**CONTROL SERVICE**

**Fábio Geovane Porto Vasques1**

1Instituto Federal de Educação, Ciência e Tecnologia Sul-rio-grandense– (IFSul)  
– 96.418-400 – Bagé – RS – Brasil

Fabioportovasques321@gmail.com

**Resumo.** O Controll Service é um sistema de troca de óleo que tem como objetivo auxiliar o lubrificador e seus gestores no gerenciamento das trocas realizadas bem como, saber quando foi realizada a troca do óleo e componentes do veículo, vencimento da próxima troca e até mesmo identificar quais produtos adequados ao veículo. Também é possível emitir relatórios e histórico do veículo com detalhes como última troca de óleo, filtro de ar, de cabine, quantidade de óleo utilizada, quilometragem e informações do veículo quando de posse do antigo proprietário.

1. **1. INTRODUÇÃO**
2. Em toda associação, seja ela de que tipo for, parte crucial do trabalho feito é o de registrar pessoas e ter acesso às suas informações armazenadas. Dependendo do tamanho e da especialização da mesma, as soluções para armazenar e resgatar essas informações variam desde controle em papel, planilhas em Excel, Access até sistemas de maior porte.

Atualmente o sindicato se encontra relativamente informatizado. A maioria das rotinas já se encontram parametrizadas em sistemas, porém ainda existem diversas rotinas ligadas a relatórios que são baseados em planilhas e fichas.

Sindicatos são, em geral, do ponto de vista do quadro de funcionários, empresas de pequeno porte e, como tal, raramente contam com um departamento de informática.

Para o desenvolvimento de sistemas de maior porte, com grande nível de integração, está se tornando cada vez mais necessário a adoção de um framework. Ele é responsável pela organização do código fonte de todo o sistema, além de prover diversas funcionalidades que facilitam o processo de desenvolvimento. O problema surge diante da diversidade de frameworks existentes no mercado, devido à quantidade de requisitos e utilidades diferentes, acaba se tornando necessária uma análise aprofundada antes de uma escolha.

Outro ponto importante, levando em consideração o funcionamento do departamento, é a facilidade de se desenvolver o sistema entre vários desenvolvedores. Devido aos padrões adotados, é possível dividir as tarefas de uma maneira muito mais simples. Diante de todas essas questões, será possível desenvolver um novo sistema, com prazos definidos, robusto e de forma ágil.

1. **2. PROBLEMA**

Os Postos da rede GBI Possuem muitas deficiências no que diz respeito ao gerenciamento de trocas de óleo, entre eles: os controles das trocas são feitos atualmente em folhas de papéis onde é anotado o contato do cliente e a placa do veículo do mesmo, e diariamente o lubrificador responsável pelas trocas de óleo revisa esses documentos para saber se há algum veículo com o vencimento próximo, fazendo assim a ligação para o cliente e notificando o mesmo sobre a troca de óleo. Com isso é gerado outros problemas como:

-perder a informação do cliente já que o contato do mesmo está anotado em folhas de papel;

-ligar mais de uma vez para o mesmo cliente em espaço de tempo menor do que o esperado, pois se o cliente realizou a troca do óleo hoje, ele vai vencer daqui a 6 meses, 1 ano ou dependendo da quilometragem rodada, então não é possível através de anotações prever todas essas situações.

-Como geralmente uma vez por ano e realizado a troca de óleo de veículos comuns, então o normal seria uma vez por ano entrar em contato com o cliente notificando- o do vencimento da troca, porém com essa deficiência pode acontecer de ligar para o cliente e o mesmo informar que não possui mais o veículo e daqui a um ano ou 6 meses ele receber outra ligação do posto avisando novamente da troca, sendo que ele pode não possuir mais o veículo, pois como não há um controle adequado pode vir a acontecer essa situação.

-Outro problema que ocorre e quando os clientes ligam para o posto questionando sobre o vencimento da troca de óleo de seu veículo, pois acontece de perder a etiqueta com os dados da troca, porém o posto não tem uma resposta para fornecer ao cliente.

-Outro problema comum de acontecer, e que no momento da troca pode ocorrer do veículo estar com algum defeito e quando passar algum tempo o proprietário do veículo retornar ao posto alegando que foi o lubrificador quem causou o defeito no mesmo. Sendo que não há como provar que o veículo já estava apresentando o problema durante a troca.

Esses são apenas alguns problemas que podem vir a acontecer por falta de um gerenciamento eficiente.

