



UNIVERSIDADE FEDERAL DO CEARÁ
MESTRADO E DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

LARISSA CANDEIA SOARES REIS

UMA ARQUITETURA DE SISTEMA DE
VISUALIZAÇÃO WEB DE DADOS
ESPAÇO-TEMPORAIS COM FOCO NA
GRANULARIDADE ESPACIAL

FORTALEZA, CEARÁ
2016

LARISSA CANDEIA SOARES REIS

Uma Arquitetura de Sistema de Visualização Web de Dados Espaço-Temporais com Foco na Granularidade Espacial

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientadora:

Profa. Dra. Emanuele Marques dos Santos

Fortaleza, Ceará

2016

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

R311a Reis, Larissa Candeia Soares.

Uma arquitetura de sistema de visualização web de dados espaço-temporais com foco na granularidade espacial / Larissa Candeia Soares Reis. – 2016.
93 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2016.

Orientação: Profa. Dra. Emanuele Marques dos Santos.

1. Arquitetura de sistema. I. Título.

CDD 005

Larissa Candeia Soares Reis

Uma Arquitetura de Sistema de Visualização Web de Dados Espaço-Temporais com Foco na Granularidade Espacial

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Aprovada em: 29 de novembro de 2016

BANCA EXAMINADORA

Profa. Dra. Emanuele Marques dos Santos(orientadora)
Universidade Federal do Ceará- UFC

Prof. Dr. Creto Augusto Vidal
Universidade Federal do Ceará- UFC

Prof. Dr. Joaquim Bento Cavalcante Neto
Universidade Federal do Ceará- UFC

Prof. Dr. Francisco Nivando Bezerra
Instituto Federal do Ceará- IFCE

AGRADECIMENTOS

Creio veementemente que cada um de nós é fruto das pessoas que passaram por nossa vida e deixaram um pouco de si conosco. Portanto, não poderia deixar de expressar os meus profundos agradecimentos a estas pessoas sem as quais este trabalho não teria sido possível.

Aos meus pais, Socorro e Edidio, e avós, Bianor e Otalvívia, a mais terna gratidão. Sou e sempre serei grata pelos ensinamentos e amor que recebi e que recebo até hoje.

Ao meu namorado, Victor, por ter me apoiado incessantemente durante essa jornada.

Agradeço a todo grupo Crab, em particular aos professores Bento, Creto e Emanuele, que me deram a oportunidade de ingressar na pós-graduação, me acolheram no grupo e compartilharam seus conhecimentos comigo.

Desejo agradecer em especial à minha orientadora, Emanuele, pela paciência e por sempre estar disposta a tirar dúvidas, enriquecer e discutir a pesquisa aqui apresentada.

Por fim, desejo agradecer à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), que financiou esta pesquisa.

I do not know what I may appear to the world, but to myself I seem to have been only like a boy playing on the seashore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.
Isaac Newton

RESUMO

O rápido crescimento da Web e da quantidade de informações disponíveis online influenciou a forma com que pessoas lidam com dados. Hoje em dia, publicar dados é uma tendência mundial. Embora seja importante tornar tais informações disponíveis para as pessoas, também é importante que pessoas tenham uma forma fácil de visualizar e explorar essas informações, para que compreendam melhor o mundo em que vivem.

O presente trabalho objetiva apresentar o CrabVis, uma arquitetura para um sistema de visualização de dados espaço-temporais capaz de explorar a granularidade de dados espaciais. A arquitetura proposta é conceitualmente dividida em camadas e utiliza um modelo cliente-servidor. O principal objetivo do CrabVis é prover um sistema intuitivo em que pessoas sem experiência em visualização de dados possam explorar os seus dados espaço-temporais através da criação e compartilhamento de dashboards compostas por visualizações, textos e imagens.

Para demonstrar a validade da arquitetura, foi implementado um sistema baseado na arquitetura proposta. Esse mesmo sistema foi utilizado na criação de dois estudos de casos. O primeiro estudo de caso relaciona o crescimento de casos de dengue à falta de saneamento básico no Brasil. O segundo estudo mostra o aumento em vendas de agrotóxico no Brasil por estado e o aumento de casos de intoxicação registrados em casas de saúde nos anos 2009 e 2014.

ABSTRACT

The fast-paced growth of the Web and of the information available online has influenced the way people deal with data. To publish data is a worldwide trend nowadays. Though it is important to make such information available for anyone, it is also essential that people with no background in data visualization have an easy way to visualize and explore different types of data, so that they can understand better the world they live in.

This work aims to present an architecture for a spatial-temporal visualization system capable of exploring the granularity of spatial data. The proposed architecture is conceptually divided into layers and uses a client-server model. The main purpose of the proposed system is to provide an intuitive system where people with no data visualization expertise can explore their spatial-temporal data through the creation of dashboards composed by visualizations, texts and images.

In order to demonstrate the validity of the architecture, a system was implemented based on the proposed architecture. This system was used on the creation of two case studies. The proposed architecture was implemented and two case studies were created using the implemented system. The first case study relates the growth of dengue fever cases to the lack of basic sanitation in Brazil. The second case study shows the increase of agrototoxic sales in Brazil by state and the increase of intoxication cases registered on health facilities in the years of 2009 and 2014.

Lista de Figuras

1.1	Portais Dados Abertos Rio (à esquerda) e Dados Abertos Fortaleza (à direita).	7
2.1	Granularidade de dados.	11
2.2	Bibliotecas públicas dos Estados Unidos no ano de 2009.	12
2.3	Taxas de desemprego nos Estados Unidos a partir de agosto de 2016.	13
2.4	Recurso solar fotovoltaico e potencial de energia solar concentrada nos Estados Unidos.	14
3.1	Este mapa mostra a distância de cada condado dos Estados Unidos para a costa do Pacífico ou do Atlântico. O gráfico foi criado com o D3.	16
3.2	Especificação de um gráfico de barras na gramática Vega.	17
3.3	Mapas criados com o Leaflet (na esquerda) e Google Maps API(na direita).	18
3.4	Many Eyes.	19
3.5	Lyra.	20
3.6	Visualização feita no Tableau Desktop: Mapa original (à esquerda) e mapa dividido em territórios definidos pelo usuário (à direita).	21
3.7	Dashboard criada utilizando o Quadrigram (à esquerda) e Dashboard criada utilizando o Plotly (à direita).	22
3.8	Qlik Sense Cloud.	23
3.9	Wikicrimes.	24
3.10	Onde fui roubado.	25
3.11	Social Explorer.	26
3.12	Dengue-GIS.	27
3.13	Cubo multidimensional representando a ocorrência de dengue em três locais diferentes (representados pelos três gráficos em vermelho, verde e azul).	28
3.14	Técnicas presentes em Ferreira <i>et al.</i> (2013).	29
4.1	Arquitetura CrabVis.	31
4.2	Ações na Camada do cliente.	31
4.3	Workflow do uso do sistema.	32
4.4	Esquema da arquitetura CrabVis.	34
4.5	Componente Map Container.	35
4.6	Map Container.	36

4.7	Map Container do Brasil.	36
4.8	Esquema do Conjunto de Dados na Camada de Informação.	37
4.9	Esquema dos Mapas na Camada de Informação.	37
4.10	Exemplo de criação de Map Containers durante a Importação de Mapas.	38
4.11	Esquema das Dashboards na Camada de Informação.	39
4.12	Esquema do arquivo estruturado da dashboard.	40
4.13	Elementos da Dashboard no arquivo estruturado.	41
4.14	Esquema dos Filtros na Camada de Informação.	41
5.1	Homepage do CrabVis.	43
5.2	Tela de registro (acima) e tela de login (abaixo).	45
5.3	Tela de gerenciamento de Mapas (acima) e tela de Importação de Mapas (abaixo).	46
5.4	Processo de importação de Mapas: Etapa 1 e Etapa 2.	47
5.5	Processo de importação de Mapas: Etapa 3 e Etapa 4.	48
5.6	Sugestão baseada em um Map Container existente.	49
5.7	Lista com propriedades dos polígonos presentes no mapa.	50
5.8	Tela de gerenciamento de Dados (acima) e tela de Importação de Dados (abaixo).	51
5.9	Processo de importação de Dados: Etapa 1 e Etapa 2.	53
5.10	Processo de importação de Dados: Etapa 3 e Etapa 4.	54
5.11	Resumo da Importação do novo conjunto de dados.	57
5.12	Seleção de dados (acima) e mapas (abaixo) para a nova dashboard.	59
5.13	Tela inicial da criação de dashboard (acima) e tela da seleção de gráficos (abaixo).	60
5.14	Processo de criação de mapa de pontos. Seleção das colunas com coordenadas (acima) e mapa resultante (abaixo).	62
5.15	<i>Quadtree</i> no mapa de pontos.	63
5.16	Processo de criação de mapa de calor. Seleção das colunas com coordenadas (acima) e mapa resultante (abaixo).	66
5.17	Saneamento básico e a dengue em 2008.	68
5.18	Análise sobre o uso de agrotóxicos no Brasil nos anos de 2009 e 2014.	71
5.19	Análise sobre o uso de agrotóxicos no Brasil nos anos de 2009 e 2014.	72

Sumário

1	Introdução	5
1.1	Contextualização do problema	5
1.2	Objetivos	8
1.3	Estrutura do trabalho	9
2	Conceitos Preliminares	10
2.1	Dados espaço-temporais	10
2.2	Granularidade de dados	10
2.3	Visualização de dados espaciais	11
2.3.1	Mapa de pontos	12
2.3.2	Mapa temático	12
2.3.3	Mapa de calor	13
3	Trabalhos Relacionados	15
3.1	Linguagens e Bibliotecas	15
3.1.1	Protovis, D3 e Processing	15
3.1.2	Vega	16
3.1.3	Leaflet e GoogleMaps	17
3.2	Sistemas de Visualização Web de propósito geral	19
3.2.1	ManyEyes	19
3.2.2	Tableau e Lyra	20
3.2.3	Plotly, Quadrigram e Qlik	21
3.3	Sistemas de Visualização Web de propósito específico	23
3.3.1	Wikicrimes e Onde fui roubado	23
3.3.2	Social Explorer	25
3.3.3	Dengue-GIS	26
3.3.4	Sistema de alerta de epidemia de dengue no Vietnã	27
3.3.5	TaxiVis	28
4	Arquitetura CrabVis	30
4.1	Visão geral da arquitetura proposta	30
4.2	Workflow operacional da arquitetura	31
4.3	Estrutura dos dados	33

4.3.1	Map Container	35
4.3.2	Conjuntos de dados	36
4.3.3	Mapas	37
4.3.4	Dashboard	39
4.4	Filtros	41
5	Implementação e Casos de Uso	43
5.1	Informações gerais	43
5.2	Registro e Autenticação	45
5.3	Importação de Mapas	46
5.4	Importação de Dados	51
5.5	Criação da Dashboard	58
5.5.1	Mapa de pontos	61
5.5.2	Mapa temático	64
5.5.3	Mapa de calor	65
5.6	Casos de Uso	67
5.6.1	Relação entre Saneamento básico e a dengue no Brasil	67
5.6.2	Agrotóxicos no Brasil: Comparativo entre os anos de 2009 e 2014	69
6	Conclusão e Trabalhos Futuros	73
A	Apêndice - Dados Utilizados nos Casos de Uso	75

1 Introdução

1.1 Contextualização do problema

A popularização da Web, consequência da mudança de paradigma proveniente da era da Informação, tem colocado as pessoas em contato com um volume imenso de informações diariamente. São dados que outrora permaneceriam restritos a poucas pessoas, sem divulgação. Cabe aos usuários, consumidores da informação, avaliar a sua utilidade e a sua veracidade, e para isso, é imprescindível saber ler e interpretar o que é lido.

A Visualização de dados figura, nesta era, como uma ferramenta eficiente para a representação do conhecimento (Fekete *et al.*, 2008). Gráficos são formas mais amistosas de adquirir informação. Para reconhecer isto, é suficiente lembrar que uma mesma informação pode ser representada por uma extensa tabela textual ou por um gráfico simples, como um gráfico de pontos.

Então, atualmente, pode-se dizer que, assim como é necessário saber ler e interpretar o que é lido, como dito anteriormente, também é necessário saber ler e interpretar gráficos. Esta tem sido uma preocupação insurgente entre cientistas (Börner *et al.*, 2015; Boy *et al.*, 2014).

Börner *et al.* (2015) realizaram uma pesquisa entre frequentadores de museus de ciências que avaliava o seu conhecimento acerca de visualizações básicas. No estudo, os autores concluem que, na área de pesquisa de Visualização da informação, muito se tem feito para criar novas e complexas visualizações, mas poucas destas visualizações são utilizadas fora de ambientes específicos como periódicos e anais de conferências. Então, se mesmo adultos e jovens com predileção por ciência falham em ler visualizações básicas, mais iniciativas devem ser tomadas para que as pessoas sejam capazes de ler e compreender visualizações.

Desta forma, um dos maiores desafios que permeia esta era é popularizar a Visualização de dados entre um público leigo. Assim como é importante que as pessoas tenham acesso às informações, é importante que elas tenham meios de visualizá-las facilmente, sem a necessidade de possuir conhecimento técnico sobre como criá-las. Popularizar a Visualização de dados é permitir a todos uma maior compreensão do mundo.

Colocar uma pessoa leiga no papel de usuário de um sistema em que ela pode visualizar os seus dados facilmente é torná-la um protagonista no processo de fazer ciência. Ao interagir com os dados utilizando visualizações, essa pessoa está efetivamente construindo

conhecimento, através da compreensão, de forma sistemática, da sua própria realidade.

Um bom exemplo de como os dados permitem as pessoas compreender melhor o mundo em que vivem é através da Visualização de dados urbanos. Dados urbanos englobam todos os dados gerados no contexto de uma cidade. No âmbito urbano, a cada dia, uma imensa quantidade de dados é gerada. Alguns desses dados, tais como dados gerados por organizações governamentais, possuem relevância e utilidade pública. Além disso, a popularização da tecnologia possibilita que cada habitante possua um dispositivo eletrônico tal como celular, *tablet* ou computador pessoal e gere dados referentes à sua localização e trajetória, ao seu consumo, entre outras informações.

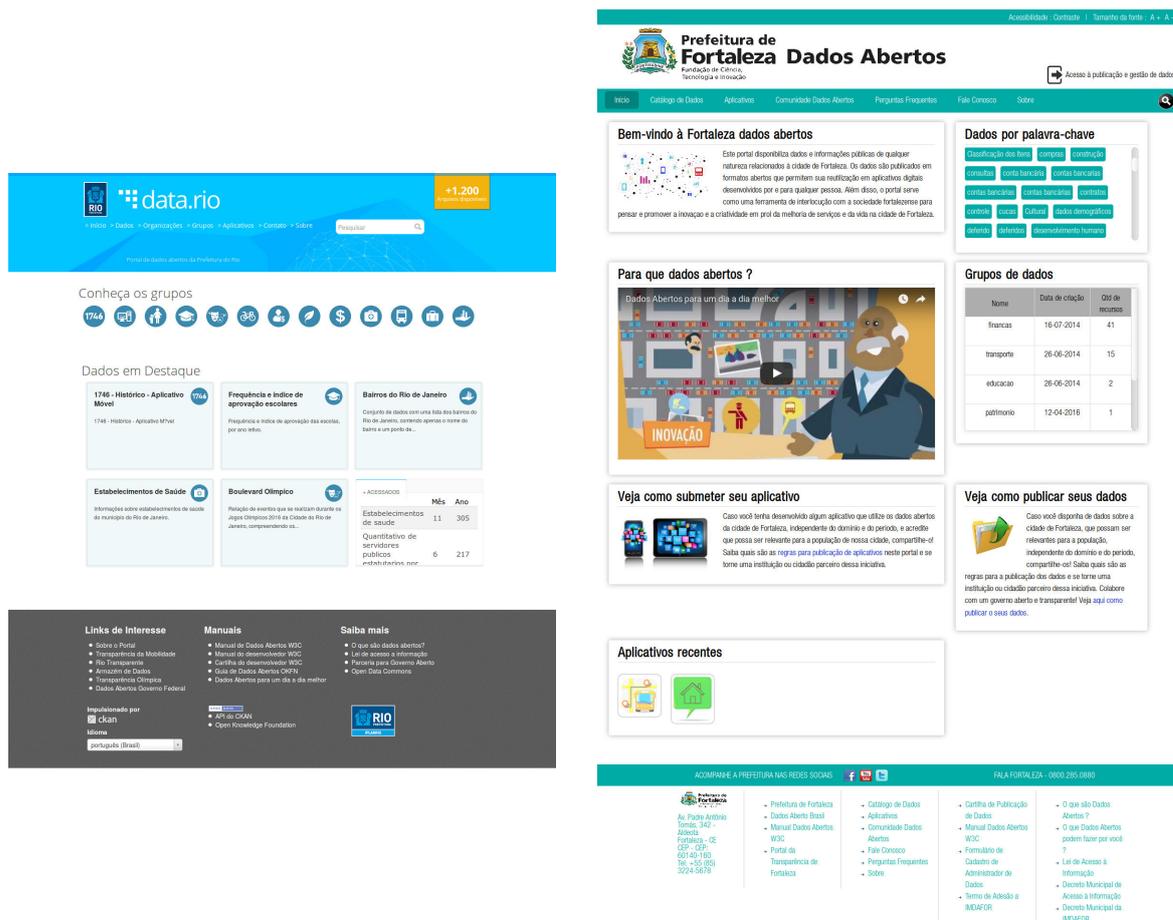
Atualmente, tais dados são coletados de forma automática por dispositivos sem que exista uma finalidade definida para esses dados. Esta crescente quantidade de dados urbanos disponíveis permite à comunidade científica o encontro de informações de valor inestimável sobre a dinâmica urbana e tudo que a cidade comporta. Raedes *et al.* (2007) comparam informações geradas pelos celulares com informações provenientes de censo demográfico. A descoberta destas informações permite encontrar padrões e formular teorias a respeito da saúde pública, do mercado imobiliário, da distribuição de transporte público, da prevenção de congestionamentos, entre outras aplicações.

Os dados urbanos possuem uma vertiginosa tendência ao crescimento, como indicado por estudos como o de Barbosa *et al.* (2014). Segundo os autores, existe uma moção dos municípios no sentido de disponibilizar dados gerados pelos seus governos, devido às políticas de transparência e existe grande valor no que são chamados de dados abertos, pois aumentam a transparência e criam oportunidades para novos usos e aplicações dos dados.

Em conformidade com a Lei brasileira de Acesso à Informação (Governo Federal, 2015), vários sites governamentais têm disponibilizado dados urbanos abertamente, tais como o **data.rio** (Prefeitura do Rio de Janeiro, 2015) e o **Dados Abertos Fortaleza** (Prefeitura de Fortaleza, 2015). Os portais mencionados podem ser vistos na Figura 1.1. A iniciativa é um passo importante para que tais dados se tornem conhecimento útil na prática, porém, para que isto ocorra, os dados precisam ser primeiramente documentados, lapidados e adequadamente representados. No entanto, atualmente a maioria das iniciativas públicas têm se comportado apenas como um repositório de dados. Ainda faltam esforços para integrar informações de diferentes fontes. Ferramentas de visualização podem ser usadas para explorar dados como estes e possivelmente descobrir relações entre eles.

Seguindo essa tendência de disponibilização de dados governamentais abertamente estão iniciativas, cada vez mais comuns, como o Hacker Cidadão (Fundação de Ciência Tecnologia e Inovação de Fortaleza, 2016). O Hacker Cidadão é um concurso que incentiva o desenvolvimento de soluções tecnológicas baseadas em dados públicos que possam contribuir positivamente para a vida das pessoas. A própria sociedade já reconhece que dados urbanos públicos podem ser de grande utilidade e o Hacker Cidadão é uma prova concreta disto.

Figura 1.1 – Portais Dados Abertos Rio (à esquerda) e Dados Abertos Fortaleza (à direita).



Fonte: Prefeitura do Rio de Janeiro (2015) e Prefeitura de Fortaleza (2015).

Existem páginas Web como o portal da Infraestrutura Nacional de Dados Espaciais (2016) em que é possível visualizar alguns mapas com dados espaciais públicos e também obter esses dados espaciais, porém na visualização do INDE, por exemplo, o número de filtros que pode ser aplicado é limitado, e, além disso, não é possível fazer anotações nos gráficos ou visualizar mais do que um gráfico ao mesmo tempo.

Exemplos interessantes de como a Visualização de dados tem se integrado à vida das pessoas são o Wikicrimes (Furtado *et al.*, 2010) e o Onde Fui Roubado (2016). São sistemas de fácil uso e aceitação que utilizam dados espaciais presentes no cotidiano de seus usuários. Há também sistemas que utilizam a Visualização de dados para auxiliar na prevenção e combate de doenças como a dengue como o Dengue-GIS (Hernández-Ávila *et al.*, 2013), o trabalho de Vinh, Nguyen e Tran (2014), entre outros (Porcasi *et al.*, 2012; Kienberger *et al.*, 2013; Neto *et al.*, 2014). Os sistemas citados configuram-se como exemplos positivos de como a era da Informação e a Visualização trouxeram benefícios para as pessoas, que têm se envolvido e se familiarizado mais com gráficos através de sistemas assim.

Contudo, é perfeitamente possível e provável que as pessoas tenham questionamentos que não podem ser respondidos por um sistema de Visualização Web de propósito

específico. Para isto, existem sistemas de Visualização Web de propósito geral, tais como como ManyEyes (Viegas *et al.*, 2007), Plotly (Inc., 2015), Quadrigram (Aguirre *et al.*, 2016), Tableau (Tableau, 2016) e Qlik (Qlik, 2016). Para um público mais técnico, há ainda opções como Protovis (Bostock e Heer, 2009), D3 (Mike Bostock, 2016), Processing (Processing, 2016) e Vega-Lite (Satyanarayan *et al.*, 2017). Estes sistemas permitem que o usuário visualize os seus próprios dados e decida que perguntas quer fazer, que visualizações deseja utilizar e qual o mapeamento utilizado (cores, formas, posição).

Essas ferramentas possibilitam a visualização da informação, porém normalmente possuem algum grau de limitação para o público leigo. Muitas vezes são difíceis de usar, limitadas a programadores ou designers de visualização. Além disso, para a Visualização de dados urbanos espaço-temporais, é importante que o usuário consiga analisar seus dados no nível de detalhe que desejar. Ademais, pode ser que o usuário deseje também criar visualizações que combinam dados que estejam em níveis de detalhes diferentes.

Porém, dependendo do caso, o usuário terá que partir para as ferramentas mais avançadas (isto é, ferramentas para o público técnico) pois não encontra uma forma de criar a visualização almejada nas ferramentas mais simples. Então, se o usuário não estiver disposto a enfrentar uma curva de aprendizado para aprender uma nova ferramenta que possibilitará que ele visualize os dados que possui, ele poderá até mesmo perder o interesse.

As ferramentas Web mais simples de Visualização de dados de propósito geral como Plotly e Quadrigram não permitem que o usuário analise dados segundo um mapa importado por ele, isto é, segundo um arquivo de geometria em que cada componente geométrico é um polígono e representa uma região geográfica. No Tableau Desktop, uma ferramenta mais sofisticada, importar um arquivo de mapa é possível, porém não é uma tarefa trivial.

1.2 Objetivos

Neste trabalho, é apresentado o CrabVis, uma arquitetura para um sistema de visualização de dados espaço-temporais com foco na facilidade de uso, interatividade e integração de diversas fontes de dados. O objetivo principal da arquitetura é facilitar a visualização de dados urbanos para cidadãos comuns, provendo um sistema intuitivo e que facilita a integração de diferentes fontes.

Os tipos de dados espaciais abordados pela arquitetura são dados georreferenciados estáticos, ou seja, dados georreferenciados cuja posição não varia no tempo e dados espaciais nominais como nome de cidade, município ou país. Adicionalmente, os dados também poderão conter marcas temporais - e poderão estar em diversos níveis de granularidade (níveis de detalhe).

A preocupação com a granularidade dos dados do usuário é um atributo frequentemente negligenciado em sistemas de Visualização mais simples, porém possui elevada importância. O CrabVis tem como uma de suas metas fazer com que os usuários do

sistema consigam visualizar dados espaciais que estão também em níveis intermediários de agregações, isto é, regiões que podem ser descritas como sendo um conjunto de regiões conhecidas. Um bom exemplo de região intermediária é a divisão do município de Fortaleza em regionais. As regionais são compostas por bairros, ou seja, uma regional é um conjunto de bairros. Não é apenas um bairro mas não chega a ser um município. Está entre dois níveis administrativos.

Além da granularidade dos dados não ser uma preocupação primordial em muitos sistemas, é também um atributo pouco explorado. Ou seja, pouco se tira proveito da estrutura inerentemente hierárquica que dados espaciais possuem. Será mostrado também, através da implementação de um protótipo e de dois casos de uso, que o sistema aproveita as informações de granularidade de dados espaciais para oferecer uma interface mais inteligente ao usuário, capaz de realizar sugestões nas diferentes etapas de interação com o sistema.

1.3 Estrutura do trabalho

Esta dissertação está organizada da seguinte maneira: inicialmente, são introduzidos no capítulo 2 alguns conceitos fundamentais tratados frequentemente no decorrer desta dissertação como dados espaciais e granularidade de dados. Em seguida, no capítulo 3, é apresentada uma análise de uma série de trabalhos que compreendem a Visualização de dados espaço-temporais Web, de propósito geral e específico.

O capítulo 4 define a arquitetura proposta CrabVis. Ela é independente de um tipo específico de tecnologia, porém, a fim de ilustrar os conceitos presentes em sua especificação, um protótipo de sistema foi implementado e é descrito no capítulo 6. Neste capítulo também são descritos dois casos de uso do sistema em questão. Por fim, o capítulo 6 apresenta algumas conclusões obtidas a partir da arquitetura definida e do sistema implementado, além de direcionar linhas de atuação para trabalhos futuros.

2 Conceitos Preliminares

Este capítulo introduz os principais temas tratados no presente trabalho, tais como o que são dados espaço-temporais e qual a influência da propriedade da granularidade nesse tipo de dado. Além desses conceitos, tipos de gráficos comumente presentes na visualização de dados espaciais são apresentados e explicados.

2.1 Dados espaço-temporais

Compreende-se por dado espaço-temporal um item de dado que possui identificação espacial e marca temporal. Declarar que um dado espaço-temporal possui uma identificação espacial não é afirmar, necessariamente, que este dado possui uma coordenada geográfica com latitude e longitude. Dados espaciais compreendem também quaisquer dados que identifiquem uma determinada região geográfica nominalmente, tais como nomes de países, municípios e bairros (Andrienko e Andrienko, 2006). Este trabalho define **grupo espacial** como um atributo em um conjunto de dados que representa uma localização ou região de acordo com um nível espacial ou administrativo (por exemplo: um atributo que possui apenas nomes de cidades ou somente nomes de estados). Além disso, dados espaciais não são, necessariamente, dados bidimensionais como latitude e longitude, podendo ser constituídos também da elevação topográfica no ponto georreferenciado.

