Ministério da Educação
Universidade Federal do Paraná
Setor Litoral

UNIVERSIDADE FEDERAL DO PARANÁ

SETOR LITORAL

PROCESSO DE ANÁLISE E DESENVOLVIMENTO DE SOFTWARE PARA PEQUENAS EMPRESAS DE OFICINA MECÂNICA

LUKAS LOPES MUSSOI

MATINHOS

2013

Universidade Federal do Paraná – Setor Litoral
Rua Jaguariaíva, 512 - Matinhos/ PR – CEP 83260-000 – Tel. (41) 3511-8300
www.litoral.ufpr.br

UNIVERSIDADE FEDERAL DO PARANÁ
SETOR LITORAL

LUKAS LOPES MUSSOI

PROCESSO DE ANÁLISE E DESENVOLVIMENTO DE SOFTWARE PARA PEQUENAS EMPRESAS DE OFICINA MECÂNICA

Trabalho de conclusão de curso apresentado como requisito parcial para a obtenção do título de Bacharel em Informática e Cidadania pela Universidade Federal do Paraná – Setor Litoral.

Orientador: Professor Neilor Fermino Camargo

MATINHOS

2013

BANCA EXAMINADORA

Titulação, Nome (Orientador):
Instituição

Titulação, Nome
Instituição

Titulação, Nome
Instituição

SUMÁRIO

AGRADECIMENTOS

Agradecer primeiramente a Deus, por me fornecer sabedoria, força e perseverança durante toda a minha trajetória.

A minha esposa Dayana que sempre me apoiou e me deu forças para continuar, a minha mãe Sandra, meu pai Flávio e meus irmãos Luciana, Juliana e Davi, por sempre me darem força e apoio em minhas decisões, e me colocarem em suas orações.

Aos meus colegas de classe, que sempre se mantiveram unidos e permitiram que cada dia de aula se tornasse cada vez um dia mais divertido, e que agora os tenho como grandes amigos.

Agradeço aos meus professores; professor Neilor Camargo que o tenho como um grande mestre e amigo, o admiro por suas conquistas e conhecimento, ao professor Emerson Joucoski, que permitiu que eu me integrasse melhor com a universidade e compartilhou muitos conhecimentos, também o tenho como um amigo, professora Silma Batezzati que é forte adepta de software livre assim como eu, e se tornou uma grande amiga, professora Luciana, e a todos os outros professores, que tenho certeza que deram o máximo de si.

RESUMO

Este é um trabalho, em que é exibido o processo de analise e projeto de um sistema de software para atender pequenas empresas prestadoras de serviço do tipo oficina mecânica.
Tem por objetivo descrever todos os passos utilizados para o desenvolvimento de um sistema de software, passando por todas as etapas de analise projeto e desenvolvimento.
E por fim, exibir todo o resultado do processo de analise de negocioso desenvolvimento dos diagramas, projeto codificação e o resultado do sistema pronto e finalizado.

Palavras-chaves: Analise – Projeto - Desenvolvimento

LISTA DE SIGLAS E ABREVIATURAS

CGI Consiste numa importante tecnologia que permite gerar páginas dinâmicas, permitindo a um navegador passar parâmetros para um programa alojado num servidor web.[1]

DRY é um conceito de programação de computadores o qual propõe que cada porção de conhecimento em um sistema deve possuir uma representação única, de autoridade e livre de ambiguidades em todo o sistema. [2]

GPL General Public License, é a designação da licença para software livre idealizada por Richard Matthew Stallman em 1989, no âmbito do projeto GNU da Free Software Foundation (FSF).[3]

ICONIX pode ser considerada uma metodologia pura, prática e simples, mas também poderosa e com um componente de análise e representação dos problemas sólido e eficaz.[4]

JAD Joint Application Design é uma metodologia criada pela IBM do Canadá em 1977 utilizada para moderação de discussões de brainstorming acelerando e consolidando o desenvolvimento de aplicações de Sistemas de Informação. [5]

MVC Model-view-controller (MVC), em português modelo-visão-controlador, é um modelo de arquitetura de software que separa a representação da informação da interação do usuário.[6]

MYSQL é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL como interface. [7]

RUP abreviação de Rational Unified Process (ou Processo Unificado Rational), é um processo proprietário de Engenharia de software criado pela Rational Software Corporation.[8]

SGDB Sistema de Gerenciamento de Banco de Dados (SGBD) é o conjunto de programas de computador responsáveis pelo gerenciamento de uma base de dados.[9]

SQL Structured Query Language, é a linguagem de pesquisa declarativa padrão para banco de dados relacional.[10]

XP Programação extrema (do inglês eXtreme Programming), ou simplesmente XP, é uma metodologia ágil para equipes pequenas e médias e que irão desenvolver software com requisitos vagos e em constante mudança.[11]

---

http://en.wikipedia.org/wiki/CGI.pm [1]
http://en.wikipedia.org/wiki/Dont_repeat_yourself [2]
http://www.gnu.org/licenses/gpl-faq.pt-br.html#WhatDoesGPLStandFor [3]
http://pt.scribd.com/doc/56081208/Artigo-sobre-ICONIX [4]
http://pt.wikipedia.org/wiki/Joint_application_development [5]
http://www.caelum.com.br/apostila-java-web/mvc-model-view-controller/ [6]
http://www.w3schools.com/php/php_mysql_intro.asp [7]
http://www.boente.eti.br/publica/seget2008rup.pdf [8]
http://espacoinfo.net/o-que-e-sgbd-bd-ii/ [9]
http://www.w3schools.com/sql/ [10]
http://desenvolvimentoagil.com.br/xp/ [11]

Introdução

Este trabalho está fundamentado em uma pesquisa de campo entre as empresas prestadoras de serviços como: oficinas mecânicas de automóveis e oficinas mecânicas de motocicletas, que ficam localizadas no litoral do paraná, mais especificamente nas cidades de Matinhos, Guaratuba e Pontal do Paraná. Observou-se uma precária infra-estrutura de software de gerenciamento e controle destas micro e pequenas empresas, e um grande interesse por parte dos empresários em possuir um sistema de software que auxilie a gerenciar melhor as micro e pequenas empresas prestadoras de serviços.

Observou-se também, que as empresas que possuem software instalados em seus computadores, estão trabalhando com softwares muito antigos e precários que foram desenvolvidos por empresas de outras cidades ou não possuem mais suporte algum, pois são desenvolvidos em linguagens de programação antigas, que não permitem mais o suporte e a manutenção dos mesmos.

Objetivo

O objetivo deste trabalho é o de conhecer como procede o funcionamento das micro e pequenas empresas prestadoras de serviços mecânicos automotivas. Iniciar um processo de análise de sistemas para recolher os requisitos necessários, e desenvolver um software para controle de uma oficina mecânica.

Pretende-ser, desenvolver um projeto de análise e desenvolvimento de sistemas, baseando nos relatos dos empresários e funcionários das empresas de prestação de serviços, com o intuito de criar um software que realmente seja favorável aos pequenos empresários, e que irá satisfazer suas necessidades.

Também pretende-se que este software de gestão e controle de oficina mecânica que seja de fácil usabilidade para que não possua tanta resistência a automação por parte dos funcionários e empresários, que seja escrito em linguagem de programação de alto nível para melhorar a manutenção e possíveis alterações que venham a proceder durante o processo. E implantar este software em uma pequena empresa e verificar se ele está de acordo com as necessidades dos empresários, e quais as possíveis melhorias a serem efetuadas.

CAPÍTULO I

1.0. Análise Inicial

Para iniciar o desenvolvimento de um software, que tem o objetivo de gerenciar uma pequena empresa que trabalhe no ramo de oficina mecânica, foi necessário analisar alguns fatores deste ramo de atividade, e começar a entender qual é o funcionamento de uma oficina mecânica, quais são seus problemas e como seria possível criar algo que de alguma maneira fosse ajudar os colaboradores que participam na mesma.

A importância deste procedimento de analise inicial, é fundamental apara definir vários fatores do sistema, como quem vai operar o sistema, em que situações o software vai operar, quais são as vantagens de utilizar um software para aquela pequena empresa.

Ao fazer uma análise inicial, em uma conversa informal com o pequeno empresário da empresa de Oficina Mecânica, foi observado que sua pequena empresa, não possuía um sistema de trabalho.

1.1. O sistema de trabalho

Um sistema de trabalho pode existir de qualquer forma, seja ele com planilhas eletrônicas, cadernos de controle de caixa ou papeis organizados em armários, o sistema de trabalho não possui dependência de um software.

Um software é uma ferramenta que deve ser desenvolvida encima do sistema de trabalho já existente na empresa, e deve servir apenas como uma ferramenta que organize, automatize tarefas repetitivas e ajude a organizar a empresa de maneira geral.

Esta descoberta foi de grande importância, para o desenvolvimento, pois quando a empresa não possui um sistema de trabalho, o empresário cria uma ilusão de que ao colocar um software de computador instalado nos computadores da empresa, irá resolver os problemas organizacionais da empresa, e foi exatamente isso o que aconteceu com o pequeno empresario de onde estava sendo desenvolvido o projeto, ele acreditava que o software iria resolver todos os problemas de organização da empresa. Depois de uma longa reunião com o empresário, ele começou a entender a diferença entre sistema de trabalho e software, com isso ele começou a organizar a empresa e criar um sistema de trabalho.

Este sistema de trabalho foi criado por ele e pelos colaboradores da empresa, e em aproximadamente duas semanas a pequena empresa já possuía uma pequena rotina de trabalho, ou seja um sistema. Então depois de aproximadamente duas semanas, da rotina de trabalho aplicada na empresa, foi marcado mais uma reunião com o empresário e os seus colaboradores para iniciar um procedimento de analise de sistemas, baseado no sistema de trabalho criado por eles.

Em geral as pessoas tem um certo receio de mudanças em suas rotinas, e principalmente pessoas não gostam de ser controladas. Como elas não possuíam um sistema de trabalho, ao desenvolver o software com um sistema de trabalho retirado de outra empresa, estas pessoas que iriam operar o software não estariam acostumadas com o sistema de trabalho, e teriam uma tendencia maior a rejeitar o software e voltar a operar de forma desordenada. (Juliana Veiga Mendes, 2002) [4]

CAPÍTULO II

Metodologia

Para esta etapa, foram estudado três método de desenvolvimento de sistemas, cada um deles possuem algumas características distintas, estas características são levadas em consideração na escolha do processo de analise e desenvolvimento de sistema, abaixo analisa-se cada um destes três métodos e fazer uma analise de qual deles é melhor para ser utilizado.

Hoje existem algumas metodologias de desenvolvimento que já são aplicadas no método tradicional, como a RUP (Rational Unified Process), esta é uma metodologia, criada pela Rational Software Corporation e adquirida pela IBM , a metodologia de desenvolvimento conhecida como XP (Extreme Programming), e a ICONIX, que não é tão burocrático como a RUP ou radical como a XP (Extreme Programming) é uma metodologia mais pura e simples, porém não deixa de ser uma solução um tanto quanto poderosa. [5]

[4] http://www.scielo.br/pdf/gp/v9n3/14570.pdf

[5] http://www.ebah.com.br/content/ABAAAf8cUAE/comparacao-entre-metodologias-rup-xp

2.1. Metodologia RUP

A metodologia RUP usa uma abordagem da orientação a objetos, e sua documentação é feita utilizando as notações da UML (Unified Modeling Language), este método é considerado pesado, e é aplicado preferencialmente em grandes equipes de desenvolvimento, que necessitam desenvolvedor grandes projetos.

A pesar da RUP ser uma metodologia muito flexível, permite a sua utilização em projetos grandes ou pequenos, ela é mais eficaz se possuir equipe de desenvolvimento, porém este projeto não dispõe de uma equipe de desenvolvimento.[6]

2.2. Metodologia Extreme Programming (XP)

Outra metodologia existente é a XP (Extreme Programming), esta metodologia se demonstra um tanto quanto mais adequada para este projeto de desenvolvimento de sistemas, pois possui algumas características, que permite melhor flexibilidade durante o processo de desenvolvimento.

A XP é uma metodologia ágil para equipes pequenas e médias e que irão desenvolver software com requisitos vagos e em constante mudança. para isso, adota a estratégia de constante acompanhamento e realização de vários pequenos ajustes durante o desenvolvimento de software.[7]

3.3. Metodologia ICONIX

A metodologia ICONIX é uma metodologia considerada pura, prática e simples, porém uma metodologia poderosa, ela possui um componente de análise e representação dos problemas de maneira sólida e eficaz.

A metodologia ICONIX é caracterizada como um processo de desenvolvimento de software utilizado e desenvolvido pela ICONIX Software Engineering.

Veja mais em:

[6] http://www.infoescola.com/engenharia-de-software/rup/

[7] http://desenvolvimentoagil.com.br/xp/

Uma das características da ICONIX, é a de ser um processo não burocrático como a RUP, pois não gera tanta documentação. Apesar de ser considerado um processo simples como o XP (Extreme Programming), ele não deixa a desejar na Análise de Design, e possui um poderoso processo de análise de software.

ICONIX utiliza para a sua modelagem a linguagem UML, e possui uma característica considerada exclusiva e conhecida como "Rastreabilidade dos Requisitos"(Traceability of Requirements), esta característica permite de forma obrigatória ao utilizar seus mecanismos, seja necessário verificar em todas as fazes, se os requisitos do desenvolvimento do software estão sendo atendidos.

Esta metodologia é considerada flexível e aberta, caso seja necessário utilizar qualquer outro recurso da UML para complementar os recursos utilizados nas fazes do ICONIX, não é considerado um problema. [8]

2.4. Definindo a Metodologia do Projeto

Ao analisar estas três metodologias, foi concluído que, as três metodologias são de excelentes qualidade, porém para o projeto apresentado, a que mais se enquadra é a ICONIX.

O projeto é um software de pequeno porte para pequena empresa, pelo fato de não possuir uma equipe de desenvolvimento, a metodologia ICONIX se torna um tanto quanto prática e mais flexível para se aplicar em um projeto de software com estas características.

3.5. Metodologia de JAD (Joint Application Design)

Foram utilizados alguns métodos de analise, o método utilizado para este processo da analise é conhecido como JAD (Joint Application Design) ele consiste em uma reunião, guiada por um líder, cujo o seu objetivo é de permitir o desenvolvimento do sistema com auxilio dos usuários e analistas juntos.

Então foi concebido com uma reunião com todos os colaboradores da empresa. Esta reunião ocorreu em uma sala com um quadro branco, e nela foram esboçadas as ideias dos integrantes da empresa.[9]

---

[8] http://www.guj.com.br/content/articles/patterns/iconix_guj.pdf

[9] http://engenhariadesoftware.info/downloads/JAD.ppt

Este método tem como objetivo, estimular a criatividade das pessoas que participam da empresa, fazendo com que elas ajudem no procedimento de analise com ideias que possam ser implementadas no software melhorando os recursos do mesmo.

