado;
	6. Fluxo encerrado;

Fonte: O Autor (2014)

Quadro 13 – Diagrama de Atividade (Gerar relatórios) – Fluxo Alternativo 1

Fluxo Alternativo 1	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de motorista, podendo escolher gerar relatório geral ou relatório por motorista;	
	2. O sistema exibe o formulário com os campos de datas;
3. O ator preenche o formulário;	
	4. O sistema encontra erro na validação dos dados;
	5. O sistema informa o erro e retorna ao formulário;
	6. Fluxo encerrado;

Fonte: O Autor (2014)

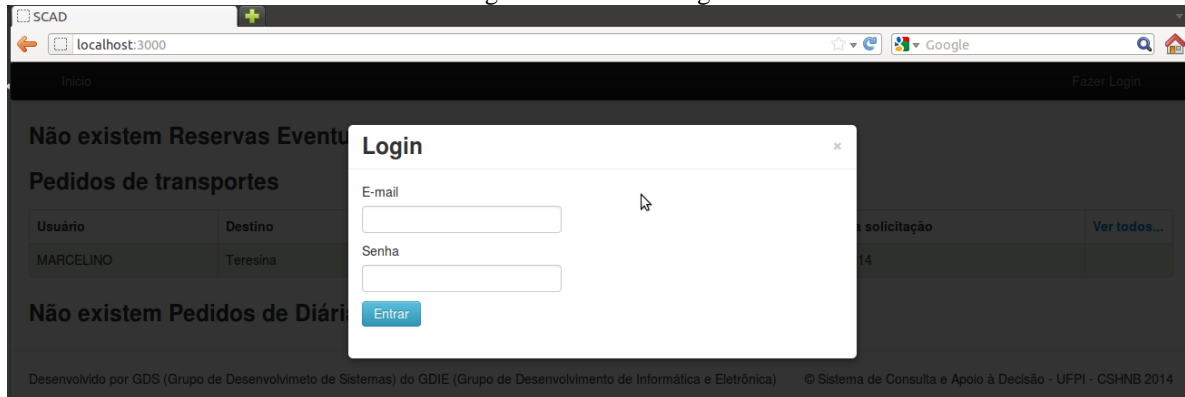
Quadro 14 – Diagrama de Atividade (Gerar relatórios) – Fluxo Alternativo 2

Fluxo Alternativo 2	
Ações do ator (Administrador)	Ações do sistema
1. O processo inicia quando o ator acessa o menu de motorista, podendo escolher gerar relatório geral ou relatório por motorista;	
	2. O sistema exibe o formulário com os campos de datas;
3. O ator preenche o formulário;	
	4. O sistema autentica os dados;
	5. Caso não tenha registros entre as datas, e gerado uma mensagem informando o mesmo;
	6. Fluxo encerrado;

Fonte: O Autor (2014)

principal, exibindo assim suas atividades e sua barra de navegação, respondendo ao RF01 e RNF06, dos Quadros 1 e 2, respectivamente.

Figura 12- Tela de Login



Fonte: O Autor (2014)

Figura 13- Tela de Login (Administrador)



Fonte: O Autor (2014)

A tela do sistema com o formulário de solicitação de veículos, contendo os campos já descritos no RF02 e um quadro de significados dos campos, para um melhor entendimento do usuário quanto ao preenchimento do formulário, como visto na Figura 13.

Figura 14- Tela de Solicitação de Transporte

SCAD Usuários Início Controle de Transporte Minhas Solicitações MARCELINO Sair

Nova Solicitação Salvar Cancelar

Data de partida:

Data de retorno:

Objetivo:

Justificativa:

Destino:

Distancia:

Campo	Significado
Destino	Informar o local de destino da viagem. Nome da cidade seguido do estado exemplo: Picos-PI.
Objetivo	Informe o objetivo da Viagem de forma sucinta, ao final informe um número para que possamos entrar em contato.
Justificativa	Informar os detalhes da viagem como: nome e local do evento, quantidade de participantes e veiculo desejado.
Datas	Informar a data do deslocamento e do Retorno. Caso as datas solicitadas incluam dias em finais de semana e feriados, os custos das diárias a serem pagas ao(s) motorista(s) deverão ser custeadas pelos responsáveis e/ou participantes do evento.
Distancia	Informe a distancia em kilometros até o destino. Caso não saiba deixe em branco.

