

**SOCIEDADE EDUCACIONAL DE SANTA CATARIA – SOCIESC
INSTITUTO SUPERIOR TUPY – IST**

**GASFINDER: PROPOSTA DE UM PORTAL DE PESQUISA DE PREÇOS DE
COMBUSTÍVEIS WEB ON-LINE**

**ALEXANDRE DA SILVA MATEUS
WILLIAM DOS SANTOS DE OLIVEIRA**

**JOINVILLE
2011/02**

**ALEXANDRE DA SILVA MATEUS
WILLIAM DOS SANTOS DE OLIVEIRA**

**GASFINDER: PROPOSTA DE UM PORTAL DE PESQUISA DE PREÇOS DE
COMBUSTÍVEIS WEB ON-LINE**

É objetivo deste projeto, apresentar uma proposta de um Portal de Pesquisa de Preços de Combustível WEB On-line com base em conceitos de escalabilidade tecnológica e de acordo com as especificações da ANP - Agência Nacional de Petróleo.

PROF. MSc. LUIZ CAMARGO
Professor Orientador

JOINVILLE
2011/02

**ALEXANDRE DA SILVA MATEUS
WILLIAM DOS SANTOS DE OLIVEIRA**

**GASFINDER: PROPOSTA DE UM PORTAL DE PESQUISA DE PREÇOS DE
COMBUSTÍVEIS WEB ON-LINE**

Este trabalho foi julgado e aprovado em sua forma final, sendo assinado pelos professores da Banca Examinadora.

Joinville, __ de _____ de 2011.

Prof^a. MSc. Luiz Camargo – IST

Dedicamos esse trabalho a todos que se aventuram
em busca de novos conhecimentos.

AGRADECIMENTOS

Aos nossos pais, professores, colegas de trabalho, amigos.

Jovem, “lembra-te, pois, do teu
grandioso Criador, nos dias da tua idade viril,
antes que passem a vir os dias calamitosos ou
cheguem os anos em que dirás: Não tenho agrado neles.”
(REI SALOMÃO)

RESUMO

Apresentam-se neste trabalho uma proposta para um Portal de Pesquisa de Preços de Combustíveis WEB On-line, baseado nas informações fornecidas pela Agência Nacional de Petróleo (ANP). Utilizando conceitos como Scrum este projeto resultará em um *website* prestador de serviço, com o objetivo de atender uma necessidade que faz parte do cotidiano de muitos proprietários de veículos automotores: busca e comparação de preços de combustíveis. Terá como objetivo secundário proporcionar economia para os consumidores e empresas que necessitam abastecer seus veículos diariamente, independente da região onde estiver situado.

Palavras-chave: Portal. Combustíveis. Posto. Scrum. Java. Python. ANP.

ABSTRACT

Are presented in this paper a proposal for a Portal Search WEB Fuel Prices Online, based on information provided by the National Petroleum Agency (ANP). Using concepts such as Scrum this project will result in a website service provider, in order to meet a need that is part of everyday life for many owners of motor vehicles, search and price comparison of fuels. Secondary objective will provide savings for consumers and companies that need their vehicles every day, whether the region is situated.

Keywords: Portal. Fuel. Gas Station. Scrum. Java. Python. ANP.

LISTA DE ILUSTRAÇÕES

Figura 1 - Fases da Engenharia de Software	18
Figura 2 - Ciclo de Vida (Modelo Cascata).....	19
Figura 3 - Método de Prototipação	21
Figura 4 - Modelo Espiral	22
Figura 5 - Metodologia de Desenvolvimento Scrum	24
Figura 6 - Metodologia de Desenvolvimento XP	27
Figura 7 - Processo WebE	32
Figura 8 - Processo Engenharia de Requisitos	35
Figura 9 - Portal ANP	37
Figura 10 - Preço do combustível disponibilizado ao consumidor no Portal ANP	39
Figura 11 - UML	40
Figura 12 - Caso de Uso Geral.....	42
Figura 13 - Diagrama de Seqüência.....	43
Figura 14 - Diagrama de Entidade Relacionamento	44
Figura 15 - Diagrama de Classe.....	45
Figura 16 - Java	46
Figura 17 - MySQL	49
Figura 18 - Exemplo de Código CSS	50
Figura 19 - Exemplo de Código HTML	51
Figura 20 - Exemplo de Código JavaScript	51
Figura 21 - Hibernate.....	52
Figura 22 - Exemplo de Código JSON	55
Figura 23 - Exemplo de Código Python.....	56
Figura 24 - Eclipse	57
Figura 25 - Jude	57
Figura 26 - Apache.....	58
Figura 27 - Matriz de Rastreabilidade de Requisitos.....	63
Figura 28 - Matriz de Rastreabilidade de Dependência	63
Figura 29 - Matriz de Rastreabilidade de Módulos do Sistema	64
Figura 30 - Matriz de Rastreabilidade de Usuário	64

Figura 31 - Wireframe - Home	69
Figura 32 - Wireframe - Contato.....	70
Figura 33 - Wireframe - Quem somos	71
Figura 34 - Wireframe - Login.....	72
Figura 35 - Wireframe - Pesquisa na lista de Postos	73
Figura 36 - Wireframe - Sem resultados para pesquisa	74
Figura 37 - Wireframe - Pesquisa diretamente no mapa	75
Figura 38 - Wireframe - Manutenção de Preços.....	76
Figura 39 - Wireframe - Informações dos Postos	77
Figura 40 - Wireframe - Erro genérico	78

LISTA DE TABELAS

Tabela 1 - Consulta SQL	48
-------------------------------	----

LISTA DE SIGLAS

- AJAX** Asynchronous Javascript and XML
- ANP** Agência Nacional do Petróleo, Gás Natural e Biocombustíveis
- CASE** Computer-Aided Software Engineering
- CSS** Cascading Style Sheets
- ERP** Enterprise Resource Planning
- HAAS** Hardware as a Service
- HTML** HyperText Markup Language
- IIS** Internet Information Services
- IOC** Inversion of Control
- IU** User interface
- JCP** Java Community Process
- JSF** JavaServer Faces
- JSON** JavaScript Object Notation
- OOHDM** Object Oriented Hypermedia Design Method
- PHP** Hypertext Preprocessor
- SAAS** Software as a service
- SGDB** Sistema de Gerenciamento de Banco de Dados
- WEB** World Wide Web
- WEBAPP** Web Based Application
- WEBE** Web Engineering
- XP** eXtreme Programming

SUMÁRIO

1 INTRODUÇÃO	16
2 EMBASAMENTO TEÓRICO	17
2.1 ENGENHARIA DE SOFTWARE	17
2.1.1 Ciclo de Vida.....	18
2.1.2 Prototipação.....	20
2.1.3 Modelo em Espiral	21
2.2 DESENVOLVIMENTO ÁGIL	23
2.2.1 Scrum	24
2.2.1.1 Fases do Scrum.....	25
2.2.2 Extreme Programming.....	27
2.3 ENGENHARIA DE SOFTWARE PARA WEB	31
2.3.1 Metodologia	32
2.3.2 OOHDM	33
2.3.3 Vantagens do OOHDM	34
2.4 ENGENHARIA DE REQUISITOS.....	34
2.5 SAAS – SOFTWARE COMO SERVIÇOS	35
2.6 AGÊNCIA NACIONAL DO PETRÓLEO (ANP)	36
3 TECNOLOGIAS APLICADAS NO DESENVOLVIMENTO DO GASFINDER	38
3.1 DATA SCRAPING	38
3.2 UML	39
3.2.1 Casos de Uso	41
3.2.2 Diagrama de Seqüência	42
3.2.3 Diagrama de Entidade Relacionamento	44
3.2.4 Diagrama de Classe	45
3.3 JAVA.....	46
3.4 MYSQL	47
3.5 CSS	49
3.6 HTML	50
3.7 JAVASCRIPT	51

3.8	FRAMEWORKS	52
3.8.1	Hibernate	52
3.8.2	JSF	53
3.8.3	Spring	53
3.9	JSON	54
3.10	PYTHON	55
3.11	ECLIPSE	56
3.11.1	Jude Community	57
3.11.2	Tomcat	58
4	DESENVOLVIMENTO DO GASFINDER	59
4.1	ESCOPO DO SISTEMA	59
4.1.1	Descrição Geral	60
4.1.2	Funções do Produto	60
4.1.3	Características dos Usuários	60
4.1.4	Restrições Gerais	61
4.1.5	Assertivas e Dependências	61
4.1.6	Requisitos de Usuário	61
4.1.7	Requisitos de Interface	62
4.1.8	Requisitos de Domínio	62
4.1.9	Matriz de Rastreabilidade de Requisitos	63
4.1.10	Matriz de Rastreabilidade de Dependência	63
4.1.11	Matriz de Rastreabilidade de Módulos do Sistema	64
4.1.12	Matriz de Rastreabilidade de Usuário	64
5	CONCLUSÃO	65
	REFERÊNCIAS	66
	ANEXO	68
	ANEXO 1 – WIREFRAME - HOME	69
	ANEXO 2 – WIREFRAME – CONTATO	70
	ANEXO 3 – WIREFRAME – QUEM SOMOS	71
	ANEXO 4 – WIREFRAME – LOGIN	72
	ANEXO 5 – WIREFRAME – PESQUISA NA LISTA DE PRODUTOS	73
	ANEXO 6 – WIREFRAME – SEM RESULTADOS PARA PESQUISA	74

ANEXO 7 – WIREFRAME – PESQUISA DIRETAMENTE NO MAPA.....	75
ANEXO 8 – WIREFRAME – MANUTENÇÃO DE PREÇOS.....	76
ANEXO 9 – WIREFRAME – INFORMAÇÕES DOS POSTOS	77
ANEXO 10 – WIREFRAME – ERRO GENÉRICO	78

1 INTRODUÇÃO

Atualmente todas as pessoas e empresas buscam alternativas que apresentem a melhor relação custo-benefício, indiferente do produto ou serviço. Sempre selecionamos na maioria das vezes aquelas soluções que oferecem “mais por menos” (leia-se: mais vantagens com menor custo). Para tanto, entram em ação diversos mecanismos para proporcionar o melhor investimento: consulta e comparação de preços, avaliação da qualidade, pesquisa sobre a opinião do produto/serviço em um grupo de adquirentes, realização de orçamento, barganha, entre outros.

Tudo isso é praticando constantemente na aquisição de novos produtos, principalmente na Internet através de ferramentas e serviços que facilitam a comparação de preço de produtos em geral. Paradoxalmente quando o assunto é custo fixo, a busca pelo melhor custo-benefício é sensivelmente alijada.

O combustível utilizado diariamente nos automóveis tornou-se mais um dentre os inúmeros custos fixo que uma pessoa ou empresa possa ter. E são poucos que realizam uma busca pelo melhor preço ou economia no momento de abastecer. Entendemos que talvez que isso ocorra pela falta de tempo das pessoas, somado pela ausência de ferramentas adequadas, pode ser também pelo pleno desconhecimento ou desinteresse sobre a economia que é possível realizar nesta atividade cotidiana.

O objetivo deste projeto é propor um Portal que ofereça como serviço a comparação dos preços dos combustíveis, sinalizando o melhor local para o abastecimento e viabilizando a busca pela economia indiferente onde se situa o cliente. Para exercer este serviço, adotaremos ferramentas WEB, utilizando conceitos ágeis e adaptáveis para todos que utilizarem deste serviço.

Para atingir estes objetivos, os tópicos apresentados estão situados da seguinte forma:

- Embasamento teórico: Compreende a Engenharia de Software
- Tecnologias aplicadas: Ferramentas de desenvolvimento
- Desenvolvimento do aplicativo: Definição de requisitos e projeto

2 EMBASAMENTO TEÓRICO

2.1 ENGENHARIA DE SOFTWARE

A definição de Engenharia de Software segundo Pressman (1995, p. 31) é “o estabelecimento e uso de sólidos princípios de engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquinas reais”.

A Engenharia de Software possui ferramentas que possibilita ao desenvolvedor ou engenheiro de software um maior controle do processo de forma geral, garantindo em sua construção a qualidade de Software, através das fases específicas conforme é possível observar na Figura 1.

Ainda segundo Pressman (1995, p. 31-32), os três elementos fundamentais da Engenharia de Software são descritos conforme abaixo:

- **Métodos:** Fornecem os detalhes de como fazer para construir um sistema. Abrangem um vasto conjunto de tarefas, que são: planejamento e estimativa de projeto, análise de requisitos de software e de sistemas, projeto da estrutura de dados, arquitetura de programa e algoritmo de processamento, codificação, teste e manutenção.
- **Ferramentas:** Oferecem apoio automatizado aos métodos. Existem ferramentas para auxiliar cada um dos métodos citados, sendo que, quando uma pode compartilhar informação criada por outra, é estabelecido um sistema de suporte ao desenvolvimento de software que se chama Engenharia de Software Auxiliada por Computador (CASE – Computer-Aided Software Engineering), também abordado neste projeto.
- **Procedimentos:** Elo entre métodos e ferramentas que permite o desenvolvimento do sistema. Através dos procedimentos, entre outras coisas, é definida a seqüência de aplicação dos métodos a serem utilizados.

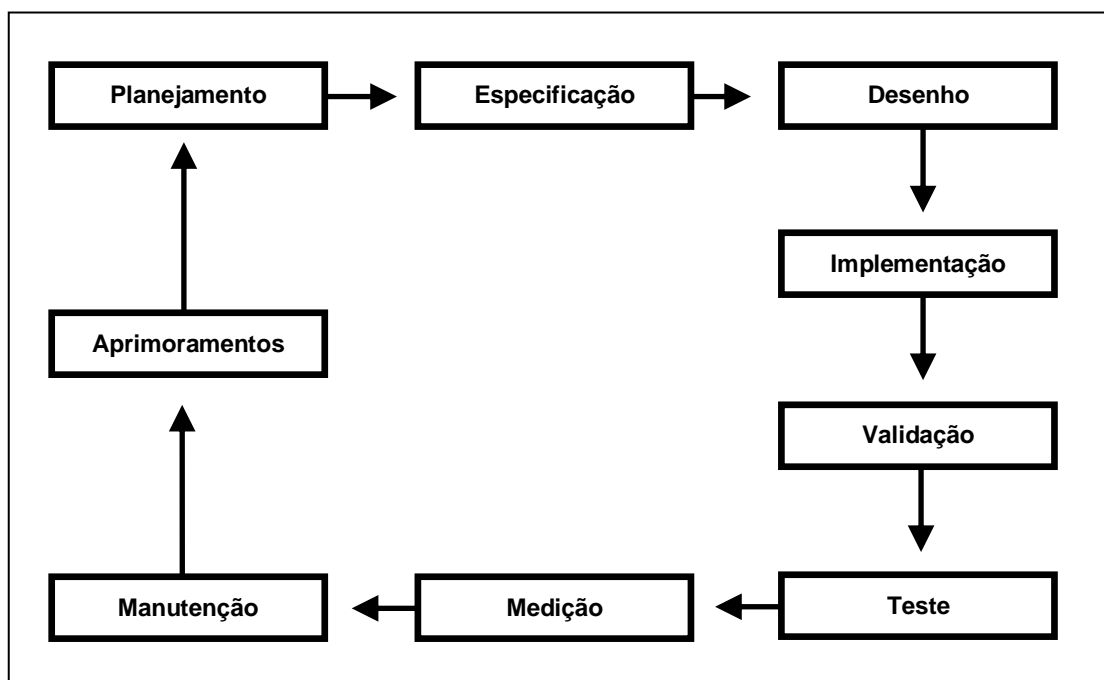


Figura 1 - Fases da Engenharia de Software

Atualmente a Internet atua como um canal de divulgação fundamental para as empresas, influencia na implementação de processos de negócio, comunicação e integração entre sistemas e, principalmente na divulgação de produto ou serviços.

Diante desta situação surge uma abordagem da Engenharia de Software voltada especificamente para a WEB que segue alguns padrões da Engenharia de Software convencional, porém, dando maior ênfase a design de interface, navegabilidade e conteúdo. (ARAÚJO, 2001).

Falaremos mais adiante sobre a Engenharia de Software para WEB.

2.1.1 Ciclo de Vida

Também conhecido como modelo cascata, o ciclo de vida é apresentado por Pressman (1995, p. 32) como um “paradigma que exige uma visão sistêmica e seqüencial do software. Sua estrutura é demonstrada pelo início no nível do sistema, de onde avança para as fases de análise, projeto, codificação, teste e manutenção”, conforme demonstra a Figura 2.