CAPÍTULO III - Modelo de Negocio

3.1. Diagrama de Disciplina

Para efetuar todo o processo de análise e desenvolvimento do sistema de software, foi utilizado o diagrama de disciplina, também conhecido como diagrama de atividades, como visto na figura 01.

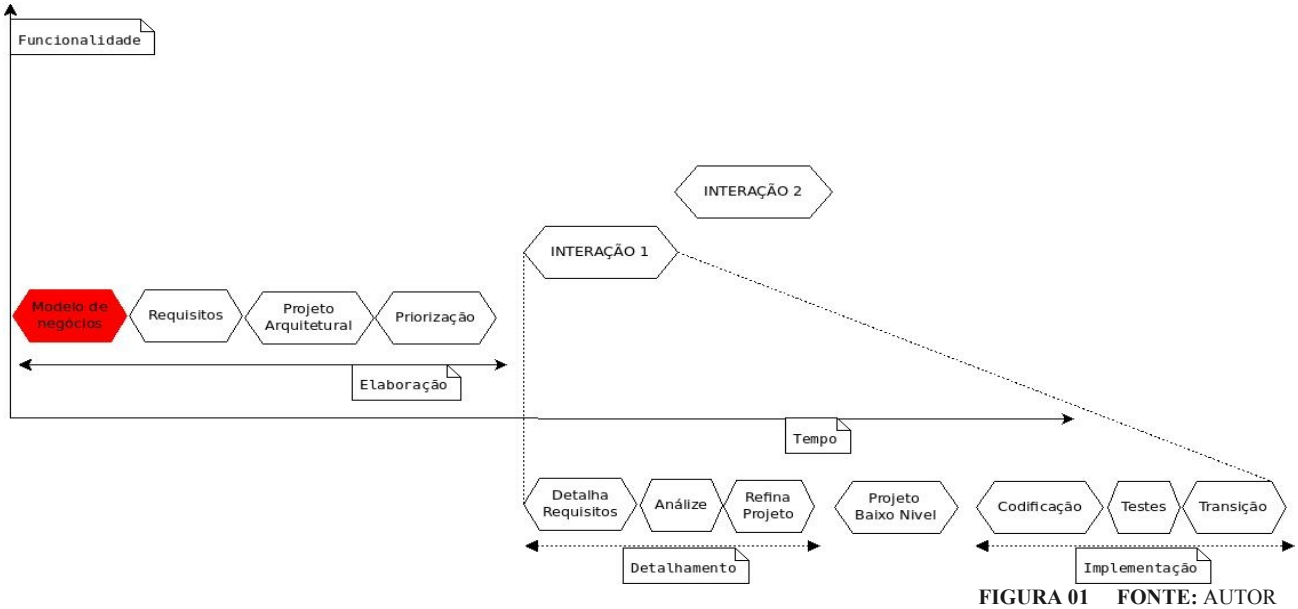


FIGURA 01 **FONTE:** AUTOR

Auxilia tanto o desenvolvedor, quanto o analista de sistemas, e seguir todos os passos necessários para o processo de desenvolvimento, dificultando assim a fuga do foco de trabalho.
Adaptado de POMPILHOS 2002

3.2. Análise inicial

Ao analisar o modelo de negocio, foram citadas várias ideias, algumas muito interessantes e que poderiam ser implementadas no software, outras nem tanto, porém o importante neste procedimento de análise, não é apenas formal e sim social , pois permite que os integrantes da empresa e os desenvolvedores do software se conheçam e criem um determinado vínculo, que irá ajudar a fazer com que o desenvolvedor crie algo mais próximo do que a empresa está precisando.

Outro fator importante nesta etapa da análise inicial, é a questão colaborativa, ao ajudar com ideias e dicas, os colaboradores da empresa ficam mais engajados no processo de desenvolvimento do software, criando uma expectativa positiva do mesmo.

Após esta primeira reunião entre os colaboradores e o empresario, foi efetuada a pesquisa em loco, onde eu participei como espectador da empresa por 3 dias consecutivos, analisando todo o comportamento dos colaboradores e dos clientes. Foram feitas algumas entrevistas com alguns clientes, perguntando a eles o que eles gostariam que a empresa oferecesse para melhorar a qualidade dos serviços prestados, visto que o ramo de atividade era de prestação de serviços de manutenção de automóvel.

Esta etapa foi bem interessante pois foi possível perceber e analisar a empresa do ponto de vista do cliente, e discutir com ele sobre algumas das melhorias que a empresa estava disposta a fazer, então foi feito uma descoberta interessante para o processo da análise e desenvolvimento, o cliente não gostou de algumas das melhorias que o empresário estava querendo.

Então ao anotar este tipo de observação, fomos entrevistar outros clientes, e perguntar a eles se aquela mudança no funcionamento da empresa, seria benéfico para ele ou não, e surpreendentemente de 10 clientes que foram consultados: 8 não gostaram da nova mudança.

Depois de fazer esta análise com os clientes da empresa, convocou-se mais uma pequena reunião, desta vez apenas com o empresário, pois os colaboradores não poderiam abandonar seus cargos no momento. Ao discutir sobre implementar ou não a nova mudança no método de trabalho da empresa com o empresário, ele optou por não acrescentar o novo recurso ao sistema.

Nesta analise foi possível observar várias falhas no funcionamento da empresa, como o atraso na entrega do orçamento para o cliente, a confusão dos funcionários a passar o valor, e definir quais peças de qual carro teriam que ser substituídas, entre outras falhas.

Depois de uma semana entre estes dois tipos de análise, foi marcado uma terceira reunião, novamente esta reunião aconteceu em uma sala com todos os colaboradores e o empresário, e foi desenhado um esboço do que seria o software no quadro branco.

Na reunião foram definidos alguns detalhes importantes para o inicio do projeto, entre elas estavam a questão do software que deveria funcionar em tablet, pois os mecânicos utilizariam o tablet para descrever qual seriam as peças do veiculo e montar o orçamento.

3.3. Cenário do Sistema

O cliente entra na empresa com o veiculo com defeito, seja ele guinchado ou não, em um primeiro momento o cliente deve se dirigir a secretária para fazer um cadastro, neste cadastro será necessário obter os seguintes dados: Nome, Sobrenome, Telefone, Celular, SEXO, CPF , RG, Rua, Numero, Bairro, e Cidade. Após efetuar o cadastro o cliente é levado até um dos mecênicos que atendem a oficina Mecânica, em conversa com o mecânico, o cliente informa sobre os defeitos do veiculo, assim como também informa sobre os serviços adicionais que o mecânico deve fazer, o mecânico obtêm todas estas informações. Então com o auxilio de um tablet, ele abre uma ordem de serviço para aquele cliente que já havia se cadastrado anteriormente. No momento em que o mecânico está conversando com o cliente, o mecânico já decide se a oficina mecânica possui a disponibilidade dos serviços, que o cliente está solicitando, ficando em sua responsabilidade abrir ou não a ordem de serviço para cada cliente.

Após a ordem de serviço devidamente aberta, o veiculo é levado a uma garagem, onde fica aguardando para ser consertado pelo mecânico que atendeu o cliente e abriu a ordem de serviço, isto evita a falha no entendimento, devido a existirem termos técnicos, e no histórico que o cliente geralmente transmite para o mecânico ao deixar um veiculo para a manutenção.

A ordem de serviço deve possuir os seguintes campos: Data de entrada, data de saída, status, defeito, observações e informações sobre o pagamento, e campo para adicionar quantidade de serviços efetuados, o nome do serviço e valores, assim como efetuar a soma dos mesmos, e calcular desconto e impostos.

Após o Mecânico identificar o defeito do veiculo, ele volta a ordem de serviço o diagnóstico as peças e o valor da mão de obra, alterando o status da ordem de serviço para diagnóstico.

Em uma tela do sistema, a secretária visualiza as ordens de serviços que estão em andamento e as com diagnóstico pronto, então ela abre esta ordem de serviço e tenta entrar em contato com o cliente, por algum meio de comunicação, seja ele e-mail, telefone ou pessoalmente, informa para o cliente o diagnóstico do defeito e as peças e o valor das peças.

Se o cliente autorizar o serviço, a secretária altera o status da ordem de serviço para autorizado, e então o Mecânico consegue visualizar em seu tablet, em uma tela do sistema, qual a ordem de serviço e consequentemente qual o veiculo que foi autorizado fazer o serviço. O Mecânico efetua o serviço e altera o status da ordem de serviço para Finalizada. A Secretária visualiza em sua tela, a ordem de serviço com o status de Finalizada, e imediatamente liga para o cliente solicitando que ele venha buscar seu veiculo o mais rápido possível, caso o cliente demore para buscar o veiculo será cobrado estadia de 30 reais por dia no patio da oficina mecânica.

CAPÍTULO IV – Analise de Desenvolvimento

4.1. Levantamento de Requisitos

Breve descrição do Sistema

O objetivo do projeto é de criar um sistema para controle de ordem de serviços para uma empresa prestadora de serviços de manutenção automotiva, dando sequencia no diagrama de disciplina, visualizado na figura 02.

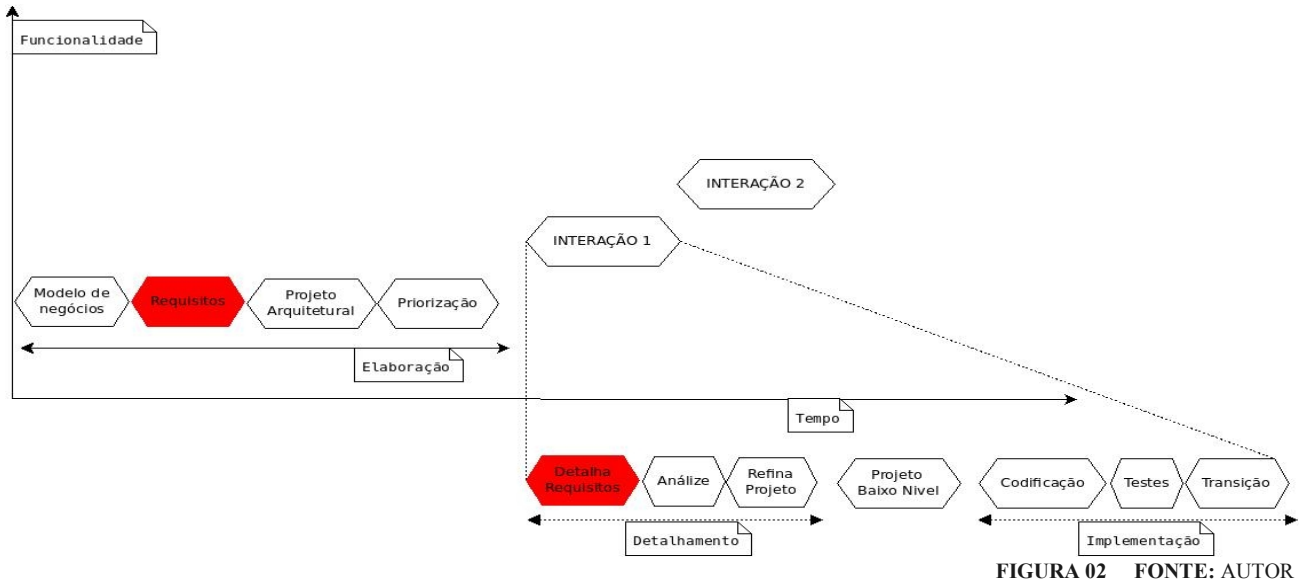


FIGURA 02 FONTE: AUTOR

4.2. Requisitos funcionais do sistema

Requisitos funcionais do sistema	
O software deve possuir cadastro de clientes	Evidente
O software deve gerar uma ordem de serviço	Evidente
A ordem de serviço deveria somar o total de serviços e produtos	Evidente
O software deve permitir imprimir esta ordem de serviço	Evidente
O software deve imprimir uma nota fiscal do serviço	Evidente
O software deve manter um log de ordem de serviços fechadas	Escondido
Um sistema de login antes de os usuários acessem o sistema	Evidente
Deve possuir um sistema de armazenamento permanente	Escondido

4.3. Levantamento requisitos não funcionais

Requisitos não funcionais
Designer para telas sensíveis ao toque
Possuir um valor relativamente baixo
Curto prazo para a entrega
O software terá que funcionar em tablet
O software terá que rodar em uma rede de computadores
O software terá que ser multi-plataforma
O software terá que possuir interface intuitiva
O software deverá rodar em computadores com poucos recursos de hardware
O software deverá ser robusto
O software terá que permitir mudanças futuras solicitadas pelo cliente

Atributos dos requisitos não funcionais do sistema

1. Robustez - O software deve rodar em condições extremas, independe de plataforma utilizada nos computadores clientes, ou condições de rede congestionada.

2. Interface Intuitiva - O software deve possuir uma interface intuitiva pois será utilizado por pessoas com pouco conhecimento na área de tecnologia, uma interface intuitiva auxilia na integração da empresa com o sistema de trabalho do software.

3. Poucos recursos de Hardware - A empresa não dispõe de grandes recursos financeiros para investir em equipamentos de hardware, então, o sistema irá funcionar em computadores já existentes, que possuem poucos recursos de hardware.

4.4. Levantamento de Casos de Uso

Com base nos dados relatados acima, continua-se a sequência, seguindo a próxima etapa, do processo de analise e desenvolvimento de sistemas.

Como pode ser analisado no diagrama de disciplina logo a baixo figura 03, neste ponto será efetuado uma analise para descobrir quais são os casos de uso do sistema.