Desenvolvido por GDS (Grupo de Desenvolvimento de Sistemas) do GDIE (Grupo de Desenvolvimento de Informática e Eletrônica) © Sistema de Consulta e Apoio à Decisão - UFPI - CSHNB 2014

Fonte: O Autor (2014)

O processo de análise das solicitações como pode ser visto nas Figuras 15 e 16, temos uma lista de solicitações a serem analisadas, com os mesmos campos que pode ser vistos na Figura 14. Mas nos campos objetivo e justificativa, temos um *link* chamado *exibir*, para a exibição do texto completo, pois sua visualização ficou comprometida com a enorme quantidade de textos.

O processo de análise tornou-se bem simples, com apenas um *click* é possível aprovar ou desaprovar uma solicitação. Como pode ser visto, temos dois botões, um “V” e um “X”, de aprovar e desaprovar, respectivamente. Atendendo ao requisito de desaprovação, deve-se justificar o porquê (Figura 17).

Figura 15- Aprovação de Solicitação

SCAD Usuários Início Controle de Transporte Minhas Solicitações MARCELINO Sair

Listando Confirmações Solicitações: Filtrar

Data	Saida	Retorno	Usuario	Destino	Objetivo	Justificativa	Participantes	Opções
8/12/2014	16/12/2014	19/12/2014	MARCELINO	Teresina	Exibir	Exibir	1	<input type="checkbox"/> <input type="checkbox"/>
8/12/2014	8/12/2014	15/12/2014	MARCELINO	Maceió	Exibir	Exibir	1	<input type="checkbox"/> <input type="checkbox"/> <input type="button" value="Aprovar"/>
20/11/2014	30/12/2014	2/1/2015	MARCELINO	Tuntum	Exibir	Exibir / Justificativa de desaprovação	1	<input type="checkbox"/> <input type="checkbox"/>

Fonte: O Autor (2014)

Figura 16- Desaprovação de Solicitação

Data	Saida	Retorno	Usuario	Destino	Objetivo	Justificativa	Participantes	Opções
8/12/2014	16/12/2014	19/12/2014	MARCELINO	Teresina	Exibir	Exibir	1	✓ ✕
8/12/2014	8/12/2014	15/12/2014	MARCELINO	Maceió	Exibir	Exibir	1	✓ ✕
20/11/2014	30/12/2014	2/1/2015	MARCELINO	Tuntum	Exibir	Exibir / Justificativa de desaprovação	1	✕ Não Aprovar

Fonte: O Autor (2014)

Figura 17- Justificativa de Desaprovação

Justificativa de desaprovação

Justificativa

Desenvolvido por GDS (Grupo de Desenvolvimento de Sistemas) do GDIE (Grupo de Desenvolvimento de Informática e Eletrônica) © Sistema de Consulta e Apoio à Decisão - UFPI - CSHNB 2014

Fonte: O Autor (2014)

A tela de agrupamento de solicitação, como pode ser visto na Figura 18. É uma funcionalidade para diminuir o gasto com os recursos do setor, pois atende várias solicitações por vez, mas devem-se seguir as regras descritas no RF06.

Na figura pode-se ver os campos dinâmicos de destino, datas de saída, retorno, número de passageiros e os campos horário de partida e retorno, que são os campos preenchidos posteriormente pelo chefe do setor para informar tais horários. Logo abaixo temos as solicitações aprovadas, sendo que cada uma tem um botão “+” do lado direito, para adicionar a viagem. Com isso o sistema filtra as solicitações e retorna as que se enquadram nas regras descritas no RF06.