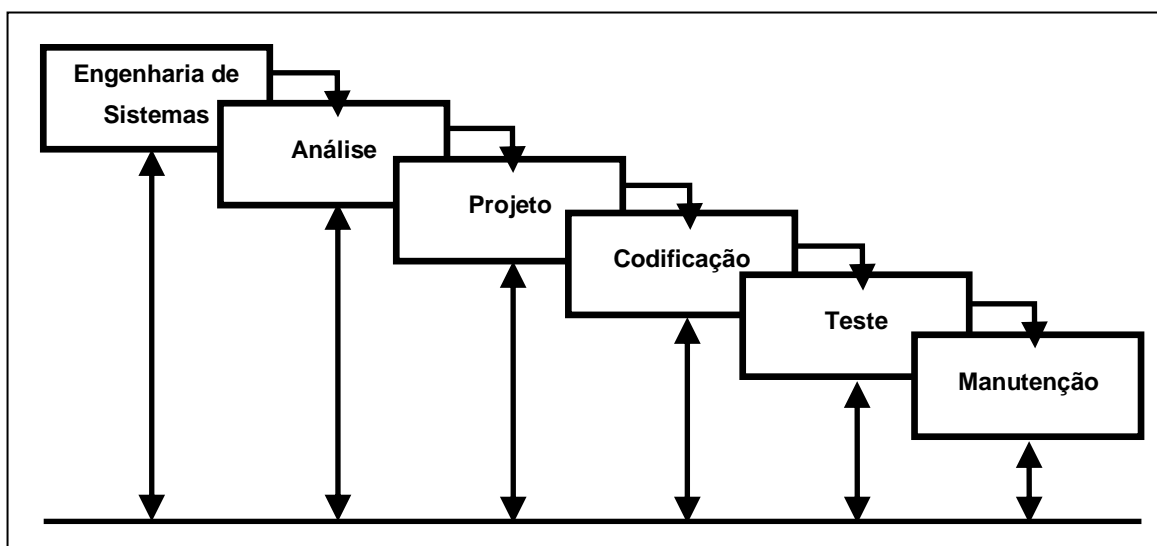


Figura 2 - Ciclo de Vida (Modelo Cascata)

Fonte: (PRESSMAN, 1995)

Este modelo, considerado clássico e antigo de desenvolvimento, possui as seguintes fases:

- Engenharia de sistemas: estabelecimento dos requisitos para todos os elementos do sistema e atribuição de subconjunto destes requisitos ao software.
- Análise de requisitos de software: intensificação na coleta de requisitos e foco específico no software.
- Projeto: processo de estudo e documentação de quatro atributos do programa: estrutura de dados, arquitetura de software, detalhes procedimentais e caracterização de interface.
- Codificação: fase em que o projeto é traduzido para uma forma legível para máquina. Quanto maior for o nível de detalhamento do projeto, menor é o tempo de execução desta fase.
- Teste: após a geração do código realizam-se os testes, que consideram os fundamentos lógicos internos, garantindo testes de todas as funcionalidades e dos aspectos funcionais externos do software, identificando erros e que a informação fornecida produza resultados que correspondam aos resultados exigidos.

- Manutenção: após ser entregue ao cliente, sem dúvida modificações serão solicitadas. Sejam por conta de erros identificados, adaptações ao meio externo ou aumento de funcionalidades.

2.1.2 Prototipação

Ocorre quando o projeto é criado de forma rápida, onde a construção do protótipo poderá ser avaliada pelo cliente e após essa avaliação sofrer pequenas alterações para que o software criado atenda as soluções que foram estabelecidas.

A prototipação possui alguns riscos clássicos: O cliente visualiza o protótipo e tem sensação de urgência para colocar em produção, não considerando a qualidade geral do sistema. Em muitos casos a equipe de desenvolvimento também deseja colocar o protótipo em execução muito rápido, com isso, é possível que um sistema operacional ou linguagem de programação inadequada seja empregado, meramente porque é a solução mais próxima a disposição do projeto.

Não somente, quando o cliente não define detalhadamente os requisitos de entrada, processamento e saída do software desejado, e simplesmente aborda um conjunto de objetivos gerais ou ainda, quando o desenvolvedor não possui segurança sobre a eficiência, adaptabilidade, forma de interação homem-máquina, entre outros fatores, este método pode ser o mais indicado. (PRESSMAN, 1995, p. 35).

Um projeto será eficaz quando este importante princípio for atendido e compreendido desde o início e não conduzido pelo cliente ao final do projeto.

A iteração ocorre quando a idéia do que está sendo solicitado é compreendida pelo desenvolvedor, e a solução proposta satisfaz o cliente. (PRESSMAN, 1995, p. 36).

A Figura 3 representa o método de prototipação em uma interação contínua que visa o constante refinamento do protótipo até a concepção de uma solução adequada.

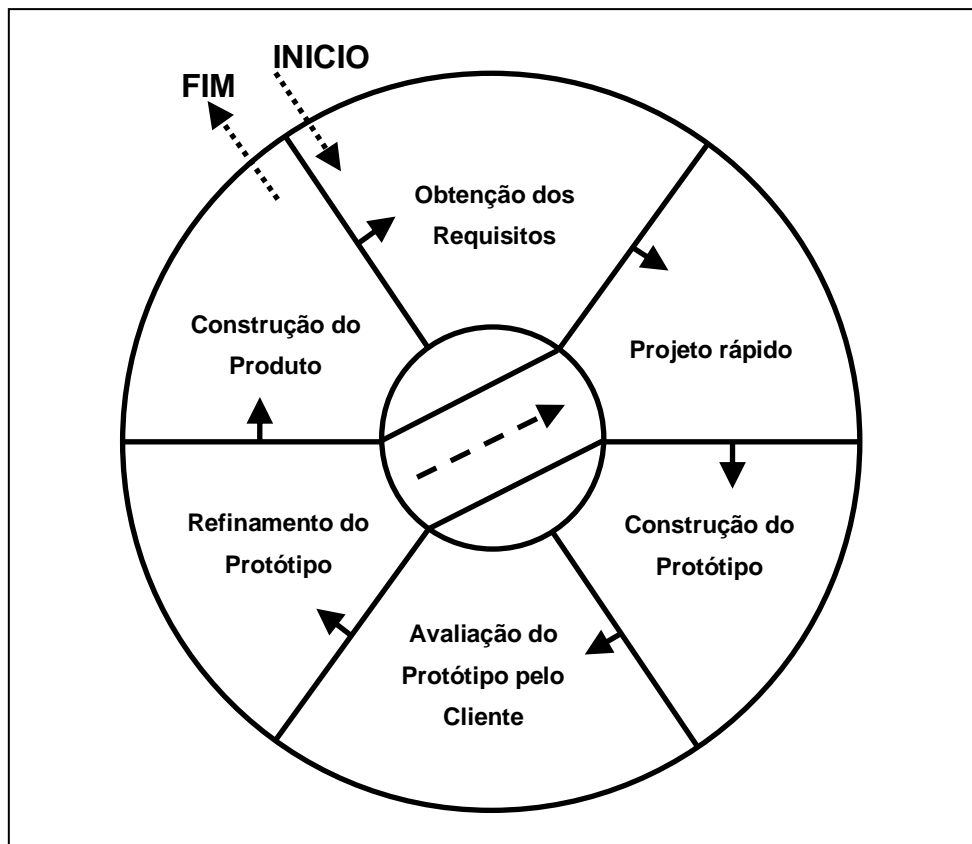


Figura 3 - Método de Prototipação

Fonte: (PRESSMAN, 1995)

2.1.3 Modelo em Espiral

Nesse processo de desenvolvimento segundo Pressman (1995, p. 40), é a “abordagem mais realística no processo de desenvolvimento de software, pois abrange as melhores características do ciclo de vida clássico e da prototipação, acrescentando a análise de riscos”.

Este modelo é definido por quatro importantes atividades, utiliza-se da prototipação para a redução de riscos e a abordagem sistêmica sugerida pelo ciclo de vida. (PRESSMAN, 1995, p. 38-40):

- Planejamento: definição de objetivos, alternativas e restrições.
- Análise de Riscos: análise de alternativas e resolução dos riscos.
- Engenharia: desenvolvimento do produto.
- Avaliação do cliente: avaliação dos resultados da engenharia.

A representação gráfica do modelo espiral é feita conforme Figura 4.

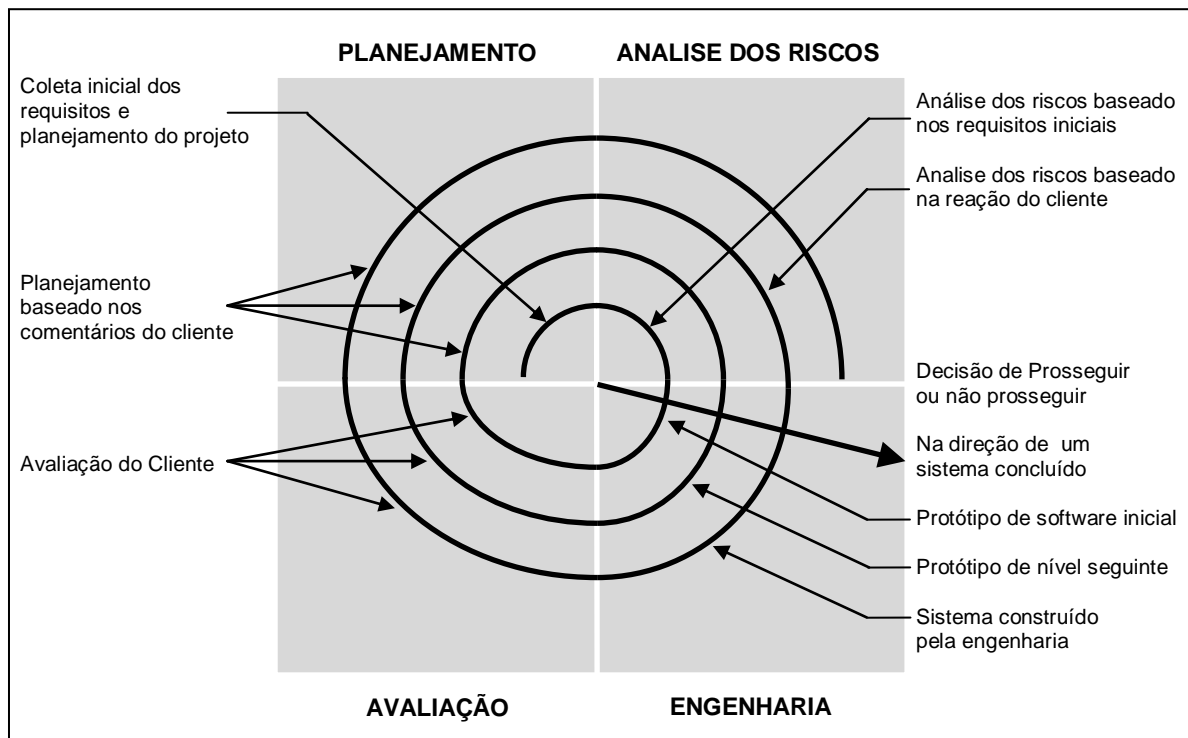


Figura 4 - Modelo Espiral
Fonte: (PRESSMAN, 1995)

2.2 DESENVOLVIMENTO ÁGIL

Segundo Pressman (2006, p.58), “a engenharia de software ágil combina uma filosofia e um conjunto de diretrizes de desenvolvimento. A filosofia encoraja a satisfação do cliente e a entrega incremental do software logo de início, equipes de projeto pequenas, altamente motivadas, métodos informais, produtos de trabalho de engenharia de software mínimos e simplicidade global de desenvolvimento. As diretrizes de desenvolvimento enfatizam a entrega em contraposição a análise e ao projeto (apesar dessas atividades não serem desencorajadas) e a comunicação ativa entre desenvolvedores e clientes”.

Os princípios do desenvolvimento ágil visam minimizar as falhas no desenvolvendo o projeto de forma incremental, em curtos períodos, chamados de iteração, sendo esses mini-projetos de software que incluirão as tarefas necessárias para alcançar a funcionalidade do produto: planejamento, análise de requisitos, projeto, codificação, teste e documentação.

Esse processo prioriza pessoas e iterações com processos e ferramentas, software executável, no lugar de uma documentação ampla e, principalmente a colaboração do cliente e resposta rápidas às alterações no escopo durante o andamento do projeto.

A motivação é um dos focos de tais modelos, as equipes de negócio e desenvolvimento trabalham juntas diariamente, deve ser dado o ambiente e o apoio necessário aos indivíduos para que eles confiem na realização do trabalho. Ter o software funcionando é a principal medida de progresso. A equipe deve receber bem as alterações nos requisitos, mesmo estando em um nível adiantada de desenvolvimento, pois o direcionamento dessas mudanças constitui em um diferencial competitivo para o cliente.

2.2.1 Scrum

Trata-se de uma metodologia para gerenciamento de projetos, também chamado de framework para desenvolvimento ágil, onde seu objetivo principal é a agilidade do processo. E para isto, tem como premissa, não fazer novas negociações e entregar sempre no prazo proposto. E se por ventura surgirem imprevistos o prazo poderá ser revisto dentro do projeto, mas isto não impedirá que o produto/serviço seja entregue no prazo.

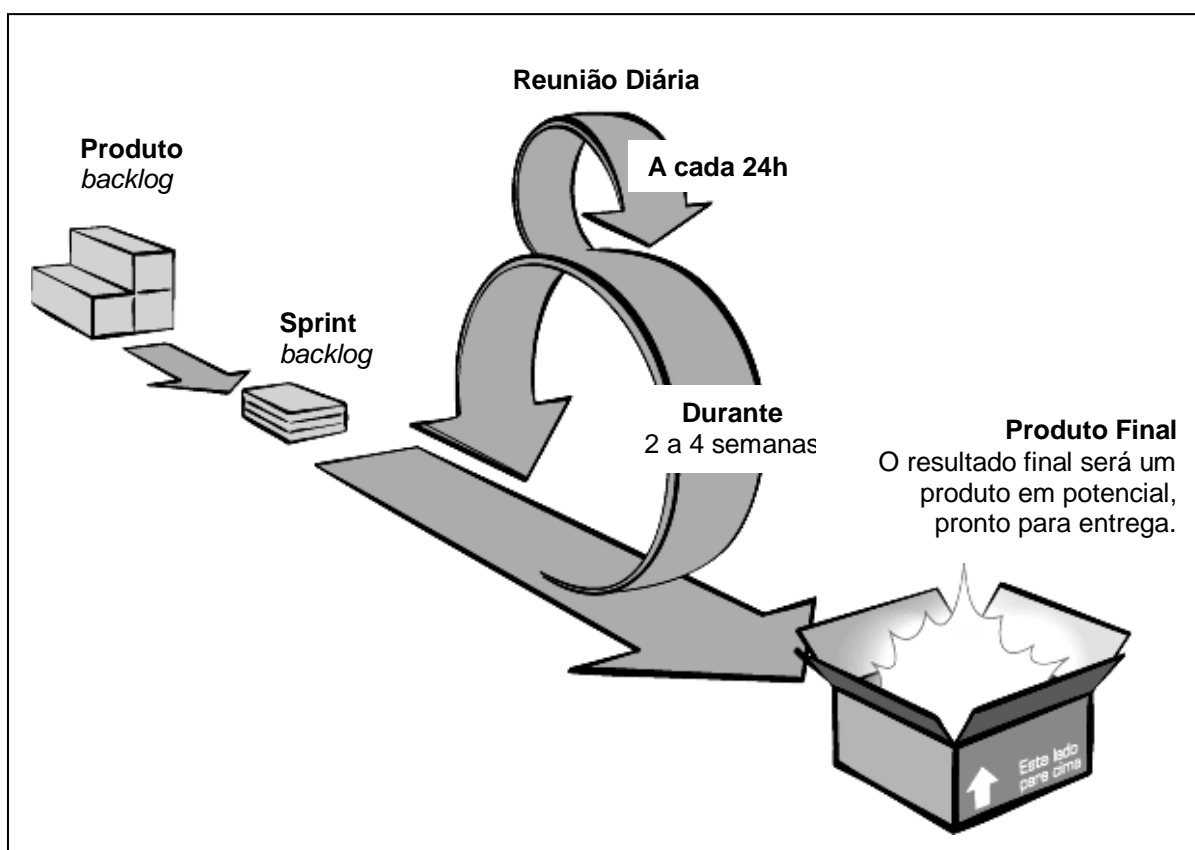


Figura 5 - Metodologia de Desenvolvimento Scrum

A metodologia Scrum segue o processo conforme demonstra a Figura 5. Segundo Larman (2003), o Scrum, é um nome oriundo do esporte rúgbi, teve suas origens em companhias nipônicas no início de 1986 por Takeuchi e Nonaka, mas começou a ser usado mais recentemente apenas em 1996. O Scrum baseia-se em

técnicas que buscam agilizar e deixam o processo de desenvolvimento e gerenciamento mais flexível.

A conceito principal do Scrum é que a produção de produtos de softwares envolve muitas inconstantes técnicas e variáveis de ambiente, como tecnologia e recursos, que podem se transformar durante o processo. Isto cria um processo de desenvolvimento aleatório e implexo, demandando muita flexibilidade para seguir as mudanças. O resultado do processo deve ser um software que é realmente útil para o cliente (SCHWABER, 1996).

