Unioeste - Universidade Estadual do Oeste do Paraná
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
Colegiado de Ciência da Computação
Curso de Bacharelado em Ciência da Computação

SIGVERTE Sistema de Gestão de Vertedouro: da especificação aos testes com o primeiro módulo do sistema

Rodrigo Senger

CASCAVEL
2015

Rodrigo Senger

SIGVERTE Sistema de Gestão de Vertedouro: da especificação aos testes com o primeiro módulo do sistema

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação, do Centro de Ciências Exatas e Tecnológicas da Universidade Estadual do Oeste do Paraná - Campus de Cascavel

Orientador: Prof. Dr. Rogério Luis Rizzi

CASCAVEL
2015

Rodrigo Senger

SIGVERTE Sistema de Gestão de Vertedouro: da especificação aos testes com o primeiro módulo do sistema

Monografia apresentada como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação, pela Universidade Estadual do Oeste do Paraná, Campus de Cascavel, aprovada pela Comissão formada pelos professores:

Prof. Rogério Luis Rizzi (Orientador)
Colegiado de Ciência da Computação,
UNIOESTE

Prof. Ivonei Freitas da Silva (Co-Orientador)
Colegiado de Ciência da Computação,
UNIOESTE

Prof. Claudia Brandelero Rizzi
Colegiado de Ciência da Computação,
UNIOESTE

Pétterson Vinicius Pramiu
Bolsista DTI vinculado ao CEASB/PIT/ITAIPU

Cascavel, 15 de Fevereiro de 2016

EPÍGRAFE

“Se há algum segredo de sucesso, consiste ele na habilidade de apreender o ponto de vista da outra pessoa e ver as coisas tão bem pelo ângulo dela como pelo seu.“ - Henry Ford

“Siga sua felicidade, e o universo vai abrir portas para você onde só havia paredes.“ - Livro The Secret

“A força não vem de vencer. Suas lutas desenvolvem suas forças. Quando você atravessa dificuldades e decide não se render, isso é força.“ - Arnold Schwarzenegger

AGRADECIMENTOS

Agradeço primeiramente minha a família, minha mãe Haidi Lange Senger, meu pai Sergio Senger e minha irmã Ana Thais Senger que sempre me apoiaram incondicionalmente. Minha namorada Silmara Juliana Likoski que sempre me apoiou e deu forças durante todo esse tempo.

Agradeço especialmente aos professores que me aturaram durante quase todo esse período como acadêmico, Claudia Brandelero Rizzi e Rogério Luis Rizzi, pelas oportunidades em projetos de Iniciação Científica, Estágio e orientação deste trabalho de conclusão de curso.

Agradeço a todos os professores que fizeram parte deste longo percurso, que de uma forma ou de outra me moldaram como pessoa e profissional, em especial ao Professor Ivonei Freitas da Silva pela importante orientação e contribuição neste trabalho.

Agradeço à Lyssa Scherer e ao Pétterson Vinicius Pramiu pelas contribuições e pelo envolvimento no projeto que vai além deste trabalho.

Agradeço aos amigos de faculdade que conviveram comigo nesses últimos anos, Thiago Pessini, Daniel Bortoluzzi, Eduardo Vansetto, Alexandre Luiz Borba Silva, Lucas Pereira, Alexandre Henrique Unfried e André Unfried, pelas diversas noites mal dormidas realizando trabalhos ou estudando pra provas e também pelas festas e balangas bem aproveitados junto de vocês.

Enfim, muito obrigado a todos!

Lista de Figuras

Lista de Abreviaturas e Siglas

SIGVERTE	Sistema de Gestão de Vertedouro
FDD	Feature Driven Development
IDE	Ambiente de Desenvolvimento Integrado
MVP	Mínimo Produto Viável
DAO	“Data Acess Object“
SSB	Sistema de Segurança de Barragens
SYSDAM	Sistema de Gestão de Segurança de Barragens

Sumário

Resumo

Este trabalho de conclusão de curso apresenta o desenvolvimento do Sistema de Gestão de Vertedouro - SIGVERTE, cujo objetivo é gerenciar dados e subsidiar tomadas de decisões em relação à gestão e ao monitoramento de determinados desgastes que ocorrem em vertedouros de usinas hidroelétricas. O Sistema foi desenvolvido utilizando a metodologia ágil de desenvolvimento de software *feature driven development*, que viabiliza a entrada dos artefatos pré-existentes, assim como a atualização e a documentação do desenvolvimento do Sistema, proporcionando o entendimento do domínio e sua validação com específicos testes. Utilizou-se, também os padrões de projeto DAO e Observer. Georreferenciou-se funcionalidades como a localização da Calha e Atividades, Amostras e Campanhas cadastradas nela. Os requisitos desenvolvidos disponibilizaram as funcionalidades de Gerenciar Calha, Lajes, Campanhas e Atividades. Trata-se de um software que atendeu os requisitos desejados, além de possibilitar a fácil manutenção e extensibilidade.

Palavras-chave: SIGVERTE, Engenharia de Software, FDD, Desenvolvimento Web, Desenvolvimento Ágil de Software.

Capítulo 1

Introdução

Neste capítulo será apresentada a motivação para implementação do SIGVERTE - Sistema de Gestão de Vertedouro, os objetivos deste trabalho de conclusão de curso e tecnologias escolhidas para dar suporte.

1.1 Motivação

A Usina Hidrelétrica de Itaipu é uma empresa binacional localizada na cidade de Foz do Iguaçu, no Rio Paraná, na fronteira entre Brasil e Paraguai. Construída por esses dois países no período de 1975 a 1982 tem capacidade atual instalada de geração da usina de 14 GW, com 20 unidades geradoras, cada uma delas fornecendo 700 MW. No ano de 2012, ela atingiu o recorde de produção antes alcançado em 2008, com 98.287.128 megawatts-hora [1].

Sua manutenção requer estudos específicos à sua conservação ao longo do tempo. Dentre os elementos de maior importância nos processos de manutenção estão os vertedouros, estruturas hidráulicas que estão sob ação erosiva devido à distintos meios de desgaste, já que são responsáveis por liberarem os excedentes de água e regular o nível do reservatório. O escoamento de água aliado a componentes químicos podem acarretar dano ao concreto constituinte da estrutura do vertedouro, comprometendo a sua estabilidade e segurança.

Como a Usina de Itaipu encontra-se em operação desde 1984, as estruturas estão evelhecendo, e dada sua importância a Usina Hidrelétrica deve manter alto padrão de segurança e de produtividade. A consequente manutenção para a conservação do vertedouro requer investimentos significativos tanto de pessoal quanto de material. E a tomada de decisões sobre qual ação realizar, em que local e em qual momento é influenciada por diversos fatores, dentre

os quais, conhecimento empírico dos peritos envolvidos.

Objetivando subsidiar a melhor tomada de decisão com relação à gestão da manutenção do vertedouro, estão sendo realizadas ações e estudos visando propor uma metodologia específica para localizar os danos superficiais existentes nas lajes do concreto, identificando e classificando aqueles significativos. Dentre os recursos necessários para a implantação e adoção desta metodologia está o desenvolvimento de um software (SIGVERTE) que viabilizará, dentre outros aspectos, o gerenciamento de pontos estratégicos, pontos amostrais identificados na calha e manutenções nela realizadas.

SIGVERTE é um software de gestão projetado em dois grandes módulos. Um para gestão de dados do vertedouro disponibilizando a visualização em um mapa georreferenciado que contempla a calha esquerda do vertedouro de Itaipu. A estes mapas será possível, através de filtros e funções específicas, visualizar pontos amostrais, atividades, campanha e as lajes que compõem a calha em estudo. No segundo módulo deste software serão implementados os critérios especificados pela equipe técnica vinculada ao projeto denominado Simulação de efeitos erosivos no Vertedouro da Barragem de Itaipu. Codificadas computacionalmente como funcionalidades do software, será possível vibializar aos gestores, subsídios adicionais à tomada de decisão, como, por exemplo, a comparação de resultados de mesmos pontos amostrais colhidos em momentos diferentes e, acompanhar o desgaste com dados coletados nas campanhas.

Para que o desenvolvimento deste primeiro módulo de gestão que é extensível e precisa ser implementado com qualidade e bem documentado para os futuros projetos que serão desenvolvidos junto dele, uma metodologia ágil de desenvolvimento de software será adotada para atender requisitos básicos, tais como a documentação mínima, porém suficiente para comunicação com futuros desenvolvimentos, entrega contínua do software e boa relação do escopo por tempo de desenvolvimento. Além disso, padrões de projetos serão aplicados para garantir um dos princípios básicos de metodologias ágeis, a excelência técnica no projeto de software [2].

1.2 Objetivos

A proposta desse trabalho de conclusão de curso consistiu em realizar a engenharia de requisitos de software para os dois módulos e utilizar uma metodologia ágil de software junto de padrões de projeto para implementar o módulo de gestão do SIGVERTE. A equipe técnica do projeto de Simulação de Efeitos Erosivos da Barragem de Itaipu atuará como parte interessada nos processos de coleta, análise, avaliação e validação dos requisitos deste trabalho. Atuará também avaliando o software desenvolvido. Os objetivos específicos do trabalho são:

- Documentar os requisitos do Sistema.
- Utilizar uma metodologia ágil de software específica e adequada para o desenvolvimento do software.
- Definir a arquitetura do Sistema.
- Utilizar padrões de projeto que forneçam boas soluções para problemas identificados em diversas atividades do desenvolvimento do software.
- Modelar o banco de dados necessário ao software.
- Implementar, testar e avaliar o primeiro módulo do Sistema.

1.3 Metodologia

Desenvolvimento de software em cenários onde os requisitos e as tecnologias não são totalmente conhecidas, são voláteis e que os "stakeholders" do software necessitam de versões parciais do futuro sistema. Isso requer que a metodologia de desenvolvimento de software lide com as incertezas ao desconhecimento e mudança dos requisitos e tecnologias durante a construção do software, fornecendo de maneira planejada as entregas parciais [3].

Nesse contexto a metodologia de desenvolvimento de software "feature-driven development" (FDD), ou Desenvolvimento Dirigido por Características, é construída sobre um conjunto de melhores práticas, que visam atender aos cenários mencionados. Ela foi projetada para entregar resultados tangíveis de desenvolvimento de software, principalmente o próprio

software executando com frequência [3]. O FDD consiste de cinco processos, que são ilustrados na Figura 1.1.

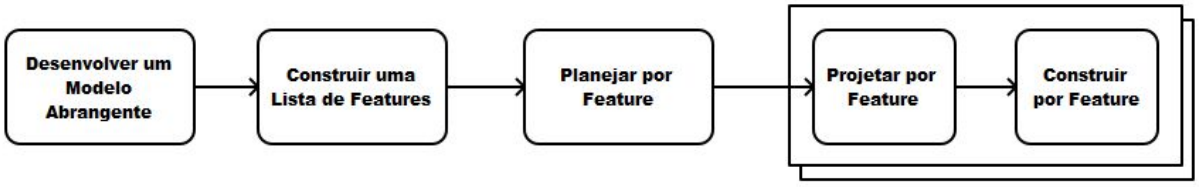

Figura 1.1: Cinco Processos do FDD.

No primeiro processo, "Desenvolver um Modelo Abrangente", o escopo do Sistema é definido em termos de objetos de domínio que o constituirão. No segundo processo, "Construir uma Lista de Features", uma listagem minimamente completa de requisitos é definida. O terceiro processo, "Planejar por Feature", ordena um conjunto de features em um plano e associa-as aos programadores-chefe. No quarto e quinto processos, "Projetar por Feature"/"Construir por Feature", os desenvolvedores projetam, com inspeção, diagramas para as features e implementam-as, com testes e inspeção de código [3].

É relevante, pois, destacar a vantagem do uso da metodologia FDD no desenvolvimento do Sistema de Gestão de Vertedouro. Ela considera explicitamente em alguns dos seus processos a existência de artefatos de entrada que foram anteriormente desenvolvidos. Por exemplo, existem descrições textuais, como protótipos em tela, de vários requisitos do Sistema a ser construído. Estes artefatos foram utilizados como entrada para os processos "Desenvolver um Modelo Abrangente", “Construir uma Lista de Features“ e "Planejar por Feature". Além desta vantagem, pode se destacar que o FDD constitui-se de práticas ágeis organizadas em um processo mais formal do que outros métodos ágeis, como por exemplo, o XP[4]. Ou seja, FDD define uma sequência lógica para executar essas práticas ágeis.

A Arquitetura do Sistema está baseada nos atributos de qualidade, tais como funcionalidade e confiabilidade [5] conforme projeto que foram escolhidos. Inicialmente, como o Sistema deve ser executado via Browser/Internet, a arquitetura em camadas é adotada. Juntamente com a arquitetura, padrões de projeto foram adotados para que as características do Sistema sejam organizadas e implementadas de acordo com determinados atributos de qualidade. Por exemplo,

incialmente, deseja-se que o Sistema seja facilmente extensível, ou seja, novos requisitos sejam facilmente acoplados aos existentes. Neste caso, padrões de projeto que permitam a extensão de funcionalidades, tais como, “DAO“ que permite encapsular os métodos de manipulação do banco de dados em um objeto [2] deveria ser considerado no projeto das classes do Sistema.

As tecnologias utilizadas no desenvolvimento do Sistema devem atender a sua característica de funcionamento integrado via internet. Devem ser estáveis, portáveis e gratuitas. As seguintes tecnologias foram usadas para alcançar os objetivos deste trabalho:

- Sistema de gerenciamento de dados: PostGreSQL - Sistema de gerenciamento de banco de dados objeto-relacional baseado no POSTGRES versão 9.5 desenvolvido pelo departamento de Ciência da Computação da Universidade da Califórnia em Berkeley [6].

- PostGis - Extensão para o PostgreSQL que permite o uso do mesmo como um banco de dados espacial para Sistemas de Informação Geográfica [7].

- PhotoModeler - Para captura e tratamento de imagens. [8].

- Linguagem de programação Java - Orientada a objetos e multiplataformas. Os programas implementados na linguagem Java podem ser executados em qualquer sistema operacional, desde que o mesmo tenha o interpretador da linguagem instalado. Possui bastante documentação e material para estudo da mesma. No contexto deste trabalho, ela será usada para implementar o “server-side“ ou lado do servidor, que seriam as funções responsáveis por manipular os dados no servidor [9].

- Linguagem de programação JavaScript - Usada para programação do HTML e da Web. Neste trabalho, sua função é implementar o client-side, lado do cliente em português, que seriam as funções de interação com o usuário [10].

- As interfaces serão desenvolvidas com HTML5 e Css - O HTML5, evolução do HTML, é uma linguagem de marcação, utilizada para criar páginas web [11] e o Css é uma linguagem de estilo usada para descrever a apresentação de uma página escrita em HTML [12].

- Framework Bootstrap - Grátis e de código aberto, é uma coleção de ferramentas para criar

websites e aplicações web. Contém designs base de HTML e Css, além de ter extenções opcionais para JavaScript[13].

- Utilização do framework Play para desenvolvimento web - É um framework de alta produtividade usando Java ou Scala. Mapeando facilmente os componentes necessários para uma aplicação web [14].
- Georreferenciamento da calha usando a biblioteca OpenLayers - Biblioteca de funções JavaScript para manipulação de dados usados em mapas computacionais [15].
- GeoJSON - É um formato para codificar as variedades de dados geográficos [16].

1.4 Organização do Texto

Os capítulos que compõem este trabalho de conclusão de curso estão assim distribuídos:

- Capítulo 2: Fundamentação Técnica, apresenta sistemas correlatos e tecnologias envolvidas.
- Capítulo 3: Aspectos Metodológicos e Padrões, traz detalhes e justificativa da metodologia e padrões de projeto escolhidos.
- Capítulo 4: Requisitos e Prototipação do Sistema, mostra o protótipo do módulo a ser implementado e seus requisitos.
- Capítulo 5: O Desenvolvimento do SIGVERTE, o desenvolvimento do SIGVERTE nos 5 processos do FDD.
- Capítulo 6: A Validação do SIGVERTE, a validação da conformidade do Sistema com os seus requisitos.
- Capítulo 7: Considerações Finais e Trabalhos Futuros, a conclusão do trabalho e sugestões de trabalhos futuros.

Capítulo 2

Fundamentação Técnica

Neste capítulo são apresentados os sistemas correlatos, ou seja, sistemas que possuem funcionalidades que de alguma forma são semelhantes com o Sistema de Gestão de Vertedouros - SIGVERTE, desenvolvido neste trabalho de conclusão de curso. E também contém as tecnologias escolhidas com suas devidas justificativas para este projeto.

2.1 Sistemas Correlatos

O levantamento realizado através de uma pesquisa na internet indicou a escassez de sistema de gestão de segurança em barragens e gestão em vertedouros. Os encontrados estão descritos a seguir.

SSB - Sistema de Segurança de Barragens: Desenvolvido pela FURNAS. Software que possibilita o gerenciamento on-line dos dados e medidas dos diversos instrumentos instalados em todas as barragens da Empresa, e que já está em funcionamento como projeto-piloto na Usina Hidrelétrica de Funil, RJ [17].

SysDAM - Sistema de Gestão de Segurança de Barragens: Desenvolvida pela Pimenta de Avila Consultoria Ltda [18]. O software possibilita o cadastro de características da barragem e suas estruturas associadas. Permite o monitoramento, com inserção de dados de leituras de instrumentos de rede de monitoramento geotécnico, planilhas de inspeção regulares e especiais, e cadastro de anomalias. Apresenta gráficos com consolidação de informações ao longo do tempo da rede de monitoramento geotécnica, pluviometria e nível de reservatório. Realiza análise de riscos. Elabora automaticamente o plano de segurança de barragens. Possibilita o cadastro e gestão de auditorias técnicas de segurança. Apresenta, de maneira conjunta, todas as

recomendações e planos de ações associadas aos resultados das auditorias, inspeções e análises de risco realizadas.

O SIGVERTE traz abordagens diferentes que ambos os Sistemas apresentados a cima, subsidiando melhor tomada de decisões relação à gestão e ao monitoramento de determinados desgastes que ocorrem em vertedouros de usinas hidroelétricas através dos dados cadastrados ou de simulação.

2.2 Tecnologias

Para a escolha das tecnologias utilizadas no desenvolvimento deste trabalho de conclusão de curso, já mencionadas no capítulo introdutório, foi necessário levar em consideração a continuidade do projeto após a finalização deste trabalho. Portanto devem ter grande aceitação do público que as usam, possuir vasto material de estudo e serem gratuitas, este último critério é adotado pois o Sistema está sendo desenvolvido dentro da UNIOESTE, que não possui licença de tecnologias proprietárias que poderiam ser usadas neste projeto.

A escolha do HTML5 deu-se por ele se encaixar em todos os critérios descritos acima além dele possibilitar que o Sistema web possua responsividade, que é a adaptação em diversos dispositivos, independente de sua resolução de tela [11]. É uma linguagem baseada no conceito de hypertexto, conjunto de elementos ligados por conexões, e entendida por diversos meios de acesso. Ela não depende de navegadores, plataformas ou outros meios, basta construir seu código HTML5 e rodar em qualquer navegador. Como ela pode ser lida por diversos meios evitou-se que a Web fosse desenvolvida em uma base proprietária, com formatos incompatíveis e limitada [11]. É ilustrado na Figura 2.1 o protótipo de cadastro de calha do SIGVERTE desenvoldido com HTML5.

Para deixar as páginas escritas em hypertexto com uma interface mais amigável, é usado o CSS - Cascading Style Sheets [12]. Seu principal benefício é padronizar o estilo das páginas de sua aplicação, pois basta que chame o arquivo de estilo dentro da escrita em hypertexto e a página estará estilizada. No caso do HTML5, a sintaxe de importação do estilo é apresentada como exemplo na Figura 2.2, onde é usado o arquivo "exemplo.css".

Páginas de hypertexto com estilos informam o leitor sobre um conteúdo definido dentro do código, e para que estas páginas tenham alguma funcionalidade é necessária uma linguagem

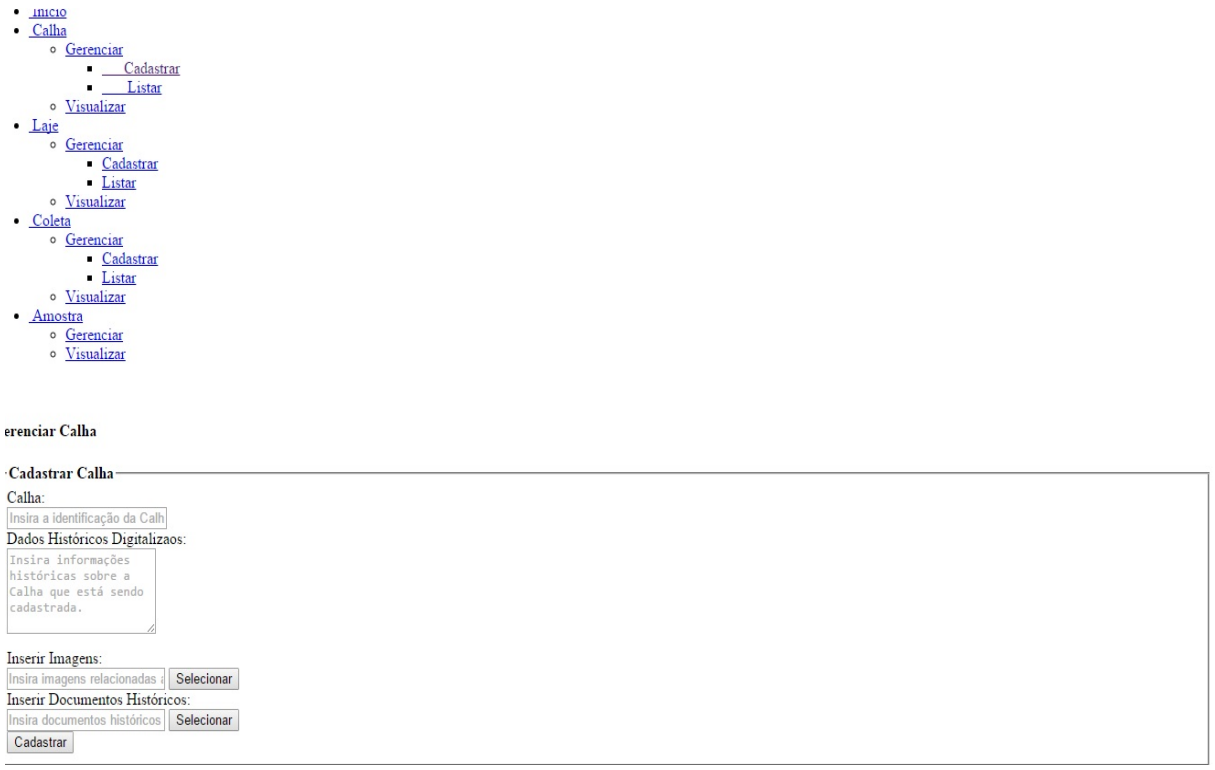

Figura 2.1: Página gerada a partir da linguagem de hypertexto HTML5.

de programação, que neste trabalho de conclusão de curso será o JavaScript, através das quais são implementadas funções de interação com o usuário. O JavaScript é uma linguagem de programação interpretada onde o código fonte é executado por um programa de computador chamado interpretador, que em seguida é executado pelo sistema operacional ou processador [10]. Atendendo um dos requisitos do sistema, a biblioteca para JavaScript OpenLayers será usada para manipulação dos dados georreferenciados. Ela é gratuita e sua documentação é fornecida em seu website [15]. Os dados que foram manipulados com o JavaScript a partir do OpenLayers estão no formato GeoJSON, onde propriedades de geoprocessamento são armazenados em um array na sintaxe do JavaScript, como se pode ver na Figura 2.3. Para manipular estes dados, basta percorrer este array atrás do dado desejado.