Figura 18- Agrupamento de Solicitações

Dados da Viagem Salvar Cancelar

Destino: Numeros de passageiros:

Data de saída: Horário de partida:

Data de retorno: Horário de retorno:

Solicitante	Saída	Retorno	Destino	Numero de participantes	Ação
MARCELINO	15/12/2014	16/12/2014	Teresina	0	

Solicitante	Saída	Retorno	Destino	Numero de participantes	Ação
-------------	-------	---------	---------	-------------------------	------

Desenvolvido por GDS (Grupo de Desenvolvemento de Sistemas) do GDIE (Grupo de Desenvolvimento de Informática e Eletrônica) © Sistema de Consulta e Apoio à Decisão - UFPI - CSHNB 2014

Fonte: O Autor (2014)

A tela de geração de relatórios é apresentada pela Figura 19, mostra a lista de motoristas cadastrados no sistema. Na Figura podemos ver a coluna de ações, com os botões representados pelo ícone de um lixeiro, que significa excluir, outro de um papel com uma caneta em cima, que significa editar, e por fim um livro que significa gerar relatório. Podemos notar que cada motorista possui esse botão com o formato de um livro, presumindo-se que esse botão é para a geração de relatórios individuais. O relatório geral, como pode ser visto encontra-se na segunda barra de ações do sistema, que fica na tela de motorista.

Figura 19- Geração de Relatórios

Listando Motoristas Criar novo Gerar Relatório Geral

Nome	Ações
JOÃO DA SILVA	
LUIS PERREIRA	
MARCOS OLIVEIRA	

Desenvolvido por GDS (Grupo de Desenvolvemento de Sistemas) do GDIE (Grupo de Desenvolvimento de Informática e Eletrônica) © Sistema de Consulta e Apoio à Decisão - UFPI - CSHNB 2014

Fonte: O Autor (2014)

CONCLUSÕES E TRABALHOS FUTUROS

O presente trabalho apresentou o desenvolvimento do sistema de alocação de transporte, seguindo os conceitos de engenharia de *software*, mostrando a fase de levantamento de requisitos, como vista anteriormente, realizada por meio de entrevista junto ao interessado pelo desenvolvimento do sistema. Com tais requisitos foi possível entender as especificações do sistema, posteriormente utilizado para a realização do detalhamento dos casos de uso, entre outros diagramas da UML.

Durante o desenvolvimento, fora estudado a tecnologia *Ruby on Rails*, utilizada como plataforma de desenvolvimento *web*. Foi feito um estudo sobre o *Scrum*, metodologia de desenvolvimento ágil utilizada para desenvolver o projeto, pelo fato de uma de suas principais características é manter o cliente sempre por perto, para que seja possível analisar e validar o que está sendo feito.

O sistema encontra-se na fase de iniciação de treinamentos dos clientes reais. Como resultado, todos os módulos do sistema foram testados em nível de protótipo, atendendo a todas as funcionalidades exigidas pelo cliente. Um dos principais pontos desse projeto foi facilitar, agilizar os processos de alocação, e melhorar a comunicação entre o setor de transporte com os seus usuários. Fazendo com que ambos tenham acesso às informações relevantes sobre esse processo, em qualquer local que possua acesso a *internet*. Contudo, o sistema encontra-se acessível somente na rede interna da UFPI. Pois estamos aguardando recurso de publicação na *internet*, para que o sistema esteja acessível em nível global.

Como sugestões de trabalhos futuros, é possível que seja realizado um estudo de usabilidade no sistema, para tornar sua *interface* mais amigável, proporcionando maior facilidade de sua utilização. Outra possibilidade do trabalho seria a implementação de responsividade, para que o sistema se adapte as telas dos dispositivos móveis mantendo a mesma qualidade de uso encontrada no formato de tela padrão para qual o sistema foi desenvolvido.