No desenvolvimento deste Portal de Preços de Combustíveis, demonstrará a agilidade em sua execução, utilizando Scrum como metodologia de gestão.

2.2.1.1 Fases do Scrum

A seguir apresenta-se uma descrição das atividades a serem realizadas em cada fase do Scrum (SCHWABER, 1996):

Planejamento:

- Desenvolvimento claro e objetivo de uma lista de atividades do produto;
- Definição da data de *release* e funcionalidades de um ou mais *sprints*;
- Definição do *release* apropriado para começar o ciclo de desenvolvimento;
- Mapeamento e estimativa das atividades;
- Definição do time do projeto;
- Avaliação e controle de riscos;
- Avaliação das ferramentas de desenvolvimento e infra-estrutura;
- Estimativa de custos.

Arquitetura:

- Revisão e possíveis ajustes na lista de atividades do produto;
- Identificação das mudanças necessárias para implementar as atividades do produto;
- Realizar a análise de domínio;
- Refinar a arquitetura do sistema;
- Identificação de possíveis problemas ou impedimentos na implementação dos requisitos;
- Reunião de revisão do design, onde são apresentados cada proposta de implementação de cada item do *backlog*.

Desenvolvimento (Sprint):

- Reunião de planejamento do *sprint*, a ser realizada sempre no primeiro dia de cada *sprint*. Essa reunião deve definir as atividades a serem incluídas na iteração corrente.
- Reuniões diárias com os membros da equipe para revisar o andamento do projeto;
- Revisão e ajustes nos requisitos do projeto;
- *Sprints* iterativos até que o produto seja considerado pronto para a entrega.

2.2.2 Extreme Programming

Programação extrema (do inglês eXtreme Programming), ou simplesmente XP, é uma metodologia ágil para equipes pequenas ou médias que buscam desenvolver software com requisitos imprecisos ou em constante transformação. Para isso, segue a estratégia de acompanhamento permanente e cumprimento de vários ajustes pontuais ao decorrer do desenvolvimento de software.

Os cinco valores fundamentais da metodologia XP são: simplicidade, *feedback*, comunicação, respeito e coragem. A partir desses valores, possui como princípios básicos: *feedbacks* pontuais, ponderar simplicidade, alterações incrementais, incentivar mudanças e trabalho com qualidade.

Entre os artefatos da gestão de projetos (escopo, tempo, qualidade e custo), existe um foco constante no item escopo. Para isso, aconselha-se a priorização de funções que agregue mais valor para o negócio. Desta forma, quando ocorrer a necessidade de reduzir o escopo, as funcionalidades com menos importância terão o desenvolvimento postergado ou até mesmo cancelado. A Figura 6 demonstra o processo da metodologia de desenvolvimento XP.

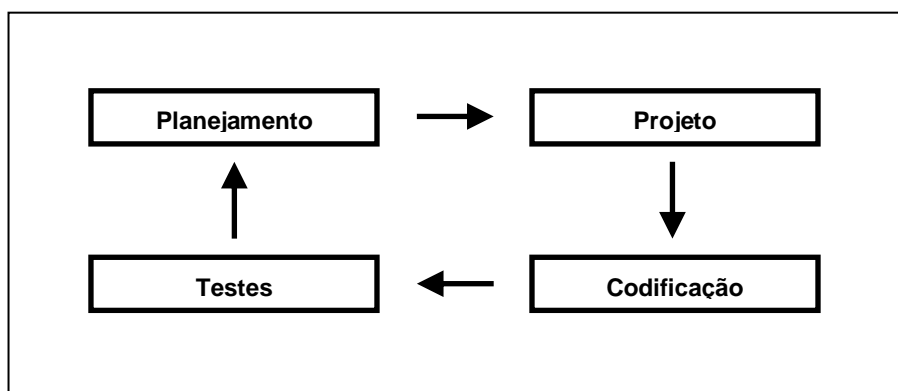


Figura 6 - Metodologia de Desenvolvimento XP

A XP estimula o domínio sobre qualidade do projeto, pois qualquer ganho razoável na produtividade, se reduzir a qualidade, então não poderá compensar as futuras perdas (ou até limitações) a médio e longo prazo.

Para colocar em prática os princípios e valores ao decorrer de um projeto de software, existe uma confiança significativa quanto a sinergia, onde os possíveis pontos fracos do projeto tendem a ser superados pelos pontos fortes. O XP ainda sugere uma lista de práticas, conforme vejamos a seguir.

- **Jogo de planejamento:** O desenvolvimento ocorre através de uma interação a cada 1 (uma) semana. Logo no começo da semana, os programadores juntamente com cliente fazem uma reunião para eleger quais itens serão desenvolvidos primeiro. Neste encontro, o cliente sinaliza suas prioridades e por sua vez, os programadores analisam a viabilidade da entrega. Além de o cliente ser uma peça fundamental neste ciclo, ele consegue obter a vantagem de sempre estar alinhado com o que está ocorrendo no projeto. Não somente, o projeto é acordado por um contrato com escopo variável, muito diferente de outras formas de contratos convencionais para projetos de software. Ao completar duas semanas, o cliente já possuiu um grupo de funcionalidades novas, que já foram testadas e estão disponíveis para serem colocadas em execução.
- **Pequenas versões:** A entrega de versões pontuais e plenamente funcionais durante o desenvolvimento do projeto propicia uma aceitação amistosa pelo cliente, que tem a chance de ir testando e validando pequenas partes do projeto em que está investindo.
- **Metáfora:** Busca promover um bom entendimento das necessidades do o cliente. Para um cliente, a definição do que é um sistema eficaz, por exemplo, pode ser ligeiramente distinto do conceito aceito por um programador. Portanto, é necessário compreender o vocabulário do cliente para decifrar aquilo que ele realmente deseja como resultado de um projeto.

- Projeto simples: Simplicidade é um dos conceitos fortes da metodologia XP. Projeto simples significa afirmar que quando o cliente solicitar que logo numa primeira versão do sistema, esteja disponível apenas o usuário "cliente" com uma senha "123", então o desenvolvedor deve providenciar exatamente esta funcionalidade, sem investir mais tempo com outras preocupações e validações.
- Equipe coesa: Nada mais é do que uma equipe de programadores que trabalha em conjunto com o cliente.
- Testes de aceitação: São testes específicos, aqueles que o próprio cliente sugere ou julga como importante para validar determinada funcionalidade do sistema. Estes testes são discutidos e definidos entre o cliente e a equipe de desenvolvimento e teste.
- Ritmo sustentável: É um fator onde os programadores busca trabalhar em uma cadencia que possibilite extrair qualidade e produtividade simultaneamente, preferencialmente sem horas extras, exceto quando as horas adicionais oferecem produtividade no desenvolvimento do projeto. Para atingir este objetivo é indispensável que exista um ambiente de trabalho amigável, com equipes bem motivadas e em sinergia.
- Reuniões em pé: Reuniões em pé visam evitar que não se perca o foco nos tópicos da reunião, produzindo, portanto reuniões de curta duração, utilizando apenas o tempo necessário para abordar as tarefas pendentes e as tarefas concluídas pela equipe.
- Posse coletiva: Esta prática defende que o código-fonte não deve possuir um proprietário, autor ou dono e, muito menos que qualquer desenvolvedor necessite solicitar autorização para modificar o sistema. O alvo desta didática é permitir que todos os envolvidos no projeto possuam conhecimento global sobre os mecanismos do sistema.

- **Programação em pares:** Ocorre quando dois programadores trabalham em um mesmo computador. Comumente, estes pares são formados com uma pessoa iniciante e outro programador proficiente na linguagem, por sua vez atuando como tutor. Nestes casos, geralmente é o programador iniciante que codifica o sistema enquanto o tutor auxilia no desenvolvimento. As vantagens são que o código-fonte sempre é revisado por duas pessoas, reduzindo consideravelmente as condições para erros no sistema. Através desta prática, é possível aliar o crescimento das habilidades da equipe ao mesmo tempo em que é gerado um código-fonte de qualidade.
- **Padrões de Codificação:** Os analistas e programadores devem determinar padrões de desenvolvimento, com regras explícitas as quais todos devem seguir. Com isto, o código-fonte fica mais legível, de fácil compreensão por novos programadores, facilita a manutenção e a continuidade do desenvolvimento, indiferente do tamanho da equipe.
- **Desenvolvimento orientado a testes:** Inicialmente são criados testes unitários e em seguida é gerado o código-fonte para que o sistema execute. Este investimento mais implexo no início, é naturalmente contrario a muitos outros métodos de desenvolvimento e, naturalmente pode gerar algum nível de resistência por parte dos programadores. Contudo é preciso elucidar que através desta prática, é construído um pilar essencial para manter a qualidade do projeto.
- **Refatoração:** Aumenta a clareza e entendimento do código-fonte através da construção de módulos lógicos que fornecem maior aproveitamento do código-fonte, evitando principalmente a duplicação de código-fonte. É uma prática que busca a melhoria progressiva da programação, minimizando as possibilidades de erros e conservando a compatibilidade do código atual.

- Integração contínua: Sempre ao finalizar novas *releases*, é indicado integrar os novos módulos preferencialmente na mesma semana junto a versão em produção. Caso esta janela de tempo seja muito grande, poderá aumentar a predisposição de conflitos entre as versões e outros problemas de compatibilidade.

2.3 ENGENHARIA DE SOFTWARE PARA WEB

A Engenharia de Software para WEB surge com uma solução específica para o crescente avanço do desenvolvimento de aplicações para WEB - também conhecida como WebApp - visto sua complexidade, similar e senão superior a projetos *off-line*. As WebApps evoluíram consideravelmente e atualmente oferecem soluções integradas com ERPs, sistemas legados, bancos de dados diversos em praticamente todos sistemas empresariais. (PRESSMAN, 2006).

Praticamente todas as aplicações WebApps compartilham de um mesmo conjunto de atributos e ao mesmo tempo, limitações:

- Suporte a redes: Acesso local ou Internet.
- Concorrência: Capacidade de aceitar acesso simultâneo.
- Carga inesperada: Em função de ataques de hackers, por exemplo.
- Desempenho: Tempo de processamento e resposta da aplicação.
- Disponibilidade: Período em que os serviços estão *on-line*.
- Orientada a objetos: Suporte a banco de dados e linguagem de programação moderna.
- Conteúdo: Oferecer o conteúdo adequado aos usuários.
- Evolução constante: Desenvolvimento e melhoria continua da aplicação.
- Segurança: Com objetivo de abrigar todo conteúdo privilegiado.
- Design: Característica visual e de interação com usuário.

A engenharia de software para WebApps pode ser definida como o processo que possui enfoque sistêmico e rigoroso para o desenvolvimento de soluções WEB (GRAEF; GAEDKE, 2000).

2.3.1 Metodologia

O processo WebE (Engenharia WEB) utiliza o conceito de desenvolvimento ágil que utiliza entregas incrementais do aplicação. Conforme é possível observar na Figura 7, o processo da WebE possui as seguintes etapas abaixo:

- Entregas incrementais: Todas as atividades serão repetidas continuamente a cada ciclo de incremento que ocorre quando uma *release* é entregue.
- Modificações contínuas: As alterações podem ocorrer em função de avaliação da *release* entregue ao cliente, ou até mesmo em consequência de alterações de escopo.
- Documentação: Em vista de prazos enxutos, é necessário redigir estratégias específicas para documentar as atividades da engenharia. Indiferente das limitações é preciso haver um acordo, pois as etapas de projeto, análise e testes precisam ser documentadas de alguma forma.

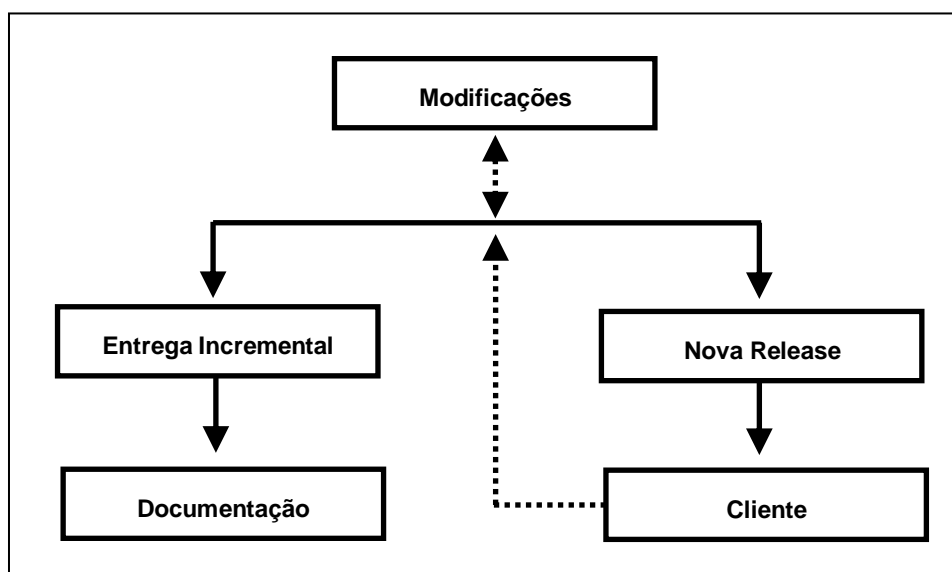


Figura 7 - Processo WebE

2.3.2 OOHDM

As WebApps continuam evoluindo e aumentando sua complexidade. Contudo, os artifícios e os métodos clássicos da Engenharia de Software ainda não são totalmente apropriados para o desenvolvimento WEB, que possui características próprias.

Segundo Rossi, Schwabe e Lyardet (1999), as principais distinções entre uma WebApp e aplicações *off-line* clássicas giram sobre pontos que envolvem a interação, interface e desenvolvimento.

De qualquer modo, uma WebApp pode ser considerado um produto de software, e portanto, várias técnicas tem sido empregadas no seu desenvolvimento ao mesmo tempo que são propostos diferentes metodologias visando sistematizar as etapas de análise, de projeto e, de desenvolvimento, focando em modelagem de domínio e também na definição da estrutura e da navegação.

Entre os inúmeros modelos empregados atualmente para especificação de WebApps, o método OOHDM tem se adaptado como método mais indicado, em função de sua ampla utilização em projetos WEB ao redor do mundo.

O desenvolvimento de um WebApp, compreende as seguintes etapas principais:

- Especificação da aplicação WEB por meio de uma técnica de projeto.
- Desenvolvimento da solução através de uma linguagem de programação adotada.

No desenvolvimento de WebApps, é necessário adotar um modelo básico, que visa proporcionar a evolução do projeto durante o desenvolvimento. Ao selecionar o OOHDM durante a etapa inicial de especificação de aplicação WEB, é previsto quatro passos que devem ser definidos:

- Modelo conceitual: Define a semântica e o domínio da aplicação.
- Projeto da navegação: Considera o perfil do usuário e a tarefa alvo, enfatizando aspectos cognitivos.
- Projeto da interface abstrata: Define os artefatos perceptíveis, metáforas e descreve a interface.

- Implementação: Refere-se ao desenvolvimento da solução, classes, sub-sistemas e demais artefatos devem ser instanciados.

2.3.3 Vantagens do OOHDM

Soluções projetadas e desenvolvidas em torno do OOHDM tendem a ser mais fáceis de usar, manter e escalar. É possível desenvolver novas aplicações apenas reutilizando componentes criados anteriormente.

2.4 ENGENHARIA DE REQUISITOS

Foi criada uma área específica dentro da Engenharia de Software, que segundo Azevedo Junior e Campos (2008), visa aplicar técnicas de engenharia em métodos de definição e análise de requisitos para garantir o atendimento das necessidades de informatização de processos através do software projetado.

Leite (1994), afirma que a Engenharia de Requisitos é a disciplina que procura sistematizar o processo de definição de requisitos. Estabelece o processo de definição de requisitos, no qual deve fazer a elicitação, modelagem e análise dos requisitos. As fases da Engenharia de Requisitos estão organizadas da seguinte forma:

- Elicitação: Tarefa de identificação dos casos que compõem os requisitos do sistema, visando proporcionar um entendimento do sistema de forma mais completa e correta.
- Modelagem: Consiste na representação dos fatos extraídos da etapa anterior num modo sistemático e através de técnicas específicas.
- Análise: É o processo de verificação e validação do resultado da modelagem, ou seja, é a análise da especificação.

Com base nesta afirmação, observamos que este processo busca variados métodos para definição de requisitos deste projeto. Pressman (2002) sugere as etapas da Engenharia de Requisitos conforme a Figura 8.