Para auxiliar no desenvolvimento de tais páginas e funções o framework Bootstrap foi escolhido, pois com ele é possível chamar classes já implementadas ou manipular as mesmas de acordo com a necessidade. É o framework mais popular para desenvolvimento web responsivo [13]. Dentre seu grande acervo de componentes, o que mais será usado neste trabalho de

```
<!DOCTYPE html>
<html lang="pt-br">

<head>

    <title>SIGVerte</title>

    <!-- Exemplo de Importação de arquivo de estilo -->
    <link href="css/exemplo.css" rel="stylesheet">


</head>

<body>

    <Código HTML5/>

</body>

</html>
```

Figura 2.2: Exemplo de Importação de Arquivo de Estilo.

```
[{ "type": "Feature", "properties": { "mslink": 57952,
"cadastro": 100006000.000000, "loteamento": 1, "quadra": "0001", "lote": "0006",
"logradouro": "PARANA - CASCAVEL", "numero": 2999,
"bairro": "CENTRO", "utilizacao": "Comercial" },
"geometry": { "type": "Polygon", "coordinates": [ [ [ -53.459316863680364, -24.95453767119508 ],
 [ -53.459301860119574, -24.954285479265042 ], [ -53.459104259790941, -24.954295168183318 ],
 [ -53.459119262953429, -24.954547360163861 ], [ -53.459316863680364, -24.95453767119508 ] ] ] } }];
```

Figura 2.3: Formato de Dados em GeoJSON.

conclusão de curso será o sistema de grid, mostrado na Figura 2.4

O sistema de grid é utilizado para dividir a página escrita em HTML5 em colunas, sendo assim, basta que se chame a classe do tamanho do grid desejado para realizar inserções de componentes naquela divisão. Na Figura 2.5 é apresentado o código que gerou a página da Figura 2.4

Depois da escolha das tecnologias para implementar o que é chamado de lado do cliente, é necessário definir como será implementado o lado do servidor, ou seja, onde são manipulados e armazenados os dados. Essa escolha é baseada nos requisitos do Sistema, no que o Sistema terá que fazer e no conhecimento da equipe, o que impactará no restante do desenvolvimento. Pensando nisso, a linguagem Java foi adotada para este projeto. Além dos critérios adotados

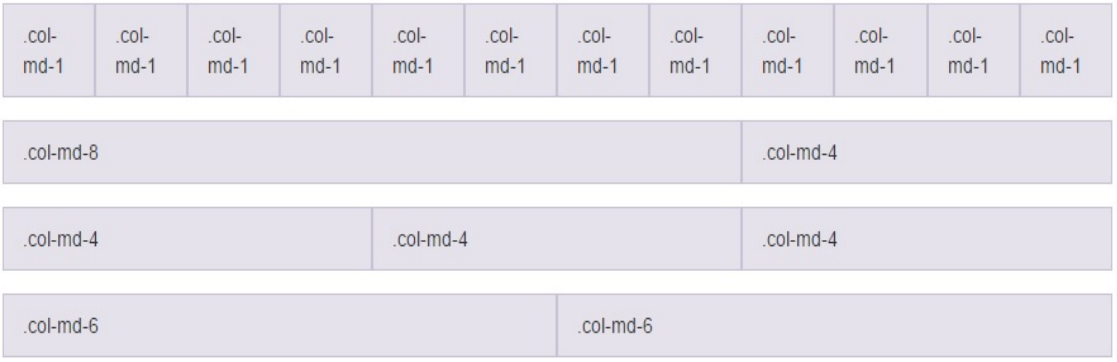

Figura 2.4: Sistema de Grid para HTML5.

para escolha, pesaram também o fato da linguagem ser multiplataforma, ou seja, um sistema implementado em Java pode ser executado em qualquer sistema operacional. Isso se dá pelo fato dela possuir um interpretador próprio que deve ser instalado antes do sistema ser rodado [9]. Para que atuais ou novos integrantes do projeto possam ler o código escrito para o desenvolvimento do sistema, ela foi escolhida por ser a linguagem utilizada para ensino em nossa instituição.

Após tecnologias para interface, interação com o usuário e manipulação dos dados, é necessário armazenar o resultado de tais funções ou guardar dados para serem manipulados. Para isso é utilizado um Sistema de Gerenciamento de Banco de Dados, onde é possível realizar operações de definição e manipulação dos dados, tais operações são realizadas com o SQL -Linguagem de Consulta Estruturada [6]. O SGBD definido para este projeto é o PostGreSQL, por se tratar de uma tecnologia gratuita e por ser objeto-relacional que combina conceitos da programação orientada a objetos e estruturas utilizadas no modelo relacional [6].

E, para que essas tecnologias possam atuar juntas, usamos o PlayFramework, um framework criado para aumentar a produtividade, pois várias funcionalidades que o desenvolvedor teria que fazer manualmente, como a exemplo funções que se relacionem com o banco de dados e dividir o sistema por camadas e mapear funções, o próprio framework implementa [14]. Para criar uma aplicação “Play“ basta que o plugin do “framework“ esteja instalado na IDE , Ambiente de Desenvolvimento Integrado, com o projeto “Play“ criado que automaticamente ele divide o projeto em arquivos ou pastas responsáveis por alguma função. Como por exemplo no arquivos

```
<div class="row">
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
    <div class="col-md-1">.col-md-1</div>
</div>
<div class="row">
    <div class="col-md-8">.col-md-8</div>
    <div class="col-md-4">.col-md-4</div>
</div>
<div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4">.col-md-4</div>
</div>
<div class="row">
    <div class="col-md-6">.col-md-6</div>
    <div class="col-md-6">.col-md-6</div>
</div>
```

Figura 2.5: Código que Mapeia o Sistema de Grid para HTML5.

“routes“ na pasta “conf“, é onde devem ser mapeadas as rotas do sistema, seria o que acontece quando um usuário acessa determinada página ou envia determinado valor. Na pasta “app“ está a divisão Modelo, Visão e Controlador por padrão, onde são armazenados os arquivos relacionados à “interface“, classes e funcionalidades do sistema respectivamente. Na Figura 2.6 estão apresentados os arquivos base para um projeto do PlayFramework e na Figura 2.7 está ilustrado o funcionamento das tecnologias em conjunto, como se portam como “client-side“ e “server-side“ e como o PlayFramework as suporta.

```
.
|-- app                              (arquivos Java da aplicação)
|   |-- controllers
|   |   `-- Application.java
|   `-- views
|       |-- index.scala.html
|       `-- main.scala.html
|-- build.sbt
|-- conf                             (arquivos de configuração)
|   |-- application.conf
|   `-- routes
|-- project
|   |-- build.properties
|   `-- plugins.sbt
|-- public                           (arquivos estáticos)
|   |-- images
|   |   `-- favicon.png
|   |-- javascripts
|   |   `-- jquery-1.9.0.min.js
|   `-- stylesheets
|       `-- main.css
|-- README
`-- test                             (arquivos para testes)
    |-- ApplicationTest.java
    `-- IntegrationTest.java
```

Figura 2.6: Formato Projeto PlayFramework.

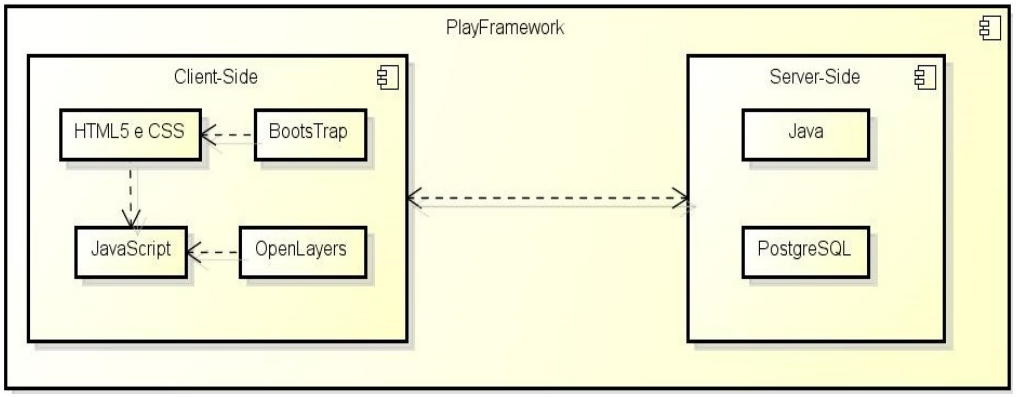

Figura 2.7: Diagrama ilustrando o funcionamento das tecnologias em conjunto.

Capítulo 3

Aspecto Metodológico e Padrões de Projeto

O FDD “Feature Driven Development“ é uma metodologia concebida originalmente por Jeff de Luca, surgiu em Singapura entre os anos de 1997 e 1999 [3]. Ela busca o desenvolvimento por “feature“ ou funcionalidade, ou seja, por requisito funcional do Sistema. É uma metodologia prática para projetos iniciais ou com codificação já existente. O FDD mostra que a soma das partes é maior que o todo. Assim, apesar de criar um modelo abrangente baseado no todo que será desenvolvido, esta fase inicia-se com o detalhamento do domínio do negócio com a divisão de áreas que serão modeladas. O modelo só estará pronto quando a equipe toda estiver de acordo e o planejamento é feito baseado na lista de funcionalidades. Tal metodologia é constituída de cinco etapas que estão detalhadas neste capítulo.

A Figura 3.1 apresenta os cinco processos básicos do FDD sistematizando o que será detalhado neste capítulo.

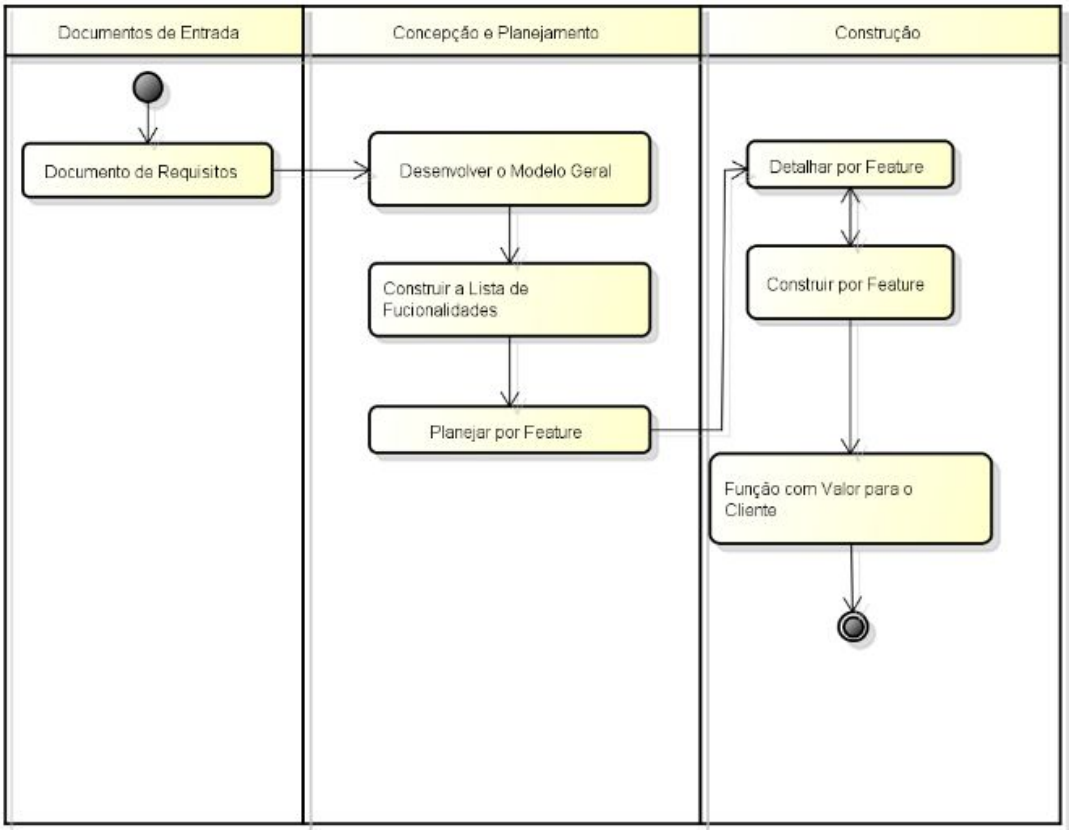

Figura 3.1: Processos do FDD.

3.1 Descrição dos Processos do FDD

Os 5 processos do FDD são bem divididos e voltados para entrega de um produto, funcionalidades desenvolvidas a cada 15 dias, que tem valor para o cliente. Esses processos constituem 2 etapas importantes para a metodologia, a primeira onde é feito o planejamento das funcionalidades que serão desenvolvidas e a segunda a construção das mesmas [19]. Segue abaixo a descrição detalhada de cada processo.

3.1.1 Desenvolver um Modelo Abrangente

Nesta atividade inicial, são selecionados os documentos necessários para o estudo do seu domínio de negócio e é realizada uma análise do escopo do sistema e seu contexto. Após os membros da equipe de modelagem terem conhecimento do domínio, o modelo geral é desenvolvido. Depois é criada uma modelagem superficial para cada área do domínio existente.

Cada um desses modelos é revisado por um grupo aleatório de membros do projeto e melhorias são discutidas. É escolhido o modelo que representa melhor o domínio, ou então é desenvolvido um modelo que é a junção de mais de um domínio projetado. Finalmente, esses modelos são fundidos para gerar um modelo que representa o domínio principal do sistema [20].

Atividades:

O Desenvolvimento do Modelo Abrangente possui como entrada a seleção dos especialistas no domínio do negócio, os programadores líderes e o arquiteto líder e é composto pelas atividades descritas a seguir [20]:

- Formar a Equipe de Modelagem, tal equipe deve ser composta por especialistas do domínio do negócio e membros da equipe de desenvolvimento.
- Estudo Dirigido Sobre o Domínio, onde um especiliasta do domínio do negócio apresenta uma visão geral do domínio que será modelada [20]. Nessa apresentação deve ser incluído informações que estão relacionadas ao domínio de negócio, mas não necessariamente á implementação.
- Estudar a Documentação, a Equipe de Modelagem estuda os documentos que deram entrada ao processo. Esta atividade é feita em caso de dúvidas relacionadas ao modelo de negócio que está sendo modelado e pode ser agendada a qualquer momento conforme necessidade.
- Desenvolver o Modelo, essa atividade é realizada por toda Equipe de Modelagem, onde ela é dividida em pequenos grupos que desenvolverão um modelo que suporte a área do domínio [20]. Assim que todos os grupos terminem seus modelos, a Equipe escolhe o melhor modelo ou compõe um modelo pela combinação dos melhores modelos.
- Refinar o Modelo de Objetos Abrangente, o modelo escolhido na atividade acima pode sofrer atualizações decorrentes de novas informações sobre o domínio de negócio, ou algum detalhe que não foi representado.
- Avaliação Interna e Externa, consiste em validar o modelo gerado nas atividades anteriores junto dos especialistas do modelo de negócio e também pode ser validado com

usuários do futuro Sistema.

3.1.2 Construir a Lista de Funcionalidades

Neste processo é realizada a construção da lista de funcionalidades baseado nos requisitos do Sistema.

Atividades:

A entrada para as atividades são especialistas no domínio de negócio, programadores líderes e arquivo líder. Este processo é composto pelas atividades que se seguem [20].

- Formar a Equipe da Lista de Funcionalidades, a equipe deve ser composta por membros da Equipe de Modelagem e de desenvolvedores.
- Construir a Lista de Funcionalidades, nesta atividade são identificadas as funcionalidades do Sistema baseado no modelo do domínio de negócio desenvolvido anteriormente. As funcionalidades são funções granulares, expressas em termos que possuem valor para o cliente, usando o seguinte modelo de nomeação <ação> <resultado> <objeto> [19]. As funcionalidades devem ser descritas considerando que seu desenvolvimento não levará mais que 15 dias, caso isso ocorra, ela deve ser dividida em duas.
- Avaliação Interna e Externa, realizada por membros da equipe da lista de funcionalidades e especialistas do modelo de negócio, onde a lista de funcionalidades é validada.

3.1.3 Planejar por Funcionalidade

Neste processo é feito o planejamento das funcionalidades listadas anteriormente. Onde é definida as entregas das funcionalidades, quais terão prioridade de desenvolvimento e serão entregues antes. Essa prioridade pode ser dada pelo cliente, que pode definir quais funcionalidades ele quer que sejam entregues primeiro ou também pode ser definida pela equipe de desenvolvimento, onde vê a necessidade de que algumas funcionalidades devem ser desenvoldidas antes que as outras.

Atividades:

A entrada do processo é a Lista de Funcionalidades produzida no processo anterior. E é composto pelas seguintes atividades [20]:

- Formar Equipe de Planejamento, que deve ser formada pelos programadores líderes e pelo gestor de desenvolvimento.
- Determinar Sequência de Desenvolvimento, a equipe de planejamento determina as datas que as funcionalidades devem ser entregues.
- Atribuir Atividades de Negócio aos Programadores Líderes, onde programadores líderes são associados á atividades de negócio, que abrangem algumas funcionalidades.
- Atribuir Classes aos Desenvolvedores, onde as classes são atribuídas aos membros da equipe de desenvolvimento.

3.1.4 Detalhar por Funcionalidade

Este processo é repetido até que todas funcionalidades tenham sido implementadas. Uma funcionalidade é atribuída a um programador líder que seleciona os desenvolvedores que estão associados as classes pertencentes á funcionalidade. As atividades abaixo compõe esse processo:

Atividades:

- Formar Equipe de Funcionalidades, realizada pelo Programador Líder, o programador líder identifica os desenvolvedores na lista de proprietários baseado nas funcionalidades que lhe foram atribuídas e forma a Equipe de Funcionalidades [20].
- Estudo Dirigido do Domínio, realizada pelo Especialista do Domínio, dependendo da complexidade da funcionalidade, a equipe de desenvolvimento pode fazer a demanda de uma apresentação do domínio da funcionalidade que está sendo desenvolvida realizada por um especialista do domínio de negócio.
- Estudar a Documentação de Referência, com a funcionalidade a ser projetada selecionada, a equipe deve estudar o materiais referentes a ela, tais como desenhos de tela, requisitos,

especificações de interface com sistemas externos e qualquer outra documentação de suporte. Também é opcional dependendo da complexidade da funcionalidade.

- Desenvolver os Diagramas de Sequência, desenvolvê-los para que a funcionalidade seja projetada.

- Refinar o Modelo de Objetos, realizada pelo Programador Líder, onde o trabalho deve ser compartilhado e estar visível pelos membros da equipe de funcionalidades, mas invisível para o resto do projeto. O programador líder refina o modelo adicionando novas classes, métodos, atributos baseado nos diagramas de sequência definidos para as funcionalidades. Então, o programador líder cria diagramas de modelo num formato publicável, que devem ser submetidos na área do projeto [20].

- Escrever Prefácios de Classes e Métodos, os desenvolvedores devem escrever prefácios para suas classes e métodos, detalhando parâmetros e entrada e saídas geradas pelas mesmas.

- Inspeção do Projeto, realizada pela Equipe de Funcionalidades, sendo obrigatória: É feita uma inspeção com os membros da Equipe de Funcionalidades. Após o aceite é gerada uma lista de tarefas para cada classe afetada [20].

3.1.5 Construir por Funcionalidade

É um processo para cada funcionalidade. Os desenvolvedores proprietários de classes implementam os itens necessários para que suas classes suportem o projeto para esta funcionalidade. O código então passa por inspeção e testes, e então é promovido para a versão atual [20].

Atividades:

- Implementar Classes e Métodos, os desenvolvedores devem implementar as classes onde são proprietários, satisfazendo os requisitos das mesmas.

- Inspecionar o Código, a inspeção do código deve ser feita pela equipe de funcionalidades antes ou depois do Teste de Unidade, que testará a funcionalidade implementada.

- Teste de Unidade, os desenvolvedores proprietários devem realizar o Teste de Unidade em suas classes e métodos para se certificarem que os mesmos estão satisfazendo os requisitos.

- Promover a Versão Atual, o programador chefe monitora as classes sendo promovidas individualmente, através de informações dos desenvolvedores.

3.2 Padrões de Projeto

Padrão de projeto é descrição ou modelo de como resolver um problema que pode ser usado em muitas situações diferentes. São melhores práticas formais que o programador pode usar para resolver problemas comuns quando projetar uma aplicação ou Sistema [2].

Os Padrões de Projeto escolhidos para este trabalho de conclusão de curso totalizam dois e são:

- Observer: Os observers registram-se no subject para receberem atualizações quando os dados do subject são alterados. Os observers também podem cancelar o seu registro e dessa forma não receber mais nenhuma atualização do subject.

- DAO: Todas as regras do mecanismo de persistência passam a ser medidas por um objeto do tipo DAO. Toda a lógica de acesso e execução do banco de dados é colocada dentro de um objeto do tipo DAO e, desta forma, se cria um isolamento entre a aplicação de persistência e as demais partes do sistema. As operações CRUD - Criar, Recuperar, Atualizar, e Excluir passam então a ser de inteira responsabilidade do objeto DAO, isolando-as do resto da aplicação.

Capítulo 4

Requisitos e Prototipação do Sistema

Requisito é uma condição necessária para alcançar certo objetivo ou obtenção de certo fim [21]. Os requisitos são a descrição das funções e restrições que o software deve possuir, apresentados de uma forma detalhada, normalmente classificados como funcionais e não funcionais. Os requisitos funcionais estão diretamente ligados a como o Sistema deve se comportar em determinadas situações. Os requisitos não funcionais podem estar relacionados às propriedades do sistema como confiabilidade, tempo de resposta, restrições sobre o processo ou padrões.

Neste capítulo são apresentados os requisitos funcionais e não funcionais do Sistema deste trabalho de conclusão de curso.

4.1 Primeiro Módulo e Seus Requisitos

O primeiro módulo do Sistema que é contemplado neste trabalho de conclusão de curso dispõe de funcionalidades que consistem em um Sistema de Informação, que é um software que abrange métodos organizados para coletar, processar, transmitir e disseminar dados que representam informação. Este sistema possui diversas funcionalidades que foram transcritas para seu documento de requisitos que contempla requisitos funcionais, não funcionais e protótipos de tela. Estes requisitos essenciais para o funcionamento do Sistema são apresentados a seguir.

Gerenciar Calha: Deve ser possível ao usuário do Sistema gerenciar as calhas do vertedouro, viabilizando cadastro, alteração e exclusão das calhas. As calhas cadastradas devem possuir um “shapefile“, arquivo que representa um objeto ou conjunto de objetos usado neste contexto

para georreferenciamento, tal arquivo deve ser inserido no sistema pelo técnico responsável. A funcionalidade Gerenciar Calha está dividida em duas funções, são elas:

- Cadastrar Calha, onde é viabilizado a inserção de sua identificação, seus dados históricos e imagens referentes à mesma. Na Figura 4.1 é apresentado seu respectivo protótipo.

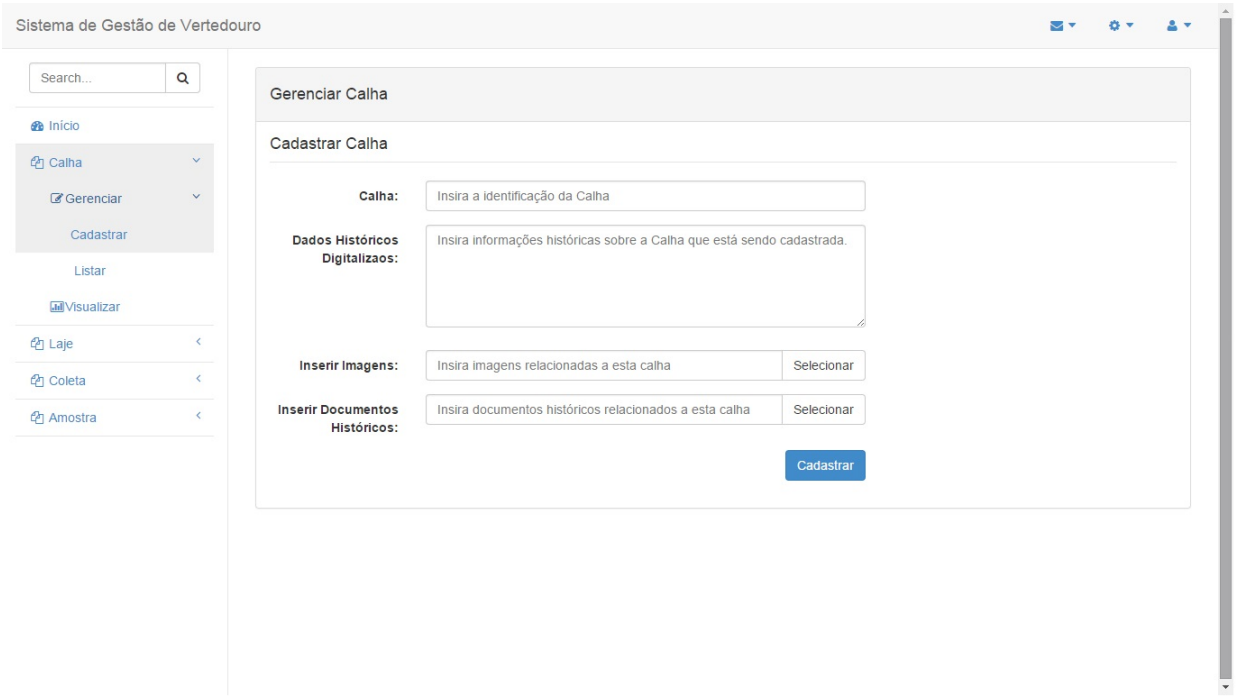

Figura 4.1: Protótipo de Cadasto de Calha

- Listar Calhas, onde as calhas cadastradas são listadas, através desta lista o usuário pode selecionar uma calha e alterar seus dados ou excluí-la. Tanto para edição quanto para exclusão, deve ser dada uma justificativa, guardar a data da alteração ou exclusão e o usuário que a fez.