REFERÊNCIAS

- BATISTA, EMERSON DE OLIVEIRA. **Sistema de Informação: o uso consciente da tecnologia para o gerenciamento.** São Paulo: Saraiva, 2004.
- COCKBURN, A., **Agile Software Development**, Addison-Wesley, 2002.
- DATE C. J. – **Introdução a Sistemas de Bancos de Dados** – editora Campus, 2003.
- FAORO, M. DE ABREU; RODRIGUES, F. R.; de VARGAS, R. F.; **Um estudo sobre os principais tipos de sistemas de informação.** UCS VACARIA – RS. Dezembro de 2010.
- FERREIRA, D.; COSTA, F.; ALONSO, F.; ALVES, P.; NUNES, T. **SCRUM - Um Modelo Ágil para Gestão de Projetos de Software.** Disponível em: <http://paginas.fe.up.pt/~aaguiar/es/artigos%20 finais/es_final_19.pdf>. Acesso em: 17 jul. 2014.
- FILHO, PAULA, PÁDUA, WILSON DE. **Engenharia de software: fundamentos, métodos e padrões.** São Paulo: LTC Editora, 2000.
- FUENTES, VINÍCIUS B. **Ruby on Rails: coloque sua aplicação web nos trilhos.** Casa do código. 2012.
- FURLAN, JOSÉ, D. **Modelagem de Objetos através da UML.** São Paulo: Makron Books, 1998.
- GILLEANES T. A. GUEDES. **UML: Uma Abordagem Prática.** Novatec Rio de Janeiro: 2004.
- HARRISON, T. H. **Intranet, data warehouse: ferramentas e técnicas para a utilização do data warehouse na intranet.** São Paulo: Siciliano, 1998.
- KESSLER, L.S.L.; ALVES, A. C. R; PAIVA, A. P. S; **Projeto, análise e desenvolvimento de um sistema web para gerenciamento de ofertas de emprego.** Faculdade de engenharia da universidade estadual paulista – campus de guaratinguetá, 2002.
- LOBO. EDISON J.R. **Criação de sites em php.** São Paulo: Digerati Books, 2007.
- MARTIN, JAMES - **Information Engineering (volume 1)** – Editora Prentice Hall, 1990.
- MOSIMANN, CLARA PELLEGRINELLO; FISCH, SÍLVIO. **Controladoria: seu papel na administração de empresas.**- 2. ed. – 6. reimpr.- São Paulo: Atlas, 2009.
- NUNES, M. & O’NEILL, H. **Fundamental da UML** (3ª edição). Lisboa: FCA, 2004.
- O’BRIEN, JAMES A. **Sistemas de informação: e as decisões gerenciais na era da internet.** 3. ed. São Paulo: Saraiva, 2010.

OLIVEIRA, D de P. R. de. **Sistemas de Informações Gerenciais: Estratégicas, Táticas e Operacionais**. 13. ed. São Paulo: Atlas, 2010.

OLIVEIRA, DJALMA DE PINHO REBOUÇAS DE. **Sistemas de informação gerenciais: estratégias, táticas, operacionais**. 8. ed., São Paulo: Atlas, 2002.

PÁDUA, WILSON DE. **Engenharia de Software: fundamentos, métodos e padrões**. Editora LTC, 2000.

PAGE-JONES, MEILIER. **Fundamentos do desenho Orientado a Objetos com UML**. trad. PASCHOA, Celso, R. ver. FURLAN, José, D. São Paulo: Makron Books, 2001.

PEPPERS AND ROGERS GROUP DO BRASIL. CRM series - marketing 1to1. 2000.

PEREIRA, MARIA JOSÉ LARA DE BRETÃS; FONSECA, JOÃO GABRIEL MARQUES. **Faces da Decisão: as mudanças de paradigmas e o poder da decisão**. São Paulo: Makron Books, 1997.

PRESSMAN, R.S. **Engenharia de Software**. 6ª Ed, McGraw-Hill, 2006.

RAILSGUIDES; **Ruby on rails guides: getting started with rails**. Disponível em <http://guides.rubyonrails.org/getting_started.html>. Acessado em 25/07/2014.

REZENDE, DENIS ALCIDES; ABREU, ALINE FRANÇA DE. **Tecnologia da informação aplicada a sistemas de informação empresariais: o papel estratégico da informação e dos sistemas de informação nas empresas**. 4.ed. São Paulo: Atlas, 2006.

RODRIGUES, A. **Desenvolvimento para Internet**. Editora LT, 2010

RUMBAUGH, JAMES; BOOCH, GRADY; JACOBSON, IVAR; **UML - Guia do Usuário**. São Paulo: Campus, [2005].

SCHWABER E SUTHERLAND, KEN; JEFF; **Um guia definitivo para o Scrum: as regras do jogo**. 2011. Disponível em <<http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20Portuguese%20BR.pdf>> Acessado em 25/07/2014.