**3. TRABALHOS CORRELATOS**

O objetivo desse trabalho é apresentar uma possível solução para aprimorar o gerenciamento de troca de óleo dos postos GBI, através do desenvolvimento de um sistema onde será feito o cadastro dos veículos junto a seus respectivos proprietários, será realizado também o cadastro de trocas de óleo dos mesmos com suas datas de trocas e datas de vencimento. Quando faltar 30 dias para o vencimento da troca de óleo, será mostrado uma notificação na dashboard do sistema com os dados do veículo e proprietário, para o responsável entrar em contato com o cliente e avisar sobre o vencimento da troca, caso o mesmo informar que não possui mais o veículo ainda é possível editar o veículo e desativá-lo, se algum dia o mesmo retornar ao posto, é possível ativá-lo novamente e vincular o veículo a um novo CPF/CNPJ. Além dessa funcionalidade ainda é possível emitir relatórios filtrando trocas por datas e por unidades da rede GBI, que por sua vez já irá exibir as informações do nome do trocador que efetuou a troca de óleo.

**4. OBJETIVOS**

O presente artigo tem como objetivo a criação de um mecanismo que auxilie a gestão de trocas de óleo dos postos GBI. Seguem os objetivos especificados desse trabalho:

a) fornecer um controle dos veículos que realizam a troca de óleo, juntamente com seus proprietátios;

b) fornecer relatórios de veículos com suas respectivas datas de trocas de óleo;

c) notificar os responsáveis sobre os veículos que estão com o vencimento próximo ou até mesmo os veículos que já estão com sua troca vencida;

d) Reduzir o índice de problemas com trocas de óleo no que diz respeito a defeitos apresentados no veículo durante a realização da troca, pois através do campo obs dentro do cadastro, é possível inserir uma obs sobre o veículo e logo após a troca será impresso uma via da troca realizada para o cliente assinar;

e) Auxiliar na gestão dos gerentes, pois é possível gerar relatório de trocas por unidades filtrando ainda por intervalo de datas, sabendo assim quais unidades estão realizando mais trocas de óleo.

**5. METODOLOGIA**

As metodologias utilizadas neste trabalho basearam-se em levantamentos de requisitos e conversas com o responsável pela troca de óleo do posto.

Através desses encontros foram levantados pontos importantes sobre o funcionamento do setor e o que necessitava melhorar.

Para criação desse sistema será utilizado a linguagem php e banco de dados mysql.

**5.1. ANÁLISE**

Os métodos para análise e melhoria de processos é um caminho lógico para identificar e solucionar problemas que utilizam, em cada fase, ferramentas apropriadas (barreto , 1999).