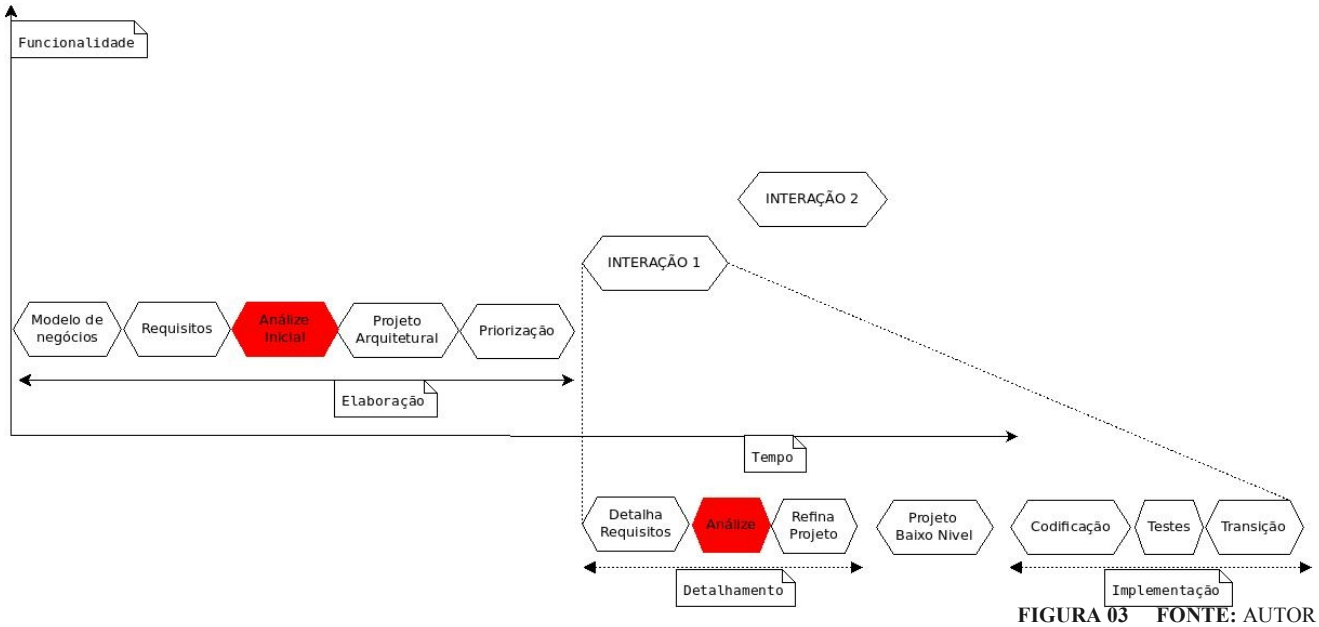


FIGURA 03 FONTE: AUTOR

Esta etapa da analise é extremamente importante, utilizar os dados captados na etapa de Modelo de Negócios, pois estes dados serão utilizados para identificar quais são os casos de uso, e então dar sequência ao processo de desenvolvimento do sistema.

Um caso de uso, como o próprio nome diz, é a definição de uma ação do usuário com o sistema, ele define uma interação completa, em que o utilizador do sistema acessa o sistema efetua alguma ação e então o software lhe responde com alguma informação, atendendo ou não a solicitação do usuário.

O usuário nos diagramas de casos de uso, são identificados como atores do sistema, estes usuários podem ser pessoas, sistema de software, ou de hardware, que interagem, solicitando algumas informações ou alimentando o sistema com alguns dados importantes.

Um caso de uso é completo quando, o ator efetua alguma interação com o sistema, o sistema responde a interação satisfazendo a solicitação do ator ou não, e informando o que é necessário para que o ator seja autorizado a efetuar tal ação.

Em síntese, é uma ação em que o ator acessa o sistema faz uma interação com o sistema e sai.[10]

CAPÍTULO V – Modelo de Caso de Uso

Seguindo a metodologia de desenvolvimento ICONIX, vemos na figura 04 a baixo o diagrama de Modelo de Caso de Uso, que auxilia na analise de projeto e desenvolvimento.

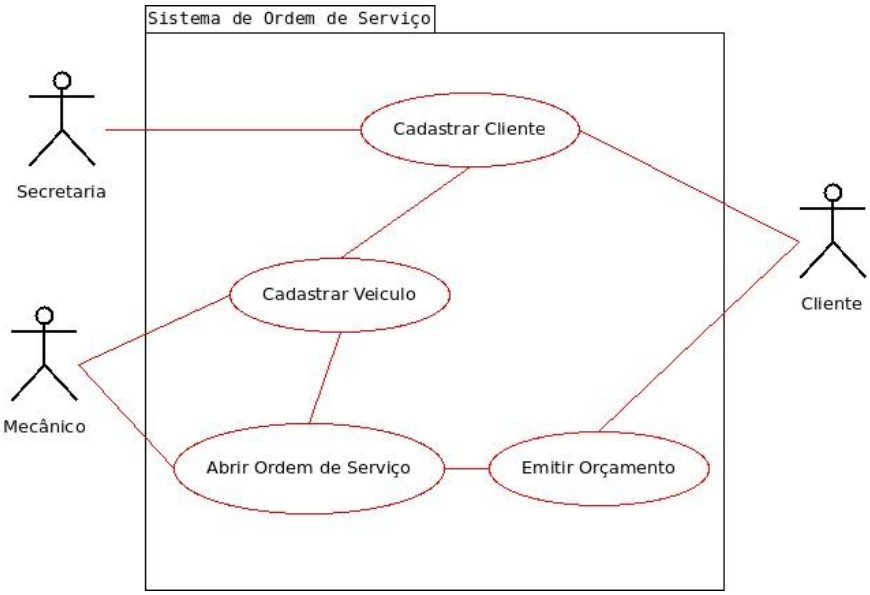


FIGURA 04 **FONTE:** AUTOR

[10]http://www.guj.com.br/content/articles/patterns/iconix_guj.pdf

5.1. Caso de Uso: Cadastrar Cliente

Caso de Uso: Cadastrar Cliente

Atores: Cliente , Secretaria

Breve descrição: Um cliente chega ao no estabelecimento. A secretária recebe o cliente e registra os seus dados no sistema. No final, os dados do cliente ficam cadastrados no sistema.

FIGURA 05 FONTE: Caso de Uso Cadastrar Cliente

Descrição da sequência tipica do evento.

1. O Cliente chega no estabelecimento e se dirige a secretária	Ação do Ator
2. A secretaria(o) recebe o cliente e solicita seus dados para cadastro no sistema	Ação do Ator
3. A Secretária solicita os dados do cliente; Nome, Sobrenome, Endereço, Cidade, Estado, CEP, Telefone Fixo, Celular 1, Celular 2, e-mail, CPF ou CNPJ e RG	Ação do Ator
4. O Sistema valida os dados e registra no banco de dados	Resposta do Sistema

5.2. Caso de Uso: Cadastrar Veículo

Caso de Uso: Cadastrar Veiculo

Atores: Mecânico , Cliente

Descrição: O mecânico cadastra o veiculo atrelando ao cliente, e adiciona dados como; placa, quilo-metragem, marca, modelo e cor.

FIGURA 06 Caso de Uso Cadastrar Veículo

Descrição da sequência tipica do evento

1. Após os dados registrados a secretária(o) solicita que um mecânico atenda o cliente.	Ação do Ator
2. O mecânico recebe o cliente, e o cliente explica os defeitos do veiculo para o mecânico	Ação do Ator
3. O mecânico acessa o sistema, utilizando um tablet, e adiciona o veiculo ao cadastro do cliente, cadastra dados como; placa, quilo-metragem, marca, modelo, cor, e observações.	Ação do Ator
4. O sistema valida e cadastra o veiculo atrelando no cadastro do cliente através dos números e letras da placa do veiculo, o sistema deve permitir que um cliente possua vários veículos.	Resposta do Sistema
5. O mecânico após efetuar o cadastro do veiculo, seleciona o mesmo e inicia um processo de abertura de ordem de serviço	Ação do Ator

5.3. Caso de Uso: Abrir Ordem de Serviço

Caso de Uso: Abrir Ordem de Serviço

Atores: Mecânico , Cliente

Descrição: Um cliente chega até o mecânico, e explica o defeito ao mesmo. O Mecânico abre a ordem de serviço no nome do cliente e registra as observações informadas pelo cliente. No final, o cliente sai e aguarda um diagnóstico.

FIGURA 07 Caso de Uso Abrir Ordem de Serviço

Descrição da sequência tipica do evento

1. O sistema solicita que sejam cadastrados os seguintes dados na ordem de serviço, data de entrada, data de possível entrega, data de entrega, tempo de garantia, defeito reclamado pelo cliente, e status.	Resposta do Sistema
2. O mecânico efetua o diagnóstico no veiculo e alimenta o sistema com o orçamento de serviços, possíveis peças a serem substituídas, e uma descrição do problema.	Ação do Ator
3. O sistema verifica os campos de diagnóstico do problema, peças serem substituídas e altera o status para aguardando autorização do cliente.	Resposta do sistema
4. O sistema exibe uma tela com as ultimas ordens de serviços com o status, aguardando autorização do cliente.	Resposta do sistema

5.4. Caso de Uso: Informar Orçamento

Caso de Uso: Informar Orçamento

Atores: Secretária, Cliente

Descrição: A secretaria visualiza a ordem de serviço com o status aguardando autorização, e informa o cliente.

FIGURA 08 **FONTE:** AUTOR

Descrição da sequência tipica do evento.

1. A Secretária visualiza a ordem de serviço com o status de aguardando autorização, e completa a ordem de serviço com os valores de peças e serviços.	Ação do Ator
2. O sistema efetua a soma total de serviços e peças, incluindo quantidade, e calcula os impostos e possível desconto sobre os produtos e serviços efetuados.	Resposta do Sistema
3. A secretária imprime uma versão do orçamento e entrega para o cliente, por e-mail, telefone ou fax.	Ação do ator

5.5. Caso de Uso: Encerrar Ordem de Serviço Não autorizada

Caso de Uso: Encerrar Ordem de Serviço Não autorizada

Atores: Mecânico, Secretária, Cliente

Descrição: O mecânico informa na ordem de serviço, o diagnóstico as peças e o valor da mão de obra. A Secretária visualiza a ordem de serviço e liga para o cliente informando o valor e pedindo a autorização do serviço. O cliente não autoriza o serviço. No final, a secretária altera o status da ordem de serviço para não autorizada e fechada.

FIGURA 09 **FONTE:** AUTOR

Descrição da sequência tipica do evento.

1 O Cliente não autoriza efetuar o serviço.	Ação do ator
2 A secretária altera o status da ordem de serviço para não autorizada, e solicita para que o cliente retire o seu veiculo da oficina em um prazo máximo de 48 horas, sob pena de multa por estadia ao exceder este período de tempo.	Ação do ator

5.6. Caso de Uso: Autorizar Ordem de Serviço

Caso de Uso: Ordem de Serviço Autorizada

Atores: Mecânico, Secretária, Cliente

Descrição: O mecânico informa na ordem de serviço, o diagnóstico as peças e o valor da mão de obra. A Secretária visualiza a ordem de serviço e liga para o cliente informando o valor e pedindo a autorização do serviço. O cliente autoriza o serviço. No final, a secretária altera o Status da Ordem de Serviço para autorizada.

FIGURA 10 **FONTE:** AUTOR

Descrição da sequência tipica do evento.

1. O cliente autoriza efetuar o serviço	Ação do ator
2. A Secretária altera o status da ordem de serviço para autorizado	Ação do ator
3. A secretária anexa as peças e uma copia da ordem de serviço autorizada junto com o veiculo do cliente	Ação do Ator
4. O sistema exibe na tela do tablet do mecânico que a ordem de serviço foi autorizada.	Resposta do sistema

5.7. Caso de Uso: Finalizar do Serviço Autorizado

Caso de Uso: Finalização do Serviço Autorizado

Atores: Mecânico, Secretária , Cliente

Descrição: O mecânico informa que o serviço está concluído e altera o status da ordem de serviço para finalizada. A Secretária visualiza a ordem de serviço e liga para o cliente informando solicitando que o cliente venha buscar o veiculo. A secretária imprime uma guia da ordem de serviço e uma nota fiscal para o cliente. No final, a secretária altera o Status da Ordem de Serviço para Finalizada e Fechada.

FIGURA 11 **FONTE:** AUTOR

Descrição da sequência tipica do evento.

1. O mecânico finaliza o serviço e e altera o status da ordem de serviço para finalizado.	Ação do Ator
2. O sistema exibe na tela da secretária a ordem de serviço com o status de Finalizada.	Ação do sistema
3. A secretária faz contato com o cliente por telefone solicitando a remoção do veiculo, com prazo máximo de 48 horas, sob pena de multa por estadia.	Ação do ator
4. O Cliente retira o veiculo e efetua o pagamento do serviço.	Ação do ator
5. A Secretária da baixa na ordem de serviço alterando o status para fechada e paga.	Ação do ator
6. O sistema armazena a ordem de serviço em um histórico de ordens de serviços fechadas.	Resposta do sistema

CAPÍTULO VI – Análise Robusta

Diagramas de Robustez

O diagrama de robustez ajuda a ter certeza de que não foi esquecido nenhum dos requisitos durante a análise robusta, será possível encontrar problemas na descrição dos casos de uso.

A construção do modelo robusto, tem a função de encontrar objetos novos que não foram identificados anteriormente, também é possível localizar conflitos entre objetos entre outros possíveis problemas.[11]

6.1. Diagrama de Robustez: Cadastrar Cliente

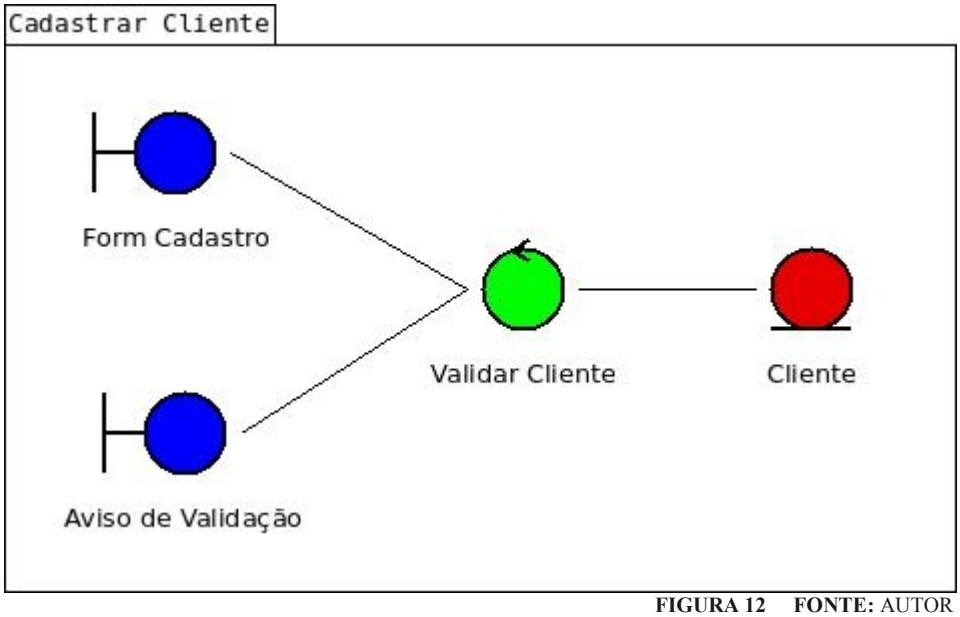


FIGURA 12 **FONTE:** AUTOR

Diagramas de robustez do caso de uso Cadastrar Cliente, visto na figura 12 neste diagrama foi identificado o controlador validar cliente.