SOUZA, C. B. **Autoria de Artefatos de Software**. Dissertação de Mestrado, Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brasil (2008).

SOUZA, LUCAS. **Ruby: aprenda a programar na linguagem mais divertida**. Casa do código. 2014.

TELES, VINÍCIUS M. **Extreme Programming**. São Paulo: Novatec Editora, 2004.

THAYER, R. H.; DORFMAN, M. **Software Requirements Engineering** . 2ª edição - Editora IEEE, 1997 - Washington

TURBAN, E.; McLEAN, E.; WETHERBE, J. **Information technology for management**. New York: John Wiley & Sons, 1996.

ZAVE, P. **Classification of Research Efforts in Requirements Engineering**. ACM Computing Surveys, Vol. 29, No 4, 1997.

APÊNDICES

APÊNDICE A – ENTREVISTAS

As entrevistas foram realizadas no decorrer do projeto. O início do projeto as entrevistas eram realizadas com intervalos de uma semana para podermos realizar o levantamento de requisitos. Com o decorrer do desenvolvimento do projeto o intervalo de entrevista foi aumentando, pois era levado o que foi desenvolvido para análise do cliente e validação do mesmo. As entrevistas foram realizadas no setor de transporte localizado na UFPI, campus CSHNB, com o funcionário responsável pelo setor.

A cada entrevista era feita uma breve explicação do que se tratava e também uma breve explicação sobre em que fase se encontrava o projeto. O chefe do setor se mostrou bastante prestativo a responder as questões levantadas, e também se mostrou bastante otimista em relação à implantação do sistema. Ele até comentou que o seu trabalho ficaria mais rápido, eficiente e prático se utilizasse o sistema.

Abaixo temos as principais questões que foram levantadas nas entrevistas:

1. Como é realizado o processo de alocação de transporte?

Resposta: “Tudo bem, o processo é feito por meio de planilhas impressas preenchidas a mão, quem solicita tem que preencher um formulário com o destino, datas de partida e retorno, objetivo, se é um congresso, apresentar trabalho, esse tipo de coisa, a justificativa, que é uma breve explicação do evento e às vezes o solicitante coloca até a distância até o destino. A partir dessas informações escalamos o(s) motorista(s) e o(s) veículo(s).”

2. Quais as dificuldades de se trabalhar dessa maneira?

Resposta: “A dificuldade é que é bastante complicado gerenciar essas informações por meio dessas planilhas, pois às vezes uma se perdi, as vezes temos que procurar alguma informação é bastante complicado, pois temos que procurar de documento em documento até achar. Outra coisa é verificar quais motoristas e veículos estão escalados para cada viagem. E também em relação ao motorista, pois devemos tomar muito cuidado, pois cada motorista só pode ter quatro diárias ao mês.”

3. Quais as informações necessárias para solicitar transporte?

Resposta: “É as informações necessárias, como já ditas, o destino, data de partida e retorno, objetivo, justificativa e distância, mas não é obrigatório, pois nós sempre conferimos a distância até o destino.”

4. Quem pode solicitar transporte?

Resposta: “Só quem pode solicitar são os alunos, professores, técnicos administrativos.”

5. Como é feito a análise das solicitações?

Resposta: “A análise para aprovar e desaprovar é feita a partir do objetivo e justificativa.”

6. Como é feito o gerenciamento dos motoristas e veículos?

Resposta: “O gerenciamento de motorista e veículos são feitos por meio de planilhas. Aí eu vou encaixando os motoristas para atender essas solicitações, para que nenhum motorista passe das quatro diárias que ele pode ter ao mês.”

7. Quais seriam as sugestões para o controle de alocação de transporte?

Resposta: “Gostaria que o sistema tivesse uma área que eu pudesse cadastrar os motoristas e outra pra cadastra os veículos. E também uma área do sistema que o solicitante pudesse solicitar transporte, pois é muito incomodo para que ele venha até nós para fazer esse processo. E com isso nós pudéssemos analisar e aprovar e desaprovar. Mas toda vez que desaprovamos uma solicitação temos que justificar o porquê de não atendermos essa solicitação.”

