INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
CAMPUS CAJAZEIRAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

UM MOTOR DE BUSCA PARA INFRAESTRUTURAS DE DADOS ESPACIAIS

LEANDERSON COELHO DOS SANTOS

Cajazeiras
2021

LEANDERSON COELHO DOS SANTOS

UM MOTOR DE BUSCA PARA INFRAESTRUTURAS DE DADOS ESPACIAIS

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Dr. Fabio Gomes de Andrade.

Cajazeiras
2021

IFPB
Campus Cajazeiras
Coordenação de Biblioteca
Biblioteca Prof. Ribamar da Silva
Catalogação na fonte: Daniel Andrade CRB-15/593

S237m

Santos, Leanderson Coelho dos

Um motor de busca para infraestruturas de dados espaciais / Leanderson Coelho dos Santos; orientador Fabio Gomes de Andrade.- 2021.
59 f.: il.

Orientador: Fabio Gomes de Andrade.
TCC (Tecnólogo em Análise e Desenvolvimento de Sistemas.) – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Cajazeiras, 2021.

1. Dados espaciais 2. Motor de busca 3. Recuperação de dados I. Título

004.6(0.067)

SERVIÇO PÚBLICO FEDERAL

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA

COORDENAÇÃO DA UNIDADE ACADÊMICA DA ÁREA DE INFORMÁTICA - CAMPUS CAJAZEIRAS

Às **16:00** horas do dia **05** do mês de **fevereiro** do ano de **2021,** via Google Meet, compareceu para defesa pública do **Trabalho de Conclusão de Curso**, requisito obrigatório para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, o(a) aluno(a) **LEANDERSON COELHO DOS SANTOS**, matrícula **201712010001**, tendo como Título do Trabalho **UM MOTOR DE BUSCA PARA INFRAESTRUTURAS DE DADOS ESPACIAIS** . Constituíram a Banca Examinadora os professores **FABIO GOMES DE ANDRADE** (orientador), **FRANCISCO PAULO DE FREITAS NETO** (examinador) e **FRANCISCO DALADIER MARQUES JÚNIOR** (examinador).

Após a apresentação e as observações dos membros da Banca Examinadora, ficou definido que o trabalho foi considerado **APROVADO** com nota **85,** com a condição de que o (a) aluno (a) entregue, no prazo máximo de 30 dias, a versão final do trabalho, via processo eletrônico à coordenação de curso. A versão deve conter a ficha catalográfica e atender às sugestões feitas pelos membros da banca. O código fonte desenvolvido no trabalho (caso haja) deve ser enviado para o e-mail da coordenação do curso (cads.cz@ifpb.edu.br).

Cajazeiras-PB, 5 de fevereiro de 2021.

Documento assinado eletronicamente por:

- **Leanderson Coelho dos Santos, ALUNO (201712010001) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 03/03/2021 20:14:14.
- **Francisco Daladier Marques Junior, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 02/03/2021 15:17:57.
- **Fabio Gomes de Andrade, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 12/02/2021 12:28:20.
- **Francisco Paulo de Freitas Neto, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 05/02/2021 21:05:31.

Este documento foi emitido pelo SUAP em 04/02/2021. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 154829
Código de Autenticação: 8bfb553250

*A mim mesmo, e a minha **Família**.*

AGRADECIMENTOS

A minha mãe Joyce e minha vó Marly, que durante toda minha caminhada ouviram minhas reclamações, perceberam meu cansaço, me deram apoio e forças para continuar.

A meu pai que esteve presente em poucos, mas grandes momentos.

Aos meus familiares que me deram apoio para cursar a graduação que mais desejava em outra cidade.

Aos amigos, do grupo de Whatsapp "paulo renjes", que fiz e conservei durante o curso, e que me motivaram cada vez mais a ser melhor em todos os apectos. A esses amigos que participaram dos melhores e piores momentos desta caminhada. Este ciclo se fecha, mas permaneceremos juntos aprendendo e ensinando.

Ao meu professor orientador, Prof. Dr. Fabio Gomes de Andrade que me ajudou a desenvolver este trabalho, tirou todas minhas dúvidas e ainda me deu bons conselhos.

”Eu sou uma floresta, sem dúvida, e uma noite de árvores escuras, mas quem não teme minha escuridão encontra também roseiras debaixo dos meus ciprestes.”

Friedrich Nietzsche

RESUMO

Nos últimos anos, infraestruturas de dados espaciais têm se tornado muito populares no mundo inteiro como a solução para facilitar a disseminação e o reuso de dados espaciais. Com o intuito de facilitar a localização desses dados por diferentes tipos de usuário, as IDEs atuais oferecem serviços de catálogo. Os clientes destas infraestruturas podem usar este serviço para localizar os dados nos quais estão interessados. Embora os serviços de catálogo facilitem a localização dos dados, eles possuem limitações importantes para a resolução de vários tipos de consulta. Alguns problemas surgem porque os catálogos atuais resolvem suas consultas apenas com base nos metadados informados pelos provedores dos dados no momento do registro, que normalmente são resumidos ou pouco precisos. Com o intuito de resolver essas limitações, este trabalho propõe uma nova ferramenta de busca, que extrai metadados mais precisos, em nível de tipos de feição, para melhorar a qualidade da recuperação de dados oferecidos em IDEs, permitindo a realização de consultas com restrições espaciais, temporais e temáticas. A ferramenta a ser desenvolvida, que também vai permitir a recuperação de dados, tanto em nível de serviços, quanto em nível de tipo de feição, utilizará como estudo de caso os dados oferecidos por meio da Infraestrutura Nacional de Dados Espaciais.