[11] http://www.guj.com.br/content/articles/patterns/iconix_guj.pdf

6.2. Diagrama de Robustez: Cadastrar Veiculo

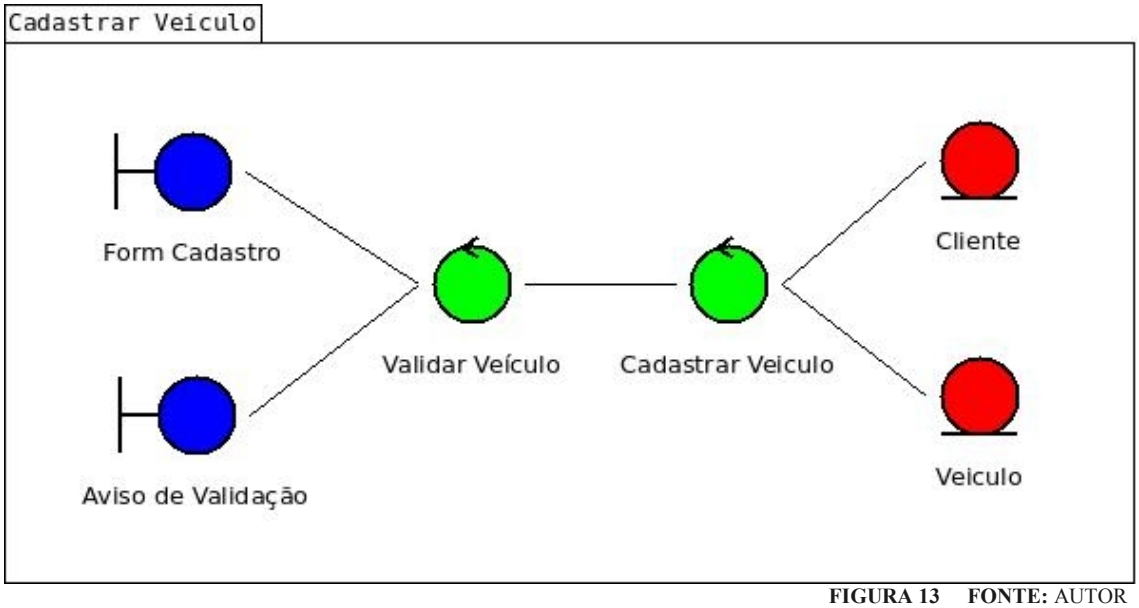


FIGURA 13 **FONTE:** AUTOR

Diagrama de robustez do caso de uso Cadastrar veículo, visto na figura 13 neste diagrama foi identificado o controlador validar veículo.

6.3. Diagrama de Robustez: Abrir Ordem de Serviço

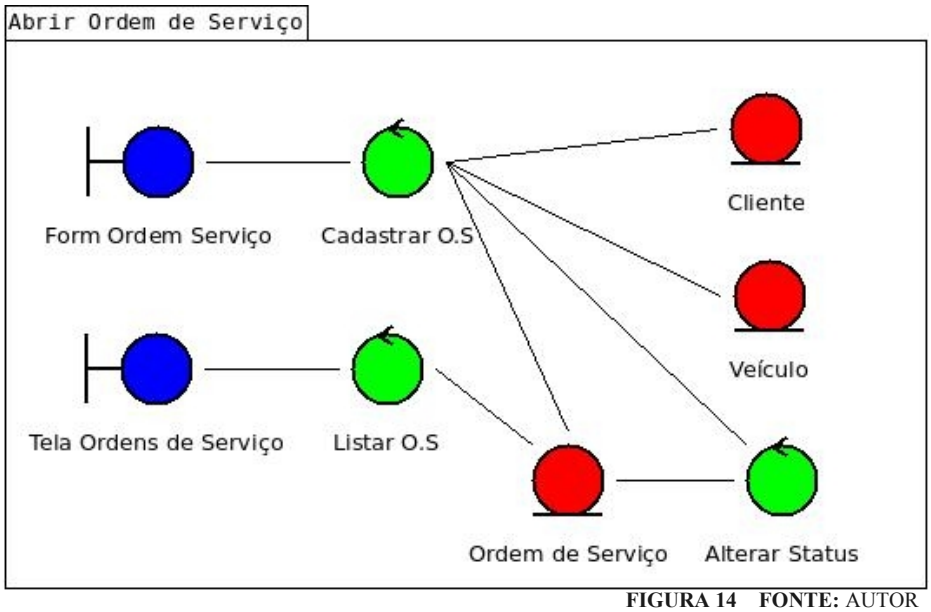


FIGURA 14 **FONTE:** AUTOR

Diagrama de robustez do caso de uso Abrir Ordem de Serviço, visto na figura 14 neste diagrama foi identificado o controlador alterar status.

6.4. Diagrama de Robustez: Informar Orçamento

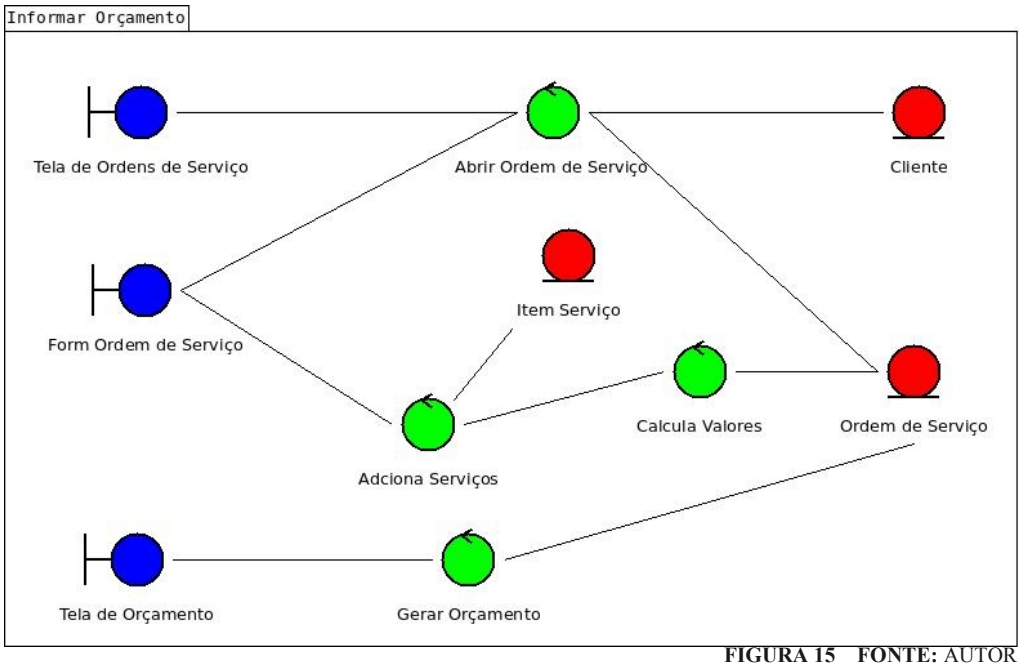


FIGURA 15 FONTE: AUTOR

Diagrama de robustez do caso de uso Informar Orçamento , visto na figura 15 neste diagrama foram identificados os controladores calcular valores, e gerar orçamento.

6.5. Diagrama de Robustez: Autorizar Ordem de Serviço

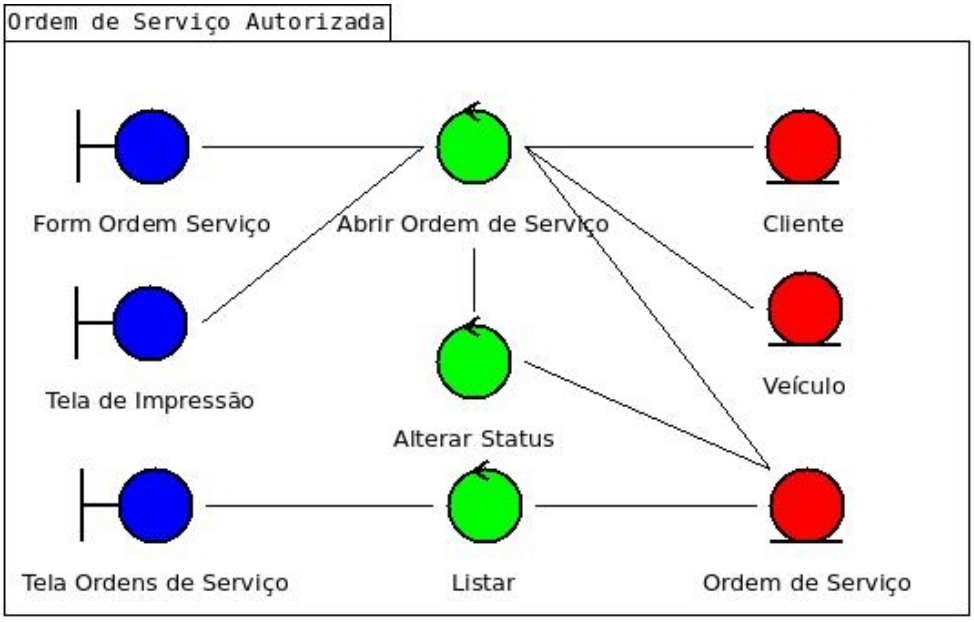


FIGURA 16 FONTE: AUTOR

Diagramas de robustez do caso de uso Abrir Ordem de Serviço, visto na figura 16 neste diagrama foi identificado o controlador alterar status.

6.6. Diagrama de Robustez: Não Autorizar Ordem de Serviço

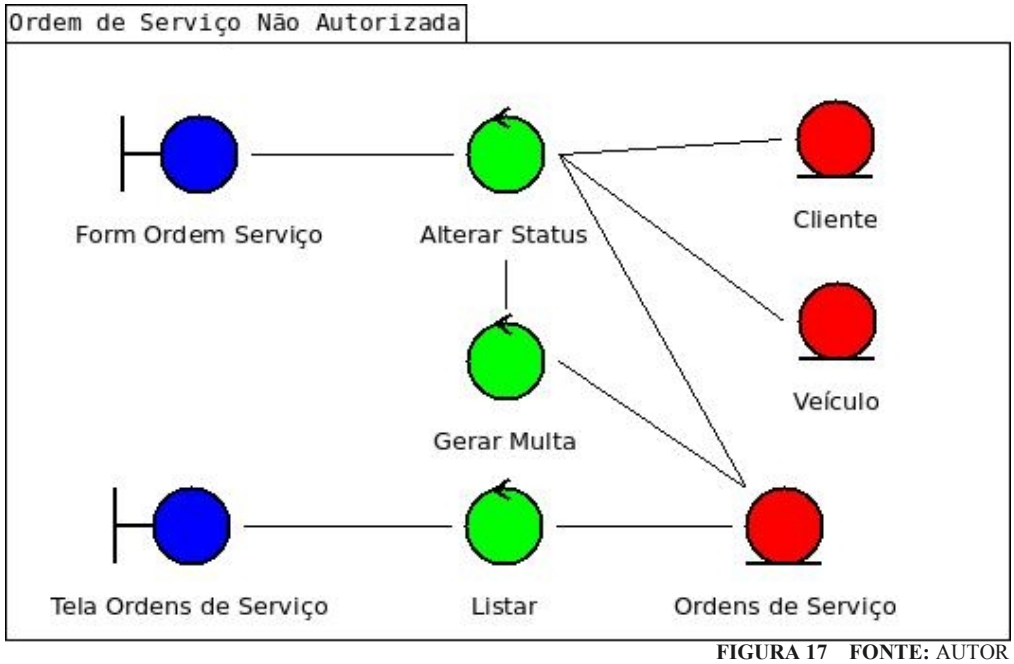


FIGURA 17 FONTE: AUTOR

Diagrama de robustez do caso de uso Não Autorizar Ordem de Serviço , visto na figura 17 neste diagrama foram identificados os controladores gerar multa, e alterar status.

6.7. Diagrama de Robustez: Finalizar Ordem de Serviço

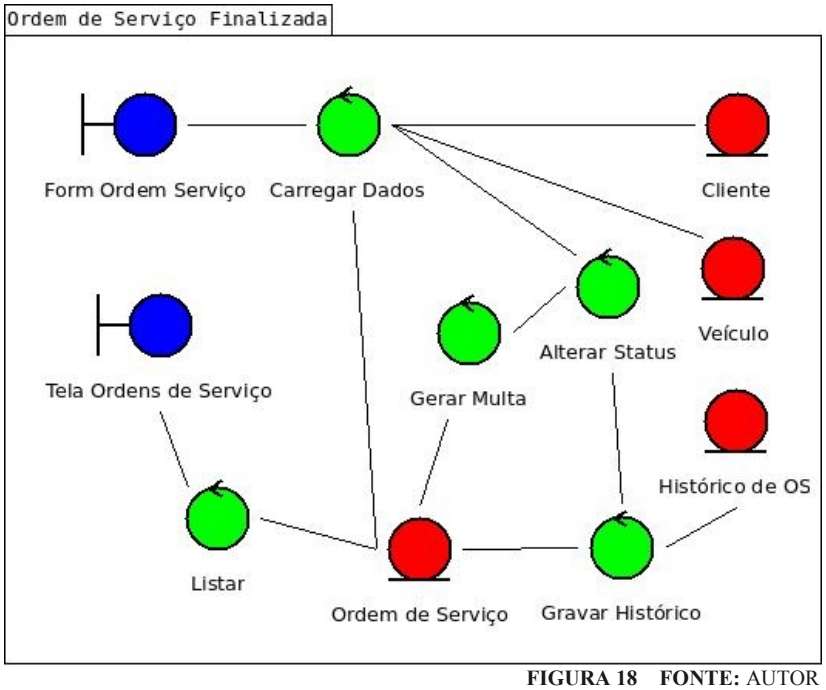


FIGURA 18 FONTE: AUTOR

Diagramas de robustez do caso de uso Finalizar Ordem de Serviço, visto na figura 18 neste diagrama foi identificado os controladores alterar status, gerar multa, listar e gravar histórico.

CAPÍTULO VII – Diagrama de sequência

Os diagramas de sequência tem o objetivo de fornecer um modelo dinâmico entre o usuário e o sistema. Este diagrama é baseado nos diagramas de robustez, descritos no capitulo VII, a função do diagrama de sequencia.