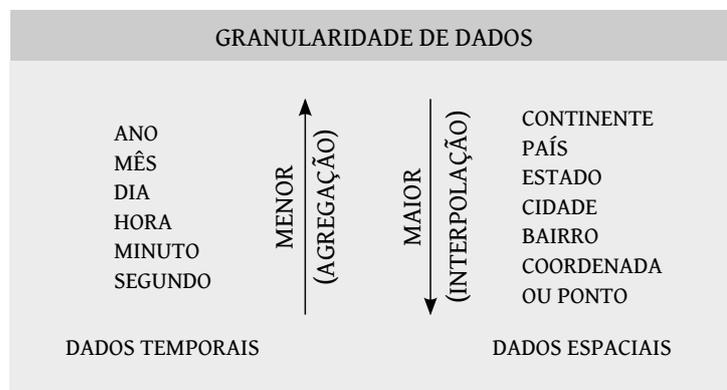
A visualização de dados espaço-temporais não é realizada de forma trivial, pois envolve uma dimensão a mais do que um gráfico bidimensional comum (o tempo). A literatura cartográfica, por exemplo, associa o aspecto temporal de um evento observável (isto é, um item de dado) à variação de suas propriedades espaciais e temáticas ao longo do tempo (Andrienko e Andrienko, 2006). Exemplos de propriedades temáticas são variações de valores numéricos e ordinais, já as propriedades espaciais variam em quesitos como altitude, posição e orientação.

2.2 Granularidade de dados

A granularidade define o quão detalhado um dado pode ser (Figura 2.1). Tratar com eficiência esta propriedade dos dados é fundamental em sistemas de visualização pois, dependendo dos níveis de granularidade, o usuário poderá caminhar por diferentes níveis de abstração dos dados. Andrienko e Andrienko (2006) caracterizam a travessia entre os

diversos níveis de granularidade no momento da análise dos dados pelas expressões “*drill up*” e “*roll down*”, sendo a primeira para descrever um aumento no nível de abstração e a segunda para descer para um nível de mais granularidade e, conseqüentemente, mais detalhe.

Figura 2.1 – Granularidade de dados.



Fonte: Elaborada pelo autor.

A granularidade dos dados influencia a análise de dados que possuem até mesmo atributos numéricos e categóricos, pois é possível analisar cada atributo com um novo enfoque dependendo do nível de granularidade considerado. Em campos numéricos, é possível passar de uma granularidade maior para menor através de uma operação de agregação, como soma ou média. Caso uma pessoa queira analisar seus dados mensalmente (considerando a granularidade temporal de mês) e os dados importados forem especificados diariamente, ele poderá obter o total mensal através de uma operação de soma em que cada valor numérico correspondente a um mesmo mês é somado.

Para passar de uma granularidade menor para uma maior, é necessário aplicar interpolação de valores. Um exemplo de operação desse tipo seria ter um campo contendo o acúmulo anual de precipitação pluviométrica. Para aumentar a granularidade e obter a precipitação mensal, o valor pode ser dividido por doze e replicado para cada mês, por exemplo. Os tipos de agregação e interpolação a serem aplicados dependem da aplicação.

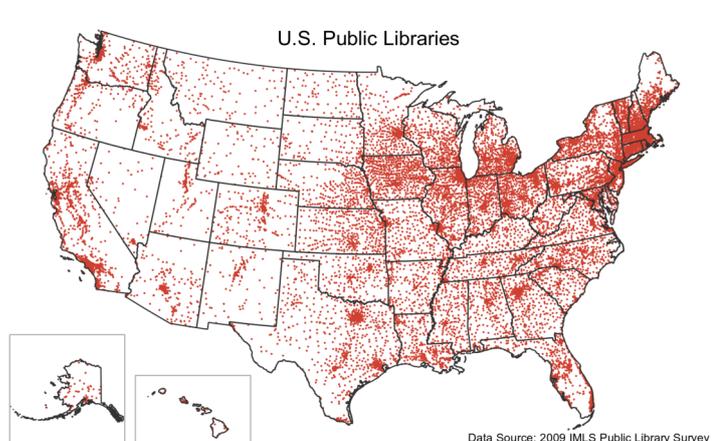
2.3 Visualização de dados espaciais

Esta seção descreve e exemplifica algumas técnicas de visualização de dados espaciais como mapa de pontos, mapa temático e mapa de calor. Essas técnicas devem estar disponíveis em qualquer sistema de visualização de dados espaço-temporais. Cada subseção define e detalha como é construída a visualização correspondente.

2.3.1 Mapa de pontos

Um mapa de distribuição de pontos, ou, mais abreviadamente, mapa de pontos, mostra a disposição de um conjunto de pontos em uma região geográfica. A visualização da distribuição de um conjunto de pontos num mapa permite a observação rápida de áreas mais ou menos densas em cada região do mapa. Áreas mais densas possuem uma maior concentração de pontos. A Figura 2.2 exemplifica um mapa de pontos em que cada ponto corresponde à localização de uma biblioteca pública nos Estados Unidos. Os pontos presentes no mapa geralmente representam ocorrências de um dado fenômeno e podem ser coloridos de acordo com alguma propriedade inerente a cada instância observada do fenômeno (cada ponto).

Figura 2.2 – Bibliotecas públicas dos Estados Unidos no ano de 2009.



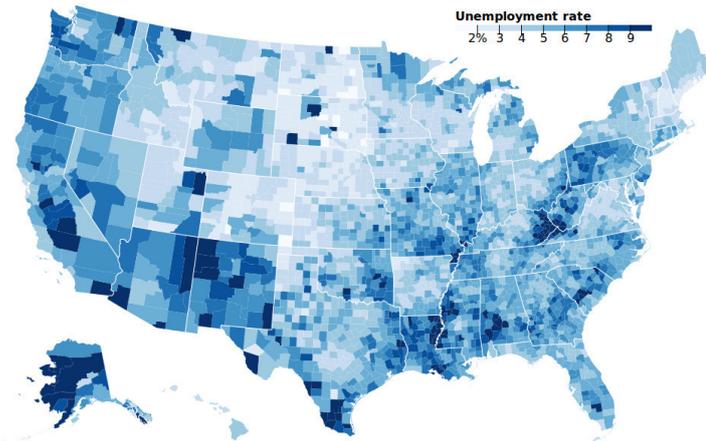
Fonte: Disponível em http://gothos.info/wp-content/uploads/2013/03/publibs_2009.png.

2.3.2 Mapa temático

Mapas temáticos colorem as unidades constituintes de uma dada geometria baseando-se em valores que são mapeados por uma escala de cor sequencial ou divergente. Mapas (ou escalas) de cores sequenciais variam apenas um atributo de cor, seja este atributo matiz, luminância ou saturação, de um valor mínimo até um máximo. Mapas de cores divergentes tipicamente representam o começo e o fim do espectro de cor por cores fortes e contrastantes (como vermelho e azul) (Kirk, 2012). A Figura 2.3 mostra um mapa temático que ilustra as taxas de desemprego nos Estados Unidos a partir de agosto de 2016 utilizando um mapa de cores sequencial.

Países divididos em estados e estados divididos em municípios são exemplos de geometrias comumente utilizadas para a construção de mapas temáticos, porém a construção de tais mapas pode ser realizada em qualquer geometria que contenha polígonos simples.

Figura 2.3 – Taxas de desemprego nos Estados Unidos a partir de agosto de 2016.



Fonte: Disponível em <https://bl.ocks.org/mbostock/4060606>.

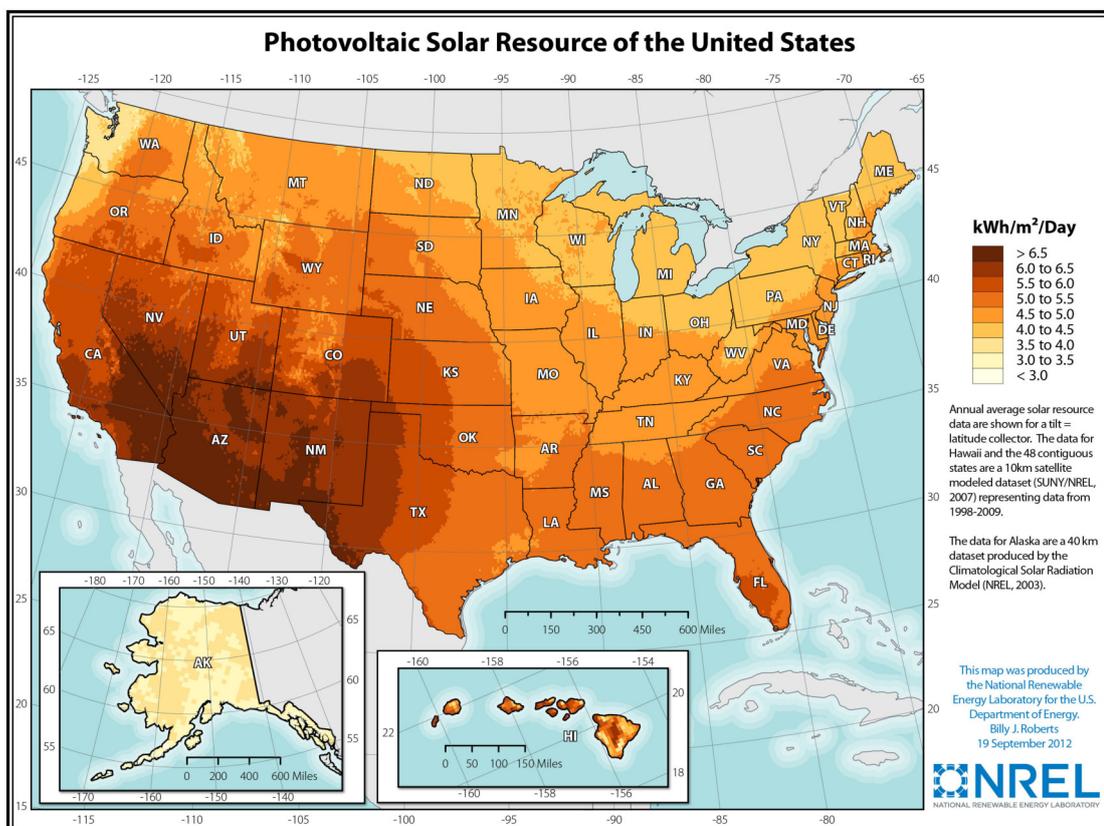
2.3.3 Mapa de calor

O mapa de calor representa uma região graficamente dividindo-a em células e atribuindo uma cor para cada célula. O mapa de calor é uma visualização que não é exclusiva para dados espaciais, mas que é comumente utilizada para tal fim. Mapas de calor podem ser mapas que representam a densidade ou a intensidade de pontos.

Um mapa de calor que propõe a representação da intensidade de pontos mapeia cada célula segundo um valor quantitativo (por exemplo, uma temperatura), enquanto o mapa de calor de densidade de pontos representa, de forma semelhante ao mapa de distribuição de pontos, a concentração de pontos em um mapa. Na Figura 2.4, o mapa de calor representa o recurso solar fotovoltaico e potencial de energia solar concentrada nos Estados Unidos em kilowatt hora por metro quadrado por dia. Quanto mais escura a cor, maior o potencial de energia solar concentrada que aquela região representa.

Munzner (2014) argumenta que um dos benefícios dos mapas de calor é conseguir prover uma boa visão geral de áreas com grande densidade de informação de uma forma bastante compacta, utilizando pequenas células com cores.

Figura 2.4 – Recurso solar fotovoltaico e potencial de energia solar concentrada nos Estados Unidos.



Fonte: Disponível em <http://www.nrel.gov/gis/solar.html>.

3 Trabalhos Relacionados

A popularização da Web e a crescente familiaridade das pessoas com as tecnologias Web têm propiciado um ambiente de estudo para diversas áreas do conhecimento, pois agora é possível analisar facilmente a interação das pessoas com a tecnologia. Entre estas áreas, encontra-se a Visualização de dados, que é uma área de pesquisa que almeja apresentar dados, das mais diversas naturezas, da forma mais eficiente possível. Munzner (2014) definiu sistemas de visualização baseados em computador como sistemas que proveem representações visuais de conjuntos de dados criadas para ajudar as pessoas a executar tarefas mais facilmente.

É num contexto de intensa geração de informações e uma crescente demanda por descobertas provenientes dessas informações que os sistemas de visualização para Web tem florescido. Embora ferramentas de visualização tenham se popularizado, nem todas as ferramentas são adequadas para leigos. Algumas delas requerem conhecimento técnico sobre como programar ou como manipular dados. Este quadro está mudando e muito tem sido discutido sobre levar a visualização para as massas (Weessies e Dotson, 2013). Neste capítulo, serão analisadas algumas ferramentas de Visualização e sistemas Web que possuem variados níveis de dificuldade (para iniciantes). No presente trabalho, faz-se distinção entre três categorias de sistemas Web: linguagens e bibliotecas, sistemas de Visualização para uso geral e sistemas de Visualização de uso específico.

3.1 Linguagens e Bibliotecas

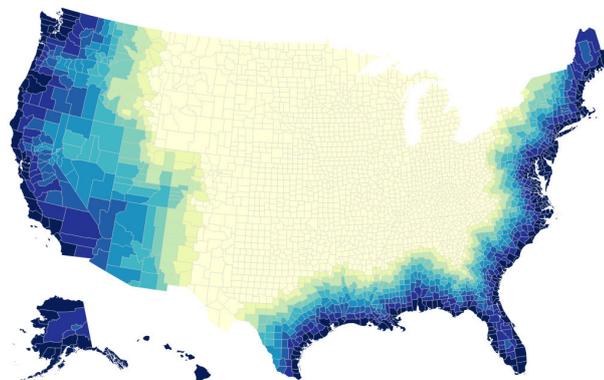
3.1.1 Protovis, D3 e Processing

O Protovis (Bostock e Heer, 2009) é uma ferramenta de visualização de dados criada para diminuir a lacuna semântica entre sistemas de visualização de alto nível e a expressividade que sistemas gráficos como APIs de renderização possuem. O propósito da biblioteca é possibilitar a criação de visualizações utilizando as funções de alto nível disponibilizadas pela API. Tanto o Protovis quanto o D3 requerem um mínimo de conhecimento de programação e envolvem uma curva de aprendizado como qualquer biblioteca em Javascript.

Como o Protovis só criava gráficos 2D sem interatividade, um de seus idealizadores, Mike Bostock, desenvolveu uma evolução mais poderosa do Protovis, o D3 (Bostock *et al.*, 2011). A sigla D3 significa Data Driven Documents (Documentos direcionados aos dados).

Esta sigla faz referência à forma que a biblioteca funciona, associando manipulações nos dados a manipulações na estrutura DOM do HTML. A Figura 3.1 mostra um mapa temático criado por Mike Bostock para demonstrar a biblioteca.

Figura 3.1 – Este mapa mostra a distância de cada condado dos Estados Unidos para a costa do Pacífico ou do Atlântico. O gráfico foi criado com o D3.



Fonte: Mike Bostock's Block. Disponível em: <http://bl.ocks.org/mbostock/9744818>.

O D3 não é uma biblioteca de criação de gráficos propriamente dita no sentido que não é possível carregar um conjunto de dados, escolher um tipo de visualização, especificar alguns parâmetros e obter um resultado visual. O D3 realiza manipulações de baixo nível na estrutura do HTML, SVG ou Canvas para criar qualquer tipo de associação que o designer da visualização queira especificar (e que pode não ser, necessariamente, um gráfico). Os exemplos de uso do D3 são variados e podem ser desde uma ação simples como carregar um conjunto de dados e criar uma lista enumerada numa página HTML contendo os dados ou criar, com os mesmos dados, um gráfico de barras.

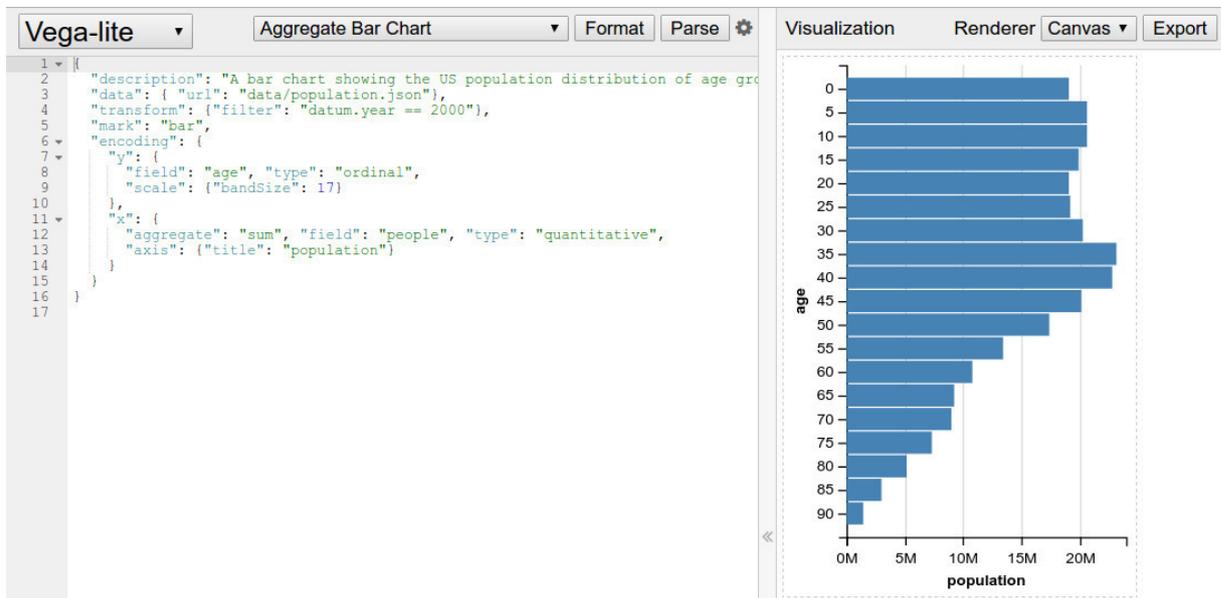
Além da utilização de bibliotecas para renderização de gráficos, é possível também renderizar gráficos utilizando uma linguagem de programação de alto nível como o Processing (Processing, 2016), por exemplo. O Processing possui um ambiente de desenvolvimento integrado próprio em que até mesmo programadores iniciantes possuem um retorno visual do que codificaram. Porém, ainda assim, criar uma visualização no Processing requer o aprendizado da linguagem e envolve conhecimento técnico sobre a codificação de visualizações. Existe ainda uma biblioteca para Javascript que busca levar as mesmas funcionalidades do Processing para a Web, a Processing.js (Processing.js, 2016).

3.1.2 Vega

A Vega (2016) é uma gramática declarativa para criação de visualizações que possibilita compartilhamento e reuso de visualizações. Para criar uma visualização utilizando esta gramática, o usuário deve criar um arquivo json que terá todas as configurações do seu gráfico. Nesse arquivo json, além de especificar quais os dados que o usuário deseja analisar, ele deverá especificar o tipo gráfico que deseja criar, quais serão as variáveis

analisadas, quais cores representarão cada variável, além de informações como largura e altura do gráfico, por exemplo (Figura 3.2). A Vega-Lite é uma abstração de mais alto nível do Vega que utiliza o D3 para a representação visual dos gráficos, ou seja, ela é uma camada de abstração que está acima do D3.

Figura 3.2 – Especificação de um gráfico de barras na gramática Vega.



Fonte: Vega: A Visualization Grammar (2016).

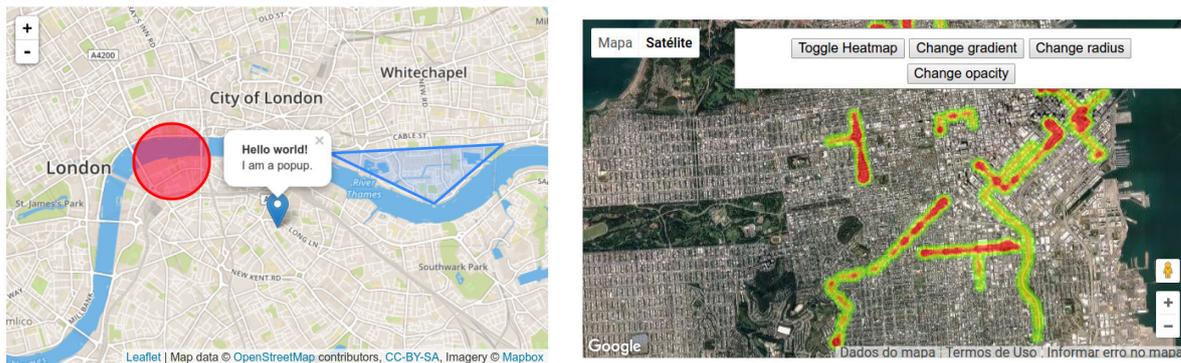
Os autores da Vega criaram posteriormente uma evolução da gramática, chamada de Vega-Lite (Satyanarayan *et al.*, 2017). A Vega-Lite foi construída em cima da Vega e, em suma, deixa a tarefa de especificar um gráfico (o arquivo json) mais concisa. Além disso, é possível compilar, ou seja, transformar uma especificação Vega-Lite em Vega.

3.1.3 Leaflet e GoogleMaps

A Leaflet (Leaflet - a JavaScript library for interactive maps, 2016) e a GoogleMaps (Google Maps Javascript API — Google Developers, 2016) são duas bibliotecas para Javascript cujo foco é a criação de mapas. É possível, nas duas bibliotecas, criar mapas de pontos, mapas temáticos e mapas de calor, entre outros. Como a Leaflet é uma biblioteca Open Source, ela possui uma série de contribuidores e, conseqüentemente, uma gama considerável de plugins. Por este motivo, é uma biblioteca amplamente difundida.

Ambas bibliotecas possibilitam que o usuário carregue *tiles* como base dos mapas. *Tiles* são imagens, geralmente de tamanho 256×256 pixels que compõem a base de mapas e representam uma região de acordo com o nível de zoom em que o mapa está. A aparência de um *tile* depende do tipo de *tiles* que o programador definiu. Por exemplo, os *tiles* podem ser o mapa de rodovias ou fotos de satélite (Figura 3.3). Geralmente, servir os *tiles* não é uma responsabilidade da API de criação de gráficos, como é o caso da Leaflet,

Figura 3.3 – Mapas criados com o Leaflet (na esquerda) e Google Maps API (na direita).

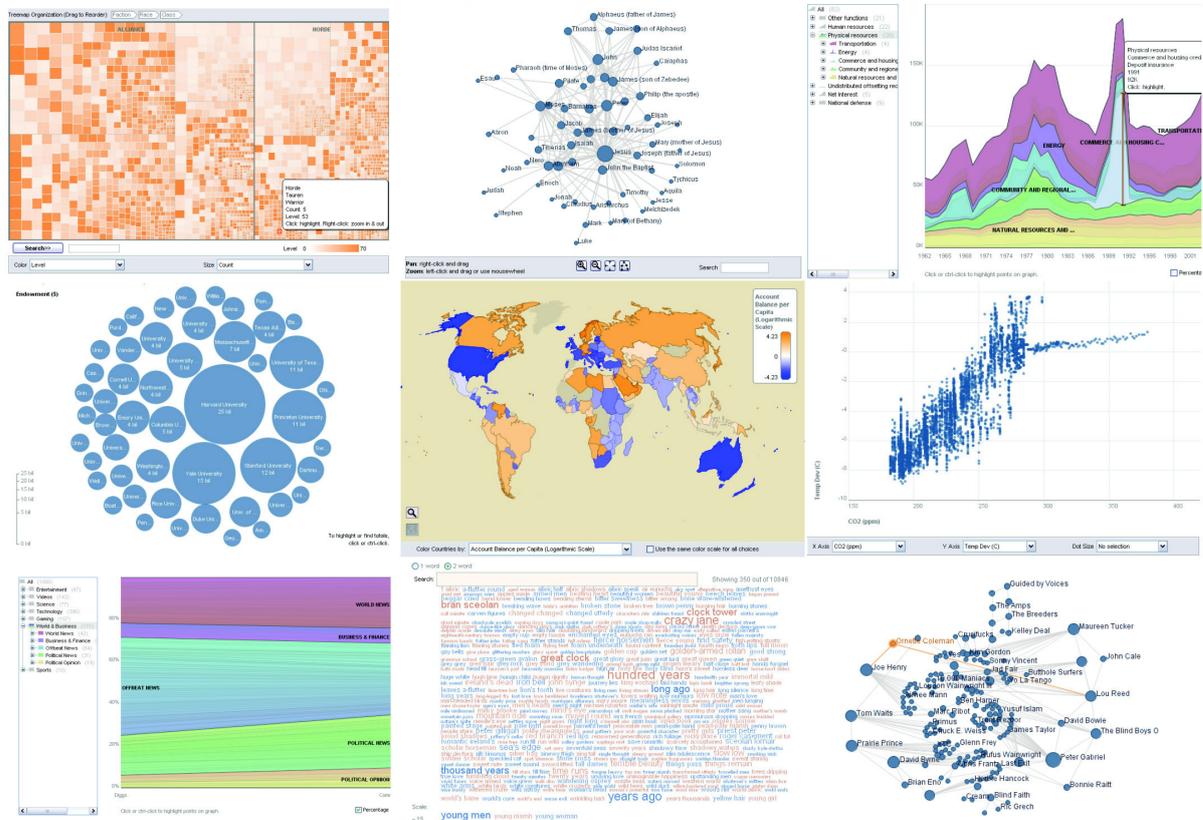


Fonte: Leaflet - a JavaScript library for interactive maps (2016) e Google Maps Javascript API — Google Developers (2016).

que utiliza um serviço terceirizado para servir os tiles para os mapas. Porém, como a Google tem um serviço próprio para fornecer os *tiles* para mapas, a GoogleMaps utiliza este serviço. Apesar de ser bastante utilizada, a API do Google possui alguns fatores limitantes para o programador (tais como limite diário de visualizações por mapa).

3.2 Sistemas de Visualização Web de propósito geral

Figura 3.4 – Many Eyes.



Fonte: Viegas *et al.* (2007).

3.2.1 ManyEyes

No ano de 2007, observando o crescimento da Web, a IBM criou o sistema de visualização ManyEyes (Viegas *et al.*, 2007). O ManyEyes foi o primeiro sistema de visualizações de dados na Web. Embora tenha sido descontinuado, enquanto estava disponível, o sistema da IBM (Figura 3.4) foi uma ferramenta colaborativa de visualização de dados para Web em que os usuários podiam importar dados, construir visualizações interativas e deixar comentários em conjuntos de dados ou visualizações de outros usuários (Viegas *et al.*, 2007).

No ManyEyes, os usuários selecionavam um conjunto de dados e uma visualização e o sistema gerava a visualização automaticamente, sem que o usuário pudesse escolher quais colunas desejava mapear.

Embora os usuários pudessem navegar em visualizações públicas e utilizar dados públicos, o ManyEyes não contemplava a criação de dashboards. Assim como o sistema proposto, o projeto da IBM também utilizava uma heurística para inferir o tipo do dado, quando o dado era tabular. A heurística, porém, só categorizava o dado como numérico

ou textual.