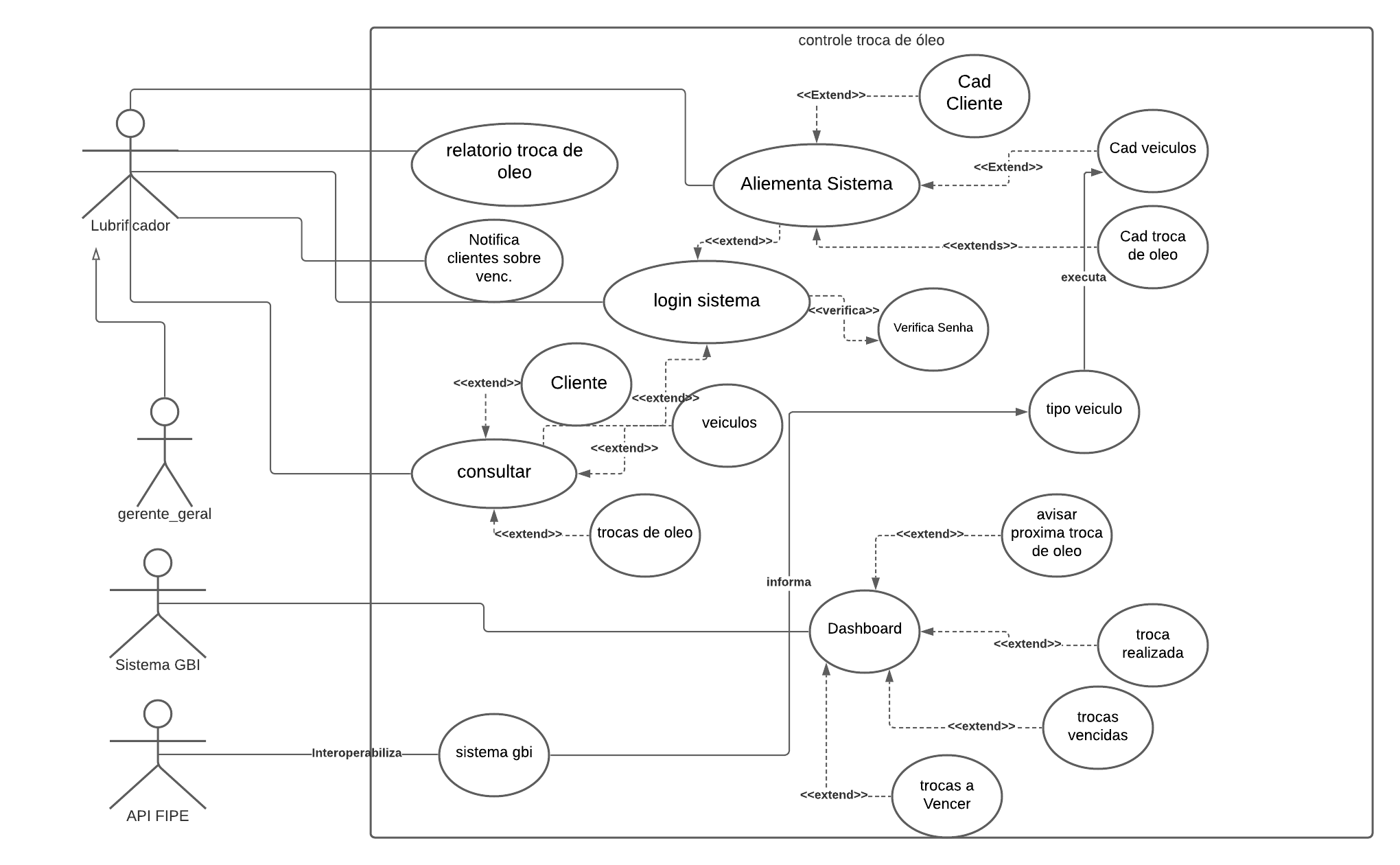
Como parte do estudo de caso, foram realizadas algumas reuniões com o responsável pela troca de óleo do posto GBI unidade 002, afim de realizar uma análise profunda do que o sistema precisaria possuir para satisfazer suas necessidades.

**5.2. PROJETO**

Nesta fase é que deve ser considerado, como o sistema funcionará internamente para que os requisitos possam ser atendidos.

Para o desenvolvimento do projeto, foram utilizados diagramas de acordo com a linguagem de modelagem unifcada (UML), (em inglês Unified Modeling Language – UML) foi concebida com o intuito de estabelecer um padrão único a ser usado para a especificação das características dos sistemas computacionais projetados para atender às necessidades dos usuários desses sistemas (Moraes Pereira), através da ferramenta lucidchart que é um software online de diagramas e comunicação visual, que permite a criação de até 3 documentos de forma gratuita.

Um caso de uso corresponde a um conjunto de ações executadas durante a realização de uma funcionalidade do sistema. Casos de uso concentram-se nas relações entre as funções do sistema e os usuários que delas participam de alguma forma (Moraes Pereira). Segundo o autor, nesse diagrama e exposto como o usuário se relaciona com o sistema, e as iterações que ocorrem entre o mesmo com o sistema ou de algum outro sistema com o próprio sistema.



**5.3. TRABALHOS CORRELATOS**

Após a realização de pesquisa a respeito de trabalho com o propósito semelhante encontrou-se o: “Um Sistema de Gestão de Associados para Sindicato”. O Software foi aplicado no CRDD/RJ (Conselho Regional dos Despachantes Documentalistas do Estado do Rio de Janeiro) que são tratados pelo sistema. Depois de realizar a primeira versão de teste de número 1.0 a versão 1.2 está em pleno funcionamento desde o ano de 2016 no CRDD/RJ.

