UNIVERSIDADE PAULISTA

PROJETO INTEGRADO MULTIDISCIPLINAR

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

**PROJETO INTEGRADO MULTIDISCIPLINAR VII**

**“PROJETO DE DESENVOLVIMENTO DE SISTEMA DE CONTROLE PARA LOCADORA DE VEÍCULOS ”**

BRASIL

2016UNIVERSIDADE PAULISTA

PROJETO INTEGRADO MULTIDISCIPLINAR

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

**PROJETO INTEGRADO MULTIDISCIPLINAR VII**

**“PROJETO DE DESENVOLVIMENTO DE SISTEMA DE CONTROLE PARA LOCADORA DE CARROS ”**

**Nome**: Fernando Muniz da Silva

**RA**:1500197

**Nome**: Vinícius Jorge Cruz Mendes

**RA**:1514681

**Nome**: Fábio Ferreira Damasceno

**RA**:1510684

**Curso:** Análise e Desenvolvimento de Sistemas

4ºSemestre 2016

BRASIL

2016**Sumário**

[RESUMO](#_gjdgxs)

[ABSTRACT](#_30j0zll)

[1. INTRODUÇÃO](#_1fob9te)

[2. DESENVOLVIMENTO](#_3znysh7)

[2.1 DESENVOLVIMENTO DE SISTEMAS DE CONTROLE DA LOCADORA DE CARROS](#_2et92p0)

[2.1.1. ARQUITETURA MVC](#_tyjcwt)

[2.1.2 MODELO ENTIDADE RELACIONAMENTO](#_3dy6vkm)

[2.1.3 UML (UNIFIED MODELING LANGUAGE)](#_1t3h5sf)

[2.2 DIAGRAMA DE CLASSES](#_4d34og8)

[OBJETOS NECESSÁRIOS À IMPLEMENTAÇÃO](#_2s8eyo1)

[2.2.2 DIAGRAMA DE CLASSES DA ESTRUTURA DE APLICAÇÃO A SER SEGUIDO](#_17dp8vu)

[2.2.3 IDENTIFICAÇÃO DE CASO DE USO](#_3rdcrjn)

[2.2.4 DIAGRAMA DE SEQUÊNCIA](#_26in1rg)

[2.2.5 DIAGRAMA DE CASO USO ÚNICO](#_lnxbz9)

[2.2.6 DIAGRAMA DE CASO DE USO SEQUÊNCIA DO FLUXO BÁSICO](#_35nkun2)

[2.2.7 DIAGRAMA DE CASO DE USO PARA FLUXO ALTERNATIVO](#_1ksv4uv)

[2.2.8 ATORES](#_44sinio)

[2.2.9 DIAGRAMA DE ATIVIDADES](#_z337ya)

[2.2.10 DIAGRAMA DE DISTRIBUIÇÃO](#_3j2qqm3)

[2.2.11 DIAGRAMA DE IMPLANTAÇÃO](#_1y810tw)

[3. CONCLUSÃO](#_4i7ojhp)

[4. REFERÊNCIAS](#_2xcytpi)

[5. APÊNDICE](#_1ci93xb)

# RESUMO

Este projeto tem pretende apresentar como proposta o desenvolvimento de um sistema integrado

**Palavras-chave:** projeto, desenvolvimento de sistema, controle, gestão empresarial, veiculos, locadora.

# ABSTRACT

**Keywords: design, system development, control, business management, vehicles, rental.**

1. INTRODUÇÃO

Este projeto tem como objetivo criar um sistema para controlar a locação de veículos em uma empresa especializada, controlar a devolução e o pagamento dessas locações. Deste modo, a primeira etapa do projeto já analítica, já realizada. Deste modo o foco do presente projeto estará na segunda fase, o designer, sendo assim serão apresentadas as seguintes etapas:

1. Diagrama Entidade-Relacionamento (banco de dados);
2. Desenho da arquitetura de referência utilizando MVC;
3. Para cada Caso de Uso, desenvolver o diagrama de classe de implementação e o diagrama de sequência de implementação;
4. Diagrama de atividades do método calcular Multa;
5. Diagrama de distribuição com os requisitos para implantação do sistema.

A empresa em questão é de propriedade dos irmãos empresários que resolveram criar uma empresa no ramo de locação de veículos sendo assim, muito planejamento e decisões tomadas para viabilizar o negócio, incluindo a necessário desenvolver um sistema para controlar as locações dos veículos.

Buscaram uma empresa especializada em desenvolvimento de softwares para criar um sistema que atendesse sua demanda.