Gerenciar Laje: O Sistema deve prover o gerenciamento das lajes, possibilitando cadastro, alteração ou exclusão delas. Cada laje está representada em forma de um polígono dentro do "shapefile" da calha. Tal funcionalidade também é dividida em três outras funções:

- Cadastrar Laje, o Sistema deverá dispor do gerenciamento das lajes que compões a calha esquerda do vertedouro. Os atributos essenciais para o formulário de casastro de laje são

identificação da calha que a laje pertence, identificação da laje e inserção de imagens ou documentos associadas a mesma. Na Figura 4.2 é apresentado seu respectivo protótipo.

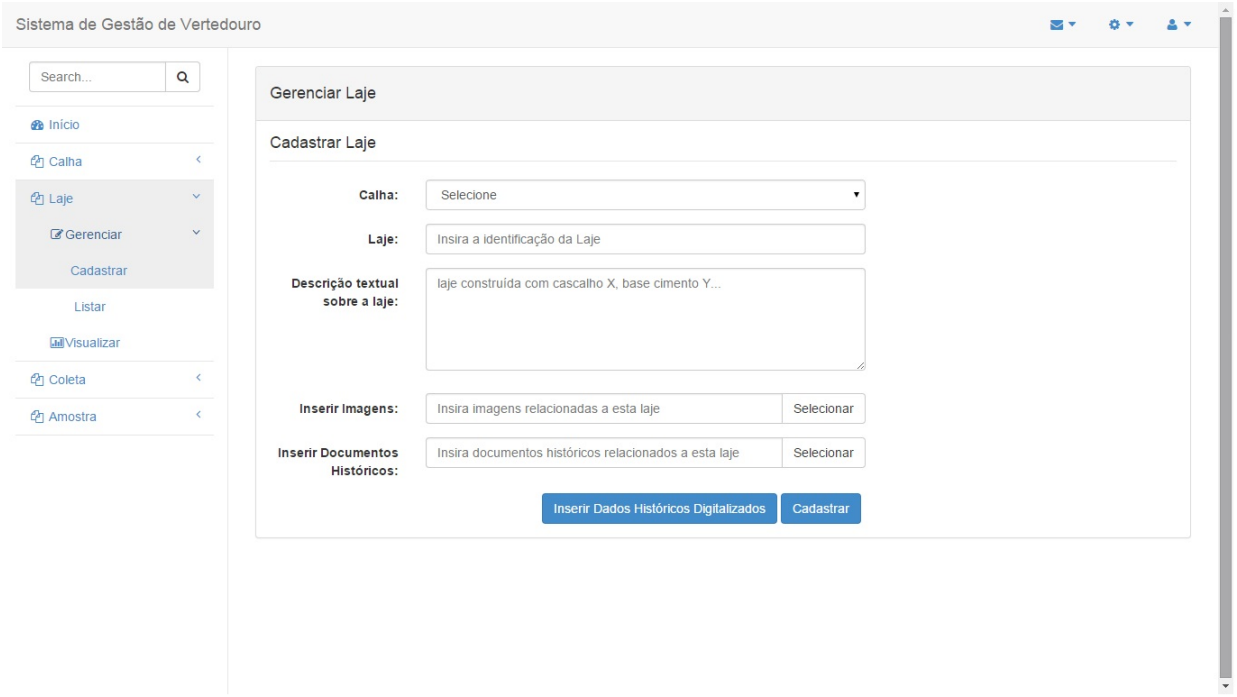

Figura 4.2: Protótipo de Cadastro de Laje.

- Frentes de Concretagem, nessa função deve-se disponibilizar para o cadastro de uma laje, a inserção de seus dados históricos digitalizados, que nada mais são que as frentes de concretagem. Um formulário que pode variar conforme o vertedouro, como para este primeiro Sistema está sendo usada na Itaipu como principal interessado, tal cadastro foi baseado em seus formulários. Na Figura 4.3 é apresentado seu respectivo protótipo.

- Listar Lajes, e assim como a funcionalidade Listar Calhas, é necessária a listagem das lajes existentes, onde é possível para o usuário realizar a visualização das lajes cadastradas no Sistema, podendo ser filtrado por calha que as lajes listadas pertencem. O usuário pode selecionar a laje para realizar alguma alteração em seus dados cadastrais ou simplesmente excluí-la do Sistema, para ambas ações o usuário deve justificar uma causa, junto da jusiticativa o Sistema deve guardar sua data e o usuário que a fez.

Gerenciar Coleta: Este requisito ainda deve ser validado, provável que seja dividido entre Gerenciar Atividade e Gerenciar Campanha. Nele temos o Cadastro e Listagem das Coletas.

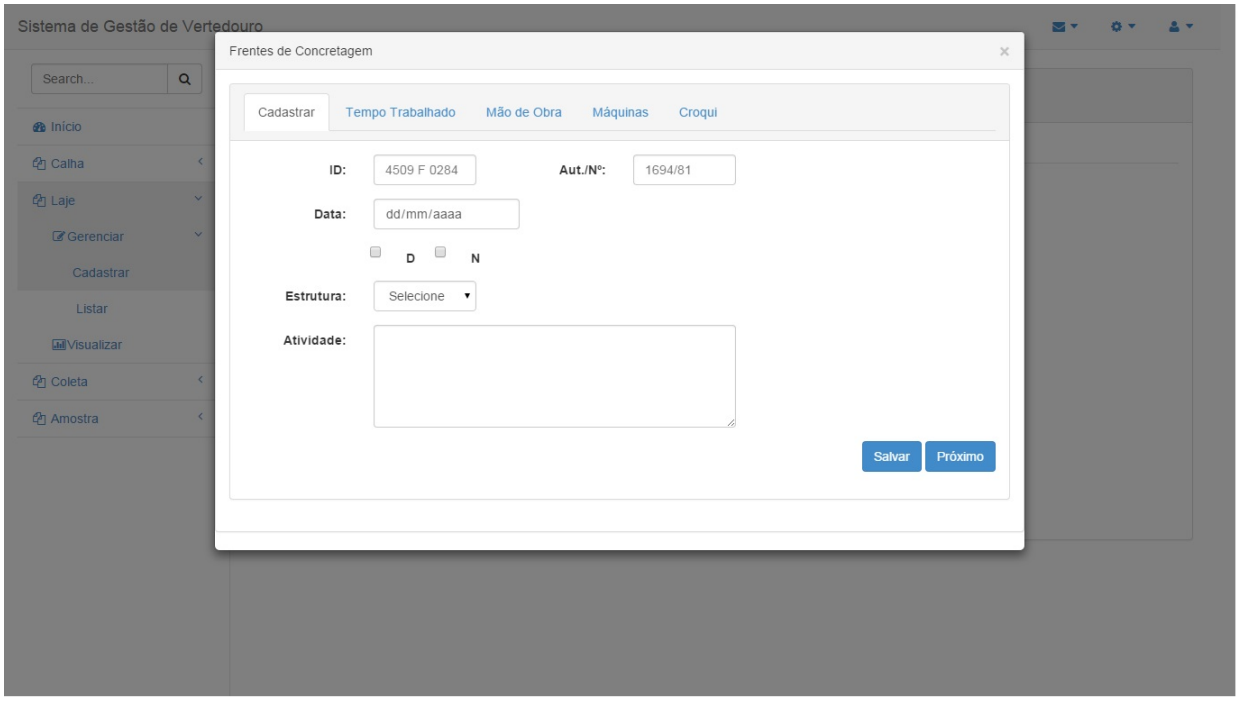

Figura 4.3: Protótipo de Cadastro de Frente de Concretagem.

- Cadastrar Coleta, nesta funcionalidade é possível o gerenciamento de coletas realizadas. O formulário para cadastro de uma coleta deve possuir qual o tipo da coleta, campanha ou atividade, identificação da coleta, data de realização, descrição, calha que pertence e seleção das lajes pertencentes a esta coleta. A Figura 4.4 apresenta o protótipo desta funcionalidade.

- Listar Coletas, onde é possível ao usuário listar as coletas cadastradas, podendo ser filtradas por calha e laje que pertencem, também por data da coleta e se é atividade ou campanha. O usuário poderá realizar uma alteração ou exclusão, ambas ações ao serem finalizadas devem exigir justificativa, guardar data de alteração ou exclusão e o usuário que a fez.

Gerenciar Amostra: Esta funcionalidade, assim como as demais funções de gerenciamento do Sistema, é dividida entre cadastro e listagem de amostras.

- Cadastrar Amostra, onde deve ser possível cadastrar amostras nas coletadas existentes. Os atributos necessários para o cadastro de uma amostra são: Identificação da Calha de

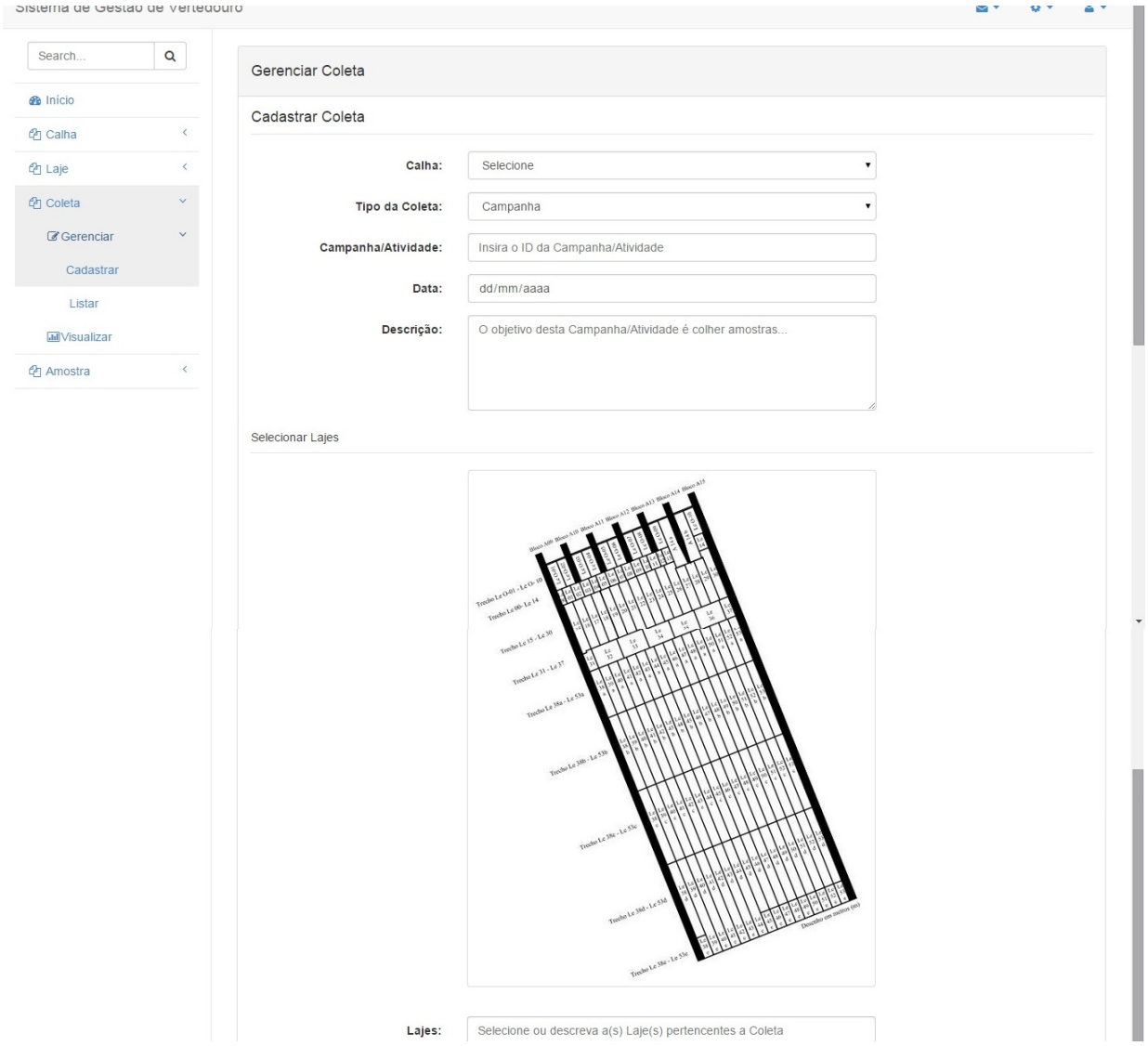

Figura 4.4: Protótipo de Cadastro de Coleta.

onde a amostra foi retirada, identificação da Laje, identifação da Coleta e identificação da amostra. Data de cadastro da amostra no Sistema, data de realização da coleta e seleção da coordenada no mapa referente à localização da amostra. A Figura 4.5 apresenta o protótipo desta funcionalidade.

- Listar Amostras, onde deve estar disponível para o usuário do sistema a listagem das amostras cadastradas. Os itens que estão contidos na lista para amostra são: Calha, Laje e Coleta que a amostra pertence e data de coleta. É possível filtrar as amostras que aperecem na consulta por qualquer item desses. Para cada amostra também é possível realizar sua

edição ou exclusão, assim como nas demais listagens, assim que um usuário disparar uma edição ou exclusão o Sistema precisará de uma justificativa, guardando também data e usuário que a fez.

Para melhor tomada de decisão em relação às funcões de cadastro acima descritas, pensou-se em uma funcionalidade que desse conta de dar uma visão do todo para o usuário, assim surgiu o requisito descrito abaixo:

Visão Geral: Parte principal para funcionamento deste primeiro módulo. Nesta funcionalidade o usuário pode acessar o vertedouro georreferenciado, ou para este primeiro caso de teste que é a calha esquerda do vertedouro, e visualizar as informações que foram cadastradas nas funcionalidades anteriores. Através de filtros pode acessar um dado específico ou somente ter acesso ao que lhe é importante no momento, como filtrar pela Laje que deseja visualizar, Campanhas ou Atividades realizadas, amostras coletadas e podendo selecionar um período como filtro secundário. A Figura 4.6 apresenta o georreferenciamento da Calha Esquerda do vertedouro de Itaipu.

Gerenciar Usuário: Como se trata de um Sistema acessado via web, é necessário o cadastro de usuários e a existência por divisão de níveis de acesso para tais. Nesta funcionalidade, o Administrador, nível mais alto de acesso, poderá realizar o cadastro de usuários, e caso necessário desativalos. A Figura 4.7 apresenta o protótipo desta funcionalidade.

Os requisitos não funcionais do Sistema estão mais relacionados a como ele é apresentado ao usuário, sua manutenção e segurança, são eles:

- Quanto à Usabilidade: A interface do Sistema será padronizada, intuitiva, com informações e funcionalidades objetivas.
- Quanto à Integridade: O Sistema gestor de dados (PostgreSQL + PostGIS) será o responsável pela integridade.
- Quanto à Disponibilidade: A disponibilidade deve ser mantida por um servidor 24 horas, sendo que o acesso ao Sistema será feito por login e senha.
- Quanto à Responsividade: O Sistema adequa sua interface aos principais navegadores de internet como exemplo Internet Explorer [22], Google Chrome [23] e Firefox [24]. O Sistema deverá ser acessível também através de dispositivos móveis.

- Quanto à Segurança: Quando da implantação do sistema, o técnico responsável deverá realizar as configurações necessárias para a utilização do shapefile correspondente ao vertedouro em questão. Por exemplo no caso de Itaipu essas configurações necessárias garantirão que todas operações serão feitas considerando o shapefile de Itaipu. Isso é válido para o caso de outro vertedouro estar utilizando o SIGVERTE. Trata-se de um requisito de segurança, visto que a definição do shapefile correspondente é a base de todas as operações georreferenciadas realizadas pelo Sistema.

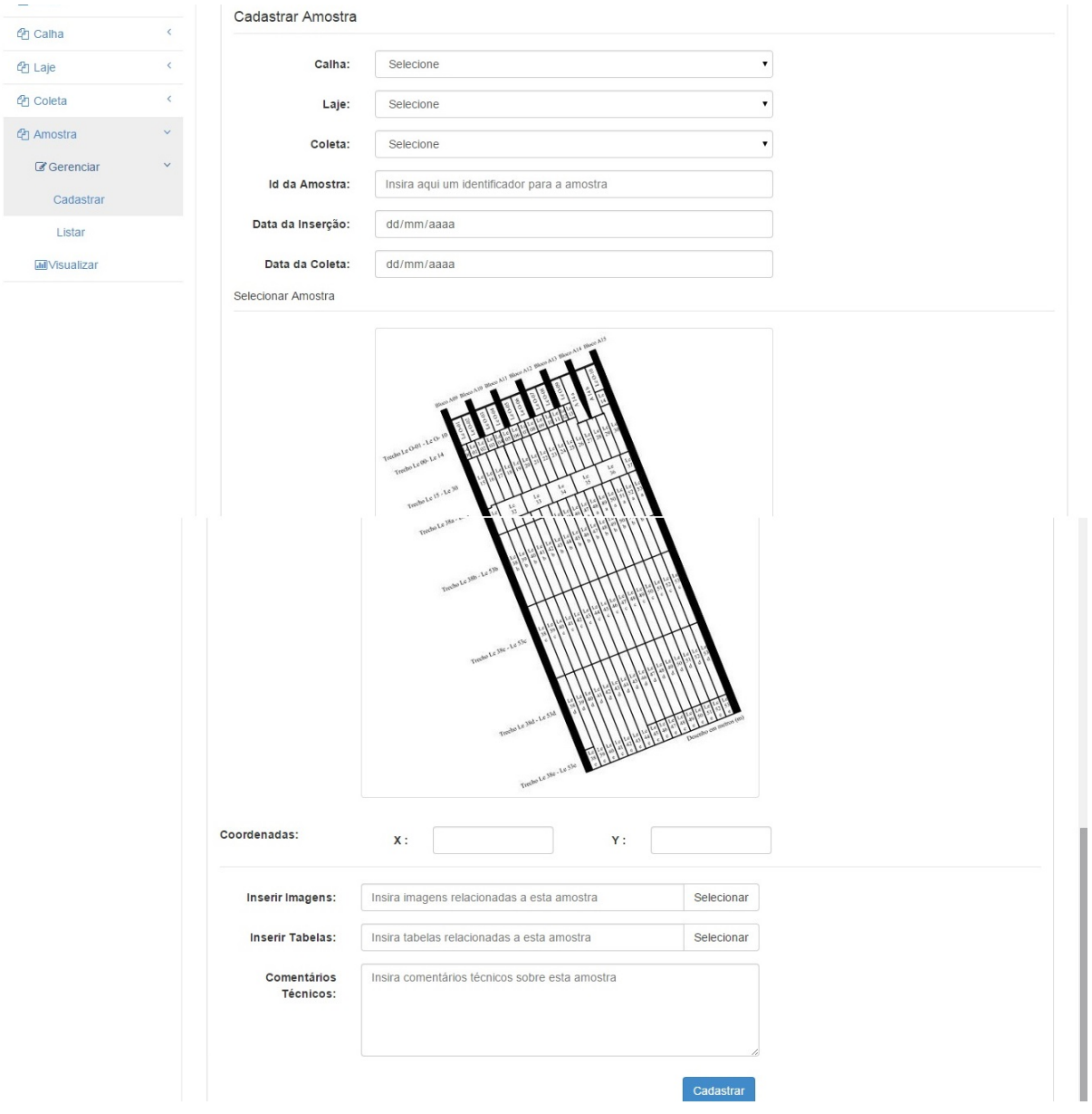

Figura 4.5: Protótipo de Cadastro de Amostra.

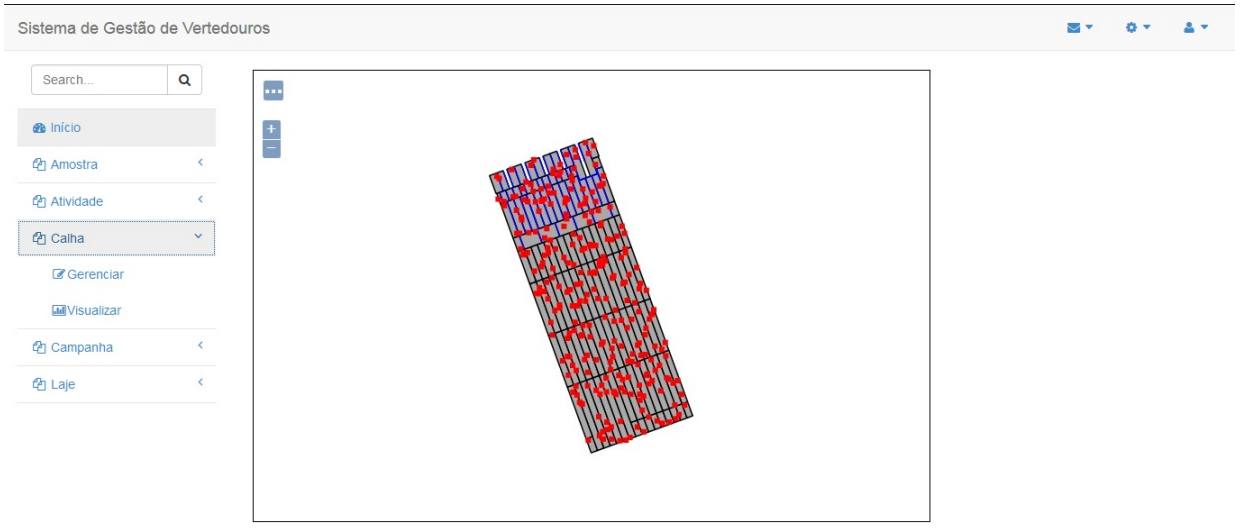

Figura 4.6: Protótipo da Calha Esquerda do Vertedouro de Itaipu Georreferenciada.

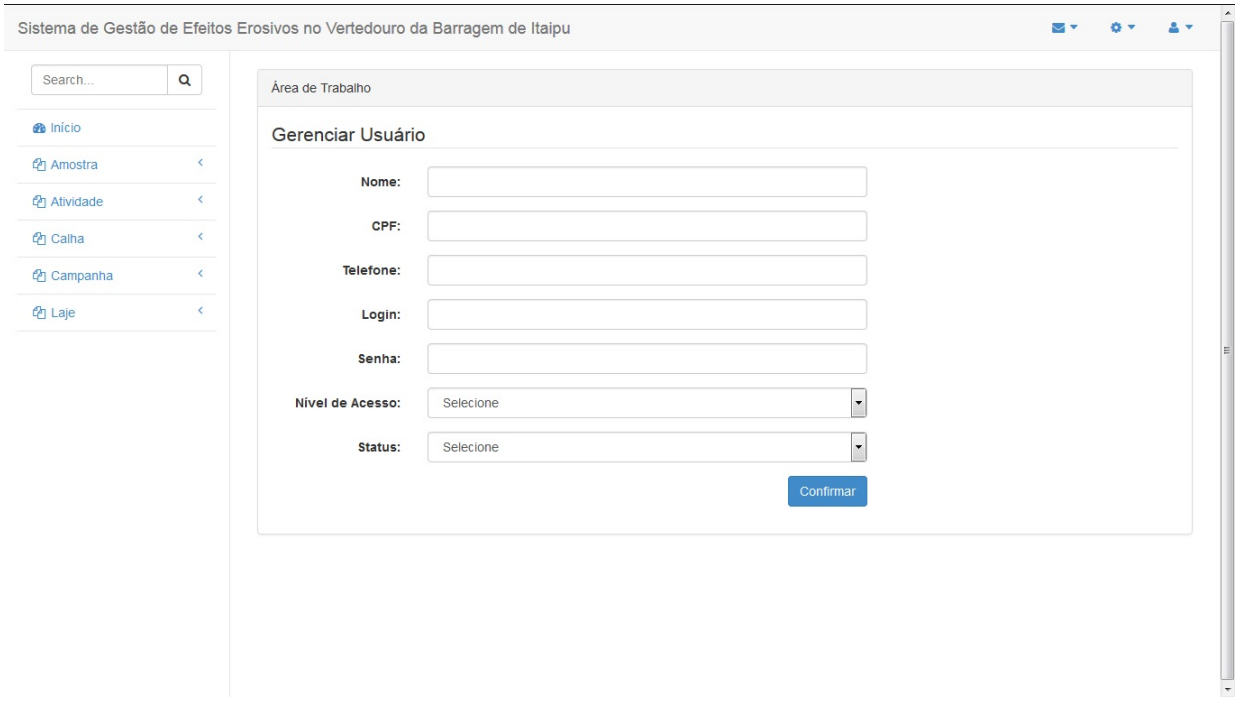

Figura 4.7: Protótipo do Cadastro de Usuário.

Capítulo 5

O Desenvolvimento do SIGVERTE

Embora não seja um pré-requisito, o levantamento de requisitos completo e um protótipo funcional do software pronto, as informações e conhecimentos sobre domínio de negócio são suficientes para se dar início ao desenvolvimento do Sistema utilizando a metodologia ágil FDD. Este capítulo mostra como isso ocorreu e as adaptações necessárias realizadas na metodologia.