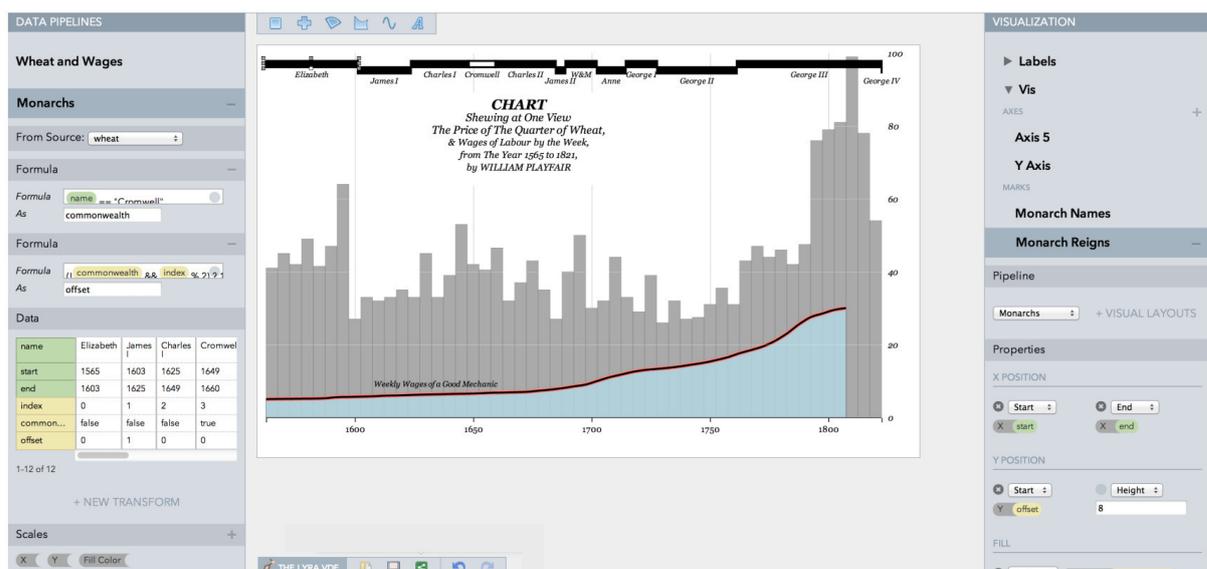
3.2.2 Tableau e Lyra

O Lyra (Satyanarayan e Heer, 2014) é um ambiente Web para a criação de visualizações de dados customizadas sem a necessidade de programar. Embora no Lyra não seja necessário codificar, ainda é ideal que o usuário tenha uma noção básica de como criar visualizações (o que requer algum conhecimento técnico). No artigo, os autores estabeleceram um público alvo de jornalistas e designers de visualização.

A criação de visualizações no Lyra é realizada através de interações drag-and-drop, porém, as visualizações criadas não são interativas. São visualizações cujo principal fim é apresentação. Na importação de dados, de forma similar ao sistema proposto, o Lyra identifica o tipo de cada variável importada. O Lyra não possibilita que o usuário crie dashboards, porém oferece a opção de exportar a visualização criada como arquivo png, SVG ou nas especificações da linguagem Vega.

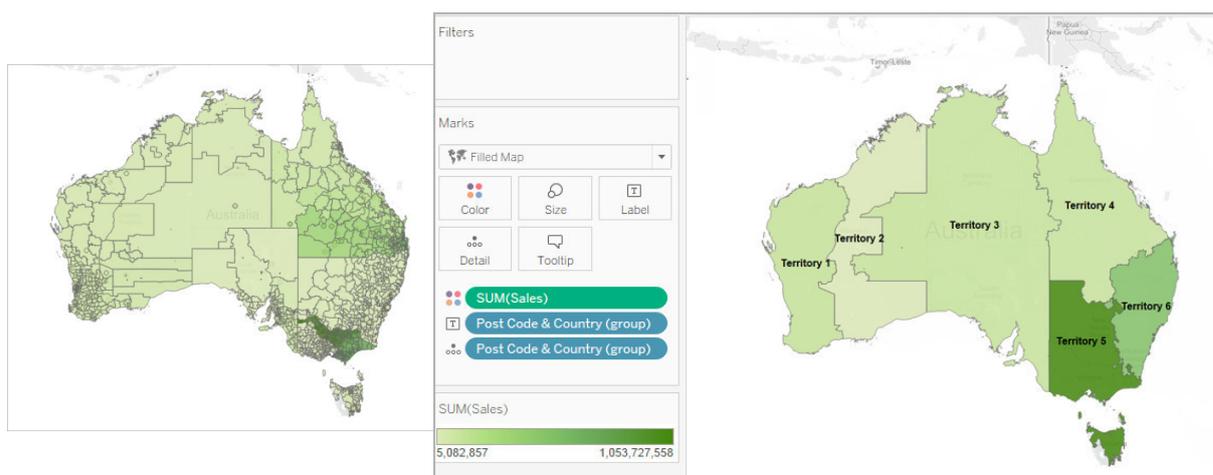
O Lyra é fortemente inspirado pelo Tableau Desktop (Tableau, 2016), um sistema comercial de visualização de dados da empresa Tableau. Essa inspiração pode ser notada ao interagir com o Lyra. De forma similar ao Tableau Desktop, o Lyra possui *drop zones*, nas quais o usuário deverá arrastar e soltar atributos dos seus dados para realizar agrupamentos entre atributos e mapeamentos visuais. A Tableau também possui uma versão gratuita do Tableau Desktop, o Tableau Public, que possibilita que o usuário envie a dashboard criada no software para a Web. O Tableau Public é gratuito, mas é dependente de plataforma, sendo disponibilizado apenas para usuários de Windows e Mac.

Figura 3.5 – Lyra.



Fonte: Satyanarayan e Heer (2014).

Figura 3.6 – Visualização feita no Tableau Desktop: Mapa original (à esquerda) e mapa dividido em territórios definidos pelo usuário (à direita).



Fonte: Disponível em: https://onlinehelp.tableau.com/current/pro/desktop/en-us/maps_custom_territories.html.

Na análise de mapas com geometrias, é possível agrupar os polígonos do mapa analisado em territórios e realizar uma análise por território definido pelo usuário ao invés de realizar análise em uma geometria pré-definida, porém importar um novo arquivo de geometria não é uma tarefa trivial. No Tableau Desktop, é possível importar arquivos de geometria do tipo Shapefile, porém, primeiramente o usuário deve utilizar um SIG (tais como QGIS ou ArcGIS) para converter os dados do Shapefile num arquivo csv, para que o Tableau consiga importar os dados.

Tanto o Tableau quanto o Lyra são sistemas que possuem uma curva de aprendizado íngreme no seu uso. O uso do Tableau, assim como o uso do Lyra, requer algum conhecimento técnico sobre a criação de gráficos e mapeamento de variáveis nos gráficos.

3.2.3 Plotly, Quadrigram e Qlik

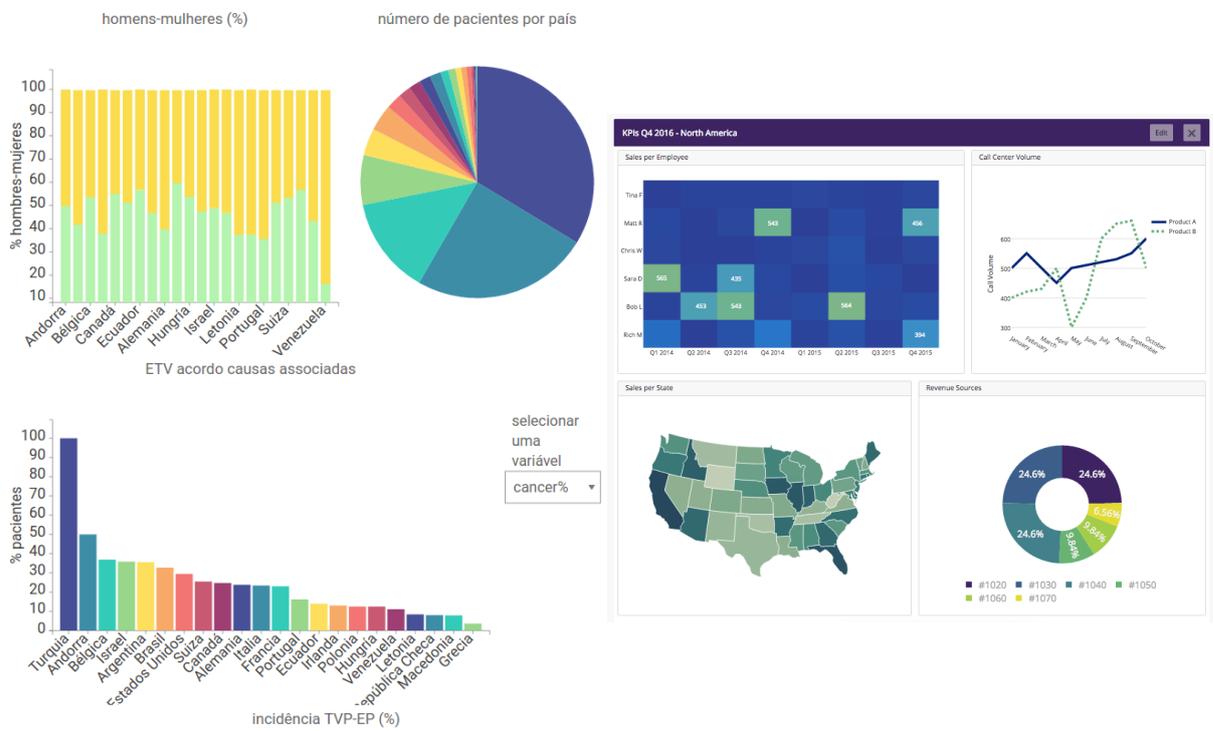
O Plotly (Inc., 2015), Quadrigram (Aguirre *et al.*, 2016) e Qlik Sense Cloud (Qlik, 2016) são ferramentas Web comerciais de Visualização que possuem planos gratuitos. O Plotly, além da ferramenta Web, disponibiliza mais uma série de bibliotecas e APIs que podem ser utilizadas com R, Python, Matlab e até mesmo Javascript e além do sistema online (Qlik Sense Cloud), a Qlik (empresa criadora do Qlik Sense Cloud), também possui aplicativo para Desktop, o QlikView.

Os três sistemas possuem a capacidade de construção de dashboards e compartilhamento dessas mesmas dashboards, sendo que no Qlik Sense Cloud este compartilhamento é limitado a até 5 pessoas no plano gratuito. A Figura 3.7 é uma imagem de uma dashboard criada com o Quadrigram e de uma dashboard criada com o Plotly e a Figura 3.8 exemplifica uma dashboard criada utilizando o Qlik Sense Cloud.

As três ferramentas oferecem a capacidade de inserir anotações na Dashboard criada. No Qlik Sense Cloud, é possível importar um arquivo KML e criar uma visualização de

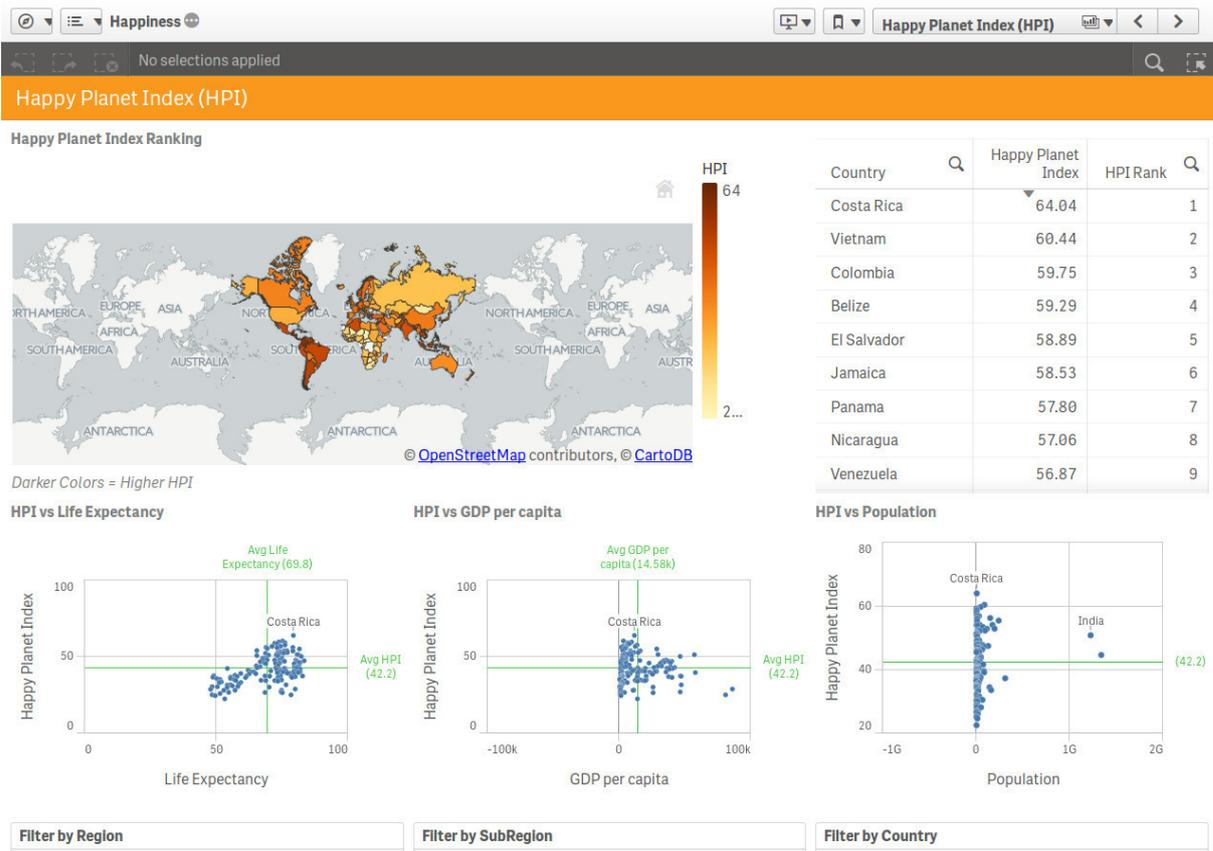
mapa temático com o mapa escolhido. Porém na criação de mapas temáticos do Plotly e do Quadrigram, não é possível importar um mapa que contenha geometria, como por exemplo um arquivo no formato Geojson (Butler *et al.*, 2016). Por este motivo, na criação de um mapa temático, os usuários dos dois sistemas em questão devem escolher algumas das opções de mapas que os sistemas ofertam. O Plotly oferta mapas mundi e um mapa dos Estados Unidos dividido em estados, já o Quadrigram oferta as mesmas opções que o Plotly e ainda o mapa do continente europeu e de Barcelona. Até mesmo ferramentas robustas como Plotly e Quadrigram não permitem que o usuário analise dados espaciais genéricos.

Figura 3.7 – Dashboard criada utilizando o Quadrigram (à esquerda) e Dashboard criada utilizando o Plotly (à direita).



Fonte: Aguirre *et al.* (2016) e Inc. (2015).

Figura 3.8 – Qlik Sense Cloud.



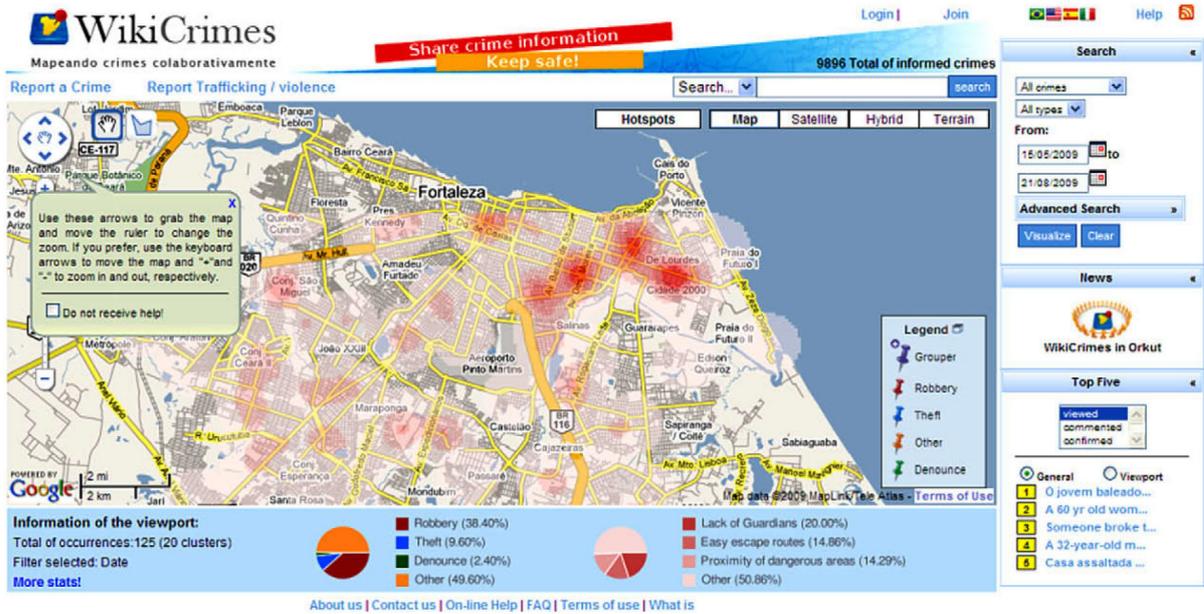
Fonte: Disponível em <https://www.qlikcloud.com/view/5804221eafe1650001f99941>.

3.3 Sistemas de Visualização Web de propósito específico

3.3.1 Wikicrimes e Onde fui roubado

O Wikicrimes (Furtado *et al.*, 2010) e “Onde fui roubado” (Onde fui roubado, 2016) são sistemas Web colaborativos com a finalidade de promover um ambiente em que os usuários colaboram registrando ocorrências de crime em um mapa. O Wikicrimes é modelado como um sistema multiagente e tira proveito dessa modelagem multiagentes para, dentre outras coisas, certificar a veracidade da informação através de entidades certificadoras e do feedback de outros usuários. No caso do “Onde fui roubado”, não há uma forma de verificar se a informação inserida no sistema é válida.

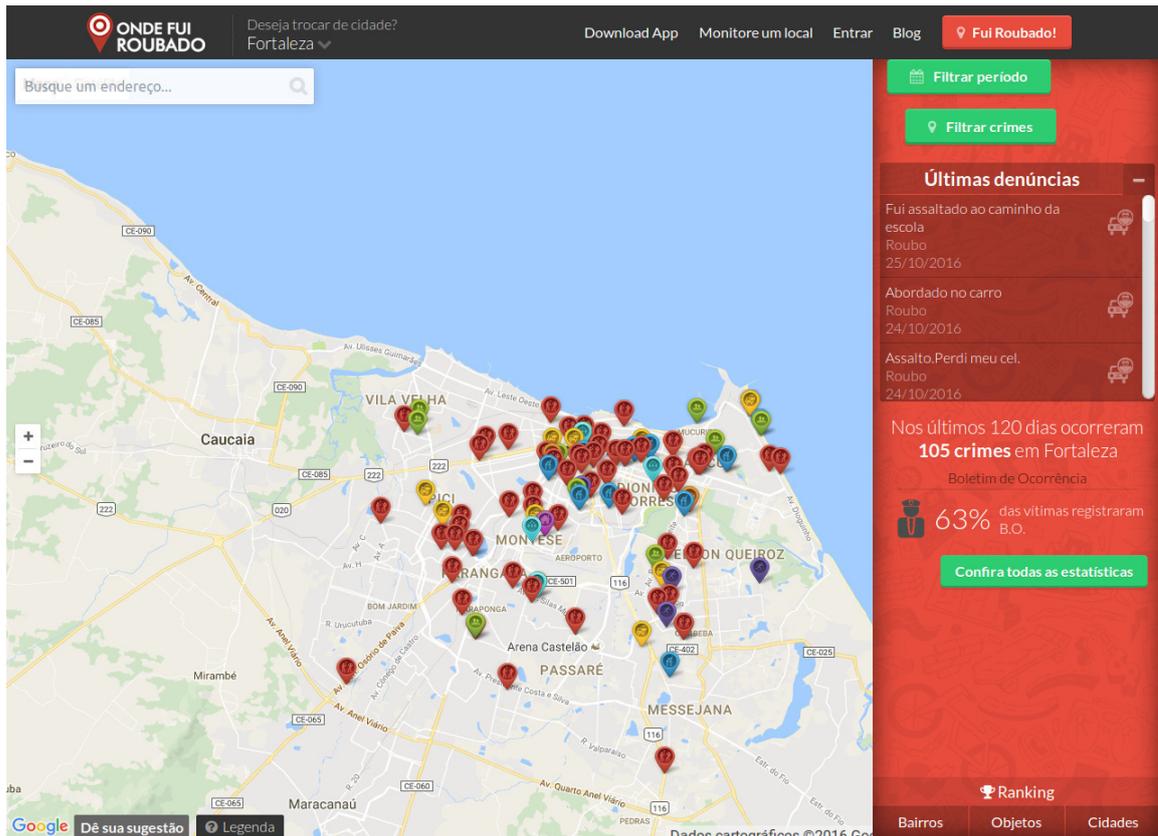
Figura 3.9 – Wikicrimes.



Fonte: Furtado *et al.* (2010).

Tanto o Wikicrimes quanto o “Onde fui roubado” são sistemas que possuem elementos de Visualização de dados. Como pode ser observado na Figura 3.9, no Wikicrimes os usuários podem interagir com um mapa de calor da região de interesse através de zoom, pan e filtros como a data e o tipo do crime (Furtado *et al.*, 2010). No “Onde fui roubado”, o usuário interage de forma similar, porém essa interação se dá principalmente com um mapa de símbolos em que cada símbolo representa uma ocorrência (Figura 3.10). Existe uma tela no sistema “Onde fui roubado” em que o usuário também pode ver um mapa de calor e algumas estatísticas sobre os dados como quais crimes acontecem mais frequentemente.

Figura 3.10 – Onde fui roubado.

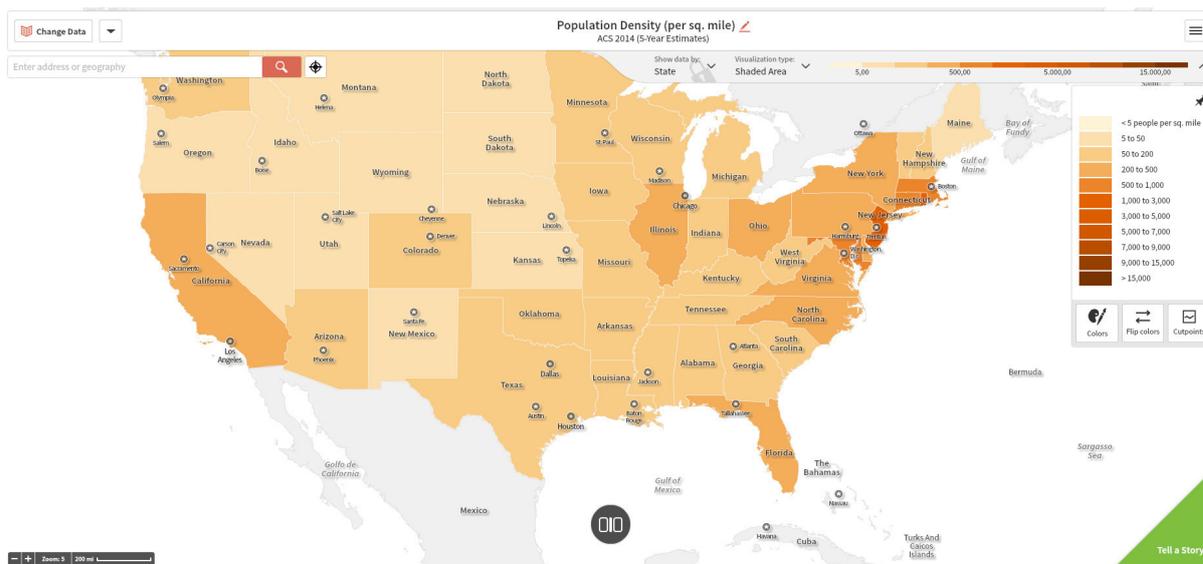


Fonte: Disponível em <http://www.ondefuiroubado.com.br/fortaleza/CE>.

3.3.2 Social Explorer

O Social Explorer (Beveridge *et al.*, 2016) é uma ferramenta criada na Universidade de Oxford que possui dados de censo dos Estados Unidos dos anos de 1790 até o ano de 2010. A interface do sistema permite que o usuário interaja com o mapa norte-americano e visualize uma série de dados que já estão presentes no sistema, referentes aos vários aspectos contemplados no censo. O mapa dos Estados Unidos pode ser analisado em uma série de granularidades diferentes, todas já pré-cadastradas no sistema. De acordo com o nível de zoom do mapa, este pode assumir uma granularidade ou outra. O sistema também permite que o usuário mude a escala de cor no mapa para escalas pré-definidas ou para uma escala de cor criada pelo usuário (Figura 3.11).

Figura 3.11 – Social Explorer.



Fonte: Beveridge *et al.* (2016).

Como argumentado em Wessies e Dotson (2013), o recente aumento de mapas publicados na Web tem despertado nos consumidores a percepção de que mapas são uma forma eficiente de apresentar dados e tem aumentado o interesse nessas pessoas de criar os seus próprios mapas. Porém, softwares SIG (Sistemas de Informação Geográfica), também denominados de GIS, tais como o Quantum Gis, requerem um certo nível de treinamento para a produção de resultados e nem todo usuário tem acesso a este tipo de treinamento. Por este motivo, ferramentas como o Social Explorer têm prosperado.

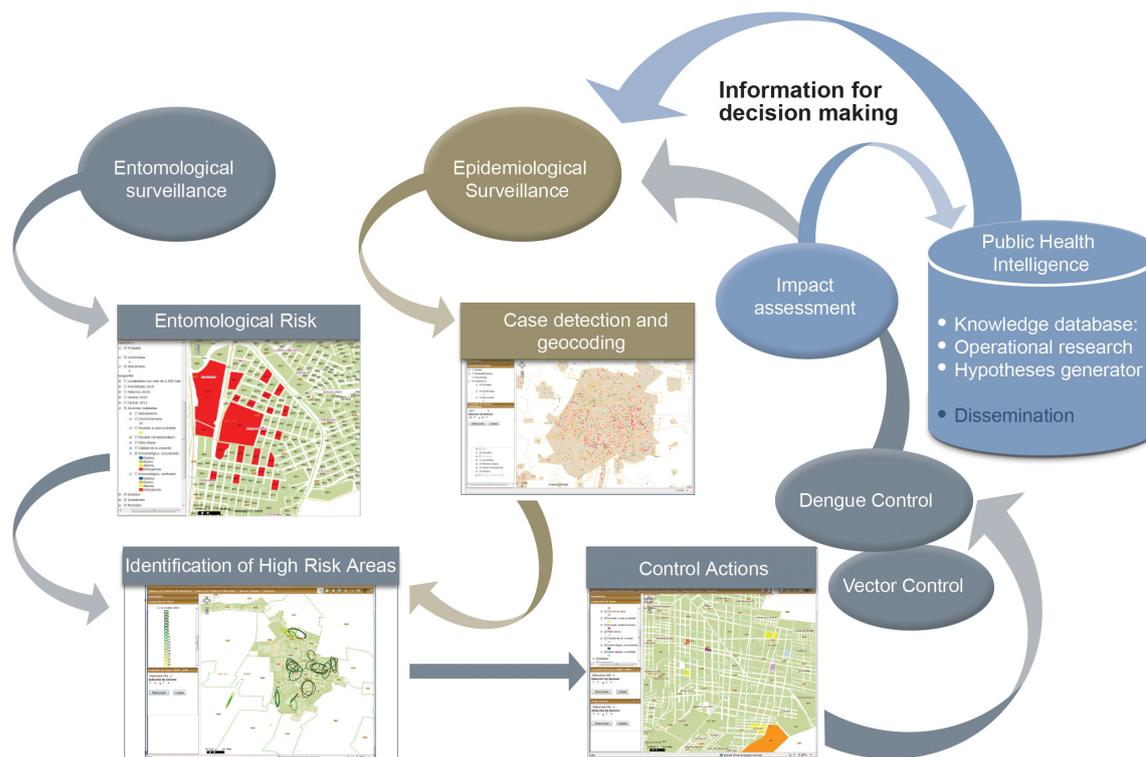
3.3.3 Dengue-GIS

Hernández-Ávila *et al.* (2013) propuseram o Dengue-GIS, um sistema para vigilância de surtos epidêmicos no México utilizando elementos de sistemas georreferenciados (SIGs) e aprendizagem de máquina com a finalidade de controlar e prevenir o risco entomológico e epidemiológico da dengue. O sistema foi proposto para monitoramento e auxílio à escolha de políticas preventivas contra a doença.

