

WEBFAZENDA - UM SISTEMA INTEGRADO PARA
GERENCIAMENTO DE AGRICULTURA

ALAN CARVALHO GALANTE

UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE
DARCY RIBEIRO

CAMPOS DOS GOYTACAZES – RJ
MAIO – 2014

WEBFAZENDA - UM SISTEMA INTEGRADO PARA GERENCIAMENTO DE AGRICULTURA

ALAN CARVALHO GALANTE

Tese apresentada ao Centro de Ciências e Tecnologias Agropecuárias da Universidade Estadual do Norte Fluminense Darcy Ribeiro, como parte das exigências para obtenção do título de Doutor em Produção Vegetal, com ênfase em mecanização agrícola.

Orientador: Ricardo Ferreira Garcia

CAMPOS DOS GOYTACAZES – RJ

MAIO – 2014

FICHA CATALOGRÁFICA

Preparada pela Biblioteca do **CCTA / UENF** 068/2014

Galante, Alan Carvalho

Webfazenda – um sistema integrado para gerenciamento de agricultura / Alan Carvalho Galante. – 2014.

109 f. : il.

Orientador: Ricardo Ferreira Garcia

Tese (Doutorado - Produção Vegetal) – Universidade Estadual do Norte Fluminense Darcy Ribeiro, Centro de Ciências e Tecnologias Agropecuárias. Campos dos Goytacazes, RJ, 2014.

Bibliografia: f. 91 – 95.

1. Sensores 2. Arduino 3. Mapa térmico 4. Sistema de aquisição de dados 5. GPS I. Universidade Estadual do Norte Fluminense Darcy Ribeiro. Centro de Ciências e Tecnologias Agropecuárias. II. Título.

CDD – 630

WEBFAZENDA - UM SISTEMA INTEGRADO PARA GERENCIAMENTO DE AGRICULTURA

ALAN CARVALHO GALANTE

Tese apresentada ao Centro de Ciências e Tecnologias Agropecuárias da Universidade Estadual do Norte Fluminense Darcy Ribeiro, como parte das exigências para obtenção do título de Doutor em Produção Vegetal, com ênfase em mecanização agrícola.

Aprovada em 30 de maio de 2014.

Comissão Examinadora

Prof. Jorge Luís Nunes e Silva Brito (PhD, Mapping and GIS) – UERJ

Prof. Elias Fernandes de Sousa (D.Sc., Produção Vegetal) – UENF

Prof. Jose Tarcisio Lima Thiebaut (D.Sc., Produção Animal) – UENF

Prof. Ricardo Ferreira Garcia (D.Sc., Engenharia Agrícola) – UENF
(Orientador)

Dedico aos meus pais, Joel Galante e Rute Galante, por sempre investirem nos meus estudos, acima de tudo na vida e por mostrarem tudo o que sou capaz de fazer. À minha esposa Juliana Galante, por estar ao meu lado sempre me apoiando. À minha irmã Aline Galante, por estar sempre me ajudando, apoiando e incentivando.

AGRADECIMENTOS

Agradeço primeiramente a Deus por permitir que tudo isso seja possível;

A Universidade Estadual do Norte Fluminense Darcy Ribeiro e ao Laboratório de Engenharia Agrícola, pela oportunidade de realização deste curso;

Ao professor Ricardo Ferreira Garcia, pela orientação e confiança durante todas as etapas deste trabalho;

A meus pais, minha irmã e meu irmão, pelo amor, pela compreensão, pelo apoio e pela ajuda nas horas difíceis;

À minha esposa, sempre companheira fazendo parte de tudo isso com apoio, sofrimento e dificuldades;

Aos meus filhos, Letícia e Alan Filho, pois, mesmo que muito novos, nos momentos mais difíceis era o sorriso deles que me fazia ficar calmo e chegar aos objetivos;

A todos aqueles que estiveram comigo, durante esta longa jornada, os meus mais sinceros agradecimentos.

SUMÁRIO

1. INTRODUÇÃO	1
2. REVISÃO DE LITERATURA	4
2.1. Mecanização na agricultura e a agricultura de precisão.....	4
2.2. Sistema de informação geográfica (SIG).....	6
2.2.1. Sistemas de Coordenadas	8
2.2.2. Consórcio “Open GIS”	9
2.2.3. Especificações OGC	13
2.3. Servidores de Mapa.....	14
2.4. Sensoriamento	17
2.4.1 Sistema de Aquisição de Dados	20
2.5. Arduíno	20
2.6. Global Position System – GPS	24
2.6.1. O Padrão NMEA 0183	25
2.7. Banco de Dados	26
2.7.1. PostgreSQL	27
2.7.2. Banco de Dados Geográficos	28
2.7.3. PostGIS	28

2.8. UML (Unified Modeling Language – Linguagem de Modelagem Unificada)	29
2.8.1. Diagrama de Caso de Uso	30
2.8.2. Diagrama de Classes	31
2.8.3. Diagrama de Atividades	32
2.9. Mapeamento objeto relacional - Hibernate	33
2.9.1. Hibernate Espacial	34
2.10. Java Server Faces (JSF)	35
2.11. OpenLayers	36
2.12. Trabalhos Relacionados	36
3. MATERIAL E MÉTODOS	39
3.1. Componentes Físicos	39
3.1.1. O Módulo de conexão fixa	39
3.1.2. O Módulo conectável	43
3.2. WebFazenda	47
3.2.1. Tecnologias de Servidores	48
3.2.2. Tecnologias de Programação	49
3.2.3. As funcionalidades do WebFazenda	57
4. RESULTADOS E DISCUSSÃO	68
4.1. Funcionamento do WebFazenda	68
4.2. Funcionamento do Sistema de Aquisição de Dados	85
5. RESUMO E CONCLUSÕES	89
REFERÊNCIAS BIBLIOGRÁFICAS	91
APÊNDICE A - MAPEAMENTO DAS CLASSES USANDO O HIBERNATE E O HIBERNATE ESPACIAL	96

LISTA DE QUADROS

Quadro 1 - Exemplos de Sistemas de Coordenadas	8
Quadro 2 - Comparativos entre os arduínos comuns no mercado.....	22
Quadro 3 - Códigos do NMEA.....	26
Quadro 4 - Formato da saída do cartão de memória do módulo fixo	43
Quadro 5 - Produtividade por repetição da área experimental.....	74

LISTA DE FIGURAS

Figura 1. Hierarquia de Classes Geométricas – <i>OpenGIS</i>	10
Figura 2. Exemplos de Polígonos.....	11
Figura 3. Exemplos de Não-Polígonos.....	11
Figura 4. Exemplos de <i>LineString</i>	12
Figura 5. Exemplos de <i>MultiLineString</i>	12
Figura 6. Esquema representando um servidor de mapas.....	15
Figura 7. Anatomia de uma aplicação MapServer.....	16
Figura 8. Arquitetura do GeoServer	17
Figura 9. Sistema de Aquisição de Dados	20
Figura 10. Arduíno Duemilanove e UNO.....	21
Figura 11. Shield de GPS e Shield de motor de passo	23
Figura 12. Exemplo de programação do Arduíno.....	24
Figura 13. Tipos de dados espaciais do PostGIS.....	29
Figura 14. Exemplo de Diagrama de Caso de Uso	31
Figura 15. Exemplo de Diagrama de Classes	32
Figura 16. Exemplo de Diagrama de Atividades	33
Figura 17. Exemplo de mapa renderizado em OpenLayers	36
Figura 18. Módulo de Aquisição de Dados com arduíno.....	40

Figura 19. Fluxograma do programa do módulo fixo do arduíno.....	42
Figura 20. Arduíno UNO.....	44
Figura 21. Sistema de Aquisição de Dados – Leitura de Temperatura	45
Figura 22. Programação do Sistema de Aquisição de Dados – Leitura de Temperatura.....	46
Figura 23. Fluxograma Completo do Programa do Módulo de Aquisição de Dados	47
Figura 24. Distribuição de Tecnologias e Equipamentos para o WebFazenda	48
Figura 25. Diagrama de Classes do WebFazenda.....	49
Figura 26. Tabelas do Banco de Dados	50
Figura 27. Estilo usado para ponto de cor amarela.....	52
Figura 28. Definição do estilo usado para ponto com os valores lidos	53
Figura 29. Definição do estilo PoligonoRegiao no GeoServer	54
Figura 30. Definição no GeoServer das de cores para o mapa térmico	55
Figura 31. Exemplo de estilos de mapas criados no GeoServer.....	56
Figura 32. Diagrama de Caso de Uso – WebFazenda.....	58
Figura 33. Fluxograma de criação de uma região no WebFazenda.....	59
Figura 34. Fluxograma de criação de uma região no WebFazenda.....	61
Figura 35. Fluxograma de importação de leitura de dados de laboratório do campo no WebFazenda.	63
Figura 36. Telas necessárias para a inclusão de dados laboratoriais no WebFazenda.....	64
Figura 37. Telas de Visualização de Região no WebFazenda.....	65
Figura 38. Telas de Visualização de Sensores no WebFazenda	66
Figura 39. Telas de Visualização de Dados de Laboratório no WebFazenda.....	67
Figura 40. Área experimental destacada em vermelho, localizada na UAP da UENF, Campos dos Goytacazes	69
Figura 41. Arquivo NMEA lido a partir do GPS Garmin GPSmap 60CSx com a área experimental	70
Figura 42. Importação do Arquivo do arquivo NMEA da Região no WebFazenda	71
Figura 43. Visualização do mapa da região UAP pelo WebFazenda	72
Figura 44. Croqui das áreas georeferenciadas	73
Figura 45. DATALOG.TXT com dados da Produtividade	74

Figura 46. Cadastro de Sensores para Importação de Dados	75
Figura 47. Mapa de Pontos colhidos pelos sensores	75
Figura 48. Mapa térmico	76
Figura 49. Mapa de Produtividade gerado pelo Surfer 8.0 método de krigagem simples.....	77
Figura 50. Mapa de Produtividade gerado pelo WebFazenda com zoom no centroide de uma das áreas experimentais da área tecnificada	78
Figura 51. Mapa de Produtividade gerado pelo WebFazenda com zoom no centroide de uma das áreas experimentais da área não tecnificada	78
Figura 52. Cadastros dos dados de Análise Laboratorial.....	80
Figura 53. Importação das localizações de coleta dos dados de Análise Laboratorial	81
Figura 54. Resultados obtidos das amostras do solo das áreas tecnificadas	82
Figura 55. Resultados obtidos das amostras do solo das áreas não tecnificadas	83
Figura 56. Mapa térmico do ISNa.....	84
Figura 57. Mapa térmico do Mn.....	85
Figura 58. Mapa de região gerado através da importação do arquivo NMEA criado pelo programa NMEA GPS de um celular com Android.....	86
Figura 59. Mapa térmico da região gerado através da importação do arquivo DATALOG.TXT gerado pelo sistema de aquisição de dados de temperatura	87

RESUMO

GALANTE, Alan Carvalho, Universidade Estadual do Norte Fluminense Darcy Ribeiro. Maio de 2014. **WEBFAZENDA - UM SISTEMA INTEGRADO PARA GERENCIAMENTO DE AGRICULTURA**. Orientador: Ricardo Ferreira Garcia.

O presente trabalho refere-se ao desenvolvimento de uma solução para o gerenciamento da produção agrícola, desde os sensores no campo até o programa que gerencia os mesmos. Para os sensores é apresentado um sistema de aquisição de dados usando o Arduíno com um GPS acoplado ao mesmo e um programa que faz leituras do GPS e do sensor que estiver conectado e armazenado em um cartão de memória. Os dados lidos são exportados em programa, denominado WebFazenda, que lê e converte os dados para valores não geográficos e valores geográficos, armazena e processa em programas como PostgreSQL, Postgis e GeoServer. O programa, além dos dados de sensores, permite a inclusão de dados vindos de laboratórios e, com uma interface amigável, disponibiliza um mecanismo gráfico para a inclusão desses dados. Ao término, o programa permite que sejam vistos os dados em mapas de dois tipos: ponto e térmicos. Para a validação do sistema, foram utilizados os dados de GPS, de campo e de laboratório de um projeto realizado em 2009 na UAP da UENF. Os mesmos foram importados através da própria interface do WebFazenda e, ao final, foram gerados mapas térmicos. Os resultados obtidos foram semelhantes

aos resultados dos mapas de produtividade apresentados pelo projeto que havia sido realizado em 2009 e que foram construídos pelo programa Surfer 8.0. Para validar o sistema de aquisição de dados, foi construído um componente físico baseado em um sensor de temperatura. As temperaturas foram coletadas em uma região previamente mapeada através de um celular com GPS com o sistema operacional Android. Os dados foram importados no WebFazenda, que gerou os mapas térmicos mostrando que as áreas em que o sensor ficou exposto ao sol apresentou maiores temperaturas do que as áreas em que o sensor ficou na sombra.

ABSTRACT

GALANTE, Alan Carvalho, Universidade Estadual do Norte Fluminense Darcy Ribeiro. 2014, may. WEBFAZENDA - AN INTEGRATED SYSTEM FOR AGRICULTURE MANAGEMENT. Advisor: Ricardo Ferreira Garcia.

The project is about the development of a solution for the management of agricultural production from the sensors in the field to the program that manages them. For the sensors a data acquisition hardware using an Arduino with a GPS connected to it and a program that reads the GPS and the sensor and stores in a memory card. The scanned data is exported to the WebFazenda, which reads and converts the data values to non-geographic and geographic values, store and processes in some programs like PostgreSQL, PostGIS and GeoServer. The program, in addition to the sensor data, allows the inclusion of data coming from laboratories and with a friendly interface, provides a graphical mechanism for the inclusion of such information. In the end, the program allows the visualization of data on maps of two types: point and Heatmap. To validate the system, were used the GPS data, field and laboratory data from a project conducted in 2009 at the UENF UAP. They were imported by the WebFazenda interface and at the end, heatmaps were generated. The results were similar to the results of yield maps of

the 2009's project which were built by Surfer 8.0 program. To validate the data acquisition system was constructed a physical component based on a temperature sensor. The temperatures were collected in a previously mapped region through a cell phone with GPS and Android operating system. All data were imported into WebFazenda that generated heatmap showing the areas in which the sensor was exposed to sun showed higher temperatures than the areas in which the sensor was in regions of shadow.

1. INTRODUÇÃO

A mecanização e a automação da produção agrícola têm crescido de importância ao longo do tempo para alcançar maior produtividade e, ao mesmo tempo, reduzir os custos de produção no campo.

A tecnologia é usada em ampla escala, em todas as etapas e em todas as áreas dos processos produtivos, tais como: dispositivos para leitura de dados; equipamentos que processam os dados lidos e os apresentam através de mapas, tabelas e outras técnicas que auxiliem o administrador na tomada de decisão; equipamentos que auxiliam diretamente na fertilização, na irrigação, no plantio, na colheita e no armazenamento; entre outros.

Com tudo isso, os mais variados tipos de fabricantes têm disponibilizado uma grande quantidade de equipamentos e programas capazes de solucionar com qualidade os mais diversos problemas. Porém, isso gera algumas implicações métricas tais como: custo, interoperabilidade e conhecimento avançado do usuário.

Em termos de custo, tanto os equipamentos quanto os programas disponibilizados apresentam custos elevados. Os custos, de alguma forma, são justificados a partir do momento que o retorno é garantido tanto em produtividade

quanto em lucratividade. Mas, esse problema acaba distanciando pequenos e, até mesmo, médios produtores.

Quanto à interoperabilidade, a comunicação entre os diversos programas e equipamentos ainda hoje é um grande problema. Normalmente essa comunicação só é possível quando os dispositivos e os programas de computadores são de um mesmo fabricante, o que, em geral, é impraticável ao usuário e como consequência deste fato, surge o problema da interoperabilidade, ou seja, quando o agricultor decide o uso de dispositivos/programas de fabricantes diferentes, isso leva a ter que digitar os resultados em algum outro programa para tratar os dados em conjunto ou a utilizar uma série de recursos não convencionais para tratar esses dados. Esse fato, além do desconforto e dos problemas já apresentados, pode levar à perda de dados e/ou de sua precisão e, até mesmo, ao problema do “conhecimento tecnológico avançado do usuário”.

A necessidade do conhecimento tecnológico avançado do usuário ocorre, pois são tantos os problemas por que passa o administrador para manipular toda essa tecnologia de mundos diferentes em “linguagens” diferentes que, para conseguir de forma ideal extrair as informações que necessita, acaba tendo que ser um especialista em tecnologia da informação. Essa tecnologia é levada ao extremo, pois o produtor agrícola tem que conhecer equipamentos diferentes, programas diferentes, planilhas eletrônicas, editores de texto, editores de imagem, e assim por diante.

Este trabalho desenvolve um procedimento automatizado para o gerenciamento da produção de uma fazenda mecanizada, independente da adoção das práticas da agricultura de precisão. Esta pesquisa viabiliza a utilização de mecanismos de baixo custo para instalar e configurar os sensores no campo, bem como disponibiliza um software que facilita a inserção dos dados de campo de qualquer sensor usado com o devido dado geográfico associado ao mesmo, assim como mostra os dados gerados através de mapas que facilitam a interpretação do que está acontecendo na produção para que o administrador possa tomar as decisões que julga mais adequadas à correção ou à solução de problemas relacionados à produção.

Torna-se importante para os atuais e futuros usuários de agricultura mecanizada, a criação de um mecanismo facilitador do uso de toda a tecnologia que envolve a área. Esse facilitador envolve o uso de equipamentos, programas

de computadores, técnicas de geoprocessamento, utilização de sensores eletrônicos, entre outros, que sirvam de suporte e, ao mesmo tempo, deixem o mais transparente possível, a complexidade que todas essas tecnologias representam para o usuário, ajudando os mais diversos tipos de agricultores, desde os grandes até os pequenos e familiares.

2. REVISÃO DE LITERATURA

2.1. Mecanização na agricultura e a agricultura de precisão

A mecanização na agricultura tem como objetivo o emprego adequado de equipamentos, máquinas agrícolas e sensores, visando à sua otimização e à viabilidade da obtenção de altas produtividades agropecuárias, com a racionalização dos custos e a preservação dos recursos naturais e do meio ambiente (MERCALDI, 2012).

Preparo de solo, semeadura, aplicações de defensivos agrícolas, fertilizantes e corretivos e colheita são as operações mecanizadas realizadas ao longo do ciclo de produção, possuindo cada uma delas suas particularidades, quer seja pelas condições de clima e solo, quer seja pelo perfil do produtor e da escala de produção (MERCALDI, 2012).

Segundo Souza (2001), o uso de sensores e máquinas no auxílio desses processos é cada vez mais comum e necessário na produção agrícola. Por trás dessas tecnologias vêm os programas de computadores que são capazes de ler os dados vindos do campo e traduzi-los para que o usuário possa tomar suas decisões relacionadas à produção agrícola. Esses dados podem ser fornecidos ao usuário em formato tabular ou em formato de mapas.

Tem-se observado uma concorrência muito grande da indústria de máquinas agrícolas, dos fornecedores de sistemas eletrônicos e hidráulicos e de programas computacionais de toda ordem. Mas, com um mercado imediatista e segmentado, tem-se gerado produtos que não atendem à demanda do agricultor. Por outro lado, a falta de definições de direção do mercado em relação ao nível tecnológico a se adotar tem levado muitos empreendimentos a serem revistos ou mesmo encerrados.

Situada a um passo adiante da mecanização agrícola tem-se a agricultura de precisão, que permite que se faça, em áreas extensas, o que os pequenos agricultores fazem, que é o tratamento dos detalhes considerando as diferenças existentes em um talhão.

A ideia básica é que o agricultor possa, inicialmente, identificar as áreas de alta e de baixa produtividade dos talhões para depois, administrar essas diferenças. Para que isso seja possível é necessário certo grau de automatização, que depende de tecnologias modernas, tais como: GPS (*Global Position System* – sistema de posicionamento global), informática, sensores e controladores utilizados no campo e nas máquinas agrícolas.

A agricultura de precisão não pode ser considerada somente como uma forma de aplicar tratamentos que variam de local para local, mas deve ser considerada como a habilidade de monitorar e acessar a atividade agrícola, precisamente. Ela permite que o administrador tenha um melhor entendimento sobre o manejo e controle dos sistemas de tratamento dos campos de produção.

De acordo com Manzatto et al. (1999), o principal conceito é aplicar os insumos no local correto, no momento adequado, nas quantidades de insumos necessárias à produção agrícola, para áreas cada vez menores e homogêneas, tanto quanto a tecnologia e os custos envolvidos assim o permitam.

Para Miranda (2011), a agricultura de precisão envolve um complexo processo, cujo fundamento é o conhecimento espacial preciso da atividade agrícola, frequentemente baseado no uso de dados obtidos com auxílio de satélites e vendedores de insumos, sob argumentos de eficiência, que veem nessa proposta uma nova oportunidade de venda de equipamentos e produtos.

Molin (2011) afirma que produtos importantes para a agricultura de precisão como monitores de produtividade, controladores de taxa variada, receptores GPS possuem custos elevados se tornando um entrave da adoção

dessas ferramentas. O custo tem sido um grande problema, além da falta de parâmetros para balizar o benefício.

Desta forma, a agricultura de precisão engloba o uso de tecnologias atuais de modo adequado às variações espaciais e temporais em fatores que afetam a produtividade das mesmas.

2.2. Sistema de informação geográfica (SIG)

Rocha (2000) define SIG como “um sistema com capacidade para aquisição, armazenamento, tratamento, integração, processamento, recuperação, transformação, manipulação, modelagem, atualização, análise e exibição de informações digitais georeferenciadas, topologicamente estruturadas, associadas ou não a um banco de dados alfanuméricos”.

Assim sendo, o SIG utiliza uma série de técnicas e disciplinas como geografia, cartografia, sensoriamento remoto, fotogrametria, modelagem de dados, entre outras.

Um Sistema de Informação Geográfica (SIG) trabalha com dois tipos de dados: dados convencionais (número, texto, booleanos, etc) e dados espaciais (geometria e relacionamentos topológicos das entidades geográficas). Os dados geográficos podem ser representados em dois formatos: vetorial, que representa objetos geométricos por intermédio de pontos, linhas, e polígonos; matricial (ou raster), que se caracteriza por uma matriz de pontos, onde cada ponto representa o valor de um atributo fornecido por uma localização geográfica do mundo real (ARONOFF, 1995).

Em um SIG, os dados são organizados em camadas ou planos de informação. Aronoff (1995) conceituou camada de dados como um conjunto de características e atributos geográficos relacionados logicamente. Deste modo, podem-se encontrar camadas do tipo rodoviária, ferroviária, hidrográfica, distritos, bairros, ruas, entre outros.

Segundo Rocha (2000), um SIG deve ter por característica: compilar e processar dados espaciais coletados de fontes diversas; armazenar, recuperar, atualizar e corrigir os dados; permitir manipulação e realização de procedimentos de análise dos dados armazenados; controlar a exibição e saída de dados.

Para Navathe & Elmasri (2000) um SIG deve ser capaz de realizar as seguintes operações espaciais: interpolação, interpretação, análise de proximidade, processamento de imagens e análises de rede.

Para ARONOFF (1995), interpolação é o procedimento de prever valores desconhecidos usando valores conhecidos em localizações vizinhas. Um exemplo seria a reconstituição de pontos perdidos através dos pontos existentes.

A edição se preocupa com operações sobre terrenos tais como: suavização, redução de detalhes, aumento, realce de bordas de feições do terreno e fusão (ARONOFF, 1995).

A análise de proximidade se refere aos algoritmos que realizam operações de distância entre objetos (ROCHA, 2000).

Quanto ao processamento de dados modelados matricialmente, podem-se distinguir duas categorias: álgebra de mapas, onde é possível a integração de características geográficas em camadas de mapas diferentes para produzir um novo mapa; e análise de imagens, que analisa uma imagem digital para a descoberta de características ou extração de objetos geográficos (ARONOFF, 1995).

De acordo com ARONOFF (1995), as análises de redes são normalmente usadas para mover recursos de um local para outro. As ruas de uma cidade, uma malha de linhas de transmissão elétrica, uma rota de serviço aéreo, ou os fluxos de drenagem de uma bacia são exemplos de redes.

Segundo Dias et al. (2002), além da percepção visual da distribuição espacial do problema, é muito útil traduzir os padrões existentes com considerações objetivas e mensuráveis. Para isso, podem ser modeladas ou determinadas as seguintes visões:

- Epidemiológicas: distribuição da doença por um espaço geográfico;
- Policiais: distribuição espacial de crimes;
- Geológicas: estimativa e estudo de uma reserva de algum recurso mineral;
- Agrícola: distribuição de cultura de acordo com tipo de solo, vegetação, relevo, entre outros.

Um dos fatores com que se deve ter muita cautela no tratamento das informações, segundo Dias et al., 2002, é a preservação dos dados individuais.

Para resolver esse problema usa-se a agregação dos dados, neste caso, dados geográficos.

2.2.1. Sistemas de Coordenadas

Os sistemas de coordenadas geográficas são formados pelos meridianos e paralelos, onde meridianos são círculos máximos que contêm o eixo dos pólos, e os paralelos são círculos mínimos perpendiculares aos eixos dos polos (KRRAK & ORMELING, 2010).

Os meridianos determinam as longitudes, valores medidos em graus que variam de 0° a 180°, a partir do meridiano de Green wich, tanto para leste quanto para o oeste. Para diferenciar os valores, os graus são acrescidos do sinal negativo a oeste (KRRAK & ORMELING, 2010).

Os paralelos determinam as latitudes, valores medidos em graus que variam de 0° a 90°, a partir da linha do Equador, tanto para o sul quanto para o norte. Para diferenciar os valores, os graus são acrescidos do sinal negativo ao sul do Equador (KRRAK & ORMELING, 2010).

As latitudes e longitudes de um local determinam a localização do mesmo, com pode ser visto na tabela do quadro 1.

Quadro 1 - Exemplos de Sistemas de Coordenadas

Longitude	Latitude	Local
43,12°O	22,54°S	Rio de Janeiro
47,55°O	15,47°S	Brasília
41,19°O	21,45°S	Campos dos Goytacazes

O Sistema Universal Transverso de Mercator (UTM) é baseado na projeção cilíndrica transversa proposta nos Estados Unidos em 1950 com o objetivo de abranger todas as longitudes. Ele resulta na composição de 60 fusos distintos que representam a superfície da Terra. Cada fuso tem a amplitude de 6° de longitude.

Em latitudes o sistema UTM é limitado pelos paralelos 84° N e 80° S, onde as deformações ainda não são significativas. Para latitudes maiores, é utilizada a

projeção Estereográfica Polar Universal (sigla em inglês UPS, de *Universal Polar Stereographic*) (ARONOF, 1995).

O *datum* (do latim dado) corresponde a um modelo matemático teórico da representação da superfície da Terra ao nível do mar. Os diversos modelos matemáticos levam a diversos *datum* que existem hoje (ARONOF, 1995).

Os modelos matemáticos mais comumente utilizados para representação da superfície física do planeta Terra são os esferoides, dentre os quais se encontram as esferas propriamente ditas e as elipsoides.

A partir dos modelos matemáticos são criados os sistemas geodésicos, de utilização prática por diferentes países e, a partir dos quais, os cartógrafos constituem suas projeções cartográficas.

A partir da década de 80, com o desenvolvimento da Geodésia Espacial, ou seja, por intermédio das constelações de satélites posicionadores (o sistema GPS, por exemplo), desenvolveram-se os sistemas geodésicos globais, isto é, para aplicação em todo planeta. Dentre os “datum” globais destaca-se o WGS-84.

O Brasil adotou, a partir de 2000 o Sistema de Referência Geocêntrico para América do Sul (SIRGAS 2000), que será o sistema geodésico de referência oficialmente adotado no país. O SIRGAS 2000 adota um referencial que é um campo calculado computacionalmente no centro de massa da Terra (IBGE, 2011).

. A importância do *datum* prende-se com a necessidade de projetar um corpo curvo e a três dimensões, em um plano a duas dimensões mantendo, no entanto, os cruzamentos em ângulo reto dos meridianos e paralelos (ARONOF, 1995), a exemplo das projeções UTM.

2.2.2. Consórcio “Open GIS”

Buehler & Mckee (2003) definem o “Open GIS Consortium” (OGC) como uma organização internacional que está criando novas padronizações técnicas e comerciais para garantir interoperabilidade em SIG. Fundada em 1994 por fornecedores de programa, companhias de telecomunicações, universidades, provedores de informação e órgãos governamentais, entre outros, a OGC busca criar uma especificação de programa e novas estratégias empresariais, a fim de tornar os sistemas de geoprocessamento abertos e integrar completamente os dados geográficos e as operações necessárias para manipulá-los.

Parte dos membros da OGC tem uma visão positiva de uma infraestrutura de informação global em que dados geográficos e recursos de geoprocessamento sejam distribuídos gratuitamente. Esses recursos estão completamente integrados com as mais recentes tecnologias de computação distribuída, acessíveis para o público em geral, habilitando uma grande variedade de atividades que, atualmente, estão fora do domínio do geoprocessamento. Entre alguns dos membros desse consórcio estão empresas como a Microsoft e a Oracle (BUEHLER E MCKEE, 2003).

O OpenGIS criou uma especificação, a fim de definir um esquema padrão de armazenamento, recuperação e atualização de dados espaciais (OGC, 1999) (figura 1).

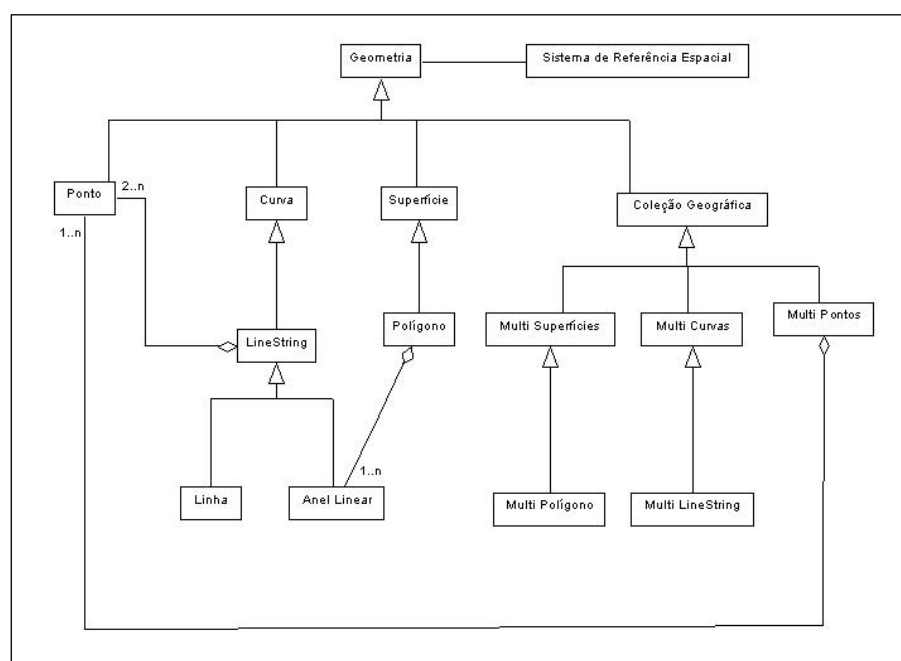


Figura 1. Hierarquia de Classes Geométricas – *OpenGIS* (OGC, 1999).

Como classes primitivas, existem as classes ponto, curva, superfície e coleção geográfica. O ponto representa uma localização geográfica única. A curva é um objeto unidimensional representado por uma sequência de pontos. A superfície é um objeto geográfico bidimensional (OGC, 1999).

Um polígono é uma superfície plana, definida por um limite exterior e um ou mais limites interiores. Cada limite interior define um buraco no polígono (OGC, 1999).

A figura 2 representa exemplos de polígonos. No primeiro exemplo, o polígono só possui o limite externo; no segundo caso existe o limite externo e um

limite interno (com um único buraco); e no terceiro exemplo, além do limite externo, existem dois limites internos (dois buracos) (OGC, 1999).