5.1 O Modelo Geral do Sistema

O modelo geral do Sistema surgiu a partir do documento de requisitos e serve como entrada para construção da Lista de Funcionalidades. Tal modelo é usado para que qualquer um que o visualize tenha uma concepção de como as funcionalidades do Sistema devem se comportar [20]. Para que fosse construído foi necessária a revisão e atualização do documento de requisitos, foram adicionadas informações que não existiam no documento que deu entrada ao projeto, são elas: Prioridade, um grau que determina a ordem de implementação dos requisitos, e Dependência, onde são descritos os requisitos que este depende. Também foram atualizadas todas as imagens dos requisitos usando como base o protótipo para representá-los. Essas mudanças foram necessárias para dar suporte ao entendimento do modelo de negócio aos membros da equipe. Como todos estavam sempre presentes nas reuniões realizadas na elicitação do documento, cada um possuía conhecimento necessário para opinar no desenvolvimento do modelo geral do Sistema, que posteriormente foi validado.

A primeira etapa na construção do modelo geral é Formar a Equipe de Modelagem [20], e para este Trabalho de Conclusão de Curso cada membro da equipe possuiu papéis que definiram suas responsalidades perante os requisitos do projeto, como ilustrados na Figura 5.1. Cada

membro foi responsável de forma direta ou indireta pela construção do modelo geral, pois para que fosse possível a modelagem do mesmo cada um teve que cumprir com seu papel, que estão apresentados na Figura 5.2. As duas próximas etapas após Formar a Equipe de Modelagem foram o Estudo Dirigido sobre o Domínio e Desenvolver o Modelo [20]. O estudo do domínio foi feito pela Equipe de Modelagem através da revisão e atualização do documento de requisitos, conforme surgissem dúvidas sobre os mesmos uma reunião era agendada com os especialistas do domínio de negócio.

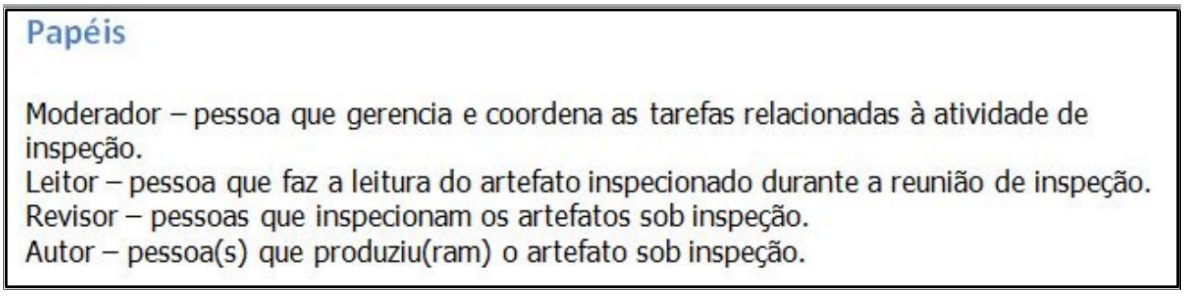

Figura 5.1: Imagem retirada do Relatório de Inspeção dos Requisitos e mostra a função de cada papel.

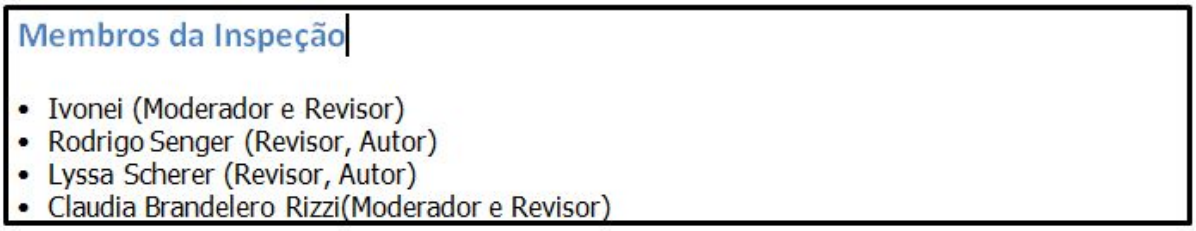

Figura 5.2: Imagem retirada do Relatório de Inspeção dos Requisitos e mostra os papeis de cada membro da equipe.

Com o conceito do modelo de negócio do Sistema concreto para a equipe, iniciou-se o desenvolvimento do modelo geral do Sistema. Como se pode ver na Figura 5.3 ele tem uma aparência parecida com um Diagrama de Classes, pois ele ilustra o comportamento do Sistema baseado nas suas classes e atributos. Também foi usado recurso de anotações para se ter um melhor grau de detalhe. Tal modelo foi validado junto aos especialistas do domínio de negócio e foi modelado usando o software Astah Community [25]

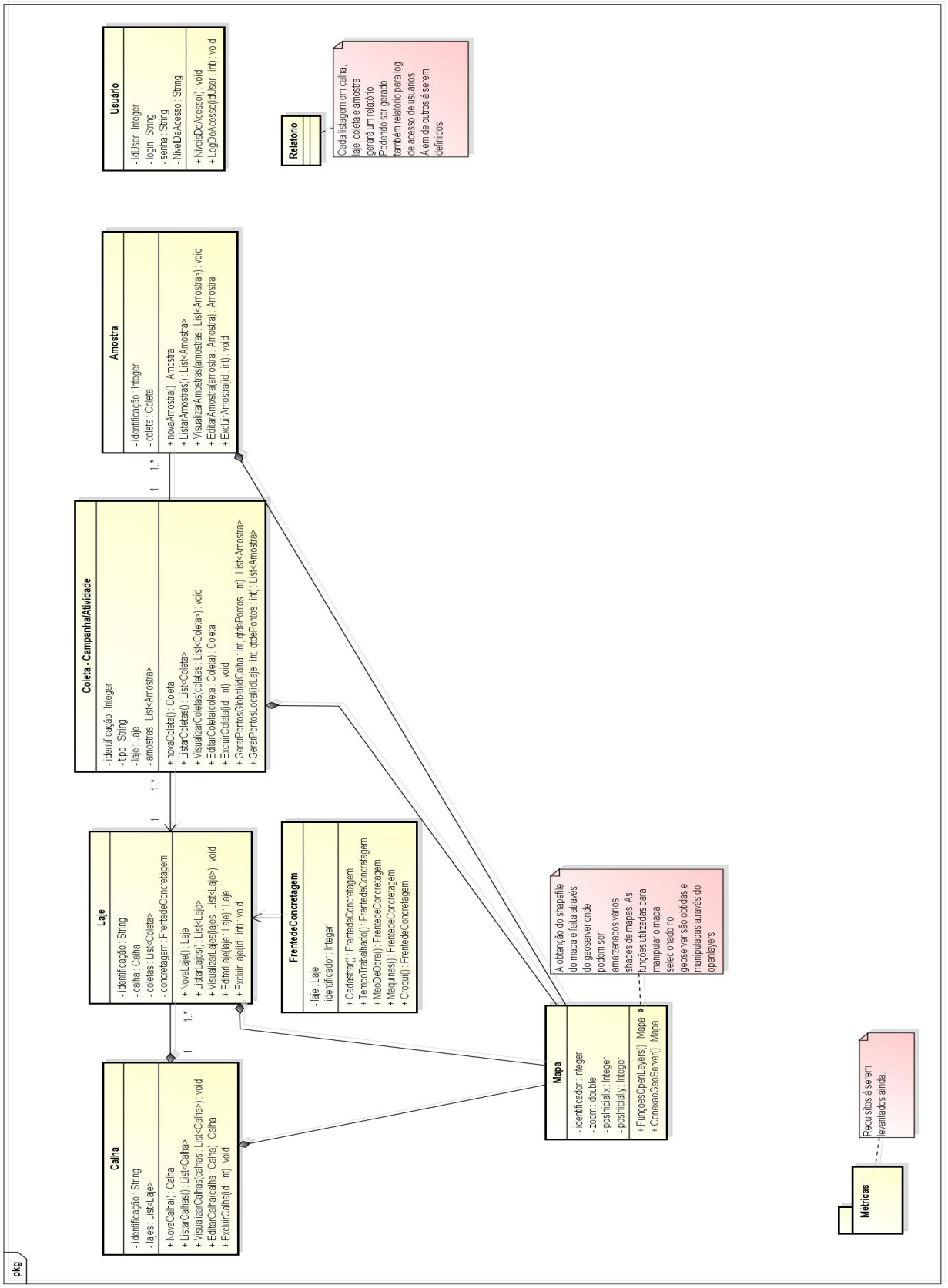

Figura 5.3: Modelo Abrangente do Domínio de Negócio do Sistema.

5.2 A Lista de Funcionalidades

Com o modelo geral pronto e maior conhecimento sobre o domínio de negócio a Lista de Funcionalidades pôde ser construída. Como a metodologia FDD foi utilizada, pois ela suporta os artefatos de entrada deste projeto, a lista foi modelada para suportar tais artefatos, sendo normalmente uma representação que substitui o documento de requisitos [20]. Porém, para este projeto o documento é muito importante para novos integrantes que o comporão futuramente e não pode ser descartado.

5.2.1 A Definição do Modelo da Lista de Funcionalidades

Como já dito no capítulo 3, as funcionalidades devem ser expressas em termos que possuem valor para o cliente, usando como padrão de nomeação o seguinte modelo <ação> <resultado> <objeto> [19]. Na mesma lista ainda deve-se ter as seguintes colunas: id da funcionalidade, o relacionamento dela com objetos do Sistema e a prioridade da mesma para se ter um controle de risco [19], colocando as funcionalidades com menor conhecimento do domínio com maior prioridade, sendo que o proprietário da funcionalidade foi removido dessa lista pois ele é sempre o mesmo neste caso. E especialmente para este trabalho foram adicionados duas novas colunas, uma representando o número do requisito que a funcionalidade pertence e outra com o nome do requisito. Ficando no seguinte formato apresentado na Figura 5.4:

Id	Requisito	Major Feature	Nome	Relacionamento	Prioridade

Figura 5.4: Imagem retirada do Documento de Requisitos e mostra as colunas que devem ser preenchidas para cada funcionalidade.

5.2.2 Construção dos Nomes das Funcionalidades

Os nomes das funcionalidades são escritos com a finalidade de expressar valor para o cliente [3]. Devem aparecer no nome a ação que está sendo realizada, no que resulta esta ação e o objeto envolvido na ação. Como na Figura 5.5, a funcionalidade que tem o id 01 e possui o nome “Cadastrar nova calha na lista de calhas“ onde a ação é “Cadastrar nova calha“, resulta numa calha cadastrada na lista de calhas e o objeto envolvido é a calha.

5.3 Planejamento por Funcionalidade

A etapa de planejamento por funcionalidade é realizada com a finalidade de definir a ordem de desenvolvimento [19]. Neste trabalho de conclusão de curso a ordem é definida através da prioridade que a funcionalidade recebe, podendo ser Alta, Média ou Baixa. Ela pode ser definida baseada em diversos fatores, dependendo do contexto em que o Sistema está sendo desenvolvido, por exemplo, caso o cliente especifique quais funcionalidades devem ser entregues a ele com maior antecedência que outras, as mesmas devem receber maior prioridade [19]. Outra forma como a prioridade para cada funcionalidade pode ser atribuída, que foi usado neste trabalho, é em relação ao conhecimento do domínio em que a mesma será implementada, caso seja necessário um estudo dos documentos envolvidos ou de tecnologias que serão utilizadas, a prioridade da funcionalidade deve ser alta. Portanto, o grau de prioridade é usado para determinar um plano de desenvolvimento, ditando quais funcionalidades devem ser entregues primeiro. A Figura 5.5 ilustra parte da lista de funcionalidades .

Id	Requisito	Major Feature	Nome	Relacionamento	Prioridade
01	RF-01.01	Gerenciar Calha	Cadastrar nova calha na lista de calhas.	-	Alta
02	RF-01.02	Gerenciar Calha	Listar calhas cadastradas.	-	Média
03	RF-01.02	Gerenciar Calha	Excluir calha existente na lista de calhas.	-	Baixa
04	RF-01.02	Gerenciar Calha	Editar calha existente na lista de calhas.	-	Baixa
05	RF-02	Visualizar Calha	Visualizar calha existente através do mapa.	Mapa	Alta
06	RF-03.01	Gerenciar Laje	Cadastrar nova Laje na lista de lajes.	Calha	Alta
07	RF-03.03	Gerenciar Laje	Listar lajes cadastradas filtradas por calha.	Calha	Média
08	RF-03.03	Gerenciar Laje	Excluir laje existente na lista de lajes.	Calha	Baixa
09	RF-03.03	Gerenciar Laje	Editar laje existente na lista de lajes.	Calha	Baixa
10	RF-03.02	Gerenciar Laje	Cadastrar Frentes de Concretagem para cada laje da lista de lajes.	Calha	Alta
11	RF-04	Visualizar Laje	Visualizar laje existente.	Calha	Alta
12	RF-05.01	Gerenciar Campanha	Cadastar nova campanha na lista de campanhas.	Calha, Laje	Alta
13	RF-05.02	Gerenciar Campanha	Listar campanhas cadastradas, filtradas por laje, período realizado ou calha.	Calha, Laje	Média
14	RF-05.03	Gerenciar Campanha	Excluir campanha na lista de campanhas.	Calha, Laje	Baixa
15	RF-05.04	Gerenciar Campanha	Editar campanha na lista de campanhas.	Calha, Laje	Baixa
16	RF-06	Visualizar Campanha	Visualizar campanha existente.	Calha, Laje	Alta
17	RF-07.01	Gerenciar Atividade	Cadastrar atividade de ponto crítico na lista de atividades de pontos críticos.	Calha, Laje	Alta
18	RF-07.02	Gerenciar	Cadastrar atividade de manutenção na lista	Calha, Laje	Alta

Figura 5.5: Imagem retirada do Documento de Requisitos e mostra parte da Lista de Funcionalidades.

5.4 Detalhar e Construir por Funcionalidade.

Para esta etapa da metodologia, o detalhamento de cada funcionalidade foi feito através da construção de seu diagrama de sequência que é usado para representar interações entre os objetos de um cenário [21], e caso necessário, atualização do Diagrama da Arquitetura do Sistema. Após ambos serem validados a implementação da funcionalidade foi feita realizando

um Teste de Unidade para verificação de erros na mesma, que é o processo de testar os componentes de programa, como métodos ou classes de objeto [21]. Ao final de Teste de Unidade, a funcionalidade foi promovida a versão atual e o ciclo de desenvolvimento, detalhar e construir por funcionalidade, era reiniciado até que o Sistema estivesse pronto. Ao final do desenvolvimento, quando todas as funcionalidades foram promovidas á versão atual, foi realizado um Teste de Sistema, que verifica se os componentes são compatíveis, se integram corretamente e transferem os dados certos no momento certo [21].

> *Uma arquitetura de software deve conter: a definição dos elementos de projeto que compõem o software; a descrição das interações entre estes elementos; os padrões de composição dos elementos; e um conjunto de restrições sobre estes padrões. [Shaw, 96]*

A Arquitetura, alguns dos Diagramas de Sequência junto de suas respectivas implementações assim como seus respectivos testes são como seguem: (Detalhes da implementação, imagens do software estão no capítulo 6).

A Arquitetura do Sistema é apresentada na Figura 5.6.

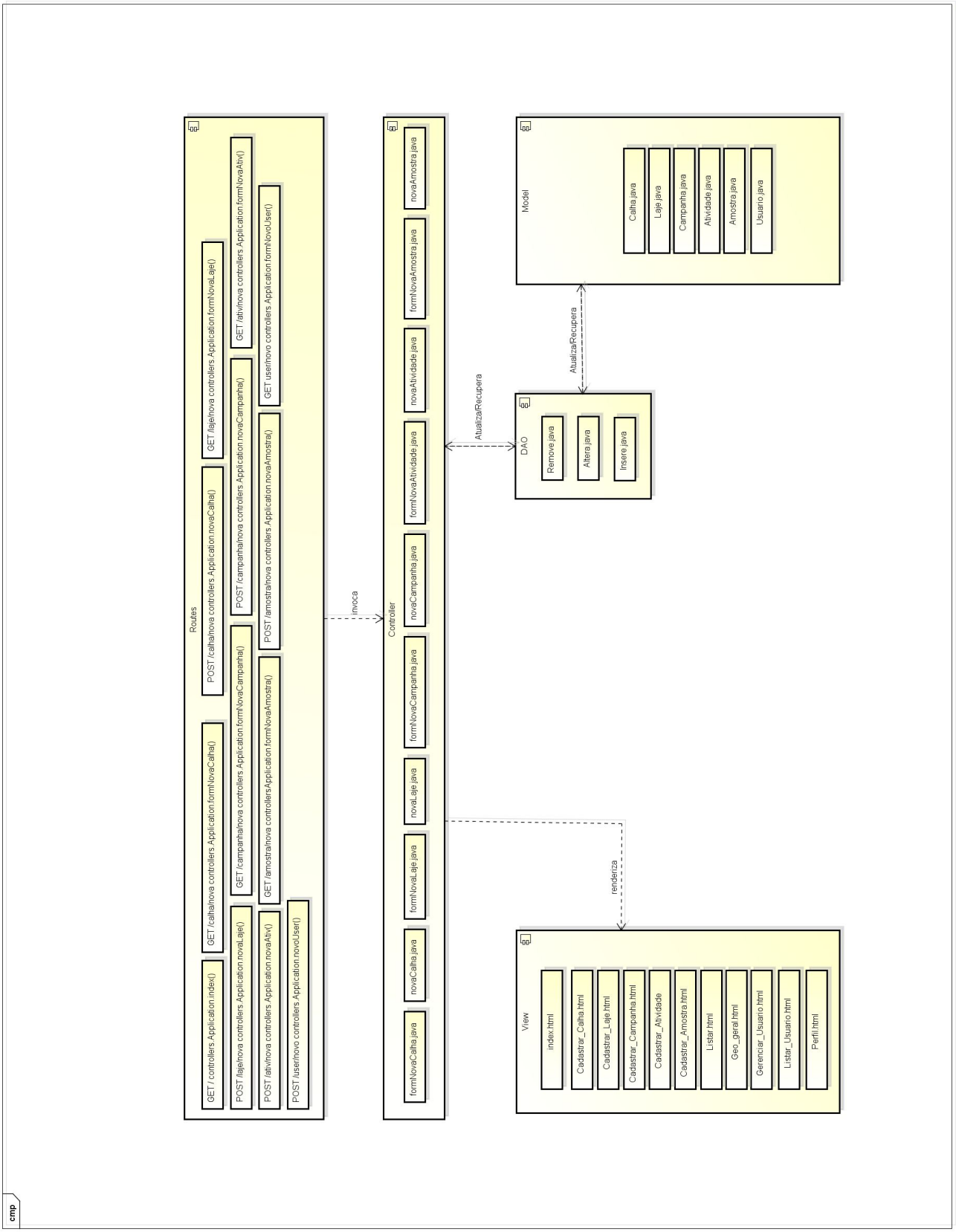

Figura 5.6: Arquitetura do SIGVERTE.

- **Cadastrar Calha:** O cenário modelado no diagrama ilustrado na Figura 6.2 consiste em que o usuário loga no Sistema, sendo direcionado para a tela inicial, o index, clica em Cadastrar Calha atualizando a Área de Trabalho, e preenche os dados do formulário de cadastro da calha que recebe o nome formCalha no controlador. Caso o controlador verifique a existência de algum erro de inserção no formulário uma mensagem de erro é retornada para a Área de Trabalho, e então o usuário pode corrigir o erro e reenviar o formulário para que o mesmo seja inserido no banco de dados sendo o usuário redirecionado para a Lista de Calhas. O **Cadastro de Laje** funciona da mesma maneira, e a única diferença é que ele deve selecionar a Calha que esta Laje pertence, para que a mesma seja instanciada no formulário. Seu teste, na Figura 5.8, identificou mais uma questão de usabilidade do que um erro propriamente dito, onde não tinha nenhuma tela de mensagem de erro.

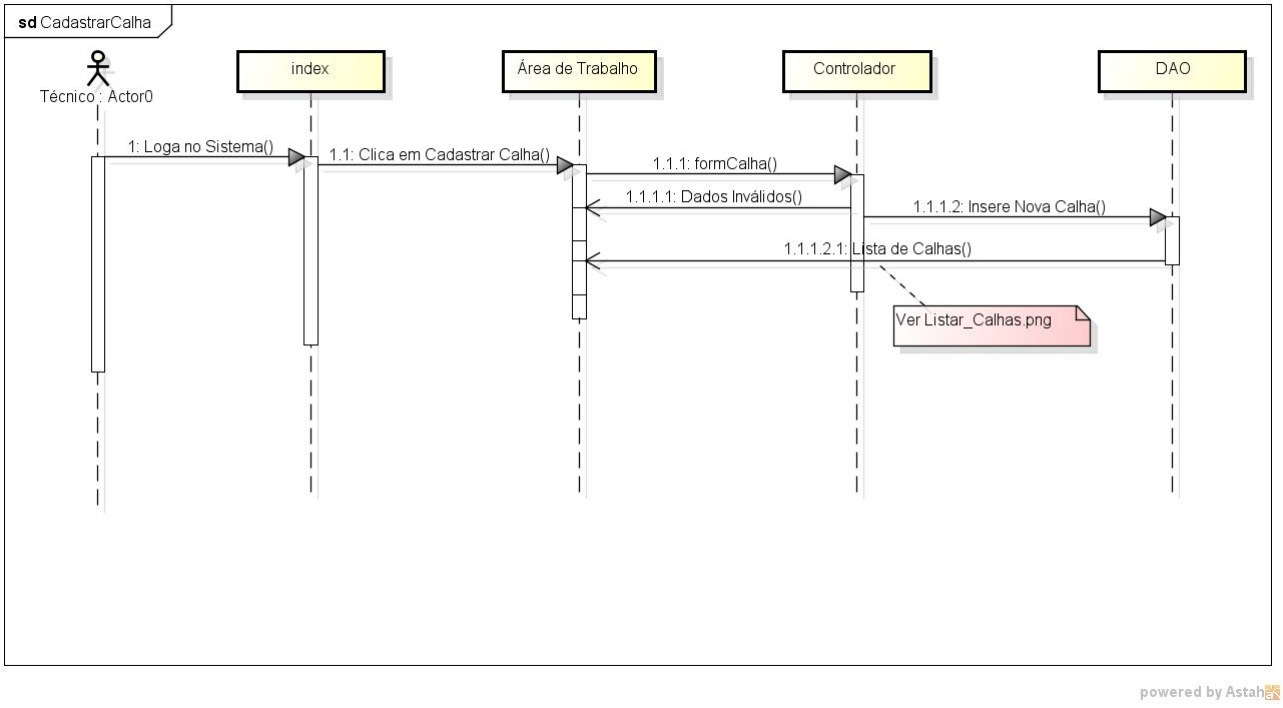

Figura 5.7: Diagrama de Sequência do Cadastro de Calha modelado na etapa de “Detalhar por Funcionalidade“.

Cadastrar Calha:

Cadastrar Calha - Nova Calha			
nº Teste	**ID da Calha (Seleção)**	**Dados Históricos**	**O que aconteceu**
01	Esquerda	Teste 1	Aceitou
02	Direita	Teste 2	Aceitou
03	Central	Teste 3	Aceitou
04	Esquerda	Teste 4	Execution exception*
05	Esquerda		Aceitou

*Execution exception: Calha Esquerda já cadastrada. Pode-se colocar uma caixinha avisando.
É possível adicionar qualquer coisa nos dados históricos, até nada.

Figura 5.8: Teste realizado para o Cadastro de Calhas.

- **Listar Calhas:** O cenário modelado no diagrama ilustrado na Figura 5.9 consiste em que o usuário loga no Sistema sendo direcionado para a tela inicial, o index, clica em Listar Calhas acionando uma método no controlador que busca no banco as Calhas cadastradas. Após a busca é retornado para a Área de Trabalho a listagem de Calhas Cadastradas. A **Listagem de Lajes, Campanhas, Amostras e Atividades** funciona da mesma forma, sendo o que muda são as ações que podem ser realizadas em cada uma. Seu teste, na figura 5.10 verificou a não existência de erros e a conformidade com a usabilidade.