**5.4. FERRAMENTAS SELECIONADAS**

Tabela 1. Utilizada como comparação do CRDD/RJ

|  |  |  |  |
| --- | --- | --- | --- |
| **Ferramenta** | **Tipo** | **Versão** | **Licença** |
| **HTML** | Linguagem de marcação | 5 | Freeware |
| **CSS** | Folha de Estilo | 3 | Freeware |
| **Jquery** | Biblioteca JavaScript | 2.0 | Freeware |
| **PHP** | Linguagem de Programação | 5.2.9 | Freeware |
| **MySQL** | Gerenciador de Banco de Dados | 5.1.48 | Freeware |
| **Atom** | Editor de Código | 1.0.11 | Freeware |
| **Astah** | Modelagem de Dados | 6.6.3 | Freeware |

.

Tabela 2. Comparação entre os Métodos de Segurança adotados pelo Frameworks

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MÉTODO** | **CAKEPHP** | **CODEIGNITER** | **SYMFONY** | **YII** | **ZEND FRAMEWORK** |
| Validação de Formulários | V | V | V | V | V |
| SQL Injection | V | V | V | V | V |
| Script Injection | V | - | V | V | V |
| Módulo de autenticação | V | V | V | V | V |

1. **6. TECNOLOGIAS UTILIZADAS**

A implementação e modelagem de banco de dados será feita utilizando o IDE (Integrated Development Environment). E o SGBD (Sistema de Gerenciamento de Banco de Dados) selecionado foi o SQL. Este conjunto de ferramentas foi escolhido levando em consideração a adequação ao tipo de aplicação proposta, além do fato de sua utilização ser gratuita. A integração entre a aplicação proposta e seu respectivo banco de dados será realizada por meio da suíte de aplicativos, que para este trabalho, oferece a implementação de um servidor de banco de dados SQL.

A presente proposta consiste no desenvolvimento de um software desktop, livre e gratuito, que execute em múltiplas plataformas, para auxiliar no gerenciamento de cadastro dos sócios.

A linguagem de programação será Java, junto ao XML para estilização das interfaces. Em seu banco de dados será utilizada a biblioteca SQlite que faz uma adaptação no banco de dados SQL dentro da aplicação.

1. **6.1. REQUISITOS DO SISTEMA**
2. Para o desenvolvimento de um sistema funcional e eficiente que seja realmente eficaz é indispensável um excelente levantamento de requisitos e o mapeamento desse processo.
3. Um requisito de sistema consiste numa descrição mais detalhada das características do sistema de forma que se possa compor um contrato entre quem desenvolve o sistema e o cliente. Desta forma, um documento de requisitos deve ser tanto técnico quanto possível sem, contudo, tornar-se incompreensível a um leigo. De maneira geral, os requisitos do sistema descrevem o que o sistema deve fazer e não o como deve ser feito. Os requisitos possuem, por convenção, uma referência que segue o modelo (nome do requisito.identificador do requisito, sendo desta maneira tratados e referenciados nos demais documentos que compõem um projeto de desenvolvimento de sistema.
4. Para o levantamento dos requisitos de um sistema diversas técnicas podem ser aplicadas, destacando-se a Amostragem, Investigação, Entrevistas, Questionários, Observação e Prototipação. Neste trabalho foi utilizada a técnica de Entrevistas Não Estruturadas pela facilidade de aplicação e para favorecer o contato direto com os envolvidos.
6. **6.2. REQUISITOS FUNCIONAIS**
7. Os requisitos classificados como funcionais descrevem operações que o sistema deve ser capaz de realizar. Nestes requisitos são especificadas as entradas e saídas do sistema, bem como o relacionamento entre elas.
8. **Login de Usuário**
9. **Descrição:** Qualquer usuário cadastrado e ativo poderá logar no sistema.
10. **Entrada:** Nome de usuário, senha.
11. **Processo:** A entrada será validada no banco e gerada uma nova sessão.
12. **Saída**: Mensagem de confirmação caso a entrada seja válida, senão, mensagem de erro.
13. **Cadastro de Usuário**
14. **Descrição:** Apenas usuários com permissão podem se cadastrar no sistema.
15. **Entrada:** Nome de usuário, senha, nível de permissão e e-mail.
16. **Processo:** Os dados de entrada serão incluídos no banco de dados.
17. **Saída:** Mensagem de confirmação caso a inclusão tenha sido realizada com sucesso, senão, mensagem de erro.

**Edição de Cadastro de Usuário**

**Descrição:** O usuário poderá editar suas informações.

**Entrada:** Nome de usuário, senha e e-mail.

**Processo:** O dados de entrada serão atualizados no banco de dados.

**Saída:** Mensagem de confirmação caso a atualização tenha sido realizada com sucesso, senão, mensagem de erro.

**Desativação de Usuário**

**Descrição:** Apenas usuários com permissão, ou seja, quando o associado tiver solicitado a exclusão do quadro de sócios.