8. Como seria realizado o acesso ao sistema pelos usuários?

Resposta: “O acesso dos usuários ao sistema tem que ser por meio de senha e *login*, e eles devem ir até a sala de administração para ser cadastrado no sistema.”

9. São necessárias restrições de acesso ao sistema?

Resposta: “O sistema tem que atribuir funções a cada usuário, para que cada um só acesse aquilo que envolvi as suas responsabilidades.”

10. Quais características o sistema deve possuir com relação ao seu funcionamento?

Resposta: “Quero que ele seja fácil de utilizar, quando tiver uma situação que eu não saiba fazer, ele me possa me dar dicas de como resolver. Quero que seja seguro.”

11. O que é importante para a implementação do sistema?

Resposta: “O mais importante no sistema é a geração de relatórios, no que desrespeito aos motoristas, informando todos os dados da viagem que ele participou no mês. Sendo um relatório para cada motorista e um com todos.”

12. Há alguma informação a mais que deve ser colocada?

Resposta: “Não, nesse momento não tenho mais o que falar para vocês, mas no decorrer do projeto vocês podem me mostrar o que vocês já fizeram e com isso vão surgir mais dúvidas e com isso mais coisas podem ser acrescentadas.”

APÊNDICE B - Descrição E Conteúdo Das Tabelas Do Banco De Dados

“O dicionário de dados é o lugar em que, dentre outras coisas, contém informações detalhadas, às vezes chamadas de metadados, com relação aos diversos objetos que são de interesse do próprio sistema.” (DATE, 2003).

Segue abaixo as tabelas do sistema, como pode ser visto na Figura 11, com a descrição e conteúdo de cada uma das tabelas que compõem o modelo de entidade relacionamento do Sistema de Alocação de Transporte:

Quadro 15 - Dicionário de dados – Cargo

Tabela: Cargo					
Descrição: Armazena os tipos de cargos.					
Tabelas Relacionadas: usuário.					
Nome da coluna	Tipo de dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
id_cargo	Int(11)	PK	NN	PK de cargo	AI
Nome	Varchar(45)		NN	Nome do Cargo	

Fonte: O Autor (2014)

Quadro 16 - Dicionário de dados – Confirmação

Tabela: Confirmação					
Descrição: tabela que grava o registro de solicitação analisadas e quem a analisou.					
Tabelas Relacionadas: usuário, solicitação					
Nome da coluna	Tipo de dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
idconfirmação	Int(11)	PK	NN	PK de confirmação	AI
Aprovado	Boolean			Guarda o estado da solicitação	
Aprovador	Int(11)			Guarda o id de quem analisou a solicitação	
Justificativa_neg	Text			Justificativa de desaprovação	
Deletado	Boolean		NN	Estado de deletado ou não.	
Idusuario	Int(11)	FK		FK de usuário	

Fonte: O Autor (2014)

Quadro 17 – Dicionário de dados – Despesas

Tabela: Despesas					
Descrição: Armazena as despesas da viagem.					
Tabelas Relacionadas: viagem.					
Nome da coluna	Tipo de dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
id_despesa	Int(11)	PK	NN	PK de despesas	AI
Valor	Float		NN	Valor da despesa	
Justificativa	Text		NN	Justificativa	
id_viagem	Int(11)	FK	NN	FK de viagem.	

Fonte: O Autor (2014)

Quadro 18 - Dicionário de dados – Estado_vigencia

Tabela: Estado_vigencia.					
Descrição: Armazena o estado da vigência.					
Tabelas Relacionadas: vigência.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idestadovigencia	Int(11)	PK	NN	PK estado vigência	AI
ativo	Varchar(45)		NN	Estado de ativo ou não	

Fonte: O Autor (2014)

Quadro 19 - Dicionário de dados – Função

Tabela: Função.					
Descrição: Armazena as funções dos usuários.					
Tabelas Relacionadas: vigência.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idfunção	Int(11)	PK	NN	PK estado vigência	AI
Nome	Varchar(45)		NN	Nome da função	
Nível	Int(11)		NN	Nível da função	

Fonte: O Autor (2014)

Quadro 20 – Dicionário de dados – Motorista

Tabela: Motorista.					
Descrição: Armazena os dados do motorista.					
Tabelas Relacionadas: repasse, viagem_has_motorista.					
Nome da	Tipo do	Chaves: Primária	Não	Descrição	Auto

coluna	dado	(PK)/Estrangeira(FK)	nulo		incremento
Idmotorista	Int(11)	PK	NN	PK motorista	AI
Nome	Varchar(45)		NN	Nome do motorista	
Disponibilidade	Boolean		NN	Campo utilizado para verificar a disponibilidade do motorista	
Deletado	Boolean		NN	Estado de deletado ou não.	