Após um mês efetivo de intenso trabalho, nessa etapa a fase de análise já estava pronta. Entretanto, o designer do projeto teve um imprevisto e por isso fez uma viagem para atender um cliente estrangeiro. Para que o projeto continuasse, nossa equipe foi contratada para o realizar o desenvolvimento do sistema a partir da análise já realizada e das informações que nos foram passadas pelo arquiteto e iremos conduzir os trabalhos até seu retorno, logo ficamos responsáveis pelo designer do projeto.

2. DESENVOLVIMENTO

## 2.1 DESENVOLVIMENTO DE SISTEMAS DE CONTROLE DA LOCADORA DE VEÍCULOS

### 2.1.1. ARQUITETURA MVC

O MVC (model – view - controller) é um padrão de arquitetura de software permite fazer a separação em camadas dos elementos de um sistema, permitindo alterações localizadas. É a base para gerenciamento em sistemas web.

A arquitetura do sistema é estruturada em três componentes lógico que fazem a interação entre si. São estes:

* Modelo: gerencia os dados e as operações que estão associadas a esses dados.
* Visão: define e gerencia como os dados são apresentados aos usuários.
* Controlador: gerencia a interação do usuário, passando essas interações para a visão e modelo.

A arquitetura MVC é atualmente utilizada em diversos frameworks de várias linguagens devido às vantagens que oferece, tais como reuso do código e maior facilidade de manutenção do software. Observa-se que as camadas View (Visão) e Model (Modelo) comunicam-se através de um Controller (Controlador) que recebe os dados de entrada do usuário através da View e envia os dados para o Model onde os dados são processados e retornados para a View. A camada View é a interface que o usuário interage.. O Model representa a camada de persistência de dados, essa camada é responsável por receber os dados vindos da View, processá-los e enviá-los de volta a View. O Controller interpreta requisições vindas do usuário através da View e faz a chamada do Model necessário para interpretar a requisição. Por fim, MVC é um padrão de arquitetura de software que separa a informação (e as suas regras de negócio) da interface com a qual o usuário interage. É uma forma de estruturar seu projeto/aplicação de forma que a interface de interação (view) esteja separada do controle da informação em si (models), separação essa que é intermediada por uma outra camada controladora (controllers).

### 2.1.2 MODELO ENTIDADE RELACIONAMENTO

O MER – Modelo Entidade-Relacionamento – É um modelo conceitual desenvolvido e proposto pelo Dr. Peter Pin-Shan Chen em 1976 e é focado nos conceitos de: entidades, atributos e relacionamentos entre as entidades. Este tipo de modelagem de bancos de dados tornou-se mundialmente famoso e possui grande reconhecimento parte da Engenharia de Software, tornando os objetos observados no modelo de negócios em entidades. Como o DER – Diagrama Entidade Relacionamento

O DER pode ser entendido como uma forma lógica, gráfica e intuitiva de visualizar as concepções que envolvem a modelagem dos dados de acordo com o modelo de dados Entidade-Relacionamento. Neste diagrama, podemos ver claramente o conjunto de entidades, o conjunto de atributos das entidades e os relacionamentos existentes alinhados de acordo com o entendimento do modelo de negócios proposto.

A estrutura DER é composta por figuras geométricas, textos e números. As figuras representam as entidades e os relacionamentos, sendo o primeiro representado por quadrados ou retângulos e o segundo por losangos. Os textos descrevem os nomes dos atributos que uma entidade possui e os números se referem à cardinalidade mapeada entre as entidades.

### 2.1.3 UML (UNIFIED MODELING LANGUAGE)

A UML (Unified Modeling Language) – Linguagem de Modelagem Unificada pode ser entendida como uma linguagem de modelagem que segue uma série de padrões aceitos por uma comunidade de engenheiros de software por todo o planeta. Seu início se deu por: Jim Rumbaugh, Grady Booch e Ivar Jacobson na década de 90. A UML é baseada nos conceitos de Orientação a Objetos e que pode ser observado em seus diagramas, que são expressos de forma lógica.