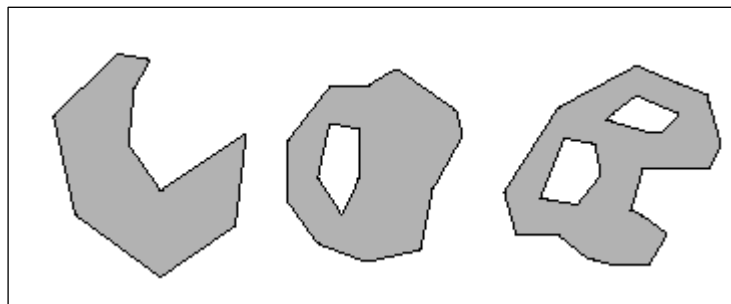


Figura 2. Exemplos de Polígonos (OGC, 1999).

A figura 3 exibe figuras que não são consideradas pelo OGC como polígonos simples. O primeiro e o quarto exemplos representam, na verdade, dois polígonos distintos. O segundo exemplo não consiste um polígono, uma vez que existe uma linha que liga dois de seus vértices. No terceiro exemplo, existem pontos de um limite externo do polígono que não estão presentes no polígono (OGC, 1999).

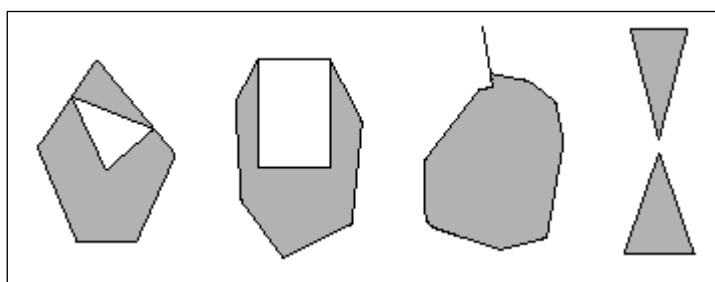


Figura 3. Exemplos de Não-Polígonos (OGC, 1999).

A curva é um objeto unidimensional geralmente armazenado com uma sequência de pontos. O *LineString* é uma curva com interpolação linear entre pontos. O *LineString* possui dois tipos: linha e anel linear, conforme pode ser visto na figura 4 (OGC, 1999).

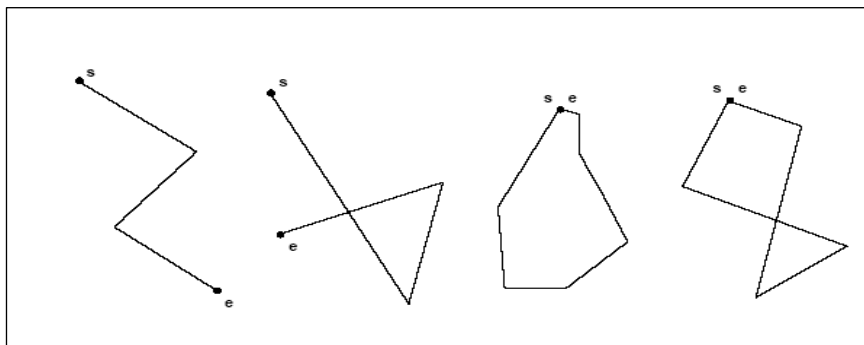


Figura 4. Exemplos de *LineString* (OGC, 1999).

A Linha é um *LineString* com exatamente dois pontos. O Anel Linear é um *LineString* fechado e simples. Entende-se como fechada a curva que possui o ponto inicial igual ao ponto final. A característica de simplicidade se dá quando não existe passagem pelo mesmo ponto duas vezes (OGC, 1999).

O Multiponto é uma coleção geométrica zero dimensional. Os pontos não são conectados ou ordenados. A multicurva é uma coleção unidimensional de curvas. O *MultiLineString* (figura 5) é uma *MultiCurva* formada por *LineString*. A MultiSSuperfície é uma coleção geométrica bidimensional de superfícies. Os interiores de qualquer duas superfícies em uma coleção MultiSSuperfície podem possuir interseção. O MultiPolígono é uma MultiSSuperfície formada por polígonos (OGC, 1999).

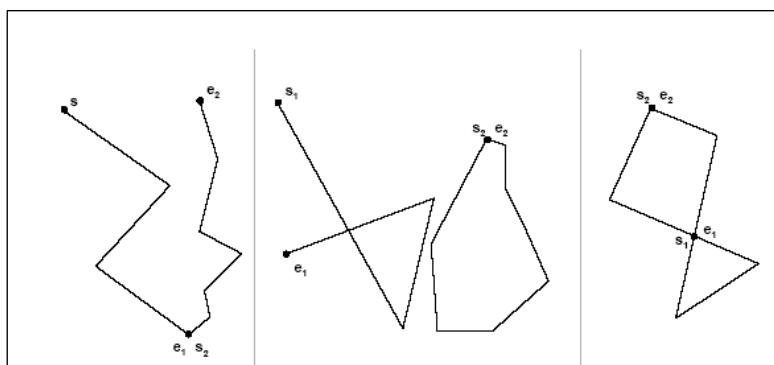


Figura 5. Exemplos de *MultiLineString* (OGC, 1999).

É importante lembrar que o Consórcio *Open GIS* define como tipo de dados instanciáveis apenas: ponto, linha, anel linear, polígono, multipolígono, multilinestring e multiponto. Serão esses os tipos geográficos que serão tratados no modelo de dados multidimensional deste trabalho (OGC, 1999).

2.2.3. Especificações OGC

Para entender melhor as especificações da OGC é necessário entender um pouco sobre XML (*eXtensible Markup Language*). O XML é semelhante ao HTML com a diferença de ser uma linguagem extensível, ou seja, não existe um conjunto predeterminado de marcadores para serem usados com ele. Os marcadores são criados de acordo com a necessidade do desenvolvedor. O XML existe para permitir o transporte e armazenamento de dados de forma aberta, pois pode ser aberto por um simples editor de texto por ser em texto simples (OGC, 2010).

O GML (*Geographic Markup Language*) é uma especificação OGC que permite codificar não somente as feições geográficas, mas também seus atributos textuais sendo ideal para armazenamento e transporte de dados vetoriais utilizando XML (OGC, 2010).

O KML (*Keyhole Markup Language*) é uma especificação para visualização de dados georeferenciados em programas que representam o globo terrestre. Ele é um XML capaz de armazenar dados georeferenciados além de como esses dados serão exibidos (OGC, 2010).

O SFS (*Simple Features Specification*) define um padrão para armazenamento, leitura, análise e atualização de “feições simples” (dados geográficos) através de uma API (*Application Program Interface*). Estas feições são baseadas em geometrias 2D com interpolação linear entre os vértices. Ela deve conter, entre outras coisas, análises espaciais/geográficas e topológicas. Este padrão já está sendo substituído pelo SFA (*Simple Feature Access*), que entre outras melhorias, prevê o tratamento de geometrias 3D (OGC, 2010).

Além do XML, é necessário o conhecimento sobre Web Services para entender as próximas especificações OGC. O Web Service é uma aplicação publicada, localizada e chamada através da internet, podendo ser acessada por outras aplicações (FREDDO & ASSMANN, 2004). Essas aplicações são consideradas clientes ou requisitoras de serviço.

O WFS (*Web Feature Service*) apresenta uma forma de acesso (inserção, atualização, exclusão e análise) à feição através do ambiente Web (HTTP). As

operações entre clientes e servidores são baseadas no formato GML (OGC, 2010).

O WMS (*Web Map Service*) define quatro protocolos que permitem a leitura de múltiplas camadas de informações georeferenciadas, contendo vetores e/ou imagens. Essa conexão permite somente consulta de dados, sendo todo o processo de renderização do mapa feito no servidor. Com isso, o cliente recebe uma imagem que corresponde a uma visualização do mapa, de acordo com as camadas (vetoriais ou matriciais) solicitadas (OGC, 2010).

O WCS (*Web Coverage Service*) é uma web service para dados de cobertura, ou seja, para representação de dados contínuos. (OGC, 2010).

O WPS (*Web Processing Service*) permite que um serviço de processamento de dados seja disponibilizado e acessado por meio de webservices. Ele não especifica quais processos podem ser implementados, e sim, um mecanismo genérico para implementar qualquer processamento de dados geoespaciais (OGC, 2010).

2.3. Servidores de Mapa

Os servidores de mapas permitem aos usuários interagirem com as informações espaciais que podem estar disponibilizadas nos mais diversos tipos e formatos (PIMENTA et al., 2012).

Segundo Pimenta et al. (2012), um servidor de mapas é capaz de receber uma solicitação do usuário, processar e devolver uma resposta em diferentes formatos (TXT, XLS, PDF, GeoTIFF, JPG, PNG, XML, KML, entre outros). O acesso às informações deve ser dinâmico para facilitar a forma de interpretação e análise dos dados pelo usuário, conforme pode ser visto na figura 6.

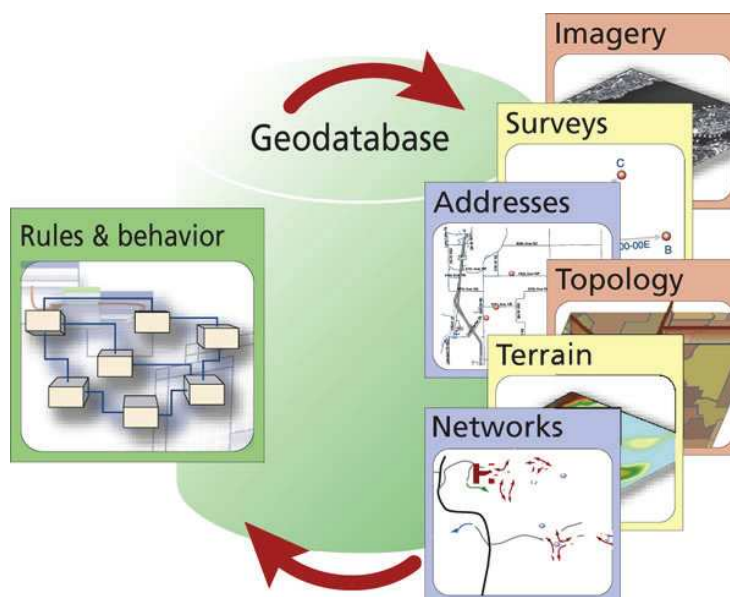


Figura 6. Esquema representando um servidor de mapas (PIMENTA et al. 2012).

Dentre os servidores de mapas disponíveis no mercado, o Geoserver e o Mapserver se destacam não só pela grande comunidade de usuários e desenvolvedores, como também por implementar em quase sua totalidade os padrões definidos pelo OpenGIS (PIMENTA et al., 2012).

Em MapServer (2013), é possível ver que suas principais características são: suporte a visualização e consulta dos diferentes tipos de formatos de raster, vetores e bancos de dados, multiplataforma, suporte a linguagens de script e outros ambientes de desenvolvimento (PHP, Python, Perl, Java, .NET), e renderização de alta qualidade, customizável e extensível. O funcionamento do MapServer pode ser visto na figura 7.

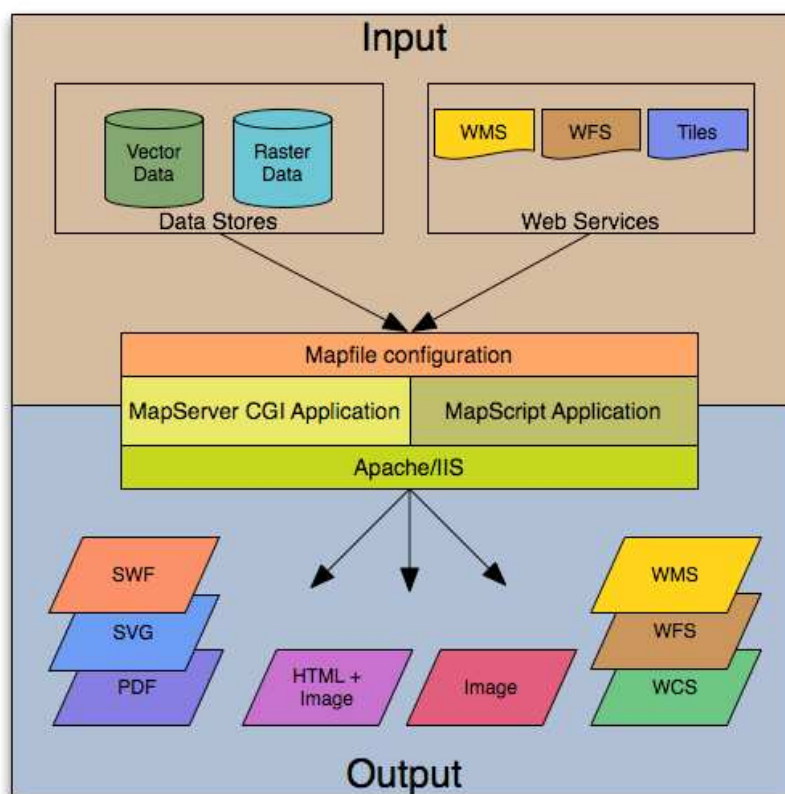


Figura 7. Anatomia de uma aplicação MapServer (MapServer, 2013).

O Geoserver é um servidor de código-fonte aberto desenvolvido sobre a plataforma Java, que possibilita aos usuários compartilhar e editar dados geográficos (GeoServer, 2013). Ele foi projetado visando à interoperabilidade entre os diversos formatos existentes, além de fornecer recursos para a publicação na Web que estão em conformidade com os padrões OpenGIS. A organização do GeoServer pode ser vista na figura 8.

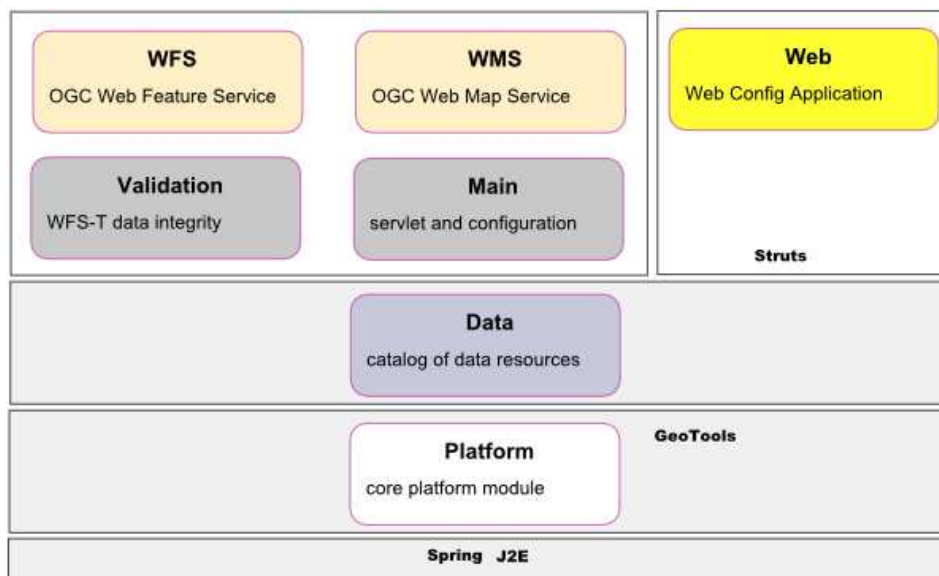


Figura 8. Arquitetura do GeoServer (GeoServer, 2013).

Em Geoserver (2013) são descritas suas principais características: Implementação de referência dos padrões OpenGIS, WFS e WCS, e WMS; totalmente configurável através de interface Web; suporte ao PostGIS, Shapefile entre outros repositórios de dados; saída dos mapas nos formatos JPEG, GIF, PNG, PDF, SVG, KML, GeoRSS.

2.4. Sensoriamento

É através dos sensores que é possível coletar as mais variadas informações do campo; que se monitoram os mais diversos equipamentos usados no cultivo/coleta; que se gerencia o funcionamento de algumas máquinas e equipamentos, entre outras funcionalidades (ROSÁRIO, 2008).

Segundo Rosário (2008), sensores são elementos provedores de informações para os sistemas de automação. Eles podem ser usados no controle de dados discretos, assim como na medição de valores lógicos ou contínuos. Um sensor é um transdutor que altera sua característica física interna devido a um fenômeno físico externo. Por exemplo: presença ou não de luz, som, gás, campo elétrico, campo magnético, etc.

De acordo com Rosário (2008), os sensores são usados em muitas áreas, tais como:

- Automação industrial – identificação de peças, medição, verificação de posição, etc.;

- Automação de escritório – leitura de código de barras, tarja magnética, impressão digital etc.;

- Automação residencial – alarmes, controle de temperatura de ambiente, controle de luminosidade, detector de presença etc.;

- Agricultura – medição de temperatura, contagem de grãos, medição de irrigação etc.

Os principais tipos de sensores são: eletromecânicos, magnéticos, indutivos, capacitivo, óptico e codificadores (ROSÁRIO, 2008).

Os sensores eletromecânicos, em geral, possuem uma alavanca que aciona um cilindro que está conectado a um contato móvel. Ele fecha ou abre o circuito. Ele pode ser usado, por exemplo, como sensor de porta, de luz de freio de carro (SOLOMAN, 2009).

Os sensores magnéticos possuem um único contato (normalmente aberto) que pode ser ativado por um campo magnético produzido por um ímã. Ele pode ser usado para medir velocidade de rotação em eixos, sistema de contagem, posição de peças, entre outros (SOLOMAN, 2009).

Os sensores indutivos, também conhecidos como sensores de proximidade, são dispositivos capazes de detectar a aproximação de componentes metálicos. É indicado para os casos que necessitem de alta velocidade de operação para detectar objetos (metálicos) (SOLOMAN, 2009).

Os sensores capacitivos baseiam-se na geração de um campo elétrico desenvolvido por um oscilador controlado por capacitor. As superfícies dos eletrodos são conectadas em uma ramificação de alimentação de um oscilador de alta frequência sintonizado de tal maneira que não oscilem quando a superfície está livre. Quando um objeto se aproxima da face ativa do sensor, ele entra no campo elétrico sob a superfície do eletrodo e causa uma mudança na capacitância do conjunto que converte em comando de chaveamento. É indicado para detecção de materiais de diversas naturezas, não condutores (SOLOMAN, 2009).

Os sensores ultrassônicos baseiam-se na emissão e reflexão de ondas sonoras entre o objeto e o receptor. O tempo de retorno do som é medido e

avaliado. É usado para detectar objetos independente de sua natureza (SOLOMAN, 2009).

Os sensores ópticos funcionam baseados em um emissor e um receptor de luz. Ele trabalha na interrupção do feixe de luz. Pode ser usado para detectar a presença ou ausência de qualquer corpo.

Segundo Capelli (2007), o codificador não é considerado um sensor, mas apenas um transdutor mecânico/elétrico. Mas, o codificador pode ser considerado um sensor de posicionamento, pois envolve deslocamento angular mecânico, reflexão óptica e conversão em sinais elétricos.

Os sensores podem ser especificados por diversos critérios, entre eles: variáveis de medida e características. As variáveis de medida representam as variações de uma grandeza física (SOLOMAN, 2010). Estas podem ser:

- analógico: é aquele que assume determinado valor compreendido dentro de uma escala. Exemplo: valor da pressão medida por um manômetro, a temperatura medida por um termômetro etc.;

- digital: assume um número finito de valores em determinada escala. Exemplo: relógio digital, um contador etc.;

- binário: é um sinal digital que pode assumir somente dois valores, 0 ou 1.

Para SOLOMAN 2010, no critério característica, os sensores podem ter os mais variados, porém os mais comuns são:

- linearidade: grau de proporcionalidade entre o sinal gerado e a grandeza física; quanto maior a linearidade, mais fiel é a resposta do sensor ao estímulo;

- faixa de atuação: intervalo de valores da grandeza em que pode ser usado o sensor, sem causar sua destruição ou a imprecisão na leitura;

- acurácia: é a razão entre o valor real e o valor medido pelo sensor;

- resolução: grandeza relacionada ao grau de precisão de leitura do sensor;

- repetibilidade: variação dos valores lidos quando uma mesma quantidade é medida várias vezes;

- domínio ou alcance (*range*): limites superior e inferior da variável a ser lida pelo sensor.

Existem os mais variados sensores no mercado. Entre eles podem-se citar os sensores para medida de posição e velocidade, sensores para medida de força e pressão, termopares, pontes extensométricas, sensores para medida da

variável de pressão, sensores para medida de aceleração e sensores de orientação e localização. (ROSÁRIO, 2008).

2.4.1 Sistema de Aquisição de Dados

Para o uso dos sensores, é necessário um sistema de aquisição de dados. Segundo Soloman 2010, para a criação de um sistema de aquisição de dados são necessários: um fenômeno do mundo real, sensores, um condicionamento de sinal, um hardware para a aquisição de dados e controle dos sensores, um sistema computacional, interfaces de comunicação e um programa, conforme pode ser visto na figura 9.

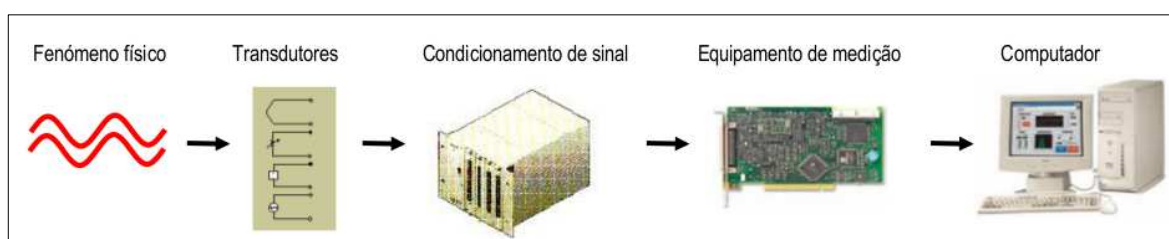


Figura 9. Sistema de Aquisição de Dados (PARK e MACKAY, 2003)

O fenômeno físico é qualquer coisa que se deseja observar na natureza ou em algum equipamento como: temperatura, penetrabilidade, etc. Transdutor é qualquer componente capaz de traduzir os fenômenos físicos em sinal elétrico/eletrônico. O condicionamento de sinal é o equipamento onde o transdutor deve ser ligado para o seu devido funcionamento. O equipamento de medição deve ser capaz de ler os dados vindos do condicionador de sinal e traduzir para os valores que representam o fenômeno para o homem. O computador é o meio de apresentação dos dados lidos pelo sistema (SOLOMAN, 2010).

2.5. Arduíno

O arduíno é um computador que se programa para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele. É uma plataforma de computação física ou embarcada, ou seja, um sistema que pode

interagir com seu ambiente por meio de hardware e programa (MCROBERTS, 2011).

Segundo McRoberts (2011), o arduino pode ser utilizado na construção de vários projetos, podendo agir de modo independente ou conectado a um computador ou ainda a uma rede de computadores.

Existem várias placas Arduino, tais como: arduino UNO (Figura 10), arduino Mega o Duemilanove Arduino (2009) (Figura 10), o arduino Severino, o arduino ADK, o arduino Leonardo (MCROBERTS, 2011). O que vai diferenciar esses arduínos é o microcontrolador e o número de portas seriais e digitais que são disponibilizadas, conforme pode ser visto no Quadro 2.

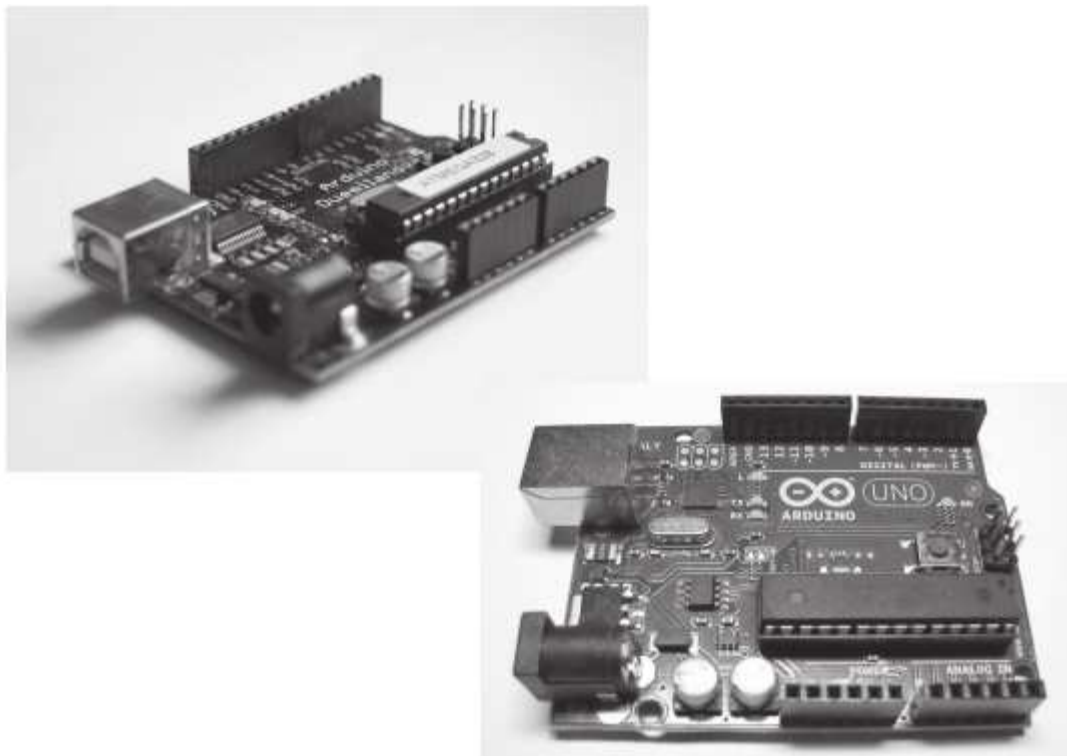


Figura 10. Arduino Duemilanove e UNO (MCROBERTS, 2011).

Quadro 2 - Comparativos entre os arduínos comuns no mercado (Adaptado de MCROBERTS, 2011)

arduíno	Microcontrolador	Portas Analógicas	Portas Digitais
UNO	ATmega328	6	14
Mega	ATmega1280	16	54
Duemilanove	ATmega168 ou ATmega328	6	14
Severino	ATMEGA168	6	14
ADK	ATmega2560	16	54
Leonardo	ATmega32U4	12	20

Um microcontrolador é um computador em um chip, contendo um processador, memória e periféricos de entrada/saída. É um microprocessador que pode ser programado para funções específicas, em contraste com outros microprocessadores de propósito geral (como os utilizados nos Pcs) (TIMMIS, 2011).

Além das entradas de alimentação, ele possui portas analógicas e digitais. Assim, é possível ligá-lo a praticamente qualquer sensor. O número de portas pode determinar quantos sensores simultaneamente um único dispositivo é capaz de gerenciar (MCROBERTS, 2011).

Segundo Timmis (2011), o arduíno também pode ser estendido utilizando os *shields* (escudos), que são placas de circuito contendo outros dispositivos (por exemplo, receptores GPS, displays de LCD, módulos de rede, entre outros), onde se pode conectar para obter funcionalidades adicionais. Os escudos também estendem os pinos até o topo de suas próprias placas de circuito, para que se continue a ter acesso a todos eles. Exemplos de escudos podem ser vistos na figura 11.

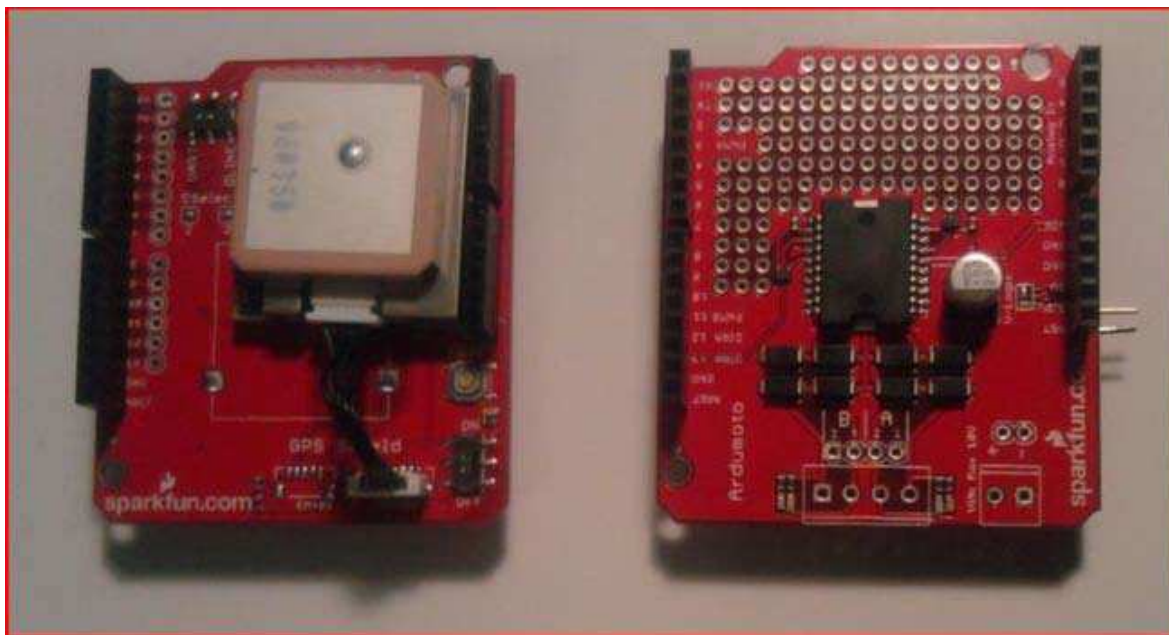


Figura 11. Shield de GPS e Shield de motor de passo (TIMMIS, 2011)

Segundo McRoberts (2011), para programar o arduino é preciso utilizar uma IDE (Integrated Development Environment) que corresponde a um ambiente integrado para desenvolvimento de programa, onde se escreve o código na linguagem que ele compreende. O software é livre e há versões para os principais sistemas operacionais do mercado. É também através desta IDE que é feito o envio do programa para o microcontrolador presente na placa arduino.

A linguagem de programação é uma adaptação da linguagem C, possuindo as seguintes funções básicas e obrigatórias (TIMMIS, 2011):

- **setup:** ocorre uma única vez e é usada, normalmente, para a inicialização das variáveis;
- **loop:** toda instrução que estiver nesta função irá se repetir indefinidamente. Caso se deseje que essa repetição tenha alguma periodicidade de tempo, deve-se determinar o tempo de pausa (instrução *delay*) para que isso ocorra.

Um exemplo de código de programação pode ser visto na figura 12.

```

int buttonVal = 0; // Will pass button value here

void setup()
{
  pinMode(buttonPin, INPUT); // Makes buttonPin an input
  pinMode(motorPin, OUTPUT); // Makes motorPin an output
  digitalWrite(buttonPin, HIGH); // Activates the pull up resistor
                                // If we did not have this, the switch
                                // would not work correctly.
}

void loop()
{
  buttonVal = digitalRead(buttonPin); // Whatever is at buttonPin is
                                      // sent to buttonVal

  if (buttonVal == 1)
  {
    digitalWrite(motorPin, HIGH); // Turn motor on if buttonVal equals one
  }
  else
  {
    digitalWrite(motorPin, LOW); // Turns motor off for all other values
  }
}

```

Figura 12. Exemplo de programação do Arduíno (TIMMIS, 2011).

2.6. Global Position System – GPS

Sistema de Posicionamento Global (Global Position System – GPS) é um sistema de navegação baseado em uma constelação de 24 satélites a uma altitude de 20.200 km em seis órbitas distintas, igualmente espaçadas de 60 graus, com quatro satélites (STABILE & BALASTREIRE, 2006).