**Entrada:** Nome de usuário

**Processo:** O sistema verifica se o usuário informado é cadastrado e, em caso positivo, o inativa.

**Saída:** Mensagem de confirmação caso a desativação tenha sido realizada com sucesso, senão, mensagem de erro.

1. **6.3. REQUISITOS NÃO FUNCIONAIS**

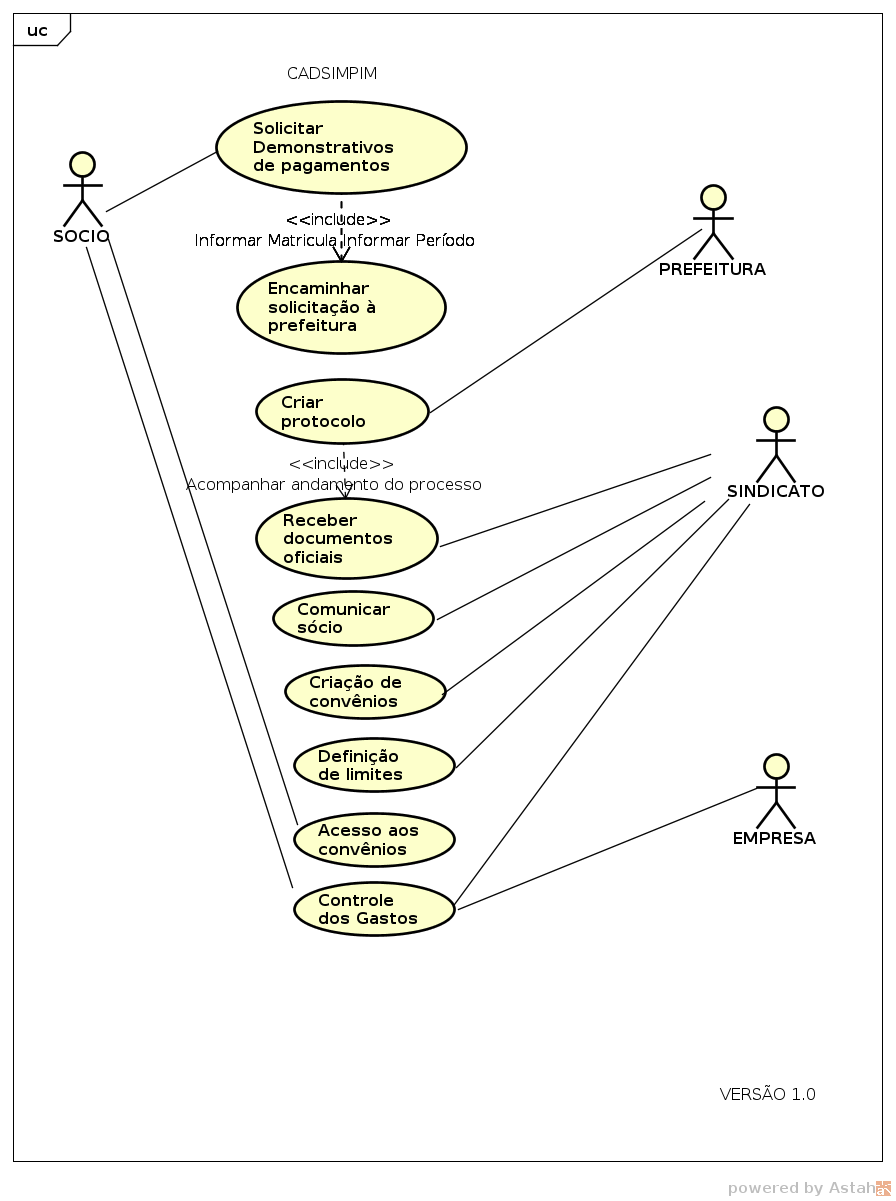
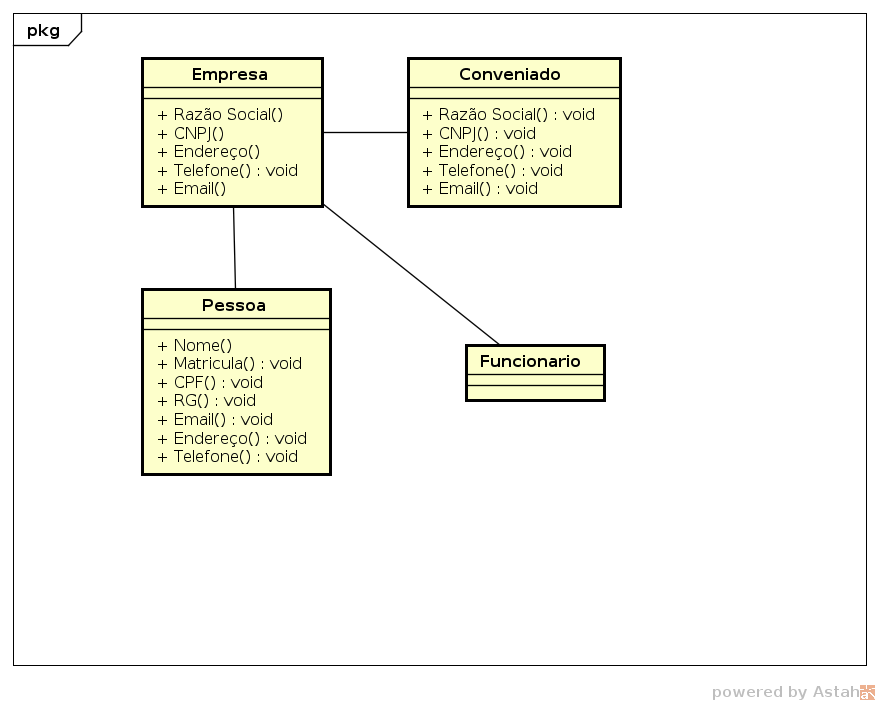