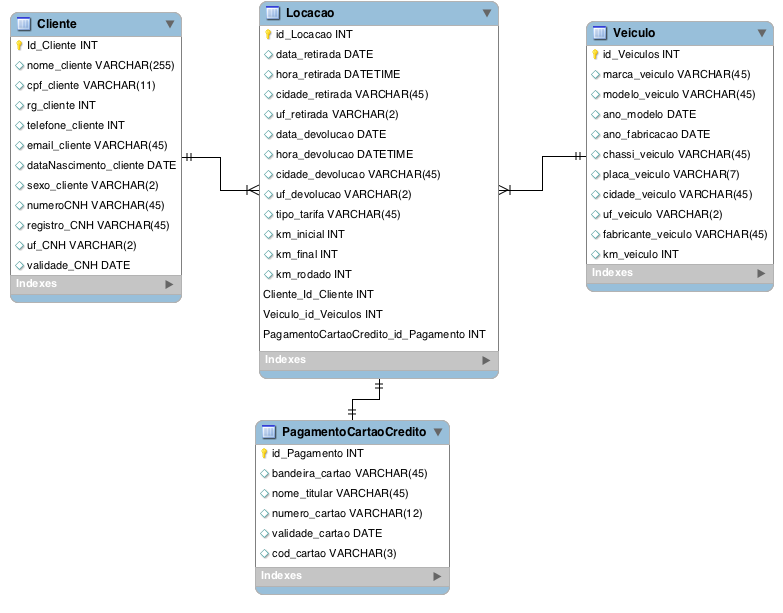
Os diagramas são divididos em dois grupos: um para a modelagem da estrutura de um sistema e um para a modelagem do comportamento. Os diagramas de estrutura são usados para representar os dados e relacionamentos estáticos que compõe um sistema. O diagrama de comportamento fornece a representação dos relacionamentos dinâmicos entre os objetos que representam o sistema. Quando o sistema é desenvolvido, os diagramas passam a incluir detalhes que levam a geração de código.

A UML pode ser utilizada para especificar sistemas de qualquer natureza, ser usada em todas as fases do processo de desenvolvimento, objetivando dar suporte visual expressivo para que os usuários possam desenvolver e trocar modelos entre si.

1. Tabelas para atributos multivalorados
2. Análise da situação em relação 1.1; 1.n e n..n
3. Relações de agregação
4. Relações de herança
5. Arquitetura de referência (modelo estático)

## 

## 3.2 DIAGRAMA ENTIDADE-RELACIONAMENTO LOCADORA



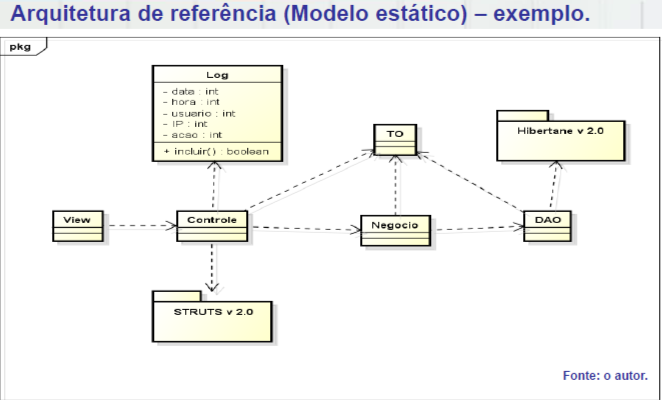
## DIAGRAMA DE CLASSES

Diagrama de classes é considerado por muitos autores como o mais importante e o mais utilizado diagrama da UML. Sua principal característica é permitir a visualização das classes que irão compor o sistema com seus respectivos atributos e métodos, bem como em demonstrar como as classes do sistema se relacionam, se complementam e transmitem informações entre si. O diagrama de classes é composto por suas classes e pelas associações existentes entre elas, ou seja, os relacionamentos entre as classes.

Para o desenvolvimento deste diagrama utilizar-se-á a arquitetura em camadas (explicar rapidamente o que é) qual o objetivo e vantagens

### OBJETOS NECESSÁRIOS À IMPLEMENTAÇÃO

### 2.2.2 DIAGRAMA DE CLASSES DA ESTRUTURA DE APLICAÇÃO A SER SEGUIDO



### 2.2.3 IDENTIFICAÇÃO DE CASO DE USO

1.Para cada caso de uso, monte um diagrama de classes de implementação e diagramas de sequência para o fluxo básico e fluxos alternativos relevantes.

2.Para construir o diagrama de classes de implementação, observe o modelo estático de arquitetura e inclua as classes de domínio envolvidas.

3.Elabore o diagrama de sequência tendo em mente a ordem em quem os objetos são associados para cada fluxo do caso de uso.

### 2.2.4 DIAGRAMA DE SEQUÊNCIA

O diagrama de sequência equivale em modelar a interação entre os objetos do sistema. Permite identificar postos específicos de execução do sistema. A visualização do diagrama consiste identificar os objetos nas linhas verticais. Os relacionamentos são através de setas e o decorrer do tempo visualização de cima para baixo.