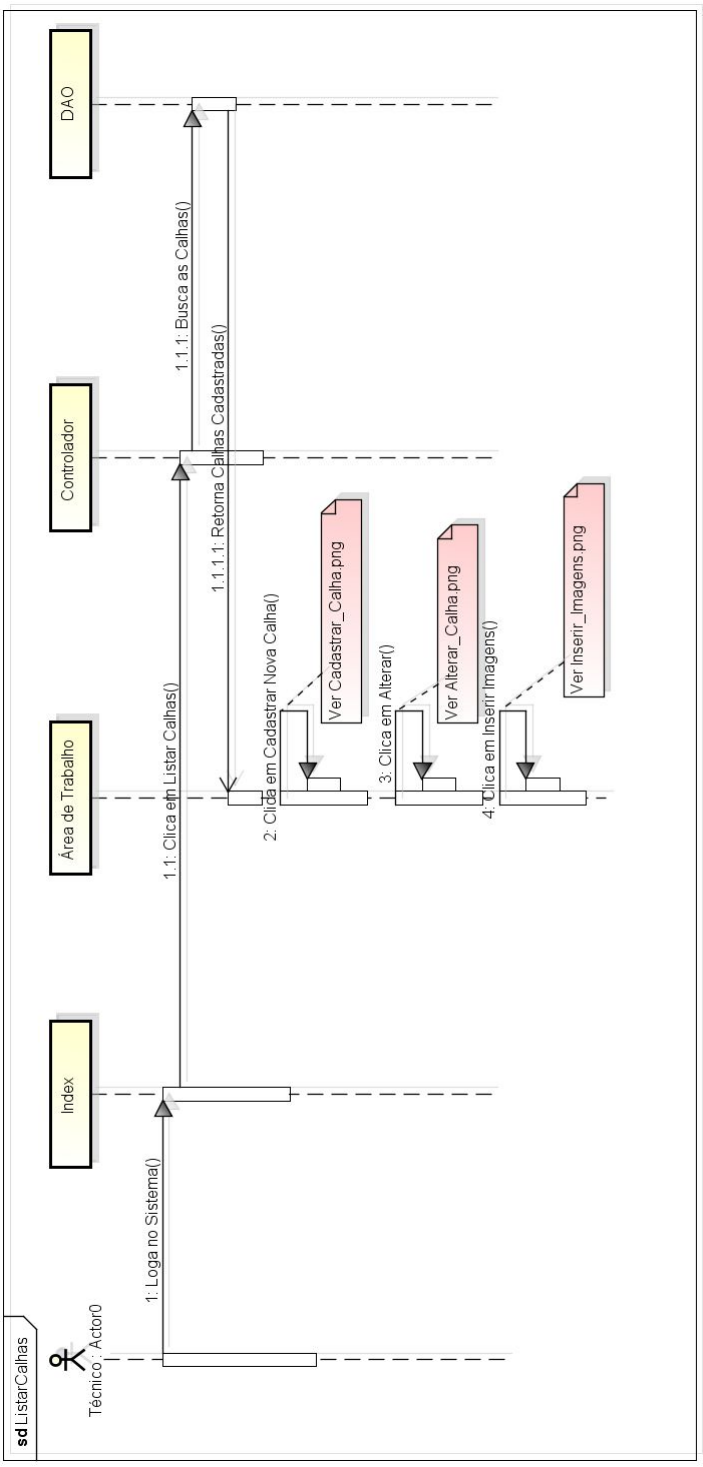

Figura 5.9: Diagrama de Sequência da Listagem de Calhas cadastradas modelado na etapa de "Detalhar por Funcionalidade".

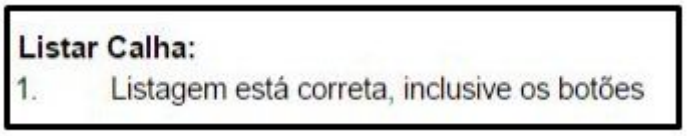

Figura 5.10: Teste realizado para a Listagem de Calhas.

- **Alterar Calha:** O cenário modelado no diagrama ilustrado na Figura 6.4 consiste em que o usuário entra no Sistema, sendo direcionado para a tela inicial, o index, clica em Listar Calhas realizando todo o cenário descrito anteriormente, Clica em Alterar ao lado de uma Calha cadastrada atualizando a Área de Trabalho e passando como parâmetro o id da Calha, que busca os dados cadastrais dessa Calha no banco de dados e retorna esses dados em um formulário preenchido na Área de Trabalho, então o usuário pode realizar as alterações necessárias e reenvia-los ao banco de dados. Caso ele queira remover a calha, basta clicar em Remover esta Calha, que passará o id da mesma para o controlador, fazendo a remoção dos dados dela no banco de dados. Seu teste, na Figura 5.12 identificou a não alteração do campo para inserção de Calha para um campo de seleção, o mesmo utilizado ao cadastrar a Calha. Na questão da usabilidade, verificou-se que o título da página era o de Cadastrar Calha, onde deveria ser Alterar Calha.

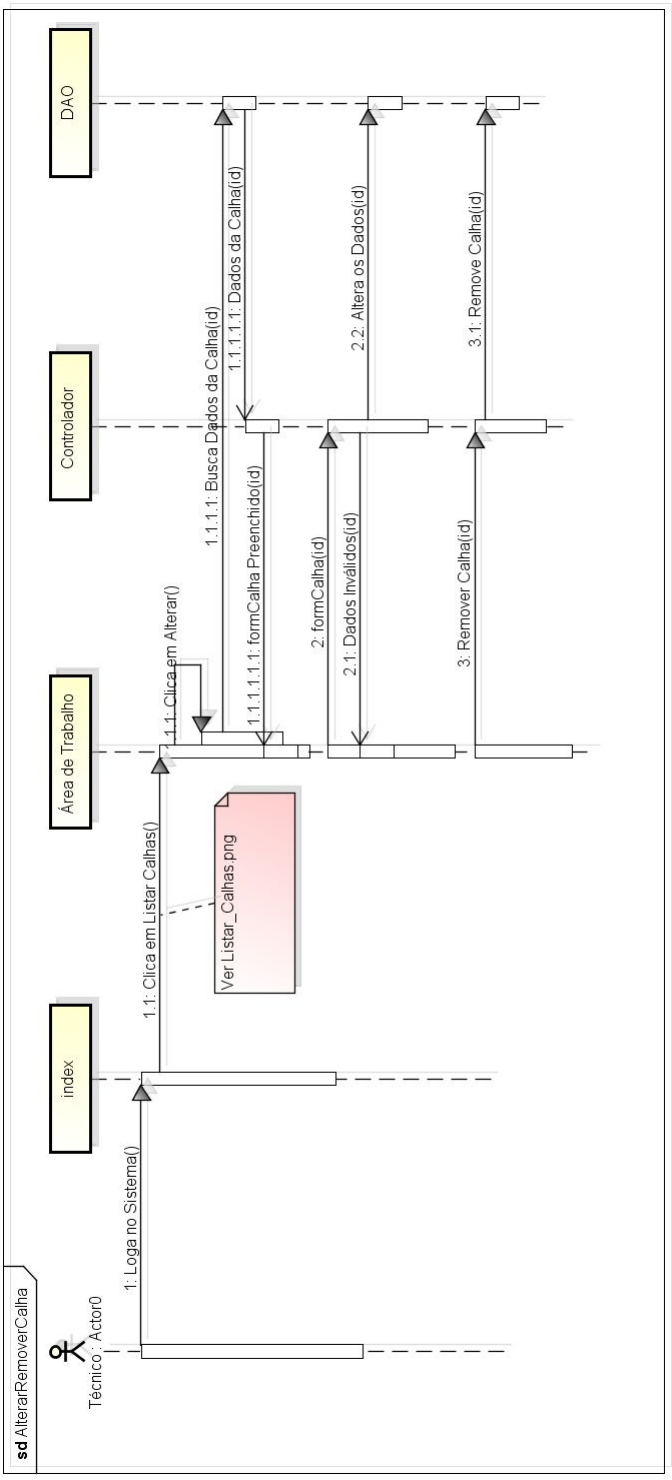

Figura 5.11: Diagrama de Sequência da alteração ou remoção dos dados cadastrais de uma Calha modelado na etapa de “Detalhar por Funcionalidade“.

Alterar Calha:

Alterar Calha			
nº Teste	**ID da Calha (Seleção)**	**Dados Históricos**	**O que aconteceu**
01	Esquerda	Teste 1 alterado	Aceitou
02	Direita	Teste 2 alterado	Aceitou
03	Central	Teste 3 alterado	Aceitou
04	CALHA	Teste 4 alterado	Aceitou
05	Esquerda	alterado	Aceitou

1. Quando clicado em alterar o título da pagina de alteração é "Cadastrar Calha", não seria melhor "Alterar Calha? Ou inserir uma label em cima como no cadastrar?
2. No alterar é possível mudar o nome da calha para qualquer coisa

Figura 5.12: Teste realizado para a Alteração de Calhas.

- **Inserir Imagens :** O cenário modelado no diagrama ilustrado na Figura 5.14 consiste em que o usuário entre no Sistema sendo direcionado para a tela inicial, o index, clica em Listar Calhas realizando todo o cenário descrito anteriormente, Clica em Inserir Imagens ao lado de uma Calha cadastrada atualizando a Área de Trabalho e passando como parâmetro o id da Calha, Insere a Imagem. O controlador transforma em bits e então armazena a imagem no banco de dados e após isso a Área de Trabalho é atualizada com a listagem de imagens. Seu teste, na Figura 5.14 verificou-se a conformidade com os requisitos e não foi encontrado nenhum erro em relação aos mesmos. Porém em relação a usabilidade foram encontrados 4 erros que podem ser vistos na imagem.

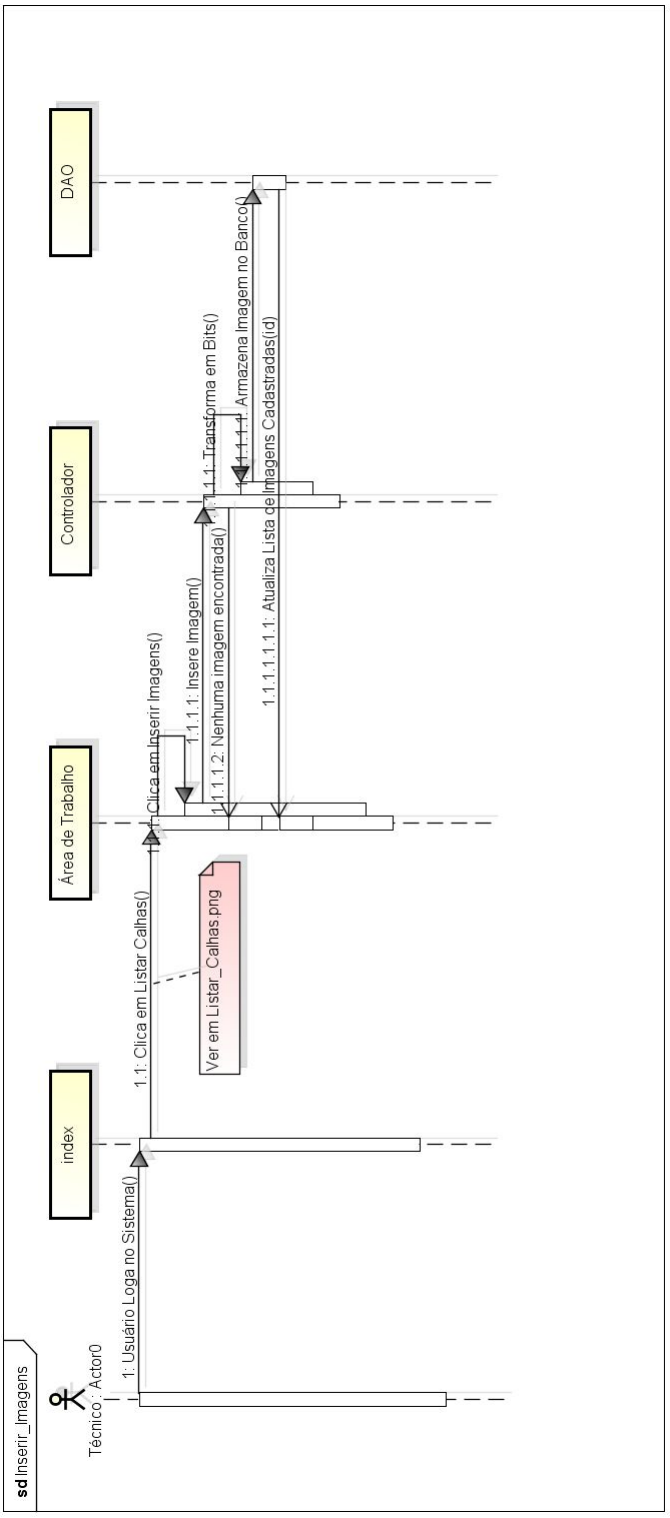

Figura 5.13: Diagrama de Sequência da inserção de imagens em calhas modelado na etapa de "Detalhar por Funcionalidade".

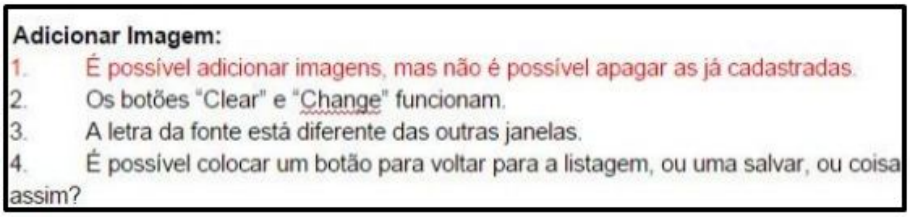

Figura 5.14: Teste realizado para a Inserção de Imagens nas Calhas.

- **Cadastrar Campanha:** O cenário modelado no diagrama ilustrado na Figura 5.15 consiste em que o usuário entre no Sistema sendo direcionado para a tela inicial, o index, clica em Cadastrar Campanha atualizando a Área de Trabalho, preenche os dados do formulário de cadastro da campanha que recebe o nome formCampanha no controlador. Caso o controlador verifique a existência de algum erro de inserção no formulário uma mensagem de erro é retornada para a Área de Trabalho, então o usuário pode corrigir o erro e reenviar o formulário para que o mesmo seja inserido no banco de dados. Ao finalizar o cadastro dessa campanha o usuário pode então representá-la no mapa. O primeiro passo é clicar em Inicializar Calha, para que seja gerado um polígono ao redor da Calha, e então pode desenhar o polígono que representa esta Campanha. Caso queira inserir Amostras dentro dessa Campanha ele clica em Gerar Amostras, então clica em Salvar Campanha para que o conjunto de dados GEOJson que representa essa Campanha seja instanciado junto do formulário. Então o usuário é redirecionado para a Lista de Campanhas Cadastradas. No quesito desenvolvimento, essa funcionalidade foi a mais complexa pois para seu desenvolvimento foi preciso estudar diversas funções do OpenLayers, como desenho de polígonos, geração de um ponto dada uma coordenada e a transoformação das informações do desenho no mapa para dados GeoJSON.

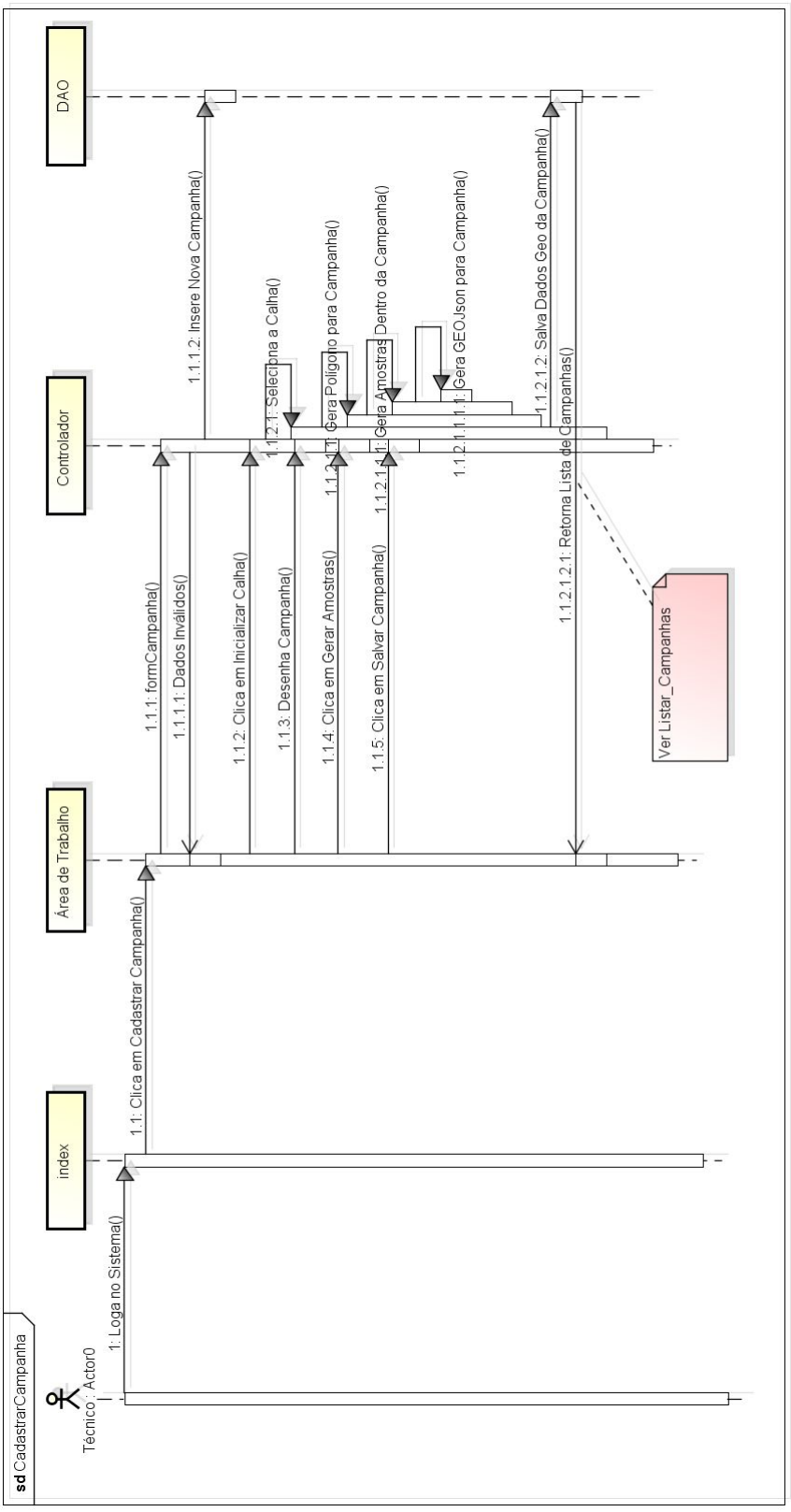

Figura 5.15: Diagrama de Sequência do Cadastro de Campanha modelado na etapa de “Detalhar por Funcionalidade“.

As demais funcionalidades, **Cadastrar Laje, Listar Lajes, Alterar Laje, Cadastrar Frente de Concretagem, Alterar Frente de Concretagem, Listar Campanhas, Alterar Campanha, Cadastrar Amostra, Cadastrar Atividade, Listar Atividade e Alterar Atividade**, também foram modeladas no mesmo formato que as descritas acima e serão apresentadas em suas versões implementadas, testadas e validadas no próximo capítulo.

Capítulo 6

A Validação do SIGVERTE

Neste capítulo é apresentada a validação do Sistema pelo usuário, através da criação de um Manual do Usuário e de um Teste de Aceitação, que faz parte da fase final do FDD. A validação de um software é o ato de assegurar que o mesmo seja adequado e atende as necessidades para o qual foi projetado. Ou seja, a confirmação de que este cumpra as especificações [21]. O Teste de Aceitação é um processo de teste de usuário no qual o objetivo é decidir se o software é bom o suficiente para ser implantado e usado em seu ambiente operacional [21]. Ambos validam o software no sentido de verificar se o mesmo está cumprindo com os requisitos propostos. Além do Manual do Usuário servir como uma forma de documentar os processos de software e ajudar o usuário a entender as funcionalidades do Sistema.

6.1 Manual do Usuário

A primeira forma de validação utilizada foi a criação de um Manual do Usuário, ele foi feito para que os especialistas do modelo de negócio e usuários pudessem avaliar e fazer sugestões para o software antes mesmo do Teste de Aceitação. Ele está separado pelas funcionalidades, divididas em Menu Lateral Esquerdo e Área de Trabalho para melhor visualização, com suas telas e descrição da sequência que deve ser feita junto das caixas amarelas na imagem ilustrando os passos que o usuário deve seguir. O Menu Lateral Esquerdo e Área de Trabalho a direita, onde todas funcionalidades são executadas, são apresentados nas Figuras 6.1, 6.8, 6.18 e 6.30.

6.1.1 Calha

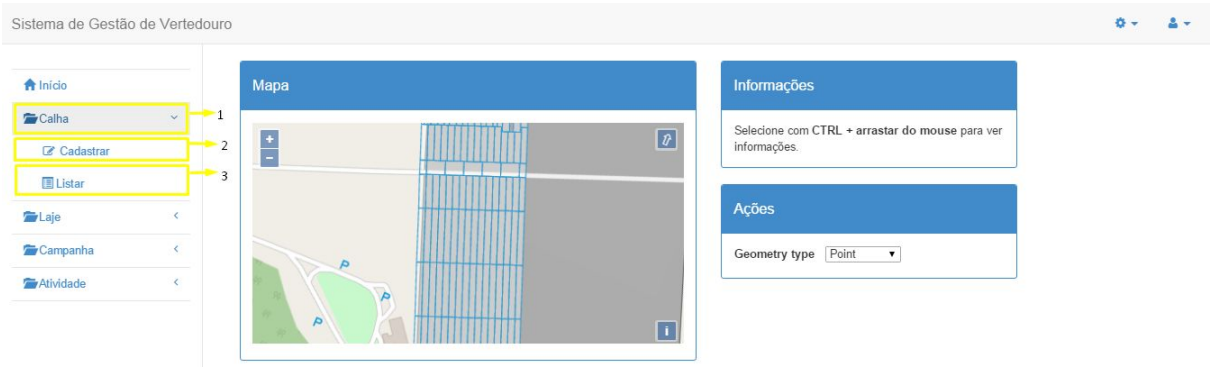

Figura 6.1: Menu de acesso das funcionalidades da Calha.

Para efetuar o cadastro de uma calha, que é um dos pré-requisitos do Sistema, basta clicar sobre Calha (1) e Cadastrar (2) como ilustrado na Figura 6.1, ação que fará o redirecionado para a Figura 6.2. Para a listagem das calhas cadastradas, basta clicar em Calha (1) e Listar (3) sendo redirecionado para a Figura 6.3.

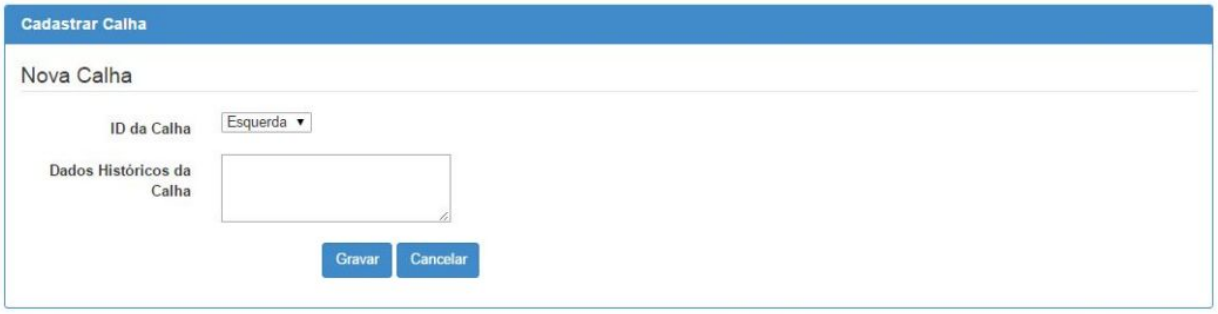

Figura 6.2: Cadastro de Calha acionada pela ação Calha(1), Cadastrar(2) da Figura 6.1.

Cadastrar Calha: Para efetivar o cadastro, preencher o formulário e clicar no botão Gravar. Caso não queira concluir o cadastro, clicar em Cancelar e será redirecionado para a lista de calhas.

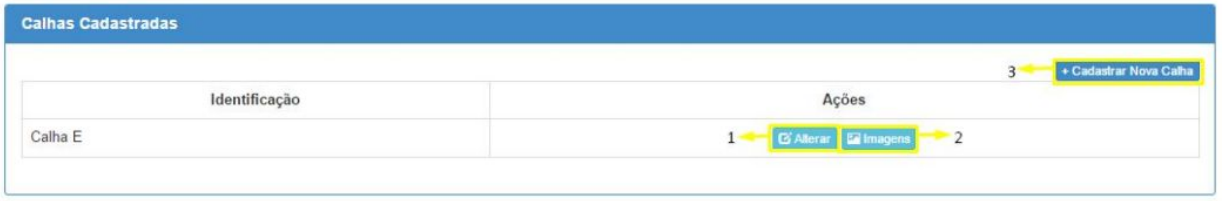

Figura 6.3: Lista de Calhas acionada pela ação Calha(1), Listar(3) da Figura 6.1.

Lista de Calhas: A lista de calhas é dividida em duas colunas, a primeira identificando a calha cadastrada e a segunda apresentando ações que podem ser realizadas sobre esta calha. As ações que podem ser feitas são a de alteração dos dados da calha através do botão Alterar (1), conforme apresentado na Figura 6.4, e de inserir imagens para esta calha através no botão Imagens (2) apresentado na Figura 6.5.

Figura 6.4: Alterar/Remover calha acionado pela ação Alterar(1) da Figura 6.3.