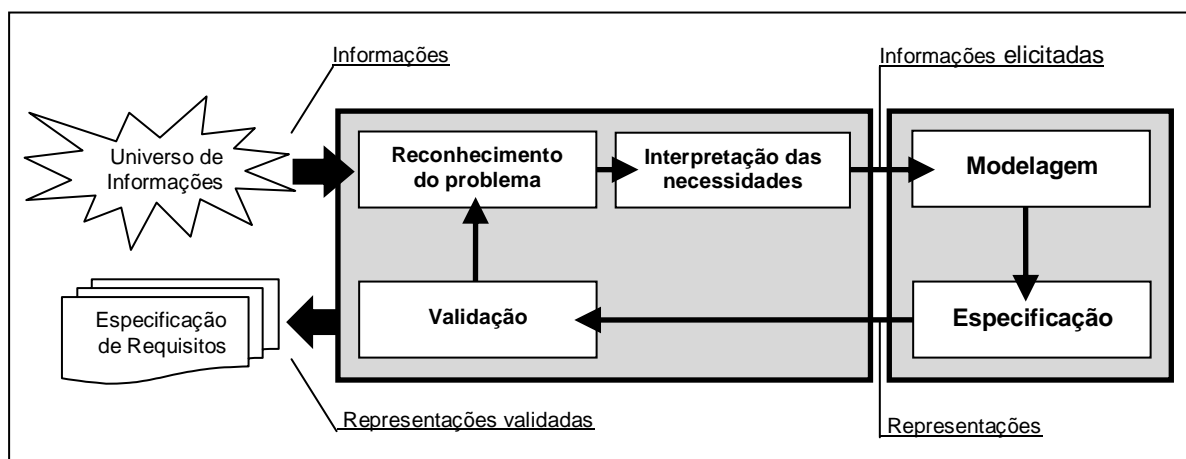


Figura 8 - Processo Engenharia de Requisitos

Fonte: (PRESSMAN, 2002)

2.5 SAAS – SOFTWARE COMO SERVIÇOS

Software como serviço, do inglês Software as a Service, é um recurso de mercado que permite fornecer um software como um serviço. O software em questão é executado em um servidor, portando, não é preciso configurar o sistema na estação dos clientes, bastando somente acessá-lo via WEB. Como exemplo o Google Docs que fornece à seus usuários um pacote completo de programas de escritório.

No modelo de venda de software clássico, geralmente o produto é entregue após o pagamento ou é cobrando uma determinada quantia pelo desenvolvimento. Por sua vez, no modelo de SaaS, é desenvolvido uma solução sem cobrar qualquer quantia do cliente, ao final do desenvolvimento o serviço será disponibilizado na WEB para vários clientes e neste caso, o cliente interessado irá investir em uma locação que concede o direito de uso do serviço por tempo determinado. Quando analisado sobre a ótica a curto prazo, o modelo SaaS é mais lucrativo para o fornecedor

e mais caro para o cliente. Em geral, ao decorrer do tempo a tendência é que o SaaS passe a trazer mais vantagens simultaneamente para o fornecedor e o cliente.

Importante frisar que o SaaS é diferente de WEB Services, que se refere a um tipo de integração entre diferentes sistemas, também muito adotado em diversos tipos de soluções. Mas é oportuno salientar, que algumas soluções SaaS ofereçam seus serviços via WEB Services, deste modo, seus serviços poderão ser acessados por outros sistemas de forma automatizada, sem necessitar de intervenção do usuário.

Os conceitos de SaaS e HaaS (Hardware como um serviço) estão se disseminando continuamente em razão de um novo conceito de computação chamado *cloud computing*.

A solução SaaS é recomendada preferencialmente para empresas de pequeno porte e médias empresas, pois o SaaS viabiliza que estas empresas consigam adquirir soluções adequadas para sua necessidade sem que se faça necessário investimentos expressivos na ampliação de infra-estrutura ou aquisição de hardware.

2.6 AGÊNCIA NACIONAL DO PETRÓLEO (ANP)

Neste estudo, foi desenvolvido um projeto com base em ferramentas tecnológicas e embasamento sobre a Agência Nacional do Petróleo, Gás Natural e Bicom-bustíveis (ANP), que trata-se do órgão regulador das atividades que integram a indústria do petróleo e gás natural e a dos bicom-bustíveis no Brasil.

ANP possui o objetivo de estabelecer regras que viabilizem a criação de um mercado mais competitivo e que tragam vantagens para o país, especialmente, para os consumidores. Para o país, estes benefícios podem ser resumido em uma maior arrecadação fiscal e também a diminuição das importações de petróleo. Por sua vez, os consumidores obtêm a melhoria na qualidade dos combustíveis junto a uma.

Através do Portal ANP, o consumidor pode consultar os preços dos combustíveis, postos e cidades onde há fiscalização. Neste projeto, as informações do Portal serão processadas de forma automatizada via Data Scraping, uma tecnologia que será abortada posteriormente neste projeto.

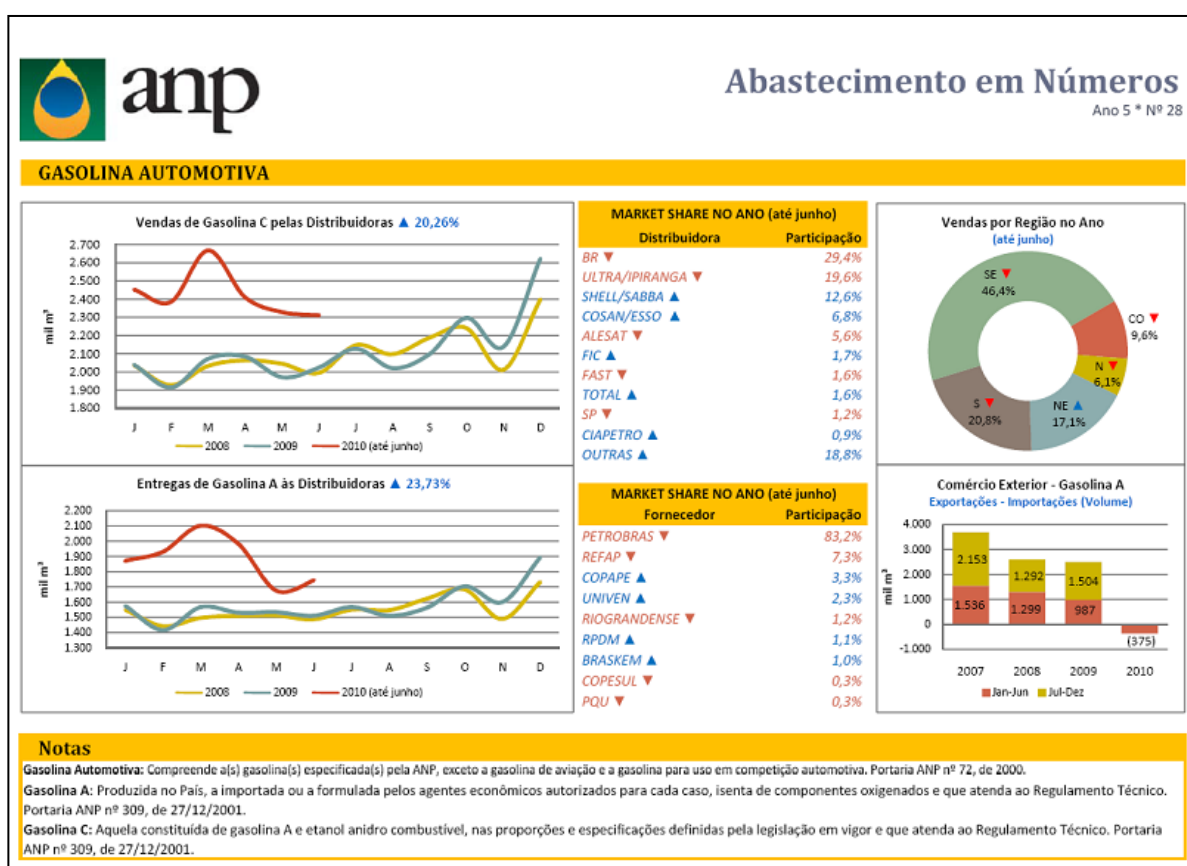


Figura 9 - Portal ANP

3 TECNOLOGIAS APLICADAS NO DESENVOLVIMENTO DO GASFINDER

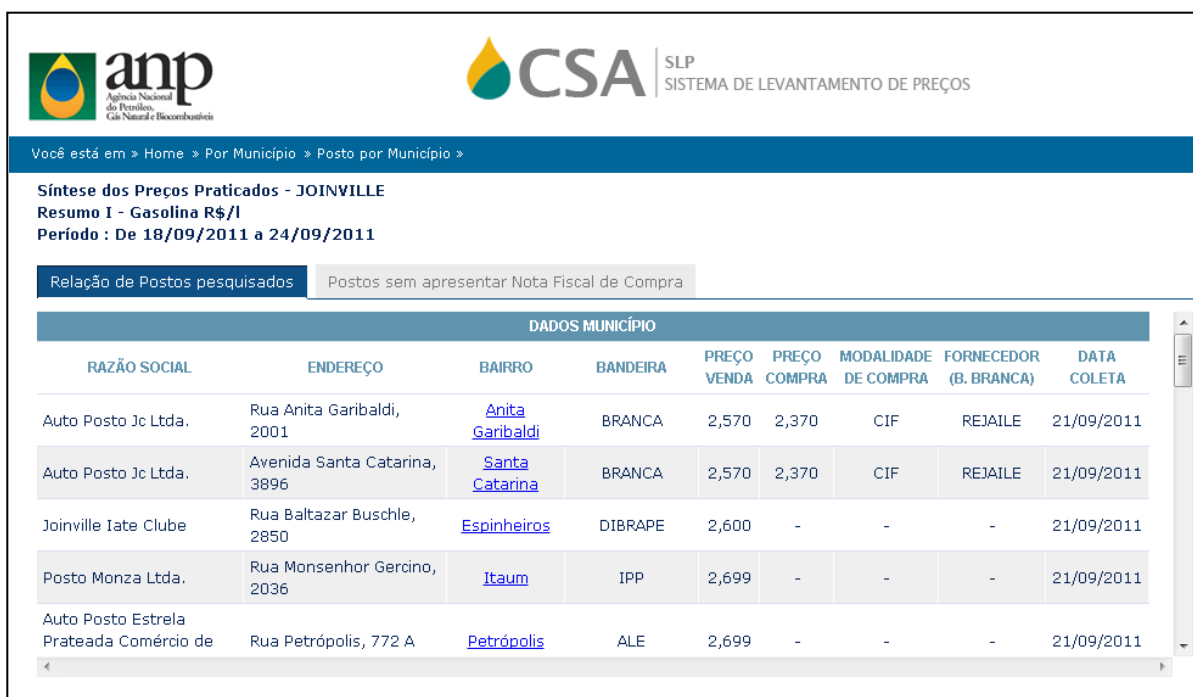
3.1 DATA SCRAPING

Data Scraping, ou simplesmente raspagem de dados, é uma prática onde o desenvolvedor extrai informações de saída dos programas que são legíveis para o usuário, e faz com que estas informações sejam convertidas de modo que possam ser utilizadas em outros programas. Por exemplo, os seres humanos não conseguem processar as informações disponibilizadas internamente em arquivos PDF. Mas um programa apropriado pode utilizar a técnica de raspagem de dados e acessar informações contidas no arquivo PDF, e por exemplo, realizar conversão para um outro tipo de arquivo, como XML ou TXT.

Na WEB esta técnica é fortemente adotada para indexação de *websites* por motores de busca, como o Google. O Data Scraping fornece um meio eficiente de acessar dados variados, não linear, indiferente da formatação ou codificação. Basta somente que o programa processador esteja preparado para interpretar as informações e tenha habilidade para converter os dados para as mais diversas finalidades ou necessidades do usuário.

Observamos que o Portal ANP não disponibiliza as informações através de XML, Web Service ou mesmo via arquivos em Excel que poderiam ser facilmente importados e processados para as finalidades deste projeto. O objetivo do Portal ANP é disponibilizar informações via WEB apenas para o consumidor final e empresas.

Por este motivo, neste projeto foi necessário adotar a técnica de Data Scraping somada com linguagem de programação Python para extrair informações sobre os preços dos combustíveis através do Portal da ANP. Em suma, o programa será capaz de parsear o HTML do Portal, que disponibiliza informações aos usuários comuns, processando e importando apenas os dados relevantes. A Figura 10 exibe uma das páginas do Portal ANP onde o programa irá acessar para realizar este procedimento.



Você está em » Home » Por Município » Posto por Município »

Síntese dos Preços Praticados - JOINVILLE
Resumo I - Gasolina R\$/l
Período : De 18/09/2011 a 24/09/2011

Relação de Postos pesquisados Postos sem apresentar Nota Fiscal de Compra

DADOS MUNICÍPIO								
RAZÃO SOCIAL	ENDEREÇO	BAIRRO	BANDEIRA	PREÇO VENDA	PREÇO COMPRA	MODALIDADE DE COMPRA	FORNECEDOR (B. BRANCA)	DATA COLETA
Auto Posto Jc Ltda.	Rua Anita Garibaldi, 2001	Anita Garibaldi	BRANCA	2,570	2,370	CIF	REJAILE	21/09/2011
Auto Posto Jc Ltda.	Avenida Santa Catarina, 3896	Santa Catarina	BRANCA	2,570	2,370	CIF	REJAILE	21/09/2011
Joinville Iate Clube	Rua Baltazar Buschle, 2850	Espinheiros	DIBRAPE	2,600	-	-	-	21/09/2011
Posto Monza Ltda.	Rua Monsenhor Gercino, 2036	Itaum	IPP	2,699	-	-	-	21/09/2011
Auto Posto Estrela Prateada Comércio de	Rua Petrópolis, 772 A	Petrópolis	ALE	2,699	-	-	-	21/09/2011

Figura 10 - Preço do combustível disponibilizado ao consumidor no Portal ANP

3.2 UML

Em 1990 nasceu um novo conceito de modelagem, a Análise Orientada a Objetos. Por volta dos anos 90, os paradigmas de padrões de projeto, componentes e frameworks avançaram progressivamente. Curiosamente, ainda neste período era trivial empregar simultaneamente técnicas de notações gráficas distintas para modelar um mesmo aspecto dos sistemas. Logo se viu necessário elaborar uma linguagem que viesse assumir o novo padrão para a modelagem de sistemas, que fosse conhecida e empregada largamente pela indústria e pelos acadêmicos. Surgiram portanto, algumas iniciativas na busca por novos padrões, o que mais tarde derivou na atual Linguagem de Modelagem Unificada. (BEZERRA, 2002).

A UML,

Figura 11, define as melhores práticas para representação dos dados reais do que será projetado, dos requisitos e suas integrações, usuários ou sistemas que interagem com o software. Os diagramas auxiliam na visualização dos requisitos da aplicação e nos transmitem uma projeção de como o sistema irá funcionar.

Trata-se de uma metodologia para gerenciamento de projetos, também chamado de framework para desenvolvimento, que segundo Bezerra (2002) é definida como uma linguagem de modelagem visual que possui um conjunto de notações e correspondente semântica, visando alcançar a representação visual de uma ou mais perspectivas do sistema. Larman (2000) caracteriza a UML como uma notação para modelagem de sistemas, fazendo uso dos conceitos de orientação a objeto.



Figura 11 - UML

A UML é composta por vários elementos de modelo que tem no seu objetivo representar as diferentes etapas de um sistema. Os elementos do UML são usados para gerar diagramas que representam um ponto de vista do sistema Omg (2007).

Segundo Furlan (1998), a UML é uma linguagem de modelagem, não uma metodologia. Assim, na construção de um software, a UML deve ser usada em conjunto com uma metodologia de Engenharia de Software Orientada a Objetos.

Para o desenvolvimento do sistema proposto, serão apresentados os diagramas de caso de uso, o diagrama de seqüência, diagrama de classe e diagrama de entidade relacionamento.

3.2.1 Casos de Uso

Segundo Rumbauch, Booch e Jacobson (2002), é possível definir um Caso de Uso como um “documento narrativo que descreve a seqüência de eventos de um ator que usa um sistema para completar um processo”. Por sua vez, o Diagrama de Caso de Uso tem como objetivo descrever relacionamentos e dependências entre um grupo de Caso de Uso e os atores participantes do processo.

Os diagramas são desenvolvidos para facilitar a comunicação entre os futuros usuários e clientes do sistema, mostrando o que o sistema deve fazer sem especificar como isso irá ocorrer (LARMAN, 2004).

O diagrama de Caso de Uso se relaciona diretamente com a fase de elicitação de requisitos. Portanto, esse diagrama representa o mecanismo para examinar o sistema, expressando as interações entre o sistema e seu ambiente, suas entradas e saídas.

Conforme Kulak (2001), os casos de uso empregam um vocabulário mais próximo do entendimento do usuário, ou seja, da linguagem natural sem necessidade de referenciar implementações. A representação dos casos de uso pode ser gráfica ou não, mas sempre apresenta o aspecto textual. Seu gráfico é composto por um círculo oval contendo essencialmente um título dentro. Independentemente da adoção ou não da representação gráfica há dois elementos essenciais nos diagramas, além dos próprios casos de uso: atores e associações.