Os satélites GPS circulam o planeta Terra duas vezes por dia em uma órbita extremamente precisa transmitindo sinais para o cálculo do posicionamento de um objeto fixo ou móvel na superfície da Terra ou próximo a esta. Eles são energizados por energia solar e usam baterias adicionais para quando ocorrem problemas de eclipse solar (STABILE & BALASTREIRE, 2006).

Os receptores GPS recebem essa informação e usam a triangulação para calcular a posição exata do usuário. Essencialmente o receptor GPS compara o tempo que um sinal foi transmitido por um satélite e o tempo em que foi recebido pelo receptor (STABILE & BALASTREIRE, 2006).

Para a determinação de uma posição 2D (latitude e longitude) são necessários pelo menos 3 satélites. Para determinar uma posição 3D (latitude, longitude e altitude) são necessários pelo menos quatro satélites.

Segundo Stabile & Balastreire (2006), os sinais emitidos pelos satélites são emitidos em duas bandas (L1 e L2) com dois códigos diferentes: o Y (Código de Precisão) e o C/A (*Coarse Acquisition code*).

Segundo Searcy (2001), o Departamento de Defesa dos Estados Unidos restringiu o uso do código de precisão (Y) aos militares, que é criptografado e até o ano de 2000 reduzia a acurácia dos sinais, deixando-a em torno de 60 a 100 m. Para contornar esse problema foi desenvolvido o DGPS (Sistema de Posicionamento Global Diferencial Relativo).

O DGPS utiliza um sinal de correção emitido por uma estação fixa, seja ela uma antena fixa na terra ou um satélite estacionário. Como sua posição no globo terrestre é conhecida, a estação calcula e determina o erro de posição, enviando o sinal de correção em tempo real para os receptores GPS (SEARCY, 2001).

2.6.1. O Padrão NMEA 0183

O padrão NMEA 0183 ou, simplesmente NMEA (*National Marine Electronics Association*), é um conjunto de especificações de dados para comunicação de dispositivos eletrônicos de navegação tais como anemômetros, ecolocalizadores, girocompassos, piloto automático, receptores GPS e muitos outros tipos de instrumentos (NMEA, 2001).

O NMEA consiste de sentenças onde a primeira palavra, denominada de tipo de dado, define a interpretação do resto da frase (NMEA, 2001).

Cada tipo teria a sua própria interpretação única que é definida no padrão NMEA. Qualquer dispositivo ou programa que lê os dados pode considerar somente a sentença de dados de seu interesse e simplesmente ignorar as outras frases (NMEA, 2001).

Para o caso de um GPS, os principais tipos de dados definidos pelo padrão NMEA estão descritos no quadro 3.

Quadro 3 - Códigos do NMEA (NMEA, 2001)

Código	Significado
\$GPBOD	<i>Bearing, origin to destination</i>
\$GPBWC	<i>Bearing and distance to waypoint, great circle</i>
\$GPGGA	<i>Global Positioning System Fix Data</i>
\$GPGLL	<i>Geographic position, latitude / longitude</i>
\$GPGSA	<i>GPS DOP and active satellites</i>
\$GPGSV	<i>GPS Satellites in view</i>
\$GPHDT	<i>Heading, True</i>
\$GPR00	<i>List of waypoints in currently active route</i>
\$GPRMA	<i>Recommended minimum specific LoranC data</i>
\$GPRMB	<i>Recommended minimum navigation info</i>
\$GPRMC	<i>Recommended minimum specific GPS/Transit data</i>
\$GPRTE	<i>Routes</i>
\$GPTRF	<i>Transit Fix Data</i>
\$GPSTN	<i>Multiple Data ID</i>
\$GPVBW	<i>Dual Ground / Water Speed</i>
\$GPVTG	<i>Track made good and ground speed</i>
\$GPWPL	<i>Waypoint location</i>
\$GPXTE	<i>Crosstrack error, Measured</i>
\$GPZDA	<i>Date & Time</i>

2.7. Banco de Dados

Banco de Dados é uma coleção de dados com uma estrutura regular. Onde dados têm a finalidade de organizar informação, normalmente agrupada e utilizada para um mesmo fim (NAVATHE & ELMASRI, 2001).

Um Sistema de Gerenciamento de Banco de Dados (SGBD) é uma coleção de arquivos inter-relacionados que permitem ao usuário o acesso à consulta e às alterações desses dados. O maior benefício é proporcionar ao usuário uma visão abstrata dos dados, isto é, o sistema acaba por ocultar determinados detalhes sobre a forma de armazenamento e manutenção desses dados (NAVATHE & ELMASRI, 2001).

Suas vantagens são: isolar os usuários dos detalhes mais internos do banco de dados (abstração de dados); prover a independência de dados às aplicações (estrutura física de armazenamento versus estratégia de acesso); rapidez na manipulação e no acesso à informação; redução do esforço humano (desenvolvimento e utilização); disponibilização da informação no tempo necessário; controle integrado de informações distribuídas fisicamente; redução de redundância e de inconsistência de informações; compartilhamento de dados; aplicação automática de restrições de segurança; redução de problemas de integridade (NAVATHE & ELMASRI, 2001).

2.7.1. PostgreSQL

O PostgreSQL é um SGBD objeto-relacional de código aberto, extremamente robusto e confiável, além de ser extremamente flexível e rico em recursos. Ele é considerado objeto-relacional por implementar, além das características de um SGBD relacional, algumas características de orientação a objetos, como herança e tipos personalizados (POSTGRESQL, 2011).

O grupo global de desenvolvimento do PostgreSQL é formado essencialmente por empresas especializadas em PostgreSQL, empresas usuárias do sistema, além dos pesquisadores acadêmicos e programadores independentes.

Além da programação, essa comunidade é responsável pela documentação, tradução, criação de ferramentas de modelagem e gerenciamento, e elaboração de extensões e acessórios (POSTGRESQL, 2011).

Pela riqueza de recursos e conformidade com os padrões, ele é um SGBD muito adequado para o estudo universitário do modelo relacional, além de ser uma ótima opção para empresas implementarem soluções de alta confiabilidade sem altos custos de licenciamento (POSTGRESQL, 2011).

O mecanismo de extensibilidade do PostgreSQL permite incorporar capacidades adicionais ao sistema de forma a torná-lo mais flexível para o gerenciamento de dados para cada classe de aplicação. No caso do SIG, isso significa a possibilidade do desenvolvimento de uma extensão geográfica capaz de armazenar, recuperar e analisar dados espaciais (POSTGRESQL, 2011).

2.7.2. Banco de Dados Geográficos

Banco de dados geográficos são coleções de dados georeferenciados, manipulados por Sistemas de Informação Geográfica (NAVATHE & ELMASRI, 2001).

O armazenamento dos dados geográficos requer as características espaciais e não-espaciais dos dados, como também a sua documentação (metadados). A integração entre os dados geográficos e a sua documentação é feita através de um SGBD (NAVATHE & ELMASRI, 2001).

Os SGBDs utilizam extensões específicas para suportarem os dados geográficos. Uma extensão tem a função de realizar a conversão, inserção, recuperação e extração dos dados geográficos junto ao SGBD. Um exemplo de extensão é o PostGIS do PostgreSQL (NAVATHE & ELMASRI, 2001).

2.7.3. PostGIS

O PostGIS é uma extensão espacial, gratuita e de código fonte livre desenvolvido por Refrections Reserch Inc (POSTGIS MANUAL, 2011). Sua construção é feita sobre o sistema de gerenciamento de banco de dados objeto relacional PostgreSQL, que permite o uso de objetos SIG de serem armazenados em banco de dados (POSTGIS MANUAL, 2011).

Nativamente o PostgreSQL suporta geometrias espaciais, porém o PostGIS adiciona a capacidade de armazenamento/recuperação segundo a especificação SFS (*Simple Features Specification*) do OGC. Além do armazenamento de dados geográficos, este módulo também implementa diversas funcionalidades de desenvolvimento de SIG corporativos (POSTGIS MANUAL, 2011).

Seu licenciamento é definido pela GNU GPL (*General Public License*), garantindo todas as liberdades de um programa livre, sendo que qualquer melhoria do código-fonte do PostGIS deve ser devolvida ao mantenedor do projeto (POSTGIS MANUAL, 2011).

Para tratar grandes volumes de dados, o PostGIS inclui suporte para índices espaciais GiST e R-Tree, além de funções para análise básica e processamento de objetos SIG (POSTGIS MANUAL, 2011).

O PostGIS contempla os seguintes tipos de dados: Ponto, Linha, Polígono, MultiPonto, MultiLinha, MultiPolígono e Coleção Geométrica. A hierarquia dos tipos espaciais do PostGIS é apresentada na figura 13 (POSTGIS MANUAL, 2011).

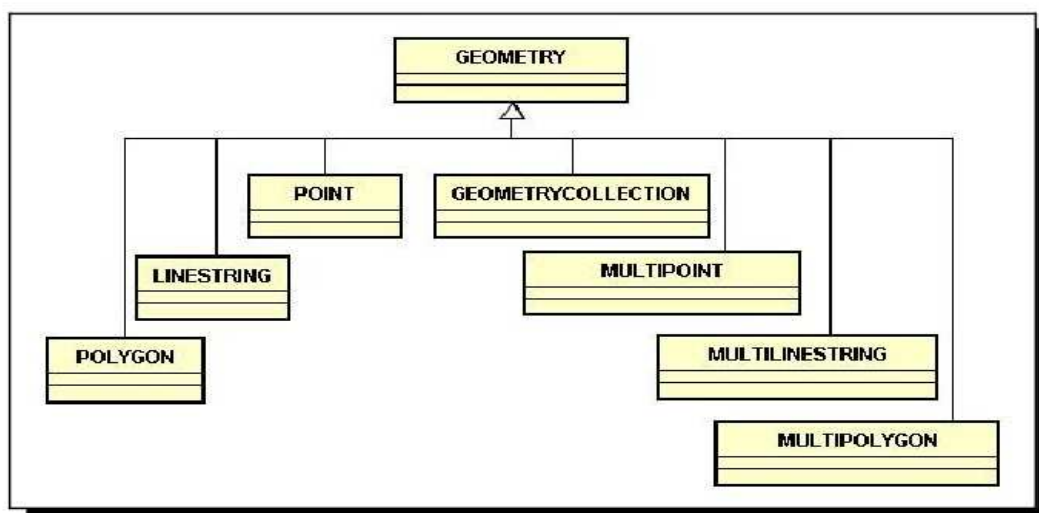


Figura 13. Tipos de dados espaciais do PostGIS

2.8. UML (Unified Modeling Language – Linguagem de Modelagem Unificada)

A UML é uma linguagem de modelagem para especificação, documentação, visualização e desenvolvimento de softwares orientados a objetos (FOWLER & SCOTT, 2000).

A UML tenta organizar de forma clara os principais modelos existentes através de diagramas, sendo possível expressar diversas perspectivas de visualização, tendo como alvo o entendimento de todas as pessoas envolvidas no

processo de desenvolvimento de um software e possível reutilização e manutenção desse sistema (FOWLER & SCOTT, 2000).

A UML é também definida como uma linguagem de modelagem visual, no sentido de prover facilidades na visualização, ou seja, o pleno entendimento das funções de um sistema a partir de diagramas que o represente (FOWLER & SCOTT, 2000).

Segundo Guedes (2011), os diagramas da UML 2.0 dividem-se em diagramas estruturais (Diagramas de Classes, de Estrutura Composta, de Objetos, de Componentes, de Implantação e de Pacotes) e diagramas comportamentais (Diagramas de Casos de Uso, Atividade, Máquina de Estados, Sequência, Comunicação, de Interação Geral e de Tempo), sendo que estes últimos possuem ainda uma subdivisão representada pelos diagramas de interação.

A seguir serão referenciados apenas o diagrama de casos de uso, o diagrama de classes e o diagrama de atividades, pois são os utilizados neste projeto.

2.8.1. Diagrama de Caso de Uso

O Diagrama de Caso de Uso serve para identificar as fronteiras do sistema e descrever os serviços que devem ser disponibilizados a cada um dos diversos utilizadores, que são os atores. A primeira tarefa a desenvolver para construir um diagrama caso de uso é a identificação dos atores do sistema (NUNES & O'NEILL, 2004).

Um ator é um papel desempenhado por entidades físicas (pessoas, organizações ou outros sistemas) que interagem com o sistema. Uma mesma entidade física pode desempenhar diferentes papéis no mesmo sistema, bem como um dado papel pode ser desempenhado por diferentes entidades. Atores são externos ao sistema. Um ator se comunica diretamente com o sistema, mas não é parte dele (FOWLER & SCOTT, 2000).

O caso de uso representa uma funcionalidade completa do sistema, formado por uma sequência de ações que irá gerar um resultado específico (FOWLER & SCOTT, 2000). Exemplos de casos de uso na figura 14.

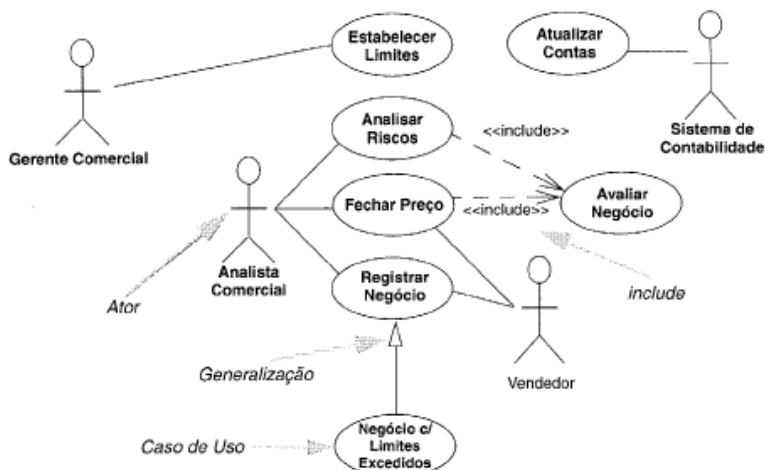


Figura 14. Exemplo de Diagrama de Caso de Uso (FOWLER & SCOTT, 2000).

2.8.2. Diagrama de Classes

Segundo a UNIMEP (2009), um diagrama de classes é considerado como a visão estática da estrutura do sistema e mostra as classes do sistema e os relacionamentos entre elas. Neste diagrama são definidas as classes, através de seus métodos e atributos, tendo como objetivo definir a base para outros diagramas apresentarem suas visões do sistema.

O atributo é uma informação para a qual cada objeto em uma classe tem o seu próprio valor. Os atributos adicionam detalhes às abstrações e são apresentados na parte central da classe. Os atributos possuem um tipo de dado, que pode ser primitivo ou específico do domínio. Ao identificar um atributo como sendo relevante, deve-se definir qual o seu tipo de dado, por exemplo, String, Integer, entre outros (FOWLER & SCOTT, 2000).

Conforme SILVA & VIDEIRA (2001), uma relação em UML estabelece a ligação entre elementos e é representada graficamente por um determinado tipo de linha. Um exemplo completo de Diagrama de Classes pode ser visto na figura 15.

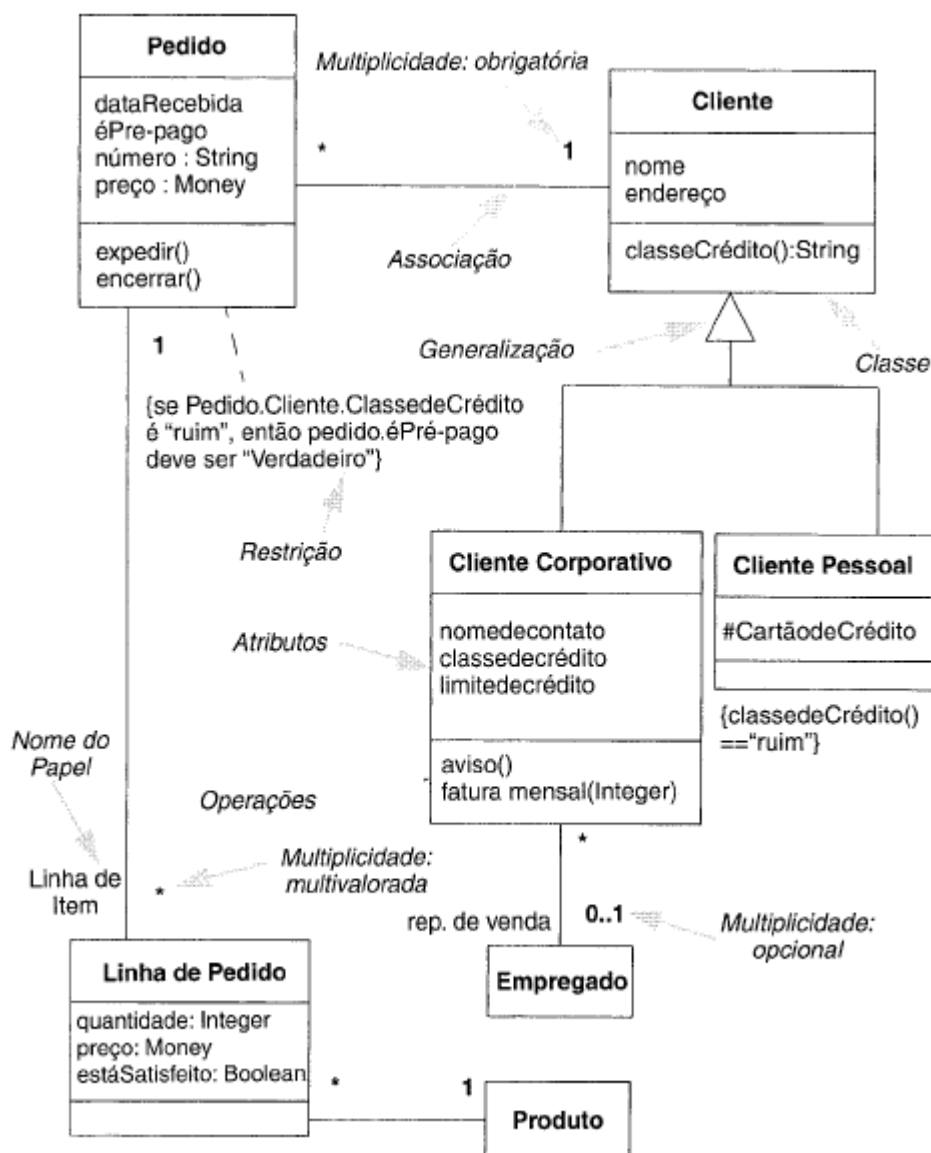


Figura 15. Exemplo de Diagrama de Classes (FOWLER & SCOTT, 2000).

2.8.3. Diagrama de Atividades

O diagrama de atividades descreve os passos a serem percorridos para a conclusão de uma atividade específica. Este diagrama concentra-se na representação do fluxo de controle de uma atividade (GUEDES, 2011).

Este diagrama é utilizado para modelar o tempo de vida de um objeto, além de ilustrar o fluxo do controle entre atividades, dando destaque a visualização das operações realizadas pelos objetos (SILVA & VIDEIRA, 2001). Um ponto comum no desenho de um diagrama de atividades consiste em especificar uma condição, que é representada por um losango, que recebe uma entrada e duas ou mais

saídas, de acordo com as condições. Apenas uma condição deve ser verdadeira ou uma condição “senão” para permitir definir um único fluxo das atividades.

Na modelagem de processos de negócio é comum a realização de atividades por várias entidades, participantes do processo. A UML propõe o conceito de raias (*swimlanes*) como elemento que permite agrupar as várias atividades da responsabilidade de cada entidade participante. Cada grupo é separado por uma linha vertical. (FOWLER & SCOTT, 2000) Um exemplo do diagrama de atividades pode ser visto na figura 16.

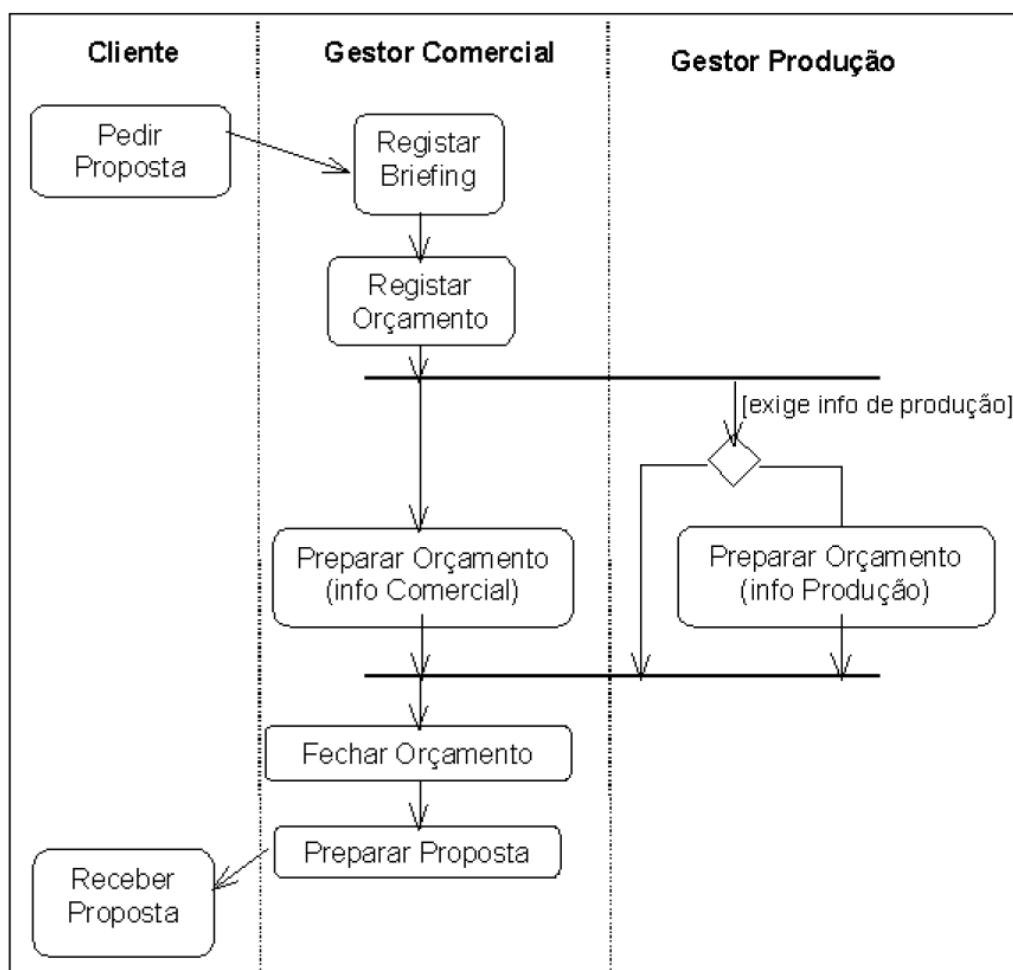


Figura 16. Exemplo de Diagrama de Atividades (SILVA & VIDEIRA, 2001).

2.9. Mapeamento objeto relacional - Hibernate

O Hibernate é uma solução de código aberto para Mapeamento Objeto Relacional, criado em meados de 2001 por Gavin King e outros desenvolvedores. O Hibernate é um framework de persistência que é utilizado por desenvolvimento

em linguagem Java e recentemente para aplicações desenvolvidas na plataforma.Net (BAUER & KING, 2005).

Em resumo, o Hibernate é responsável por mapear as classes desenvolvidas na linguagem de programação e correlacioná-las com tabelas do banco de dados de forma que, para o desenvolvedor, fique transparente o ambiente de programação e o banco de dados (BAUER & KING, 2005).

Como vantagens para utilização, o Hibernate apresenta ótima documentação, facilidade de uso, excelentes recursos e uma forma inteligente para gerenciamento do projeto, liberando o desenvolvedor de codificar manualmente o acesso ao banco de dados, utilizando os conceitos de domínios de objetos e de forma simples (BAUER & KING, 2005).

A relação entre os objetos e o banco de dados é feita por arquivos de mapeamento baseados em XML (eXtensible Markup Language). Estes arquivos indicam quais atributos (em um objeto) serão mapeados para suas respectivas colunas, na tabela (BAUER & KING, 2005).

O Hibernate provê um meio de se controlar operações de criação, remoção, consulta e atualização no banco de dados, através de métodos de suas interfaces (BAUER & KING, 2005).

2.9.1. Hibernate Espacial

O Hibernate Espacial é uma extensão, de código aberto, do Hibernate para lidar com dados geográficos (HIBERNATE, 2013).

Ele permite lidar com dados geográficos de uma forma padronizada. Deixa transparente a capacidade espacial do banco de dados e permite que a aplicação consiga usar toda a capacidade do mesmo. Permite realizar funções de armazenamento e consultas sobre o banco de dados espacial (HIBERNATE, 2013).

O Hibernate Espacial suporta a maioria das funções do OGC, e tem suporte aos seguintes bancos de dados: o Oracle 10g/11g, PostgreSQL / Postgis, MySQL, Microsoft SQL Server e H2/GeoDB (HIBERNATE, 2013).

2.10. Java Server Faces (JSF)

De acordo com Pitanga (2009), Java Server Faces é uma tecnologia que incorpora características de um *framework Model-View-Controller* (MVC) para web e de um modelo de interfaces gráficas baseado em eventos. Uma de suas melhores vantagens é a clara separação entre a visualização e as regras de negócio (modelo).

O JSF é um *framework* desenvolvido pela Sun Micro Systems, e é parte integrante da tecnologia do Java EE. Ele foi desenhado para facilitar o desenvolvimento de aplicações web através de componentes de interface do usuário (SHANKAR, 2008).

O sistema utiliza o paradigma MVC para trabalhar com sua apresentação e navegação de dados. A ideia do padrão MVC é dividir uma aplicação em três camadas: modelo, visualização e controle (PITANGA, 2009).

O modelo é responsável por representar os objetos de negócio, manter o estado da aplicação e fornecer ao controlador o acesso aos dados (PITANGA, 2009).

A visualização representa a interface com o usuário, sendo responsável por definir a forma como os dados serão apresentados e encaminhar as ações dos usuários para o controlador.

Já a camada de controle é responsável por fazer a ligação entre o modelo e a visualização, além de interpretar as ações do usuário e as traduzir para uma operação sobre o modelo, onde são realizadas mudanças e, então, gerar uma visualização apropriada (PITANGA, 2009).

O JSF é responsável por interagir com o usuário (cliente) e fornece ferramentas para criar uma apresentação visual, a parte lógica e a lógica de negócios de uma aplicação web (PITANGA, 2009).

Segundo Shankar (2008), menciona que o JSF é um *framework* para a construção de aplicativos web com componentes de desenvolvimento de interface ao usuário orientado a eventos. Ele veio para simplificar o trabalho dos desenvolvedores de aplicações web.

2.11. OpenLayers

O OpenLayers é um *framework* desenvolvido em *Javascript* para a plotagem de mapas nos principais navegadores de internet do mercado. Desta forma, constitui-se em uma opção interessante para elaboração de um sistema de informação geográfica (OPENLAYERS, 2008).

O OpenLayers é uma biblioteca de código aberto feito em JavaScript que pode ser usado para disponibilizar/exibir dados geográficos na internet ou na rede local (OPENLAYERS, 2008).

Ele pode obter dados de diversos recursos, tais como os padrões do OGC Web Map Service (WMS), Web Feature Service (WFS) bem como Google Maps, OpenStreetMap, Bing Maps, MapServer, GeoServer e muitos outros. Possui ainda suporte à GeoRSS, navegação via mouse e pelo teclado, adição de marcadores e seleção de camadas (OPENLAYERS, 2008). A figura 17 representa um mapa renderizado com o OpenLayers.

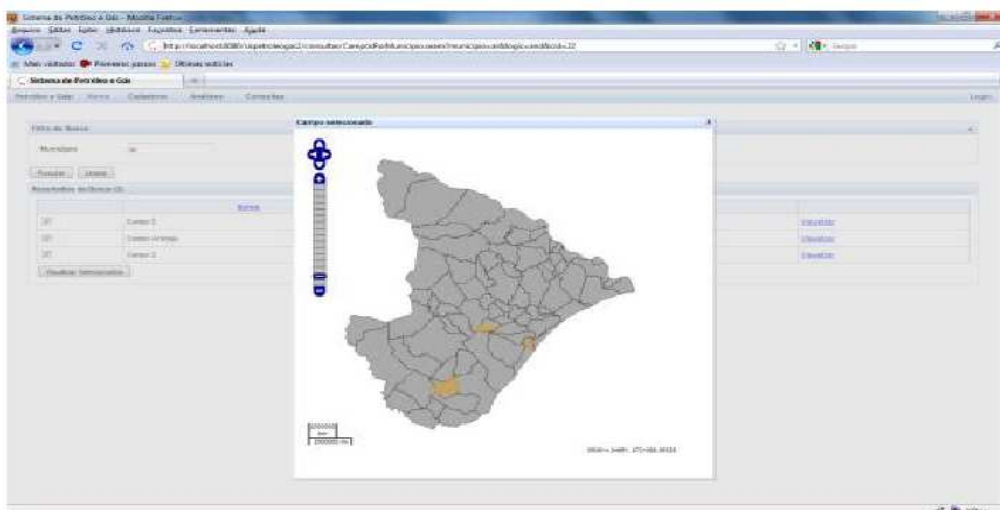


Figura 17. Exemplo de mapa renderizado em OpenLayers (OLIVERA et al. 2003).

2.12. Trabalhos Relacionados

Atualmente há no mercado uma série de sistemas que reúnem as funcionalidades requeridas pela agricultura de precisão. Muitos desses programas são proprietários (para a aquisição o usuário deve comprar a sua licença), gratuitos, mas o usuário não tem acesso ao código-fonte (é disponibilizado para

aquisição sem nenhum custo) e, outros, de código aberto (mantido normalmente por uma comunidade, o usuário além de poder adquirir o programa gratuitamente pode também adquirir seu código-fonte para possíveis customizações do programa).

O Farm Trac, (FARMWORKS, 2013) é um programa proprietário (US\$ 300,00) voltado para usuários sem muita experiência no uso de computador que mantêm registros e mapas da propriedade. Integra-se com outros programas da empresa para aumentar a sua capacidade.

O Quick Yields (FARMWORKS, 2013), é disponibilizado gratuitamente, com o propósito de visualizar e imprimir mapas com dados do campo.

O programa Farm Site (FARMWORKS, 2013), considerado para uso de usuários avançados, é um programa proprietário (US\$ 450,00) para mapear a propriedade. Para o seu funcionamento é necessário a aquisição do programa FarmTrac.

O Farm Site Pro (FARMWORKS, 2013) é um programa proprietário para usuários avançados da Farm Work Software que pode ser usado para registro e mapeamento de talhões usando coordenadas reais. Possui funções de múltiplas camadas.

O SigaField for Windows é um programa proprietário da empresa SIGA Farm Software (SIGA, 2013) considerado de uso simples para o usuário, capaz de gerenciar talhões com módulo para análise de custo/benefício, geração de mapas de recomendações de fertilizantes e registro de informação de campo.

O WinCrop é um programa proprietário da BCL LandView Systems Inc. (BCL, 2013) que mantém um registro das produções. Também é usado para gerenciar propriedades e talhões.

O FarmView Record Keeper, também da BCL LandView Systems Inc., é um programa proprietário que mantém registro de dados de produção e planejamento. É possível através dele gerenciar a propriedade (BCL, 2013).

O LandView Mapper Pro é um programa proprietário para usuários avançados da empresa BCL LandView Systems Inc. para geração de mapas multiusuário, multicamadas e GPS (BCL, 2013).

O LandView DSS Pro é um programa proprietário para usuários avançados da empresa BCL LandView Systems Inc. para manter registro de propriedades. É multiusuário e faz análise de mapas (BCL, 2013).

O JMap, é um programa proprietário de mapeamento de campo que fornece funcionalidades básicas de mapeamento de talhões (DEERE, 2013).

O ARCVIEW GIS é um SIG para usuários profissionais, proprietário, que pode ser usado em várias áreas para mapeamento e Sistema de Informação Geográfico. É uma ferramenta para mostrar, consultar e analisar informações espaciais. Este não possui customização para a agricultura (ESRI, 2013).