Remover Calha: Para remover a calha basta clicar sobre o botão Remover essa Calha (1). **Alterar Calha:** Para alterar os dados cadastrais da calha, basta alterar as informações já cadastradas e clicar no botão Gravar (2).

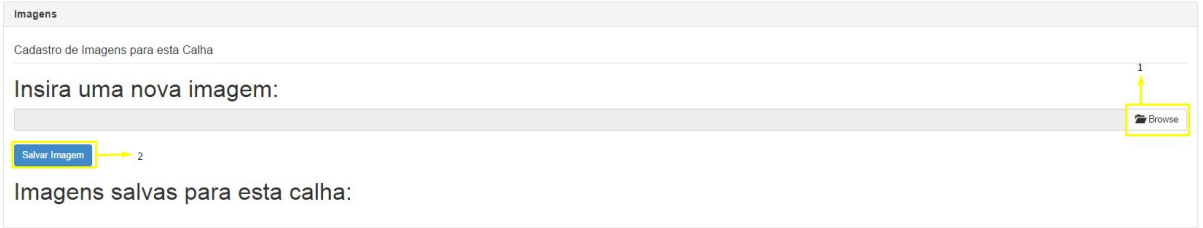

Figura 6.5: Inserir imagens da calha acionada pela ação Imagens(2) da Figura 6.3.

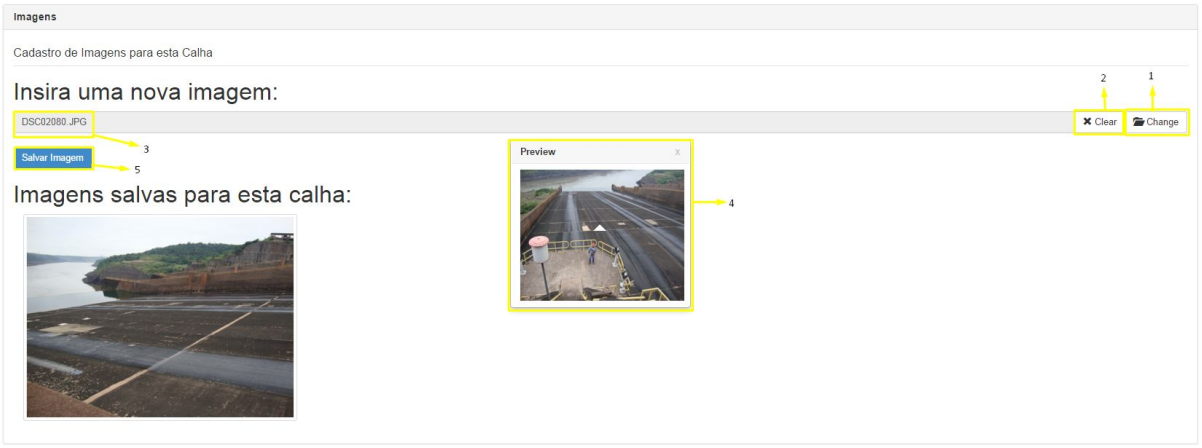

Figura 6.6: Inserir imagens da calha acionada pela ação Imagens(2) da Figura 6.3.

Inserir Imagens na calha: Para inserção de imagens da calha, basta clicar sobre o botão Browse (1). Ao selecionar uma imagem, seu nome é alterado para Change (1). Caso queira substituir a imagem, surge ao seu lado o botão Clear (2) com a função de limpar o campo de cadastro (3). Também é apresentada a imagem selecionada no campo Preview (4). Para salvar essa imagem para a calha basta clicar no botão Salvar Imagem (5). Após efetivado o cadastro, ela já é listada na mesma tela como mostrado na Figura 6.7.

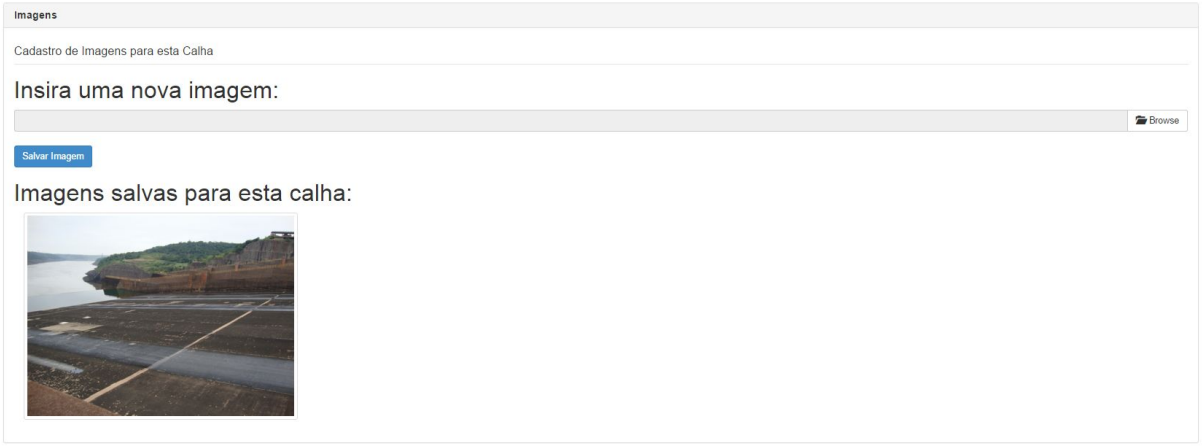

Figura 6.7: Inserir imagens da calha acionada pela ação Imagens(2) da Figura 6.3.

Lista de imagens da calha: São listadas todas as imagens da calha na mesma tela de inserção das mesmas, possibilitando a inserção de quantas imagens forem necessárias.

6.1.2 Laje

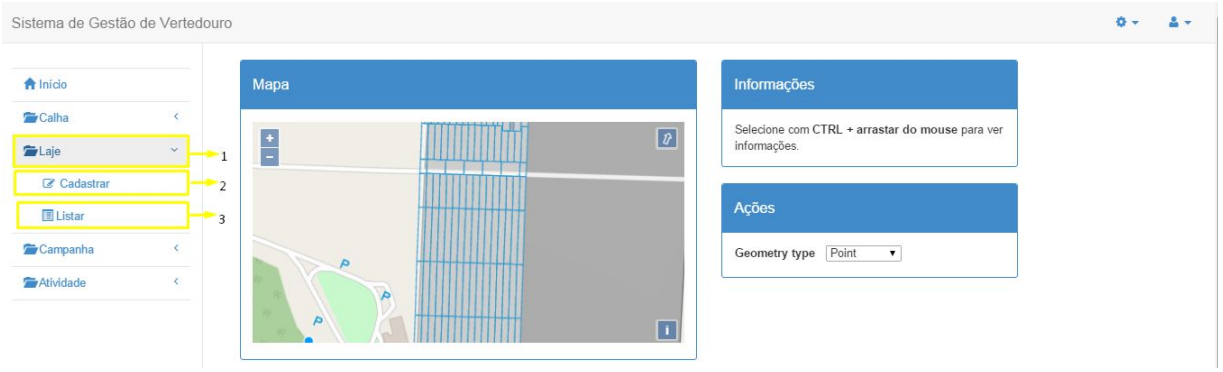

Figura 6.8: Menu de acesso das funcionalidades da Laje.

Para efetuar o cadastro de uma laje, basta clicar sobre Laje (1) e Cadastrar (2) como ilustrado na Figura 6.8, sendo redirecionado para a tela ilustrada na Figura 6.9. Para a listagem das lajes cadastradas, basta clicar em Laje (1) e Listar (3) sendo redirecionado para a Figura 6.10.

Figura 6.9: Cadastro de Laje acionada pela ação Laje(1), Cadastrar(2) da Figura 6.8.

Cadastrar Laje: Na tela de cadastro de laje, basta selecionar uma calha cadastrada, preencher os outros campos e clicar no botão Gravar. Para cancelar a ação basta clicar no botão Cancelar.

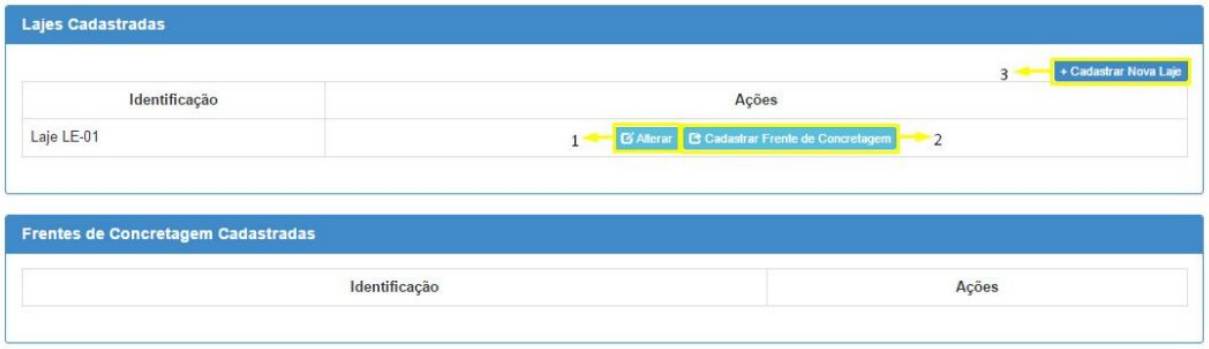

Figura 6.10: Lista de Lajes acionada pela ação Laje(1), Listar(2) da Figura 6.8.

Lista de Lajes Cadastradas: A lista de lajes é dividida em 2 colunas, uma de identificação e a outra de ações que podem ser realizadas sobre a mesma, que são a de alterar seus dados cadastrais clicando sobre o botão Alterar (1) apresentado na Figura 6.11 e cadastrar seus dados de frente de concretagem em Cadastrar Frente de Concretagem (2) apresentado na Figura 6.12. Para cadastrar uma nova laje, basta clicar sobre Cadastrar Nova Laje (3).

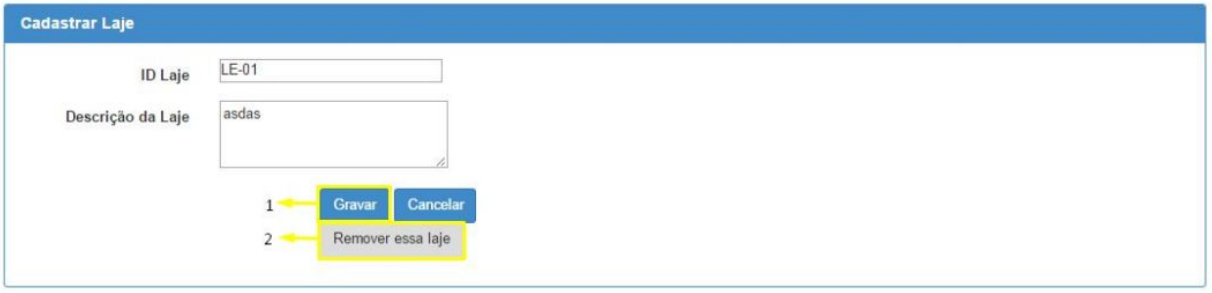

Figura 6.11: Alterar/Remover Laje acionada pela ação Alterar(1) da Figura 6.10.

Alterar Laje: Para realizar a alteração, basta alterar qualquer dado desejado e clicar em Gravar (1). **Remover Laje:** Para remover a laje basta clicar em Remover essa laje (2).

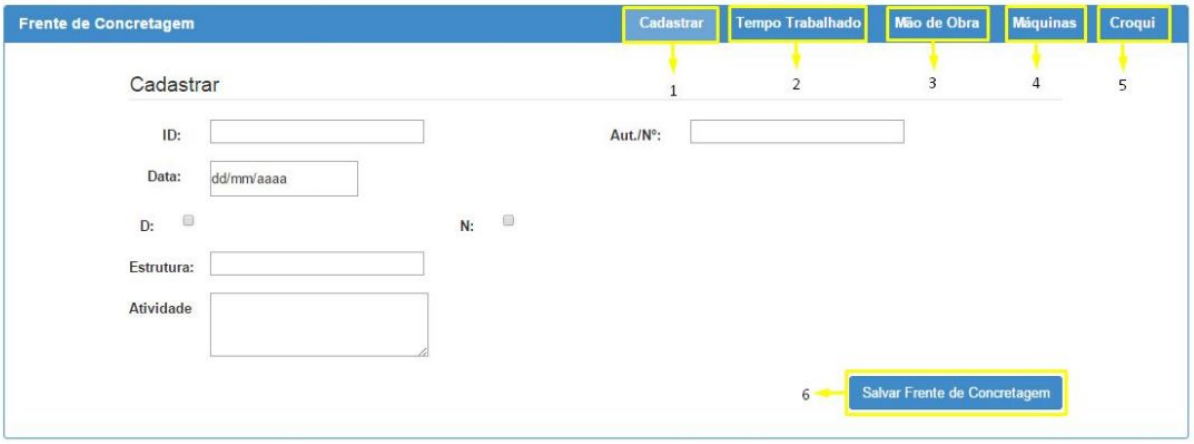

Figura 6.12: Cadastro de Frente de Concretagem acionado pela ação Cadastrar Frente de Concretagem (2) da Figura 6.10.

Cadastro de Frente de Concretagem: O formulário da frente de concretagem é dividido em 5 abas, são elas: Cadastrar (1) Figura 6.12, Tempo Trabalhado (2) Figura 6.13, Mão de Obra (3) Figura 6.14, Máquinas (4) Figura 6.15 e Croqui (5) Figura 6.16. Cada aba pode ser preenchida separadamente, sem que o usuário precise cadastrar as outras, bastando clicar no botão Salvar Frente de Concretagem (6).

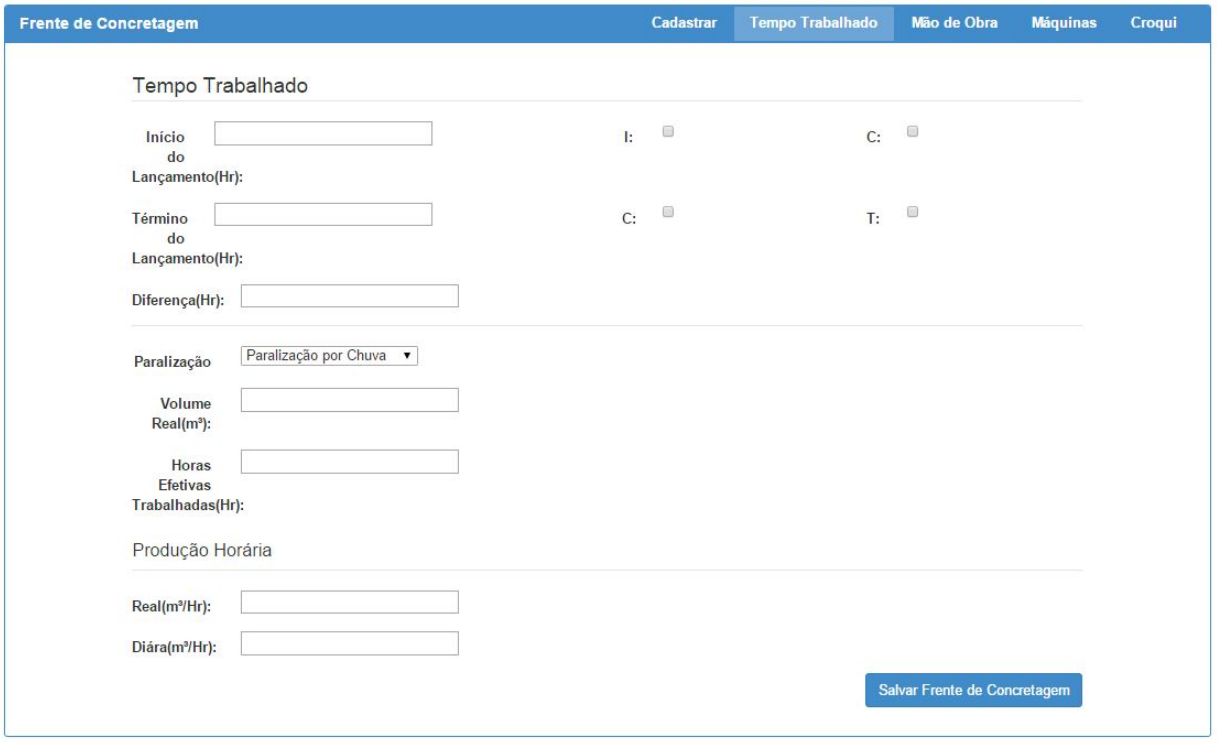

Figura 6.13: Tempo Trabalhado, aba 2 da Figura 6.12.

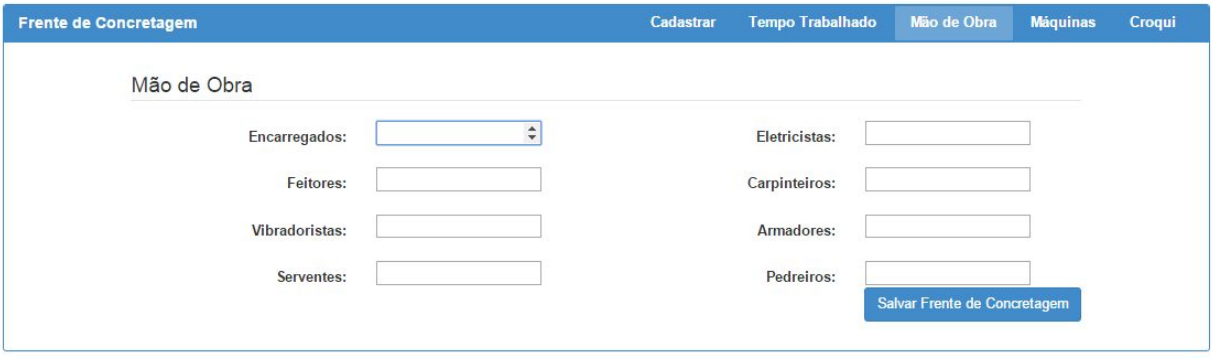

Figura 6.14: Mão de Obra, aba 3 da Figura 6.12.

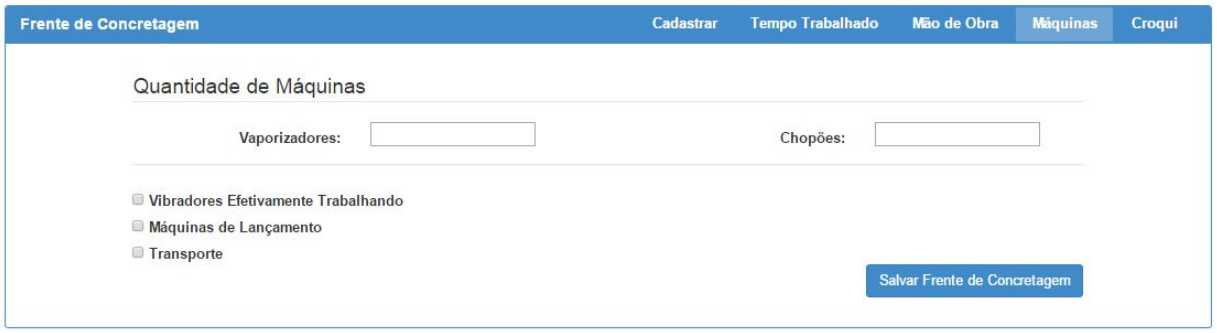

Figura 6.15: Máquinas, aba 4 da Figura 6.12.

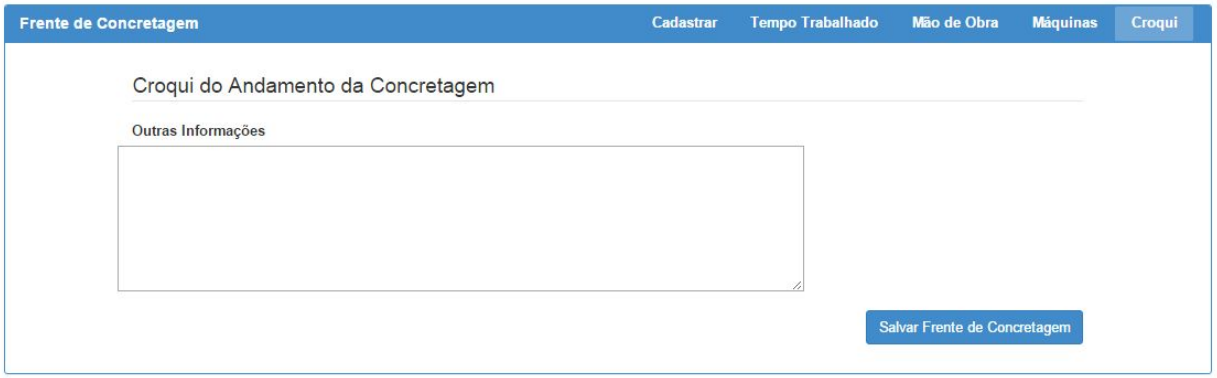

Figura 6.16: Croqui, aba 5 da Figura 6.12.

Para efetuar o cadastro basta preencher os dados em todas as abas e na de Croqui clicar no botão Salvar Frente de Concretagem (1) e será redirecionado para Lista de Lajes e de Frentes de Concretagem Figura 6.17.

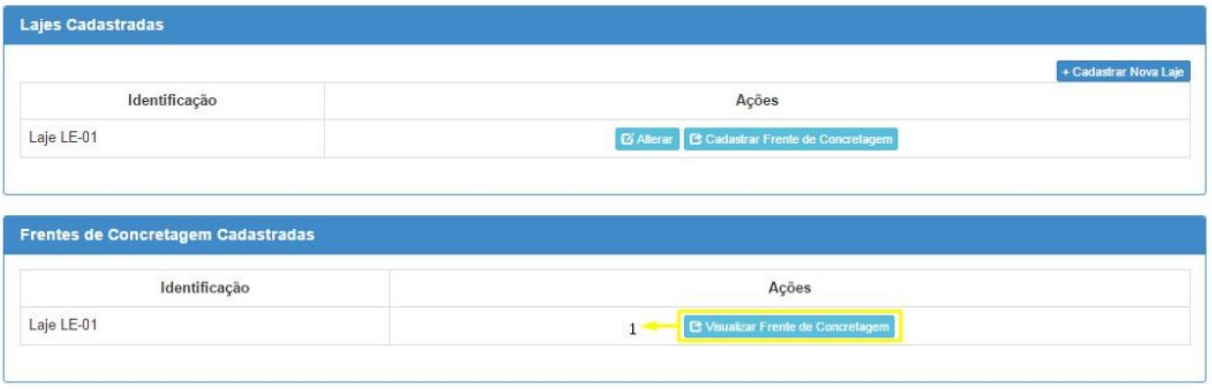

Figura 6.17: Lista de Lajes e Frentes de Concretagem, apresentada abaixo da lista de Lajes.

Lista de Lajes e Frentes de Concretagem: A lista de Frentes de Concretagem Cadastradas é apresentada abaixo da Lista de Lajes. Ela é dividida em duas colunas, uma de Identificação da Laje que pertence e outra de Ações. Ao clicar em Visualizar Frente de Concretagem (1) pode-se visualizar os dados correspondentes á Frente de Concretagem.

6.1.3 Campanha

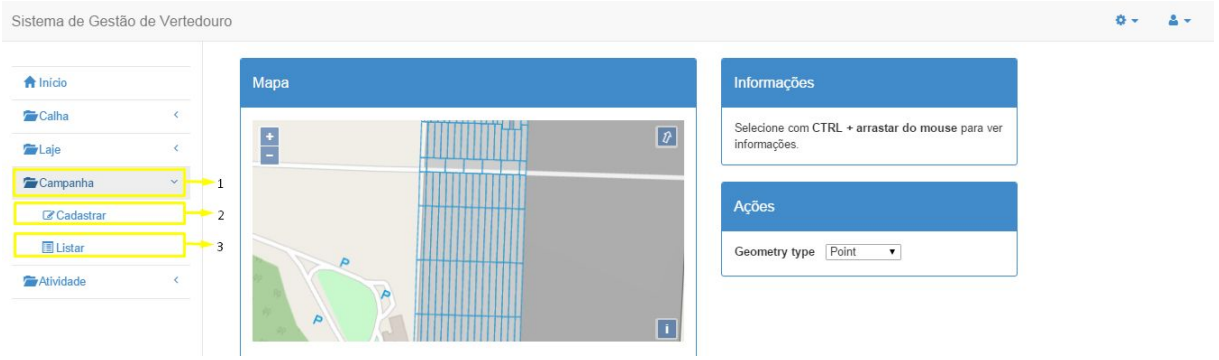

Figura 6.18: Menu de acesso das funcionalidades da Campanha.

Para efetuar o cadastro de uma campanha, basta clicar sobre Campanha (1) e Cadastrar (2) como ilustrado na Figura 6.18, sendo redirecionado para a tela ilustrada na Figura 6.19. Para a listagem das campanhas cadastradas, basta clicar em Campanha (1) e Listar (3) sendo redirecionado para a Figura 6.24.