Nestes diagramas que são projetados como realmente o software vai funcionar, então é modelado como o sistema executa comportamentos úteis.[12]

7.1. Diagrama de sequência: Cadastrar Cliente

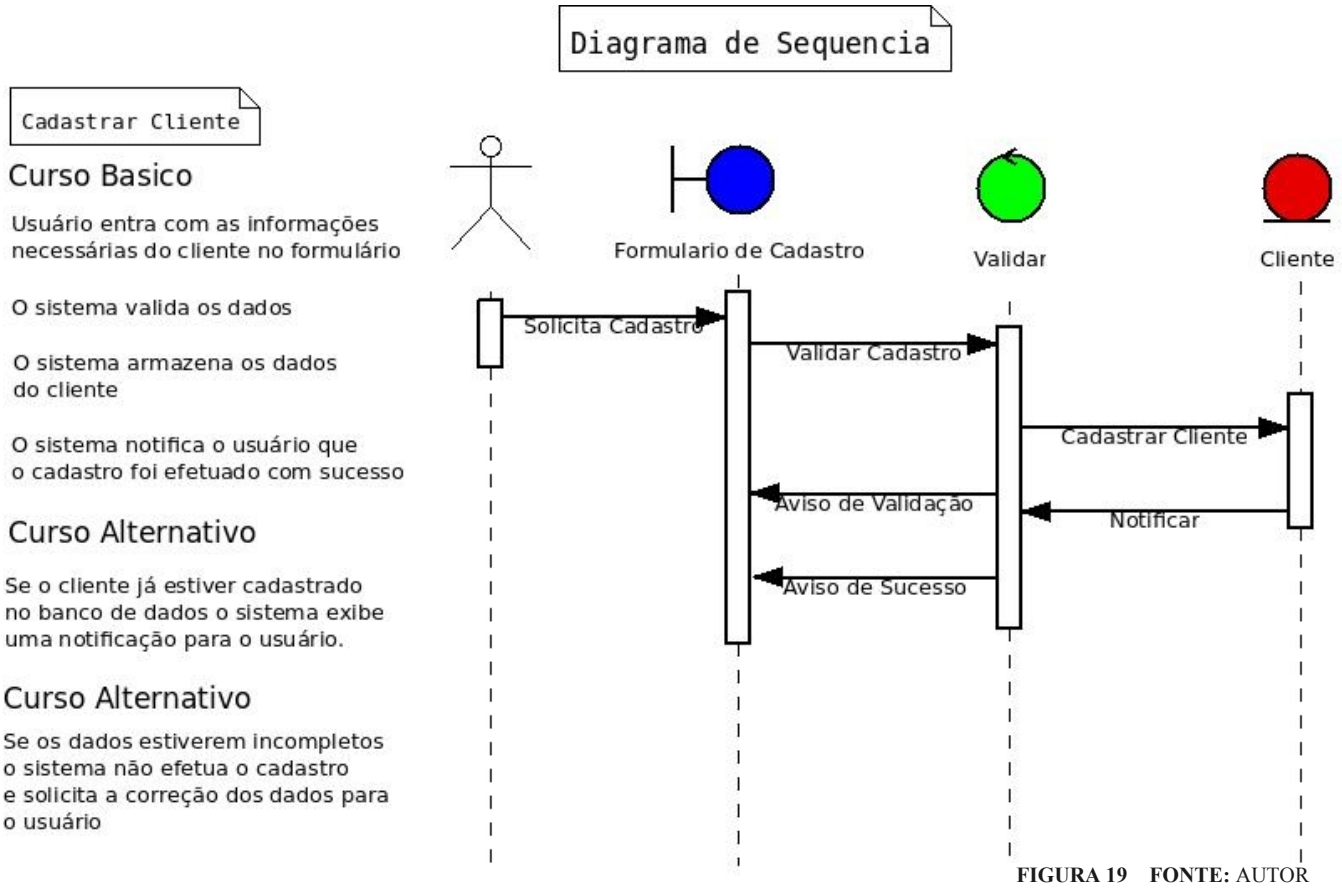


FIGURA 19 FONTE: AUTOR

[12]http://www.guj.com.br/content/articles/patterns/iconix_guj.pdf

7.2. Diagrama de Sequência: Cadastrar Veículo

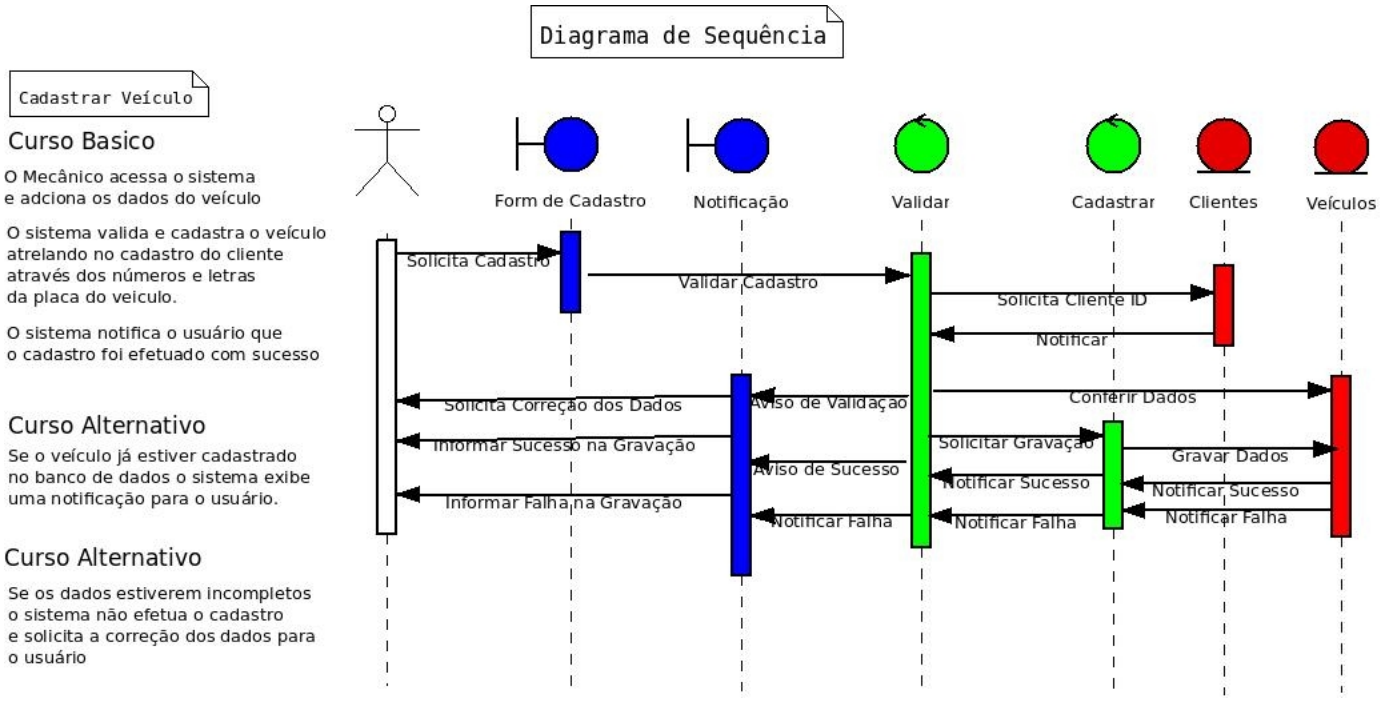


FIGURA 20 FONTE: AUTOR

7.3. Diagrama de sequência: Abrir Ordem de Serviço

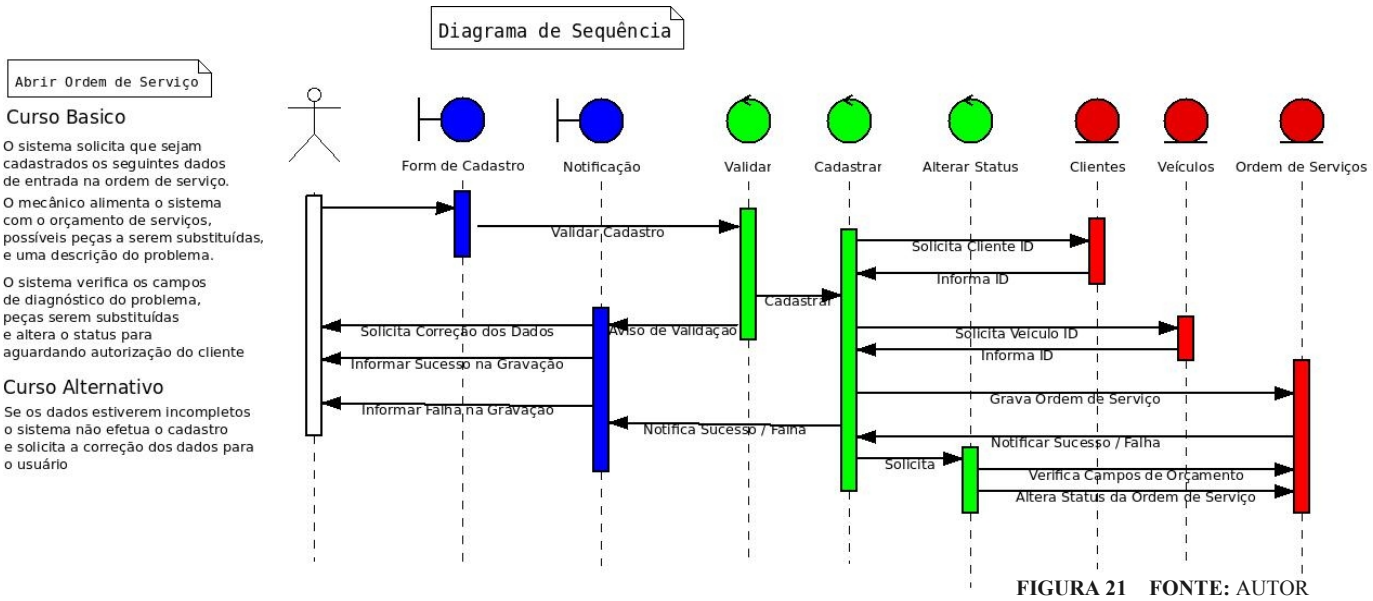


FIGURA 21 FONTE: AUTOR

7.4. Diagrama de Sequencia: Informar Orçamento

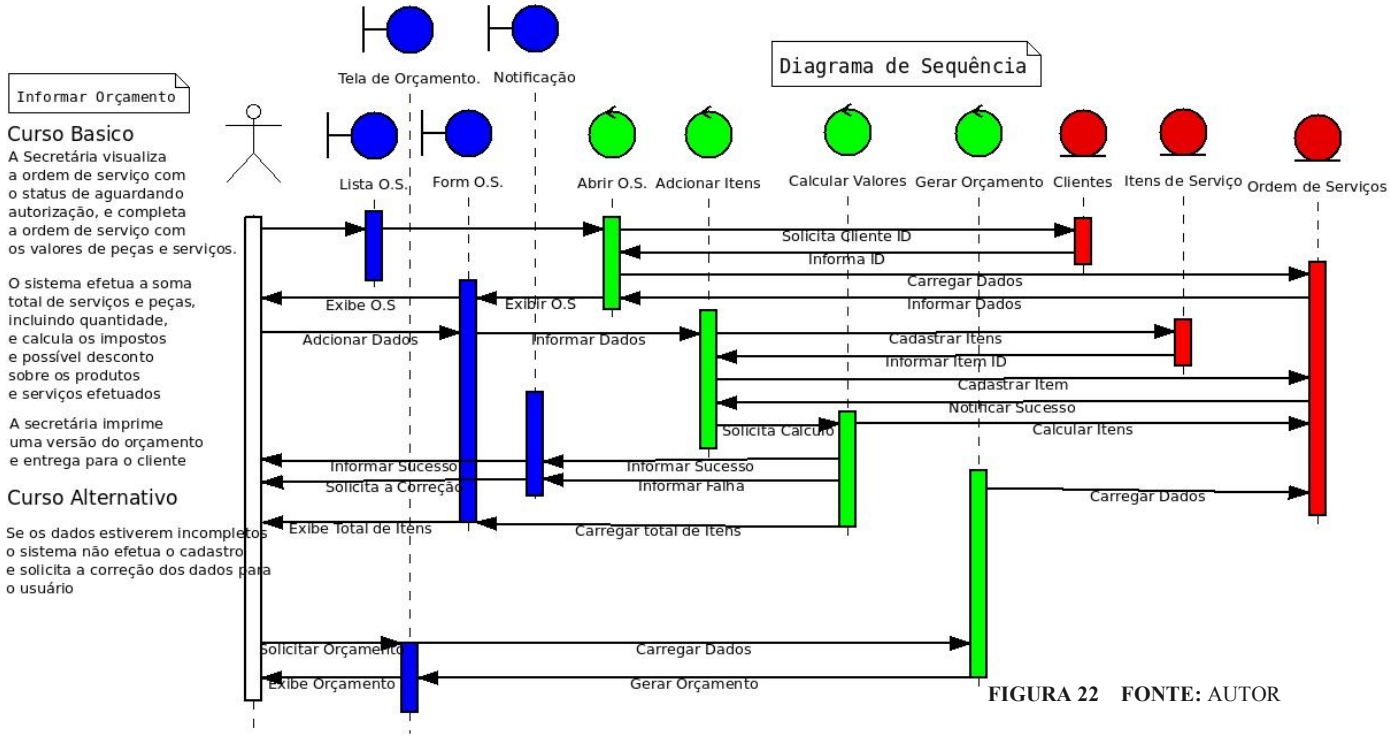


FIGURA 22 FONTE: AUTOR

7.5. Diagrama de Sequência: Autorizar Ordem de Serviço

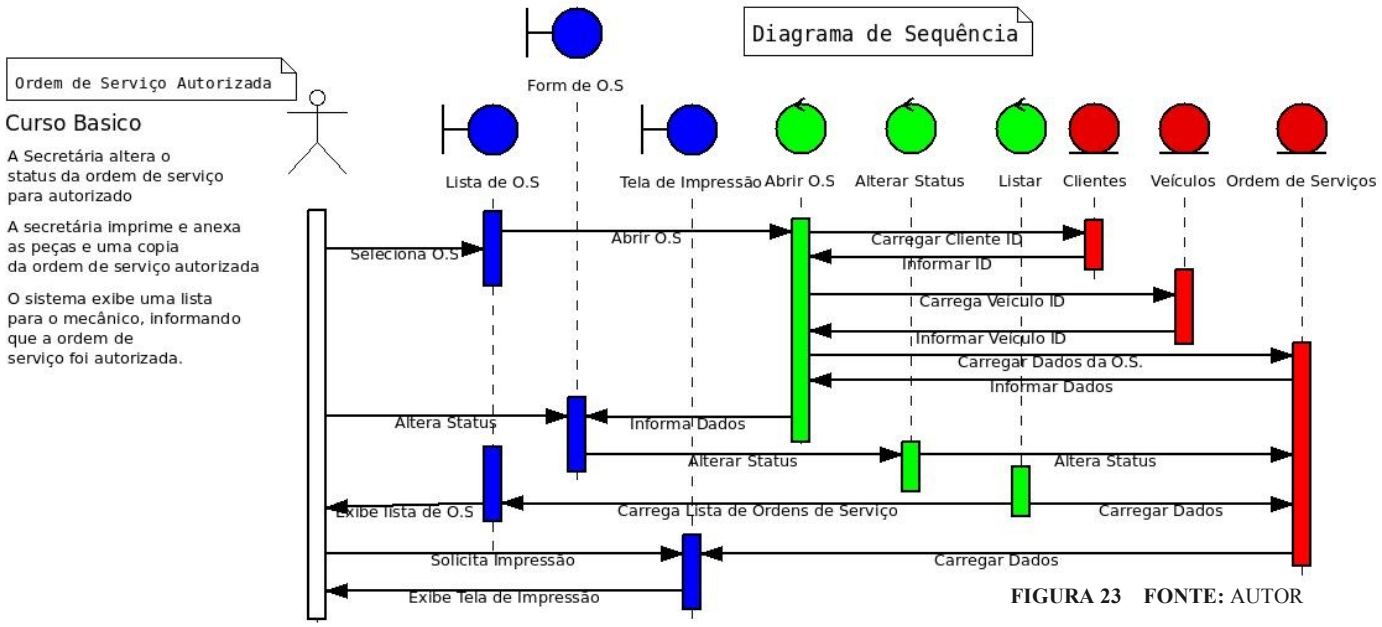


FIGURA 23 FONTE: AUTOR

7.6. Diagrama de Sequência: Não Autorizar Ordem de Serviço

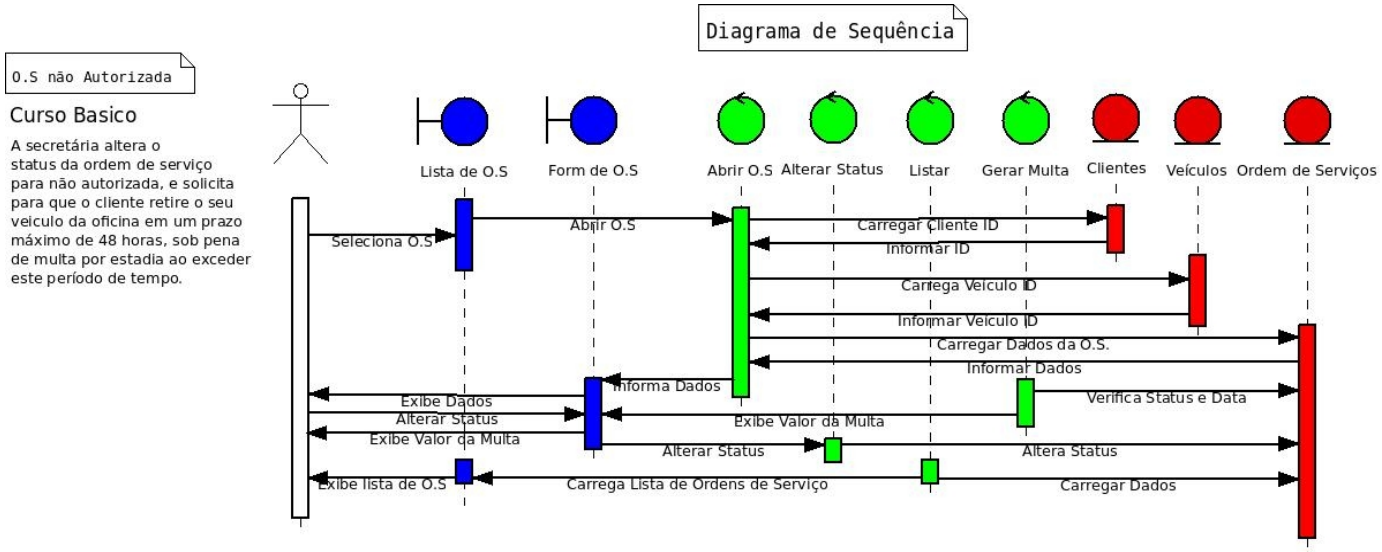


FIGURA 24 FONTE: AUTOR

7.7. Diagrama de Sequência: Finalizar Ordem de Serviço

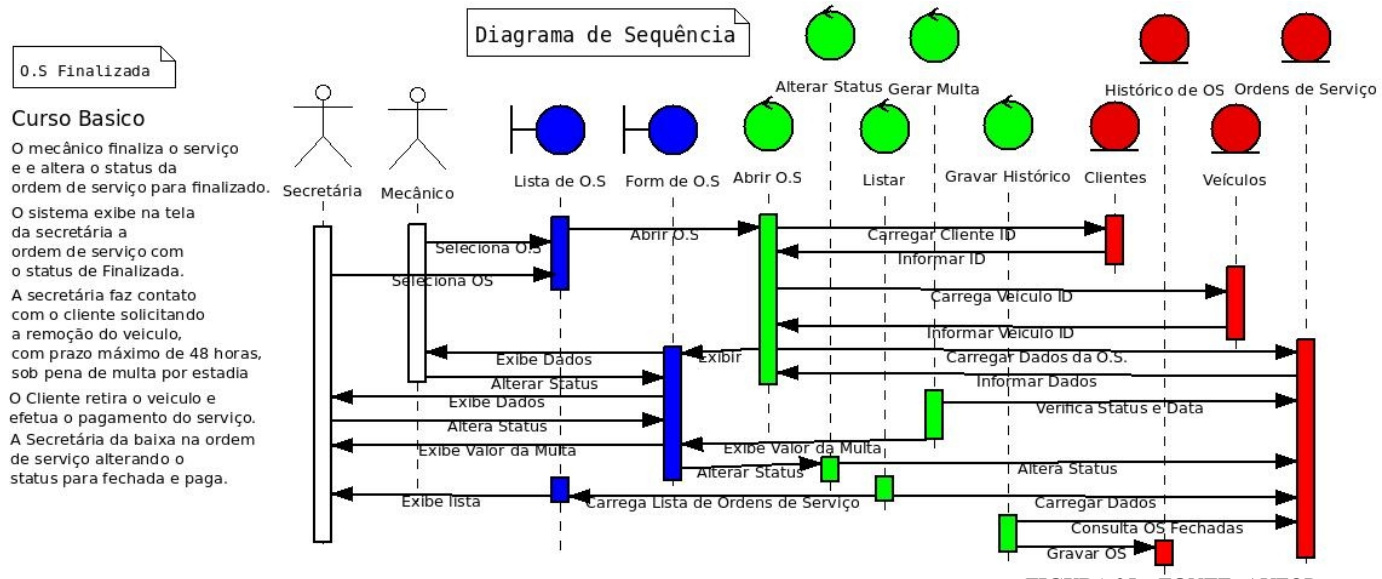


FIGURA 25 FONTE: AUTOR

CAPÍTULO VIII – Analise de Classes

Com base nos dados da analise e nos diagramas de casos de uso, relacionados acima, e na analise do Modelo de Negócios, foi criado o diagrama de classes conceitual, que permite visualizar o sistema de maneira conceitual.

8.1. Diagrama de Classes Conceitual

FIGURA 26 FONTE: AUTOR

8.2. Diagrama de Classes Projeto

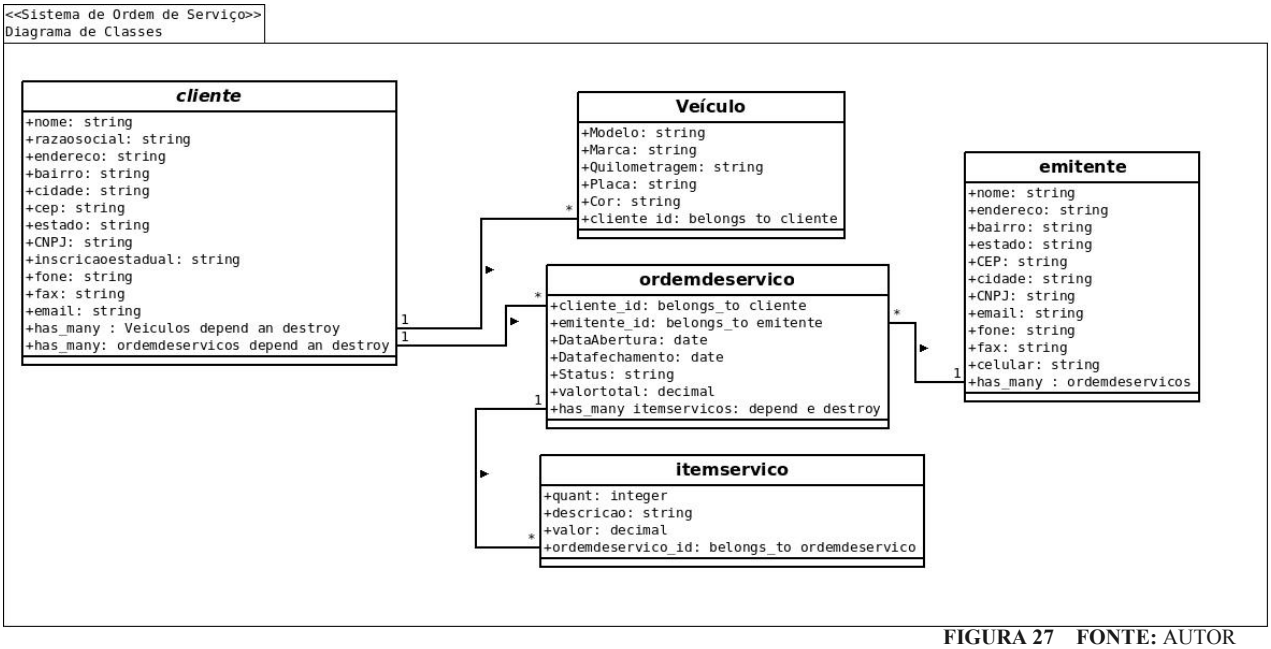


FIGURA 27 FONTE: AUTOR

CAPÍTULO IX – Implementação

Após todo o processo de analise do plano de negócios, e da analise de desenvolvimento, se inicia a etapa do processo de desenvolvimento do software. Como pode ser observado no diagrama de disciplina na figura 28 logo abaixo.