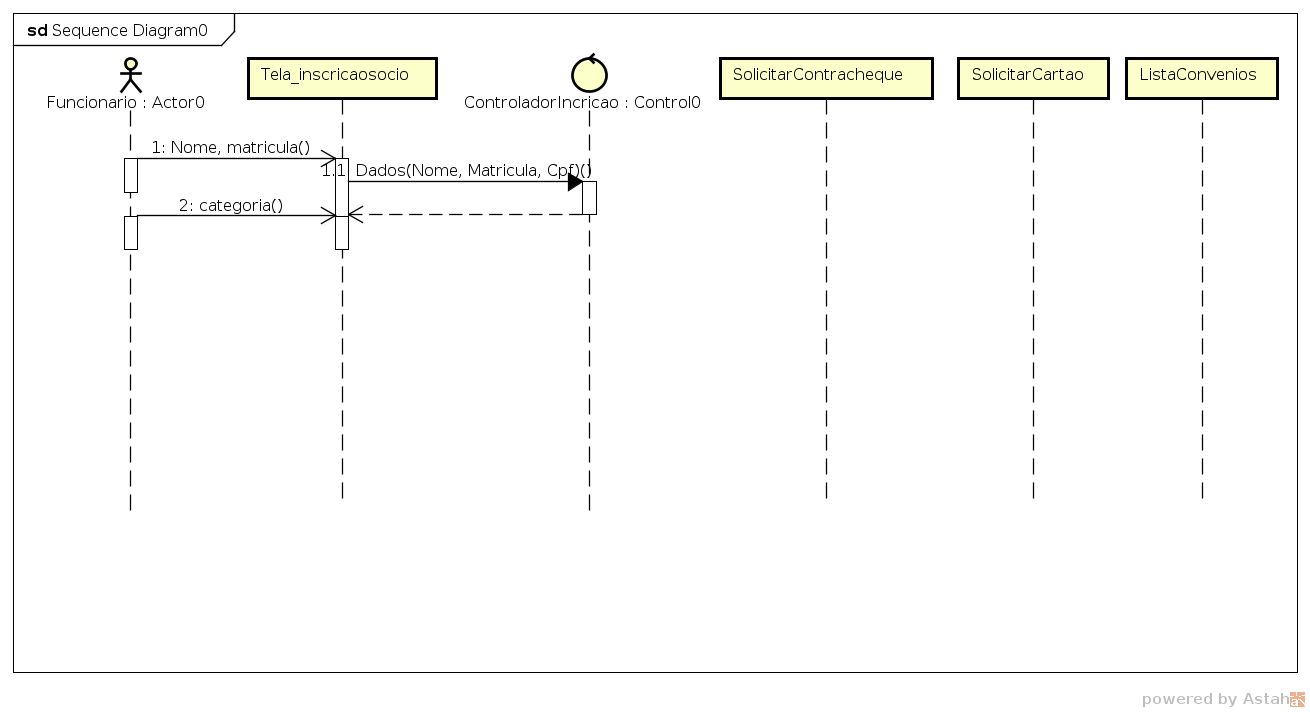
Os requisitos não funcionais são aqueles que não apresentam relação direta com as funcionalidades do sistema e, desta maneira, complementam os requisitos funcionais. Essa categoria de requisito está ligada a restrições no sistema, no processo de desenvolvimento ou mesmo restrições externas que o sistema precisa atender.