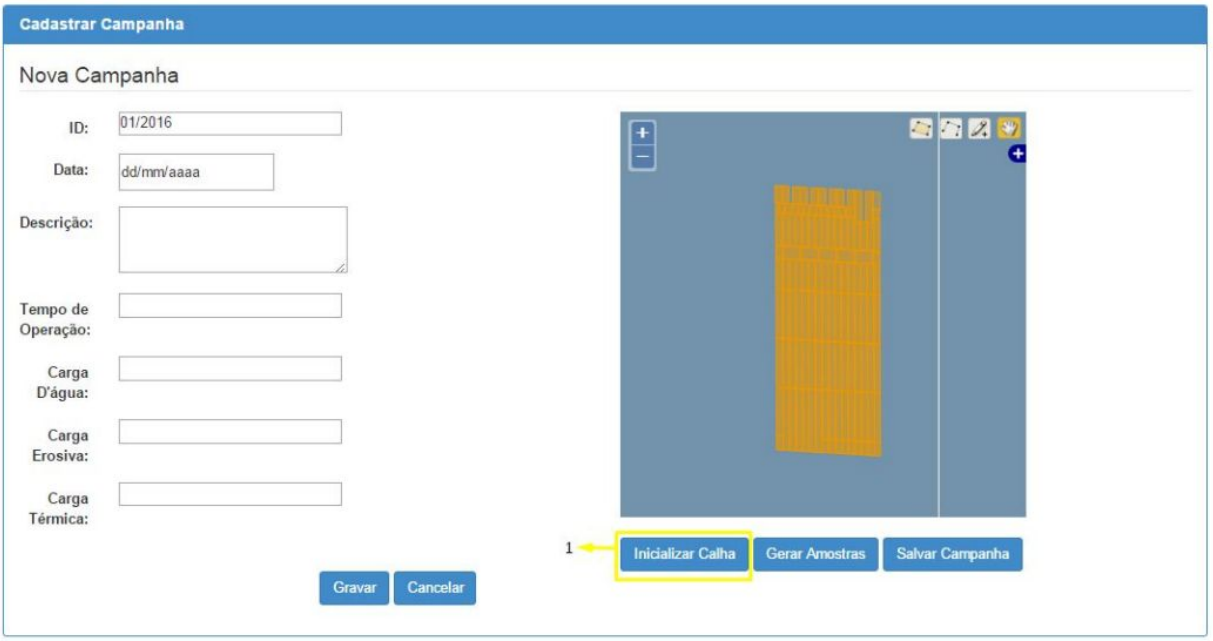

Figura 6.19: Cadastro de Campanha acionado pela ação Campanha (1), Cadastrar (2) da figura 6.18.

Cadastrar Campanha: Ao realizar o cadastro de uma campanha, após efetuar o cadastro dos dados, é preciso inicializar a calha. Para isso, o primeiro passo é clicar no botão Inicializar Calha (1) criando um polígono ao redor da calha, como mostra a Figura 6.20.

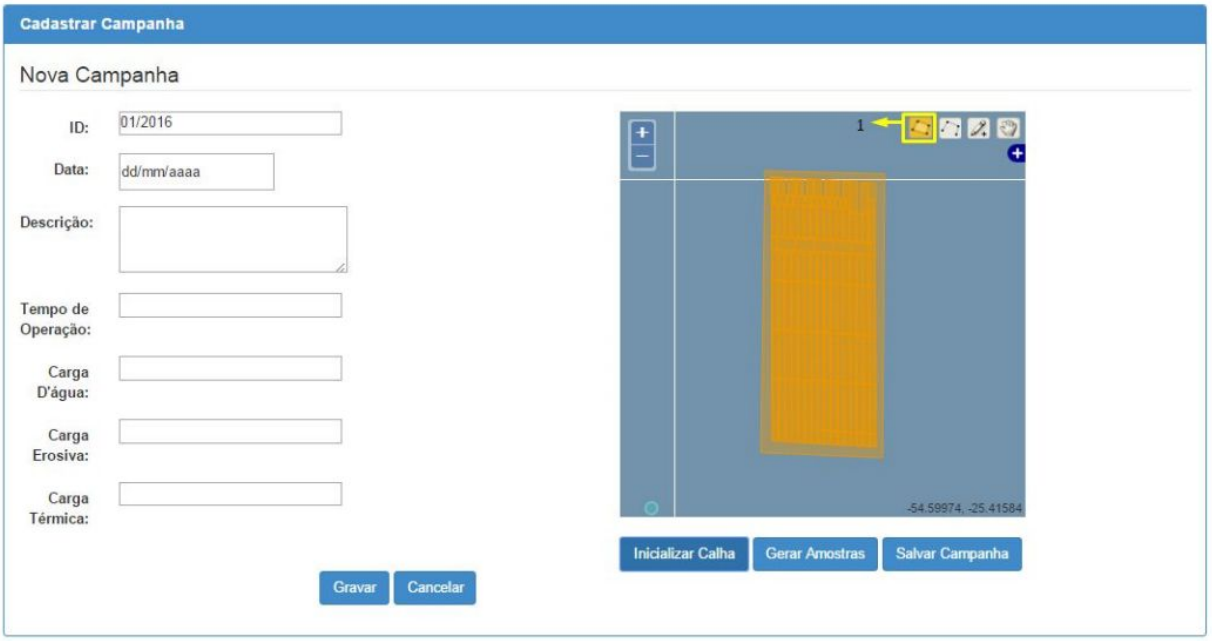

Figura 6.20: Desenhar Campanha, após Inicializar a Calha.

Desenhar Campanha: Após inicializar a calha, a próxima etapa é desenhar um polígono que representará a Campanha na Calha. Para fazer isso basta clicar no botão (1) e com o mouse configurar os vértices do mesmo no mapa.

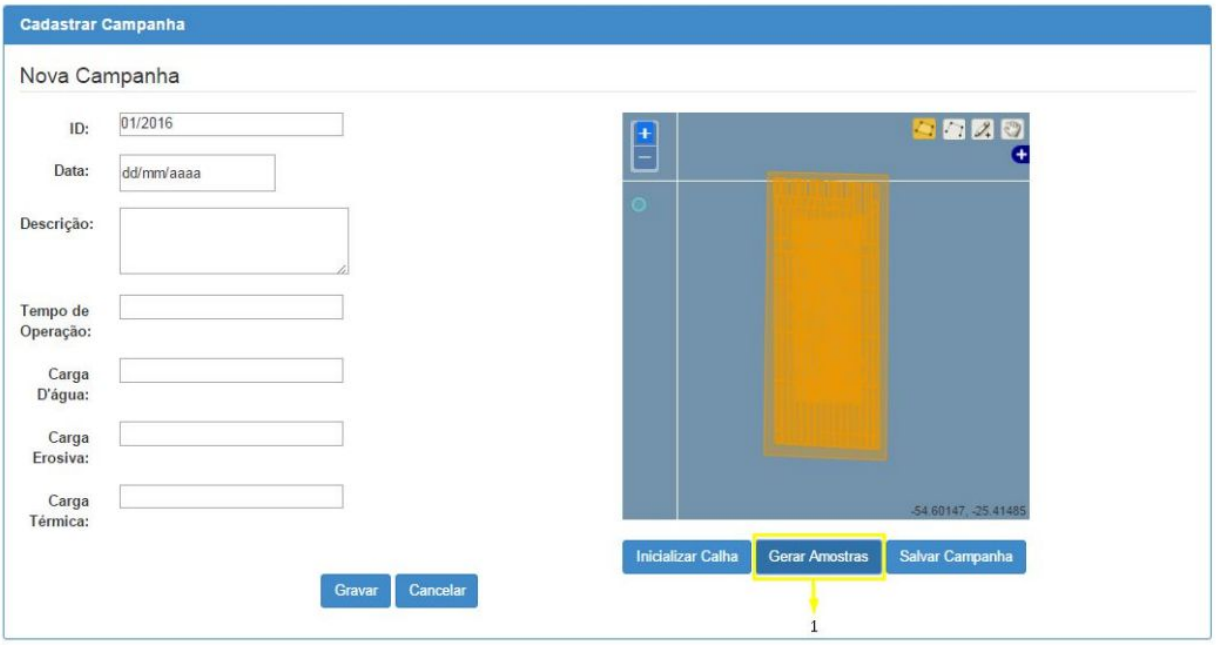

Figura 6.21: Gerar Amostras para o polígono representante da Campanha.

Gerar Amostras: Depois de ter a Campanha desenhada, basta clicar no botão Gerar Amostras (1) como ilustrado na Figura 6.21. Essa funcionalidade gera amostras aleatórias para a campanha dentro do polígono que a representa.

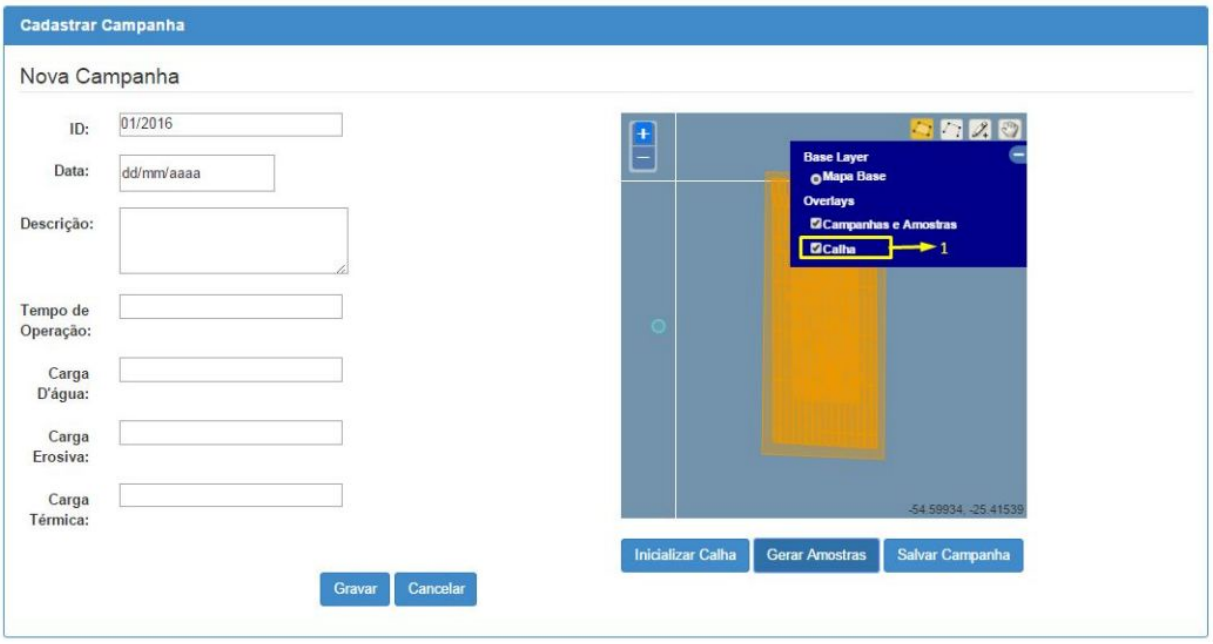

Figura 6.22: Filtrar Camadas para visualizar as Campanhas e Amostras.

Filtrar Camadas: Para melhor visualizar o polígono que representa a campanha e as amostras geradas, basta clicar no botão + no canto superior direito do mapa que abrirá um menu onde podem ser selecionadas as camadas que aparecerão no mapa. Uma camada representa o que é visível no mapa. Clicando na caixa Calha (1) na Figura 6.22, é possível filtrar para que só a camada de Campanha e Amostras apareça, como pode ser visto na Figura 6.23.

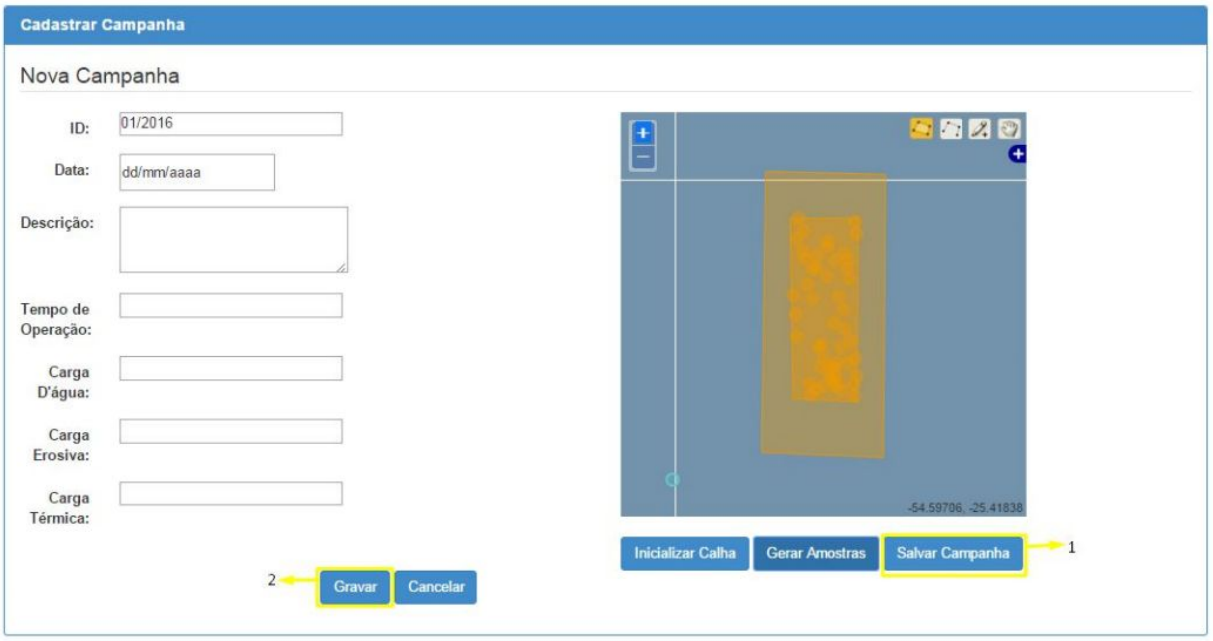

Figura 6.23: Salvar Campanha para vincular os dados dos polígonos desenhados ao cadastro.

Salvar Campanha: Após todas essas etapas, basta clicar no botão Salvar Campanha (1) para que o formulário receba os dados dos polígonos desenhados e clicar em Gravar (2) para efetuar o cadastro da campanha sendo redirecionado para Lista de Campanhas representada na Figura 6.24.

Figura 6.24: Lista de Campanhas, acionada pela ação Campanha (1), Listar (3) da figura 6.18.

Lista de Campanhas: A lista de campanhas é dividida em duas colunas, a de identificação da Campanha cadastrada e a outra de Ações onde o usuário pode realizar alteração e remoção da Campanha clicando sobre o botão Alterar (1) e cadastrar amostra para esta campanha clicando no botão Cadastrar Amostra (2), como ilustrado na Figura 6.24. Ao clicar sobre o

botão Cadastrar Nova Campanha (3), o usuário é direcionado à tela do cadastro de uma nova campanha.

6.1.4 Amostra

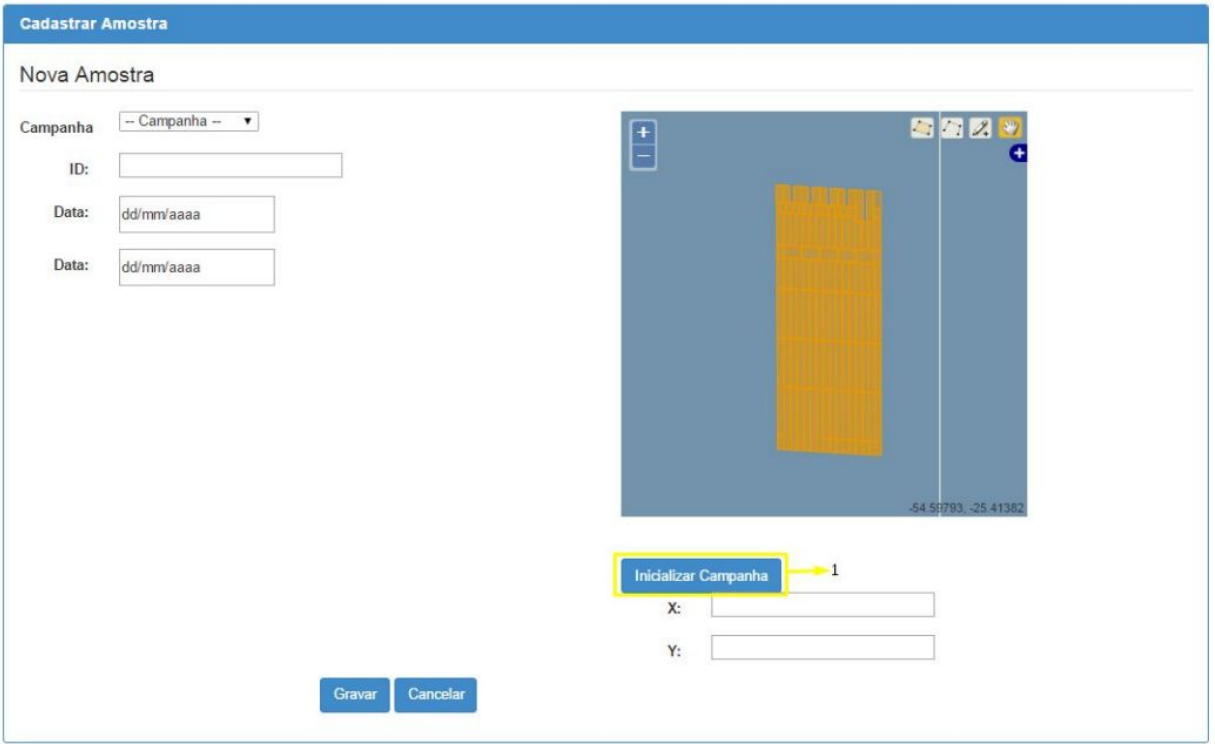

Figura 6.25: Cadastrar amostra, acionado pela ação Cadastrar Amostra (2) da figura 6.24.

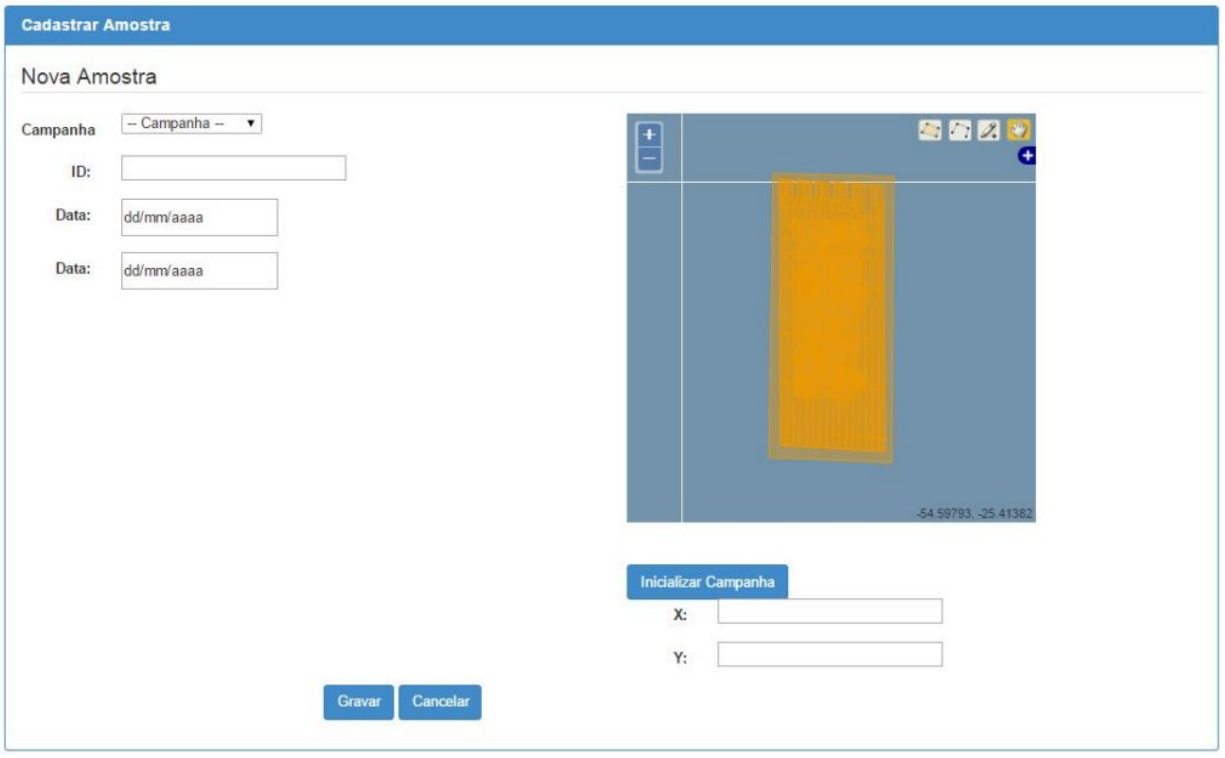

Figura 6.26: Campanha inicializada, acionada pela ação Inicializar Campanha (1) da Figura 6.25.

Nova Amostra: Após efetuar o preenchimento do formulário do cadastro de amostra, para selecionar as coordenadas da amostra, o primeiro passo é clicar no botão Inicializar Campanha (1) como mostrado na Figura 6.25, inicializando o polígono da campanha que a amostra pertence ilustrado na Figura 6.26.

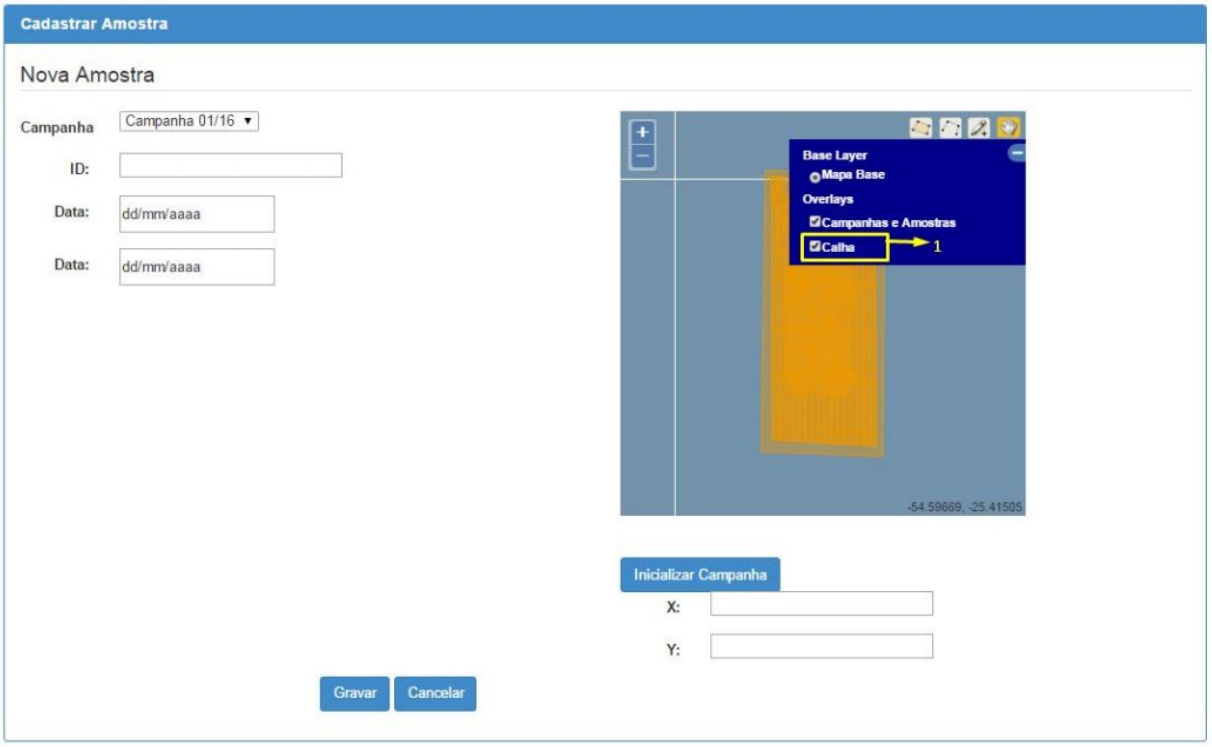

Figura 6.27: Filtrar a camada Calha (1) para melhor visualizar Campanhas e Amostras.

Filtrar Camada Calha: Para poder selecionar a amostra, deve ser desativada a camada da calha, clicando no botão + e desmarcando o campo Calha (1) ilustrado na Figura 6.27. Essa desativação deve ser feita para melhor visualizar as amostras, facilitando a seleção de uma delas.