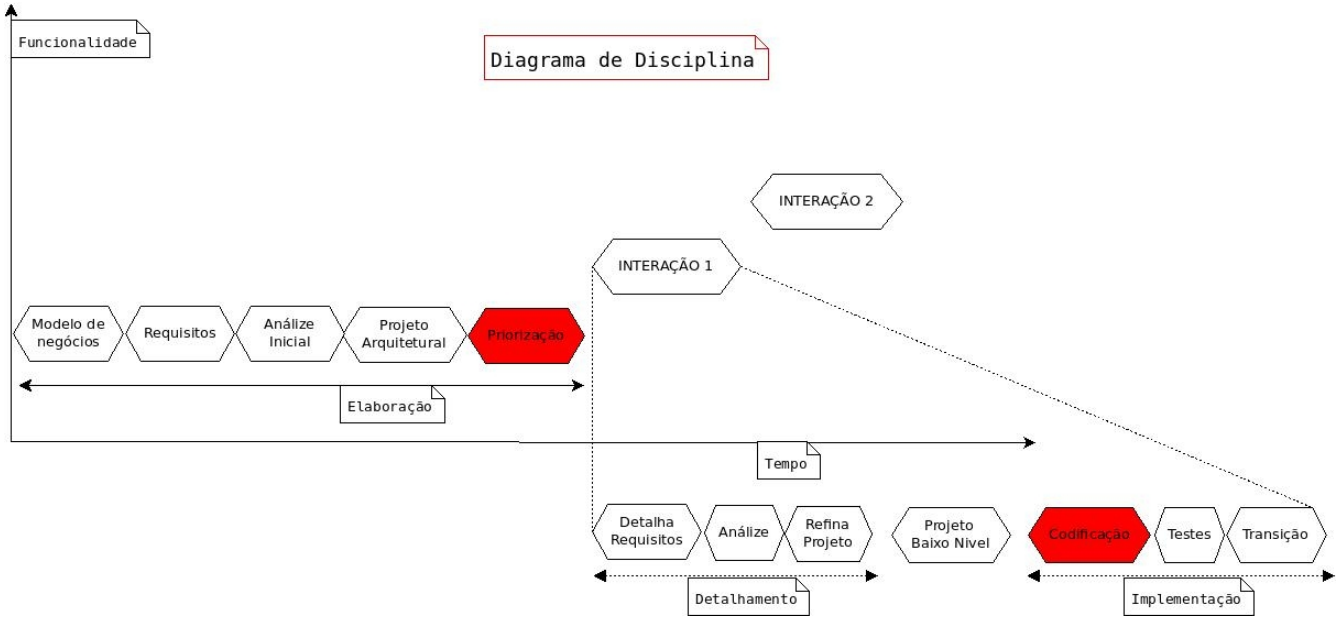


FIGURA 28 FONTE: AUTOR

O processo de codificação é onde inicia a escrita de código em si, nesta etapa a equipe deve definir qual é a plataforma de banco de dados a ser utilizada e a linguagem de programação.

Este processo de inicio de codificação deve ser definido com base nos requisitos funcionais e não funcionais do sistema.

9.1. Codificação com base nos requisitos levantados

O processo de codificação é onde inicia a escrita de código em si, nesta etapa a equipe deve definir qual é a plataforma de banco de dados a ser utilizada e a linguagem de programação.

Este processo de inicio de codificação deve ser definido com base nos requisitos funcionais e não funcionais do sistema.

Neste projeto de desenvolvimento de software para controle de oficina mecânica, durante o processo de levantamento de requisitos, identificamos alguns requisitos que podem influenciar diretamente no processo de codificação, são eles:

- Designer para telas sensíveis ao toque
- Possuir um baixo custo
- Deverá funcionar em tablet
- Terá que funcionar em uma rede de computadores
- A necessidade de ser multi-plataforma
- Rodar em computadores com pouco recurso de hardware
- Permitir possíveis mudanças solicitadas pelo cliente no futuro

Com base nos requisitos não funcionais, é possível identificar que, para codificar este software, devemos optar por uma plataforma web, com este recurso será possível satisfazer alguns dos requisitos não funcionais como: designer para telas sensíveis ao toque, possuir um baixo custo, deverá funcionar em tablet, terá que funcionar em uma rede de computadores, a necessidade de ser multi-plataforma e rodar em computadores com pouco recurso de hardware.