Um exemplo de técnica de Aprendizagem de Máquina utilizada no Dengue-GIS é a utilização de um algoritmo de clusterização hierárquica de vizinho mais próximo para identificar prováveis áreas de acumulação de casos de dengue e tomar as medidas cabíveis para o combate da doença (Figura 3.12).

Em seu trabalho, Carroll *et al.* (2014) fazem um extensivo estudo sobre as funcionalidades e a usabilidade de programas para visualização e análise de doenças infecciosas. Um dos principais pontos destacados no estudo é a intuitividade e fácil uso que estes sistemas devem possuir e uma das conclusões de seu trabalho é que sistemas Web têm sido a principal plataforma de escolha em muitos casos. O artigo enfatiza ainda o emprego

Figura 3.12 – Dengue-GIS.



Fonte: Hernández-Ávila *et al.* (2013).

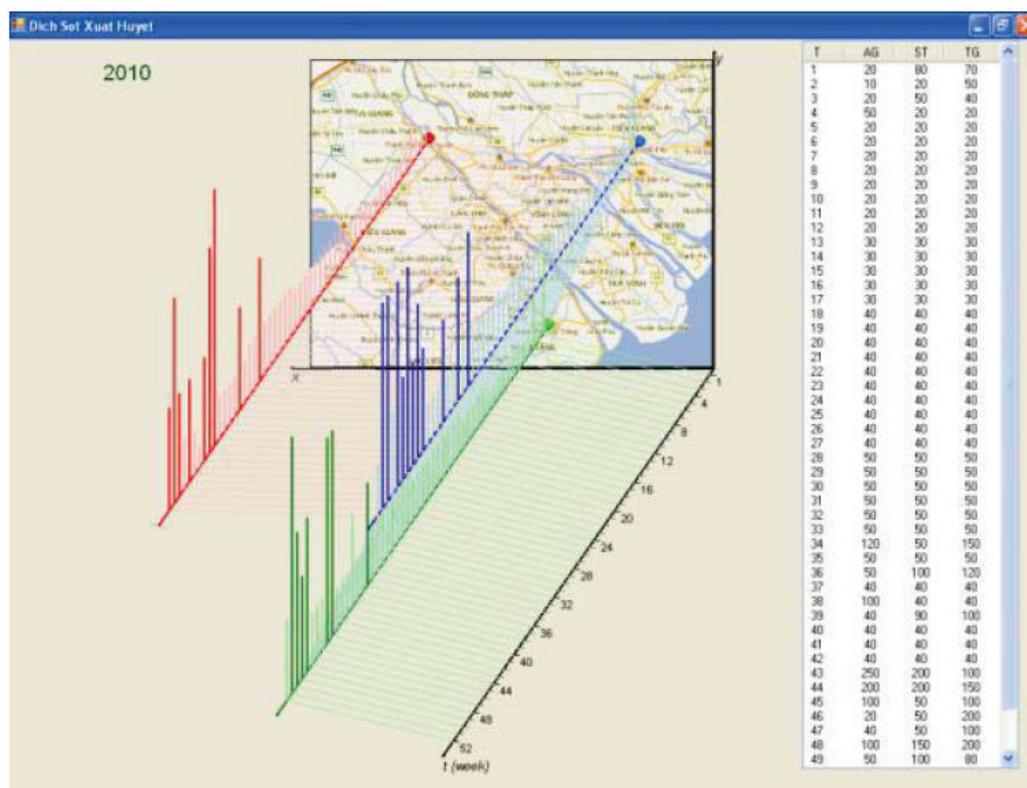
de sistemas georreferenciados para a melhor compreensão da distribuição geoespacial de enfermidades.

3.3.4 Sistema de alerta de epidemia de dengue no Vietnã

Uma abordagem comum é tratar os dados espaço-temporais como um cubo espaço-temporal. No cubo-espaço temporal, cada célula no cubo de dados corresponde a uma combinação única de valores para as dimensões (tempo e espaço) e um valor quantitativo (Andrienko *et al.*, 2003). Esta é uma forma densa e complexa de representação de dados, o que torna a visualização de difícil leitura e interpretação.

Em Bach *et al.* (2014), os autores listam e explicam uma série de operações e técnicas frequentemente associadas a esta abordagem. Vinh, Nguyen e Tran (2014) utilizaram esta abordagem no seu trabalho sobre um sistema de alerta de epidemia da dengue no Vietnã (Figura 3.13). Além do Dengue-GIS e do trabalho de Vinh, Nguyen e Tran (2014), pode-se citar alguns exemplos em que sistemas Web georreferenciados foram usados para estudo do problema da dengue (Porcasi *et al.*, 2012; Kienberger *et al.*, 2013; Neto *et al.*, 2014).

Figura 3.13 – Cubo multidimensional representando a ocorrência de dengue em três locais diferentes (representados pelos três gráficos em vermelho, verde e azul).



Fonte: Tran *et al.* (2014).

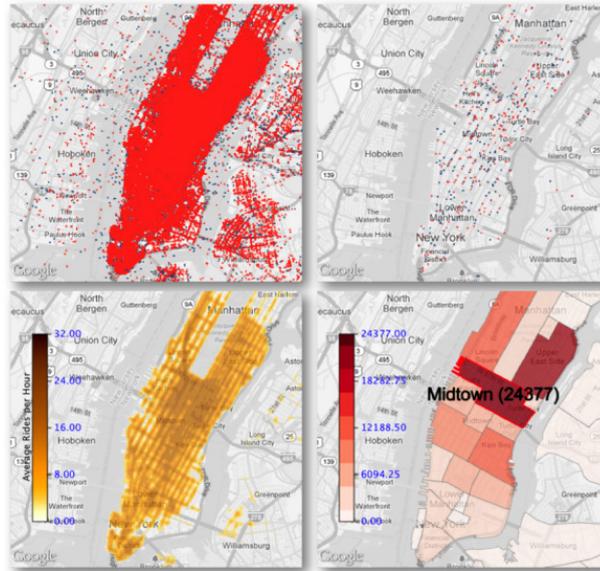
O eixo do tempo, cuja unidade está em semana epidemiológica, está em preto e é compartilhado pelos três gráficos. Na tabela ao lado, cada linha representa uma semana epidemiológica e cada coluna representam um local selecionado. Cada item da tabela mostra a ocorrência de dengue em um determinado local e semana epidemiológica.

3.3.5 TaxiVis

Ferreira *et al.* (2013) propuseram um modelo que possibilita que os usuários consultem visualmente dados de corridas de táxi da cidade de Nova Iorque. Os dados eram inerentemente espaço-temporais pois envolviam a posição e o momento de cada táxi ao longo do tempo. Os autores aplicam técnicas para lidar com dados espaço-temporais, a saber: algoritmos de **LOD** (*level of detail*) para redução da nuvem de pontos e mapas de calor para representação da distribuição dos pontos pelo mapa (Figura 3.14). Além disso, as técnicas de visualização mencionadas na seção 2.3 (mapa de pontos, mapa de calor e mapa temático) foram utilizadas no referido trabalho, como pode ser visto na Figura 3.14.

O trabalho trata de objetos móveis e concentra-se em técnicas para problemas cujo foco é a trajetória dos objetos (também chamados de problemas de origem-destino, ou OD). Existem técnicas específicas para tais problemas, tais como *Flow Maps* (Guo, 2009) e *Flowstrates* (Boyandin *et al.*, 2011).

Figura 3.14 – Técnicas presentes em Ferreira *et al.* (2013).



Fonte: Ferreira *et al.* (2013).

O resultado de um algoritmo do tipo **LOD** pode ser visto no quadrado superior direito. Na porção inferior esquerda, um mapa de calor. Na porção inferior direita, um mapa Choropleth.

No presente capítulo, foram analisados alguns sistemas Web que possuem capacidade de visualização de dados espaciais e espaço-temporais. Linguagens e bibliotecas são ferramentas que podem ser utilizadas para tal fim, mas que requerem um certo nível de conhecimento sobre programação e sobre Visualização de dados. Os sistemas Web de propósito específico são sistemas que podem possuir elementos de Visualização de dados espaciais e espaço-temporais, mas que não possuem a capacidade do usuário poder analisar outros tipos de dados. Os dados que são visualizados em sistemas de propósito específico são vinculados ao problema retratado naquele sistema.

Em outra categoria, estão os sistemas Web de visualização de propósito geral. São sistemas que possuem capacidade de visualização de dados importados pelo usuário, independente do conteúdo tratado nos dados (ou seja, independente de domínio). O fato de que esses sistemas são voltados para a Web não quer dizer que eles sejam voltados para um público leigo. No tocante à visualização de dados espaço-temporais e espaciais, também é possível observar que o usuário que deseja visualizar este tipo de dado em um sistema de propósito geral encontra algumas limitações, sejam elas a falta de intuitividade ou mesmo a falta de visualizações próprias para estes tipos de dados.

Este trabalho propõe CrabVis, uma arquitetura de sistema Web de Visualização com foco na visualização de dados espaço-temporais e espaciais. A arquitetura em questão é detalhada no capítulo 4. Um sistema baseado na arquitetura proposta foi implementado e utilizado na criação de dois casos de uso. O sistema e os casos de uso são descritos no capítulo 5.

4 Arquitetura CrabVis

A arquitetura CrabVis é definida nesta seção. Inicialmente, uma visão geral é apresentada e, em seguida, um workflow operacional de funcionalidades definidas na arquitetura é estabelecido. Finalmente, as estruturas de dados previstas pela arquitetura CrabVis são especificadas.

4.1 Visão geral da arquitetura proposta

A finalidade da arquitetura CrabVis é ser a base de um sistema Web de visualização com foco nos dados espaço-temporais ou puramente espaciais. A arquitetura é definida em camadas e segue o modelo cliente-servidor, de forma que usuários em meios heterogêneos, como diferentes sistemas operacionais e diferentes arquiteturas de processadores, possam visualizar os seus dados espaço-temporais. Os dados espaciais considerados pela arquitetura podem ser provenientes de **coordenadas geográficas** e de **grupos espaciais**.

A representação em camadas da arquitetura do sistema pode ser vista na Figura 4.1. O sistema possui três camadas principais: **camada de informações** (camada com os dados), **camada do cliente** e **camada do servidor**.

A **camada de informação** deverá conter o banco de dados que é acessado unicamente pelo servidor. A arquitetura define a **camada do servidor** como sendo aquela responsável por todo processamento de dados, pois, desta forma, sistemas implementados a partir de suas especificações não sobrecarregarão a máquina dos usuários finais, possibilitando que tais sistemas requisitem menos recursos computacionais da máquina do cliente. Além disso, a camada do servidor é encarregada de realizar consultas no banco, armazenar as dashboards e gerar as páginas do sistema que serão exibidas pela camada do cliente.

A arquitetura estabelece que as operações básicas utilizadas em bancos de dados relacionais comumente referidas pelo acrônimo CRUD (Create, Read, Update e Delete) deverão ser contempladas na camada do cliente. Essas operações, que podem ser vistas através do esquema na Figura 4.2, deverão ser realizadas através de uma interface gráfica em que o usuário poderá importar dados tabulares e mapas em arquivos de geometria (como Geojson), deletar, visualizar e editar informações de mapas e dados importados por ele e criar dashboards com os mapas e conjuntos de dados que selecionar (ou editar, visualizar e deletar estas mesmas dashboards). A subseção 4.2 analisa as operações estipuladas na

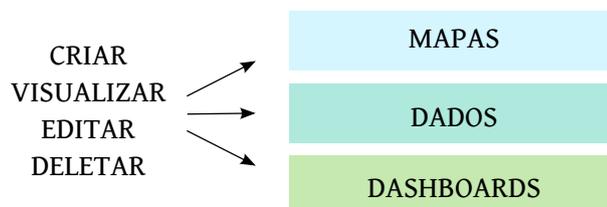
Figura 4.1 – Arquitetura CrabVis.



Fonte: Elaborada pelo autor.

arquitetura e determina um workflow para elas.

Figura 4.2 – Ações na Camada do cliente.



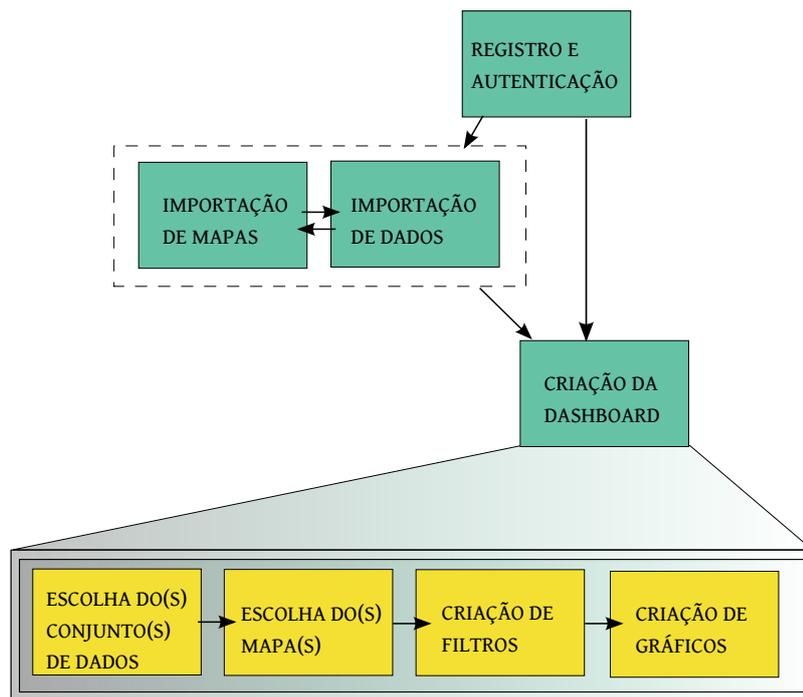
Fonte: Elaborada pelo autor.

4.2 Workflow operacional da arquitetura

A Figura 4.3 representa um modelo de workflow das operações contempladas na arquitetura proposta. Os retângulos destacados em verde identificam os quatro momentos do workflow pelos quais um novo usuário passa e cada um representa um módulo diferente da arquitetura. Os retângulos destacados em amarelo são uma expansão do módulo de criação de dashboards.

Após o registro e autenticação, as etapas de importação de mapas e de importação de dados podem ocorrer em qualquer ordem (vide Figura 4.3). Um usuário poderá criar

Figura 4.3 – Workflow do uso do sistema.



Fonte: Elaborada pelo autor.

uma nova dashboard mesmo se não tiver dados e mapas importados, caso utilize dados públicos disponíveis no sistema.

Inicialmente, um novo usuário deverá criar o seu cadastro e realizar o login (módulo de registro e autenticação). Após autenticado, poderá gerenciar os seus mapas, os seus conjuntos de dados e as suas dashboards, bem como criar, editar e excluir mapas, conjuntos de dados e dashboards.

No módulo de importação de dados, o usuário poderá importar um arquivo em formato tabular. Este formato foi escolhido por ser o formato mais utilizado nos portais de dados abertos. No arquivo tabular, cada linha do arquivo representa uma entrada dos dados do usuário e os diversos campos da tabela do usuário são separados por um mesmo caractere (mais comumente, um caractere de tabulação ou uma vírgula) e organizados em colunas.

Ao importar o arquivo tabular, um banco de dados é populado com os dados do arquivo fornecido. A inserção dos dados no banco é feita coluna a coluna. Durante a etapa de importação de dados, o usuário deverá informar quais colunas deseja importar e quais os tipos destas colunas.

Para acelerar e facilitar o processo de importação de dados na camada do cliente, uma estimativa inicial dos tipos de cada coluna deverá ser apresentada para o usuário. Essa estimativa inicial poderá ser feita, por exemplo, através de uma heurística de inferência de tipo executada na **camada do servidor**. Além disso, o usuário deverá ser capaz de corrigir as estimativas incorretas antes de finalizar o processo de importação de dados. A

arquitetura admite cinco possíveis tipos: **Número**, **Categoria**, **Tempo**, **Coordenada** e **Grupo Espacial**.

Além de ser necessária a definição do tipo de cada coluna do conjunto de dados, se um conjunto de dados possuir alguma coluna com dados de Coordenada, é necessário calcular uma *bounding box* dos dados. Esta *bounding box* deverá ser calculada a partir das colunas com Coordenadas presentes no conjunto de dados importado. *Bounding box* pode ser definida como o retângulo de área mínima que contém um determinado número de pontos. Esta informação de *bounding box* do conjunto de dados será utilizada posteriormente no processo de criação de dashboards (seção 5.5).

Cada mapa importado será associado a um nível de granularidade espacial máxima e mínima. Na importação de mapas, o usuário poderá importar um arquivo que contenha informações de geometria tais como Shapefile (Esri, 2016) e Geojson. O Geojson, por exemplo, é um formato popular de dados que codifica estruturas geográficas em polígonos e guarda propriedades para cada polígono componente, que podem representar, por exemplo, informações como nome, população e área de cada polígono.

A criação de dashboards se dá através da associação da dashboard criada com mapas e colunas de dados existentes. A seção apresenta com maior detalhe de que forma a camada de informação dá suporte às funcionalidades retratadas na Figura 4.3.

4.3 Estrutura dos dados

Nesta seção, a estrutura dos dados da arquitetura proposta é detalhada. Um esquema geral é apresentado na Figura 4.4 e, nas subseções seguintes, cada parte componente do esquema é explorada.

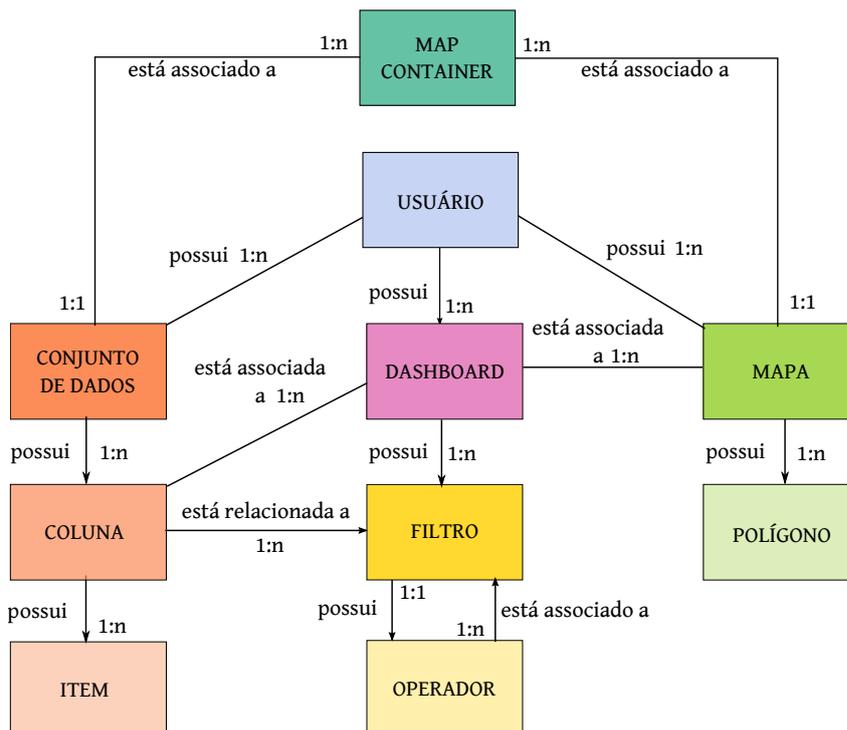
O tratamento da granularidade dos dados é um dos principais focos da arquitetura CrabVis e impacta diretamente a forma como os dados são organizados na camada de informação. As granularidades consideradas nesta especificação são a granularidade temporal e a espacial. A granularidade temporal pode ser inferida com precisão a partir de marcas temporais (*timestamps*), pois, partindo da marca temporal é possível identificar qual a granularidade mínima ou mais grossa no tempo do dado importado.

Diferentemente da granularidade temporal, a granularidade espacial não pode ser inferida com facilidade. Na importação de um novo conjunto de dados, se este possuir coordenadas, a granularidade máxima assume o nível de ponto. Caso estes dados não possuam coordenadas, mas possuam grupos espaciais, o valor da granularidade espacial do conjunto de dados assume a granularidade do grupo espacial cuja granularidade é máxima.

Na lista abaixo estão representadas as diferentes granularidades espaciais que a arquitetura CrabVis abrange:

- Ponto
- Bairro

Figura 4.4 – Esquema da arquitetura CrabVis.



Fonte: Elaborada pelo autor.

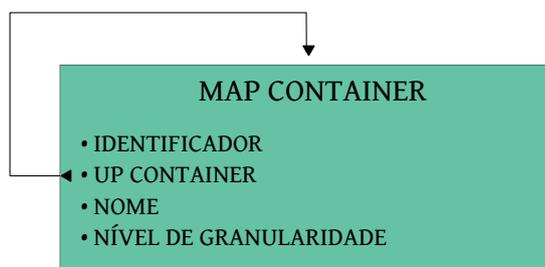
- Conjunto de Bairros
- Cidade
- Conjunto de Cidades
- Estado
- Conjunto de Estados
- País
- Conjunto de Países
- Continente
- Mundo

Esta divisão de granularidades espaciais é uma divisão hierárquica de aspecto essencialmente político, porém, outros tipos de divisão são admitidos, desde que contenham um aspecto hierárquico. A representação de dados e mapas em níveis intermediários tais como conjunto de bairros, de cidades, de estados e de países oferece um maior grau de flexibilidade na etapa de exploração de dados que ainda não está presente em sistemas de Visualização Web tais como Plotly e Quadrigram, pois não restringe a análise de dados e mapas a granularidades pré-definidas e permite a travessia entre diferentes granularidades espaciais.

4.3.1 Map Container

A fim de se obter melhor proveito de todas as informações sobre a granularidade dos dados e dos mapas importados, foi criado, como parte do módulo de importação de mapas, um componente chamado *Map Container*.

Figura 4.5 – Componente Map Container.



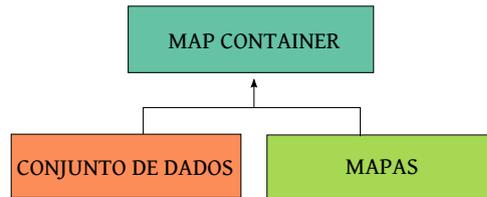
Fonte: Elaborada pelo autor.

A Figura 4.5 contém um esquema básico do componente Map Container. Cada instância de Map Container é referente a uma região única com um nível de granularidade espacial próprio (por isso, um item de Map Container possui um identificador e um número inteiro que indica em que nível de granularidade aquele item está). Uma das necessidades de ter o nível de granularidade nesse componente é diferenciar casos em que existe, por exemplo, uma cidade com o mesmo nome que um estado (por exemplo, São Paulo). A fim de construir uma hierarquia de regiões posteriormente utilizada na camada do servidor, o Map Container também possui um valor Up que referencia um outro componente Map Container cuja granularidade espacial é menor (menos detalhe) do que a dele.

No momento em que o usuário informa que o conjunto de dados ou o mapa estão em uma determinada granularidade e fornece também o nome da região correspondente, ele cria uma nova entrada em Map Container e associa aqueles conjuntos de dados ou mapas àquela entrada (Figura 4.6). O Map Container criado será associado ao mapa importado em questão e poderá também ser associado a outros mapas importados futuramente. Isso facilita que vários mapas e dados sejam associados a um mesmo Map Container, como pode ser observado na Figura 4.7.

A vantagem de unir dados e mapas em um mesmo Map Container é incorporar na arquitetura a possibilidade de realizar sugestões ao usuário. Um exemplo disso é quando o usuário cria uma nova dashboard. Ao selecionar os dados que serão visualizados, como esses dados foram associados a um Map Container, é possível realizar uma busca no banco de dados da camada de informação por mapas públicos associados ao mesmo Map Container. Dessa forma, é possível sugerir mapas que podem ser de interesse do usuário. Então, mesmo que o usuário possua apenas dados e não possua mapas da sua região de interesse, ele poderá, ainda assim criar uma dashboard.

Figura 4.6 – Map Container.



Fonte: Elaborada pelo autor.

Figura 4.7 – Map Container do Brasil.



Fonte: Elaborada pelo autor.

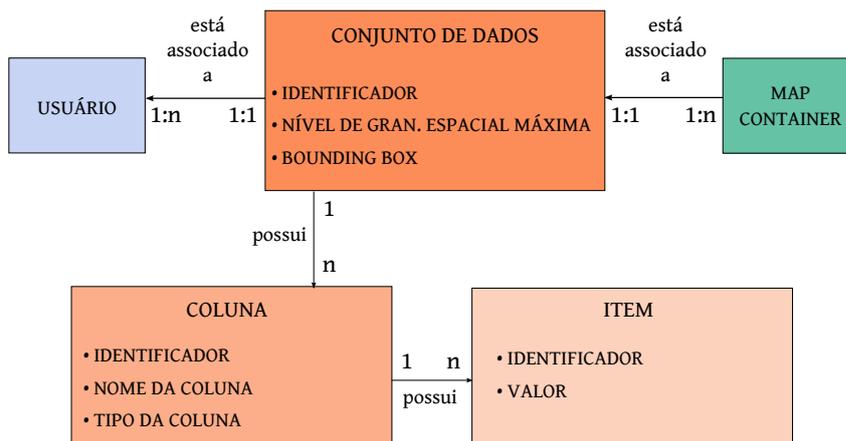
Por exemplo: o usuário deseja visualizar alguns dados que ele possui da cidade de Fortaleza e ele possui apenas o mapa de Fortaleza dividida em bairros. Na etapa de criação de dashboards, ele verifica que um mapa da cidade de Fortaleza dividido em regionais (que são conjuntos de bairros) também está disponível para uso. Com o novo mapa, ou seja, com o uso do mapa de Fortaleza dividido em regionais, ele poderá ter uma nova perspectiva a respeito dos dados que possui.

4.3.2 Conjuntos de dados

A Figura 4.8 apresenta um esquema abstrato de como os conjuntos de dados do usuário são guardados na camada de informação. Cada conjunto de dados possui um identificador único e está ligado a um usuário e um Map Container. Além disso, cada conjunto de dados importado possui um nível de granularidade espacial máxima e, caso possua dados de coordenada, possui uma *bounding box*.