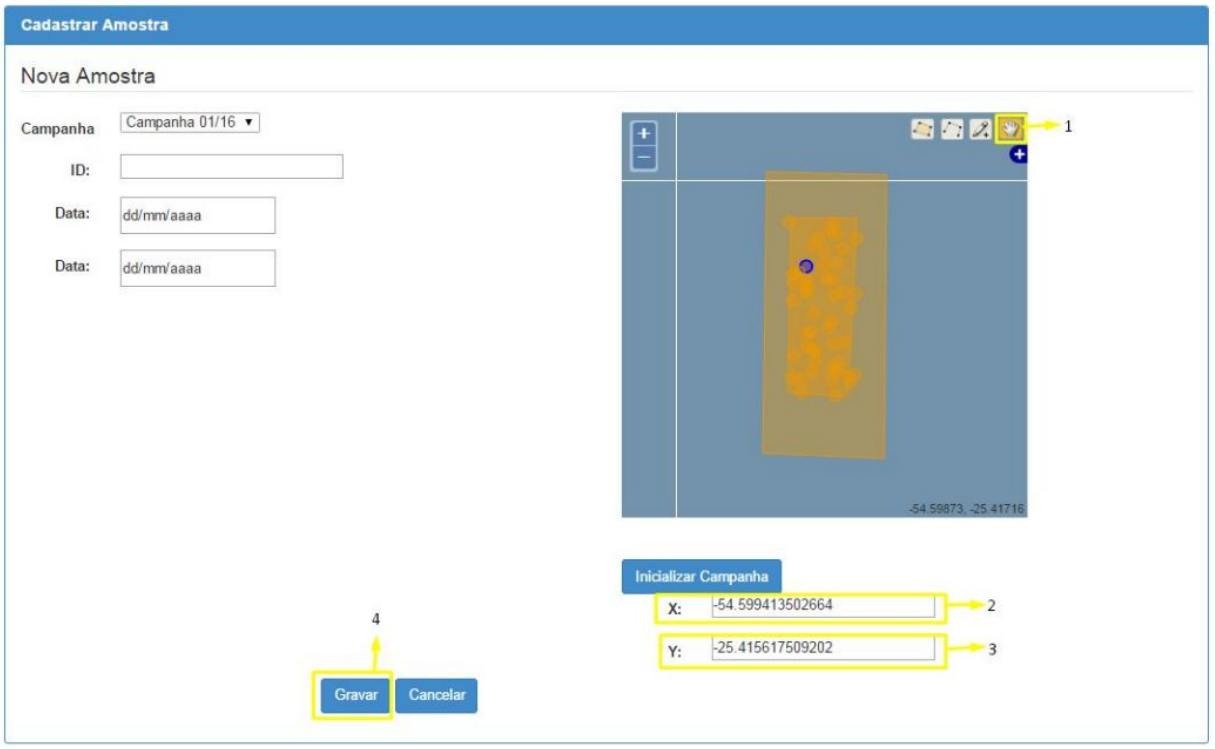

Figura 6.28: Selecionar Amostra vinculando os dados a mesma.

Selecionar Amostra: Para selecionar a amostra no mapa que representa a que está sendo cadastrada, deve-se clicar sobre o botão (1), selecionar uma amostra no mapa, então os campos X (2) e Y (3) serão preenchidos automaticamente. Após isso, basta clicar no botão Gravar (4), ações apresentadas na Figura 6.28 que a amostra será salva redirecionando para lista de campanhas e amostras como ilustra a Figura 6.29.

Figura 6.29: Lista de Amostras acionada após o cadastro de uma Amostra.

Lista de Amostras: A lista de amostras é dividida em duas colunas, a primeira é a

identificação da Amostra e da Campanha que ela pertence e a segunda coluna é a de ações que podem ser feitas sobre essa amostra, neste caso a de Remover (1), que tem como função de remover a amostra cadastrada.

6.1.5 Atividade

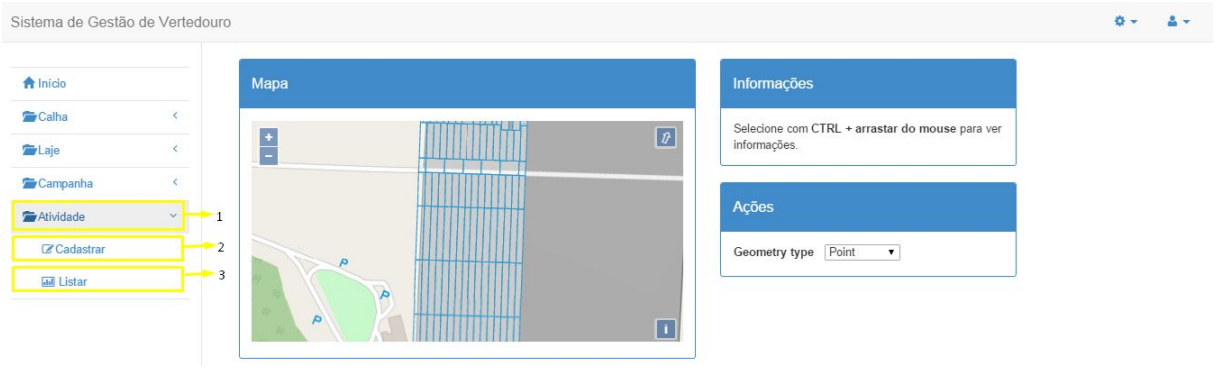

Figura 6.30: Menu de acesso das funcionalidades das Atividades.

Para efetuar o cadastro de uma atividade, basta clicar sobre Atividade (1) e Cadastrar (2) mostrados na Figura 6.30 sendo redirecionado para tela ilustrada na Figura 6.31. Para a listagem das atividades cadastradas, basta clicar em Atividade (1) e Listar (3) sendo redirecionado para a Figura 6.34.

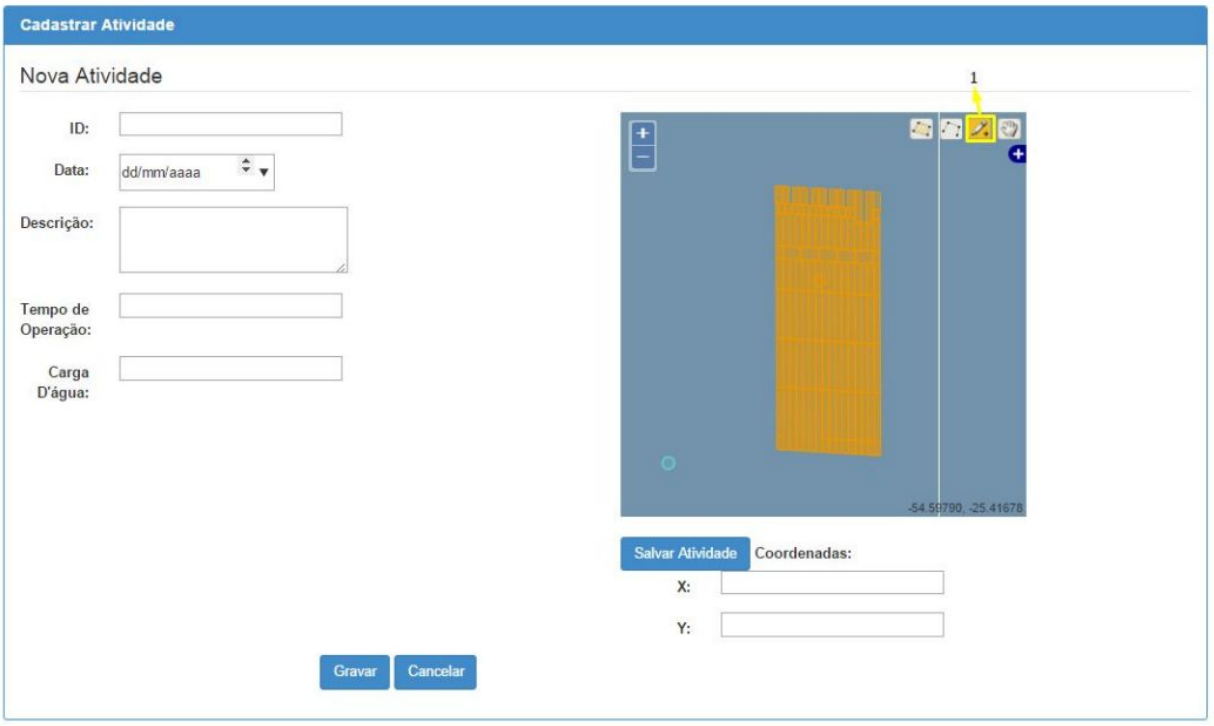

Figura 6.31: Cadastrar Atividade acionado pela ação Atividade (1), Cadastrar (2) da Figura 6.30.

Cadastrar Atividade: Ao realizar o cadastro de uma nova atividade, devem ser preenchidos os campos do formulário e após isso selecionar o botão (1) para desenho de uma amostra.

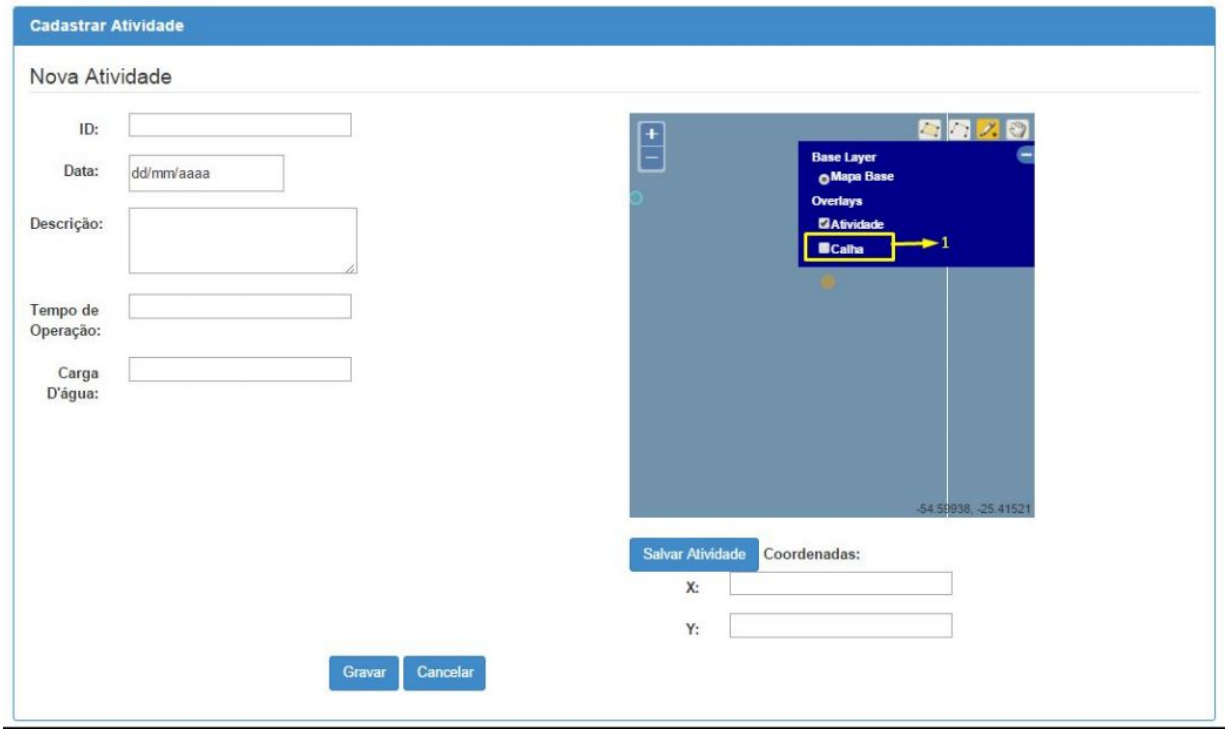

Figura 6.32: Desmarcar Camada Calha para visualização da atividade desenhada.

Desmarcar Camada Calha: Para ser possível selecionar a atividade desenhada, basta clicar no botão + no canto superior direito e desmarcar a caixa Calha (1) deixando somente a Camada de Atividade no mapa como é apresentado na Figura 6.32.

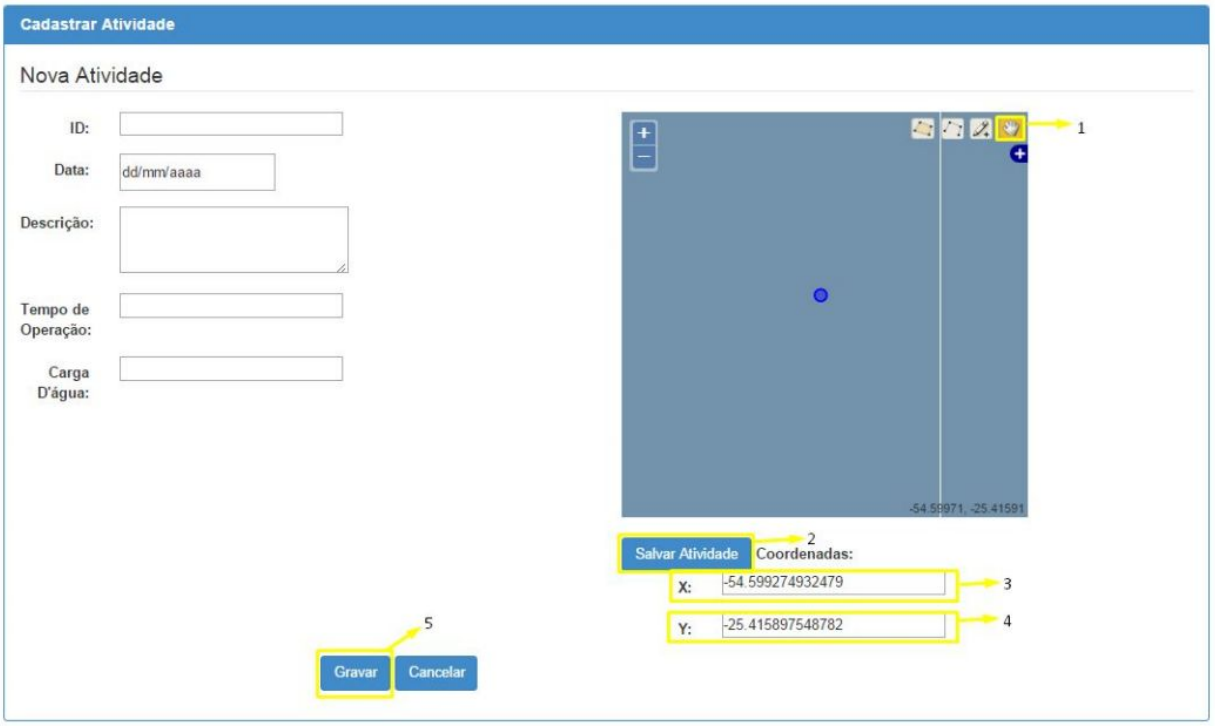

Figura 6.33: Selecionar Atividade para geração de suas coordenadas.

Selecionar Atividade: Para selecionar a atividade basta ativar o botão de seleção (1) e clicar sobre amostra desenhada que representa a atividade, fazendo com que os campos das coordenadas X(2) e Y(3) sejam preenchidos automaticamente. Após isso, deve-se clicar no botão Salvar Atividade (4) para vincular a amostra desenhada no mapa com a atividade e clicar em Gravar (5), ações apresentadas na Figura 6.33 para finalizar o cadastro sendo direcionado para Lista de Atividades ilustrada na Figura 6.34.

Figura 6.34: Lista de Atividades acionada após cadastrar a Atividade.

Lista de Atividades: A lista de atividades é dividida em duas colunas, a primeira representa a identificação da Atividade, e a segunda contém as ações que podem ser realizadas sobre a

Atividade, neste caso a alteração dos dados da atividade ao clicar sobre o botão Alterar (1). No botão Cadastrar Nova Atividade (2) o usuário pode cadastrar uma nova atividade.

6.2 Teste de Aceitação

O Teste de Aceitação do SIGVERTE foi realizado junto de um especialista do modelo de negócio e também usuário do Sistema, o Prof. Dr. Rogério Luis Rizzi, também orientador deste Trabalho de Conclusão de Curso. Foram inseridos dados fictícios e em alguns casos a tentativa de indução ao erro. O modelo de negócio do software foi aceito para esta versão, porém com alguns pedidos de mudanças em relação a interface e apresentação do Sistema. Foram sugeridas também novas funcionalidades, e o ajuste de algumas que serão apresentadas no próximo capítulo e farão parte da futura versão do SIGVERTE. Algumas mudanças na interface como apresentação mais iterativa dos erros ao inserir dados inconsistentes no Sistema e trocar o posicionamento de alguns botões já foi realizada, as demais mudanças foram inseridas na Lista de Funcionalidades para serem realizadas como atividades futuras pela equipe de desenvolvimento, as novas funcionalidades são apresentadas na Figura 6.35.

36	*	Gerenciar Campanha	Novos Critérios para Distruibuição de Pontos na Campanha**	Campanha	Alta
37	*	Gerenciar Usuário	Criar Níveis de Usuário***	-	Média
38	*	-	Aprimorar Funções da Geração e Desenho de Amostras	Campanha, Atividade, Amostra	Alta
39	*	-	Aprimorar Filtros do Mapa****	Mapa	Baixa

* Levantar Requisito

** Deixar o usuário escolher critérios. Randomico ou critérios específicos. Inserir, remover ou modificar um ou mais pontos, segundo o critério de inserção ou remoção (inserir randomicamente 3, remover randomicamente 3), modificar pontos

*** Qualquer modificação sensível, fazer backup automático e somente adm tem permissão para excluir, confirmação de exclusão.

**** Modificação nos filtros do mapa, onde aparece a layer base, remover.

Figura 6.35: Lista de Funcionalides novas acionadas após Teste de Aceitação.

Capítulo 7

Considerações Finais e Trabalhos Futuros

Este capítulo tem por objetivo demonstrar as conclusões a respeito desse trabalho de conclusão de curso e sugestões de trabalhos futuros.

7.1 Conclusão

O objetivo geral deste trabalho de conclusão de curso foi especificar, desenvolver e validar o Sistema de Gestão de Vertedouro SIGVERTE, utilizando a metodologia ágil de desenvolvimento de software "feature driven development" FDD. O objetivo principal foi desenvolver uma primeira versão do SIGVERTE, disponibilizando sua documentação completa, códigos fontes devidamente documentados, e outros artefatos necessários à sua continuidade por outros bolsistas que darão continuidade ao Projeto. Ao longo do texto apresentado pretendeu-se assegurar que tais objetivos foram cumpridos. O FDD proporcionou fundamentos às entradas de artefatos existentes, o desenvolvimento ágil do software e toda sua documentação, viabilizando a realização de todos os objetivos propostos. O Sistema foi desenvolvido com distintas tecnologias, com apoio do vasto material disponibilizado na internet. Inclusive o HTML5, CSS, JavaScript, Java e SQL possuem cursos no CodeAcademy [26] facilitando seu estudo. A tecnologia escolhida para o desenvolvimento de funcionalidades relativas ao georreferenciamento foi o OpenLayers, que está em sua versão 3. Porém, como essa versão está em testes Beta, sua documentação não está completa, e assim sendo, optou-se pela versão 2, já que ela tem documentação adequada. O SIGVERTE prove o gerenciamento das ações realizadas em prol da manutenção de qualquer Vertedouro. Em sua versão atual ele poderá ser usado como um mínimo produto viável, onde usuários podem fazer uma avaliação do Sistema

e darem sugestões de novas funcionalidades. Com o refinamento das atuais funcionalidades e o desenvolvimento de módulos como de relatórios e simulações o SIGVERTE tem potencial para se tornar uns dos melhores em seu domínio assim como o SSB e o SysDAM.

7.2 Trabalhos Futuros

A ênfase deste trabalho de conclusão de curso foi nas questões teóricas e metodológicas relacionadas à engenharia de requisitos de software do SIGVERTE, e para cumprir o proposto foram realizadas todas as atividades elencadas nos objetivos específicos do trabalho, inclusive a implementação do primeiro módulo do Sistema. Contudo, são várias as melhorias que devem ainda ser realizadas a tal módulo, como verificado na fase de validação da conformidade do Sistema com seus requisitos. Exemplos são o aprimoramento do critério de distribuição de pontos, pois atualmente eles são distribuídos aleatoriamente no caso da Campanha. Sugere-se a disponibilização de diferentes critérios para essa distribuição, viabilizando que o usuário selecione o critério de seu interesse. Também se sugere desenvolver o requisito de fila de remoção, permitindo que qualquer elemento que for removido durante a utilização do Sistema permaneça em uma fila de espera visível por qualquer usuário por determinado período. Após, passado o período, a remoção se efetiva. Deve ser implementado o Padrão de Projeto Observer assim que for desenvolvida a funcionalidade que registra as ações dos usuários. Deve também, ser implementada funções que usarão inserção de imagens, como por exemplo o Croqui de Frente de Concretagem. E se sugere desenvolver a inserção, remoção ou modificação de um ou mais pontos, segundo determinado critério, viabilizando flexibilidade adicional às especificas funcionalidades do Sistema. Também se verificou que é relevante atualizar a versão utilizada do OpenLayers, de 2 para a 3, pois provê melhor suporte às diversas funcionalidades, a exemplo, da visualização e de ações correlatas. Por fim, é imprescindível ampliar os testes de aceitação para uma quantidade maior de stakeholders. Tais aspectos já identificados e relacionados tais melhorias e aprimoramentos serão repassados detalhadamente aos bolsistas integrantes do Projeto para efetivar a transferência de know-how.

Referências Bibliográficas

[1] BINACIONAL, I. *Itaipu supera o próprio recorde mundial de produção de energia*. 2016. Consultado na INTERNET: https://www.itaipu.gov.br/sala-de-imprensa, 2016.

[2] GAMMA RICHARD HELM, R. J. E.; VLISSIDES, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. [S.l.]: Addison-Wesley Professional, 1995.

[3] PALMER, S. R.; FELSING, J. M. *A Practical Guide to Feature Driven Development*. [S.l.]: Prentice Hall, 2002.

[4] BECK, K.; ANDRES, C. *Extreme Programming Explained: Embrace Change*. [S.l.]: Addison-Wesley Professional, 2004.

[5] GERMOGLIO, G. *Arquitetura de Software*. 2015. Consultado na INTERNET: http://cnx.org/content/col107022/1.9/, 2015.

[6] POSTGRESQL. *PostgreSQL*. 2015. Consultado na INTERNET: http://www.postgresql.org, 2015.

[7] POSTGIS. *PostGis*. 2015. Consultado na INTERNET: http://postgis.net/, 2015.

[8] PHOTOMODELER. *PhotoModeler*. 2015. Consultado na INTERNET: http://www.photomodeler.com, 2015.

[9] JAVA. *Java*. 2015. Consultado na INTERNET: http://www.oracle.com/technetwork/pt/java/index.html, 2015.

[10] JAVASCRIPT. *JavaScript*. 2015. Consultado na INTERNET: https://www.javascript.com/, 2015.

[11] HTML5. *HTML5*. 2015. Consultado na INTERNET: http://www.w3.org/TR/html5/, 2015.

[12] CSS. *Css*. 2015. Consultado na INTERNET: http://www.w3.org/TR/CSS/, 2015.

[13] BOOTSTRAP, T. *BOOTSTRAP*. 2015. Consultado na INTERNET: http://getbootstrap, 2015.

[14] PLAYFRAMEWORK. *PlayFramework*. 2015. Consultado na INTERNET: http://playframework.com, 2015.

[15] OPENLAYERS. *OpenLayers*. 2015. Consultado na INTERNET: http://openlayers.org/.org, 2015.

[16] GEOJSON. *GeoJson*. 2015. Consultado na INTERNET: http://geojson.org/, 2015.

[17] SSB. *SSB*. 2015. Consultado na INTERNET: http://www.furnas.com.br/arqtrab/ddppg/revistaonline/lir

[18] SYSDAM. *SysDAM*. 2015. Consultado na INTERNET: http://www.pimentadeavila.com.br/index.php?option=com$_{c}ontentview = articleid = 132 : seguranca - de - barragens - e - sistemas - de - gestao - de - riscoscatid = 46 : area - de - atuacaoItemid = 122, 2015$.

[19] NEBULON. *Feature Driven Development Overview*. 2005. Consultado na INTERNET: http://www.nebulon.com/articles/fdd/download/fddoverview.pdf, 2015.

[20] NEBULON. *FDD Processes*. 2016. Consultado na INTERNET: http://www.featuredrivendevelopment.com/files/fddprocessesA4.pdf, 2016.

[21] SOMMERVILLE, I. *Engenharia de Software*. [S.l.]: Pearson Prentice Hall, 2011.

[22] WINDOWS, M. *Internet Explorer*. 2016. Consultado na INTERNET: http://windows.microsoft.com/pt-br/internet-explorer/download-ie, 2016.

[23] GOOGLE. *Google Chrome*. 2016. Consultado na INTERNET: https://www.google.com.br/chrome/browser/desktop/, 2016.

[24] FIREFOX. *Firefox*. 2016. Consultado na INTERNET: https://www.mozilla.org/pt-BR/firefox/new/, 2016.

[25] ASTAH. *Astah Community*. 2016. Consultado na INTERNET: http://astah.net/editions/community, 2016.

[26] ACADEMY, C. *Code Academy*. 2016. Consultado na INTERNET: https://www.codecademy.com/pt, 2016.