Fonte: O Autor (2014)

Quadro 21 – Dicionário de dados – Participantes

Tabela: Participantes					
Descrição: Armazena os participantes da viagem.					
Tabelas Relacionadas: usuário, solicitação.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idparticipante	Int(11)	PK	NN	PK participante	AI
Deletado	Boolean		NN	Estado de deletado ou não	
Id_solicitação	Int(11)	FK	NN	FK solicitação	
Id_usuario	Int(11)	FK	NN	FK usuário	

Fonte: O Autor (2014)

Quadro 22 - Dicionário de dados – Repasse

Tabela: Repasses.					
Descrição: Armazena o número de diárias do motorista.					
Tabelas Relacionadas: motorista, viagem.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idrepasso	Int(11)	PK	NN	PK repasse	AI
Data	Date		NN	Data do repasse	
Qtde	Int(11)		NN	Quantidade de diárias	
Repasso	Boolean		NN	Estado do repasse	
Deletado	Boolean		NN	Estado de deletado ou não.	
Idmotorista	Int(11)	FK	NN	FK motorista	
Idviagem	Int(11)	FK	NN	FK viagem	

Fonte: O Autor (2014)

Quadro 23 - Dicionário de dados – Setor

Tabela: Setor.					
Descrição: Armazena o setor do usuário.					
Tabelas Relacionadas: usuário, vigência.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idsetor	Int(11)	PK	NN	PK setor	AI
Nome	Varchar(45)		NN	Nome do setor	

Fonte: O Autor (2014)

Quadro 24 – Dicionário de dados – Solicitação

Tabela: Solicitação					
Descrição: Armazena os dados da solicitação.					
Tabelas Relacionadas: usuário, confirmação, participantes, viagem_has_solicitação.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idsolicitação	Int(11)	PK	NN	PK solicitação	AI
Data	Date		NN	Data que foi realizada a solicitação	
Partida	Date		NN	Data de partida da viagem	
Retorno	Date		NN	Data de retorno da viagem	
Objetivo	Text		NN	Objetivo da viagem	
Justificativa	Text		NN	Justificativa da viagem	
Destino	Varchar(45)		NN	Destino	
Distancia	Int(11)			Distancia até o destino	
Idusuario	Int(11)	FK	NN	FK usuário	
IdConfirmação	Int(11)	FK	NN	FK confirmação	

Fonte: O Autor (2014)

Quadro 25 - Dicionário de dados – Tipo_Usuário

Tabela: Tipo_usuario.					
Descrição: Armazena os tipos de usuário.					
Tabelas Relacionadas: usuário.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idtipo_usuario	Int(11)	PK	NN	PK tipo_usuario	AI

Tipo	Varchar(45)		NN	Tipo de usuário	
------	-------------	--	----	-----------------	--

Fonte: O Autor (2014)

Quadro 26 - Dicionário de dados – Usuário

Tabela: Usuário.					
Descrição: Armazena os dados do usuário.					
Tabelas Relacionadas: cargo, tipo_usuario, setor, confirmação, solicitação, participantes.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idusuario	Int(11)	PK	NN	PK usuário	AI
Nome	Varchar(45)		NN	Nome do usuário	
Siape	Varchar(45)		NN	Siape	
Senha	Varchar(45)		NN	Senha	
E-mail	Varchar(45)		NN	E-mail do usuário	
Telefone	Varchar(45)		NN	Telefone	
Ativo	Boolean		NN	Usuário ativo ou não	
Matricula	Varchar(45)		NN	Matricula	
CPF	Varchar(45)		NN	CPF	
Idcargo	Int(11)	FK	NN	FK cargo	
Idtipo_usuario	Int(11)	FK	NN	FK tipo_usuario	
Idsetor	Int(11)	FK	NN	FK setor	