A *bounding box* possibilita que, no momento da criação da dashboard, a camada do

Figura 4.8 – Esquema do Conjunto de Dados na Camada de Informação.



Fonte: Elaborada pelo autor.

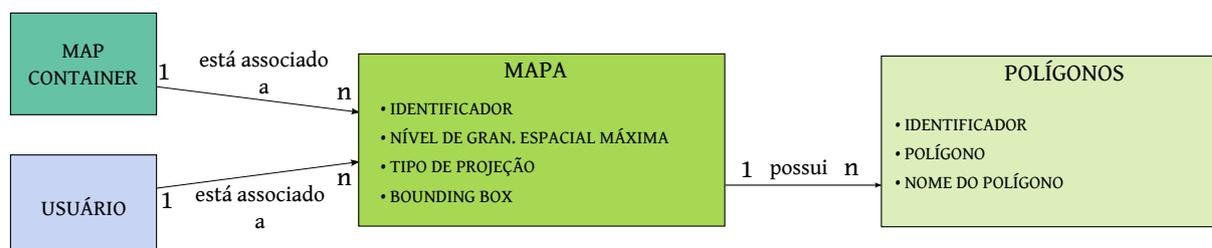
servidor faça uma consulta em todos os mapas do usuário e todos os mapas públicos, procurando por mapas em que aquele conjunto de dados está contido. Esta consulta é realizada para que seja possível sugerir mapas ao usuário de modo mais preciso do que se fosse utilizada uma consulta simples baseada apenas por nomes.

Cada conjunto de dados possui várias colunas cujos tipos podem ser classificados em Coordenada, Número, Grupo espacial, Categoria e Tempo. O usuário informa o tipo da coluna no momento da importação dos dados. Cada coluna pode ser identificada unicamente e possui também um nome.

Além das colunas, o conjunto de dados possui itens. Cada item de dado possui seu identificador único e também possui um determinado valor, dependendo do tipo de coluna que está associado àquele item. Caso a coluna referente seja do tipo número, o valor será numérico, por exemplo.

4.3.3 Mapas

Figura 4.9 – Esquema dos Mapas na Camada de Informação.



Fonte: Elaborada pelo autor.

A importação de mapas, ou seja, de arquivos de geometria divididos em polígonos tais como Geojson ou Shapefile gera dados de mapas e de polígonos de mapas. Um

esquema de como a camada de informação guarda os mapas do cliente pode ser visto na Figura 4.9. Cada mapa possui um identificador único, e está associado a um usuário e a um Map Container. Um nível de granularidade espacial máxima está associado a cada mapa. A arquitetura CrabVis parte da suposição que todas divisões de um mapa estão em um mesmo nível de granularidade, pois é normalmente assim que arquivos de geometria são organizados. O nível de granularidade máxima de um mapa é igual ao nível de granularidade de seus polígonos componentes.

Figura 4.10 – Exemplo de criação de Map Containers durante a Importação de Mapas.



Fonte: Elaborada pelo autor.

A arquitetura também considera as diversas projeções de mapas existentes. Dependendo da projeção em que um mapa é especificado, ele deverá ser tratado de forma

diferente pela camada do servidor e na camada do cliente, pois a projeção influencia na visualização do mapa.

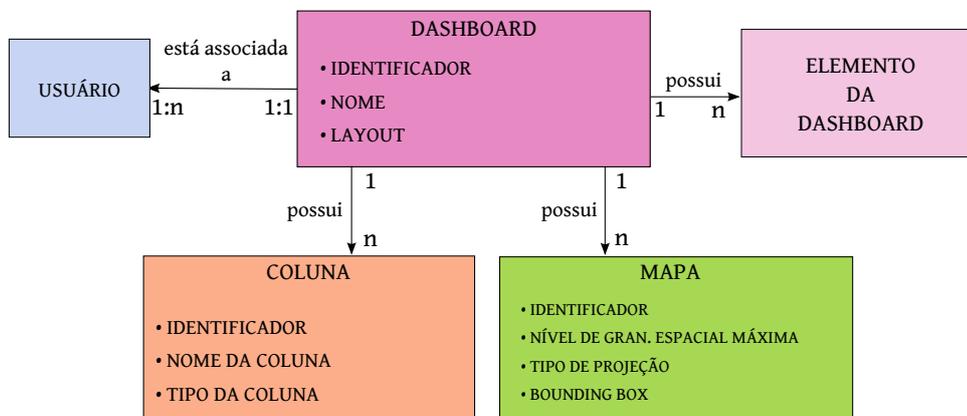
Arquivos de geometria (como Shapefiles e GeoJson) permitem, resumidamente, estruturar uma região geográfica em polígonos cujos vértices são coordenadas geográficas e que cada polígono possui um nome diferente. Isto é, cada polígono corresponde a uma unidade geográfica, seja esta unidade uma cidade, estado ou região. Similarmente, cada instância de mapa é associada a vários polígonos componentes daquele mapa.

Durante a importação do mapa, como o usuário informou a granularidade máxima e mínima do mapa importado, cada polígono do mapa será inserido no banco de dados como um novo Map Container cuja granularidade é a granularidade máxima do mapa importado e cujo nome é o nome do polígono no mapa. O Up Container de cada um desses Map Containers apontará para o Map Container associado ao mapa. Ou seja, importar um mapa da Região Nordeste dividida em Estados significa criar um Map Container da Região Nordeste e criar outros nove Map Containers adicionais, um para cada estado do Nordeste. O Up Container de cada um destes nove Map Containers é o identificador do Map Container da Região Nordeste (o exemplo pode ser visto na Figura 4.10). A forma como os dados e mapas foram organizados impacta diretamente na criação de uma nova dashboard, descrita na subseção 4.3.4.

4.3.4 Dashboard

A dashboard é composta de um identificador único e um nome. Associada a ela, estão, um conjunto de colunas e mapas selecionados para aquela dashboard (Figura 4.11). Na etapa de criação de uma nova dashboard, quando o usuário escolhe um conjunto de dados, a camada do servidor realiza uma consulta para determinar se existe algum mapa do usuário ou algum mapa público que contenha a *bounding box* daquele conjunto de dados. Caso exista, o mapa é sugerido ao usuário. Dessa forma o usuário estará associando os mapas e dados que ele escolheu àquela dashboard.

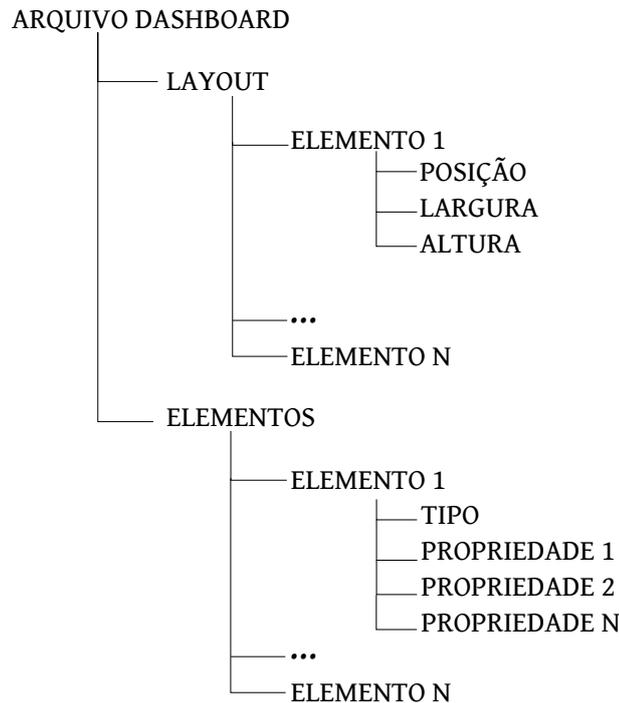
Figura 4.11 – Esquema das Dashboards na Camada de Informação.



Fonte: Elaborada pelo autor.

As informações a respeito de cada visualização da dashboard são guardadas num arquivo estruturado que representa cada dashboard. O arquivo de dashboard guarda informações referentes à posição dos elementos (mapas e textos) na dashboard, além de guardar informações próprias de cada tipo de gráfico contido naquela dashboard. A Figura 4.12 mostra a estrutura abstrata de um arquivo de dashboard. Estas informações são utilizadas pela camada do cliente para reconstruir os gráficos, sempre que a dashboard é carregada. São informações como qual coluna de coordenadas um mapa de calor presente na dashboard utiliza ou quais são as cores dos pontos atribuídas a cada coluna de coordenadas presente em um mapa de pontos.

Figura 4.12 – Esquema do arquivo estruturado da dashboard.



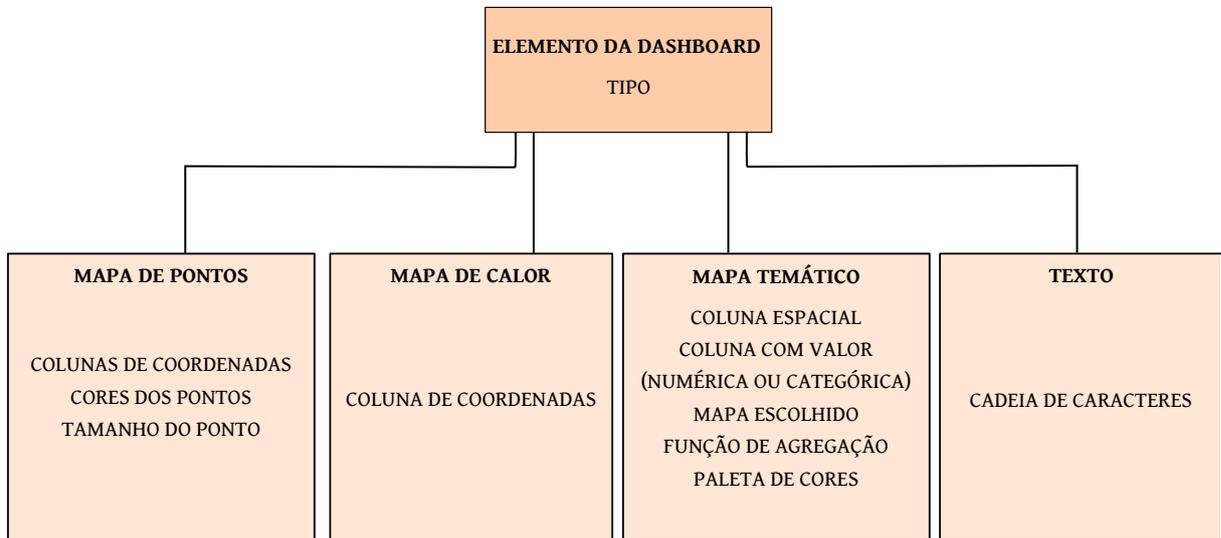
Fonte: Elaborada pelo autor.

Para a inserção de um novo tipo de gráfico na camada do cliente, um desenvolvedor deve estabelecer uma estrutura que será adicionada ao arquivo estruturado quando aquele novo gráfico for salvo na dashboard. Ou seja, ele deverá dispor de uma forma estruturada todas as propriedades (parâmetros) que o algoritmo para criação do gráfico necessita para construir o gráfico do zero. São propriedades como qual o tipo de gráfico, quais colunas deverão ser utilizadas e qual o mapeamento (para definir cor ou forma) que deverá ser aplicado nos elementos de cada coluna (se houver mapeamento). Estas propriedades podem variar de acordo com a particularidade de cada gráfico.

As propriedades também podem ser, por sua vez, hierárquicas. Um exemplo hipotético disso é quando o usuário possui uma propriedade chamada “mapeamento” que armazena o mapeamento de todas as colunas utilizadas no gráfico. Caso o usuário realize este mapeamento utilizando um par chave-valor para cada coluna, neste caso, a propriedade

“mapeamento” guardaria vários pares de entradas chave-valor. Portanto, a propriedade possuiria uma estrutura hierárquica. Um exemplo de como elementos de uma dashboard seriam armazenados em um arquivo pode ser visto na Figura 4.13.

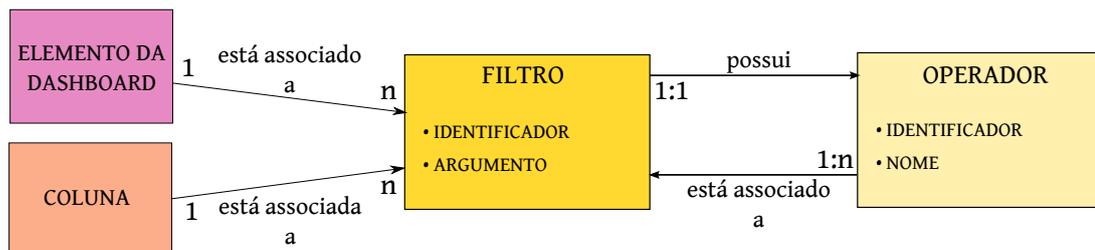
Figura 4.13 – Elementos da Dashboard no arquivo estruturado.



Fonte: Elaborada pelo autor.

4.4 Filtros

Figura 4.14 – Esquema dos Filtros na Camada de Informação.



Fonte: Elaborada pelo autor.

A arquitetura CrabVis possibilita, durante a criação ou edição de uma dashboard, que os dados associados à dashboard sejam filtrados segundo alguma coluna (ou mais de uma coluna). Esses filtros possuem um esquema exemplificado na Figura 4.14. Cada filtro é associado a uma coluna e um operador. Os operadores possíveis são $<$, \leq , $>$, \geq , $=$ e \neq . A operação será feita referenciando uma coluna de uma dashboard, utilizando um operador especificado e um argumento. O argumento pode ser numérico, textual ou temporal.

Um exemplo de filtro seria: usuário deseja obter apenas, na dashboard D, as entradas em que o valor da coluna C é diferente de X. O filtro criado referenciará o identificador da

coluna C, associará o identificador do operador “ \neq ” àquele filtro e instanciará o campo argumento (no filtro) como “X”.

Neste capítulo, uma arquitetura de sistema de Visualização Web com foco na granularidade espacial foi apresentada. Inicialmente, uma visão geral da arquitetura foi mostrada, para ser, então, detalhada em termos de funcionalidades e de workflow entre as operações definidas pela arquitetura. Além disso, a estrutura dos dados utilizada pela arquitetura foi detalhada, e esta é composta por conjuntos de dados, Map Containers, mapas, dashboards e filtros.

5 Implementação e Casos de Uso

Este capítulo apresenta um protótipo de sistema de Visualização Web, que foi implementado a partir das especificações da arquitetura CrabVis, proposta no capítulo 4. Algumas informações gerais a respeito de tecnologias utilizadas são inicialmente apresentadas, e, na sequência, cada módulo do sistema é detalhado. Por fim, alguns casos de uso construídos utilizando o sistema são apresentados.

5.1 Informações gerais

Figura 5.1 – Homepage do CrabVis.



Fonte: Elaborada pelo autor.

O protótipo do sistema (Figura 5.1) foi implementado na linguagem PHP utilizando o framework MVC (model-view-controller) Laravel (Taylor Otwell, 2016). O padrão MVC é um padrão de arquitetura de software que separa o modelo (Model) da visão (View), onde o modelo é o banco de dados e a visão é o conjunto de páginas Web. A comunicação entre essas duas camadas se dá através do controlador (Controller). O controlador é o intermédio da visão com o modelo, e repassa dados presentes no banco quando estes são

requisitados pelas páginas Web. Isto é possível porque o controlador se comunica com o banco diretamente realizando consultas, inserindo, atualizando e deletando dados.

O banco de dados presente na camada de informação foi implementado no PostgreSQL com a extensão PostGIS. O Postgis é uma extensão do PostgreSQL cujo objetivo é otimizar e facilitar a manipulação de dados espaciais como polígonos e pontos. A camada do cliente foi implementada utilizando Javascript, JQuery, Leaflet, Node.js, WebGL, D3, TinyMCE e Ajax.

Atualmente, a camada do servidor e de informação encontram-se na mesma máquina, porém isso é opcional porque o framework Laravel pode comunicar-se com um banco que está em outra máquina. Portanto, as camadas do servidor e de informação podem ficar em duas máquinas diferentes. Como o foco do sistema é a Web, para a utilização do sistema, ou seja, interação com a camada do cliente, o usuário deverá apenas possuir um navegador de internet.

5.2 Registro e Autenticação

Figura 5.2 – Tela de registro (acima) e tela de login (abaixo).

The image displays two screenshots of the CrabVis web application. The top screenshot shows the registration page, which features a white form with four input fields: 'Nome', 'E-mail', 'Senha', and 'Confirma Senha'. A green 'CADASTRAR' button is positioned below the form. The bottom screenshot shows the login page, which features a white form with two input fields: 'E-mail' and 'Senha'. Below the 'Senha' field is a checkbox labeled 'Lembrar de mim'. A green 'LOGIN' button is positioned below the form. Both screenshots have a dark blue header with the 'CrabVis' logo and 'Login' and 'Cadastro' links.

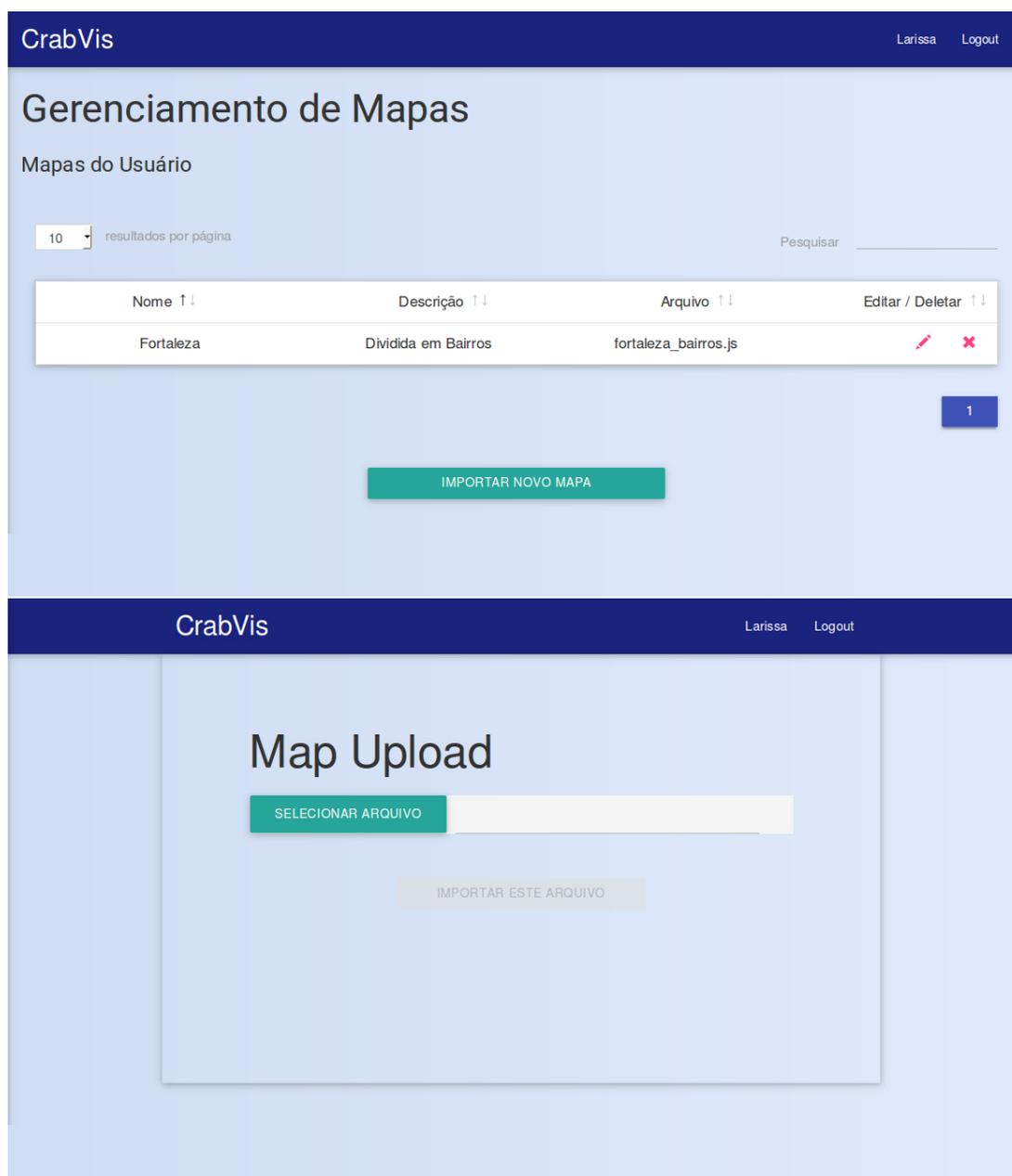
Fonte: Elaborada pelo autor.

O módulo de registro e autenticação (Figura 5.2) é composto pelo cadastro do usuário no sistema e o seu subsequente login. O usuário só poderá interagir com o sistema se estiver logado (autenticado). Para utilização do sistema, o usuário deverá realizar um cadastro em que deverá fornecer apenas o seu nome de usuário, senha e e-mail. Essas informações serão guardadas no banco, onde é atribuído, a cada usuário, um identificador que associa aquele usuário aos mapas e dados que importou e às dashboards que criou. A senha do usuário é criptografada antes de ser salva no sistema. O login do sistema deverá ser realizado com o e-mail e senha fornecidos na etapa de registro.

5.3 Importação de Mapas

O módulo de importação de mapas é constituído por um wizard em que o usuário fornece informações sobre o arquivo Geojson que está importando. Uma vez que o usuário clica na opção de “Importar novo mapa”, na tela de Gerenciamento de Mapas, ele será levado para uma tela onde poderá realizar o upload de um arquivo Geojson para o sistema (Figura 5.3). Ao enviar este arquivo, o usuário é apresentado ao wizard propriamente dito.

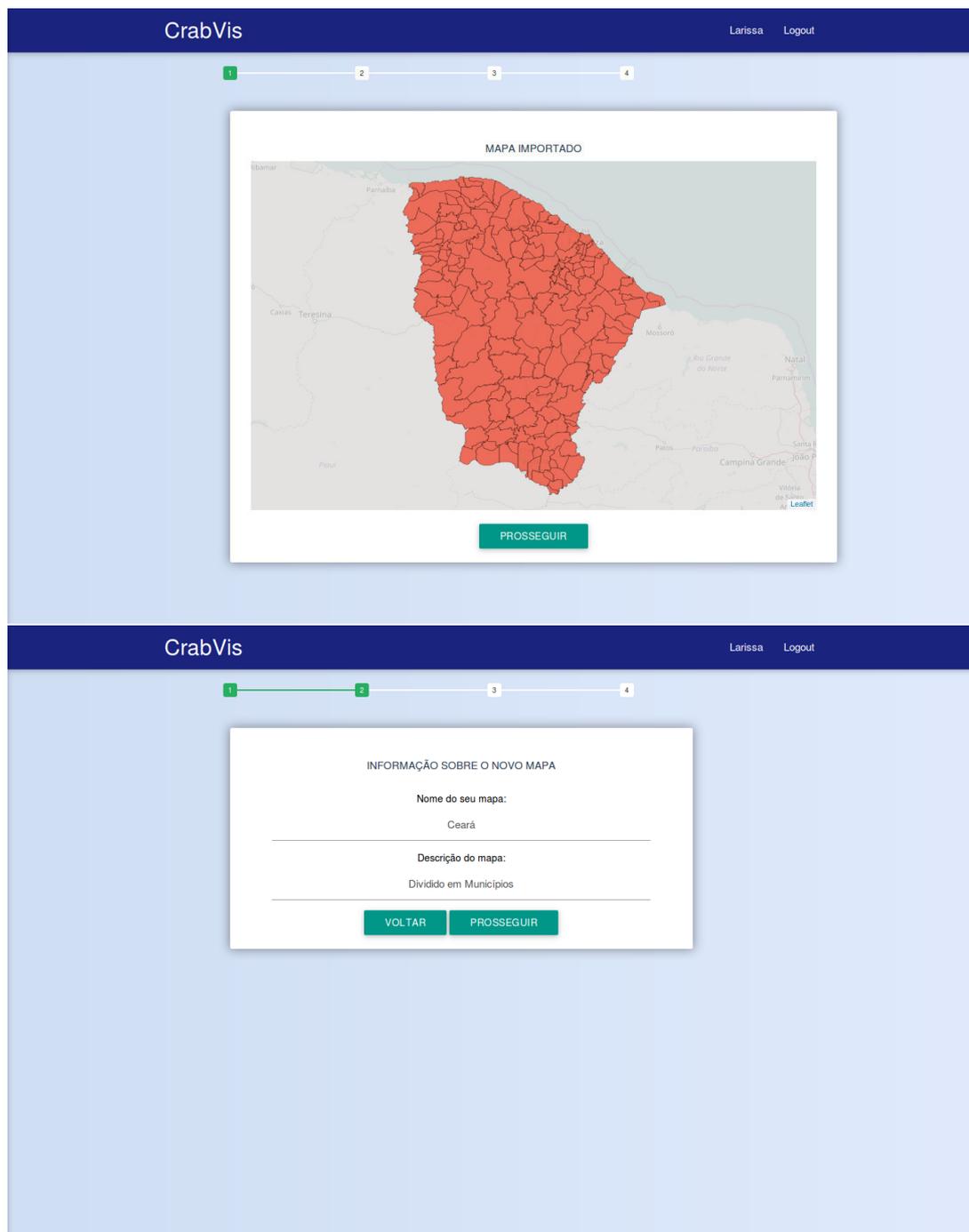
Figura 5.3 – Tela de gerenciamento de Mapas (acima) e tela de Importação de Mapas (abaixo).



Fonte: Elaborada pelo autor.

O wizard de importação de mapas possui quatro etapas. Na primeira etapa, o usuário vê uma imagem do mapa importado com todos os polígonos componentes nele contidos.

Figura 5.4 – Processo de importação de Mapas: Etapa 1 e Etapa 2.