O SSToolkit é um programa proprietário para usuários avançados para gerenciamento de dados agrícolas na plataforma ARCVIEW GIS (SST, 2013).

O SSToolBox, é um programa proprietário para uso de profissionais da área de computação para gerenciamento de dados espaciais agrícolas, nas plataformas ARCVIEW GIS e Surfer, para usuários avançados (SST, 2013).

O Croplands-The System é um programa proprietário para usuários profissionais da área de computação com recursos de SIG que cobrem grande parte das operações de gerenciamento de serviços envolvidos na produção agrícola (LINNET, 2012).

3. MATERIAL E MÉTODOS

3.1. Componentes Físicos

Para monitorar a produção e os insumos agrícolas de um agronegócio serão necessários sensores distribuídos pelo campo ou presentes em máquinas agrícolas tais como tratores.

A placa DAQ é substituída por um arduíno, onde será conectada uma placa de GPS dotada de um gravador de cartão SD ou uma placa a parte com o cartão SD.

O importante é que todo módulo de aquisição de dados tenha integrado um GPS e um módulo de gravação de SD. Esta integração (placa GPS, cartão SD e Arduíno) denomina-se módulo fixo. A integração do sensor ao módulo fixo denomina-se módulo conectável.

3.1.1. O Módulo de conexão fixa

Dados que forem lidos no campo devem ter uma posição geográfica atrelada aos mesmos, isto porque, todos os dados serão informados pelo WebFazenda através de um mapa.

Como os dados não serão monitorados em tempo real, toda leitura realizada será gravada em um cartão SD para posterior importação dos dados pelo WebFazenda. Este é um meio de baixo custo, seguro, simples, de dimensões pequenas e de baixo consumo de energia elétrica.

A figura 18 mostra o módulo de aquisição de dados proposto nesta pesquisa usando um arduíno (parte inferior) e uma placa de GPS com gravador de cartão de memória (parte superior).



Figura 18. Módulo de Aquisição de Dados com arduíno

Este pequeno dispositivo utiliza uma alimentação de corrente contínua de até 9V. Entretanto, o módulo consome somente 3.3V, liberando para a alimentação do sensor a ser usado 5.7V. Caso haja necessidade de uma excitação maior para o dispositivo, torna-se necessário realizar essa alimentação externamente.

A conexão física da placa GPS com o arduíno é bem simples, pois o mesmo já vem com as conexões prontas para tal. O único cuidado a se ter é com as escolhas corretas das portas a serem usadas para transmissão de dados (TX) e recepção de dados (RX) do GPS. O módulo GPS disponibiliza o uso de todas as portas digitais (0 a 7). No entanto, as portas 0 e 1 já são usadas para a comunicação do arduíno com o computador para a transferência do programa que ficará gravado em sua memória interna. Assim sendo, o seu uso pode gerar problemas no momento da programação do módulo. Logo, o mais comum está no uso das portas 3 (RX) e 2 (TX). O módulo GPS disponibiliza acesso a um cartão de memória na própria placa. O acesso a esse cartão se dá através da porta digital 10.

A maior dificuldade está na programação do acesso a esses módulos. O arduíno utiliza uma linguagem de programação própria que é muito próxima da própria linguagem de programação C. O uso desses dispositivos (GPS e cartão) é realizado através de um programa que deve ser transferido para o equipamento e, toda a comunicação com o GPS e o cartão SD ocorrerá automaticamente.

O programa desenvolvido neste projeto se comunica com o GPS utilizando as bibliotecas “TinyGPS” e “SD”. A primeira encontra-se disponível gratuitamente na internet enquanto a segunda acompanha o pacote de instalação.

Para a comunicação com o GPS devem ser confirmadas as portas físicas, no caso as portas 3 e 4, conforme se mencionou previamente.

Além destas portas será também necessário confirmar a porta de comunicação com o leitor de cartão de memória (o módulo GPS disponibiliza a porta 10) e a taxa de transmissão de dados com o GPS (9600 baud) e com o computador (115200 baud).

O programa desenvolvido para comunicação com o módulo fixo segue o diagrama de atividades da figura 19.

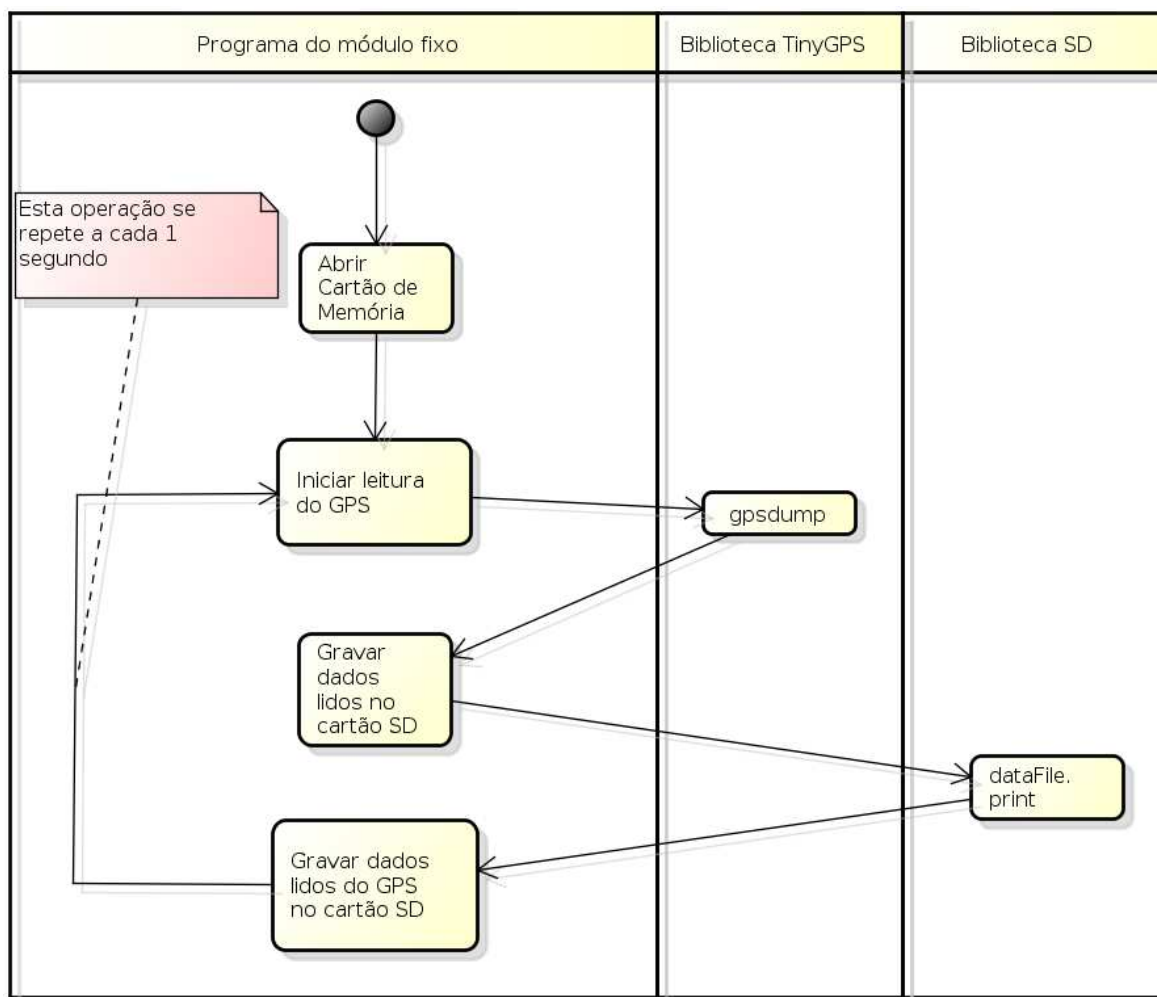


Figura 19. Fluxograma do programa do módulo fixo do arduino

O procedimento de leitura de GPS e a gravação dos dados no cartão de memória se repetem a cada 1 segundo. Dependendo do sensor a ser utilizado, esse valor pode ser facilmente ajustado, alterando-o no próprio programa e reenviando o mesmo para o DAQ.

O GPS fornece sua saída através do padrão aberto NMEA. A biblioteca TinyGPS, através de uma função `gpsdump`, faz a conversão do dado lido de forma transparente. Assim sendo os dados de latitude, longitude, altitude, data e hora são obtidos separadamente, sem muito esforço computacional.

Os dados recebidos da função `gpsdump` são organizados em modo texto, separados por ponto e vírgula e em seguida os dados são repassados para o `dataFile.print`. O `dataFile` é apenas uma chamada ao módulo SD e o `print` é uma função que adiciona ao arquivo dado. No caso deste projeto, o arquivo gerado se chama `DATALOG.TXT`. Um exemplo pode ser visto no quadro 4.

Quadro 4 - Formato da saída do cartão de memória do módulo fixo

```
-22.39671 ; -41.79267 ; 346 ; 10/21/2013 00:27:27 368 ; 26.84000  
-22.39672 ; -41.79269 ; 224 ; 10/21/2013 00:27:31 246 ; 26.84000  
-22.39672 ; -41.79269 ; 255 ; 10/21/2013 00:27:32 276 ; 26.84000  
-22.39672 ; -41.79268 ; 283 ; 10/21/2013 00:27:33 307 ; 26.84000
```

Este módulo fixo deve estar presente em todos os sensores que serão usados no campo. Ele é fixo não porque não se tem como movimentá-lo, mas porque tudo que está presente nele não pode ser alterado. Por outro lado, este módulo deve ser estendido para ser capaz de adicionar ao arquivo gerado os dados de leitura do sensor.

A instalação, a configuração e a programação do sensor situam-se no módulo conectável, que será explicado a seguir.

3.1.2. O Módulo conectável

Eletronicamente, existem dois tipos de sensores: analógicos e digitais. O arduíno, apresentado na figura 20, possui 6 portas analógicas e 12 portas digitais. Serão nessas portas que os sensores e os circuitos eletrônicos necessários serão ligados. A figura 20 apresenta o arduíno Uno com suas portas e conexões.

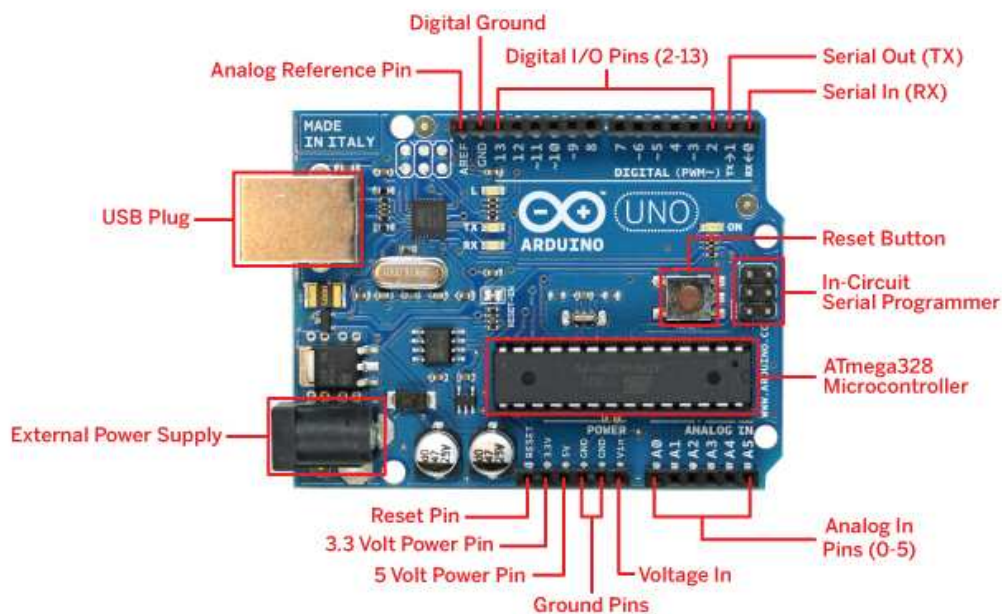


Figura 20. Arduíno UNO (TIMMIS, 2011).

Para explicar o procedimento de conexão de um novo dispositivo, será usado como exemplo um sensor de temperatura do ambiente. O sistema de aquisição de dados proposto, além de ser capaz de gravar no cartão de memória o arquivo DATALOG.TXT com os dados de localização e a temperatura lida pelo sensor a cada 1 s, também tem dois leds exibindo a situação se a temperatura estava elevada ou baixa. Desta forma, pode-se simular um circuito que, além de realizar a aquisição de dados também permite uma intervenção imediata caso ocorra algum problema, por exemplo com uma temperatura muito elevada.

Para a montagem da parte móvel, é usado um conector pino DIN, um led vermelho, um led amarelo, uma placa PCI perfurada, dois resistores 39 ohms e um sensor de temperatura LM35. Todos esses elementos são usados para dar um aspecto de escudo à parte móvel do sensor. O sistema físico de aquisição de dados pode ser visto na figura 21.

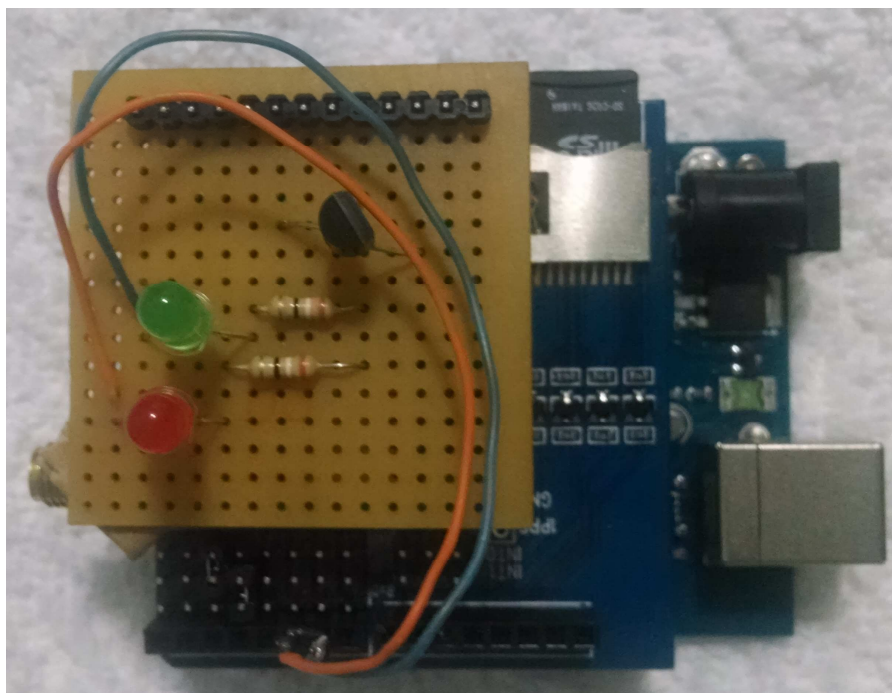


Figura 21. Sistema de Aquisição de Dados – Leitura de Temperatura

O procedimento para a instalação do módulo conectável passa pelos seguintes passos:

- 1) observar se o sensor a ser conectado é analógico ou digital;
- 2) verificar se o sensor vai necessitar de alimentação externa;
- 3) conectar o sensor às portas corretas;
- 4) Fazer a alimentação externa necessária, caso necessário;
- 5) estender o programa para ler a porta necessária, fazer as conversões devidas e adicionar o dado lido à geração do arquivo DATALOG.TXT do módulo fixo.

Na montagem, o sensor de temperatura foi conectado ao pino analógico zero, o led verde no pino digital cinco e, o led vermelho no pino digital seis.

O próximo passo é a alteração do programa do módulo fixo para a leitura do sensor. Neste ponto deve-se saber exatamente a qual porta está acoplado o sensor. Em sendo na porta serial, deve-se usar o comando *analogRead*, no caso do sensor ser digital, o comando é *digitalRead*.

Foi adicionada a função da figura 22. Nesta função, ele lê o pino zero, converte a temperatura para graus célsius e, depois, trata os leds, acendendo o led vermelho caso a temperatura esteja acima de 34 graus ou, acendendo o led verde se a temperatura estiver abaixo de 30 graus. Para valores contidos nesse

intervalo, o sistema de aquisição de dados não irá acender nenhum dos dois leds. No final, adiciona ao arquivo do DATALOG.TXT o valor lido na posição 5 da linha.

```

DataLogKitGPSTemperatura $
static void temperaturadump()
{
  valorLido = analogRead(pinoLM35);
  temperatura = (valorLido * 0.00488); ///
  temperatura = temperatura * 100; /// Conversão do valor obtido através da porta analogica

  if (temperatura <= 30) {
    digitalWrite(5, HIGH);
    digitalWrite(6, LOW);
  } else {
    if (temperatura >= 34) {
      digitalWrite(5, LOW);
      digitalWrite(6, HIGH);
    } else {
      digitalWrite(5, LOW);
      digitalWrite(6, LOW);
    }
  }

  print_float(temperatura, 0, 10, 5);
}

```

Figura 22. Programação do Sistema de Aquisição de Dados – Leitura de Temperatura

Deve-se ter cuidado de consultar os *datasheets* e manuais dos sensores para que nenhuma ligação seja feita de forma indevida para não gerar nenhum tipo de problema, seja de defeito de algum componente, como também influenciar em uma leitura de dados incorretos.

O próximo passo é ligar o módulo criado com o código do módulo fixo. Para isso, basta adicionar a função de leitura do módulo do sensor ao código e, por conseguinte, adicionar mais uma coluna na gravação do cartão SD com o valor lido na unidade de medida desejada pelo usuário (aqui selecionado graus célsius). Assim sendo, a cada segundo (ou outro período configurável no programa), o módulo irá capturar a posição geográfica (latitude, longitude e altitude) e o valor lido do LM35, traçando uma rota por todo o local em que o trator passar qual o valor lido pelo sensor.

O programa final, que é transferido para o módulo de aquisição de dados, segue o fluxograma da figura 23.

Uma vez que se sabe como fazer a conexão física do sensor com a obtenção dos dados geográficos associados ao mesmo, o próximo passo é

explicar o funcionamento e as tecnologias associadas ao programa gerenciador dos dados obtidos no campo, neste projeto denominado de WebFazenda.

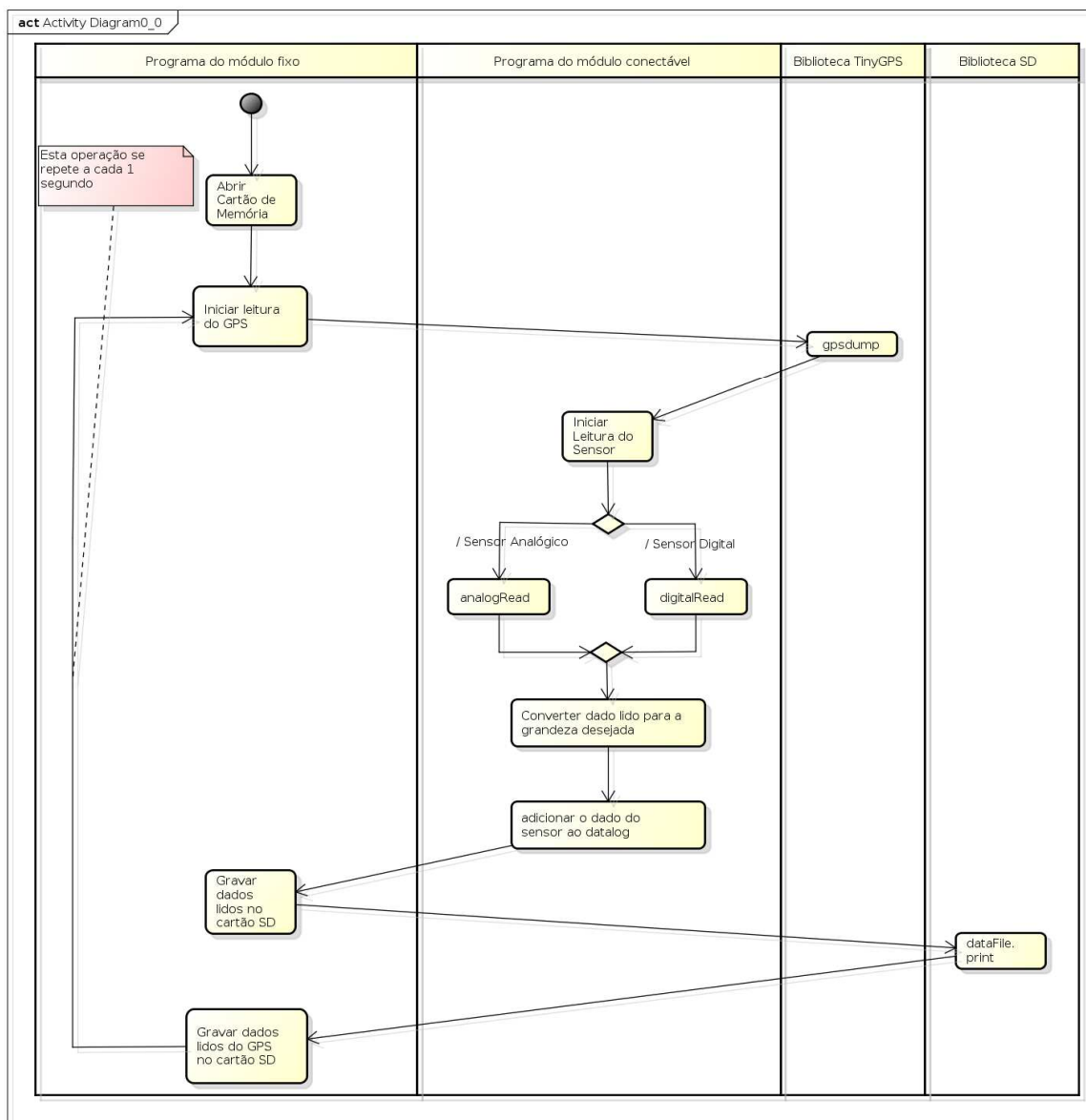


Figura 23. Fluxograma Completo do Programa do Módulo de Aquisição de Dados

3.2. WebFazenda

O WebFazenda é um programa desenvolvido para ser executado em navegadores de internet tanto em uma rede local quanto na internet, se esse for o desejo do usuário.

Para o desenvolvimento do WebFazenda foram usadas as seguintes tecnologias: PostgreSQL, PostGIS, GeoServer, Hibernate, OpenLayers e Java Servlet Faces (JSF).

3.2.1. Tecnologias de Servidores

O PostgreSQL é o Sistema Gerenciador de Banco de Dados (SGBD) responsável por todos os dados armazenados (geográficos ou não). Mas, mesmo sendo capaz de armazenar e recuperar, o PostgreSQL não é capaz de interpretar os dados nem tampouco realizar as operações sobre os dados geográficos. Para tanto, neste projeto foi usada a extensão do PostgreSQL denominada PostGIS.

Para processar os dados geográficos armazenados no SGBD é usado o servidor de mapas GeoServer. O Hibernate é responsável por mapear os dados não geográficos no banco de dados para a linguagem de programação (Java). Para a exibição e consulta dos dados geográficos, é usado o OpenLayers e para a consulta e exibição de dados não geográficos é usado o JSF. Toda essa ligação tecnológica das ferramentas usadas para o desenvolvimento do programa pode ser observada na figura 24.

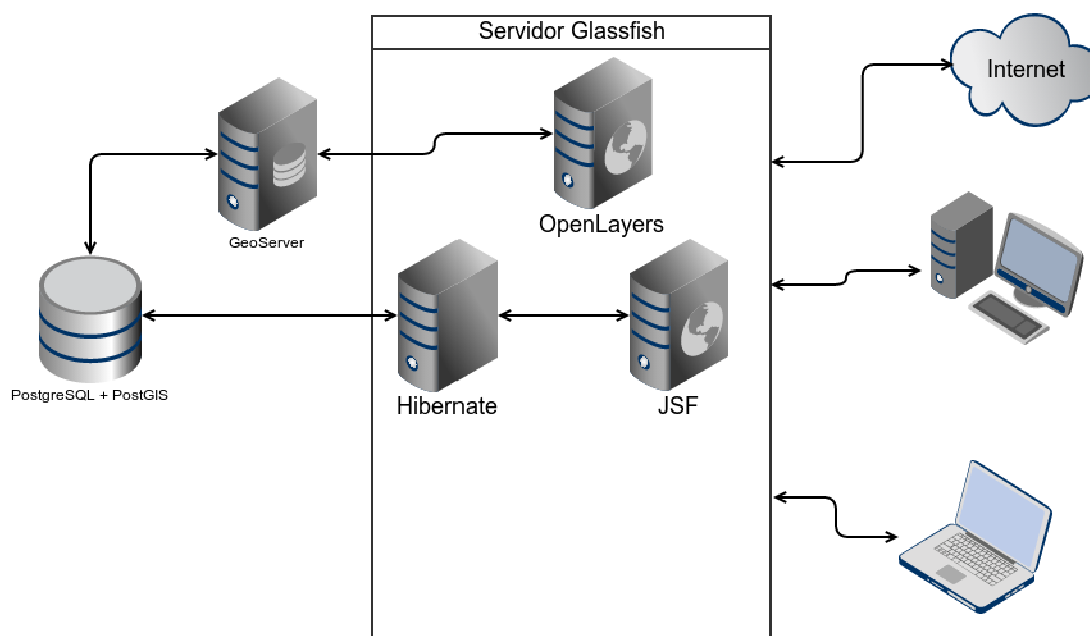


Figura 24. Distribuição de Tecnologias e Equipamentos para o WebFazenda

3.2.2. Tecnologias de Programação

Em termos de estrutura interna, o WebFazenda foi desenvolvido seguindo o modelo de dados (diagrama de classes) apresentado na figura 25. A classe Regiao representa o local que será controlado, ou seja, a área que se deseja controlar. O programa é capaz de gerenciar mais de uma região, e cada região terá um polígono para representá-la.

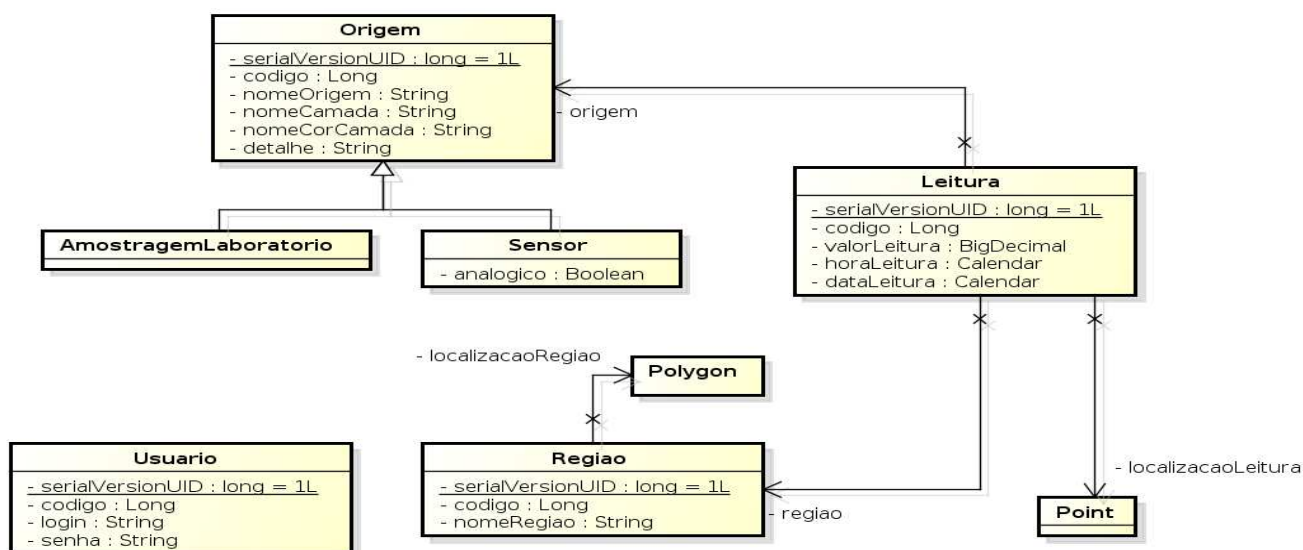


Figura 25. Diagrama de Classes do WebFazenda

A Origem representa o tipo de dado que será lido. Conforme pode ser visto no diagrama da figura 25, a origem pode ser um sensor ou uma amostragem laboratorial. Independente do seu tipo possui um nome (`nomeOrigem`), alguma observação que se deseja ter (`detalhe`) e dados relativos ao gerenciamento geográfico (`nome da camada` e `nome da cor da camada`).

A origem do tipo sensor é referente aos dados que serão coletados no campo a partir de um sensor usando o módulo de aquisição de dados. Esse sensor pode ser analógico ou digital. Essa informação é importante para a futura interpretação dos dados. A amostragem laboratorial corresponde a dados coletados do campo, mas processados em laboratório.

A leitura corresponde aos dados obtidos no campo, por um sensor ou obtidos em laboratório. Toda leitura deve corresponder a uma localização

geográfica do tipo ponto. É sobre os dados dessa classe que os mapas serão gerados para que o administrador da fazenda seja capaz de realizar seu processo de tomada de decisão.

A classe usuário não tem interferência sobre os dados que serão processados. Apenas é o local onde serão armazenados os dados dos usuários que poderão usar o programa para estabelecer uma segurança mínima de acesso ao mesmo.

Todas as classes foram transformadas em tabelas no banco de dados bdwebfazenda do PostgreSQL. Os dados geográficos (localizacaoRegiao e localizacaoPonto) são armazenados usando a extensão PostGIS. As classes Origem, Sensor e Analise Laboratorial se transformam em uma única tabela – Origem, conforme pode ser visto na figura 26. A tabela *spatial_ref_sys* é criada pelo PostGIS para controle do sistema de referência geográfica. No caso desse trabalho, o PostGIS usado foi com a versão 2.1.1 e o PostgreSQL versão 9.3.

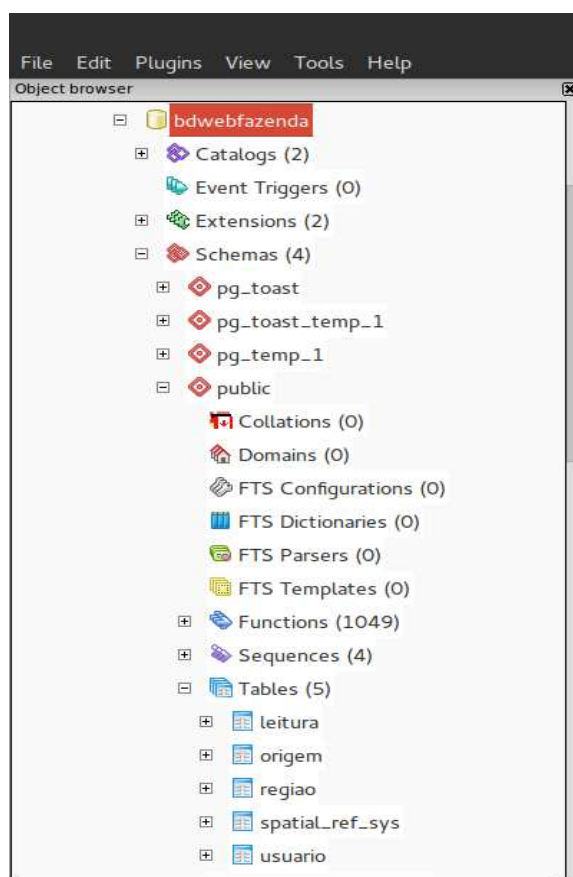


Figura 26. Tabelas do Banco de Dados

Para a manipulação, o processamento e a disponibilização dos dados geográficos, foi utilizado o programa GeoServer na versão 2.5. O primeiro passo

foi a criação do *workspace*, que corresponde à área de trabalho onde todos os dados do projeto WebFazenda irão ocorrer. Foi criada a área de trabalho com o próprio nome do projeto (WebFazenda).