**Usabilidade:** O sistema deverá possuir uma interface amigável ao usuário mais leigo permitindo fácil acesso mesmo para funções de alta complexidade.

**Interface**: O usuário terá acesso ao sistema através de qualquer navegador WEB (web browser) com suporte ao HTML5.

**SGBD**: O Sistema de Gerenciamento de Banco de Dados (SGBD) utilizado será o PostgreSql; Em seu banco de dados será utilizada a biblioteca SQlite que faz uma adaptação no banco de dados SQL dentro da aplicação.

**Linguagem de Programação:** A linguagem de programação será Java, junto ao XML para estilização das interfaces.

1. **6.4. ATRIBUTOS DO SISTEMA**
2. **Disponibilidade**
3. O Sistema deverá estar sempre disponível e, caso ocorra alguma interrupção, ele deverá ser restabelecido o mais breve possível.
4. **Segurança**
5. Como o acesso será via WEB, o quesito segurança deverá ser observado em cada etapa do desenvolvimento a fim de evitar ataques e acessos não autorizados.
6. **Diagrama de Casos de Uso**
7. 
8. **Diagrama de Classe**
9. 
10. **Diagrama de Sequência**
11. 

**8. RESULTADOS**

Após todo o período de desenvolvimento e a devida homologação para uso através de uma bateria de testes, inevitavelmente chegará o momento em que serão necessárias alterações no sistema. Entra novamente em cena o *framework*, dele dependerá a facilidade de se desenvolver funcionalidades novas, alterar o funcionamento de fluxos anteriormente utilizados, e até mesmo corrigir eventuais problemas que passaram despercebidos pela fase de testes. Geralmente é nesse ponto que muitas complicações ocorrem.

As manutenções conseguem ser feitas com sucesso, porém a um alto custo. Isso pode ser evitado com uma organização estrutural do código já preocupada com as futuras manutenções.

Um *framework* que possibilite eventuais alterações, e que não seja totalmente engessado, ajudará muito no momento de se expandir módulos específicos do sistema.

1. **9. CONCLUSÃO**

Com a escolha em se utilizar um framework, passamos a seguir um padrão. Padrão esse que já foi exaustivamente testado, e geralmente seguem regras de boas práticas em programação. Todos os frameworks utilizam linguagem PHP, que permite um bom desempenho e segurança razoável, se adotadas as medidas de segurança necessárias.

Por questões de legibilidade de código e principalmente para facilidade de manutenção, é utilizada a orientação a objetos em todos os frameworks que, junto com a divisão em camadas (modelo, visão e controle) torna a manutenção do sistema mais simples e clara, além de possibilitar o trabalho com diversos profissionais ao mesmo tempo.

A escolha de um framework deve ser muito bem baseada, pois mesmo se tratando de frameworks semelhantes, utilizando a mesma linguagem e metodologias, existem diversos parâmetros diferentes. A análise do sistema que será desenvolvido deve ser detalhada, pois é a partir desse detalhamento que será possível definir qual framework atenderá melhor às necessidades da empresa.

Com a implementação do sistema desenvolvido ao longo deste trabalho, foi possível notar um ganho na qualidade das informações armazenadas no banco pois agora elas são validadas antes de serem salvas, tendo a oportunidade de correção.

1. **10. REFERÊNCIAS**

Zend Technologies. Programmer’s Reference Guide of Zend Framework 2 http://framework.zend.com/manual/2.2/en/index.html . 2013. (Acessado em outubro de 2013).

Zend Technologies. Zend Framework 2. The most popular framework for modern, high-performing PHP applications. http://www.zend.com/. 2013. (Acessado em outubro de 2013).

CRDD/RJ. Institucional, Quem somos.<http://www.crddrj.com.br/institucional> Acesso em 10 ago. 2015.