Hoje em dia, existem vários recursos para o desenvolvimento em plataforma web, é possível desenvolver com os mais vária-dos tipos de frameworks, assim como linguagem de programação, a escolha correta do framework e a linguagem de programação adequada pode definir o sucesso ou fracasso do projeto de desenvolvimento de software.

Ao utilizar a plataforma web, neste projeto, podemos obter algumas vantagens, como por exemplo; o designer para telas sensíveis ao toque, hoje em dia possuí muitas ferramentas web que possibilita a criação de uma interface web, com excelentes recursos para telas sensíveis ao toque, outro fator é relacionado ao funcionamento em tablet.

Para este problema, poderia ser desenvolvido um aplicativo que rode sobre o sistema android, windows phone ou iOS, porém ao desenvolver aplicativos específicos para cada plataforma, estaríamos indo contra o requisito, que solicita que o software seja multiplataforma, outro problema seria a respeito do prazo de entrega, uma vez que teria que desenvolver três vezes o mesmo software, sendo um para cada plataforma.

Outro fator importante a ser levantado, é a respeito da plataforma de hardware em que o sistema vai rodar, está que deve ser uma plataforma com poucos recursos de hardware, ao utilizar um sistema com interface web, a execução total de processamento e armazenamento de banco de dados, ficam totalmente no servidor, tornando o computador cliente, apenas um terminal de acesso, não exigindo que o mesmo possua grandes recursos de hardware, como poder de processamento ou de armazenamento, permitindo que se mantenha um bom desempenho, do sistema.

Outro requisito de modelo de negócios que será atendido, ao decidir a utilização de uma plataforma web, é a respeito da necessidade do sistema rodar sobre uma rede de computadores, esta necessidade é sanada ao utilizar uma plataforma de sistema web, pois o mesmo roda totalmente em rede, seja ela uma rede local, como na rede de internet, permitindo o acesso ao sistema de software, de qualquer local do mundo, bastando apenas possuir o endereço IP (Internet Protocol), usuário e senha, para o acesso ao sistema.

O requisito de robustez também é sanado ao utilizar um sistema web, uma vez que não necessita de nenhum tipo de instalação nos computadores cliente, é possível substituir o computador, e imediatamente em um outro computador acessar o sistema e continuar o processo normalmente, não se tornando necessário a intervenção de um técnico para a reinstalação do software na nova maquina.

Ao levantar estes fatores, decidimos desenvolver o software web, que rode em sua totalidade dentro de um servidor, e utiliza o navegador de internet para interface com o usuário, com isso estaríamos desenvolvendo um sistema, que atenda boa parte dos requisitos não funcionais do modelo de negócios.

9.2. Definindo Banco de Dados

No processo de definição do banco de dados, devemos optar pela plataforma de banco de dados que melhor se encaixe no projeto, de maneira que, os requisitos não funcionais e funcionais não interferem diretamente nesta decisão, que deve ser tomada pela equipe de desenvolvimento.

A linguagem de banco de dados, escolhido para ser utilizado neste projeto de desenvolvimento de software, foi o SQL (Structured Query Language), que é atualmente a linguagem de banco de dados mais utilizada no mundo, foi escolhido por ser possuir uma licença GPL.

Para gerenciar o banco de dados, foi escolhido o SGDB (Sistema de gerenciamento de banco de dados), conhecido como Mysql, é um gerenciador de banco de dados, desenvolvido para trabalhar própria-mente com sistemas Web, é um dos SGDB mais utilizados no mundo, alguns dos motivos que foram levados em consideração no momento da escolha do sistema gerenciador de banco de dados, foi o custo de licença, que no caso do Mysql trabalha sobre a licença GPL (Licença Pública Geral), permitindo que seja utilizado sem gerar maiores custos para o consumidor final, ou para a equipe de desenvolvimento, outros fatores também foram importantes para a escolha do SGDB Mysql, como o suporte, estabilidade, robustez e baixo consumo de hardware.

9.3. Definindo Linguagem de Programação

Outro fator importante, foi a definição da linguagem de programação que deveria ser utilizada no projeto. Nesta etapa foram estudados algumas possíveis linguagens, para descobrir qual supria melhor os requisitos do sistema.

9.4. Analisando Linguagem de Java

A primeira linguagem a ser estudada a possibilidade de utilização, foi a linguagem de programação Java. Java é uma linguagem de programação Orientada a Objetos, que possui algumas vantagens como; portabilidade, recursos de rede, segurança.

Porém ao analisar mais profundamente sobre a linguem java, foi descoberto a dificuldade em encontrar um framework, que possibilitasse um desenvolvimento web de maneira totalmente integrada, e assim desenvolver um sistema web em java de maneira organizada e direta, é necessário a utilização de uma série de frameworks, o que causaria uma maior demanda de mão de obra, e um tempo maior para o desenvolvimento, levando em consideração a necessidade da equipe aprender e dominar todos os frameworks necessários, isto impossibilitaria o desenvolvimento de um software de baixo custo, e com um curto prazo de entrega, não satisfazendo assim dois requisitos do projeto.[13]

9.5. Analisando Linguagem de PHP

Uma segunda linguagem analisada foi PHP, foi criada em meados de 1994 como um pacote de programas CGI, com o nome de Personal Home Page Tools, utilizado para substituir um conjunto de scripts Perl que se utilizava no desenvolvimento de pagina pessoal.[15]

O PHP se tornou uma das linguagens web mais utilizadas no mundo, e possui muita documentação, e permite a conexão com diferentes sistemas gerenciadores de baco de dados. Ao analisar a linguagem PHP foi encontrado um framework conhecido como Cake PHP, que é relativamente completo, que permitiria o desenvolvimento de um sistema totalmente web.[14]

[13] http://www.java.com/pt_BR/download/faq/whatis_java.xml
[14] http://cakephp.org/
[15] http://php.net/manual/pt_BR/intro-whatis.php

9.6. Analisando Linguagem Ruby

A terceira linguagem de programação analisada foi uma linguagem conhecida como Ruby, que possui algumas das seguintes características; é uma linguagem de programação interpretada de multi-plataforma, possui alguns recursos de gerenciamento de memória automática, foi originalmente planejada e desenvolvida no Japão em 1995, foi desenvolvida para que fosse mais poderosa que Perl e mais Orientada a Objeto do que a linguagem Pyton, Ruby é hoje a 11º linguagem mais popular no mundo.[15]

Uma das vantagens de Ruby em relação as outras linguagens de programação, é a possibilidade de se utilizar o framework Rails. Ruby on Rails, é um framework livre que possibilita aumentar a velocidade do desenvolvimento de sistemas orientados ao banco de dados, uma vez que é possível criar aplicações com base em estruturas pré definidas. As aplicações que são criadas em Rails, são desenvolvidas com o padrão de arquitetura MVC (Model Control View). Outro conceito importante do Rails, é o DRY (Don't Repeat Yourself, Não se repita), que oferece uma estrutura de código que pode ser utilizado em várias outras partes da aplicação, sem que seja necessário a copia do código pelo programador, evitando a duplicação de código e aumentando relativamente a produtividade.

Após análise e das linguagens de programação e sistemas gerenciadores de banco de dados, foi definido a utilização da linguagem de programação Ruby com o framework Rails, em conjunto com o sistema gerenciador de banco de dados Mysql.

Por Ian Castelli em 24 de Julho de 2013

Position Jul 2013	Position Jul 2012	Delta in Position	Programming Language	Ratings Jul 2013	Delta Jul 2012	Status
1	1	=	C	17.628%	-0.70%	A
2	2	=	Java	15.906%	-0.18%	A
3	3	=	Objective-C	10.248%	+0.91%	A
4	4	=	C++	8.749%	-0.37%	A
5	7	↑↑	PHP	7.186%	+2.17%	A
6	5	↓	C#	6.212%	-0.46%	A
7	6	↓	(Visual) Basic	4.336%	-1.36%	A
8	8	=	Python	4.035%	+0.03%	A
9	9	=	Perl	2.148%	+0.10%	A
10	11	↑	JavaScript	1.844%	+0.39%	A
11	10	↓	Ruby	1.582%	-0.19%	A
12	14	↑↑	Transact-SQL	1.568%	+0.61%	A
13	15	↑↑	Visual Basic .NET	1.254%	+0.34%	A
14	19	↑↑↑↑↑	PL/SQL	0.920%	+0.28%	A-
15	13	↓↓	Lisp	0.868%	-0.13%	A
16	16	=	Pascal	0.792%	-0.04%	A
17	12	↓↓↓↓↓	Delphi/Object Pascal	0.691%	-0.47%	B
18	20	↑↑	MATLAB	0.680%	+0.04%	B
19	23	↑↑↑↑	Bash	0.622%	+0.04%	A-
20	25	↑↑↑↑↑	Assembly	0.581%	+0.03%	B

[15]https://www.ruby-lang.org/pt/about/
[16]http://www.rubyonrails.com.br/

9.7. Telas do Sistema

O sistema de software, foi produzido e implementado com sucesso, segue abaixo as telas do software em operação.

Devido a motivos de segurança e privacidade, foram apagados informações de IP do e dados de clientes exibidos nas imagens.

Tela de Login

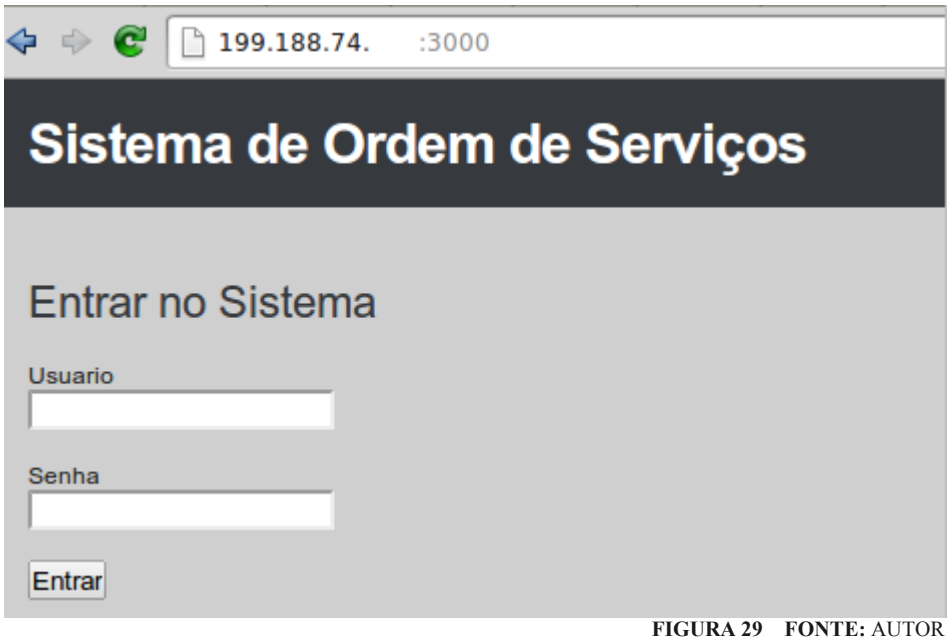


FIGURA 29 FONTE: AUTOR

Tela Listar Clientes

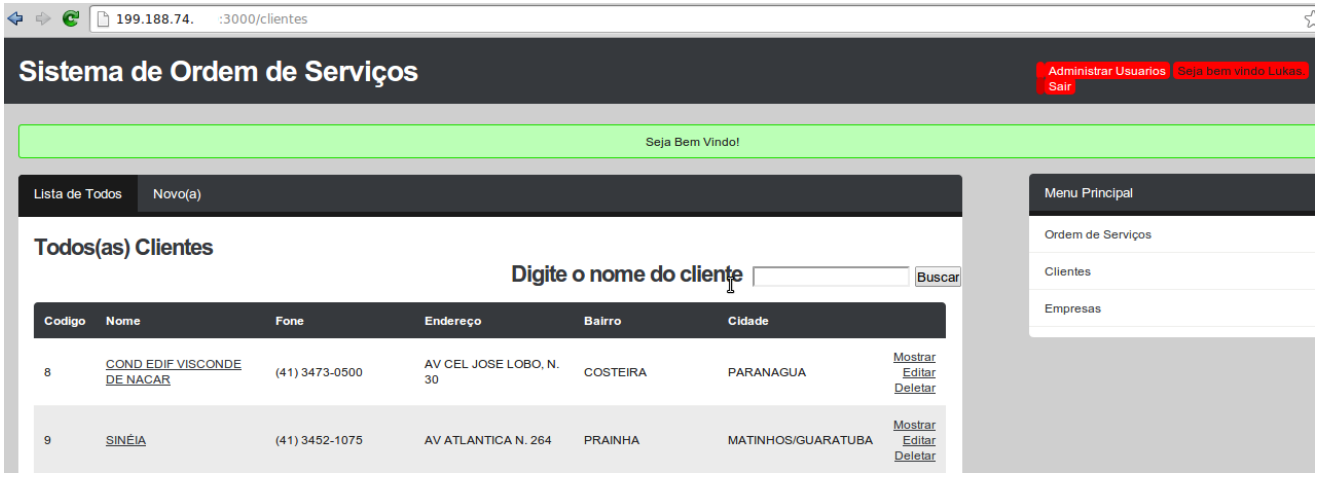
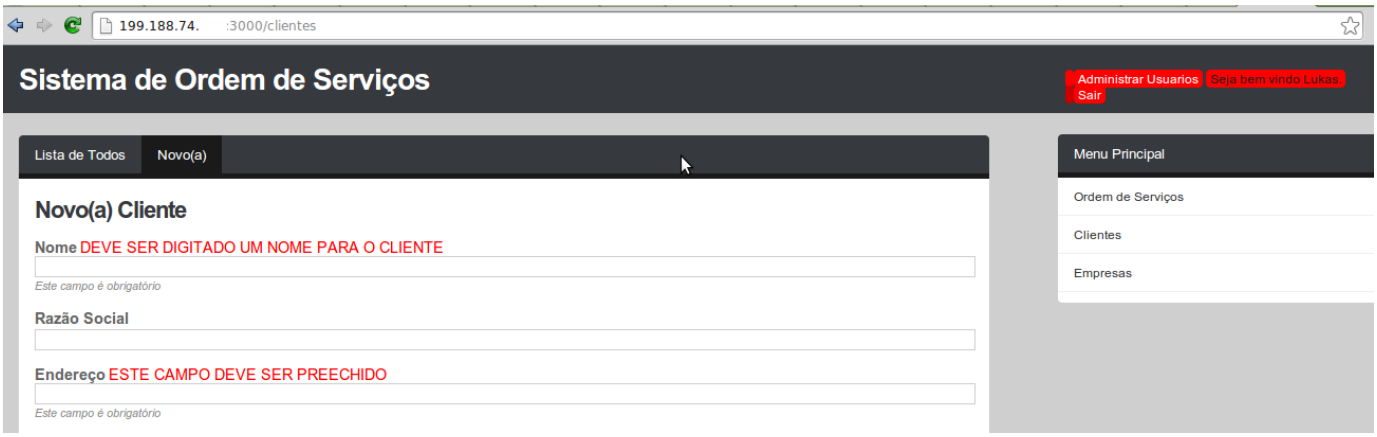