Apresentando o sistema em formas de diagramas é possível obter uma visão panorâmica do funcionamento da aplicação. Portanto, a Linguagem de Modelagem Unificada (Unified Modelling Language - UML) é um padrão de notação com intuito de visualizar, especificar e documentar modelos de sistemas Orientados a Objeto.

A Figura 12 está evidenciando o Diagrama de Caso de Uso com as funcionalidades para o sistema proposto, destacando as seguintes interações: Cadastros, Gerenciamento e Busca.

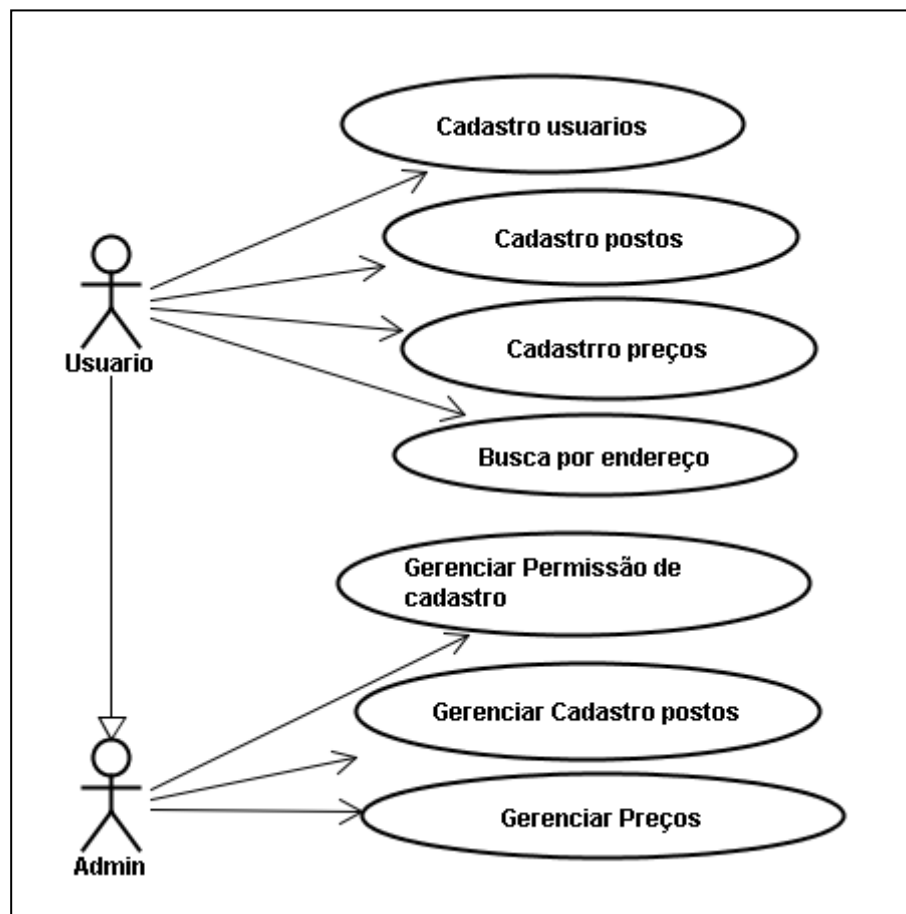


Figura 12 - Caso de Uso Geral

3.2.2 Diagrama de Seqüência

Furlan (1998), define o diagrama de seqüência como responsável por elucidar o comportamento dos objetos em um sistema, demonstrando suas interações, colaborações, operações e estado.

Este tipo de diagrama demonstra a interação de seqüência de tempo dos objetos que participam na interação. O diagrama de seqüência também mostra a troca

de mensagens entre diversos objetos, em uma situação específica e delimitada no tempo. Existe uma ênfase específica na seqüência e nos momentos quando as mensagens são enviadas.

Graficamente, um diagrama de seqüência possui os elementos representados por meio de linhas verticais tracejadas (indicadas como linha de existência), com o nome do objeto no topo. O eixo do tempo é também vertical, estendendo para baixo, de forma que as mensagens são direcionadas em entre os objetos.

BOOCH, 2000, defende que todos os objetos que começam as interações precisam ser alinhados mais à esquerda e por sua vez, as mensagens são posicionadas no eixo Y, de cima para baixo, sempre em ordem crescente. A Figura 13 exibe o diagrama de seqüência para o sistema proposto.

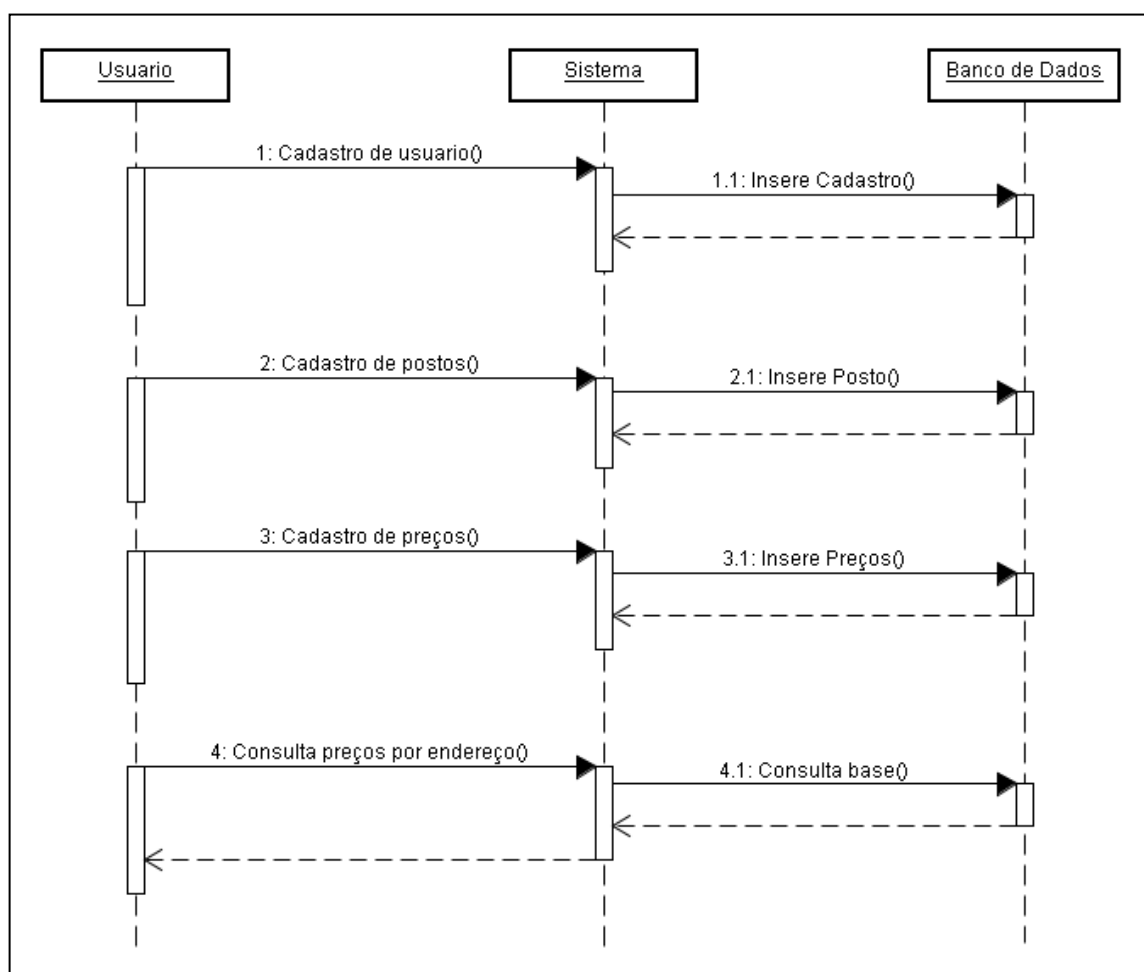


Figura 13 - Diagrama de Seqüência

3.2.3 Diagrama de Entidade Relacionamento

O Diagrama Entidade Relacionamento ou DER é utilizado para mostrar a estrutura de um banco de dados. A Figura 14 o diagrama construído para este projeto.

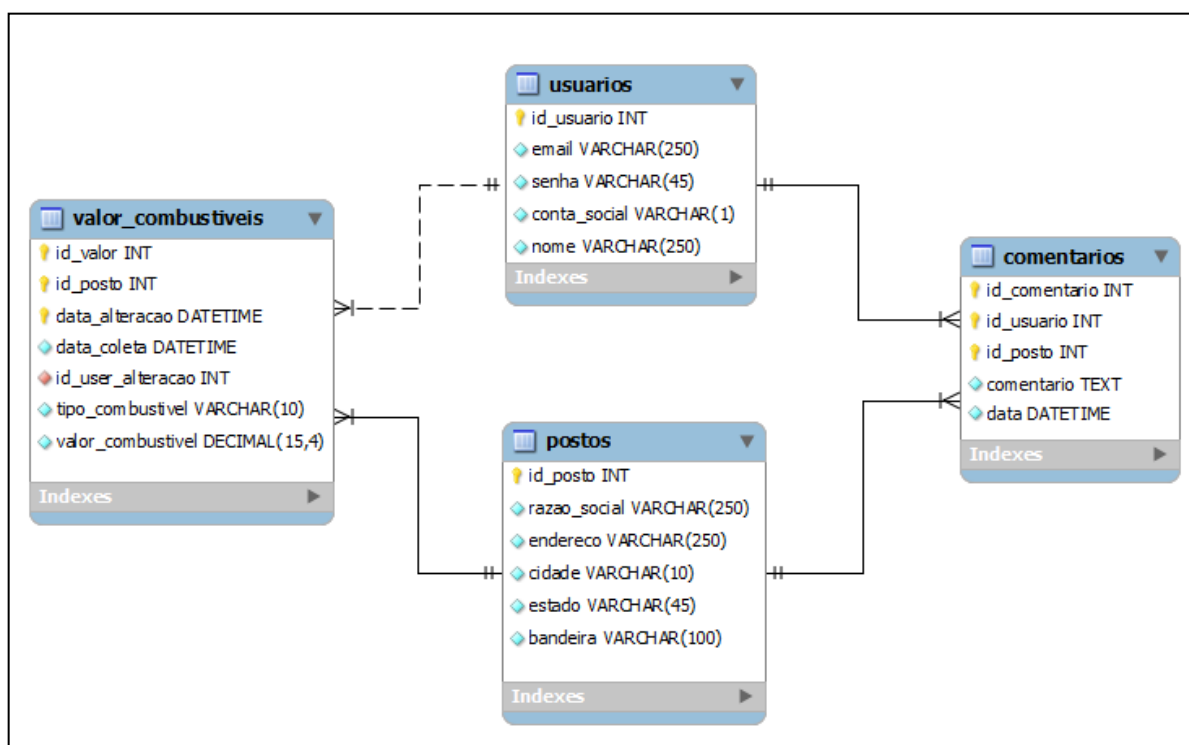


Figura 14 - Diagrama de Entidade Relacionamento

3.2.4 Diagrama de Classe

O diagrama de classe demonstrado na Figura 15 foi utilizado para modelar todas as classes utilizadas neste projeto.

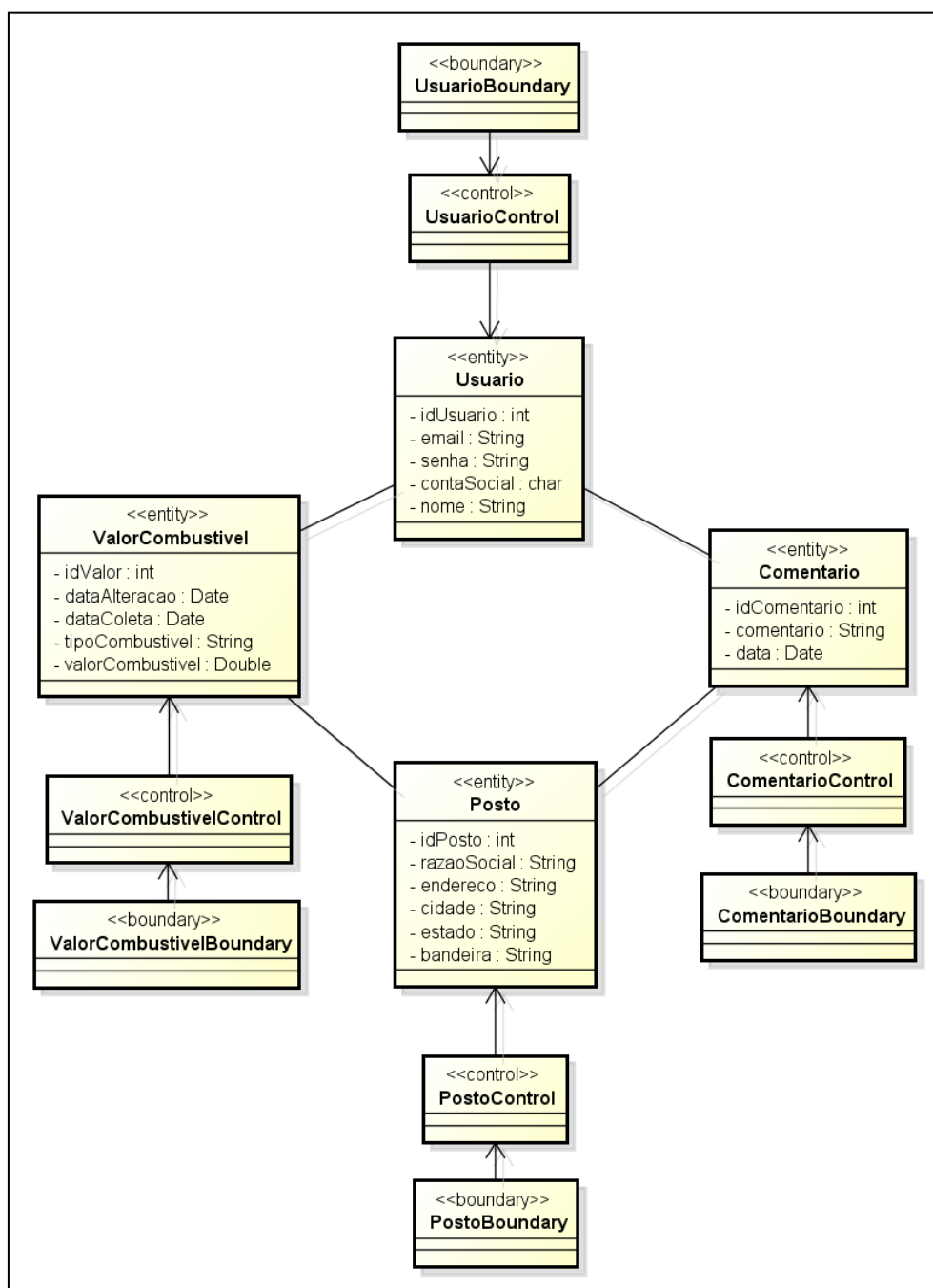


Figura 15 - Diagrama de Classe

3.3 JAVA

Java é uma linguagem de programação desenvolvida pela Sun Microsystems, conforme. Resultado de uma pesquisa interna da empresa, em 1991 chamada pelo codinome Green, Java (destacado na Figura 16) resultou do desenvolvimento de outra linguagem de programação baseada em C e C++. Seu criador, James Gosling, chamou a nova linguagem de Oak (carvalho) fazendo uma homenagem a uma árvore que havia próximo a janela do seu escritório na Sun. Mais tarde foi descoberto que já existia uma linguagem de programação chamada Oak, e quando a equipe da Sun visitava uma cafeteria, o nome Java foi sugerido e curiosamente adotado por ser o nome da cidade de um tipo de café importado, típico nas cafeterias locais da cidade.



Figura 16 - Java

O modelo computacional para o desenvolvimento do Java foi construído a partir de paradigma da orientação a objeto. A metodologia de orientação a objetos define uma notação e um processo para o desenvolvimento de sistemas. Destaca a importância de extrair uma hierarquia de forma clara através de classes e objetos do sistema que são modelados. (COURTOIS in LAPOLLI, 1994).

Java possui as características necessárias para implementação de um sistema completo orientado a objetos. Diferentemente das linguagens convencionais que são compiladas para código nativo, a linguagem Java é compilada para um código intermediário – chamado de *bytecode* – que é executado por uma máquina virtual, a Java Virtual Machine (JVM). Esta facilidade do código ser interpretado em tempo de execução faz com que a linguagem seja independente de plataforma.

Esta característica destaca um dos maiores diferenciais da linguagem, agregando portabilidade e possibilitando que os programas desenvolvidos em Java possam ser executados em qualquer sistema que possua suporte a uma JVM, sejam micro-computadores, dispositivos móveis ou sistemas embarcados.