O próximo passo é permitir que a área de trabalho tenha acesso ao banco de dados PostgreSQL, que é onde os dados geográficos e não geográficos estarão armazenados. Esse procedimento acontece a partir de uma *store* (armazenamento) no GeoServer. Foi criado então um armazenamento se conectando diretamente ao banco de dados com suporte ao PostGIS *bdwebfazenda*. Para não criar nomes em excesso esse armazenamento também foi nomeado de *bdwebfazenda*.

Uma configuração importante a ser feita no GeoServer é a de estilos (nomeado de *styles*). Os estilos vão além de aparências dos mapas quando visualizados. Eles também realizam processamentos importantes como o mapa de calor (*HeatMap*), que é usado neste projeto para possibilitar a visualização degradê da intensidade de valores de um determinado sensor ou mesmo da produção, gerando assim o mapa de produtividade, por exemplo.

Os estilos criados por este projeto são os pontos, nomeados pelas possíveis cores a serem usadas nos sensores ou nas análises laboratoriais (amarelo, azul, bege, branco, laranja, lima, marrom, oliva, ouro, preto, purpura, rosa, salmão, verde e vermelho). Esse estilo possibilita uma visualização simples dos pontos. A definição desse estilo pode ser vista na figura 27.

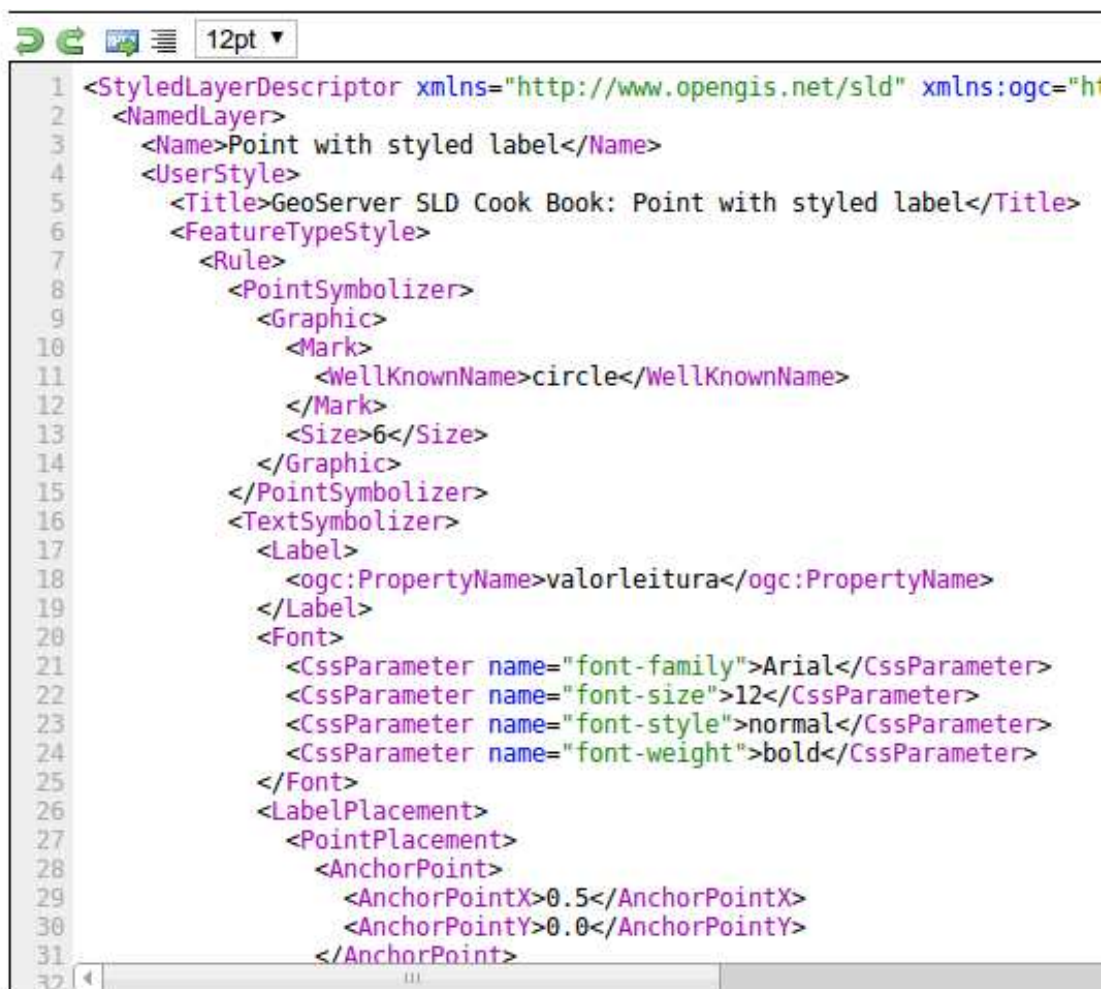
```

7  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
8  <!-- a Named Layer is the basic building block of an SLD document -->
9  <NamedLayer>
10 <Name>default_point</Name>
11 <UserStyle>
12 <!-- Styles can have names, titles and abstracts -->
13 <Title>Default Point</Title>
14 <Abstract>A sample style that draws a point</Abstract>
15 <!-- FeatureTypeStyles describe how to render different features -->
16 <!-- A FeatureTypeStyle for rendering points -->
17 <FeatureTypeStyle>
18 <Rule>
19 <Name>rule1</Name>
20 <Title>Red Square</Title>
21 <Abstract>A 6 pixel square with a yellow fill and no stroke</Abstract>
22 <PointSymbolizer>
23 <Graphic>
24 <Mark>
25 <WellKnownName>square</WellKnownName>
26 <Fill>
27 <CssParameter name="fill">#FFFF00</CssParameter>
28 </Fill>
29 </Mark>
30 <Size>6</Size>
31 </Graphic>
32 </PointSymbolizer>
33 </Rule>
34 </FeatureTypeStyle>
35 </UserStyle>
36 </NamedLayer>
37 </StyledLayerDescriptor>
38

```

Figura 27. Estilo usado para ponto de cor amarela

Outro estilo utilizado foi o Ponto com Texto, onde, ao invés de mostrar o ponto, é exibido o valor lido pelo sensor ou o valor informado pelo laboratório. A definição desse estilo no GeoServer é demonstrada na figura 28.

A screenshot of a code editor window. The window title bar shows a green icon, a blue icon, a menu icon, and a font size dropdown set to '12pt'. The code is XML for a Styled Layer Descriptor (SLD). It defines a layer named 'Point with styled label'. The layer has a title 'GeoServer SLD Cook Book: Point with styled label'. It features a 'FeatureTypeStyle' with a 'Rule' containing two symbolizers: a 'PointSymbolizer' and a 'TextSymbolizer'. The 'PointSymbolizer' uses a 'Graphic' with a 'Mark' of type 'circle' and a size of 6. The 'TextSymbolizer' uses a 'Label' with the property name 'valorleitura'. The label's font is defined with 'font-family' as 'Arial', 'font-size' as '12', 'font-style' as 'normal', and 'font-weight' as 'bold'. The label placement is set to 'PointPlacement' with an 'AnchorPoint' of (0.5, 0.0).

```
1 <StyledLayerDescriptor xmlns="http://www.opengis.net/sld" xmlns:ogc="ht
2   <NamedLayer>
3     <Name>Point with styled label</Name>
4     <UserStyle>
5       <Title>GeoServer SLD Cook Book: Point with styled label</Title>
6       <FeatureTypeStyle>
7         <Rule>
8           <PointSymbolizer>
9             <Graphic>
10              <Mark>
11                <WellKnownName>circle</WellKnownName>
12              </Mark>
13              <Size>6</Size>
14            </Graphic>
15          </PointSymbolizer>
16          <TextSymbolizer>
17            <Label>
18              <ogc:PropertyName>valorleitura</ogc:PropertyName>
19            </Label>
20            <Font>
21              <CssParameter name="font-family">Arial</CssParameter>
22              <CssParameter name="font-size">12</CssParameter>
23              <CssParameter name="font-style">normal</CssParameter>
24              <CssParameter name="font-weight">bold</CssParameter>
25            </Font>
26            <LabelPlacement>
27              <PointPlacement>
28                <AnchorPoint>
29                  <AnchorPointX>0.5</AnchorPointX>
30                  <AnchorPointY>0.0</AnchorPointY>
31                </AnchorPoint>
32
```

Figura 28. Definição do estilo usado para ponto com os valores lidos

Outro estilo criado foi o PolígonoRegiao, criado para demonstrar a região a ser trabalhada no projeto. A definição desse estilo no GeoServer pode ser vista na figura 29.

```

5  xmlns:ogc="http://www.opengis.net/ogc"
6  xmlns:xlink="http://www.w3.org/1999/xlink"
7  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
8  <!-- a Named Layer is the basic building block of an SLD document -->
9  <NamedLayer>
10   <Name>default_polygon</Name>
11   <UserStyle>
12     <!-- Styles can have names, titles and abstracts -->
13     <Title>Default Polygon</Title>
14     <Abstract>A sample style that draws a polygon</Abstract>
15     <!-- FeatureTypeStyles describe how to render different features -->
16     <!-- A FeatureTypeStyle for rendering polygons -->
17     <FeatureTypeStyle>
18       <Rule>
19         <Name>rule1</Name>
20         <Title>Gray Polygon with Black Outline</Title>
21         <Abstract>A polygon with a gray fill and a 1 pixel black outline</Abstract>
22         <PolygonSymbolizer>
23           <Fill>
24             <CssParameter name="fill">#593B90</CssParameter>
25           </Fill>
26           <Stroke>
27             <CssParameter name="stroke">#000000</CssParameter>
28             <CssParameter name="stroke-width">1</CssParameter>
29           </Stroke>
30         </PolygonSymbolizer>
31       </Rule>
32     </FeatureTypeStyle>
33   </UserStyle>
34 </NamedLayer>
35 </StyledLayerDescriptor>
36

```

Figura 29. Definição do estilo PoligonoRegiao no GeoServer

Por último foi criado para esse projeto o estilo Mapa Térmico (*Heatmap*), que corresponde a um mapa onde os valores individuais contidos em uma matriz são representados como cores. Logo, neste estilo, o GeoServer converte os dados armazenados no PostGIS em formato vetorial e converte em dados raster para então criar o espectro de cores de dez níveis, do amarelo ao vermelho. O amarelo mais claro representa os menores valores e o vermelho corresponde aos maiores valores.

Para a criação desse estilo é definido o percentual do valor em relação à cor desejada para que o GeoServer possa processar a camada desejada e exibir o mapa na escala de cores correspondente. Essa escala de cores pode ser vista na figura 30.

```

60         <Opacity>0.6</Opacity>
61         <ColorMap type="ramp" >
62 <ColorMapEntry color= "#593B90" quantity="0" label="0" opacity="0"/>
63 <ColorMapEntry color= "#593B90" quantity=".1" label="10%" opacity="0"/>
64 <ColorMapEntry color= "#374395" quantity=".2" label="20%" opacity="0" />
65 <ColorMapEntry color= "#32719A" quantity=".3" label="30%" opacity="0"/>
66 <ColorMapEntry color= "#2C9F95" quantity=".4" label="40%" />
67 <ColorMapEntry color= "#26A45D" quantity=".5" label="50%" />
68 <ColorMapEntry color= "#25AA1F" quantity=".6" label="60%" />
69 <ColorMapEntry color= "#66AF18" quantity=".7" label="70%" />
70 <ColorMapEntry color= "#B4B410" quantity=".8" label="80%" />
71 <ColorMapEntry color= "#B96508" quantity=".9" label="90%" />
72 <ColorMapEntry color= "#BF0800" quantity="1.0" label="100%" />
73
74         </ColorMap>
75     </RasterSymbolizer>
76 </Rule>

```

Figura 30. Definição no GeoServer das cores para o mapa térmico

Assim sendo, a partir desse estilo, o programa é capaz de exibir qualquer camada do banco de dados como um mapa térmico, não só a camada de produtividade.

A figura 31 mostra um estilo de cada, que foi criado no GeoServer, seguindo a ordem de cima para baixo, esquerda para direita: Ponto, Região, Ponto com Texto e Heatmap.

Há de se destacar que o GeoServer é somente um servidor de mapas que faz a conversão entre os dados geográficos armazenados no banco de dados, o processamento e a entrega dos mapas para serem visualizados através de um serviço de disponibilização de mapas. Neste trabalho, foi usado o WMS (Web Map Service - Serviço de Mapas da Web), que é um padrão livre de disponibilização de mapas pela rede. Esse padrão é facilmente disponibilizado pelo GeoServer, que deve ser habilitado na área de trabalho.

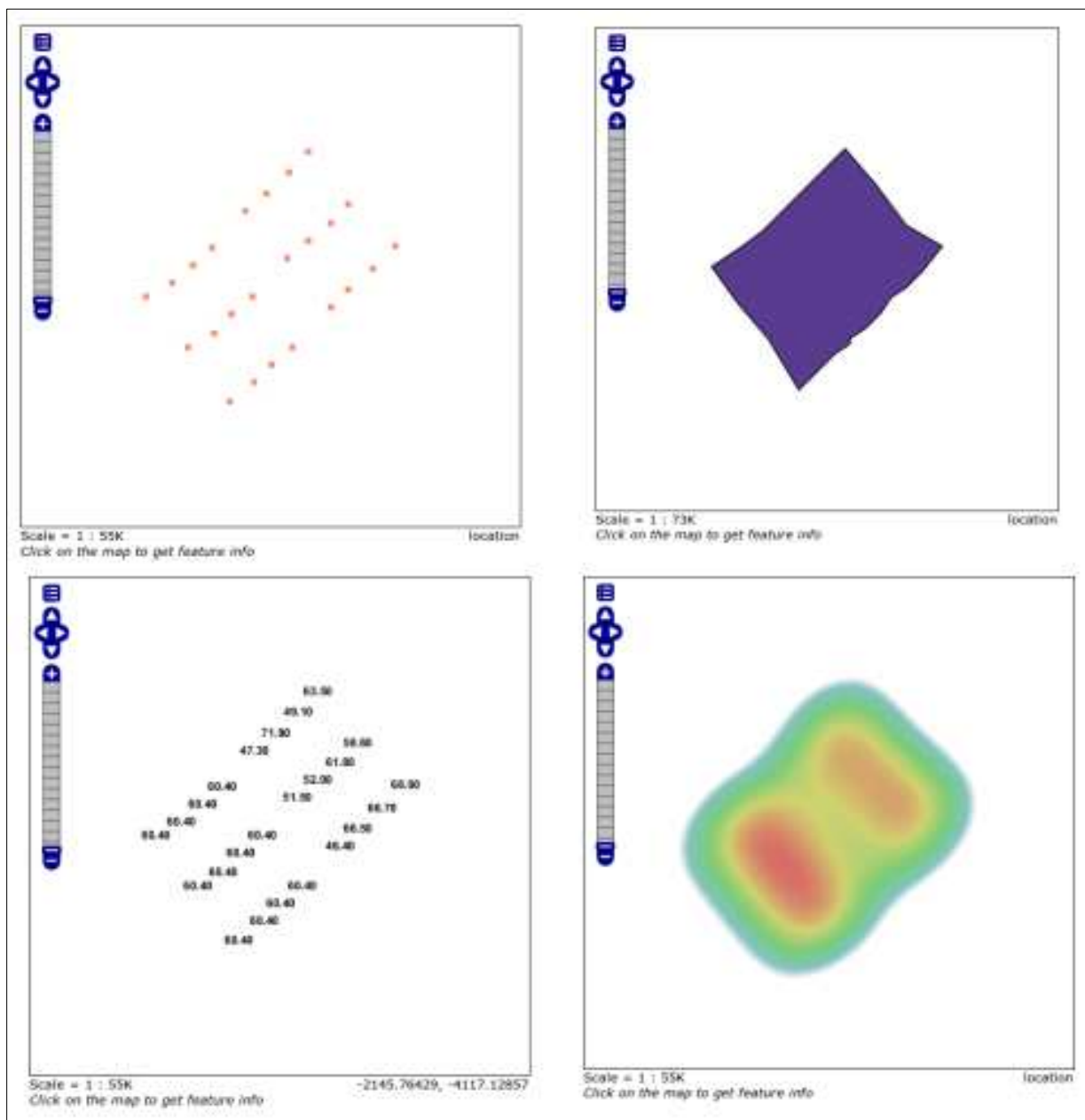


Figura 31. Exemplo de estilos de mapas criados no GeoServer

A aplicação constante desta pesquisa foi desenvolvida usando diversas tecnologias: Java, Hibernate, Hibernate Espacial, JSF e OpenLayers. O Java corresponde à linguagem de programação que é responsável não somente pela implementação de todos os procedimentos e as classes de negócio como também pela integração de todas essas ferramentas, uma vez que todas as tecnologias utilizadas foram desenvolvidas em Java, facilitando, portanto, esse procedimento.

Com a biblioteca OpenLayers, foram desenvolvidas três telas: uma para a visualização da região, outra para a visualização de mapas exibindo os pontos (sensores lidos) e, uma última capaz de visualizar os mapas térmicos. O OpenLayers faz uma requisição via WMS ao GeoServer que devolve a ele o WMS referente ao mapa processado. Assim, cabe ao OpenLayers renderizar os dados do WMS. Além da renderização, o OpenLayers permite uma comunicação direta com o GeoServer através de comandos, tais como zoom, visualização dos dados não geográficos para cada camada (apenas clicando no mapa), visualização em miniatura para melhor navegação pelo mapa, visualização da latitude e longitude por onde o cursor passa.

Outras duas tecnologias usadas foram o Hibernate e o Hibernate Espacial. O Hibernate realiza o mapeamento entre o programa e o banco de dados dos dados não geográficos e o Hibernate Espacial dos dados geográficos. A localização região (Polígono) e a localização leitura (Ponto), conforme pode ser visto na figura 25, são mapeadas pelo Hibernate Espacial, enquanto que os demais dados são mapeados no Hibernate. Desta forma, o uso dos dados se torna transparente para o programa. O mapeamento das classes usando o Hibernate e o Hibernate Espacial pode ser visto no Apêndice A.

O *Java Servlet Faces* (JSF) é o framework usado para toda implementação da interface gráfica para o usuário. Todas as telas desenvolvidas (cadastros, entradas de dados e legendas dos mapas), com exceção dos mapas, foram desenvolvidas usando esta tecnologia.

3.2.3. As funcionalidades do WebFazenda

O Webfazenda foi desenvolvido conforme as funcionalidades que podem ser vistas no diagrama de caso de uso da figura 32.

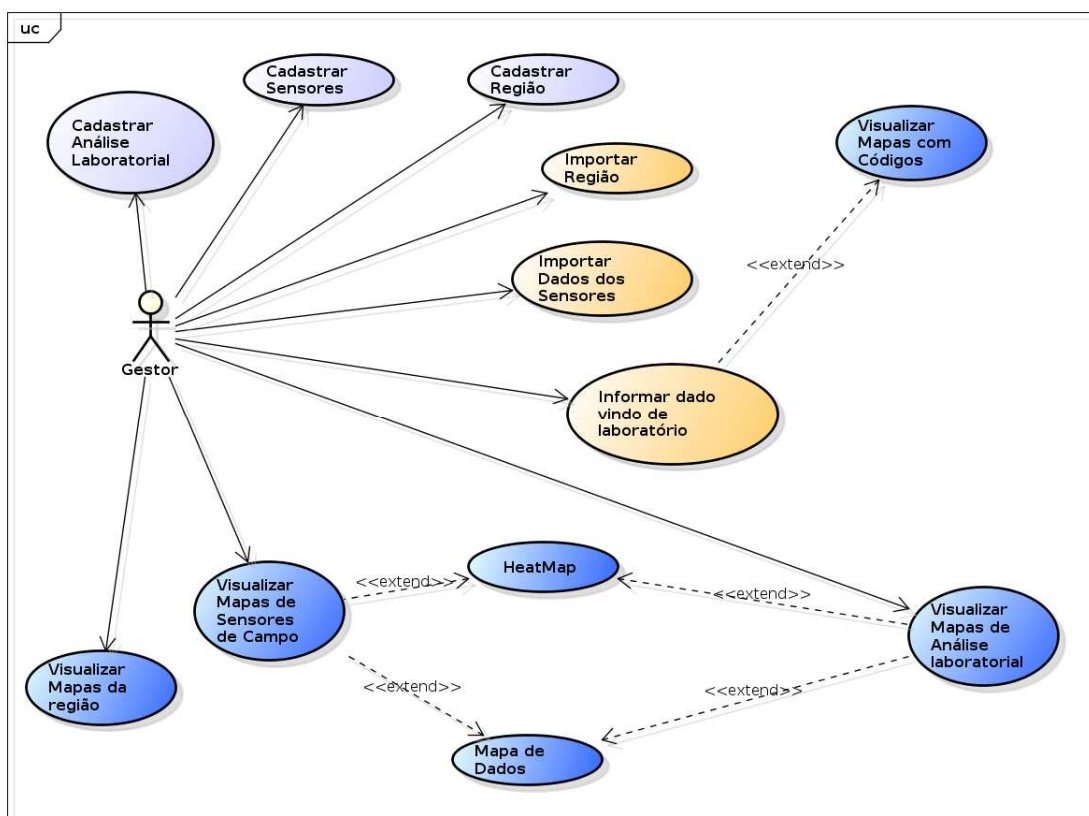


Figura 32. Diagrama de Caso de Uso – WebFazenda

Em um primeiro momento, o ator terá que cadastrar a região em que irá fazer todo o trabalho. Nesse cadastro o sistema disponibiliza a informação do nome da região e, logo em seguida, deve informar o arquivo NMEA que virá do GPS usado para a determinação da área a ser controlada. O padrão NMEA está disponível em todos os aparelhos de GPS por ser um padrão aberto.

O arquivo NMEA apresenta diversas linhas codificadas com a primeira palavra com códigos que representam o que os dados naquela linha trazem.

Para a construção da região é necessária a obtenção das latitudes e longitudes dos mais diversos pontos que constituem o seu polígono. Assim sendo, o programa pega a linha de código \$GPGGA (Sistema de Posicionamento Global e Data) e descarta todas as demais. Essa linha traz a seguinte sequência de dados, separados por vírgula: hora utc, latitude, indicador Norte ou Sul da latitude,

longitude, indicador leste ou oeste da longitude, posição fixa, satélites usados, HDOP, Altitude, Unidade da Altitude, Geoid, Unidade Geoid, idade DGPS, identificador da estação do DGPS e código de verificação (*checksum*).

Com isso, o programa extrai os dados separados por vírgula em um vetor e retira desse vetor somente a latitude, norte ou sul, e a longitude, leste ou oeste. Cada valor extraído é repassado para o Hibernate Espacial converter em um dado do tipo ponto. Essa operação é realizada até o final do arquivo obtendo assim, vários pontos, conforme foi a aquisição dos dados pelo GPS em campo.

Ao final do arquivo, o programa seleciona todos os pontos e, novamente, os passa para o Hibernate Espacial, através da função *GeometryFactory* (Fábrica de Geometrias), que converte todos os pontos em um polígono que é então gravado no banco de dados. Ao gravar, o programa passa para o GeoServer via WMS, a criação de uma camada visual baseada na linha criada na tabela Região do banco, usando o nome da região como sendo o nome da camada (*Layer*) e o estilo Polígono. O fluxograma que descreve a criação da Região no programa pode ser visto na figura 33.

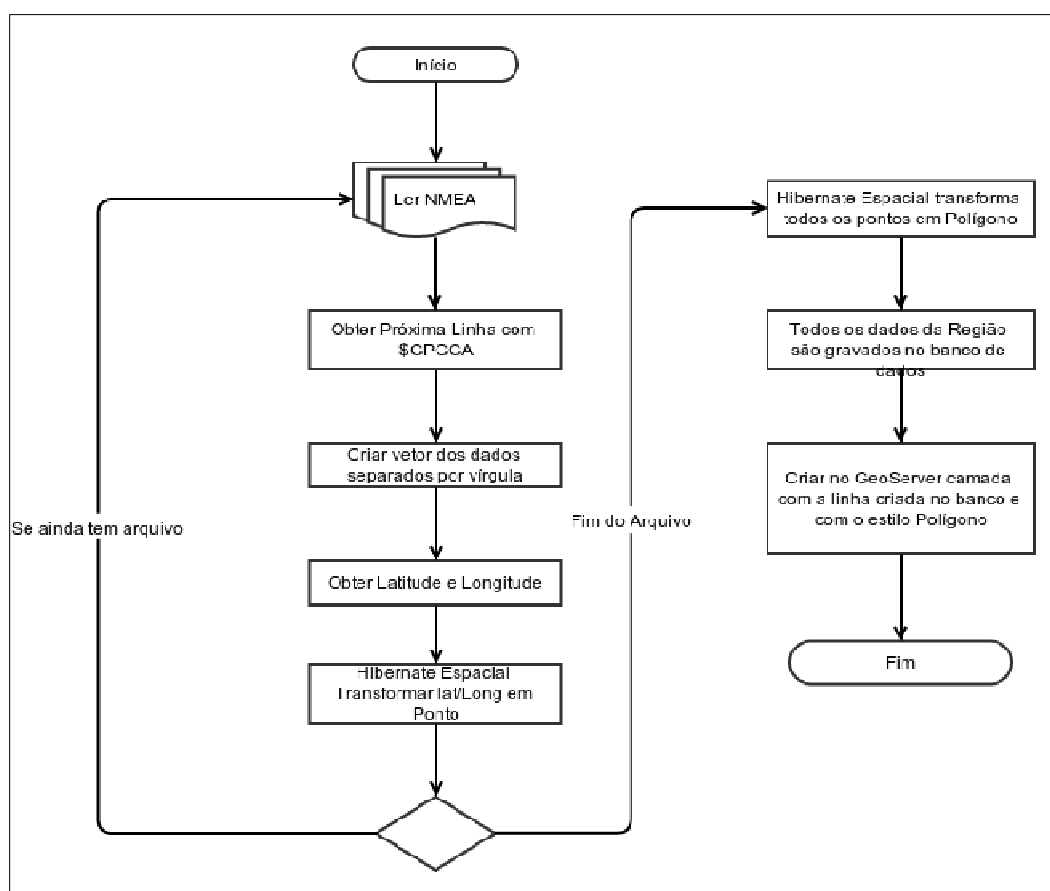


Figura 33. Fluxograma de criação de uma região no WebFazenda

Para o cadastro de sensores, o programa envolve dois casos de uso para inclusão de dados: Cadastrar sensores e Importar Dados dos Sensores. O primeiro é uma tela, feita somente com JSF e Hibernate, onde é disponibilizado ao usuário somente a inclusão de dados não geográficos (nome, detalhes, se o sensor é ou não analógico e a cor de representação do mapa). No segundo caso de uso, é disponibilizada uma listagem de sensores pré-cadastrados, com JSF. O usuário, ao escolher um sensor, é exibido uma lista de regiões e é disponibilizada a opção de escolha do arquivo gerado pelo sistema de aquisição de dados no campo, que estará no cartão de memória. Até aqui o caso de uso é todo implementado em JSF com Hibernate. O “importar” será responsável por ler o arquivo disponibilizado pelo sistema de aquisição de dados e armazenar no banco de dados. Neste momento, volta-se à integração das tecnologias: JSF, Hibernate e Hibernate Espacial.

O arquivo disponibilizado pelo sistema de aquisição de dados (DATALOG.TXT) segue a seguinte sequência: latitude, longitude, era, data hora, altitude, valor lido do sensor. Todos os dados são separados por ponto e vírgula. A latitude e longitude já são gravadas com os sinais (positivo para norte e leste e negativo para sul e oeste) e o valor lido do sensor já está convertido para a unidade de interesse pelo usuário quando da programação do sistema de aquisição de dados.

Este caso de uso, então, irá percorrer cada linha desse arquivo e vai gerar um ponto com o sensor escolhido no banco de dados com o valor lido. O procedimento de trabalho deste algoritmo pode ser visto no fluxograma apresentado na figura 34.

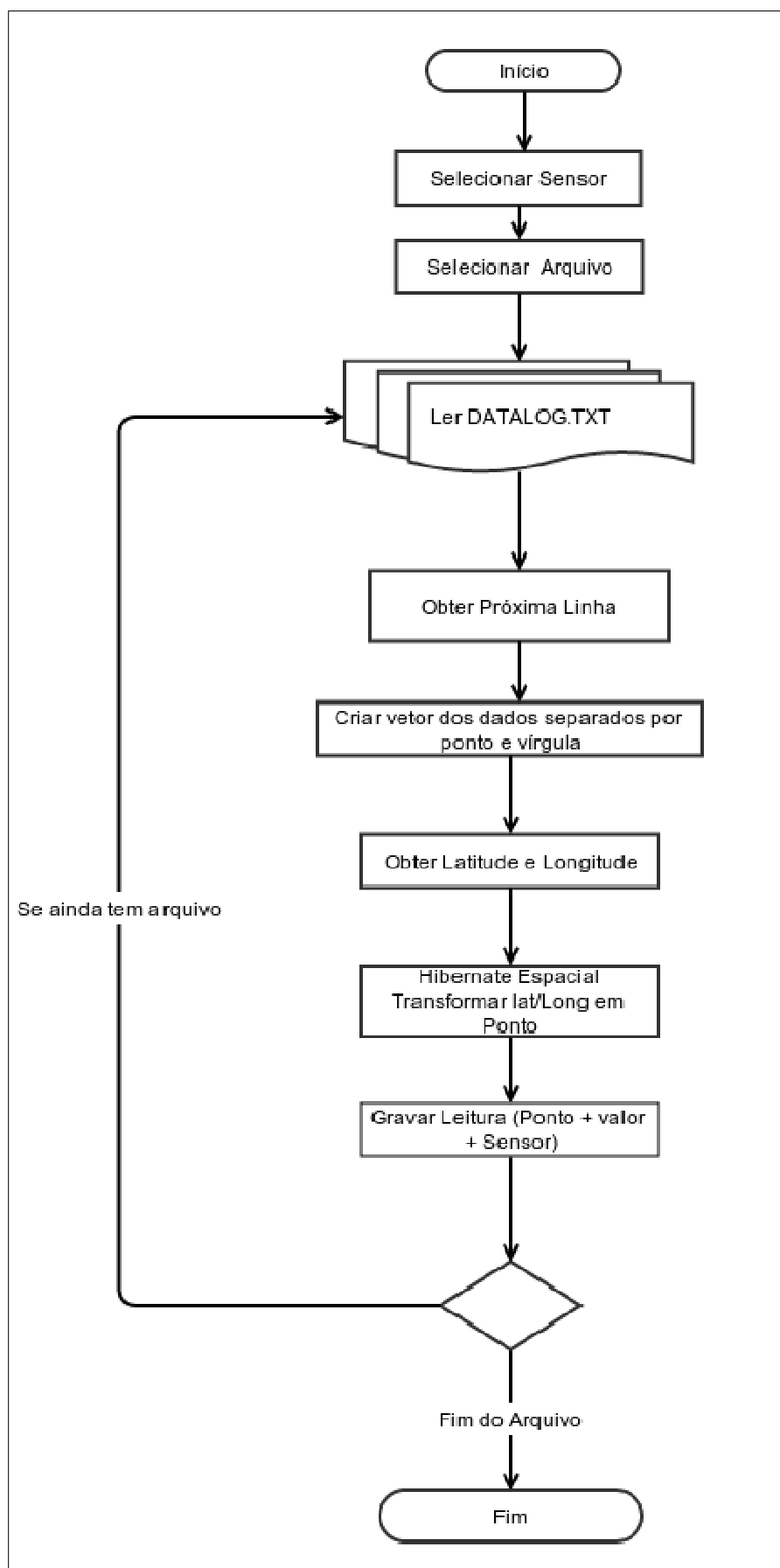


Figura 34. Fluxograma de criação de uma região no WebFazenda

Os dados laboratoriais ocorrem quando o material é coletado no campo e enviado para um laboratório para análise e envio dos valores obtidos. Desta forma, em um primeiro momento o sistema disponibiliza a possibilidade de cadastro dos dados que serão analisados em laboratório. Para esse caso de uso (Cadastrar Análise Laboratorial), foi usado somente JSF e Hibernate para que seja possível a informação de cada dado que virá na análise.