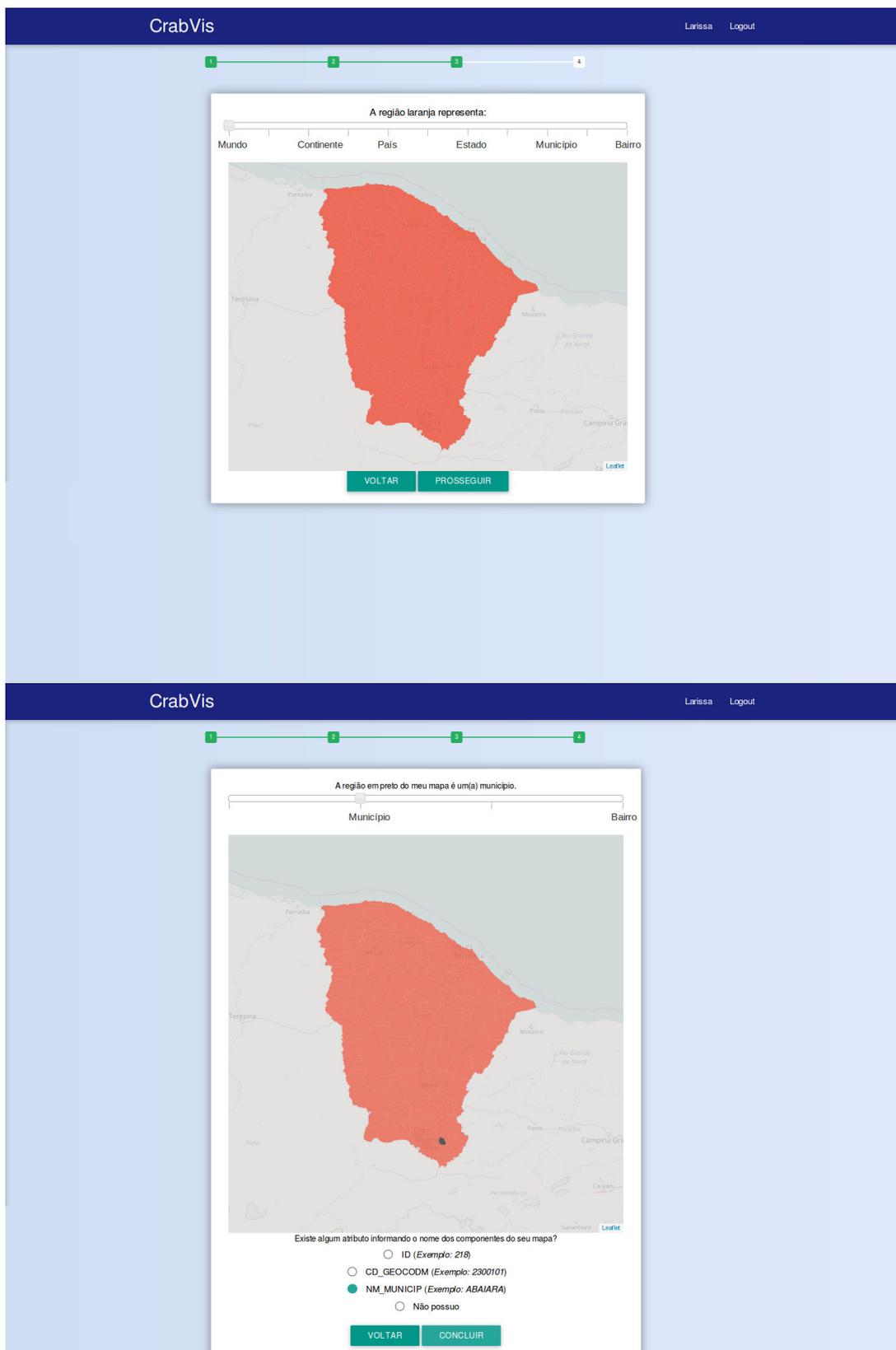


Fonte: Elaborada pelo autor.

Ao prosseguir no wizard, o usuário deverá fornecer um nome para o mapa importado e uma breve descrição (segunda etapa). A primeira e a segunda etapa estão ilustradas na Figura 5.4. Estas informações serão utilizadas posteriormente para que o próprio usuário identifique aquele mapa. Para o sistema, o identificador único do mapa sempre será o id numérico presente na tabela, mas o usuário necessita de uma informação mais alto nível para diferenciar aquele mapa importado dos demais.

A Figura 5.5 ilustra a terceira e a quarta etapa. Na terceira etapa do wizard, o usuário vê novamente o mapa que importou, porém sem divisões, de forma que o destaque é

Figura 5.5 – Processo de importação de Mapas: Etapa 3 e Etapa 4.



Fonte: Elaborada pelo autor.

voltado para a área do mapa como um todo. Além da imagem do mapa, o sistema questiona a granularidade mínima daquele mapa, ou seja, o que aquele mapa como um

todo representa. Esse questionamento é feito de forma que o usuário não precise saber o conceito de granularidade espacial.

Figura 5.6 – Sugestão baseada em um Map Container existente.

A região laranja representa:

Qual o nome desse(a) estado?

Ceará

Ceará > Brasil
Ceará > Brasil

Fonte: Elaborada pelo autor.

Como o usuário já informou o nível de granularidade mínima, ele deverá agora informar qual o nome daquela região para que possa ser criado um Map Container para ela. Conforme o usuário digita o nome da região, o sistema realiza uma busca no banco de dados por Map Containers que possuem a mesma granularidade e que possuem no campo nome o mesmo valor textual que o usuário está digitando.

O sistema apresenta então, em tempo real, três sugestões que o usuário pode aceitar ou não. O número três é um limitante do número de resultados que a busca no banco de dados retorna quando procura por Map Containers de mesma granularidade. Caso não houvesse um limitante, o banco de dados retornaria possivelmente muitos resultados para aquela consulta e isto poderia poluir a interface visual do sistema.

Se o usuário aceitar a sugestão do sistema, aquele mapa será vinculado ao Map Container sugerido. Caso o usuário não aceite a sugestão, um novo Map Container será criado. No exemplo, o usuário está importando um mapa do Ceará. Como já existia um Map Container com o nome “Ceará” e a mesma granularidade (Estado), o CrabVis sugeriu o Map Container do Ceará (Figura 5.6).

Por exemplo: se o usuário estiver importando um mapa da Paraíba dividido em municípios, a granularidade mínima será Estado. O usuário interagirá com um *slider* presente naquela tela e informará que a região destacada representa um estado. Essa interação é realizada através de um *slider* cujas opções são as granularidades espaciais possíveis. O *slider* foi escolhido por ser um elemento familiar para quem utiliza a Internet, pois sua presença é comum em formulários Web. Além disso, o *slider* permite que todas as opções possíveis de escolha fiquem visíveis ao mesmo tempo, sem que isso pareça intimidador para o usuário do sistema.

A busca pelo Map Container que será atribuído àquele mapa foi implementada no sistema desta forma, porém, a arquitetura proposta não limita que a identificação do Map Container atribuído àquele mapa seja realizada assim. Uma forma alternativa de

implementar esta etapa seria realizar uma comparação entre a *bounding box* do mapa que está sendo importado e a *bounding box* de outros mapas presentes no sistema (que tiverem a mesma granularidade mínima do mapa importado). Se o Map Container do mapa importado estiver contido na *bounding box* de um mapa presente no sistema, sugerir este Map Container ao usuário.

Ao prosseguir, o usuário entra na última etapa da importação de mapas, que é informar a granularidade máxima do mapa. O usuário vê o mapa importado novamente, porém, nesta etapa, um dos polígonos é destacado em preto. O usuário é questionado através de um *slider* similar ao anterior o que representa aquela região.

Figura 5.7 – Lista com propriedades dos polígonos presentes no mapa.

Existe algum atributo informando o nome dos componentes do seu mapa?

- ID (Exemplo: 218)
- CD_GEOCODM (Exemplo: 2300101)
- NM_MUNICIP (Exemplo: ABAIARA)
- Não possui

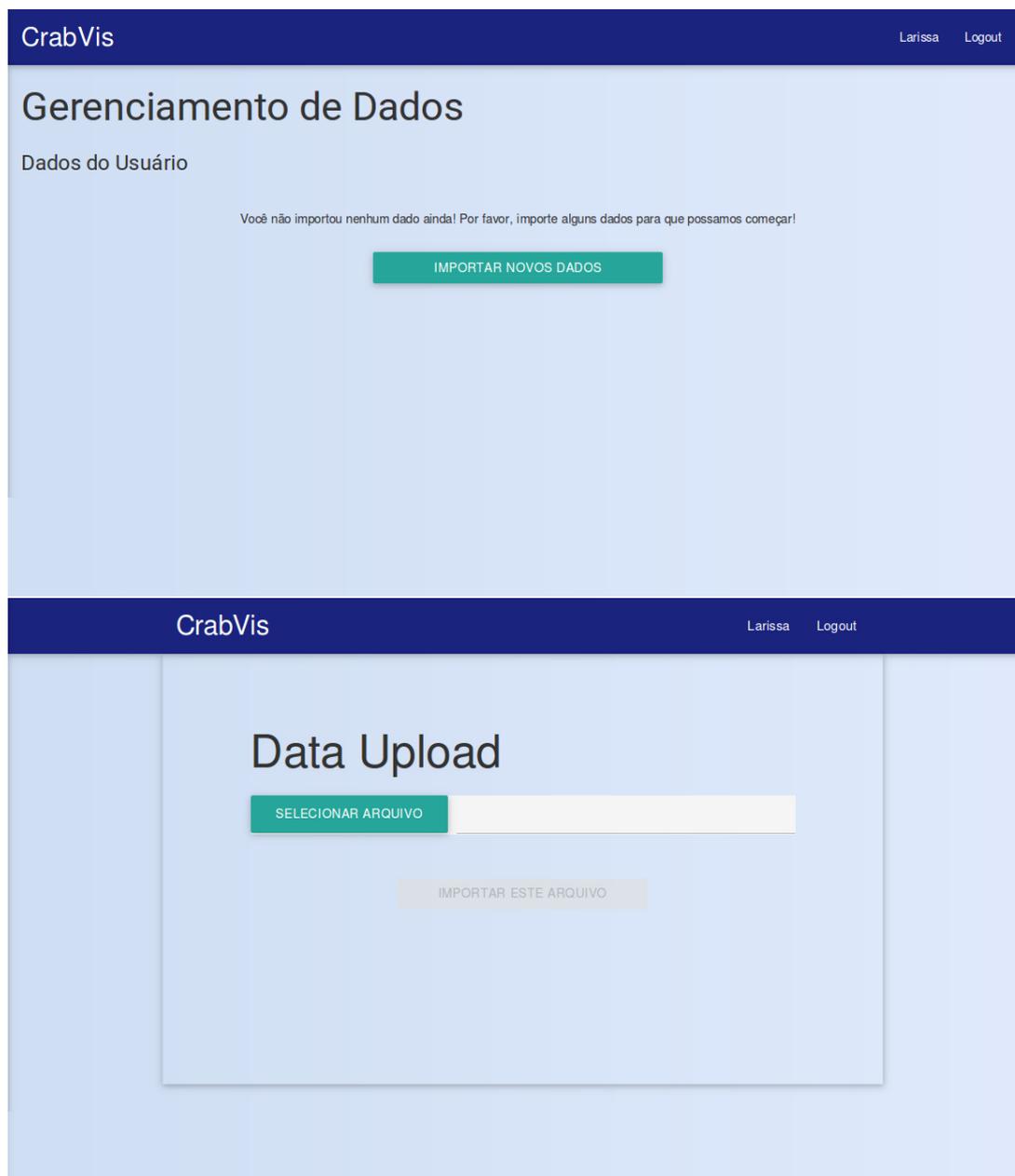
Fonte: Elaborada pelo autor.

Comumente, os arquivos de geometria possuem, associadas a cada polígono, diversas propriedades. Alguns possíveis exemplos de propriedades são id, área, população e nome do polígono. A propriedade mais frequentemente especificada e mais útil que um arquivo de geometria pode fornecer é o nome de cada polígono. Por isso, nesta etapa, o sistema lista todas as propriedades que aquele arquivo possui e questiona, através de um input do tipo radio, se alguma daquelas propriedades é referente ao nome dos polígonos e, caso exista, qual delas armazena o nome dos polígonos (Figura 5.7). Ao lado de cada propriedade está um exemplo, que também foi obtido a partir do arquivo do usuário, de forma que o usuário não precisa ter memorizado o conteúdo de cada propriedade do arquivo importado. Caso o arquivo não possua tal propriedade, o usuário pode marcar que o arquivo não possui uma propriedade informando o nome dos polígonos.

Ao fim do processo de importação, a camada do cliente enviará as informações preenchidas na importação de mapas para o controlador, que fará o processamento daqueles dados e os enviará para o banco. A importação dos mapas criará uma entrada na tabela de mapas e várias entradas na tabela de polígonos de mapa. Neste momento, o controlador calculará a *bounding box* do mapa e salvará essa informação na entrada da tabela de mapas criada para aquele arquivo. Além dessas informações, um Map Container será criado para aquele mapa e para cada polígono importado.

5.4 Importação de Dados

Figura 5.8 – Tela de gerenciamento de Dados (acima) e tela de Importação de Dados (abaixo).



Fonte: Elaborada pelo autor.

Para importar os seus dados, o usuário deverá ir até a tela de Gerenciamento de dados e clicar em Importar novos dados (Figura 5.8). Ele será levado, então, para uma tela onde deverá escolher o conjunto de dados que deseja importar. O arquivo a ser importado deverá ter formato tabular. Atualmente, o sistema aceita arquivos csv (comma-separated-values), que é o formato mais simples de planilhas. Este arquivo será transferido para o sistema e o processo de importação começará quando o usuário clicar em “Importar este arquivo”.

A importação de dados também acontece em etapas. São cinco etapas. Na primeira etapa, o sistema obtém informações gerais sobre aquele conjunto de dados como nome e descrição e a confirmação de que o sistema compreendeu o nome das colunas do arquivo corretamente.

Na tela seguinte (segunda etapa), o usuário deverá escolher quais colunas dos dados deseja importar. O sistema apresenta uma tabela com as três primeiras linhas dos dados dele em que a última linha da tabela é composta de *checkboxes*. O usuário deverá checar apenas as *checkboxes* correspondentes às colunas que deseja importar. A primeira e segunda etapas estão ilustradas na Figura 5.9.

Figura 5.9 – Processo de importação de Dados: Etapa 1 e Etapa 2.

CrabVis Larissa Logout

1 2 3 4 5

INFORMAÇÃO SOBRE O NOVO CONJUNTO DE DADOS

Nome dos seus dados:
Dengue no Ceará

Descrição dos seus dados:
Dados de 2013

Estes são os nomes das colunas do seu arquivo?

DT_NOTIFIC, lat, lng, city

Não Sim

PROSSEGUIR

CrabVis Larissa Logout

1 2 3 4 5

VARIÁVEIS A IMPORTAR

Quais variáveis você deseja importar?

DT_NOTIFIC	lat	lng	city
7/9/2013	-3.7481034	-39.7881889	Irauçuba
7/10/2013	-4.0256737	-38.8699461	Maranguape
6/18/2013	-3.7318616	-38.5266704	Fortaleza

✓ ✓ ✓ ✓

✓ Selecionar tudo

VOLTAR PROSSEGUIR

Fonte: Elaborada pelo autor.

Ao prosseguir, o usuário será levado para uma tela de identificação de tipo de dado (terceira etapa). Para promover a interatividade e intuitividade na importação de dados, foi implementada uma heurística simples, mas eficaz, de detecção automática de tipo do campo (Algoritmo 1). A adição desta heurística no sistema causa uma maior fluidez na etapa de importação dos dados, pois reduz o número de decisões e ações que

Figura 5.10 – Processo de importação de Dados: Etapa 3 e Etapa 4.

The image displays two screenshots of the CrabVis web application interface, illustrating the data import process in two stages.

Top Screenshot (Etapa 3): TIPOS DAS VARIÁVEIS

This screen shows the mapping of data fields to variable types. On the left, under "CAMPOS", there are four fields: "DT_NOTIFIC : 7/9/2013" (green), "lat : -3.7481034" (blue), "lng : -39.7881889" (blue), and "city : Irauçuba" (red). On the right, under "TIPOS", there are five options: "Timestamp" (green), "Grupo Espacial" (yellow), "Número" (purple), "Categoria" (red), and "Coordenada" (blue). A central instruction box says: "Arraste cada tipo na direita até o campo correspondente na esquerda!". At the bottom are "VOLTAR" and "PROSSEGUIR" buttons.

Bottom Screenshot (Etapa 4): INFORMAÇÕES ESPACIAIS

This screen shows the mapping of spatial information. On the left, under "CAMPOS COM COORDENADAS", there are two fields: "lat : -3.7481034" (orange) and "lng : -39.7881889" (green). On the right, under "TIPOS", there are four options: "Latitude" (orange), "Longitude" (green), "Latitude / Longitude" (pink), and "Longitude / Latitude" (grey). A central instruction box says: "Arraste cada tipo na direita até o campo correspondente na esquerda!". Below this, a text box contains "city: Fortaleza, Maranguape, Irauçuba". At the bottom, a slider asks "Os dados acima representam o que?" with labels: "Mundo", "Continente", "País", "Estado", "Município", and "Bairro". At the bottom are "VOLTAR" and "PROSSEGUIR" buttons.

Fonte: Elaborada pelo autor.

o usuário deverá tomar durante essa etapa. Na ocorrência de um usuário que importou um arquivo com uma grande quantidade de colunas, ele não terá que classificar todas as colunas do zero. Deverá apenas, caso necessário, corrigir os campos que foram classificados erroneamente pela heurística. Dependendo do tipo declarado pelo usuário na etapa de importação de dados, cada informação presente no arquivo tabular é armazenada em uma tabela diferente.

Algoritmo 1 Detecção Automática de Tipo.

```

1: function DETECTA_TIPO(dados_importados)
2:   nomes_colunas ← lê_nomes_colunas(dados_importados)
3:   tipos_iniciais ← lê_tipos_colunas(dados_importados)
4:   total_entradas ← total_entradas(dados_importados)
5:   for i ← 1 to comprimento(tipos_iniciais) do
6:     if tipos_iniciais[i] = “real” then
7:       if detecta_expressao(“[Ll]ng|[Ll]ong”, nomes_colunas[i]) then
8:         tipo[i] ← “Longitude”
9:       else if detecta_expressao(“[Ll]at”, nomes_colunas[i]) then
10:        tipo[i] ← “Latitude”
11:      else
12:        tipo[i] ← “Numero”
13:      end if
14:    end if
15:    if tipos_iniciais[i] = “inteiro” then
16:      total_entradas_unicas = total_entradas_diferentes(dados_importados)
17:      if total_entradas_unicas/total_entradas < 0.5 then
18:        tipo[i] ← “Categoria”
19:      else
20:        tipo[i] ← “Numero”
21:      end if
22:    end if
23:    if tipos_iniciais[i] = “String” then
24:      datas = executar_parsing_datas(dados_importados[, i])
25:      total_parsing_falho = total_datas_nao_identificadas(datas)
26:      if total_parsing_falho/total_entradas ≤ 0.5 then
27:        tipo[i] ← “Data”
28:      else
29:        tipo[i] ← “Categoria”
30:      end if
31:    end if
32:    if tipos_iniciais[i] = “Booleano” then
33:      tipo[i] ← “Categoria”
34:    end if
35:  end for
36: end function

```

A heurística de detecção de tipo do campo foi implementada na linguagem R e consiste num script que é executado quando o usuário, na tela inicial de importação de dados, seleciona o arquivo com seus dados e o envia para o servidor. O ponto inicial da heurística é

realizar uma pré-deteção do tipo do campo utilizando uma função da linguagem R (função `lê_tipos_colunas` na linha 3 do Algoritmo 1). O resultado da chamada dessa função, ou seja, os tipos encontrados para cada coluna e que são os tipos da própria linguagem R, está armazenado na variável denominada “tipos_iniciais”.

Em seguida, o algoritmo verifica se aquela coluna no dado possui dados alfanuméricos ou puramente numéricos. Se o dado for puramente numérico e de ponto flutuante, ele pode representar um **Dado numérico** ou um componente de **Coordenada**. Caso o nome desta coluna obedeça a uma expressão regular que busca palavras que contém “lat” no início, a heurística presume que aquela coluna contém uma latitude. Caso o nome da coluna obedeça a uma expressão regular que busca palavras que contém “lng” ou “lon”, que são tipicamente associadas à longitude, a heurística presume que aquela coluna contém uma longitude. Se o nome da coluna não obedecer a uma destas duas expressões, a heurística presumirá que aquele dado corresponde a um campo numérico.

Se o dado for detectado inicialmente como inteiro, são verificadas quantos valores diferentes existe naquela coluna. O campo é classificado como **Categórico** se mais da metade dos valores se repetir. Caso contrário, o campo é dado como **Número**.

Se a pré-deteção informar que aquele campo é composto de valores alfanuméricos, ou seja, uma *String*, uma função de *parsing* de Datas é executado. Esta função tenta aplicar uma máscara do tipo dia/mês/ano na string, e caso não consiga, retorna um valor especial que sinaliza a falha (*non available*, ou *NA*).

Depois disto são contados quantos valores a função de *parsing* falhou em detectar, e caso ela tenha realizado um *parsing* bem sucedido de mais de metade das entradas, o campo é classificado como **Timestamp**. Caso contrário, o campo é classificado como **Categórico**.

Se a pré-deteção identificar um campo do tipo lógico, ou seja, constituído apenas de valores que representam verdadeiro ou falso, o campo é classificado como **Categórico**. Caso discorde das sugestões feitas pela heurística de deteção automática de tipo, o usuário poderá atribuir novos tipos para os campos utilizando uma interface *drag-and-drop*, em que o usuário pode arrastar um tipo até um campo.

Na interface *drag-and-drop* de seleção de tipo, cada tipo é identificado pelo seu nome e por uma cor diferente seguindo uma escala de cor qualitativa (Harrower e Brewer, 2013). A natureza da escala de cor qualitativa reforça a ideia de que aqueles tipos são distintos. A utilização das cores foi um artifício inserido nessa etapa do sistema para tornar sua interface mais amigável para o usuário. Ao arrastar e soltar um tipo sobre um campo, aquele campo fica da mesma cor representante do tipo.

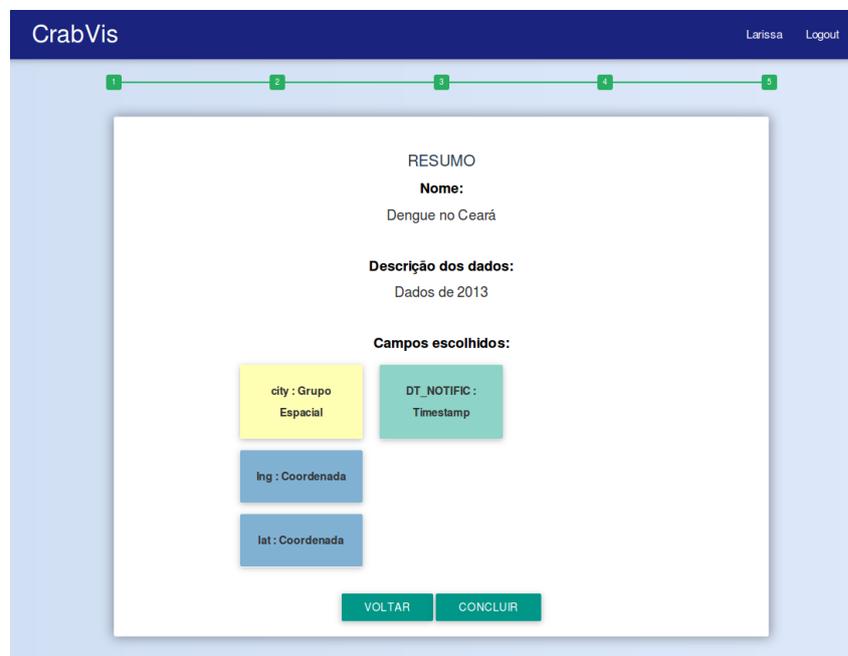
A etapa seguinte (quarta etapa) a de deteção dos tipos de campos obtém informações sobre a granularidade espacial dos dados do usuário. Quando o usuário possui apenas dados de **Grupos espaciais**, o sistema requisita que ele informe a granularidade máxima e mínima daqueles dados. Caso ele possua um campo cujo tipo é **Grupo espacial** e este campo possui as seguintes entradas: “João Pessoa, Fortaleza, Salvador”, o sistema exibirá

essas entradas e questionará o usuário o que aquelas entradas representam. O usuário deverá responder que são cidades. Com essa pergunta, o sistema obtém informação da granularidade espacial máxima dos dados (maior detalhe).

A pergunta subsequente questiona o usuário acerca da granularidade espacial mínima. Caso o usuário possua informações tanto de **Coordenada** e **Grupo espacial** ou somente **Coordenada**, o sistema apresentará apenas a pergunta que questiona a granularidade espacial mínima dos dados, ou seja, qual a região que abrange aqueles dados. O motivo disto é o fato de que, como o usuário já possui dados de **Coordenada**, seus dados já estão na granularidade mais detalhada que um dado espacial pode possuir.

Ao inserir dados que contêm coordenadas no sistema, o servidor calculará uma *bounding box* para aqueles dados e guardará aquela informação para consultas futuras.

Figura 5.11 – Resumo da Importação do novo conjunto de dados.



Fonte: Elaborada pelo autor.

Na última etapa da importação de dados, o sistema apresenta um resumo de todas as escolhas tomadas pelo usuário durante o processo (Figura 5.11). Quando o usuário confirma o que foi apresentado no resumo, o sistema realiza a importação desses dados para a camada de informação. O controlador irá, então, receber os dados da importação de mapas e criar as entradas nas tabelas de dados, colunas dos dados, entradas de dados, coordenadas, leituras numéricas, leituras categóricas, grupos espaciais, coordenadas e timestamps.

5.5 Criação da Dashboard

A camada do servidor guarda a estrutura da dashboard num arquivo json único para cada dashboard. Este arquivo é constantemente modificado dependendo das ações que o usuário tomar durante a edição da dashboard. Algumas ações possíveis são criar, redimensionar, deletar ou mover um objeto de texto e criar, redimensionar, deletar ou mover um gráfico.

A decisão de utilizar um arquivo json para salvar a estrutura da dashboard foi baseada, dentre outras motivações, no fato de que, como informações de posicionamento ou tamanho de um gráfico específico em uma dashboard são, dificilmente, objetos de consultas em um sistema de visualização, estas informações podem ficar separadas do banco e ser entregues aos usuários apenas quando eles solicitarem a visualização daquela dashboard específica. Caso estas informações ficassem no banco, o banco estaria armazenando informações inúteis, que raramente seriam consultadas (prejudicando assim o principal propósito da utilização de um banco de dados).

O armazenamento desse tipo de informação no banco de dados aumentaria o volume do banco e não traria grandes ganhos em termos de performance, pois, de qualquer forma, ao requisitar uma dashboard, um usuário teria que realizar várias consultas para obter todas as informações necessárias para recuperá-la e ainda sim realizar o download do arquivo json contendo as posições de cada elemento da dashboard.

A interação com o módulo de visualização, responsável pela criação de dashboards, possui um workflow próprio, continuação do workflow principal do sistema (como pode ser visto na Figura 4.3). Inicialmente, o usuário seleciona os conjuntos de dados que serão retratados naquela visualização (imagem à esquerda na Figura 5.12). Neste momento, o sistema realiza uma consulta pelos mapas que contém inteiramente aquela *bounding box* e os sugere para o usuário. O usuário deverá escolher alguns mapas que serão utilizados naquela dashboard (imagem à direita na Figura 5.12).