Tendo como necessidade o desenvolvimento de um Portal WEB, foi decidido adotar a linguagem Java em virtude do conhecimento adquirido em aulas teóricas e práticas durante o curso, e também pela vasta gama de materiais e livros disponíveis sobre esta linguagem de programação.

3.4 MYSQL

O MySQL foi desenvolvido na Suécia, por um finlandês e por dois suecos pela companhia TCX em 1996. Atualmente a MySQL AB desenvolve e é mantenedora do programa. MySQL AB é a companhia dos fundadores e principais desenvolvedores do MySQL. Eles criaram o MySQL porque buscavam um banco de dados que pudesse trabalhar com grandes quantidades de informações em máquinas de baixo custo.

É atualmente um dos bancos de dados mais rápidos no mercado, apresenta praticamente todas as funcionalidades de um grande banco de dados comercial. O MySQL é adotado geralmente em sistemas que compartilham da filosofia UNIX, porém outros sistemas operacionais também apresentam suporte, como Windows, por exemplo.

O MySQL é um projeto de código-fonte aberto e nível corporativo com um sistema completo para gerenciamento de banco de dados relacional que atende as diversas dimensões e necessidades sejam uma pequena aplicação ou aplicações de grande porte que necessitam de conexões simultâneas.

O MySQL utiliza a linguagem SQL (Structured Query Language), que permite gravar, alterar e recuperar informações com segurança e rapidez. A Tabela 1 demonstra exemplos de consultas SQL em uma tabela modelo com seus respectivos resultados.

Tabela 1 - Consulta SQL

Tabela 'T'	Consulta SQL	Resultado												
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	SELECT * FROM T	<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b
C1	C2													
1	a													
2	b													
C1	C2													
1	a													
2	b													
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	SELECT C1 FROM T	<table><tr><th>C1</th></tr><tr><td>1</td></tr><tr><td>2</td></tr></table>	C1	1	2			
C1	C2													
1	a													
2	b													
C1														
1														
2														
<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table>	C1	C2	1	a	2	b	SELECT * FROM T WHERE C1=1	<table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr></table>	C1	C2	1	a		
C1	C2													
1	a													
2	b													
C1	C2													
1	a													

MySQL representado na Figura 17, agrega todas as características relevantes e consideradas importantes para sua escolha como SGDB para este projeto. É uma solução robusta para quase todo tipo de aplicação, combina a estabilidade com e baixo custo de propriedade e portanto, oferece o melhor cenário, executa em muitas plataformas e finalmente, é muito estável e compatível com a linguagem de programação Java.



Figura 17 - MySQL

3.5 CSS

CSS refere-se a uma tecnologia utilizada para formatação de páginas WEB. O CSS proporciona maior liberdade na criação de possibilidades de formatação, podendo atingir muitos resultados que não é possível utilizando somente HTML. É possível por exemplo, incluir imagens, definir margem entre os objetos na tela, alterar o posicionamento dos elementos, definir tipos de fonte, letras e cores. CSS é a abreviação para os termos em inglês Cascading Style Sheet, ou simplesmente para o português como folhas de estilo em cascata. (SILVA, 2007).

CSS é adotado praticamente em todos os projetos WEB modernos, sua aplicação é imprescindível para garantir maior nível de acessibilidade entre outros padrões de interação atuais. A

Figura 18 demonstra um exemplo de uso de CSS, com duas cláusulas alterando os padrões da página WEB em que for aplicada. Este projeto utiliza CSS para formatar os elementos das páginas.

```
<style>
  body {
    color: #000;
    background: #fff;
    font-weight: bold;
  }
  p {
    font-size: 12px;
  }
</style>
```

Figura 18 - Exemplo de Código CSS

3.6 HTML

HTML é uma linguagem utilizada para definir a estrutura de página WEB. São usados um conjunto *tags* com objetivo de estruturar a forma como será apresentado o texto e outros elementos da página. Cada *tag* (ou elemento) possui atributos correspondentes que diversificam as aplicações. Há dois elementos principais, chamado *head* que deve conter informações de controle da página, referencia para folhas de estilos, títulos e código-fonte de aplicações cliente, como JavaScript. E o outro chamado *body* que comporta o conteúdo da página propriamente dito, elementos de texto, imagens, vídeos e demais objetos de interação visíveis pelo usuário. (SILVA, 2007)

O HTML é uma linguagem muito fácil de compreender, o que possibilita que qualquer pessoa sem conhecimentos sólidos de programação, possa desenvolver um *website* com relativa simplicidade. A Figura 19 demonstra sua estrutura básica, composta por um elemento *head* e *body*.

```
<html>  
  <head>  
    Título da página e elementos de controle  
  </head>  
  <body>  
    Conteúdo, texto, imagens, etc.  
  </body>  
</html>
```

Figura 19 - Exemplo de Código HTML

3.7 JAVASCRIPT

É uma linguagem de programação onde o programa (chamado *script*) é executado no lado cliente, ou seja, no *browser* do usuário. JavaScript tem uma sintaxe muito parecida com Java e C++ e devido a compatibilidade com vários *browser* modernos, o JavaScript é amplamente utilizado pela comunidade de desenvolvimento WEB (FLANAGAN, 2007).

Com JavaScript é possível aumentar a interação do *website*, através de efeitos na página e também proporciona outras utilidades, como validação de formulários e recentemente, o uso de funções AJAX que permitem interações entre o cliente-servidor sem que seja necessário recarregar ou processar toda a página. A Figura 20 demonstra a simples exemplo da sintaxe do JavaScript.

```
<script>  
  function Ola() {  
    alert("Olá Mundo!");  
  }  
</script>
```

Figura 20 - Exemplo de Código JavaScript

3.8 FRAMEWORKS

3.8.1 Hibernate

Hibernate representado pela Figura 21 é um framework Java, open source que pertence a JBoss Inc. É uma distribuição com código aberto e gratuito, por este motivo a comunidade de desenvolvedores pode ajustar ou adicionar novas funcionalidades ao projeto, tornando o framework cada vez mais robusto. O Hibernate possibilita que os desenvolvedores tenham persistência em suas aplicações, sem que seja necessário utilizar instruções SQL em meio ao código-fonte Java. (LUCKOW, MELO. 2010)

O Hibernate fica situado entre a camada aplicativo e o banco de dados e seu principal objetivo é proporcionar aos desenvolvedores uma forma simples e transparente de mapear e desenvolver a aplicação, sem se preocupar com o desenvolvimento de classes de persistência. Não somente, o Hibernate oferece muitas outras vantagens, principalmente por poupar tempo de desenvolvimento e mapeamento, e também por ser compatível com a maioria dos bancos de dados de mercado.

O Hibernate foi adotado neste projeto para realizar parte de interação com o banco de dados, reduzindo a necessidade de construir código específico para a persistência de dados.



Figura 21 - Hibernate

3.8.2 JSF

JSF ou JavaServer Faces é um framework que possibilita o desenvolvimento de interfaces de usuário em aplicações WEB. Permite adicionar componentes em uma página, como um formulário e interligar diretamente com objetos Java. Oferece um bom conjunto de recursos para as necessidades de desenvolvimento, e possibilita ao programador focar somente na lógica de negócio, de tal modo que as tarefas dispendiosas são geridas pelo framework. (LUCKOW, MELO. 2010).

O JSF disponibiliza componentes básicos, como botões e controles assim como componentes mais elaborados, como tabelas. Entretanto, o maior diferencial é a integração entre o padrão Java EE que permite que a aplicação seja facilmente incorporando dentro de um container WEB.

O JSF apresenta os seguintes conjuntos aos desenvolvedores:

- Componentes de interface de usuário.
- Modelo de programação orientado a eventos.
- Modelo para criar componentes adicionais.

Mantido por uma comunidade de desenvolvedores chamada JCP (Java Community Process), o JSF é atualmente o framework mais adotado em projetos de aplicações WEB baseado em tecnologias de desenvolvimento Java.

3.8.3 Spring

O Spring Framework é uma ferramenta Java, de código-fonte aberto que visa facilitar o desenvolvimento JavaEE. Ele é baseado nos padrões de Inversão de Controle (IoC) e Injeção de Dependências. A funcionalidade básica do Spring é a instanciamento de classes, realizando a injeção de dependências com base em definições em um arquivo de configuração XML criado pelo desenvolvedor, resultando em um baixo acoplamento entre classes.

Porem, o Spring fornece também outros projetos que utilizam essa funcionalidade básica, como WebServices com o Spring WebService, o Spring WebFlow, que fornece alguns recursos para o desenvolvimento de aplicativos WEB e integração com JavaServer Faces e Spring Security.

3.9 JSON

JSON ou ainda JavaScript Object Notation, é um formato que permite a troca de informações entre sistemas consumindo poucos recursos e gerando baixo tráfego de dados. JSON é um subconjunto do JavaScript, mas sua implementação não necessita JavaScript obrigatoriamente. A facilidade de desenvolver soluções utilizando JSON tem resultado em seu uso difundido entre a comunidade de desenvolvedores, principalmente como uma opção para o XML utilizado juntamente com AJAX.

Um dos diferenciais do JSON comparado ao XML, é que a implementação de um analisador de dados é mais fácil de desenvolver. Não somente, é possível incorporar funções nativas em JavaScript, pois a maioria dos *browsers* atuais possuem compatibilidade.

JSON é fortemente utilizado em ambiente onde existe grande taxa de transferência de dados entre o cliente servidor, dentre outros é uma tecnologia adotada nas soluções da Google e Yahoo para atender centenas de milhares de usuários.

A

Figura 22 demonstra um exemplo de container de dados desenvolvido utilizando JSON.

```
{
  "cidade": "Joinvile",
  "posto":
  {
    "nome": "Posto Anita",
    "endereco": "Anita Garibaldi, 2001",
  },
  "combustivel":
  [
    {
      "tipo": "alcool",
      "preco": "1.879"
    },
    {
      "tipo": "gasolina",
      "preco": "2.599"
    }
  ]
}
```

Figura 22 - Exemplo de Código JSON

3.10 PYTHON

Desenvolvida em 1991 por Guido van Rossum, Python é uma linguagem de programação com suporte a orientação a objetos. É distribuída livremente e desenvolvida por uma comunidade de desenvolvedores, gerida pela organização Python Software Foundation.

Devido a facilidade de implementação e disponibilidade de componentes, Python foi adotada neste projeto para construção do módulo responsável por realizar o Data Scraping no Portal ANP. Um exemplo de código pode ser visto na Figura 23.

```
lista = ['gasolina', 'alcool', 'gnv']  
lista.sort()  
  
for item in lista:  
    print item.capitalize()
```

Figura 23 - Exemplo de Código Python

3.11 ECLIPSE

O Eclipse é uma plataforma para integração de ferramentas construídas por uma comunidade aberta de provedores. Escrita completamente em Java, a plataforma pode ser utilizada em estações de desenvolvimento que incluam os sistemas operacionais Linux, OSX e Windows. (MACENAS, 2006)

Em novembro de 2001, um grupo de empresas formou um conselho de administradores do eclipse.org, o website da Fundação Eclipse. Nesse grupo de empresa estavam IBM, QNX Software System, Rational Software, Red Hat, SuSE, Together-Soft e Webgain.

O Eclipse com sua logomarca vista na Figura 24, fornece aos desenvolvedores uma IDE completa para a construção de aplicações Java. Em função de sua vasta utilização pela comunidade de desenvolvedores, fácil instalação e uso, também foi adotada como ferramenta para o desenvolvimento deste projeto.



Figura 24 - Eclipse

3.11.1 Jude Community

Desenvolvida pela empresa ChangeVision Inc, Jude Community é uma ferramenta CASE (Case Aided Software Engineering) destinada a criação de diagramas e criação de código-fonte a partir dos diagramas, gerenciamento de versões e mudanças de requisitos, entre outros. (VISION, 2009)

Jude representada na Figura 25 é uma ferramenta distribuída livremente e foi adotada neste projeto como ferramenta para construção de diagramas padrão UML.

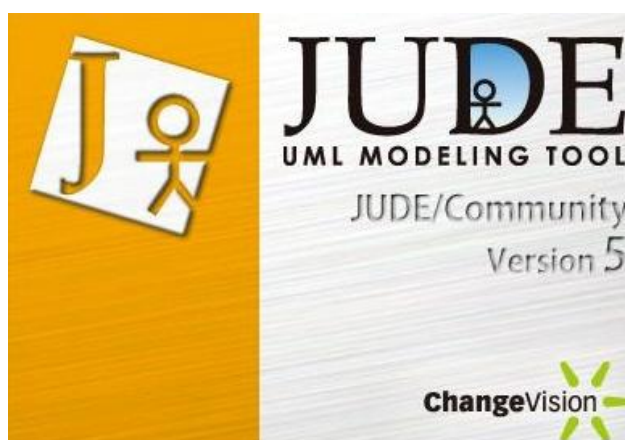


Figura 25 - Jude

3.11.2 Tomcat

Desenvolvido pela Apache Software Foundation, o Tomcat é um servidor HTTP com suporte para aplicações desenvolvidas em Java, utilizando Servlet e JSP. O Tomcat é um projeto com distribuição livre e tem sido referenciado por grandes empresas como melhor opção para distribuição de aplicações WEB Java.

O servidor agrega ferramentas para gerenciamento e configuração avançados, e pode ser integrado a outros servidores WEB do mercado como o Microsoft IIS. Não somente, o servidor também permite rodar aplicações desenvolvidas em outras linguagens de programação, como por exemplo o PHP.



Figura 26 - Apache

4 DESENVOLVIMENTO DO GASFINDER

Atualmente a popularização da internet permite que as pessoas efetuem pesquisas e comparações de diversas informações sobre produtos e serviços. O aumento constante dos combustíveis fez que os consumidores criassem o hábito da pesquisa e comparação dos preços dos combustíveis, para terem o melhor custo-benefício, que é composto pela menor distância do posto ao seu local de origem somando ao menor valor por litro do combustível.

Com isso criamos um projeto para sanar uma *gap* na disposição das informações e dos preços dos combustíveis de uma forma mais eficiente e clara para os motoristas. Abordaremos os requisitos necessários para a criação de uma ferramenta de busca e comparação dos preços dos combustíveis do país.

Pretende-se suprir uma necessidade que os motoristas tem de efetuar a buscar a localização dos postos e comparação dos preços dos combustíveis. Em suma, a solução consiste em importar as informações do Portal ANP e possibilitar que o consumidor faça busca dos postos sempre permitindo que ele efetue uma comparação de preços dos combustíveis de uma forma mais amigável e objetiva.

4.1 ESCOPO DO SISTEMA

O escopo desse projeto compreende o desenvolvimento de um aplicativo WEB, com as seguintes características:

- Apresentar a média dos preços dos combustíveis da cidade de acesso, no acesso inicial do aplicativo, a identificação da cidade será feita através de *geolocation* por IP.
- Busca e comparação dos preços dos combustíveis, através de busca livre ou avançada (região, estado, cidade, bairro ou rua).
- Opção de marcar no mapa a localização do posto de combustível, proveniente da busca.

4.1.1 Descrição Geral

O sistema tem por objetivo disponibilizar um serviço de busca, marcação de informações em um mapa e comparação dos preços dos combustíveis dos postos de uma região.

4.1.2 Funções do Produto

As funções desse aplicativo são fornecer uma comparação de preços dos combustíveis entre os postos de uma região. Fornecendo assim o menor preço dos combustíveis. A solução irá carregar automaticamente as informações ao usuário baseado na cidade de origem do acesso. Não somente, será através de um campo de busca que o consumidor poderá pesquisar os preços dos combustíveis de qualquer cidade do país. A utilização de mapas visa identificar o local onde estão localizados os postos.

4.1.3 Características dos Usuários

Existirão 3 tipos de usuário:

- Usuários do tipo 1, serão os usuários que não terão contas no sistema e, utilizarão o aplicativo somente como para pesquisa. (ex.: usuario final do sistema, ou consumidor).
- Usuários do tipo 2, serão os usuários que terão contas no sistema e, que poderão controlar informações/anúncios referentes as seus postos. (ex.: proprietario de um posto de combustível).

- Usuários do tipo 3, serão os usuários que terão contas, que poderão controlar propagandas referentes as distribuidoras e produtos para carros. (ex.: empresas parceiras).

4.1.4 Restrições Gerais

O sistema poderá ser usado por qualquer usuário que tenha acesso a internet e *browser* compatível com JavaScript. IE 7 ou superior. Firefox 3.5 ou superior.

4.1.5 Assertivas e Dependências

Este projeto apresenta conforme escopo, um software inovador para suprir uma necessidade sistemática.

4.1.6 Requisitos de Usuário

- RU01 - O tem que ser colaborativo.
- RU02 - Tem que ter cadastro de usuário.
- RU03 - Ter busca (por região, cidade, estado etc).
- RU04 - Denúncia para contas moderadas, para preços não verdadeiros.
- RU05 - Ter uma área de detalhes do posto (ex.: serviços prestados e diferenciais).
- RU06 - Áreas para publicidades.
- RU07 - Criação de administração de conta por posto.