FIGURA 30 FONTE: AUTOR

Tela de Cadastro de Cliente



Tela listar ordem de serviço

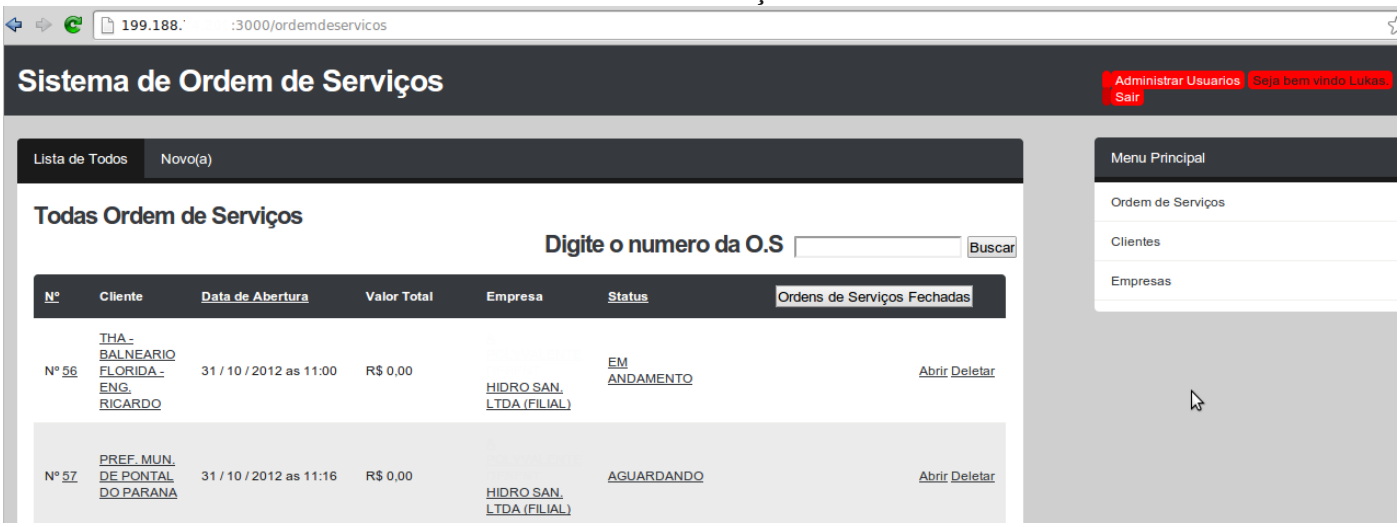


FIGURA 31 FONTE: AUTOR

Tela Abrir Ordem de Serviço

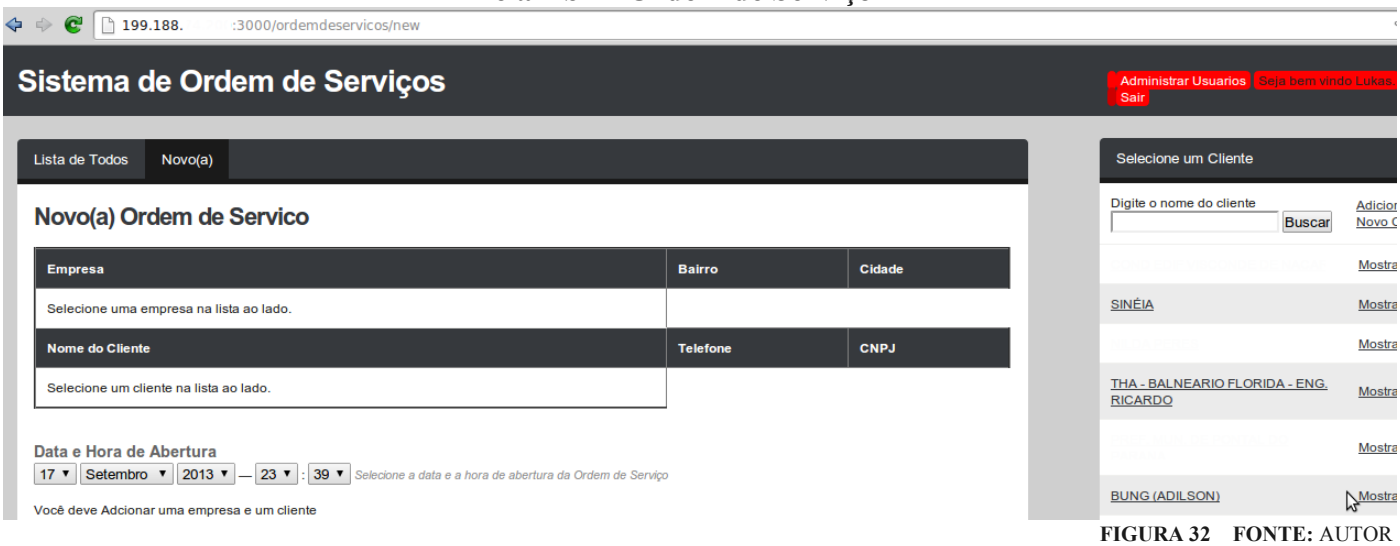


FIGURA 32 FONTE: AUTOR

Tela de ordem de serviço

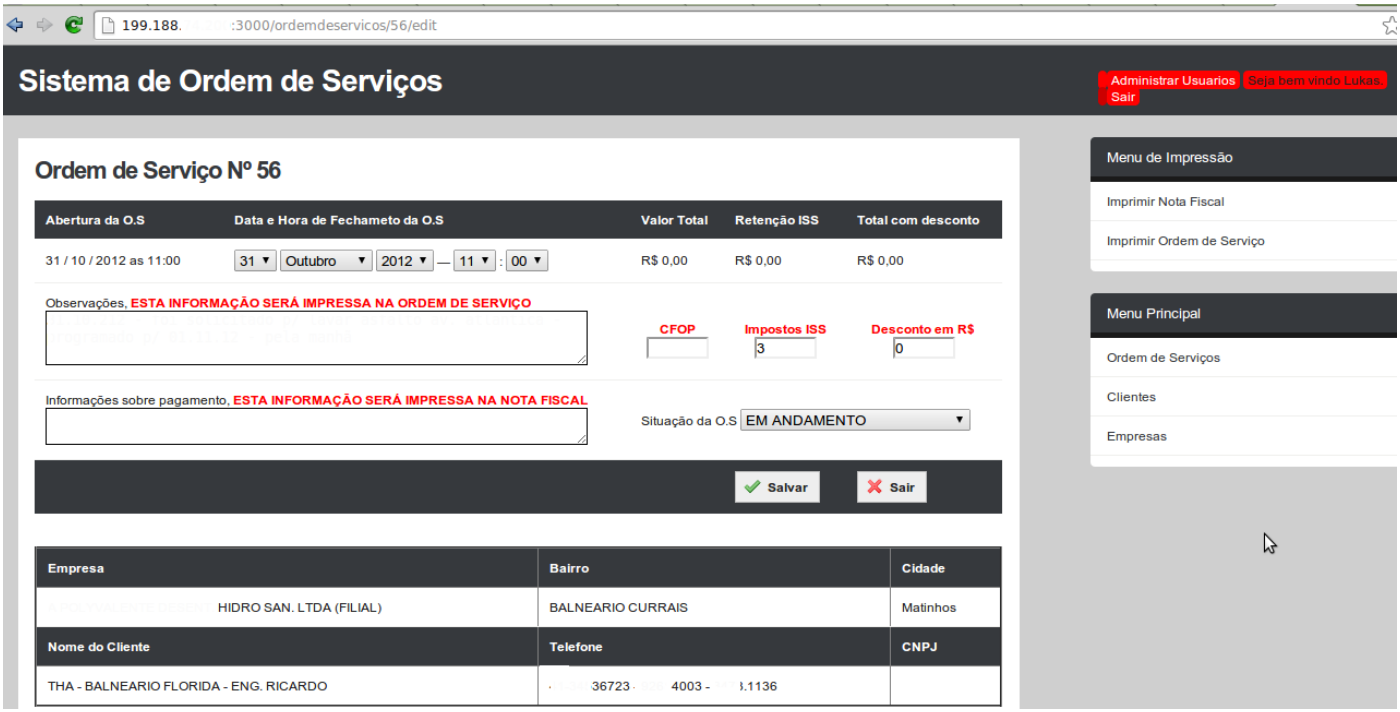
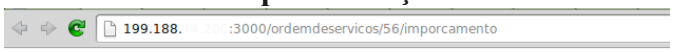


FIGURA 33 FONTE: AUTOR

Tela Imprimir Orçamento



Prestadora de Serviços	
Empresa:	
Bairro:	**BALNEARIO CURRAIS**
Cidade:	**Matinhos**

Ordem de Serviço	
Numero:	**56**
Situação:	**EM ANDAMENTO**
Data de Abertura:	**31 / 10 / 2012 as 11:00**
Observações:	**31.10.212 - foi solicitado p/**

Dados do Cliente	
Cliente:	**THA - BALNEARIO FLORIDA - ENG. RICARDO**
Endereço:	**AV. PARANAGUA e ou AV. ATLANTICA**
Bairro:	**BALN. FLORIDA**
Ponto de Referência:	
Celular:	
Telefone:	**41-3453 - 92614003 - 3473.**
Cidade:	**MATINHOS**
CNPJ:	

Produtos e Serviços			
Quantidade	**Descrição**	**Valor Unitário**	**Subtotal**
0.0	alto av. atlatica para 01.11	**R$ 0,00**	**R$ 0,00**

Valor Total:	**R$ 0,00**
Desconto:	**-R$ 0,00**
Total com Desconto:	**R$ 0,00**

FIGURA 34 FONTE: AUTOR

Considerações

Ao utilizar as metodologias de análise e desenvolvimento de sistemas, foi possível obter uma analise bem detalhada do modelo de negocio dando sequencia ao processo de desenvolvimento sem obter maiores problemas.

Foi possível observar o quanto é importante utilizar metodologias para efetuar um desenvolvimento de sistemas de software com a menor taxa de erro possível. No projeto de análise de desenvolvimento de software, é fundamental a utilização das metodologias de análise e desenvolvimento, ao desenvolver os diagramas citados na metodologia, permite uma visão geral do sistema de software mesmo antes da codificação.

Este projeto foi desenvolvido utilizando a metodologia de análise e projeto, conhecida como ICONIX, esta metodologia se mostrou muito eficaz, por ser uma metodologia considerada simples em relação as metodologias RUP e XP.

Ao desenvolver um projeto de software sem possuir uma equipe de desenvolvimento, seria de grande complexidade se fosse utilizar uma metodologia muito complexa ou com uma documentação muito detalhada, como é o caso da metodologia RUP. Porém ao utilizar a metodologia ICONIX, foi possível projetar com uma precisão excelente o projeto de software, auxiliando muito no momento da codificação e implementação do sistema.

Em relação a linguagem de programação, foi utilizada a linguagem Ruby juntamente com o framework Rails. Esta ferramenta foi crucial para que-se obtivesse sucesso no sistema, e atendesse os requisitos não funcionais citados no capitulo IV. A linguagem de programação Ruby possui uma sintaxe muito amigável, o que auxiliou na codificação dos algorítimos do sistema, e o framework Rails tornou possível manter a organização do código, permitindo codificar o software sem que ouve-se fuga do modelo desenhado nos diagramas.

Após a análise, projeto e desenvolvimento do sistema, baseado no modelo de negócio, citado no capitulo II deste trabalho, não houve muita resistência da parte dos usuários do sistema, uma vez que eles auxiliarão a desenhar o software, que recebeu uma excelente aceitação por parte dos usuários, e necessitando de apenas pequenos ajustes finais para que o sistema de software ficasse totalmente funcional para a empresa em que foi implementado.

Hoje o sistema está funcionando normalmente na empresa em que foi implementado, está sendo utilizado nos seguintes sistemas operacionais; Android, Windows, Linux . O processo de implementação demorou em média de 30 dias, contando o processo de treinamento e ajustes finais, após este período o sistema se mostrou muito estável pois não houve a necessidade de suporte relacionado a bugs do sistema.

Todo este processo descrito neste trabalho, foram frutos dos estudos e conhecimentos adquiridos durante o projeto de aprendizagem. Que se tornou de fundamental importância para que se tornasse possível colocar em prática meus conhecimentos adquiridos durante o curso de graduação, permitindo assim fundamentar meus conhecimentos em analise, projeto e desenvolvimento de software.

REFERÊNCIAS BIBLIOGRÁFICAS

Extreme Programming and Rational Unified Process – Contrasts or Synonyms?. Disponível em: < http://www.acis.org.co/fileadmin/Curso_Memorias/Curso_CMMI_Sep06/Modulo -Product Engineering/xp_rup.pdf>. Acesso em: 26/08/2013.

Fabio Akita. Disponível em: <http://www.akitaonrails.com>. Acesso em: 17/09/2013.

José Anízio Maia. Construindo Softwares com Qualidade e Rapidez Usando ICONIX. **JugManaus**, http://www.guj.com.br, v.1, n.1, p.1-18, 2010. Disponível em: <http://www.guj.com.br/content/articles/patterns/iconix_guj.pdf>. Acesso em: 26/08/2013.

Kruchten, Philippe. **The rational unified process**: Computer software—Development.2.Software engineering.I.Titl. 2, ed. : 2000.

Luiz David. Disponível em: <http://engenhariadesoftware.info/downloads/JAD.ppt>. Acesso em: 17/09/2013.

Marina Martinez. Disponível em: <http://www.infoescola.com/engenharia-de-software/rup/>. Acesso em: 17/09/2013.

OLIVEIRA, JAYR FIGUEIREDO DE. **METODOLOGIA PARA DESENVOLVIMENTO DE PROJETOS DE SISTEMAS**: GUIA PRATICO. 5º : Erica, 2002.

Ryan Bates. Disponível em: <http://railscasts.com/>. Acesso em: 17/09/213.

S. Pompilho. **Livro Análise Essencial**: Guia Prático de Análise de Sistemas. 1º Edição : Ciência Moderna, 202. Disponível em: <2002>. Acesso em: 26/08/2013.

Vinícius Teles. Disponível em: <http://desenvolvimentoagil.com.br/xp/>. Acesso em: 17/09/2013.