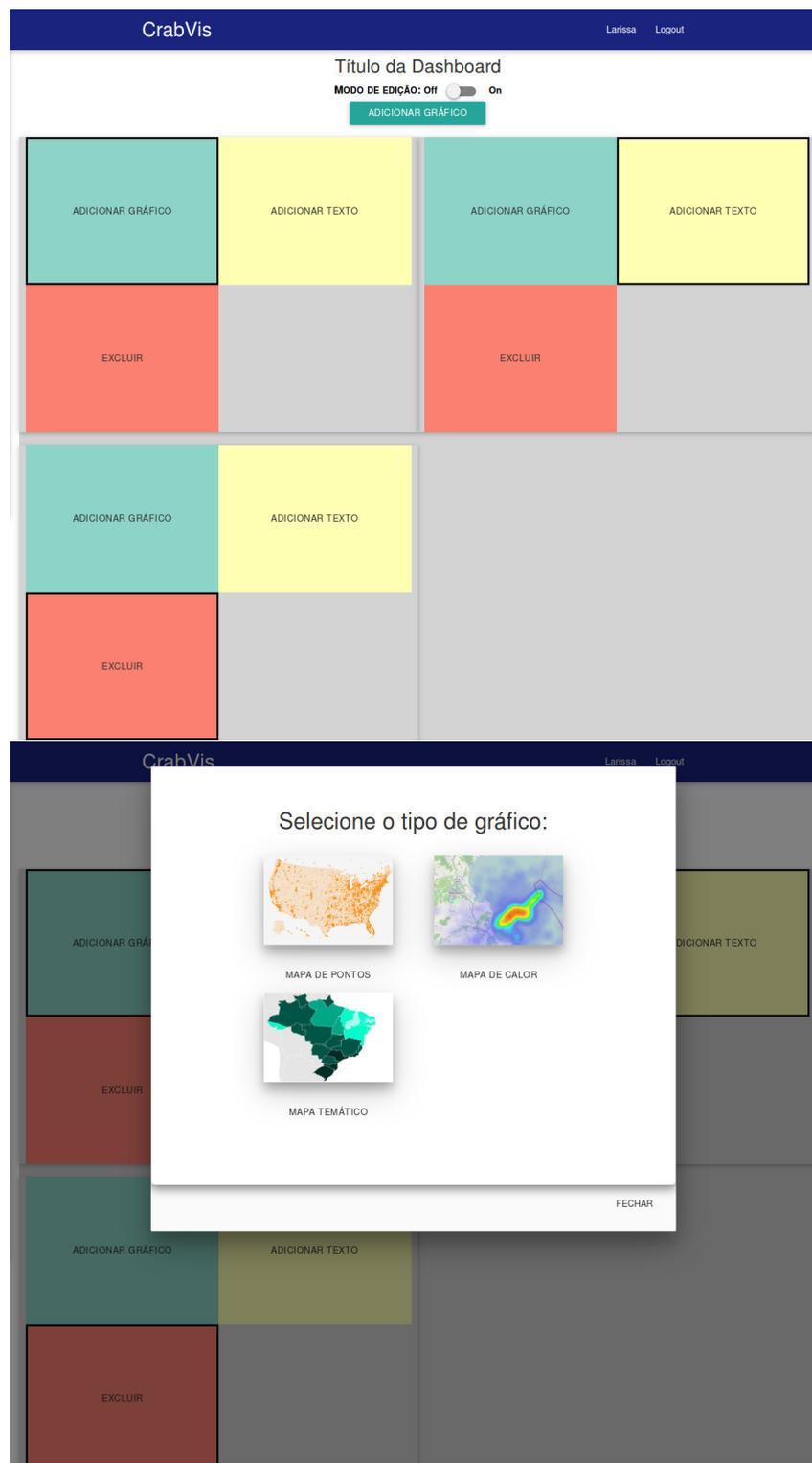
Figura 5.12 – Seleção de dados (acima) e mapas (abaixo) para a nova dashboard.



Fonte: Elaborada pelo autor.

Uma vez que o usuário tiver escolhido os dados e mapas e prosseguir, ele irá para a tela de criação de dashboard propriamente dita (onde poderá criar suas visualizações). A área da dashboard é composta de widgets que podem ser movidos e redimensionados (Figura 5.13). Cada widget poderá conter um mapa ou texto. O usuário poderá adicionar novos widgets ou removê-los.

Figura 5.13 – Tela inicial da criação de dashboard (acima) e tela da seleção de gráficos (abaixo).



Fonte: Elaborada pelo autor.

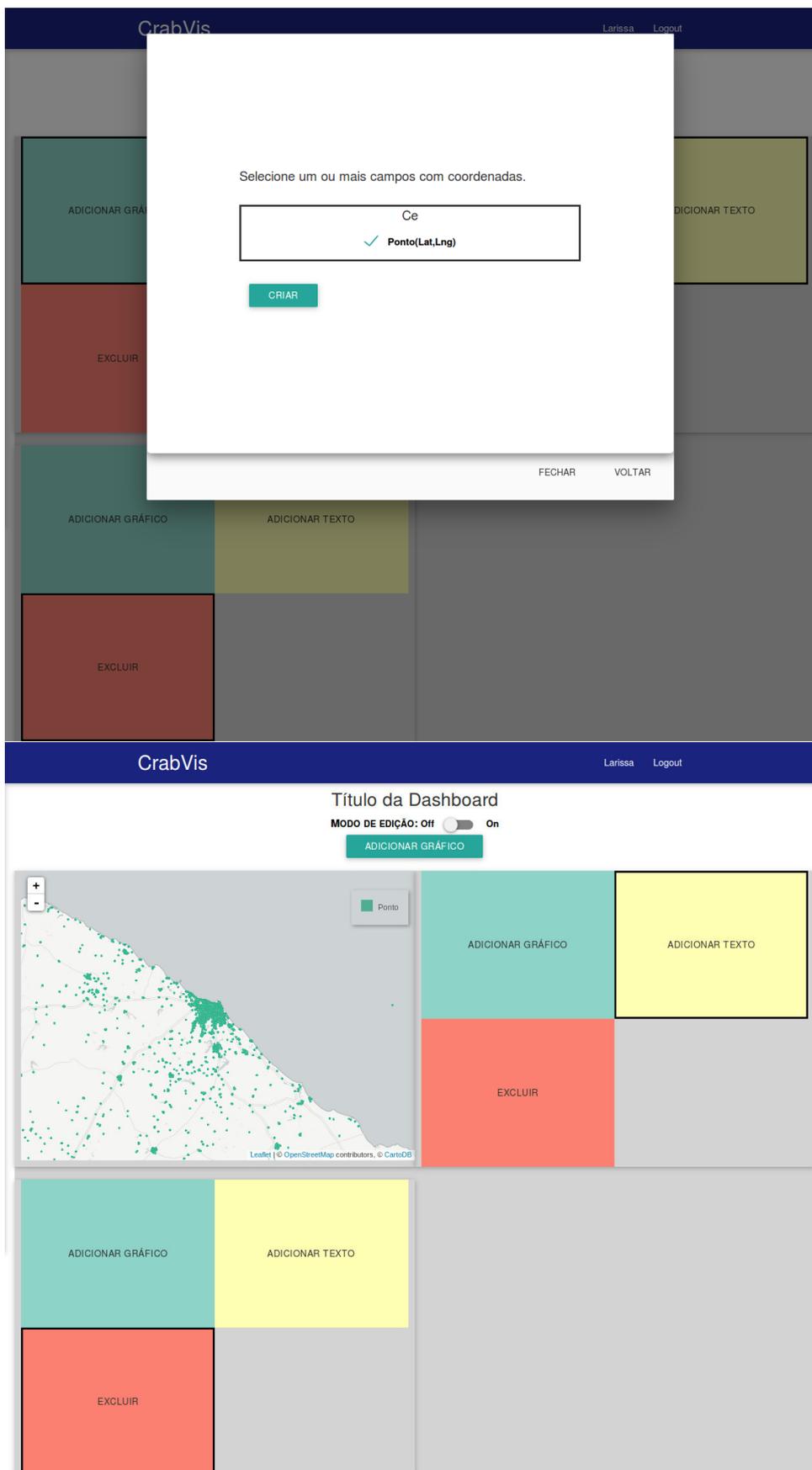
Ao clicar em “Adicionar novo gráfico”, uma tela de seleção de gráfico irá aparecer. Nesta tela, o usuário deverá escolher o tipo de gráfico que deseja criar (Figura 5.13). Atualmente, os seguintes gráficos estão disponíveis no sistema: gráfico de pontos, mapa de calor e mapa temático.

A implementação dos gráficos presentes no sistema tem como alvo principal a exploração de dados urbanos. Levando em consideração a rapidez com que tais dados ganham volume, será cada vez mais comum que os usuários tenham que visualizar dados com extenso volume. Por este motivo, dar suporte ao que é referido na literatura como *big data* é necessário.

5.5.1 Mapa de pontos

Na criação de mapa de pontos, o usuário seleciona uma ou mais colunas de coordenadas, que podem ser de conjuntos de dados diferentes (Figura 5.14). O usuário não precisa informar qual cor deseja utilizar para representar cada coluna, pois o sistema já atribui cores automaticamente para os pontos de diferentes colunas espaciais. Esta decisão foi tomada de forma a acelerar o processo de criação do mapa de pontos, para que, com poucos cliques, o usuário possa visualizar os dados que possui. Após a criação do mapa, caso deseje, o usuário poderá alterar as cores dos pontos.

Figura 5.14 – Processo de criação de mapa de pontos. Seleção das colunas com coordenadas (acima) e mapa resultante (abaixo).

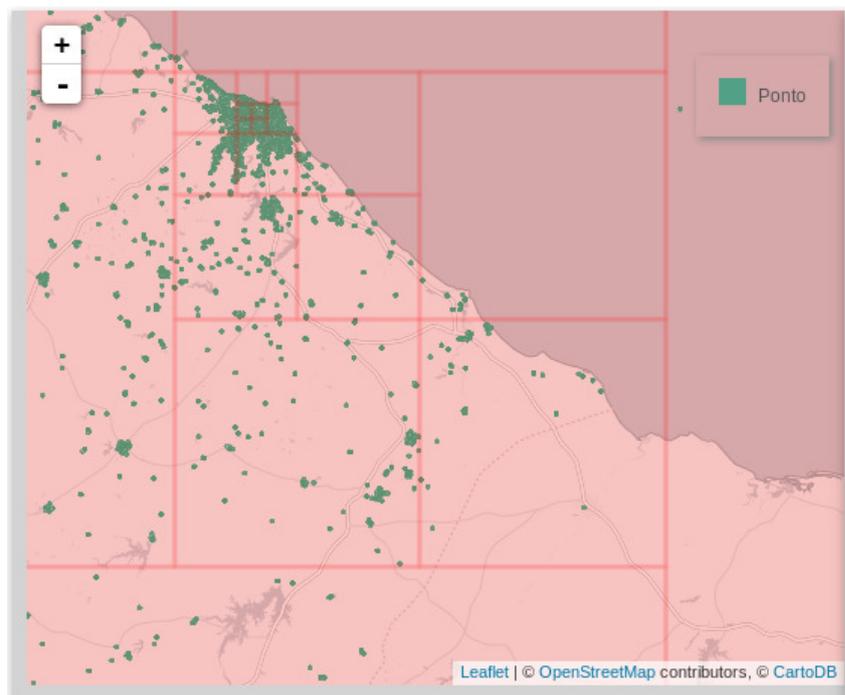


Fonte: Elaborada pelo autor.

O mapa de pontos no sistema foi implementado utilizando WebGL. Outras tecnologias como Svg (Scalable Vector Graphics) e Canvas (do Html5) também foram testadas, porém o desempenho da renderização de pontos com o WebGL foi significativamente superior se comparado ao Svg e Canvas.

Como o mapa de pontos implementado suporta um número elevado de pontos (chegando até a 10 milhões de pontos no navegador Mozilla Firefox), é desejável que a interação com o mapa se dê de uma forma igualmente eficiente. O clique é um exemplo de interação com o mapa. Ao clicar num ponto do mapa, por exemplo, o usuário vê um pop up que possui as demais informações relacionadas àquele ponto (tais como marca temporal, valores numéricos e categóricos).

Figura 5.15 – *Quadtree* no mapa de pontos.



Fonte: Elaborada pelo autor.

Para atingir maior eficiência, uma *quadtree* foi implementada na função de renderização de mapa de pontos (detalhe na Figura 5.15). Uma *quadtree* é uma árvore em que cada nível corresponde a uma subdivisão e cada nó possui quatro filhos (Angel e Shreiner, 2011). A *quadtree* implementada contém todos os pontos a serem exibidos pelo mapa de pontos. Um limiar foi estabelecido empiricamente de quantos pontos cada nó da *quadtree* poderia suportar antes de se subdividir. Caso o nó tenha mais pontos do que o limiar, ele se subdivide em mais quatro nós até que cada nó da *quadtree* possua menos pontos do que o limiar estabelecido.

Ao clicar no mapa de pontos, o sistema captura a posição de latitude e longitude atual e verifica em qual dos nós da *quadtree* o usuário clicou. Ao obter o nó, o sistema calcula a distância euclidiana até os outros pontos dentro daquele nó na *quadtree*. O ponto que

tiver a menor distância euclidiana até a latitude e longitude do clique é designado como ponto “clicado” na *quadtree*. A eficiência de utilizar a *quadtree* no mapa de pontos reside no fato que esta limita o espaço de busca pelo ponto clicado.

5.5.2 Mapa temático

Para a construção de um mapa temático, além do mapa a ser colorido, são necessários dados em forma de tupla em que o primeiro elemento da tupla é o nome de uma região do mapa e o segundo é um valor. O mapa é colorido de acordo com uma escala de cores em que cada valor é mapeado para uma cor da escala. Numa escala de cores sequencial, por exemplo, o menor valor é representado pela cor menos saturada presente na escala, enquanto o maior valor é representado pela cor mais saturada presente na escala.

Para criar um mapa temático no sistema, o usuário deverá escolher um mapa, que será a base do mapa temático, uma coluna com informação espacial, uma coluna com valores numéricos e uma função. A função escolhida realizará um mapeamento dos dados do usuário para os valores que serão fornecidos à escala de cor que irá colorir cada polígono do mapa. As funções ofertadas pelo sistema são Média, Mínimo, Máximo, Soma e Contagem.

Para colorir corretamente cada polígono é necessário que, para cada polígono, exista apenas um valor correspondente. Por este motivo, é imprescindível que ocorra uma junção espacial entre os dados do usuário e cada polígono do mapa. O sistema utiliza a coluna com informação espacial para realizar a junção espacial dos dados do usuário com o mapa escolhido. O cálculo da função é realizado por polígono. A função que calcula o valor que será atribuído a um polígono específico, leva em conta apenas os itens de dados que estiverem associados àquele polígono.

A associação de um item de dados a um polígono pode ser verificada utilizando uma coluna com grupos espaciais ou com coordenadas. Caso o usuário tenha dados de grupos espaciais e a granularidade dos dados for igual à granularidade máxima do mapa, é possível realizar uma associação direta dos valores na coluna de grupo espacial e os nomes dos polígonos do mapa (por exemplo, e o grupo espacial for composto por nomes de municípios do Ceará e o mapa escolhido representar o Ceará em municípios).

Caso o usuário não tenha informação de grupo espacial mas possua coordenadas, o cálculo dos dados necessários para o mapa temático é realizado através de uma consulta no banco que verifica quais pontos estão dentro de cada polígono. A consulta de agregação espacial utilizando coordenadas pode ser muito custosa para o servidor. Por esta razão, o sistema realiza duas otimizações na consulta. A primeira é o uso de um índice Gist (PostgreSQL: Documentation: 9.5: Introduction, 2016) do PostGis nas tabelas de polígonos. O índice Gist do PostGres representa um índice em uma árvore de busca e este índice é usado de forma implícita pelas funções do PostGis para a realização de cálculos espaciais. A segunda otimização é verificar se um ponto está dentro da *bounding box* do polígono antes de verificar se ele está dentro do polígono. Mesmo com as otimizações realizadas, caso os polígonos contenham uma geometria muito complexa (isto é, caso cada polígono possua

uma grande quantidade de arestas), a consulta pode ser lenta. Em suma, a eficiência desta consulta depende do número de pontos e da complexidade da geometria do mapa.

Como o sistema tenta simplificar o máximo a criação de mapas, a escolha da paleta de cores do mapa temático é feita pelo sistema e pode ser mudada, posteriormente, pelo usuário. Inicialmente, o sistema atribui ao mapa temático uma paleta de oito cores sugerida pelo Color Brewer (Harrower e Brewer, 2013). O sistema supõe, então, que os dados do usuário estão igualmente distribuídos nessas faixas e calcula a largura de cada intervalo de cor baseado nos valores máximo e mínimo da coluna de valores numéricos. Essa suposição não é sempre correta, pois há casos em que os dados são muito díspares e há uma grande diferença entre o maior valor e o menor valor, além da possibilidade de muitos dados estarem concentrados em apenas algumas faixas da escala de cor.

5.5.3 Mapa de calor

De forma similar ao mapa de pontos, para a criação de um mapa de calor no sistema o usuário deverá especificar apenas uma coluna com coordenadas (Figura 5.16).

Figura 5.16 – Processo de criação de mapa de calor. Seleção das colunas com coordenadas (acima) e mapa resultante (abaixo).



Fonte: Elaborada pelo autor.

5.6 Casos de Uso

Nesta seção, estão retratados dois casos de uso utilizados para validar o sistema criado. Os dois casos de uso são referentes ao Brasil e utilizaram o mesmo arquivo de mapa. O caso de uso presente na subseção 5.6.2 utilizou um mesmo arquivo de mapa do Ceará para os dois mapas temáticos do Ceará presentes na dashboard. Os dados que foram utilizados na construção dos dois casos de uso estão disponíveis no anexo desta dissertação.

5.6.1 Relação entre Saneamento básico e a dengue no Brasil

A dashboard mostrada na Figura 5.17 visa explorar se existe uma relação entre saneamento básico e a dengue. Os dados foram obtidos de duas fontes, do portal público Infraestrutura Nacional de Dados Espaciais (IBGE, 2016) e do Instituto Brasileiro de Geografia e Estatística (2015). O INDE disponibiliza alguns dados espaciais públicos e, dentre estes dados, está um conjunto de dados que contém todas as cidades que possuíam todos os serviços de saneamento básico no ano de 2008. Os dados das notificações dengue por estado no ano de 2008 foram obtidos do Instituto Brasileiro de Geografia e Estatística (2015).

Os gráficos utilizados na dashboard foram mapa de pontos (para apresentar os municípios com todos os serviços de saneamento básico) e mapa temático (para apresentar todas as notificações de dengue do ano de 2008). Os dados da dengue estavam no formato xls e foram convertidos para csv.

O arquivo com os dados da dengue possui uma coluna referente à unidade federativa e o restante das colunas referentes a cada ano (de 1990 a 2008) (arquivo tabular presente na tabela A.1 no anexo A). Ao importar o arquivo para o sistema, o sistema reconheceu a coluna referente aos nomes dos estados como categoria e a coluna referente ao número de intoxicações como número. O autor da dashboard interagiu com o sistema para informar, então, que a coluna referente ao nome dos estados era uma coluna do tipo grupo espacial. O usuário selecionou as duas colunas presentes no arquivo para importação.

O arquivo com os dados obtidos do INDE possuem uma coluna referente ao nome das cidades e uma coluna referente à coordenada geográfica, além de outras colunas com códigos do IBGE. O arquivo está disponível na página “Municípios com todos os serviços de Saneamento Básico em 2008”. A coluna referente ao nome das cidades e a coluna com coordenadas geográficas foram selecionadas para importação. O sistema reconheceu a primeira coluna como categoria e a segunda como coordenada.

Na etapa de criação de dashboards, o usuário selecionou os dados referentes às informações de saneamento e de dengue. O sistema utilizou, então, a informação de *bounding box* dos dados de saneamento para sugerir um mapa do Brasil. O usuário selecionou, então, o mapa do Brasil dividido em estados.

Para criação do mapa de pontos, o usuário precisou apenas selecionar uma coluna referente a coordenadas. Na criação do mapa temático, o usuário selecionou o mapa

Figura 5.17 – Saneamento básico e a dengue em 2008.



Fonte: Elaborada pelo autor.

que havia associado à dashboard e os dados de dengue. No menu de criação do mapa temático, o usuário escolheu uma coluna referente aos dados espaciais (a coluna de nomes de estados presente nos dados de casos de dengue) e escolheu realizar uma operação de soma em outra coluna com valor numérico (a coluna que possuía o número de casos de dengue para aquele estado). O sistema conseguiu realizar a junção espacial dos nomes dos estados (presentes nos dados de dengue) com o nome dos estados presentes no mapa

do usuário.

Como o sistema implementado possibilita a inserção de imagens e texto, é possível fazer uma dashboard que combina vários elementos e mapas interativos. A fim de expor o seu ponto de vista, o usuário utilizou recursos textuais, destacando algumas expressões e manchetes na dashboard. O sistema também permite que o usuário redimensione e reposicione cada elemento da dashboard.

No caso de uso, o autor destacou que em algumas áreas que possuíam pouca presença saneamento básico havia alta taxa de número casos de dengue. Para reforçar a ideia apresentada na dashboard, o autor colocou uma parte de uma entrevista com especialista ressaltando a importância de políticas governamentais de saneamento no combate à dengue.

5.6.2 Agrotóxicos no Brasil: Comparativo entre os anos de 2009 e 2014

As dashboards nas Figuras 5.18 e 5.19 almejam realizar um comparativo entre os anos 2009 e 2014 no Brasil no tocante à questão da venda de agrotóxicos. A fonte de dados utilizada para argumentação no texto foi um documento do Instituto Nacional do Câncer (2016). Através da dashboard o usuário pode clicar no link para o documento e ser redirecionado para o mesmo.

As fontes de dados utilizadas nos gráficos foram o portal Dados Contra Agrotóxicos (Tygel¹, Alan Freihof and Gonçalves, Leonardo and Santos, Mayara and Marques, Gabriel and Campos, Maria Luiza Machado, 2016), para os dados referentes ao número de intoxicações por estado e o Instituto Brasileiro do Meio Ambiente e dos Recursos Naturais Renováveis (2016) para o número de vendas de toneladas de agrotóxico por estado. Para os dados de intoxicação por município do Ceará, a fonte de dados utilizada foi o Departamento de Informática do Sistema Único de Saúde (2016).

Os gráficos presentes na Figura 5.18 comparam a venda de toneladas de ingrediente ativo de agrotóxico no Brasil, por estado, nos anos em questão. O mapa é interativo e é possível ver quantas toneladas cada estado comprou naquele ano.

A Figura 5.19 apresenta mapas temáticos em que é possível ver o número de notificações de intoxicação em cada estado e também nos municípios do estado do Ceará (mapas inferiores da Figura). O autor quis dar destaque para os municípios de Sobral e Fortaleza, e, por este motivo, ele ampliou o mapa naqueles pontos. Caso contrário, ele veria mapas temáticos similares aos outros mapas presentes na dashboard.

Os arquivos disponibilizados pelo IBAMA sobre venda de agrotóxicos foram convertidos em arquivos csv e possuem apenas duas colunas: a primeira coluna refere-se à sigla da unidade federativa, enquanto a segunda refere-se ao total das vendas de toneladas de agrotóxicos para aquela unidade federativa (arquivos tabulares presentes nas tabelas A.2 e A.3 do anexo A). Como o IBAMA disponibilizou um arquivo para cada ano, foi necessária a importação de dois arquivos (para 2009 e 2014). Durante a importação, o

sistema reconheceu a coluna de unidade federativa como categoria e a coluna com o valor de vendas como número. O usuário modificou o tipo da coluna de unidade federativa para grupo espacial.

Os dados de intoxicação do portal Dados Contra Agrotóxicos possuem uma coluna que representa os nomes de estados brasileiros e outras colunas que representam o número de intoxicações em cada ano (arquivo tabular presente na tabela A.4 do anexo A). O sistema reconheceu os nomes de estados brasileiros como categoria e o restante das colunas como números. O usuário interagiu com o sistema e explicitou que a coluna de nomes de estados referia-se a um grupo espacial. De forma similar, os dados de intoxicação do Datasus possuíam apenas uma coluna com o nome dos municípios cearenses e outra coluna com o número de notificações (arquivos tabulares presentes nas tabelas A.5 e A.6 do anexo A). O nome de municípios foi reconhecido como categoria e posteriormente modificado para grupo espacial, enquanto o número de notificações foi reconhecido como número.

Nesta dashboard foram utilizados três mapas, um mapa do Brasil dividido em unidades federativas (com os nomes das siglas identificando cada unidade federativa), um mapa do Brasil dividido em estados (com o nome de cada estado por extenso) e um mapa do Ceará dividido em municípios. Dois mapas do Brasil foram utilizados a fim de que fosse possível realizar a junção espacial dos mapas com os dados, pois alguns dados utilizavam as siglas dos estados e outros o nome.

Durante o processo de importação do mapa do Ceará, como o mapa do Brasil já havia sido importado, durante a etapa de definição da granularidade mínima do mapa, o sistema já detectou que havia um Map Container chamado Ceará e o sugeriu para o usuário. A criação dos mapas temáticos foi realizada através da seleção do mapa, seleção da coluna que possuía o dado de grupo espacial e seleção da operação.

O sistema implementado permitiu que, com poucos cliques, gráficos como mapa temático e mapa de pontos fossem construídos. Caso o mesmo gráfico fosse criado do zero utilizando uma linguagem ou biblioteca, este processo seria mais custoso e obrigaria o usuário a realizar a junção espacial entre dados e mapas.

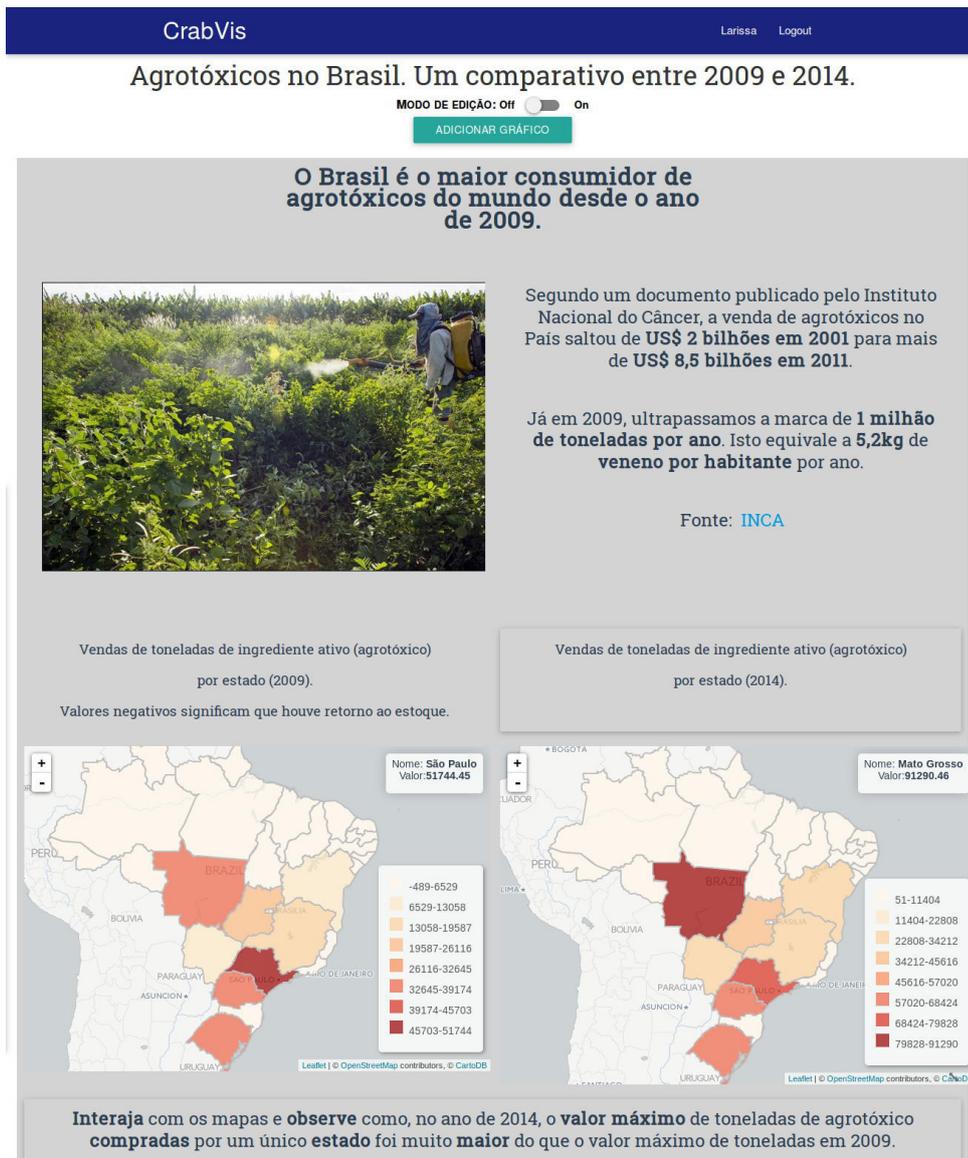
Construir os casos de uso apresentados neste trabalho em um sistema como o Plotly ou o Quadrigram não seria possível pois tais sistemas não permitem que o usuário importe um arquivo de geometria. Para construir os casos de uso utilizando o Tableau Desktop, o usuário teria que utilizar um Sistema de Informação Geográfica para converter o seu arquivo em um formato que o Tableau consiga compreender, para, só então, iniciar a construção da dashboard.