4.1.7 Requisitos de Interface

- RI01 - Ser um aplicativo WEB.
- RI02 - Ter uma interface amigável.
- RI03 - Deve ter um campo de busca na página inicial.
- RI04 - Os campos obrigatórios para preenchimento devem ser identificados com um asterisco no lado direito.
- RI05 - Todos os campos devem aparecer um *tooltip*, com informações sobre o mesmo, ao ser dado foco.

4.1.8 Requisitos de Domínio

- RD01 - Estatísticas de acesso, com Postos mais pesquisados, etc.
- RD02 - O cadastro dos postos deve ser baseado nas informações da ANP.
- RD03 - Só o administrador do sistema e os usuários com contas poderão alterar atualizar os preços dos combustíveis.
- RD04 - Apresentar os resultados das buscas em forma de lista ou com marcação no mapa.
- RD05 - Efetuar a comparação de preços dos combustíveis dos postos de uma determinada região.
- RD06 - Na página principal deve ter média dos preços dos combustíveis, da cidade que usuário está acessando.
- RD07 - *GeoTags* para pesquisas iniciais.

4.1.9 Matriz de Rastreabilidade de Requisitos

Matriz de Requisitos do Sistema									
ID	Nome do Requisito	Descrição	Prioridade	Status	Classificação	Complexidade	Data	Solicitado Por	Aprovado Por
001	RU01	O tem que ser colaborativo	Média	Aprovado	Funcional	Médio	6/11/2010	William	William
002	RU02	Tem que ter cadastro de usuário	Baixa	Aprovado	Funcional	Muito Simples	6/11/2010	William	William
003	RU03	Ter busca avançada (por região, cidade, estado etc)	Alta	Aprovado	Regra de Negócio	Complexo	6/11/2010	William	William
004	RU04	Denúncia para contas moderadas, para preços não verdadeiros	Baixa	Aprovado	Funcional	Simple	6/11/2010	William	William
005	RU05	Ter uma área de detalhes do posto(ex: serviços prestados e diferenciais)	Baixa	Aprovado	Funcional	Simple	6/11/2010	William	William
006	RU06	Áreas para publicidades	Baixa	Aprovado	Funcional	Médio	6/11/2010	William	William
007	RU07	Criação de administração de conta por posto	Média	Aprovado	Funcional	Médio	6/11/2010	William	William
008	RI01	Ser um aplicativo web	Alta	Aprovado	Não-Funcional	Simple	6/11/2010	William	William
009	RI02	Ter uma interface amigável	Média	Aprovado	Não-Funcional	Simple	6/11/2010	William	William
010	RI03	Deve ter um campo de busca na página inicial	Alta	Aprovado	Funcional	Simple	6/11/2010	William	William
011	RI04	Os campos obrigatórios para preenchimento devem ser identificados com um asterisco no lado direito	Baixa	Aprovado	Não-Funcional	Muito Simple	6/11/2010	William	William
012	RI05	Todos os campos devem aparecer um tooltip, com informações sobre o mesmo, ao ser dado foco	Baixa	Aprovado	Não-Funcional	Muito Simple	6/11/2010	William	William
013	RD01	Estatísticas de acesso, mais buscados etc	Média	Aprovado	Regra de Negócio	Complexo	6/11/2010	William	William
014	RD02	O cadastro dos postos de ser conforme o cadastro da ANP	Alta	Aprovado	Regra de Negócio	Simple	6/11/2010	William	William
015	RD03	Só o administrador do sistema e os usuários com contas poderão alterar/atualizar os preços dos combustíveis	Alta	Aprovado	Regra de Negócio	Simple	6/11/2010	William	William
016	RD04	Apresentar os resultados das buscas em forma de lista ou com marcação no mapa	Alta	Aprovado	Regra de Negócio	Médio	6/11/2010	William	William
017	RD05	Efetuar a comparação de preços dos combustíveis dos postos de uma determinada região	Alta	Aprovado	Regra de Negócio	Complexo	6/11/2010	William	William
018	RD06	Na página principal deve ter média dos preços dos combustíveis, da cidade que usuário está acessando	Alta	Aprovado	Regra de Negócio	Médio	6/11/2010	William	William
019	RD07	Geotags para pesquisas iniciais	Alta	Aprovado	Funcional	Médio	6/11/2010	William	William
ID: TCC01			Projeto: Protótipo01				Data:11/06/10 Versão:1.0		

Figura 27 - Matriz de Rastreabilidade de Requisitos

4.1.10 Matriz de Rastreabilidade de Dependência

Requisitos/Requisitos	RU01	RU02	RU03	RU04	RU05	RU06	RU07	RI03	RD01	RD02	RD03	RD04	RD05	RD06	RD07
RU01															
RU02											X				
RU03															
RU04		X													
RU05							X								
RU06		X													
RU07		X													
RI03			X						X						
RD01															
RD02												X	X	X	
RD03															
RD04															
RD05															
RD06															
RD07															

Figura 28 - Matriz de Rastreabilidade de Dependência

4.1.11 Matriz de Rastreabilidade de Módulos do Sistema

Modulo/Requisito	RU01	RU02	RU03	RU04	RU05	RU06	RU07	RI03	RD01	RD02	RD03	RD04	RD05	RD06	RD07
Administração				X						X					
Controle de Usuário		X			X	X	X				X				
Motor de Busca	X		X					X	X			X	X	X	X

Figura 29 - Matriz de Rastreabilidade de Módulos do Sistema

4.1.12 Matriz de Rastreabilidade de Usuário

Conforme apresentado nas características do usuário, os atores envolvidos no sistema são classificados em três níveis: usuário básico; intermediário e administrador. Com isso mostramos em forma de matriz o que cada usuário pode fazer no sistema.

Requisitos/Atores	Usuário	Admin
RU01	X	X
RU02	X	X
RU03	X	X
RU04	X	X
RU05		X
RU06		X
RU07		X
RI03	X	X
RD01	X	X
RD02		X
RD03		X
RD04	X	X
RD05	X	X
RD06	X	X
RD07	X	X

Figura 30 - Matriz de Rastreabilidade de Usuário

5 CONCLUSÃO

Neste projeto foram apresentadas diretrizes da arquitetura de software de um *website*, GASFINDER, onde seu modelo, será representativo na internet e conterá informações valiosas para quem utilizar deste serviço, pois com sua plataforma amigável, garantirá simplicidade na usabilidade de seus dados.

E como hoje, com o número grande e sempre aumentando de usuários conectados a internet, muitos *websites* tem como objetivo servir vários usuários ao mesmo tempo, estando sempre ativo e com um bom desempenho.

Por isto, mostramos ferramentas de gerenciamento e internet visando modelar a rotina e desenvolvimento de um *website* direcionado ao público em geral. Através de ferramentas como Eclipse, utilizando Java, esta proposta pode ser testada, evidenciando um *website* de serviços atraentes e fácil e rápido acesso.

Uma característica fundamental para adaptar o *website* a novas mudanças e utilização, será seu processo de migração para um ambiente WEB que mantenha o serviço cada vez mais atrativo.

Por isto, por se tratar de linguagens e ferramentas que estão em visível avanço tecnológico no mercado, será um *website* visto pelo um grande número de pessoas interessadas em procurar este tipo de serviço, principalmente quem não conhece os estabelecimentos próximos a sua rota. E visando que este *website* seja acessado por todas as pessoas, independente do local que estejam, ou seja, quando não houver acesso WEB próximo, deixamos este novo recurso como uma proposta para um trabalho futuro, dando continuidade a esse serviço WEB.

REFERÊNCIAS

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML – Guia do Usuário**. 2ª ed. Rio de Janeiro: Elsevier, 2002.

CAMPIONE, M; WALRATH, K. **The Java Tutorial: Object-Oriented Programming for the Internet**. EUA: SunSoft Press, 1996.

FURLAN, J. Davi. **Modelagem de Objetos através da UML - The Unified Modeling Language**. São Paulo: Makron Books, 1998.

GRAEF, G.; GAEDKE, G. **Development and Evolution of Web-Applications using the WebComposition Process Model**. Amsterdam: The Netherlands, 2000.

LARMAN, C. **Agile and Iterative Development: A Manager's Guide**. EUA: Pearson, 2004.

LARMAN, Craif. **Utilizando UML e Padrões**. São Paulo: Bookman, 2004.

LAUDON, Kenneth C.; LAUDON, Jane Price. **Gerenciamento de Sistemas de Informação**. Rio de Janeiro: Prentice Hall, 2001.

LUCKOW, D.; MELO, A. **Programando Java para WEB**. São Paulo: Novatec, 2010.

MACENAS, Ivan. **Eclipse 3.1 Programando com Visual Editor**. 2ª ed. Rio de Janeiro: Alta Books, 2006.

MORAES, Lucas. **A vida com Scrum**. <<http://alemdati.wordpress.com/2010/03/26/a-vida-com-scrum/>> Acesso em 26 Ago. 2011.

PRESSMAN, R.S. **Engenharia de Software**. 5ª ed. Rio de Janeiro: McGraw-Hill, 2002.

PRESSMAN, R.S. **Software Engineering: a practitioner's approach**. 3ª ed. New York: McGraw-Hill, 1992.

HADOOP. **Apache Hadoop Core**. Disponível em <<http://hadoop.apache.org/core/>>. Acesso em 17 Jun. 07 2011.

GLOGER. **The Zen of Scrum**. <<http://www.glogerconsulting.de>> Acesso em 27 Ago. 2011.

SCHWABER, K. **Agile Project Management With Scrum**. EUA: Microsoft, 2004.

SCHWABER, K., SCRUM. **Development Process**. EUA: Burlington, 1996.

SILVA, M. S. **Construindo sites com CSS e (X) HTML**. 1ª ed. São Paulo: Novatec, 2007.

ANEXO

ANEXO 1 – WIREFRAME - HOME

ANEXO 2 – WIREFRAME – CONTATO

ANEXO 3 – WIREFRAME – QUEM SOMOS

ANEXO 4 – WIREFRAME – LOGIN

ANEXO 5 – WIREFRAME – PESQUISA NA LISTA DE PRODUTOS

ANEXO 6 – WIREFRAME – SEM RESULTADOS PARA PESQUISA

ANEXO 7 – WIREFRAME – PESQUISA DIRETAMENTE NO MAPA

ANEXO 8 – WIREFRAME – MANUTENÇÃO DE PREÇOS

ANEXO 9 – WIREFRAME – INFORMAÇÕES DOS POSTOS

ANEXO 10 – WIREFRAME – ERRO GENÉRICO

ANEXO 1 – WIREFRAME - HOME

A página principal do Portal irá resumir os melhores preços dos combustíveis, sugerindo automaticamente os Postos baseado na localidade de origem do acesso. A página Home destaca o preço da Gasolina, Alcool, Diesel e GNV, conforme demonstra a

Figura 31.

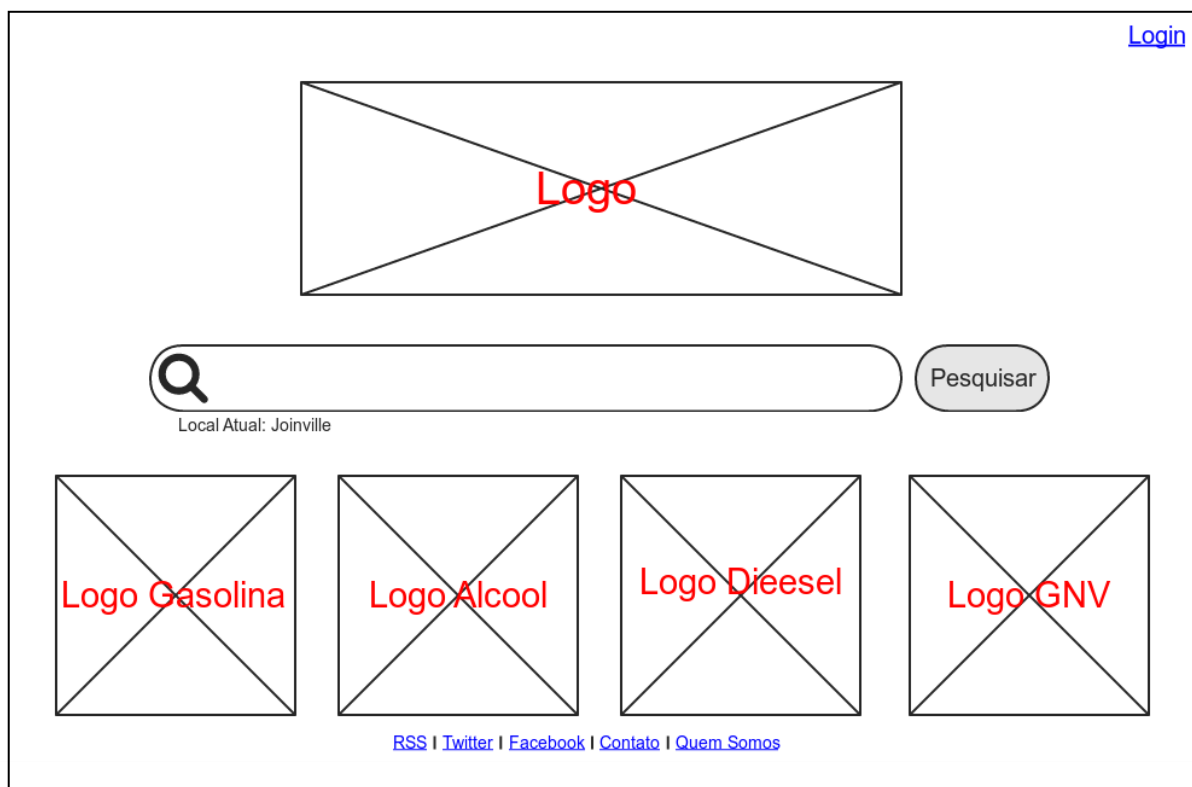


Figura 31 - Wireframe - Home

ANEXO 2 – WIREFRAME – CONTATO

A página de Contato fornece um formulário onde o usuário poderá postar suas dúvidas e comentários acerca dos serviços do Portal. A interface é proposta conforme a Figura 32.

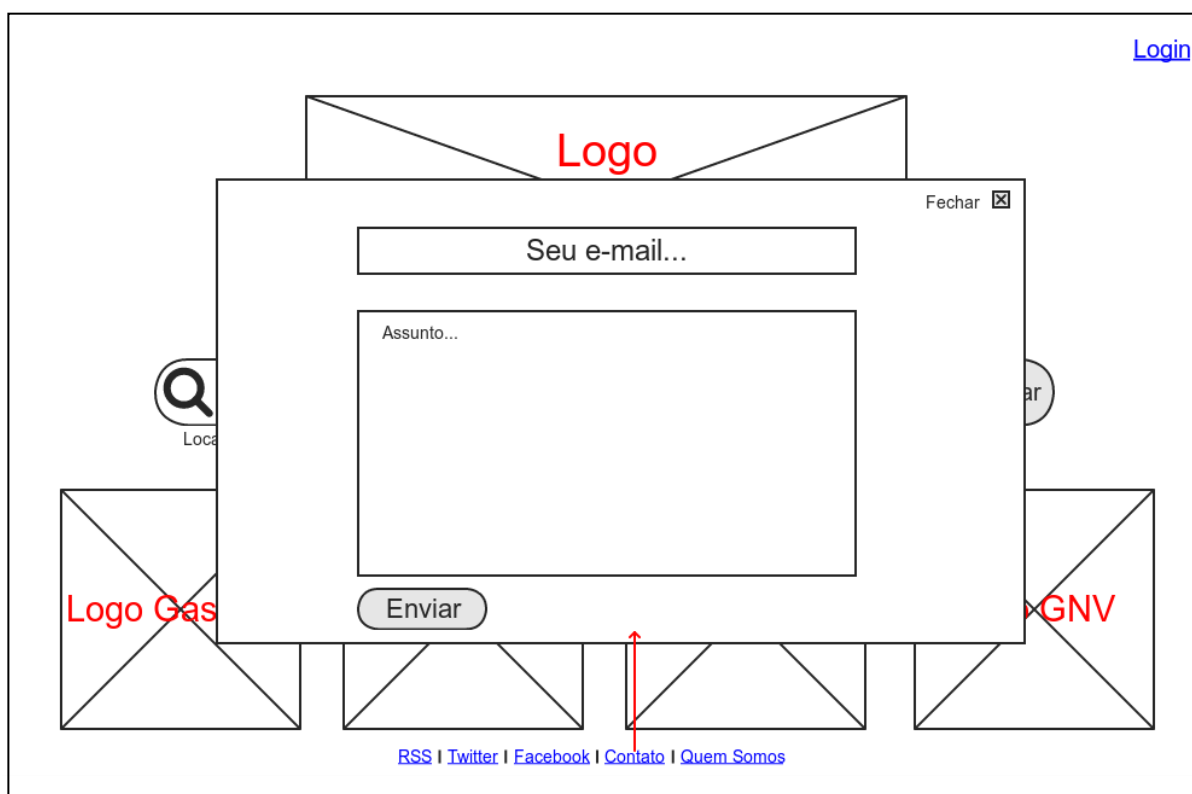


Figura 32 - Wireframe - Contato

ANEXO 3 – WIREFRAME – QUEM SOMOS

A página de Quem Somos visa anunciar as principais informações acerca dos serviços do Portal, o objetivo do serviço e outras informações dos colaboradores. A interface é proposta conforme a Figura 33.