Com os dados cadastrados, o usuário terá que importar os pontos onde ocorreram as coletas de dados do campo. Essa importação corresponde, na prática, à opção de importação dos pontos do GPS diretamente. Assim, a leitura é direta do arquivo no formato NMEA disponibilizado no GPS de preferência do usuário. O fluxograma que descreve como funciona o caso de uso importar dados do laboratório pode ser visto na figura 35.

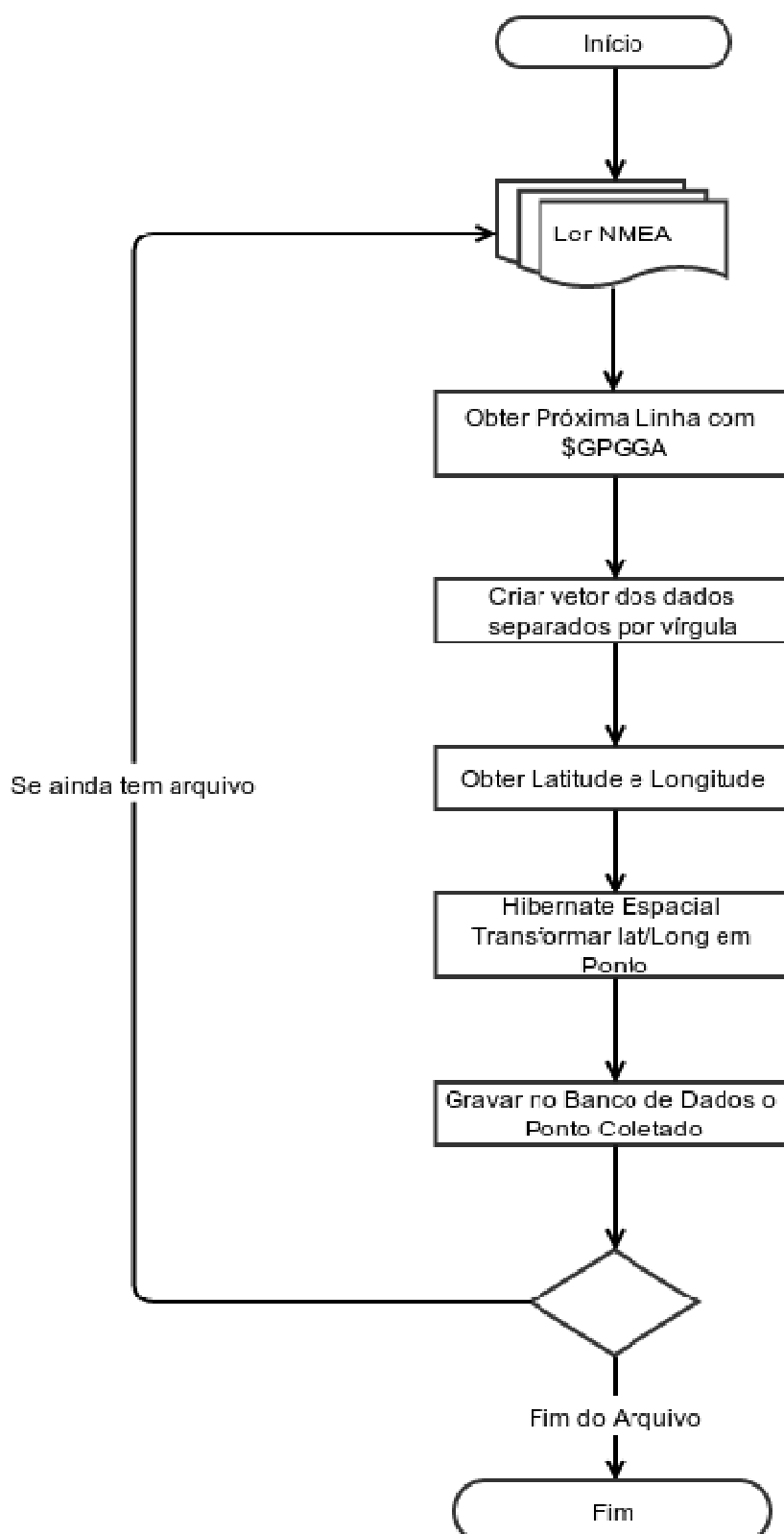


Figura 35. Fluxograma de importação de leitura de dados de laboratório do campo no WebFazenda

O próximo passo é a informação dos dados obtidos pelo laboratório. Para facilitar esse passo, o programa disponibiliza um mapa (desenvolvido em Openlayers) que acessa via WMS o GeoServer passando a camada desejada (cada dado laboratorial é criado como uma camada no GeoServer) e exibindo o mapa com a Região e os Pontos com os Códigos (estilo Ponto com Texto do GeoServer). Então, o usuário consegue visualmente saber os códigos utilizados para a digitação dos dados obtidos do laboratório.

Para a digitação dos dados, uma tela foi desenvolvida com JSF e Hibernate com uma tabela que facilita ao usuário a inclusão de todos os dados de uma única vez. As telas de informação dos dados laboratoriais podem ser visualizadas na figura 36. Da esquerda para a direita, de cima para baixo tem-se a tela de cadastro de dados laboratoriais, a tela de importação de dados do GPS, o mapa de visualização dos códigos no campo e, por último, a tela para a digitação dos dados vindos do laboratório.

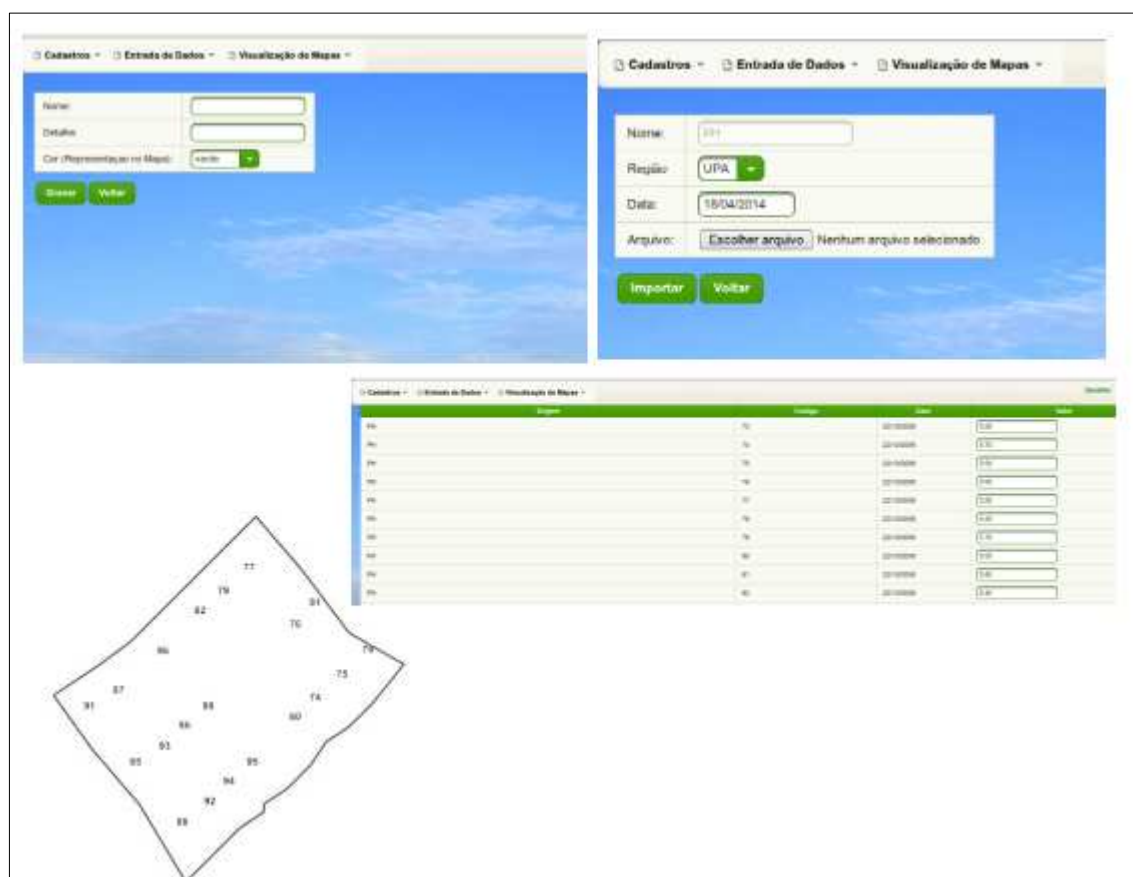


Figura 36. Telas necessárias para a inclusão de dados laboratoriais no WebFazenda

O caso de uso mapa de região lista as regiões cadastradas, através do JSF e do Hibernate e permite ao usuário a visualização do mapa da região que foi importada através dos dados do NMEA. É a solicitação da camada com o nome da região para o GeoServer e a exibição desse mapa (em forma de polígono) pelo OpenLayers. Esses recursos podem ser vistos na figura 37.

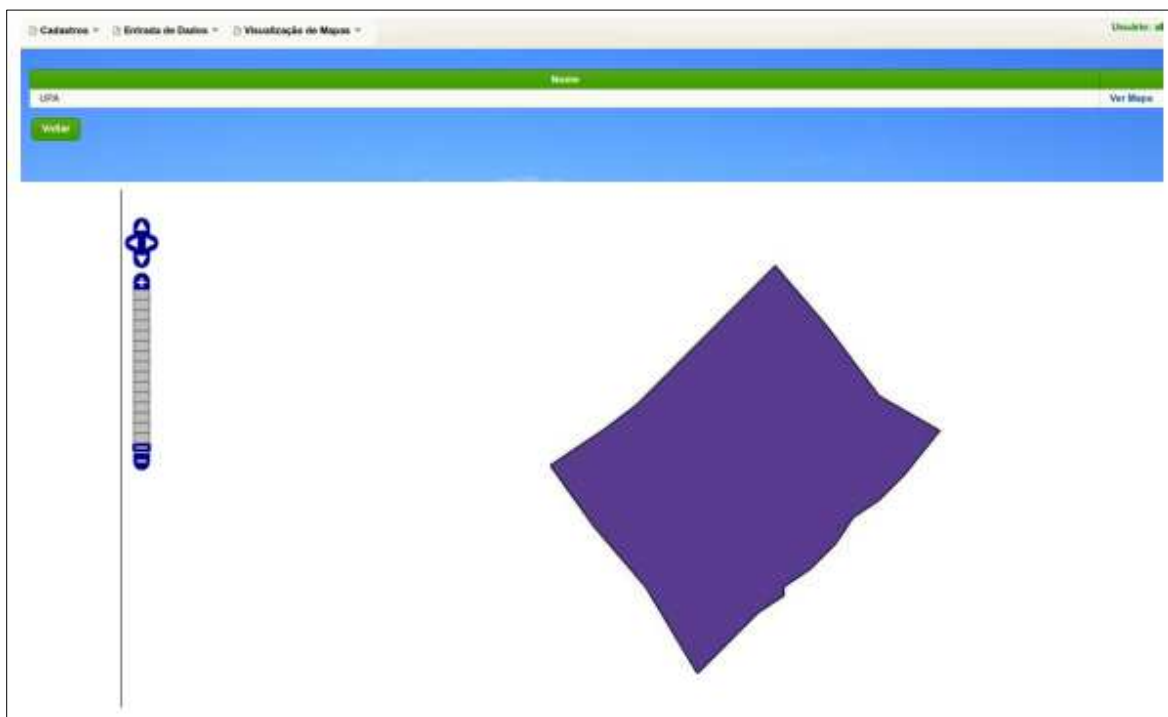


Figura 37. Telas de Visualização de Região no WebFazenda

Os mapas de sensores de campo também podem ser escolhidos a partir de uma lista disponibilizada pelo JSF + Hibernate. Neste caso, dois tipos de mapas podem ser escolhidos: o mapa com os pontos lidos ou o Heatmap (Mapa Térmico).

Os mapas são os mesmos, mas quando o OpenLayers faz a solicitação ao Geoserver passa qual o estilo desejado para a visualização do mapa (ponto ou HeatMap), a partir dessa escolha o geoServer devolve para o OpenLayers o mapa com o estilo desejado que é renderizado pelo OpenLayers. No caso do mapa térmico, o GeoServer não devolve uma legenda com os valores e as respectivas cores que representam dentro do mapa, ele devolve somente uma legenda com o percentual relativo e a cor. Assim sendo, foi desativada a legenda do GeoServer e a legenda foi criada em JSF, onde o programa calcula os percentuais dos valores possíveis, exibindo assim os valores correspondentes na

legenda. O mapa térmico foi criado com valores de dez em dez por cento. Logo, busca-se o maior valor e vai extraindo 10% dele. Todos os recursos de visualização de mapas para sensores podem ser visualizados na figura 38.

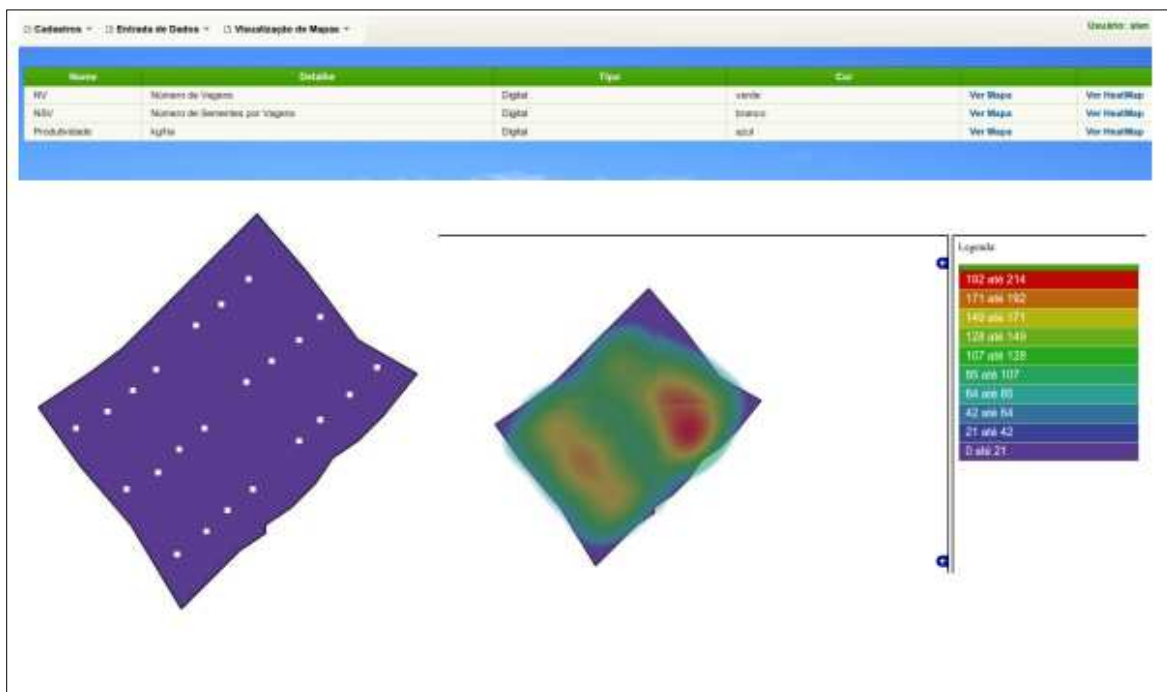


Figura 38. Telas de Visualização de Sensores no WebFazenda

Quanto a mapas de dados laboratoriais, são idênticos aos mapas disponibilizados para sensores, conforme pode ser visto na figura 39.

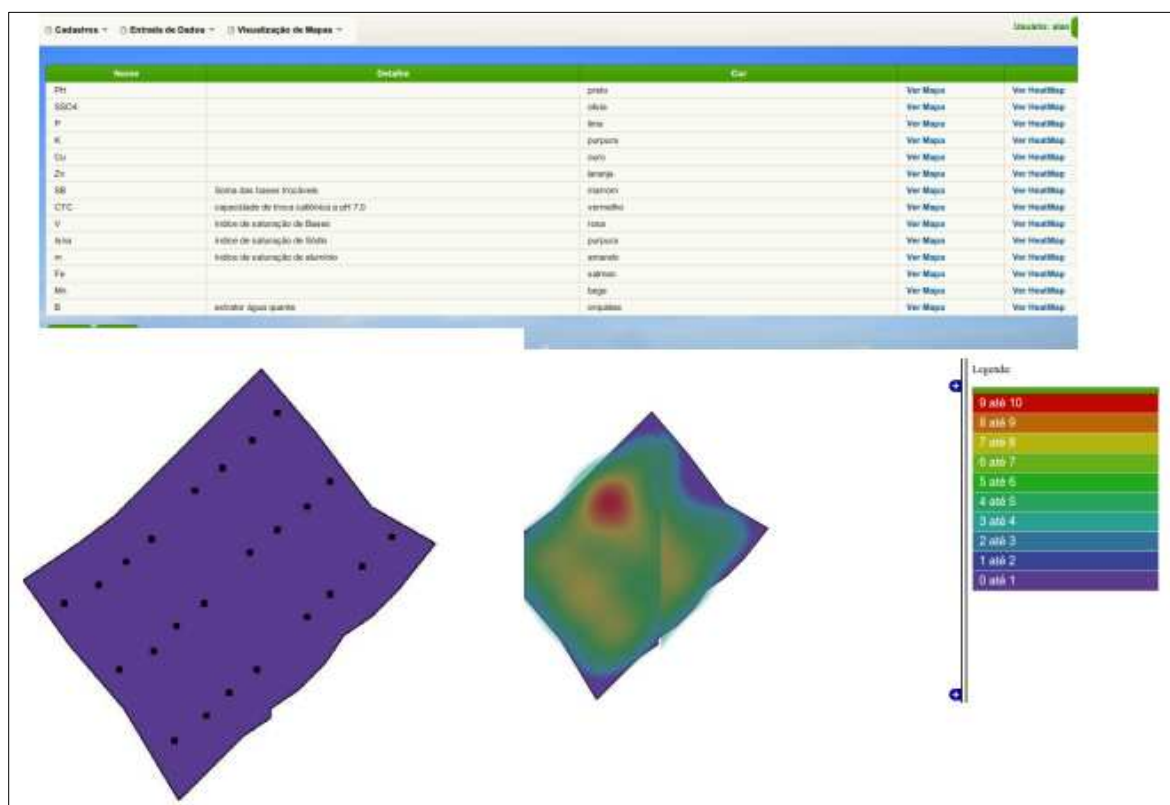


Figura 39. Telas de Visualização de Dados de Laboratório no WebFazenda

4. RESULTADOS E DISCUSSÃO

4.1. Funcionamento do WebFazenda

O projeto foi desenvolvido na Unidade de Apoio à Pesquisa (UAP) em uma área do Centro de Ciências e Tecnologias Agropecuárias da Universidade Estadual do Norte Fluminense Darcy Ribeiro, “em uma área com 50 x 60 m, sendo esta dividida em duas áreas de 27 m x 47 m. Em cada área, foram demarcadas e georeferenciadas doze parcelas de 15 m x 6 m, com 1 m separando as parcelas” (JUNIOR, 2009). A figura 40 representa a área usada no projeto.



Figura 40. Área experimental destacada em vermelho, localizada na UAP da UENF, Campos dos Goytacazes (JUNIOR, 2009)

Para o projeto, em um primeiro momento, foi obtido o arquivo gerado pelo GPS em formato NMEA para a criação da região no WebFazenda. O GPS usado foi um Garmin GPSmap 60CSx. Infelizmente, o fabricante do GPS utilizado não dá “liberdade” de acesso aos arquivos do equipamento. Existem programas disponíveis na internet para efetuar a leitura direta dos mapas. Entretanto, neste projeto, foi adotado o uso do próprio programa disponibilizado pelo fabricante (MapSource) para ler o dispositivo e gerar o arquivo em formato NMEA. Cada fabricante tem sua forma de trabalho. A proposta deste trabalho não é modificar a forma como os dispositivos vão funcionar, mas permitir uma interface única de trabalho com todos os dispositivos.

Assim sendo, usando o MapSource, foi lido e gerado o arquivo NMEA (figura 41) da região.

```

$GPGGA,203139.389,,,,,0,00,,M,0.0,M,,0000*5E
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPRMC,203139.389,V,,,,,040913,,N*4A
$GPGGA,203140.389,,,,,0,00,,M,0.0,M,,0000*50
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPRMC,203140.389,V,,,,,040913,,N*44
$GPGGA,203141.391,,,,,0,00,,M,0.0,M,,0000*58
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPRMC,203141.391,V,,,,,040913,,N*4C
$GPGGA,203142.391,,,,,0,00,,M,0.0,M,,0000*5B
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPRMC,203142.391,V,,,,,040913,,N*4F
$GPGGA,203143.389,,,,,0,00,,M,0.0,M,,0000*53
$GPGSA,A,1,,,,,,,,,,,,,*1E
$GPGSV,3,1,12,09,45,189,22,07,40,140,30,23,15,046,,19,15,128,29*7E

```

Figura 41. Arquivo NMEA lido a partir do GPS Garmin GPSmap 60CSx com a área experimental

Posteriormente, esse arquivo foi importado pelo programa em Entrada de Dados, Importar Região. O acesso pode ser visto na figura 42 junto com o processo de importação do arquivo NMEA para o programa WebFazenda, criando a região aqui definida com o nome UPA.

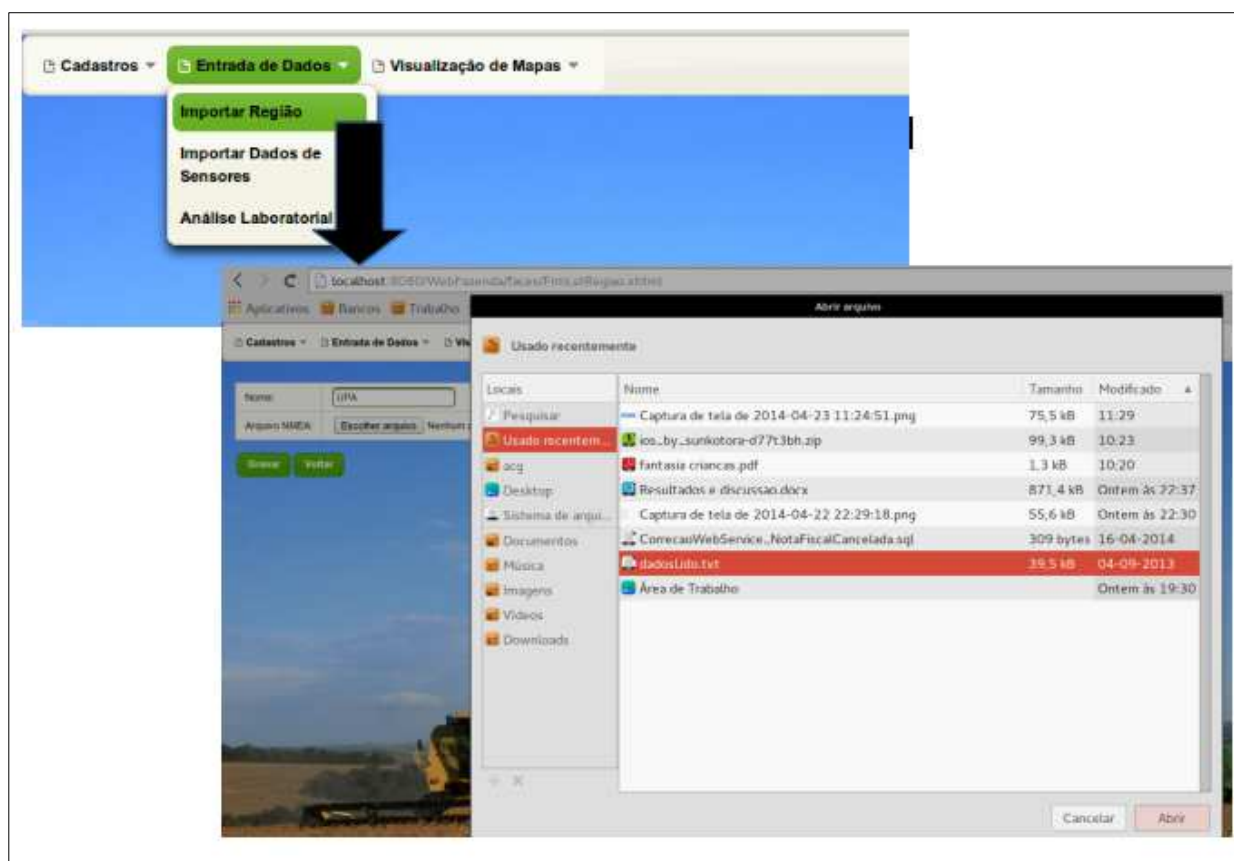


Figura 42. Importação do Arquivo do arquivo NMEA da Região no WebFazenda

O sistema fez a leitura dos dados NMEA e armazenou no banco de dados. O mapa gerado pode ser visto em Visualização de Mapas, Mapa da Região do programa WebFazenda, conforme a figura 43.

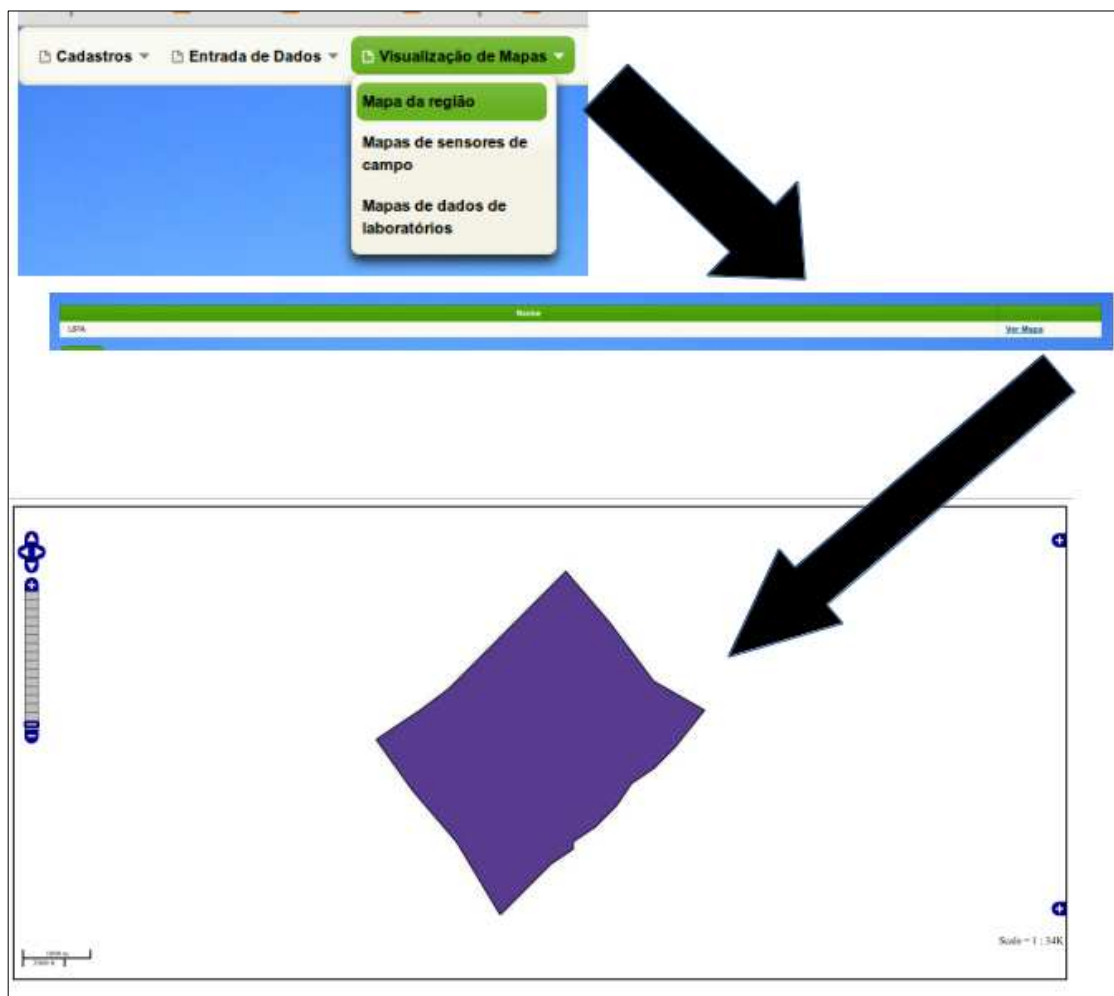


Figura 43. Visualização do mapa da região UAP pelo WebFazenda

No estudo de caso apresentado, foram instalados na região (área experimental) dois sistemas de cultivo de feijão, variedade BR1 Xodó, sendo a região nordeste realizada com um sistema tecnificado, neste caso aplicando técnicas de agricultura de precisão. Na parte sudoeste foi utilizado o sistema tradicional (JUNIOR, 2009).

Como o experimento já havia sido realizado no projeto de Junior (2009), não foi possível a aplicação dos sensores, pois os dados já haviam sido obtidos. Porém, todos os dados existentes foram adaptados para que o WebFazenda os pudesse interpretar. Nesta parte do projeto, nenhum dado foi inserido manualmente; todos os dados foram inseridos através da interface do próprio programa para que a validação fosse realizada da forma mais correta possível.

Na figura 44 pode ser visto o croqui das duas áreas georeferenciadas, onde as repetições R representam o sistema tecnificado e as repetições J representam o sistema tradicional (JUNIOR, 2009).

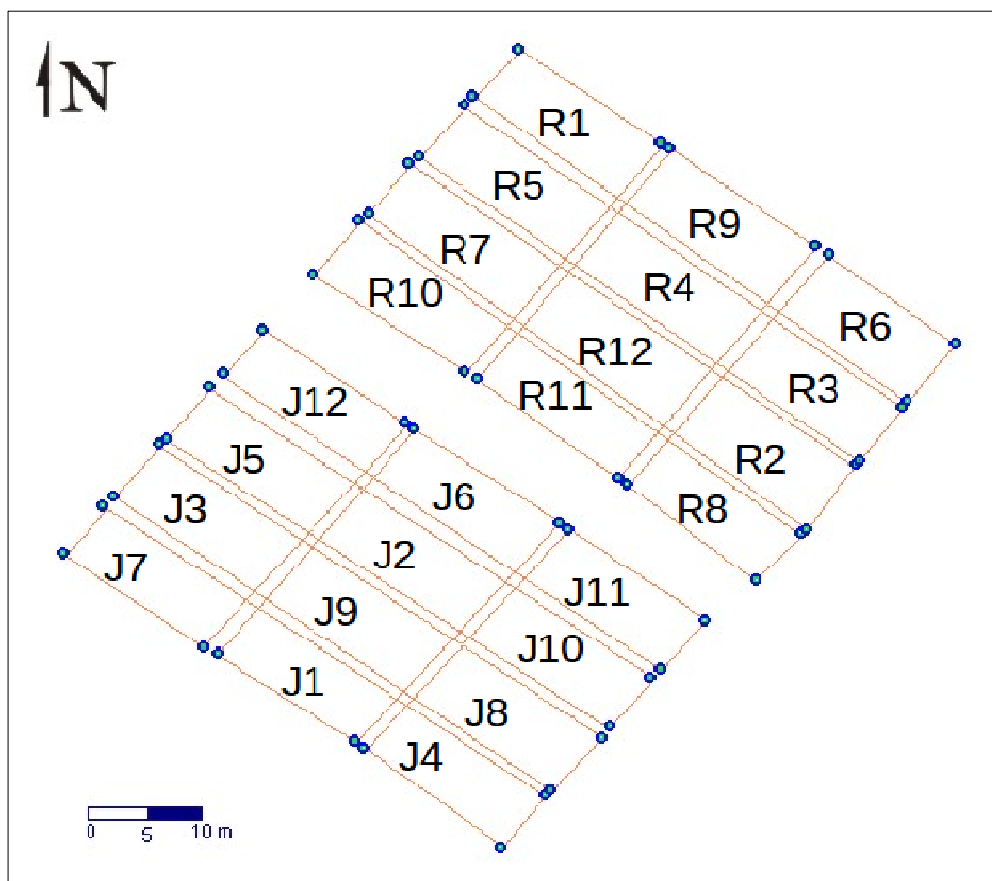


Figura 44. Croqui das áreas georeferenciadas (JUNIOR, 2009)

O arquivo do GPS fornecia os limites de cada repetição. No projeto foi utilizado um valor para cada repetição. Para que cada valor entregue tivesse um valor no mapa, foi realizado o cálculo do centroide de cada repetição para ser usado como referência geográfica dos dados de cada repetição. Esse trabalho foi necessário, pois quando o experimento foi realizado não havia ainda o sistema de aquisição de dados baseado no arduino com o GPS integrado, que facilitaria com apenas a importação dos dados diretamente do cartão de memória do mesmo.