Neste capítulo, um sistema implementado com base na arquitetura proposta, o Crab-Vis, foi apresentado. Todas as tecnologias utilizadas no desenvolvimento do sistema foram listadas e, além disso, foram detalhados os algoritmos adicionais implementados no sistema a fim de otimizar a sua performance. Foram apresentados dois casos de uso do sistema, que tratavam da relação entre as notificações de dengue e a falta de saneamento básico e da relação entre vendas de agrotóxicos nas unidades federativas e o número crescente de

notificações de intoxicação registradas nas entidades de saúde.

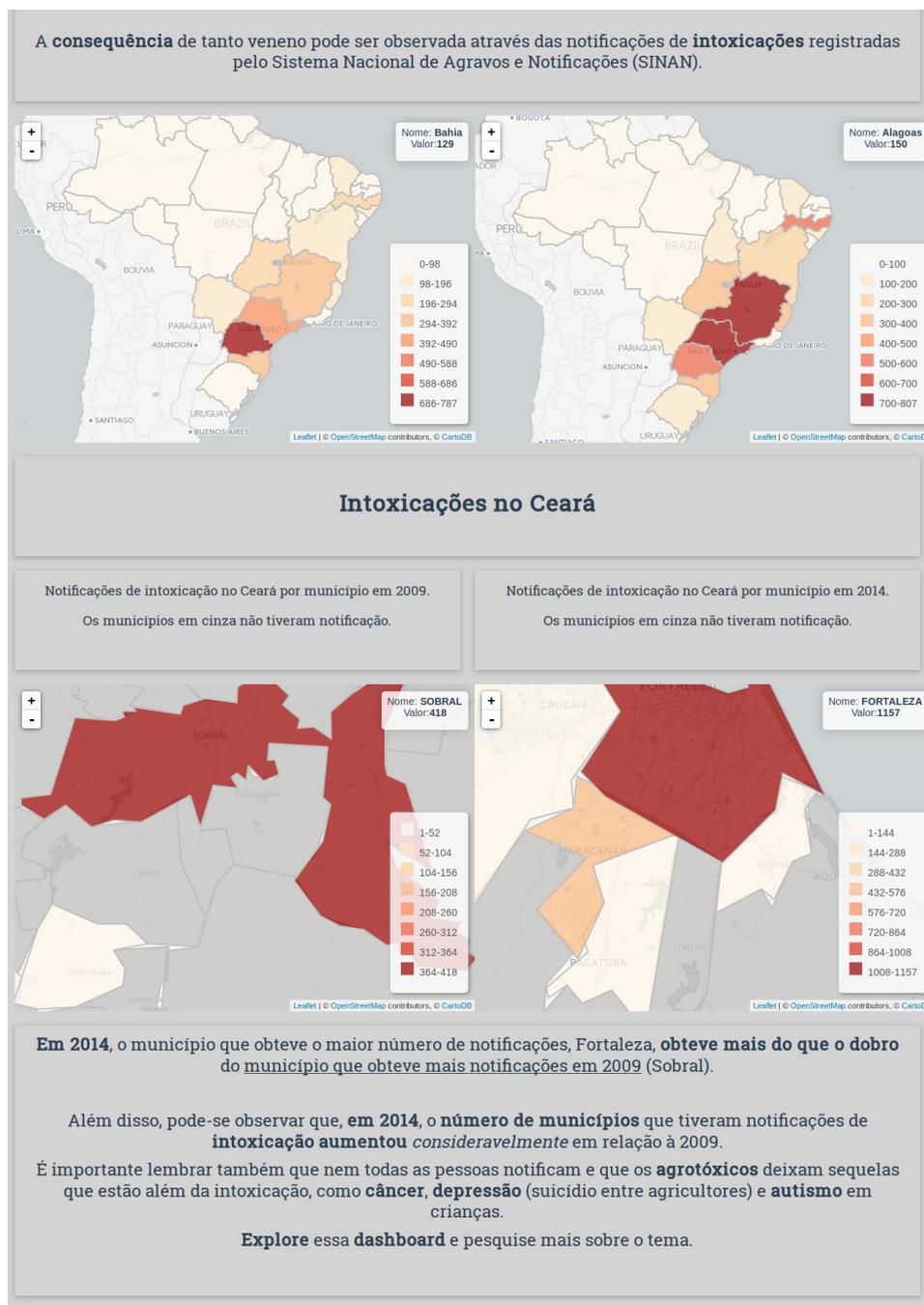
Os casos de uso apresentados são um exemplo de que o sistema permite a junção de diferentes fontes de dados para criar dashboards com temas coerentes e relevantes para a sociedade. A dashboard é um recurso útil para contar histórias e promover discussões, pois os dados presentes em uma dashboard podem ser explorados de forma interativa e possibilitam a formação de hipóteses. A capacidade de gerar discussões sobre temas multidisciplinares torna o sistema implementado em uma ferramenta que poderia ser utilizada por alunos em escolas, por exemplo. Os alunos poderiam tanto analisar uma dashboard já existente como poderiam criar suas próprias dashboards no sistema.

Figura 5.18 – Análise sobre o uso de agrotóxicos no Brasil nos anos de 2009 e 2014.



Fonte: Elaborada pelo autor.

Figura 5.19 – Análise sobre o uso de agrotóxicos no Brasil nos anos de 2009 e 2014.



Fonte: Elaborada pelo autor.

6 Conclusão e Trabalhos Futuros

Tornar públicos vários tipos de dados em portais da Web é apenas um passo para transformar estes dados em conhecimento útil que poderá ajudar a melhorar a vida das pessoas. O outro passo que deve ser dado é criar ferramentas de Visualização que são acessíveis a um público heterogêneo. É importante reconhecer que as pessoas que acessam os portais governamentais, bem como outros portais de dados, possuem os mais diversos graus de conhecimento em Visualização de Dados.

Na Visualização de dados espaciais e espaço-temporais, dependendo do tipo de Visualização que um usuário quiser criar, ele terá que aprender a manipular um SIG ou aprender a programar, pois a Visualização de dados espaciais e espaço-temporais ainda está começando a ser um aspecto explorado em sistemas de Visualização Web.

Neste trabalho foi apresentada a CrabVis, arquitetura de sistema Web para Visualização de dados espaço-temporais capaz de explorar a granularidade de dados espaciais. A arquitetura é conceitualmente dividida em camadas e segue o modelo cliente-servidor. A viabilidade e funcionalidades da arquitetura foram verificadas através de sua implementação e da criação de dois casos de uso.

A granularidade de um dado espacial é uma informação muito rica, porém pouco explorada por sistemas de Visualização Web. A riqueza da granularidade reside na sua natureza inerentemente hierárquica. O sistema proposto explorou esse aspecto criando uma base de conhecimento (Map Container) sobre as mais diversas regiões na medida em que as pessoas interagem com o sistema proposto e informam a granularidade máxima e mínima de seus dados. O tratamento da granularidade de dados espaciais foi a decisão que mais influenciou na estrutura da arquitetura do sistema como um todo.

Reduzir o número de escolhas que um usuário toma ao construir a sua visualização gera uma rápida prototipagem de gráficos mas também possui a desvantagem de gerar uma perda na customização e generalidade do sistema. Porém, muito frequentemente, quanto mais geral um sistema se torna, mais conhecimento aprofundado de visualização é exigido de seus usuários e menos intuitivo ele se torna. Ou seja, geralmente, quanto mais específico um sistema é, mais rápido e produtivo é o seu uso e menos decisões o usuário toma para conseguir algum resultado, e quanto mais geral um sistema é, maiores são as possibilidades que ele pode fazer dentro dele, e, por consequência, maior é a sua curva de aprendizado para utilizar o sistema.

Até o presente momento, o sistema conta com gráficos específicos para dados espaciais

mas serão adicionados também gráficos de propósito geral como gráficos de linhas e barras, todos com foco na integração com os dados espaciais. A abstração da dashboard na arquitetura proposta permite uma fácil integração de diversos tipos de gráficos ao sistema com pouca dificuldade. A arquitetura também pode ser facilmente estendida para oferecer comentários em dashboards e anotações nos gráficos.

A presença da dashboard em um sistema de Visualização é um importante recurso para tecer histórias e apresentar idéias. Poder personalizar a dashboard com outros elementos além de mapas como texto e imagens é um recurso simples porém de grande utilidade. Além disso, formatar o texto escrito destacando partes importantes e centralizando o texto enriquece a narrativa presente na dashboard.

O meio em que o sistema está, que é a Web, torna ele mais acessível e democrático, pois torna a informação que o usuário deseja compartilhar disponível independente de plataforma (programas) e formatos de arquivos específicos.

Uma possibilidade de trabalho futuro é fazer um estudo de campo no qual o sistema seria avaliado ao ser utilizado por pessoas leigas em situações comuns do dia-a-dia. A princípio, o sistema poderia ser utilizado em escolas, no estudo de assuntos multidisciplinares e que envolvam dados espaço-temporais.

A Apêndice - Dados Utilizados nos Casos de Uso

Tabela A.6 – Intoxicações nos Municípios do Ceará (2014)

Município	Total_not
Acaraú	6
Aracati	21
Aracoiaba	2
Assaré	1
Barbalha	36
Barro	10
Bela Cruz	15
Canindé	427
Cariré	1
Caririaçu	1
Cariús	3
Carnaubal	4
Caucaia	47
Choró	1
Coreaú	3
Crateús	4
Crato	66
Croatá	5
Cruz	1
Eusébio	2
Farias Brito	22
Fortaleza	1157
Fortim	1
Frecheirinha	4
Granjeiro	1
Guaraciaba do Norte	13

Continua na próxima página

Tabela A.6 – continuação da página anterior

Município	Total_not
Horizonte	3
Ibaretama	1
Ibiapina	50
Icapuí	3
Iguatu	18
Ipu	5
Ipueiras	2
Iracema	3
Itapipoca	10
Itapiúna	2
Itarema	20
Jaguaribe	3
Jaguaruana	3
Juazeiro do Norte	16
Limoeiro do Norte	9
Maracanaú	448
Marco	2
Mauriti	1
Missão Velha	2
Morrinhos	1
Mulungu	17
Nova Russas	1
Ocara	1
Pacajus	3
Pacatuba	3
Palhano	1
Pedra Branca	4
Pentecoste	1
Piquet Carneiro	5
Porteiras	2
Quiterianópolis	5
Quixadá	24
Quixeramobim	22
Quixeré	3
Reriutaba	16
Russas	4
Salitre	1

Continua na próxima página

Tabela A.6 – continuação da página anterior

Município	Total_not
São Benedito	16
São Gonçalo do Amarante	1
Sobral	116
Solonópole	3
Tamboril	1
Tianguá	51
Ubajara	19
Várzea Alegre	30
Viçosa do Ceará	12

Tabela A.1 – Casos de Dengue por UF (2008)

UF	2008
Rondônia	44
Acre	727
Amazonas	132
Roraima	351
Pará	13
Amapá	37
Tocantins	85
Maranhão	7
Piauí	359
Ceará	6
Rio Grande do Norte	1330
Paraíba	12
Pernambuco	5
Alagoas	22
Sergipe	286
Bahia	212
Minas Gerais	84
Espírito Santo	66
Rio de Janeiro	3
São Paulo	28
Paraná	42
Santa Catarina	5
Rio Grande do Sul	24
Mato Grosso do Sul	65
Mato Grosso	65
Goiás	210
Distrito Federal	28

Tabela A.2 – Vendas de agrotóxicos por UF (2009)

UF	Toneladas
AC	105.98
AM	45.19
AP	62.83
PA	1106.33
RO	1671.84
RR	75.25
TO	1302.09
DF	528.60
GO	19173.67
MS	10147.19
MT	38158.89
AL	694.16
BA	11697.02
CE	-488.38
MA	3306.12
PB	212.24
PE	1458.62
PI	1519.09
RN	208.23
SE	215.39
ES	1599.41
MG	16056.49
RJ	2385.32
SP	51744.45
PR	32858.56
RS	33516.21
SC	5622.93

Tabela A.3 – Vendas de agrotóxicos por UF (2014)

UF	Toneladas
AC	544.81
AM	51.54
AP	166.10
PA	5312.34
RO	4708.84
RR	585.10
TO	6073.36
DF	773.38
GO	44855.57
MS	29262.38
MT	91290.46
AL	1839.38
BA	28320.77
CE	543.56
MA	9741.63
PB	654.20
PE	2624.80
PI	5520.97
RN	386.74
SE	565.67
ES	3519.58
MG	33460.32
RJ	715.13
SP	73123.38
PR	57856.89
RS	58355.53
SC	10788.18

Tabela A.4 – Intoxicações (anos 2007-2014)

UF Notificação	<2007	2007	2008	2009	2010	2011
Rondonia	2	54	50	32	53	88
Acre	0	0	0	0	0	1
Amazonas	0	1	2	2	9	15
Roraima	0	1	0	3	2	3
Para	2	6	6	4	11	18
Amapa	0	0	0	0	0	0
Tocantins	4	47	45	97	103	131
Maranhao	0	8	11	18	18	18
Piaui	0	25	18	28	24	30
Ceara	2	82	43	134	201	302
Rio Grande do Norte	0	6	13	11	12	17
Paraiba	23	13	10	2	4	12
Pernambuco	3	318	312	223	242	356
Alagoas	4	122	118	126	84	109
Sergipe	1	1	0	2	9	41
Bahia	6	106	67	129	135	218
Minas Gerais	39	197	223	371	413	626
Espirito Santo	4	76	100	141	191	290
Rio de Janeiro	13	22	40	75	69	83
Sao Paulo	112	281	333	464	408	635
Parana	16	921	779	787	686	758
Santa Catarina	1	158	263	296	261	279
Rio Grande do Sul	3	51	59	97	108	181
Mato Grosso do Sul	4	65	66	89	84	134
Mato Grosso	3	56	125	132	155	143
Goias	1	96	112	216	198	196
Distrito Federal	28	8	14	9	20	32
Total	271	2721	2809	3488	3500	4716

Intoxicações (anos 2007-2014) – continuação

UF Notificação	2012	2013	2014	Total
Rondonia	84	68	82	513
Acre	0	0	0	1
Amazonas	23	22	20	94
Roraima	6	7	22	44
Para	40	64	42	193
Amapa	0	2	0	2
Tocantins	127	145	170	869
Maranhao	17	27	17	134
Piaui	47	49	22	243
Ceara	324	263	128	1479
Rio Grande do Norte	17	10	16	102
Paraiba	61	87	58	270
Pernambuco	371	650	539	3014
Alagoas	165	156	150	1034
Sergipe	39	31	18	142
Bahia	268	327	238	1494
Minas Gerais	820	1049	807	4545
Espirito Santo	327	450	324	1903
Rio de Janeiro	51	60	59	472
Sao Paulo	653	753	725	4364
Parana	758	829	565	6099
Santa Catarina	301	319	321	2199
Rio Grande do Sul	134	112	143	888
Mato Grosso do Sul	159	139	115	855
Mato Grosso	131	155	82	982
Goiias	275	457	334	1885
Distrito Federal	47	129	40	327
Total	5245	6360	5037	34147

Tabela A.5 – Intoxicações nos Municípios do Ceará (2009)

Município	Total_not
Acaraú	2
Acopiara	3
Canindé	76
Caridade	2
Croatá	2
Eusébio	4
Fortaleza	167
Guaraciaba do Norte	15
Horizonte	2
Iguatu	29
Itarema	7
Juazeiro do Norte	19
Jucás	2
Limoeiro do Norte	2
Maracanaú	115
Marco	1
Mauriti	3
Ocara	1
Quixadá	15
Quixelô	1
Quixeramobim	12
Quixeré	2
Reriutaba	16
Russas	1
São Benedito	2
Sobral	418
Tamboril	9
Tianguá	46
Ubajara	8
Viçosa do Ceará	6

Referências Bibliográficas

- Aguirre, J., Ortiz, A., Jaramillo, J., Miranda, R., Ordóñez, C., González, A., Albet, O., Puente, H., Avellaneda, P., Rodríguez, J., Partipilo, E., Torroba, Á., Gómez, X., e Ramos, R. (2016). Quadrigram: Data visualization & presentation tool. Disponível em: <http://http://www.quadrigram.com/>. Acessado em: 22-10-2016.
- Andrienko, N. e Andrienko, G. (2006). *Exploratory analysis of spatial and temporal data: a systematic approach*. Springer Science & Business Media.
- Andrienko, N., Andrienko, G., e Gatalsky, P. (2003). Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing*, 14(6):503–541.
- Angel, E. e Shreiner, D. (2011). Interactive computer graphics: A topdown approach with shader-based opengl.
- Bach, B., Dragicevic, P., Archambault, D., Hurter, C., Carpendale, S., *et al.* (2014). A review of temporal data visualizations based on space-time cube operations. In *Eurographics Conference on Visualization*.
- Barbosa, L., Pham, K., Silva, C., Vieira, M. R., e Freire, J. (2014). Structured open urban data: understanding the landscape. *Big data*, 2(3):144–154.
- Beveridge, A. A., Lacevic, A., Rudell, F., Weber, S., e Beveridge, S. (2016). Social Explorer. Disponível em: <http://www.socialexplorer.com/>. Acessado em: 22-10-2016.
- Börner, K., Maltese, A., Balliet, R. N., e Heimlich, J. (2015). Investigating aspects of data visualization literacy using 20 information visualizations and 273 science museum visitors. *Information Visualization*, page 1473871615594652.
- Bostock, M. e Heer, J. (2009). Protovis: A graphical toolkit for visualization. *IEEE transactions on visualization and computer graphics*, 15(6):1121–1128.
- Bostock, M., Ogievetsky, V., e Heer, J. (2011). D³ data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309.
- Boy, J., Rensink, R. A., Bertini, E., e Fekete, J.-D. (2014). A principled way of assessing visualization literacy. *IEEE transactions on visualization and computer graphics*, 20(12):1963–1972.
- Boyandin, I., Bertini, E., Bak, P., e Lalanne, D. (2011). Flowstrates: An approach for visual exploration of temporal origin-destination data. In *Computer Graphics Forum*, volume 30, pages 971–980. Wiley Online Library.
- Butler, H., Daly, M., Doyle, A., Gillies, S., Schaub, T., e Schmidt, C. (2016). GeoJSON Specification. Disponível em: <http://geojson.org/geojson-spec.html>. Acessado em: 24-10-2016.

- Carroll, L. N., Au, A. P., Detwiler, L. T., Fu, T.-c., Painter, I. S., e Abernethy, N. F. (2014). Visualization and analytics tools for infectious disease epidemiology: A systematic review. *Journal of biomedical informatics*.
- Datasus (2016). Datasus. Disponível em: <http://tabnet.datasus.gov.br/cgi/tabcgi.exe?sinanet/cnv/Intoxce.def>. Acessado em: 03-11-2016.
- Esri (2016). ESRI Shapefile Technical Description. Disponível em: <https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf>. Acessado em: 24-10-2016.
- Fekete, J.-D., Van Wijk, J. J., Stasko, J. T., e North, C. (2008). The value of information visualization. In *Information visualization*, pages 1–18. Springer.
- Ferreira, N., Poco, J., Vo, H. T., Freire, J., e Silva, C. T. (2013). Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2149–2158.
- Fundação de Ciência Tecnologia e Inovação de Fortaleza (2016). Hacker Cidadão. Disponível em: <http://hackercidadao.fortaleza.ce.gov.br/>. Acessado em: 06-11-2016.
- Furtado, V., Ayres, L., De Oliveira, M., Vasconcelos, E., Caminha, C., D’Orleans, J., e Belchior, M. (2010). Collective intelligence in law enforcement—the wikicrimes system. *Information Sciences*, 180(1):4–17.
- Google Maps Javascript API — Google Developers (2016). Google Maps Javascript API — Google Developers. Disponível em: <https://developers.google.com/maps/documentation/javascript/?hl=pt-br>. Acessado em: 22-10-2016.
- Governo Federal (2015). LAI: A Lei de Acesso à Informação. Disponível em: <http://www.acaoainformacao.gov.br/assuntos/conheca-seu-direito/a-lei-de-acesso-a-informacao>. Acessado em: 10-02-2015.
- Guo, D. (2009). Flow mapping and multivariate visualization of large spatial interaction data. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):1041–1048.
- Harrower, M. e Brewer, C. A. (2013). Colorbrewer. org: an online tool for selecting colour schemes for maps. *The Cartographic Journal*.
- Hernández-Ávila, J. E., Rodríguez, M.-H., Santos-Luna, R., Sánchez-Castañeda, V., Román-Pérez, S., Ríos-Salgado, V. H., e Salas-Sarmiento, J. A. (2013). Nation-wide, web-based, geographic information system for the integrated surveillance and control of dengue fever in mexico. *PloS one*, 8(8):e70231.
- IBAMA (2016). Relatórios de Comercialização de Agrotóxicos. Disponível em: <http://www.ibama.gov.br/areas-tematicas-qa/relatorios-de-comercializacao-de-agrotoxicos/pagina-3>. Acessado em: 06-11-2016.
- IBGE (2015). IBGE. Disponível em: <http://dados.gov.br/dataset/taxa-de-incidencia-da-dengue>. Acessado em: 28-01-2015.
- IBGE (2016). Infraestrutura Nacional de Dados Espaciais. Disponível em: <http://www.inde.gov.br/>. Acessado em: 03-11-2016.
- Inc., P. T. (2015). Collaborative data science.

- Instituto Nacional de Câncer (2016). INCA. Disponível em: http://www1.inca.gov.br/inca/Arquivos/comunicacao/posicionamento_do_inca_sobre_os_agrotoxicos_06_abr_15.pdf. Acessado em: 06-11-2016.
- Kienberger, S., Hagenlocher, M., Delmelle, E., e Casas, I. (2013). A webgis tool for visualizing and exploring socioeconomic vulnerability to dengue fever in cali, colombia. *Geospatial health*, 8(1):313–316.
- Kirk, A. (2012). *Data Visualization: a successful design process*. Packt Publishing Ltd.
- Leaflet - a JavaScript library for interactive maps (2016). Leaflet - a JavaScript library for interactive maps. Disponível em: <http://leafletjs.com/>. Acessado em: 22-10-2016.
- Mike Bostock (2016). D3.js - Data-Driven Documents. Disponível em: <https://d3js.org/>. Acessado em: 22-10-2016.
- Munzner, T. (2014). *Visualization Analysis and Design*. CRC Press.
- Neto, V. C., Chiari, N. S., Carvalho, I., Pisa, I. T., e Alves, D. (2014). Desenvolvimento e integração de mapas dinâmicos georreferenciados para o gerenciamento e vigilância em saúde. *Journal of Health Informatics*, 6(1).
- Onde fui roubado (2016). Onde fui roubado. Disponível em: <http://www.ondefuiroubado.com.br/fortaleza/CE>. Acessado em: 24-10-2016.
- Porcasi, X., Rotela, C. H., Introini, M. V., Frutos, N., Lanfri, S., Peralta, G., De Elia, E. A., Lanfri, M. A., e Scavuzzo, C. M. (2012). An operative dengue risk stratification system in argentina based on geospatial technology. *Geospatial health*, 6(3):31–42.
- PostgreSQL: Documentation: 9.5: Introduction (2016). PostgreSQL: Documentation: 9.5: Introduction. Disponível em: <https://www.postgresql.org/docs/9.5/static/gist-intro.html>. Acessado em: 29-10-2016.
- Prefeitura de Fortaleza (2015). Fortaleza Dados Abertos. Disponível em: <http://dados.fortaleza.ce.gov.br/portal/inicio>. Acessado em: 30-01-2015.
- Prefeitura do Rio de Janeiro (2015). Dados Rio - Prefeitura da Cidade do Rio de Janeiro. Disponível em: <http://data.rio.rj.gov.br/>. Acessado em: 30-01-2015.
- Processing (2016). Processing. Disponível em: <http://www.processing.org/>. Acessado em: 22-10-2016.
- Processing.js (2016). Processing.js. Disponível em: <http://processingjs.org/>. Acessado em: 01-10-2016.
- Qlik (2016). Business Intelligence — Ferramentas de Visualização de Dados — Qlik. Disponível em: <http://http://www.qlik.com/pt-br>. Acessado em: 22-10-2016.
- Reades, J., Calabrese, F., Sevtsuk, A., e Ratti, C. (2007). Cellular census: Explorations in urban data collection. *IEEE Pervasive Computing*, 6(3):30–38.
- Satyanarayan, A. e Heer, J. (2014). Lyra: An interactive visualization design environment. In *Computer Graphics Forum*, volume 33, pages 351–360. Wiley Online Library.
- Satyanarayan, A., Moritz, D., Wongsuphasawat, K., e Heer, J. (2017). Vega-lite: A grammar of interactive graphics. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2017.

- Tableau (2016). Tableau Desktop — Tableau Software. Disponível em: <http://www.tableau.com/products/desktop>. Acessado em: 31-10-2016.
- Taylor Otwell (2016). Laravel - The PHP Framework For Web Artisans. Disponível em: <https://laravel.com/>. Acessado em: 24-10-2016.
- Tran, P. V., Nguyen, H. T., e Tran, T. V. (2014). Approaching multi-dimensional cube for visualization-based epidemic warning system-dengue fever. In *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*, page 86. ACM.
- Tygel¹, Alan Freihof and Gonçalves, Leonardo and Santos, Mayara and Marques, Gabriel and Campos, Maria Luiza Machado (2016). Portal de Dados Abertos Sobre Agrotóxicos. Disponível em: <http://dados.contraosagrototoxicos.org/>. Acessado em: 06-11-2016.
- Vega: A Visualization Grammar (2016). Vega: A Visualization Grammar. Disponível em: <https://github.com/trifacta/vega>. Acessado em: 22-10-2016.
- Viegas, F. B., Wattenberg, M., Van Ham, F., Kriss, J., e McKeon, M. (2007). Manyeyes: a site for visualization at internet scale. *IEEE transactions on visualization and computer graphics*, 13(6):1121–1128.
- Weessies, K. W. e Dotson, D. S. (2013). Mapping for the masses: Gis lite and online mapping tools in academic libraries.