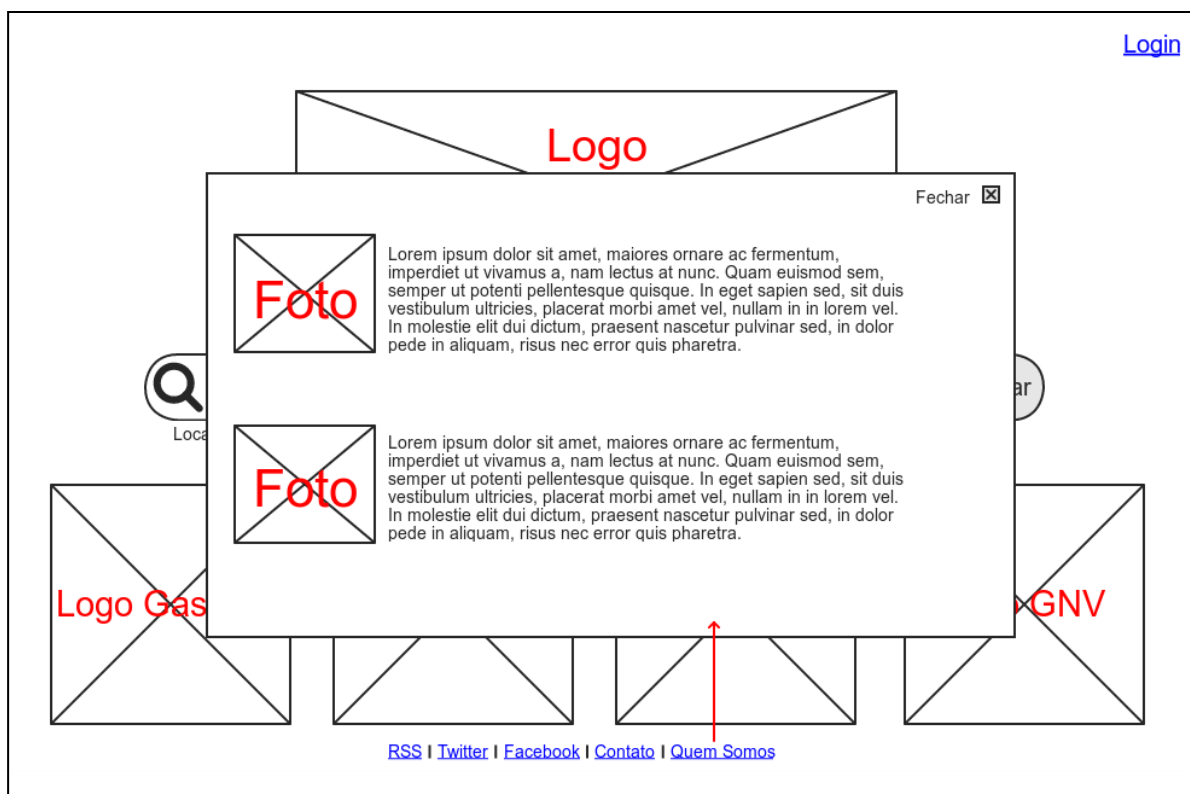


Figura 33 - Wireframe - Quem somos

ANEXO 4 – WIREFRAME – LOGIN

A página de Login permite que os usuário acesse a interface de gerenciamento do sistema, conforme demonstra a

Figura 34.

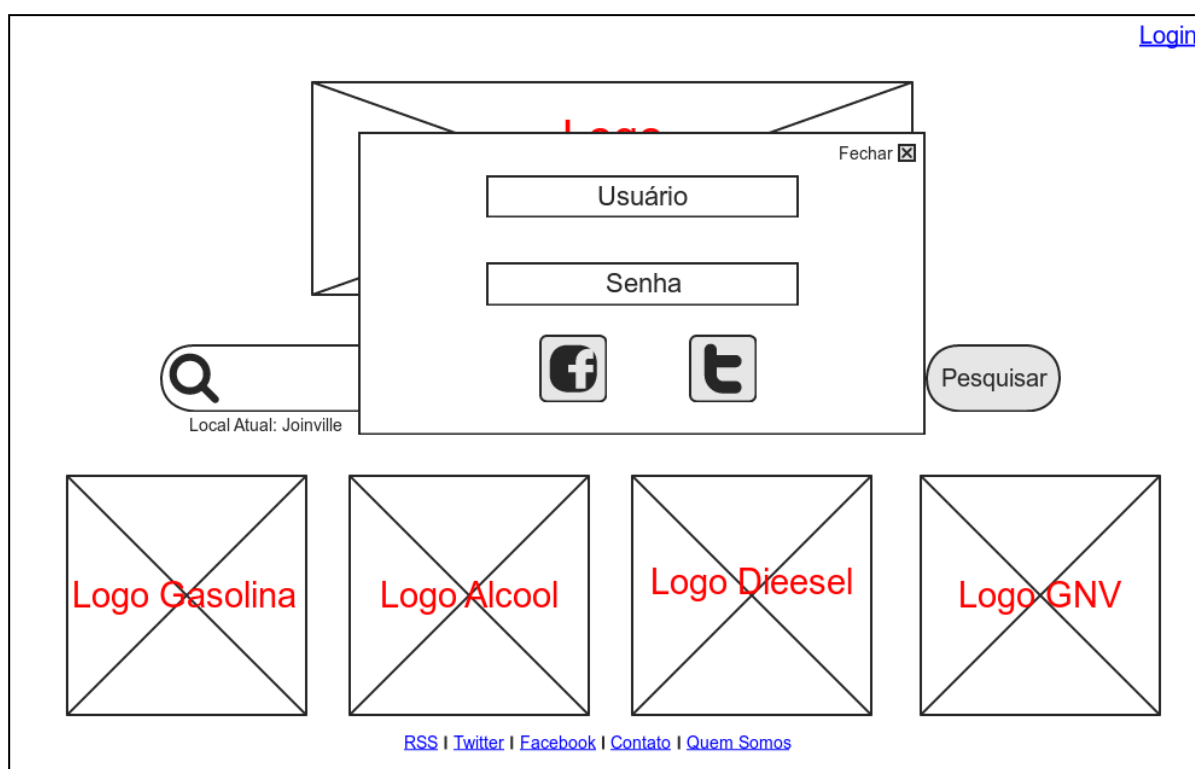
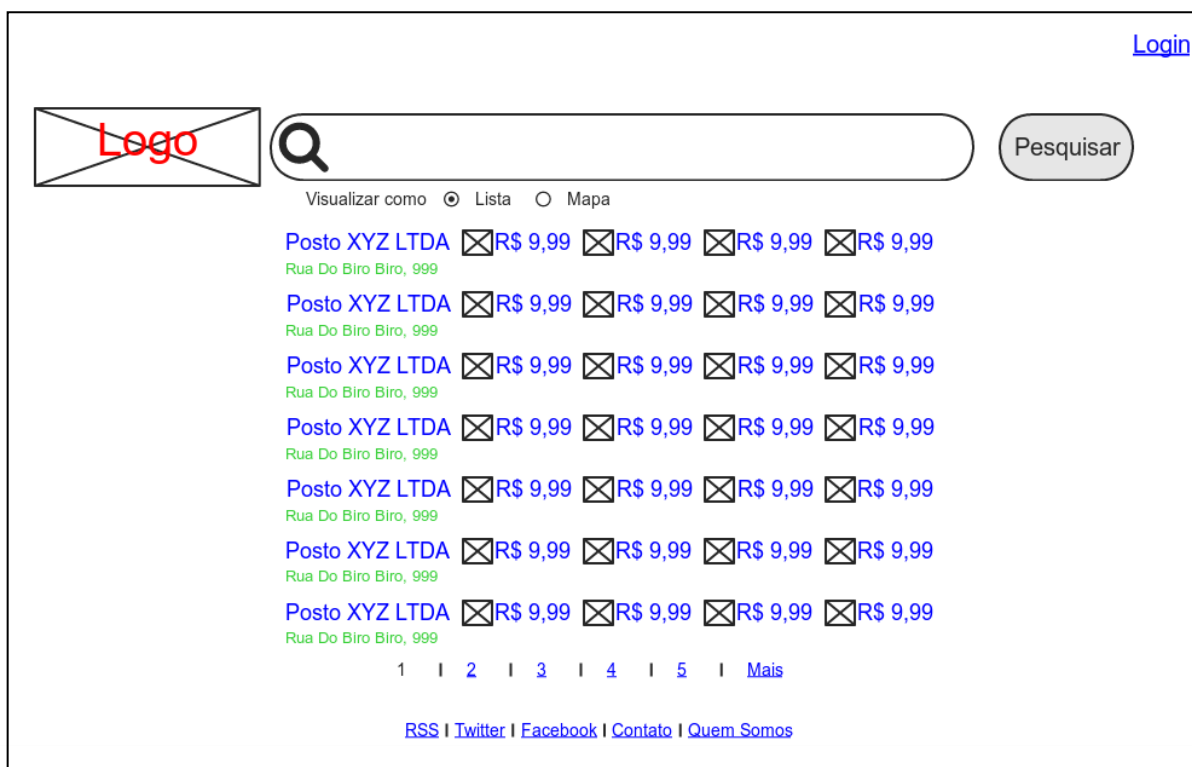


Figura 34 - Wireframe - Login

ANEXO 5 – WIREFRAME – PESQUISA NA LISTA DE PRODUTOS

A página de Pesquisa permite consultar a lista de postos e comparar o preço dos combustíveis, conforme proposto na

Figura 35.



Logo

Q

Pesquisar

Visualizar como ☒ Lista ☐ Mapa

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

Posto XYZ LTDA ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99 ✕ R\$ 9,99
Rua Do Biro Biro, 999

1 | 2 | 3 | 4 | 5 | Mais

[RSS](#) | [Twitter](#) | [Facebook](#) | [Contato](#) | [Quem Somos](#)

Figura 35 - Wireframe - Pesquisa na lista de Postos

ANEXO 6 – WIREFRAME – SEM RESULTADOS PARA PESQUISA

Esta página de resultados pode ser exibida quando a pesquisa não retorna nenhum porto na localidade desejada. Quando ocorrer, uma mensagem poderá ser exibida conforme o exemplo na

Figura 36.



Figura 36 - Wireframe - Sem resultados para pesquisa

ANEXO 7 – WIREFRAME – PESQUISA DIRETAMENTE NO MAPA

Esta página permite que o usuário navegue diretamente no mapa, deslizando a para uma área desejada e visualizando graficamente os postos disponíveis em uma determinada região. O modelo proposto é exibido na

Figura 37.

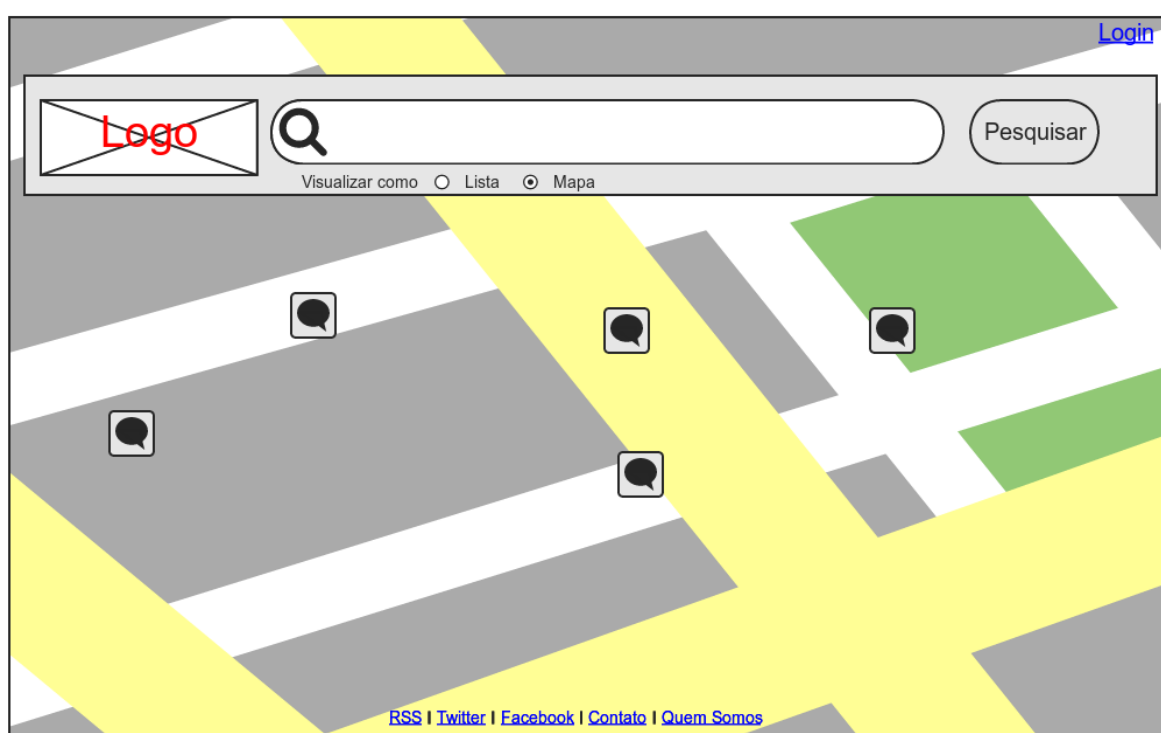


Figura 37 - Wireframe - Pesquisa diretamente no mapa

ANEXO 8 – WIREFRAME – MANUTENÇÃO DE PREÇOS

Esta é a interface de administração de preços dos combustíveis onde o administrador ou proprietário do Posto pode alterar os valores dos combustíveis. Esta tela projetada conforme a Figura 38.


The wireframe shows a web interface for managing fuel prices. At the top right is a [Login](#) link. On the left is a placeholder for a logo. In the center is a search bar with a magnifying glass icon and a 'Pesquisar' button. Below the search bar are radio buttons for 'Visualizar como' with 'Lista' selected and 'Mapa' as an option. The main content area displays 'Posto XYZ LTDA' with a location address 'Rua Do Biro Biro, 999'. Below this is a table of fuel types and their prices, each with a delete icon (a square with an 'X'). A 'Fechar' button with an 'X' icon is in the top right of the table. An 'Atualizar' button is at the bottom center of the table.


Fuel Type	Price	Action
Gasolina	R\$ 9,99	Delete
Alcool	R\$ 9,99	Delete
Dieseel	R\$ 9,99	Delete
GNV	R\$ 9,99	Delete

Figura 38 - Wireframe - Manutenção de Preços


ANEXO 9 – WIREFRAME – INFORMAÇÕES DOS POSTOS

Nesta tela é possível realizar um detalhamento das informações de cada Posto, visualizando comentário de outros usuários e também consultando o histórico dos preços praticados por um determinado Posto, conforme proposto na Figura 39.



Bandeira
William S. de Oliveira




9,99



9,99



9,99




9,99

Gasolina Alcool Diseel GNV

Histórico de Alterações

Data Alterção	Gasolina	Álcool	Diseel	GNV
10/01/2011	R\$ 9.99	R\$ 9.99	R\$ 9.99	R\$ 9.99
10/01/2011	R\$ 9.99	R\$ 9.99	R\$ 9.99	R\$ 9.99
10/01/2011	R\$ 9.99	R\$ 9.99	R\$ 9.99	R\$ 9.99
10/01/2011	R\$ 9.99	R\$ 9.99	R\$ 9.99	R\$ 9.99
10/01/2011	R\$ 9.99	R\$ 9.99	R\$ 9.99	R\$ 9.99
10/01/2011	R\$ 9.99	R\$ 9.99	R\$ 9.99	R\$ 9.99

Comentários



Fulado de tal

Lorem ipsum dolor sit amet, maiores ornare ac fermentum, imperdiet ut vivamus a, nam lectus at nunc. Quam euismod sem, semper ut potenti pellentesque quisque. In eget sapien sed, sit duis vestibulum ultricies, placerat morbi

Postar

[RSS](#) | [Twitter](#) | [Facebook](#) | [Contato](#) | [Quem Somos](#)

Figura 39 - Wireframe - Informações dos Postos

ANEXO 10 – WIREFRAME – ERRO GENÉRICO

Esta página de erro tem por objetivo alertar o usuário na ocorrência de erros genéricos. O modelo proposto é demonstrado na Figura 40.



Figura 40 - Wireframe - Erro genérico