No quadro 5 pode-se ver a tabela de produtividade obtida por repetição.

Quadro 5. Produtividade por repetição da área experimental (JUNIOR, 2009)

R1	306,74	gramas	3411	Kg/ha	J1	191,13	gramas	2613	Kg/ha
R2	336,75	gramas	3747	Kg/ha	J2	135,91	gramas	1564	Kg/ha
R3	330,05	gramas	3581	Kg/ha	J3	118,41	gramas	1855	Kg/ha
R4	294,49	gramas	3280	Kg/ha	J4	183,66	gramas	2101	Kg/ha
R5	306,82	gramas	3240	Kg/ha	J5	141,12	gramas	1546	Kg/ha
R6	312,84	gramas	3305	Kg/ha	J6	128,06	gramas	1612	Kg/ha
R7	314,06	gramas	3513	Kg/ha	J7	146,08	gramas	1963	Kg/ha
R8	324,27	gramas	3543	Kg/ha	J8	138,21	gramas	1659	Kg/ha
R9	336,99	gramas	3709	Kg/ha	J9	163,49	gramas	1999	Kg/ha
R10	287,83	gramas	3163	Kg/ha	J10	144,98	gramas	2194	Kg/ha
R11	343,45	gramas	3856	Kg/ha	J11	197,66	gramas	2652	Kg/ha
R12	321,52	gramas	3810	Kg/ha	J12	157,47	gramas	1809	Kg/ha

Desta forma, foi gerado um arquivo (DATALOG.TXT) no formato esperado pelo WebFazenda, contendo os centroides de cada repetição e os valores da produtividade. O arquivo gerado está apresentado na figura 45.

1	-2145.77525; -4117.14575; ;22/10/08 10:30;59.9;306.74
2	-2145.7695; -4117.1655; ;23/10/08 10:30;59.9;336.75
3	-2145.766; -4117.1625; ;24/10/08 10:30;59.9;330.05
4	-2145.772; -4117.156; ;25/10/08 10:30;59.9;294.49
5	-2145.778; -4117.14875; ;26/10/08 10:30;59.9;306.82
6	-2145.76275; -4117.15925; ;27/10/08 10:30;59.9;312.84
7	-2145.78125; -4117.15175; ;28/10/08 10:30;59.9;314.06
8	-2145.772; -4117.168; ;29/10/08 10:30;59.9;324.27
9	-2145.7695; -4117.15325; ;30/10/08 10:30;59.9;336.99
10	-2145.78425; -4117.15425; ;31/10/08 10:30;59.9;287.83
11	-2145.77825; -4117.161; ;01/11/08 10:30;59.9;343.45
12	-2145.77525; -4117.1585; ;02/11/08 10:30;59.9;321.52
13	-2145.7925; -4117.17375; ;03/11/08 10:30;59.9;191.13
14	-2145.78625; -4117.169; ;04/11/08 10:30;59.9;135.91
15	-2145.79475; -4117.1645; ;05/11/08 10:30;59.9;118.41
16	-2145.7865; -4117.1815; ;06/11/08 10:30;59.9;183.66
17	-2145.79175; -4117.162; ;07/11/08 10:30;59.9;141.12
18	-2145.78325; -4117.1665; ;08/11/08 10:30;59.9;128.06
19	-2145.7985; -4117.1665; ;09/11/08 10:30;59.9;146.08
20	-2145.783; -4117.17875; ;10/11/08 10:30;59.9;138.21
21	-2145.78875; -4117.17175; ;11/11/08 10:30;59.9;163.49
22	-2145.7805; -4117.17625; ;12/11/08 10:30;59.9;144.98

Figura 45. DATALOG.TXT com dados da Produtividade

Com o arquivo DATALOG.TXT dos dados da produtividade no mesmo formato que seria disponibilizado pelo sistema de aquisição de dados, o próximo passo foi importar os dados no programa. Antes de importar foi necessário cadastrar o sensor (aqui chamado de produtividade – não é um sensor, mas o que seus dados representam e o nome foi mantido para ter maior compatibilidade

com o projeto de JUNIOR, 2009). Isso ocorre em Cadastro, Sensores. O cadastro da produtividade pode ser visto na figura 46.



Nome:	Produtividade
Detalhe:	kg/ha
Analógico (se falso, digital):	<input checked="" type="checkbox"/>
Cor (Representação no Mapa):	azul

Gravar Voltar

Figura 46. Cadastro de Sensores para Importação de Dados

Após o cadastro do sensor, ficou disponível a importação dos dados dos sensores em Entrada de Dados, Importar Dados de Sensores. Nesta tela o usuário escolheu a Região (UPA) e o arquivo (DATALOG.TXT). O programa leu e gravou os dados e gerou os mapas necessários para esse sensor, conforme pode ser visto nas figuras 47 (mapa de pontos colhidos) e 48 (Heatmap – Mapa térmico).

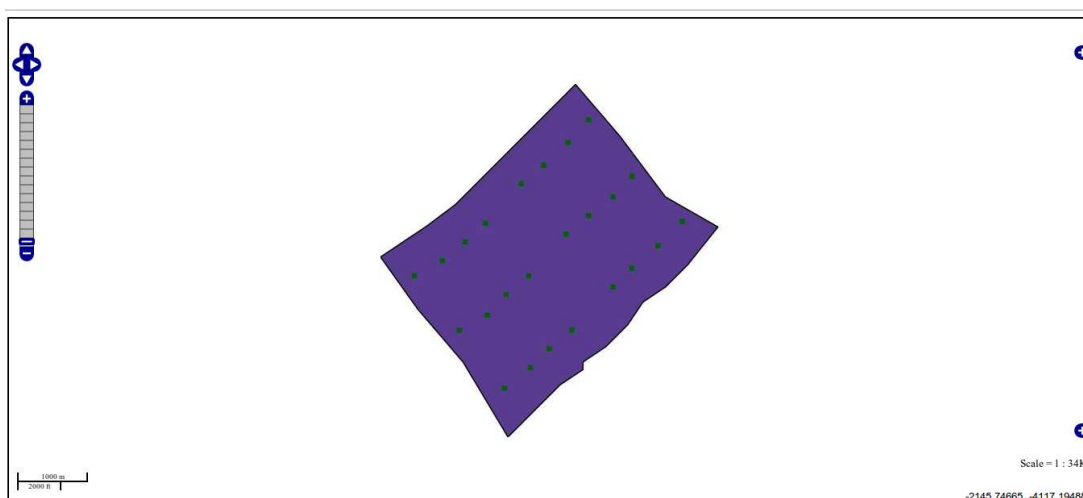


Figura 47. Mapa de Pontos colhidos pelos sensores

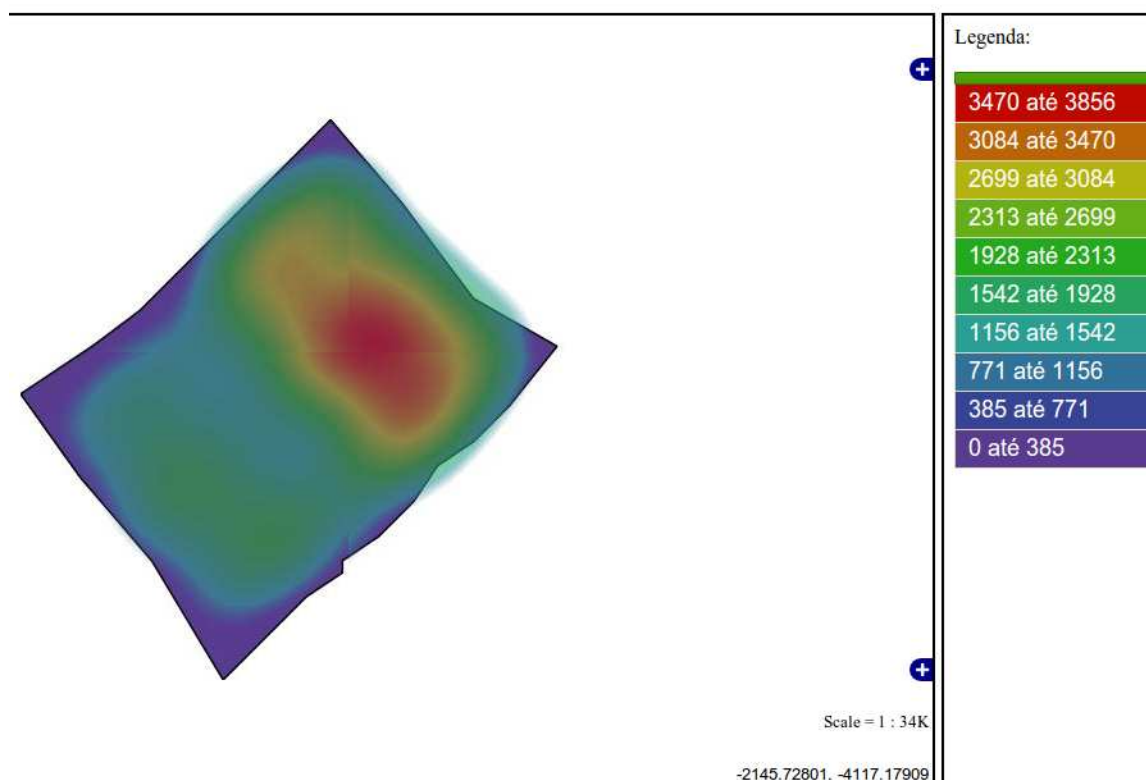


Figura 48. Mapa temático

O mapa de pontos serve apenas para visualizar as posições onde os dados foram colhidos, ou seja, no centroide das áreas experimentais. Por outro lado, o Mapa Térmico mostra nitidamente a diferença da área tecnificada para a área não tecnificada, com destaque as repetições 2,3,4,8,11 e 12 (figura 44) em que a produtividade chegou a valores de 3084 a 3856 kg ha⁻¹. Por outro lado as repetições não tecnificadas não passaram de 2699 kg ha⁻¹.

No trabalho de JUNIOR (2009) foram gerados os mapas de produtividade utilizando o programa Surfer 8.0, pelo método de krigagem simples. Desta forma, foram gerados dois mapas: um mapa de produtividade da área tecnificada (figura 49) e um mapa de produtividade da área não tecnificada (figura 51).

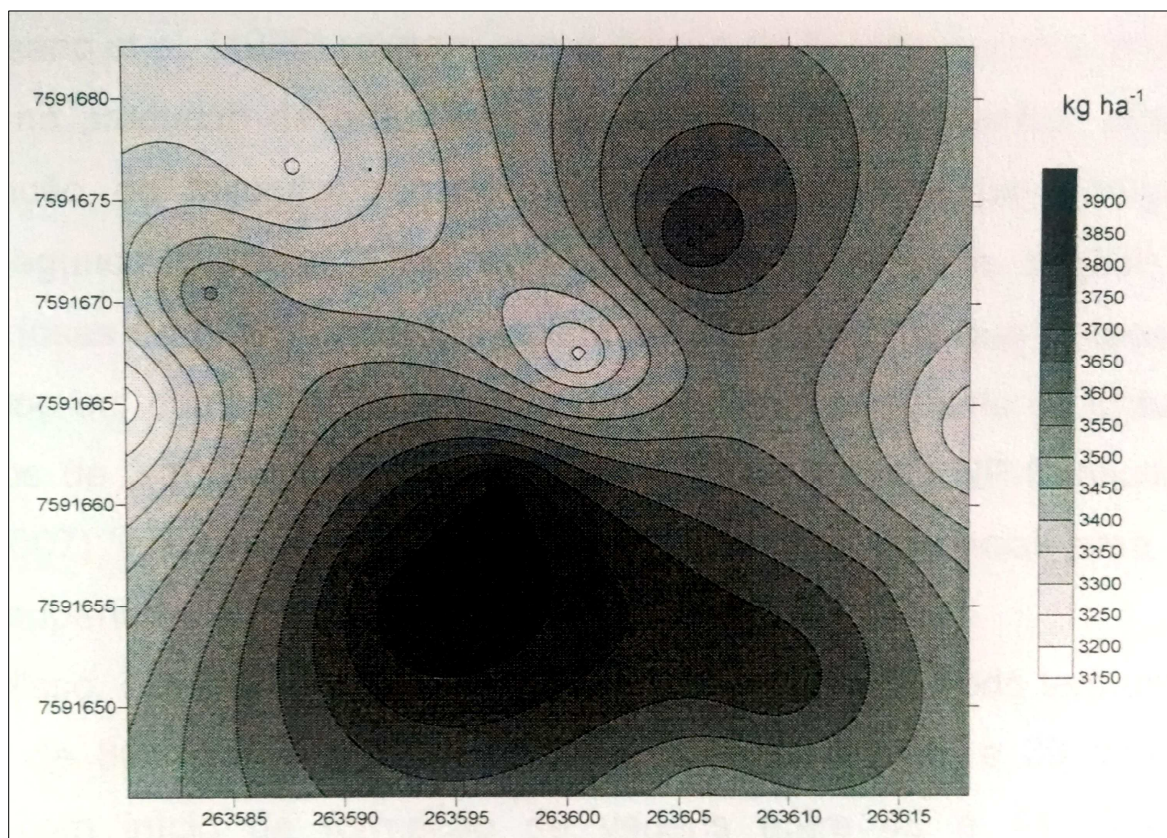


Figura 49. Mapa de Produtividade gerado pelo Surfer 8.0 método de krigagem simples (JUNIOR, 2009)

A área tecnicada mostra que a variabilidade ficou entre 3150 a 3900 kg ha^{-1} (JUNIOR, 2009). Entretanto, na área não tecnicada, a variabilidade ficou entre 1550 a 2650 kg ha^{-1} .

No WebFazenda, ampliando o zoom no mapa térmico de produtividade em uma das áreas tecnicadas (Figura 50), observa-se que a maior concentração de valores está no intervalo de 3084.80 a 3856 kg ha^{-1} . É importante lembrar que o excesso de valores concentrados em intervalos de valores baixos (cor azul) se deve ao fato de ter usado os valores somente nos centroides das áreas experimentais, limitando a análise a somente um ponto. O mapa térmico sempre parte dos valores baixos até valores altos para dar a ideia de “aquecimento” no mapa. Esses valores máximos encontrados mostram a coerência com os valores encontrados no trabalho de JUNIOR, 2009.

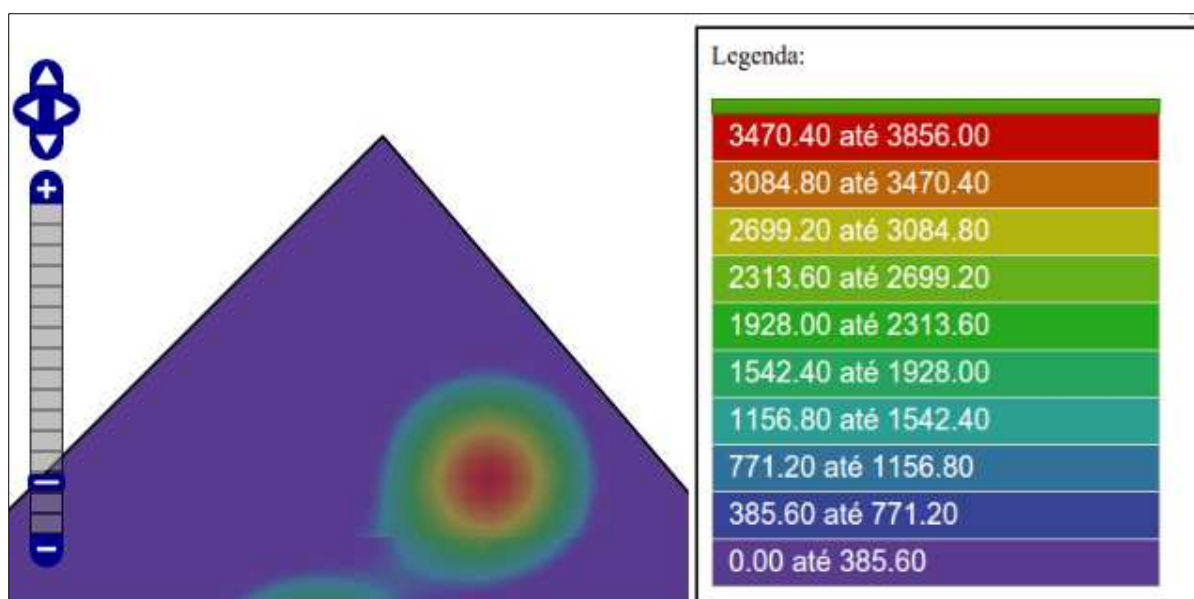


Figura 50. Mapa de Produtividade gerado pelo WebFazenda com zoom no centroide de uma das áreas experimentais da área tecnicificada

Realizando o mesmo procedimento com a área não tecnicificada (Figura 50), observa-se que a concentração maior de valores ficou no intervalo de 2313.60 a 2699.20 kg ha⁻¹. Novamente, esses valores ficam de acordo com os valores encontrados no trabalho de JUNIOR, 2009 na área não tecnicificada (menor que 2699 kg ha⁻¹).

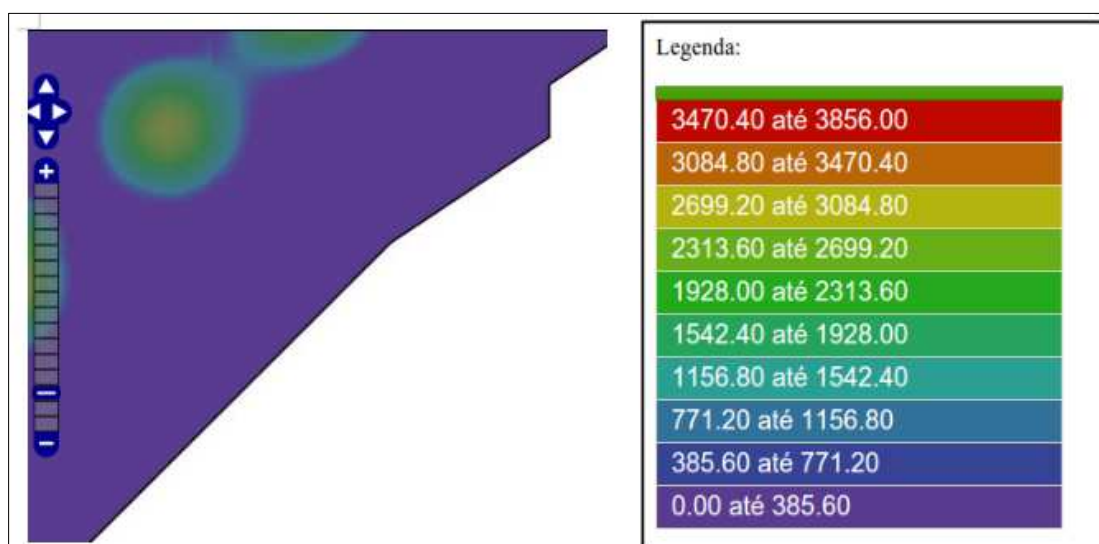


Figura 51. Mapa de Produtividade gerado pelo WebFazenda com zoom no centroide de uma das áreas experimentais da área não tecnicificada

O mesmo procedimento usado para a importação dos dados de produtividade foi realizado com os dados de número de vagens, cadastrando-se o sensor, importando-se os dados e gerando-se os mapas.

Além disso, foram realizadas coletas de amostra do solo para análise em laboratório. Esta coleta foi realizada para cada área experimental, no caso da área tecnificada. Para a área não tecnificada foi feita uma única amostra, pois a mesma não foi trabalhada com agricultura de precisão.

Das amostras, o laboratório enviou os seguintes dados: pH em H₂O, S-SO₄ mg/dm³, P mg/dm³, K mmol c/dm³, Ca mmol c/dm³, Mg mmol c/dm³, Al mmol c/dm³, H+Al mmol c/dm³, Na mmol c/dm³, C g/dm³, MO g/dm³, CTC mmol c/dm³, SB mmol c/dm³, V (%), m (%), ISNa (%), Fe mg/dm³, Cu mg/dm³, Zn mg/dm³, Mn mg/dm³, B mg/dm³.

Esses resultados da análise laboratorial foram então cadastrados no WebFazenda a partir da interface Cadastros – Análise Laboratorial, conforme pode ser visto na figura 52.

Nome	Detalhe	Cor	Ações
H2O		preto	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
SSO4	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
P	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
K	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
Cu	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
Zn	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
Fe	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
Mn	mg/dm3	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
Si	Soma dos íons trocáveis (mmol c / dm3)	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
OTC	capacidade de troca catiônica a pH 7,0 (mmol c / dm3)	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
V	Índice de saturação de bases (%)	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
OPe	Índice de saturação de sódio (%)	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
m	Índice de saturação de alumínio (%)	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>
D	extrator água quente (mg/dm3)	verde	<input type="button" value="Alterar"/> <input type="button" value="Excluir"/>

Nome:	<input type="text" value="PH"/>
Detalhe:	<input type="text" value="H2O"/>
Cor (Representação no Mapa):	<input type="text" value="preto"/>
<input type="button" value="Gravar"/> <input type="button" value="Voltar"/>	

Figura 52. Cadastros dos dados de Análise Laboratorial

Feito o cadastro, foram importadas as áreas em que os dados foram coletados. Em um procedimento do dia a dia, o usuário ao coletar os dados deve registrar com o GPS a localização, depois importar o arquivo NMEA para ter os pontos de coleta para posterior informação dos dados. Porém, nesse projeto, os pontos que se têm são dos limites das áreas experimentais, conforme visto anteriormente. Assim sendo, foi usado o mesmo arquivo gerado anteriormente com os centroides das áreas experimentais. Esse processo de importação ocorre na tela de Análise Laboratorial, Importar Dados (figura 53).

Nome	Detalhe	Cor	1º Passo	2º Passo	
PH	H2O	preto	Ver Mapa com Códigos	Importar Dados	Digitar Dados
SB04	mg/dm ³	oliva	Ver Mapa com Códigos	Importar Dados	Digitar Dados
P	mg/dm ³	lima	Ver Mapa com Códigos	Importar Dados	Digitar Dados
K	mmol c / dm ³	purpura	Ver Mapa com Códigos	Importar Dados	Digitar Dados
Cu	mg/dm ³	ouro	Ver Mapa com Códigos	Importar Dados	Digitar Dados
Zn	mg/dm ³	laranja	Ver Mapa com Códigos	Importar Dados	Digitar Dados
Fe	mg/dm ³	sabão	Ver Mapa com Códigos	Importar Dados	Digitar Dados
Mn	mg/dm ³	bege	Ver Mapa com Códigos	Importar Dados	Digitar Dados
SB	Soma das bases trocáveis (mmol c / dm ³)	larrom	Ver Mapa com Códigos	Importar Dados	Digitar Dados
CTC	capacidade de troca catiônica a pH 7,0 (mmol c / dm ³)	vermelho	Ver Mapa com Códigos	Importar Dados	Digitar Dados
V	Índice de saturação de Bases (%)	rosa	Ver Mapa com Códigos	Importar Dados	Digitar Dados

Nome:

Região:

Data:

Arquivo: Nenhum arquivo selecionado

Figura 53. Importação das localizações de coleta dos dados de Análise Laboratorial

Assim que o resultado chegou do laboratório, seus valores foram digitados manualmente no programa. Para facilitar o entendimento do local de cada dado, o programa disponibilizou um mapa mostrando os códigos de cada posição de coleta do mapa.

A figura 54 mostra os resultados da área tecnicada com os valores por área experimental enquanto que a figura 55 mostra o dado da área não tecnicada. Os resultados das áreas tecnicadas foram digitados um por um para cada área experimental enquanto que na área não tecnicada os valores foram repetidos por todas as áreas experimentais. Toda essa digitação ocorre em Entrada de Dados, Análise Laboratorial, Digitar Dados.

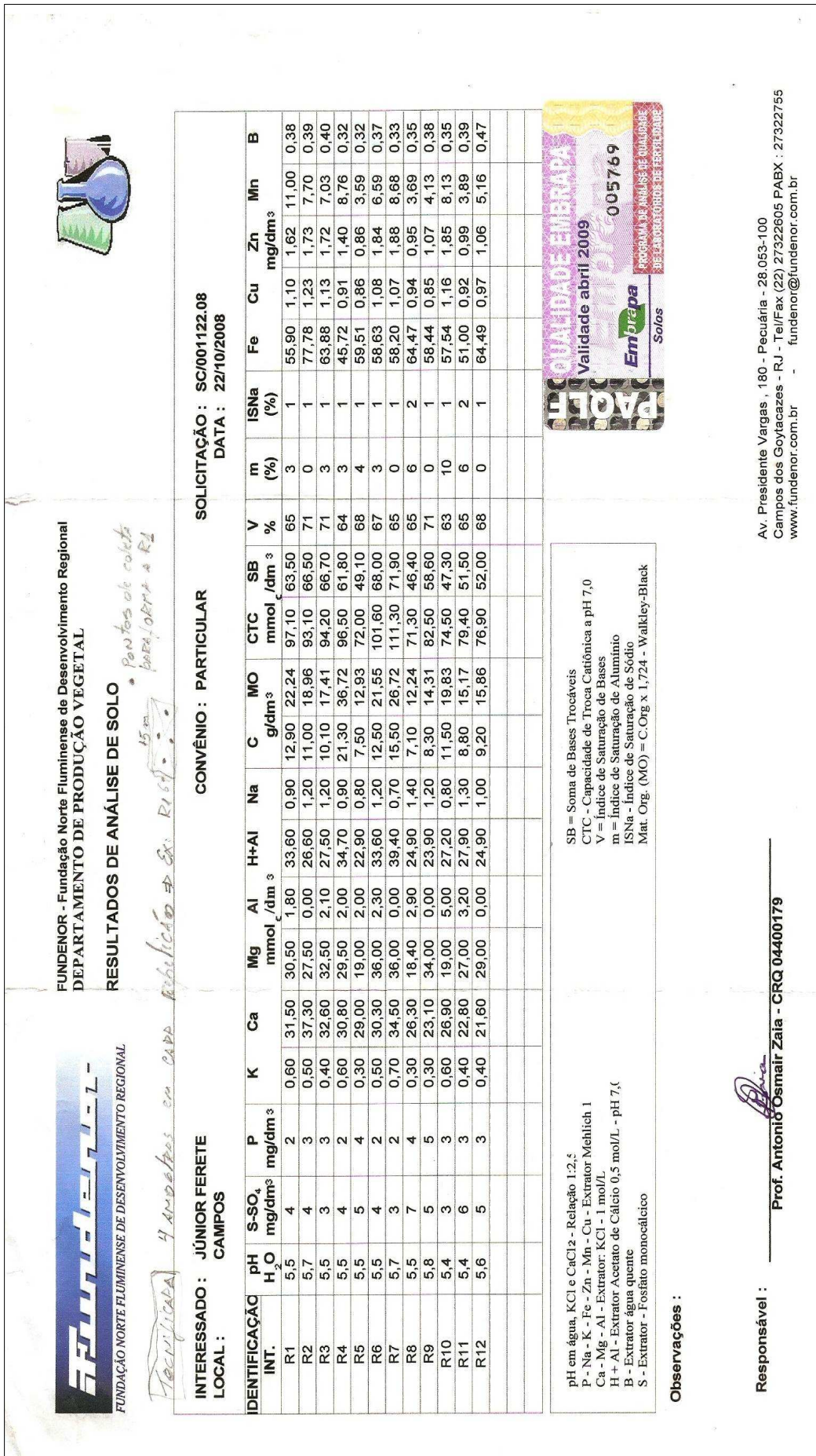


Figura 54. Resultados obtidos das amostras do solo das áreas tecnicadas

Da mesma forma que nos dados obtidos a partir de sensores, os obtidos em análise laboratorial podem ser analisados por mapas térmicos (Heatmap) que são acessados em Visualização de Mapas, Mapas de Dados de Laboratório.

Analisando os mapas térmicos desses dados analisados em laboratório, é possível ver que alguns elementos não influenciaram na produtividade e outros influenciaram diretamente. Essa análise é possível visualmente sem muitos trabalhos para o usuário final.

No caso do índice de saturação do sódio (ISNa), ele tem uma sinergia com a produtividade, apresentando um mapa térmico muito próximo ao próprio mapa de produtividade. Logo, é um candidato a preocupação para melhorar a produtividade em todo o campo. Esse mapa pode ser visto na figura 56.

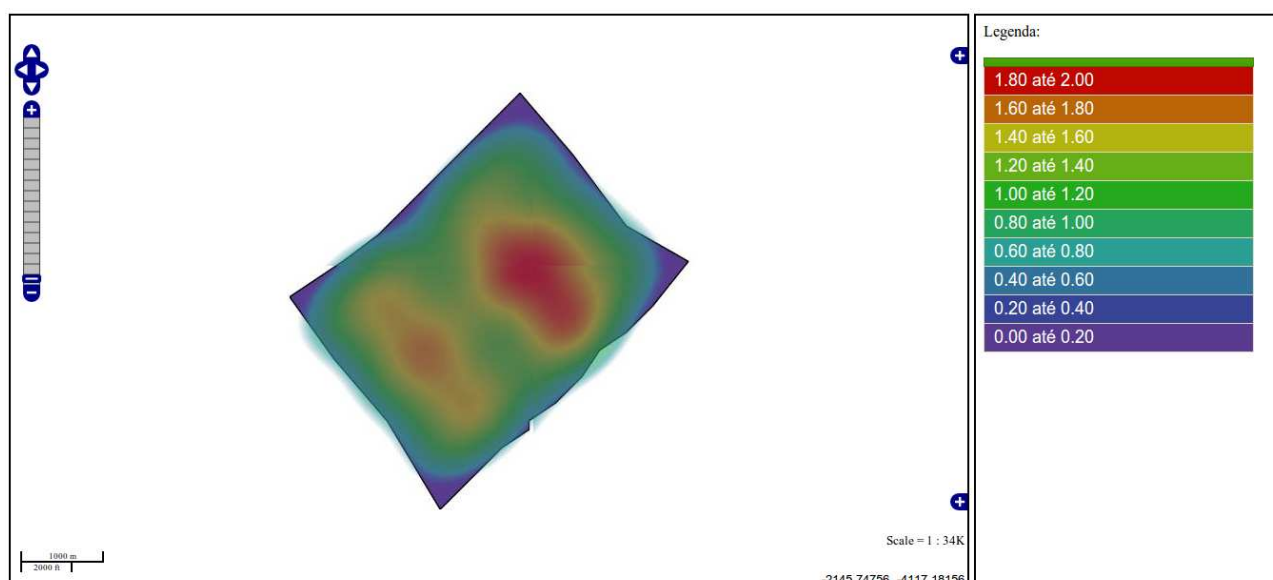


Figura 56. Mapa térmico do ISNa

Por outro lado, o Mn tem uma resposta contrária mostrando valores maiores onde a produtividade foi menor. Isso mostra a sua provável influência na produtividade também. Essa visualização pode ser vista na figura 57.