Fonte: O Autor (2014)

Quadro 27 – Dicionário de dados – Veículo

Tabela: Veículo.					
Descrição: Armazena os dados do veículo.					
Tabelas Relacionadas: viagem_has_veiculo.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idveiculo	Int(11)	PK	NN	PK veiculo.	AI
Placa	Varchar(45)		NN	Placa do veículo.	
Tipo	Varchar(45)		NN	Tipo do veículo.	
Num_pessoas	Int(11)		NN	Número de passageiros	
Disponibilidade	Boolean		NN	Campo utilizado para verificar a disponibilidade do veículo.	
Deletado	Boolean		NN	Estado de deletado ou não.	

Fonte: O Autor (2014)

Quadro 28 – Dicionário de dados – Viagem

Tabela: Viagem.					
Descrição: Armazena os dados da viagem.					
Tabelas Relacionadas: despesas, viagem_has_veiculo, repasse, viagem_has_motorista, viagem_has_solicitação.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idviagem	Int(11)	PK	NN	PK viagem	AI
Data de partida	Date		NN	Partida	
Data de retorno	Date		NN	Retorno	
Destino	Varchar(45)		NN	Destino	
Justificativa	Text		NN	Justificativa	
Deletado	Boolean		NN	Estado de deletado ou não.	

Fonte: O Autor (2014)

Quadro 29 – Dicionário de dados – Viagem_has_motorista

Tabela: Viagem_has_motorista.					
Descrição: Tabela de associação entre viagem e motorista.					
Tabelas Relacionadas: viagem, motorista.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idmotorista	Int(11)	FK	NN	FK motorista.	
Idviagem	Int(11)	FK	NN	FK viagem	

Fonte: O Autor (2014)

Quadro 30 – Dicionário de dados – Viagem_has_solicitação

Tabela: Viagem_has_solicitação.					
Descrição: Tabela de associação entre viagem e solicitação.					
Tabelas Relacionadas: viagem, solicitação.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idsolicitação	Int(11)	FK	NN	FK solicitação.	
Idviagem	Int(11)	FK	NN	FK viagem	

Fonte: O Autor (2014)

Quadro 31 – Dicionário de dados – Viagem_has_veiculo

Tabela: Viagem_has_veiculo.					
Descrição: Tabela de associação entre viagem e veículo.					
Tabelas Relacionadas: viagem, motorista.					
Nome da	Tipo do	Chaves: Primária	Não	Descrição	Auto

coluna	dado	(PK)/Estrangeira(FK)	nulo		incremento
Idveiculo	Int(11)	FK	NN	FK veículo.	
Idviagem	Int(11)	FK	NN	FK viagem	

Fonte: O Autor (2014)

Quadro 32 – Dicionário de dados – Vigência

Tabela: Vigência.					
Descrição: Guarda os dados da vigência do usuário.					
Tabelas Relacionadas: estado_vigencia, função, setor.					
Nome da coluna	Tipo do dado	Chaves: Primária (PK)/Estrangeira(FK)	Não nulo	Descrição	Auto incremento
Idvigencia	Int(11)	PK	NN	PK vigência.	AI
Inicio	Date		NN	Início da Vigência	
Fim	Date		NN	Fim da Vigência	
Num_ato	Varchar(45)		NN	Número do ato de vigência.	
Idsetor	Int(11)	FK	NN	FK setor	
Idfunção	Int(11)	FK	NN	FK função	
Idestadovigencia	Int(11)	FK	NN	FK estado vigência	

Fonte: O Autor (2014)




**TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA
“JOSÉ ALBANO DE MACEDO”**

Identificação do Tipo de Documento

- () Tese
- () Dissertação
- (X) Monografia
- () Artigo

Eu, **Marcelino Mendes da Silva Neto**, autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação **Sistema Web para controle dos processos de alocação de veículos do setor de transporte da Universidade Federal do Piauí – Campus Senador Helvídio Nunes de Barros** de minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos-PI 20 de janeiro de 2015.


Assinatura