### 2.2.5 DIAGRAMA DE CASO USO ÚNICO

CLIENTE

LOCADORA

SERASA

**LOCAÇÃO**

**VEÍCULO**

**SERASA**

### 2.2.6 DIAGRAMA DE CASO DE USO SEQUÊNCIA DO FLUXO BÁSICO

### 

### 2.2.7 DIAGRAMA DE CASO DE USO PARA FLUXO ALTERNATIVO

Nesta descrição não haverá descrição em caso de dados inválidos, trataremos apenas de casos relevantes, nesse sentido a pesquisa no SPC (Serviço de Proteção ao Crédito)

### 2.2.8 ATORES

Atores são os agentes externos ao sistema que executam uma determinada ação e que esperam algum resultado, ou seja, interagem diretamente com o sistema a partir dos casos de uso. Neste nosso sistema encontramos os seguintes funções e definições:

**Cliente:**

**Locadora (vendedor):**

**Serasa:**

### 

### 2.2.9 DIAGRAMA DE ATIVIDADES

### 

O Diagrama de atividade é um diagrama referenciado pela Linguagem de Modelagem Unificada (UML), e representa os fluxos conduzidos por processamentos. É essencialmente um gráfico de fluxo, mostrando o fluxo de controle de uma atividade para outra. Comumente isso envolve a modelagem das etapas sequenciais em um processo computacional.

Os diagramas de atividade são importantes para a modelagem de aspectos dinâmicos de um sistema ou um fluxograma, e também para a construção de sistemas executáveis por meio de engenharia de produção reversa. Um diagrama de atividade é um diagrama de estados, em que são representados os estados de uma atividade, em vez dos estados de um objeto. Ao contrário dos diagramas de estados que são orientados a eventos, diagramas de atividade são orientados a fluxos de controle.

**Para a elaboração dos diagramas de atividades e distribuição:**

**1.O diagrama de atividades é um fluxograma do método, ou seja, é o algoritmo a ser seguido para a implementação.**

### 2.2.10 DIAGRAMA DE DISTRIBUIÇÃO

### 2.2.11 DIAGRAMA DE IMPLANTAÇÃO

O diagrama de distribuição ou implantação representa a configuração e a arquitetura do sistema em que estarão ligados os respectivos componentes. O diagrama de distribuição, ou de implantação, mostra como os componentes são configurados para execução, em “nós” de processamento. Um diagrama de implantação representa a topologia física do sistema e, opcionalmente, os componentes que são executados nessa topologia. Pode-se dizer que esse diagrama apresenta um mapeamento entre os componentes de software e o hardware utilizado pelo sistema.

**2.O diagrama de distribuição mostra a arquitetura física da aplicação, mostrando o que deve ser instalado em cada camada física**.

QUAL SOFTWARE ROTA E QUAL SISTEMA PRECISA

MAQUINA CLIENTE (INTERNET EXPLORER VERSÃO 10)

MAQUINA SERVIDOR

# 3. CONCLUSÃO

# 4. REFERÊNCIAS

**ABTN**, ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR14724**: informação e documentação: *trabalhos acadêmicos: apresentação*. Rio de janeiro, 2011.

**ECO**, Humberto. Como se faz uma tese. 12ª ed. SP: Perspectiva, 1995

**FURASTÉ**, Pedro Augusto. *Normas Técnicas para o Trabalho Científico: Explicitação das Normas da ABNT*. – 16. Ed. – Porto Alegre: Dáctilo Plus, 2012.

**KRIESER**. MDA – *Model Driven Architecture & UML – Unified Modeling Language*. Acesso em 20.09.2016

**LARMAN**, C. *Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e ao processo unificado*. 2. ed. Porto Alegre: Bookman, 2007.

**MALDONADO**, J.C. S.C.P.F. **FABBRI**, “*Teste de Software*”*.* In: *Qualidade de Software: Teoria e Prática*. Eds. A.R.C. Rocha, J.C. Maldonado, K. Weber, Prentice Hall, 2001.

**MORGADO**, Gisele P. *Rapdis: um processo mda para desenvolvimento de sistema de informação.*2006. Artigo Simpósio Brasileiro de Sistema de Informação. Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2007.

**Pressman**, R.S *Engenharia de Software*, Rio de Janeiro: McGraw Hill, 5ª edição, 2002.

**SOMMERVILLE**, *Engenharia de Software*, São Paulo: Addison-Wesley, 6ª edição, 2003.

# 5. APÊNDICE