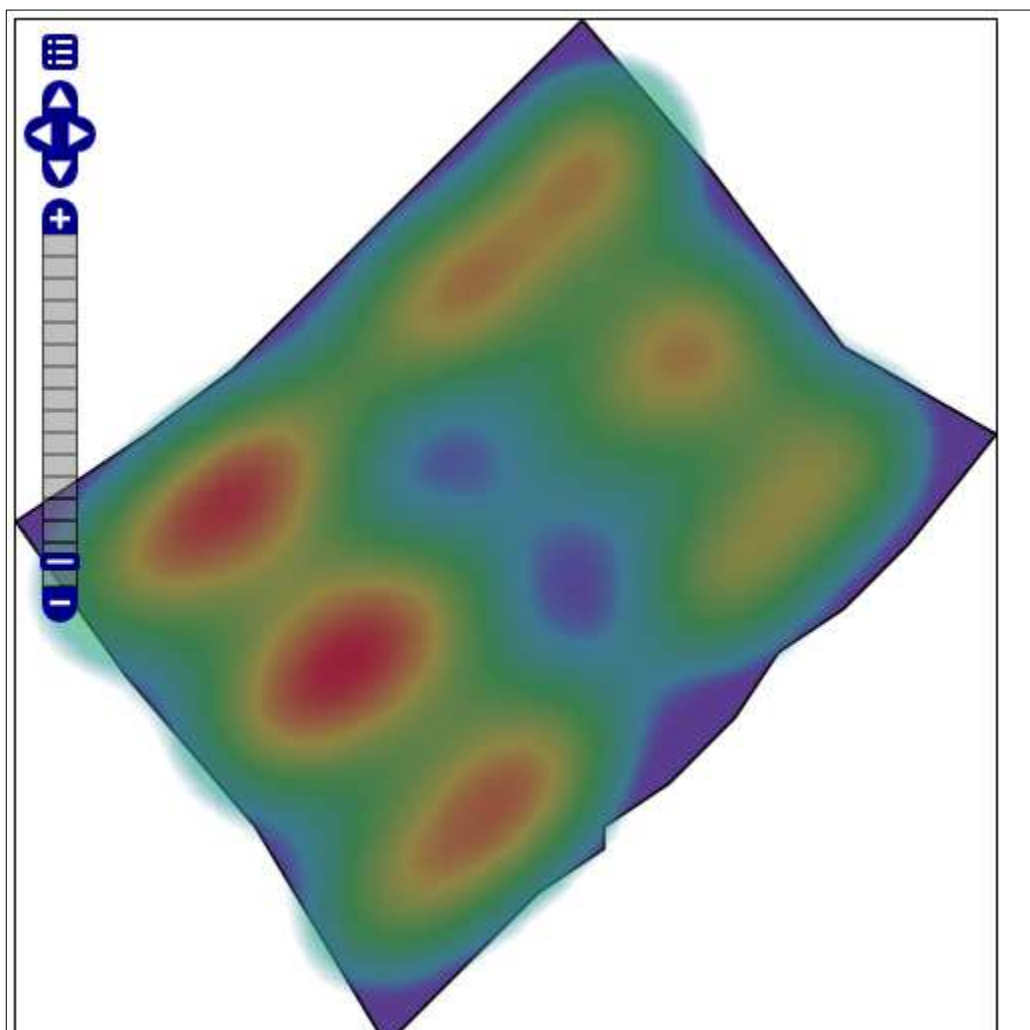


Figura 57. Mapa térmico do Mn

A proposta do WebFazenda é facilitar a entrada de dados, se baseando em importações diretas de arquivos textos gerados ou por aparelhos GPS ou sensores através do sistema de aquisição de dados proposto e, ao final, gerar mapas que permitam ao usuário uma análise melhor do ocorrido no campo e, assim, tomar decisões para a melhoria da sua produtividade.

4.2. Funcionamento do Sistema de Aquisição de Dados

Para avaliar o uso do sistema de aquisição de dados proposto neste trabalho, foi montado um kit de leitura de temperatura.

Para validar os dados lidos, foi realizado um procedimento completo: leitura da área, importação da área, leitura do sensor, cadastro e importação do sensor e visualização dos mapas.

A delimitação da área foi realizada a partir do GPS do celular Motorola Moto X com Android com um programa gratuito chamado GPS NMEA que monitora toda a área gerando o arquivo no formato NMEA. O uso do celular nesta fase do projeto é para mostrar que o uso de ferramentas comuns facilita e baixa o custo de uso da ferramenta como um todo. A primeira parte deste capítulo mostrou a geração da região através de um dispositivo GPS de maior precisão. Assim sendo, dependendo da precisão desejada, não há necessidade da aquisição de ferramentas de alta precisão para o uso do sistema proposto. A importação do arquivo gerado pelo celular no webFazenda, gerou o mapa da figura 58.

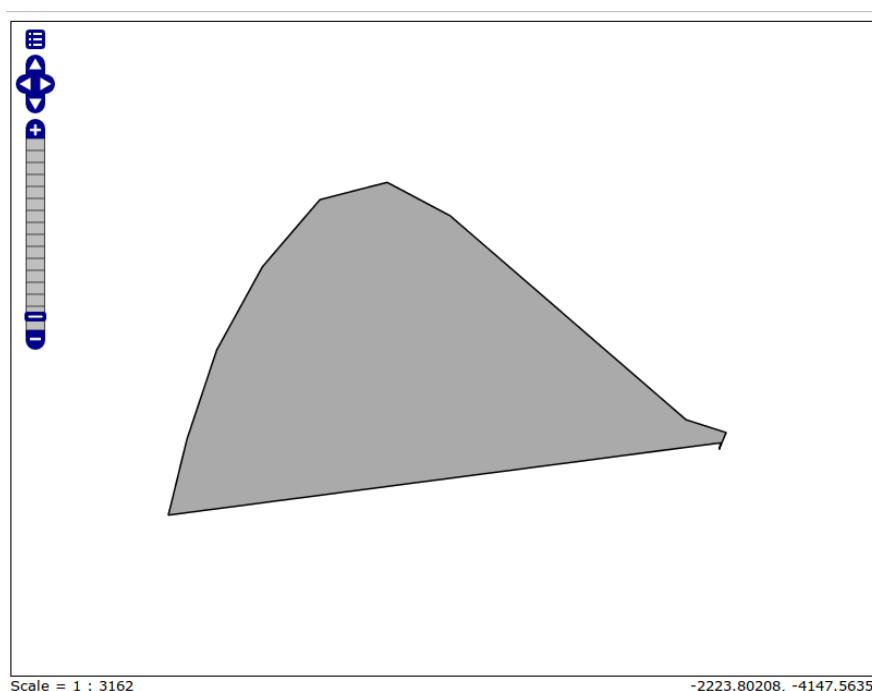


Figura 58. Mapa de região gerado através da importação do arquivo NMEA criado pelo programa NMEA GPS de um celular com Android

Na mesma área, foi realizada a leitura de dados de temperatura, percorrendo alguns pontos da área com o sistema de aquisição de dados de temperatura da figura 59. Para a alimentação do sistema foi usada uma bateria de 9V Ni-Mh de 250 mAh. Caso esse dispositivo fosse instalado em algum trator, poderia usar a própria alimentação que em alguns casos é de 12V em outros de 24V. A alimentação do dispositivo é de no mínimo 5V. O arduíno suporta até 12 V de entrada. Caso seja superior a esse valor seria necessário fazer um circuito para redução de tensão.

Após a obtenção das leituras, foi feito o cadastro do sensor de temperatura no sistema e, em seguida, a importação no WebFazenda dos dados lidos. O DATALOG.TXT e o mapa térmico gerado a partir das leituras podem ser visualizados na figura 60, mostrando a variabilidade da temperatura. A área do mapa que não obteve cor (em cinza) é onde não ocorreram leituras.

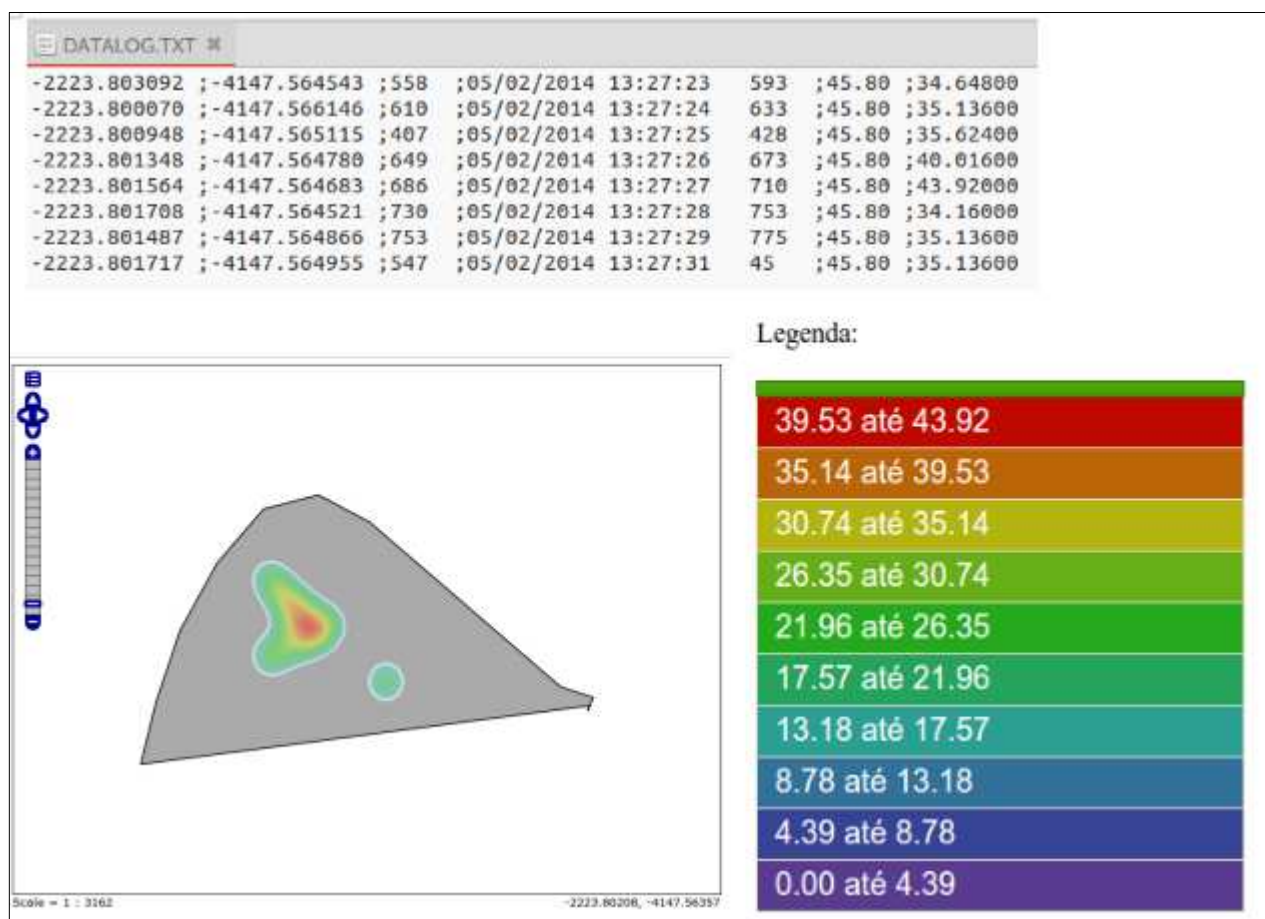


Figura 59. Mapa térmico da região gerado através da importação do arquivo DATALOG.TXT gerado pelo sistema de aquisição de dados de temperatura

A leitura foi realizada em céu aberto, em um dia de temperatura média de 25° C. Entretanto, com a presença do Sol os pontos em que a temperatura lida alcançou 43.92°C correspondem às áreas diretamente expostas à radiação solar. A pequena área que alcançou a temperatura média de 21.96°C corresponde a uma sombra, fazendo com que o sensor registrasse uma temperatura mais baixa.

Assim sendo, dentro do proposto pelo projeto, o sistema de aquisição de dados funcionou plenamente e a sua integração com o programa WebFazenda

gerando mapas para facilitar o entendimento da região também foi bastante eficiente.

5. RESUMO E CONCLUSÕES

No trabalho realizado implementou-se um sistema de aquisição de dados para sensores de qualquer fabricante e porte, analógicos ou digitais, com um GPS acoplado e um programa, denominado WebFazenda, capaz de ler os dados do campo e gerar mapas de ponto ou temáticos para a interpretação do administrador da fazenda.

Para a validação do programa, foi usado um projeto de análise de dados sobre uma produção mecanizada e não mecanizada para realizar comparativo de eficiência entre os métodos. Para tal, os dados foram adaptados para o formato esperado pelo WebFazenda, e todo o processo de uso do programa foi: cadastro de sensores ou dados laboratoriais e importação dos dados georeferenciados. Após foi feita uma comparação entre os mapas gerados pelo WebFazenda (mapa temático) e os mapas gerados pelo trabalho base (mapa de produtividade pelo método de krigagem) e observou-se que os resultados foram os mesmos.

Para a análise da integração do programa com o sistema de aquisição de dados proposto neste trabalho, foi implementado um sistema de aquisição de dados de temperatura. Para tal, foi obtida uma área de análise usando um smartphone com Android e, depois, algumas temperaturas foram coletadas com o sistema de aquisição de dados. Todos os dados foram importados no programa e o mapa térmico foi gerado, mostrando que as áreas fora de sombras obtiveram

as temperaturas mais altas e, as áreas sob sombras obtiveram temperaturas menores, como era de se supor.

O trabalho mostrou que é possível, a um custo acessível, obter um sistema automatizado em sua produção. É bem claro que a eficiência dos dados lidos depende diretamente da qualidade dos sensores usados. Mas, a análise comparativa entre sensores não faz parte desse trabalho; o que se pretende é deixar um caminho para o uso dos sensores sem a necessidade de aquisição de qualquer componente adicional para o seu uso.

Outro ponto importante é a integração: independente do GPS, dos sensores e, da análise laboratorial que foi realizada, todos os dados passam por um único programa, o seu uso pelo administrador da fazenda.

Um ponto a ser destacado neste trabalho é o uso de plataformas de programas, tais como: GeoServer, PostgreSQL, PostGIS, Java, Glassfish, JSF, Hibernate e Hibernate Spatial que são todos gratuitos e, o Arduíno e o escudo de GPS que são dispositivos físicos de baixo custo.

Um ponto que pode ser considerado negativo para o projeto é que sua implementação no cliente não é tão simples: deve-se fazer uma análise para determinar os melhores sensores a serem usados. Após essa etapa é necessário fazer a ligação eletrônica dos sensores ao sistema de aquisição de dados, depois é necessário estender o programa do sistema de aquisição de dados para a interpretação dos dados dos sensores. Além disso, ainda é necessária a implantação do programa no cliente, operação que passa pela instalação dos servidores propostos nesse projeto.

Como sugestão para trabalhos futuros no contexto da pesquisa ora em questão pode-se desenvolver outras formas de saída de dados pelo programa, tais como: gráficos, tabelas e outros tipos de mapas. No sistema de aquisição de dados, seria também de suma importância um estudo dos principais tipos de sensores usados no campo, a fim de criar leitores já prontos para esses sensores, para que já estivessem disponíveis para os produtores de forma mais fácil, reduzindo assim o impacto na implantação do sistema junto ao produtor.

Espera-se que esta pesquisa tenha trazido contribuição científica para o campo do conhecimento na qual se insere, ou seja, para a automatização da produção vegetal.

REFERÊNCIAS BIBLIOGRÁFICAS

- ARONOFF, S. (1995) **Geographic Information Systems: A Management Perspective**. Ottawa, Canadá: WDL publications, 204p.
- BAUER, C. & KING, G. (2005) **Java Persistence with Hibernate: Revised Edition of Hibernate in Action**. Manning Publication Co, 841p.
- BCL – LandView Systems Inc. (2013). Disponível em: <<http://www.landview.com>>. Acesso em: Março de 2013.
- BUEHLER, K. & MCKEE, L. **The OpenGIS Reference**. Massachusetts, USA. Disponível em: <<http://www.opengis.org/info/orm/03-040.doc>> Acesso em: Dezembro de 2010.
- CAPELLI, A. (2007) **Automação Industrial: Controle do Movimento e Processos Contínuos**. 2ª Edição, São Paulo: Ética.
- DEERE, JOHN. **JOHN DEERE** (2013). Disponível em: <<http://www.deere.com.br>> Acesso em: 9 mar. de 2013.
- DIAS, T.L.; OLIVEIRA, M.P.G.; CÂMARA, G.; CARVALHO, M.de SÁ. (2002) **Problemas de Escala e a Relação Área-Indivíduo em Análise Espacial de Dados Censitários**. Informática Pública, Ano 4, Número 1. Disponível em: <<http://www.ip.pbh.gov.br/sum0401.html>>. Acesso em: 14 de nov. de 2010.
- ESRI (2013). **ESRI**. Disponível em: <<http://www.esri.com>> Acesso em 9 de mar. de 2013.
- FARMWORKS (2013). **Farmworks Software: A Division of Trimble**. Disponível em: <<http://www.farmworks.com>>. Acesso em: 9 de mar. de 2013.
- FOWLER, M., SCOTT, K. (2000) **UML Essencial: Um breve guia para a linguagem-padrão de modelagem de objetos**. 2ª Edição, Porto Alegre. Bookman.

FREDDO, A.R.; ASSMANN, T.S. (2004) **Sistema de informação através de Web Service para avaliação da fertilidade do solo e recomendação de adubação e calagem**. 1º Congresso Luso-Brasileiro de Tecnologias de Informação e Comunicação na Agro-Pecuária. 12 de jun. de 2004.

GeoServer. (2013). **GeoServer**. Disponível em: <<http://geoserver.org/display/GEOS/Welcome>>. Acesso em: 14 de nov. de 2013.

GUEDES, G.T.A. (2011) **UML 2 : uma abordagem prática**. 2ª ed. São Paulo: Novatec.

HIBERNATE SPATIAL (2013). **Hibernate Spatial**. Disponível em: <<http://www.hibernate.org>>. Acesso em: 14 de nov. de 2013.

IBGE (2011). **Geodésia**. Disponível em: <<Http://www.ibge.gov.br/home/geociencias/geodesia/>>. Acesso em: 10 de jun. de 2011.

JUNIOR, J.F.S.V. (2009) **Produtividade do Feijoeiro em Cultivo Tradicional e Tecnificado no Norte Fluminense**. Dissertação (Mestrado em Produção Vegetal) - Campos dos Goytacazes – RJ, Universidade Estadual do Norte Fluminense, UENF.

KRAAK, M.J., ORMELING, F. (2010) **Cartography: Visualization of Spatial Data**. 3ª Edição. Inglaterra, Prentice Hall.

LINNET GEOMATICS (2012). **Croplands – The System**. Disponível em: <<http://www.linnet.ca>> Acesso em 14 de nov. de 2012.

MANZATTO, C.V; BHERING, S.B.; SIMÕES, M. (1999) **Agricultura de precisão: propostas e ações da EMBRAPA solos**. EMBRAPA Solos. Disponível em: <<http://www.cnps.embrapa.br/search/pesqs/proj01/proj01.html>> Acesso em: 14 de jan. de 2011.

MAPSERVER. (2013). **An Introduction to MapServer**. Disponível em: <<http://mapserver.org/introduction.html>>. Acesso em: 14 de nov. de 2013.

MCROBERTS, M. (2011) **Arduíno Básico**. Novatec Editora. São Paulo: Novatec.

MERCALDI, H.V. (2012) **Automação de um sistema de pulverização para aplicação de agroquímicos a taxa variada usando injeção direta**. Dissertação (Mestrado em Ciências) - São Carlos – SP, Universidade de São Paulo, USP.

MIRANDA, E.E. (2011) **Informática Brasileira em Análise. Quem precisa da Agricultura de Precisão?** Informática em Análise. EMBRAPA Monitoramento por Satélite. Disponível em: <http://www.cesar.org.br/analise/n_20/artigon_20.html> acesso em: 09 de mar. de 2011.

MOLIN, J.P. (2011). **Agricultura de Precisão: Situação Atual e Perspectivas**. EMBRAPA. Disponível em: <http://www.agencia.cnptia.embrapa.br/repositorio/agricultura-precisaosituacao_000fkl0ctoe02wyiv80sq98yqpxloebw.pdf> Acesso em: 31 de ago. de 2011.

NAVATHE, S.B; ELMASRI, R. (2011) **Fundamentals of Database Systems**. USA: Addison-Wesley, Pearson Education.

NMEA, (2001). **THE NMEA 0183 Protocol**. Disponível em: <<http://fort21.ru/download/NMEAdescription.pdf>>. Acesso em: 11 de nov. de 2013.

NUNES M. & O'NEILL H. (2004) **Fundamental de UML**. 5ª ed., São Paulo: FCA, 225p.

OGC, (2010). **OGC Web Services Common Standard**. OpenGis Project Document 06-121r9. Publicado em 07 de abr. de 2010.

OGC, (1999). **OpenGIS Simple Specifications for SQL Revision 1.1**. OpenGis Project Document 99-049. Publicado em 05 de mai. de 1999.

OLIVEIRA, R.A.N.; REIS, A.A.; VASCONCELOS, R.O; SILVA, F.S. (2003) **OL4JSF: Uma biblioteca de componentes para elaboração de aplicações geoespaciais**. XXI Congresso da Sociedade Brasileira de Cartografia. Setembro, 2003.

OpenLayers (2008). **OpenLayers**. Disponível em: <<http://docs.openlayers.org/>>. Acesso em: 12 de out. de 2013.

PARK, J. & MACKAY, S. (2003). **Practical Data Acquisition for Instrumentation and Control Systems**, Elsevier, Oxford, 409p.

PIMENTA, F.M., LANDAU, E.C., HIRSCH, A., GUIMARÃES, D.P. (2012) **SERVIDORES DE MAPAS - Programação para Disponibilizar Dados Geográficos Multidisciplinares Utilizando Tecnologias Livres**. Brasília: Embrapa.

PITANGA, T. **JAVA SERVER FACES** (2009). Disponível em: <<http://www.guj.com.br/content/articles/jsf/jsf.pdf>>. Acesso em: 14 de jan. de 2011.

POSTGIS MANUAL (2011). **PostGIS 1.5.3 Manual**. Disponível em: <<http://postgis.refractor.net/documentation/manual-1.5/index.html>>. Acesso em: 01 de mai. de 2011.

POSTGRESQL (2011). **PostgreSQL 9.0.4 Documentation**. Disponível em: <<http://www.postgresql.org/docs/9.0/static/intro-what-is.html>>. Acesso em: 01 de mai. de 2011.

ROCHA, C.H.B. (2000) **Geoprocessamento: tecnologia transdisciplinas**. Juiz de Fora, MG: Ed. do Autor.

ROSÁRIO, J.M. (2008) **Princípios de Mecatrônica**. São Paulo, SP. Editora Pearson Prentice Hall.

SEARCY, S. (2001) **Special topics in precision agriculture: Lectures Notes**. Disponível em: <<http://www.agen.tamu.edu/txprecag/agism489/>>. Acesso em: 10 de abr. de 2010.

SHANKAR, N.R. (2008) **UMA ALTERNATIVA PARA A JAVA SERVER FACES**. Disponível em: <<http://www.javaworld.com/javaworld/jw-01-2008/jw-01-swf4jsf.html>> Acesso em: 14 de jan. de 2011.

SIGA (2013). **SIGA: 30 ans à votre service Depuis 1981**. Disponível em <<http://www.siga.net/accueil>>. Acesso em 9 de mar. de 2013.

SILVA, A.M.R. & VIDEIRA C.A.E. (2011) **UML, Metodologias e Ferramentas CASE: linguagem de modelação UML, metodologias e ferramentas CASE na concepção e desenvolvimento de software**. Portugal: Edições Centro Atlântico.

SOLOMAN, S. (2010) **Sensors and Control Systems in Manufacturing**. 2ª Edição. Nova Iorque, EUA.

SOLOMAN, S. (2009) **Sensors Handbook**. 2ª Edição. Nova Iorque, EUA. McGrawHill.

SOUZA, R.O.R.M. (2001) **Desenvolvimento e avaliação de um sistema de irrigação automatizado para áreas experimentais**. Dissertação (Mestrado em Agronomia) – Piracicaba – SP, Universidade de São Paulo, USP.

SST Development Group, Inc. (2013) **SST Software: Manage Data. Harvest Information**. Disponível em: <<http://http://www.sstdevgroup.com/>>. Acesso em: 9 de mar. de 2013.

STABILE, M.C.C.; BALASTREIRE, L.A. **Comparação de três receptores GPS para uso em agricultura de precisão**. Revista Engenharia Agrícola Jaboticabal, 26:215-223, 2006

TIMMIS, H. (2011), **Practical Arduino Engineering**. Technology In Action. Apress, EUA.

UNIMEP (Universidade Metodista de Piracicaba), (2009). **Engenharia de Software. Tópico 6- Diagrama de Classes**. Piracicaba, SP, Brasil, 2009. Disponível em: <http://www.unimep.br/~aeasilva/topico6_es.pdf >. Acesso em: 17 de mar. 2013.

APÊNDICE A - Mapeamento das Classes Usando o Hibernate e o Hibernate Espacial

1) Classe Região:

```
package model;

import com.vividsolutions.jts.geom.Polygon;
import java.io.Serializable;
import javax.persistence.*;
import org.hibernate.annotations.Type;

@Entity

public class Regiao implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long codigo;

    private String nomeRegiao;

    @Column(columnDefinition = "Geometry")
    @Type(type = "org.hibernate.spatial.GeometryType")
    private Polygon localizacaoRegiao;

    @Override
    public String toString() {
        return this.nomeRegiao;
    }
}
```



```
public Long getCodigo() {  
    return codigo;  
}  
  
public void setCodigo(Long codigo) {  
    this.codigo = codigo;  
}  
  
public Polygon getLocalizacaoRegiao() {  
    return localizacaoRegiao;  
}  
  
public void setLocalizacaoRegiao(Polygon localizacaoRegiao) {  
    this.localizacaoRegiao = localizacaoRegiao;  
}  
  
public String getNomeRegiao() {  
    return nomeRegiao;  
}  
  
public void setNomeRegiao(String nomeRegiao) {  
    this.nomeRegiao = nomeRegiao;  
}  
  
}
```

2) Classe Origem:

```
package model;
```

```
import java.io.Serializable;
```

```
import javax.persistence.*;
```

```
@Entity
```

```
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
```

```
public class Origem implements Serializable {
```

```
    //incluir atributo para controlar data
```

```
    private static final long serialVersionUID = 1L;
```

```
    @Id
```

```
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Long codigo;
```

```
    private String nomeOrigem;
```

```
    private String nomeCamada;
```

```
    private String nomeCorCamada;
```

```
    private String detalhe;
```

```
    @Override
```

```
    public String toString() {
```

```
        return this.nomeOrigem;
```

```
    }
```

```
public Long getCodigo() {  
    return codigo;  
}
```

```
public void setCodigo(Long codigo) {  
    this.codigo = codigo;  
}
```

```
public String getNomeOrigem() {  
    return nomeOrigem;  
}
```

```
public void setNomeOrigem(String nomeOrigem) {  
    this.nomeOrigem = nomeOrigem;  
}
```

```
public String getNomeCamada() {  
    return nomeCamada;  
}
```

```
public void setNomeCamada(String nomeCamada) {  
    this.nomeCamada = nomeCamada;  
}
```

```
public String getNomeCorCamada() {  
    return nomeCorCamada;  
}
```

```
}
```

```
public void setNomeCorCamada(String nomeCorCamada) {
```

```
    this.nomeCorCamada = nomeCorCamada;
```

```
}
```

```
public String getDetalhe() {
```

```
    return detalhe;
```

```
}
```

```
public void setDetalhe(String detalhe) {
```

```
    this.detalhe = detalhe;
```

```
}
```

```
}
```

3) Classe Sensor:

```
package model;
```

```
import java.io.Serializable;
```

```
import javax.persistence.Entity;
```

```
@Entity
```

```
public class Sensor extends Origem implements Serializable {
```

```
    private Boolean analogico;
```

```
    public Boolean getAnalogico() {
```

```
        return analogico;
```

```
    }
```

```
    public void setAnalogico(Boolean analogico) {
```

```
        this.analogico = analogico;
```

```
    }
```

```
}
```

4) Classe Amostragem Laboratorial:

```
package model;
```

```
import java.io.Serializable;
```

```
import javax.persistence.Entity;
```

```
@Entity
```

```
public class AmostragemLaboratorio extends Origem implements Serializable {
```

```
}
```

5) Classe Leitura:

```
package model;

import com.vividsolutions.jts.geom.Point;

import java.io.Serializable;

import java.math.BigDecimal;

import java.util.Calendar;

import javax.persistence.*;

import org.hibernate.annotations.Type;

@Entity

public class Leitura implements Serializable {

    private static final long serialVersionUID = 1L;

    @Id

    @GeneratedValue(strategy = GenerationType.IDENTITY)

    private Long codigo;

    private BigDecimal valorLeitura;

    @Temporal(javax.persistence.TemporalType.TIMESTAMP)

    private Calendar horaLeitura;

    @Temporal(javax.persistence.TemporalType.DATE)

    private Calendar dataLeitura;

    @Column(columnDefinition = "Geometry")

    @Type(type = "org.hibernate.spatial.GeometryType")

    private Point localizacaoLeitura;
```

```
@ManyToOne
```

```
private Origem origem;
```

```
@ManyToOne
```

```
private Regiao regiao;
```

```
@Override
```

```
public String toString() {
```

```
    return this.origem.getNomeOrigem() + " - " + this.valorLeitura.toString();
```

```
}
```

```
public Long getCodigo() {
```

```
    return codigo;
```

```
}
```

```
public void setCodigo(Long codigo) {
```

```
    this.codigo = codigo;
```

```
}
```

```
public Calendar getHoraLeitura() {
```

```
    return horaLeitura;
```

```
}
```

```
public void setHoraLeitura(Calendar horaLeitura) {
```

```
    this.horaLeitura = horaLeitura;
```

```
}
```



```
public Point getLocalizacaoLeitura() {  
    return localizacaoLeitura;  
}
```

```
public void setLocalizacaoLeitura(Point localizacaoLeitura) {  
    this.localizacaoLeitura = localizacaoLeitura;  
}
```

```
public Origem getOrigem() {  
    return origem;  
}
```

```
public void setOrigem(Origem origem) {  
    this.origem = origem;  
}
```

```
public Regiao getRegiao() {  
    return regiao;  
}
```

```
public void setRegiao(Regiao regiao) {  
    this.regiao = regiao;  
}
```

```
public BigDecimal getValorLeitura() {  
    return valorLeitura;  
}
```

```
public void setValorLeitura(BigDecimal valorLeitura) {  
    this.valorLeitura = valorLeitura;  
}
```

```
public Calendar getDataLeitura() {  
    return dataLeitura;  
}
```

```
public void setDataLeitura(Calendar dataLeitura) {  
    this.dataLeitura = dataLeitura;  
}
```

```
}
```

6) Classe Usuário:

```
package model;

import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity

public class Usuario implements Serializable {
    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long codigo;
    private String login;
    private String senha;

    private static Usuario usuarioAtual;

    public Long getCodigo() {
        return codigo;
    }

    public void setCodigo(Long codigo) {
```

```
        this.codigo = codigo;
    }

    public String getLogin() {
        return login;
    }

    public void setLogin(String login) {
        this.login = login;
    }

    public String getSenha() {
        return senha;
    }

    public void setSenha(String senha) {
        this.senha = senha;
    }

    public static Usuario getUsuarioAtual() {
        return usuarioAtual;
    }

    public static void setUsuarioAtual(Usuario usuarioAtual) {
        Usuario.usuarioAtual = usuarioAtual;
    }
}
```

}

}