Palavras-chave: Infraestruturas de Dados Espaciais. Motor de Busca. Dados Espaciais.

ABSTRACT

In recent years, spatial data infrastructures have been become very popular worldwide as the solution to facilitate the dissemination and reuse of spatial data. In order to facilitate the retrieval of this data by different types of users current IDEs offer a catalog service. Users of these infrastructures can use such services to find out data they are interested in. Although catalog services make it easier to find data, they have important limitations for solving various types of queries. Some problems arise because the current catalogs solve their queries based solely on the metadata informed by the data providers at the time of registration, which are usually short or inaccurate. In order to overcome these limitations, this work proposes a new search engine which extracts more precise metadata, at the level of feature types, to improve the quality of data retrieval in IDEs making users able to solve queries with spatial, temporal, and thematic constraints. The tool to be developed, which will also allow data retrieval both at the level of services feature types will use as a case study the data offered through the National Spatial Data Infrastructure.

Keywords: Spatial Data Infrastructure. Search Engine. Spatial Data.

LISTA DE FIGURAS

Figura 1 – Arquitetura SOA 23
Figura 2 – Exemplo de requisição *GetCapabilities* 24
Figura 3 – Informações sobre o WMS buscado 24
Figura 4 – Seção de operações disponíveis em um WMS 25
Figura 5 – Seção de camada de um serviço WMS 25
Figura 6 – Requisição *GetMap* para um serviço WMS 26
Figura 7 – Resultado da requisição *GetMap* do exemplo 27
Figura 8 – Exemplo de requisição *GetCapabilities* para um serviço WFS 28
Figura 9 – Seção que descreve o serviço WFS 28
Figura 10 – Seção que descreve as requisições disponíveis do serviço WFS . . 29
Figura 11 – Seção que descreve as exceções do serviço WFS 29
Figura 12 – Seção de camada de um serviço WFS 30
Figura 13 – Exemplo de requisição *DescribeFeatureType* para um serviço WFS 30
Figura 14 – Resultado da requisição *DescribeFeatureType* para um serviço WFS 31
Figura 15 – Exemplo de requisição *GetFeature* para um serviço WFS 31
Figura 16 – Parte da resposta da requisição *GetFeature* 32
Figura 17 – Exemplo de requisição *GetRecords* para um serviço CSW 33
Figura 18 – Descrição do registro como resultado da requisição *GetRecords* . . 33
Figura 19 – URI disponíveis de um *Record* CSW 34
Figura 20 – Principal seção do arquivo de esquema do *Solr* 35
Figura 21 – Tipo de campo *text_general* 36
Figura 22 – Consulta via API REST do *Solr* 37
Figura 23 – Resultado da consulta feita ao *Solr* 38
Figura 24 – Projeto Arquitetural 42
Figura 25 – Esquema Lógico do Banco de Dados 44
Figura 26 – Diagrama de atividade para os passos feito pelo módulo de recuperação e tratamento 46
Figura 27 – Exemplo de *regex* para identificar datas 47
Figura 28 – Exemplo de datas com semestre que serão capturadas pelo *regex* . 47
Figura 29 – Exemplo de datas com textos que serão capturadas pelo *regex* . . . 48
Figura 30 – Exemplo de datas com barras ou hífen que serão capturadas pelo *regex* 48
Figura 31 – Menu de acesso aos recursos disponíveis para busca 52
Figura 32 – Formulário de pesquisa do módulo de visão web 52
Figura 33 – Resposta para consulta "Belo Horizonte", “01/01/2010” a “01/01/2011” 53

Figura 34 – Resposta para consulta "Belo Horizonte", “01/01/2010” a “01/01/2011” no portal da INDE 53
Figura 35 – Resposta encontrada pela ferramenta desenvolvida para consulta “são paulo”, "alphaville", “01/01/2013” a “31/12/2013” 54
Figura 36 – Nenhum resultado encontrado pelo portal da INDE para a consulta “são paulo”, "alphaville", “01/01/2013” a “31/12/2013” 54

LISTA DE QUADROS

Quadro 1 – características deste trabalho com os trabalhos relacionados abordados. 41

LISTA DE ABREVIATURAS E SIGLAS

ANA	Agência Nacional de Águas
CONCAR	Comissão Nacional de Cartografia
CSW	*Catalogue Service for the Web*
DNIT	Departamento Nacional de Infraestrutura de Transportes
EMBRAPA	Empresa Brasileira de Pesquisa Agropecuária
FUNAI	Fundação Nacional do Índio
GML	*Geography Markup Language*
HTTP	*Hypertext Transfer Protocol*
IBAMA	Instituto Brasileiro do Meio Ambiente
IBGE	Instituto Brasileiro de Geografia e Estatística
INDE	Infraestrutura Nacional de Dados Espaciais
OGC	*Open Geographic Consortium*
REST	*Representational State Transfer*
SOA	*Service Oriented Architecture*
TCC	Trabalho de Conclusão de Curso
URI	*Uniform Resource Identifier*
URL	*Uniform Resource Locator*
WFS	*Web Feature Service*
WMS	*Web Map Service*
XML	*Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO 14

1.1 MOTIVAÇÃO 15

1.2 OBJETIVOS 16

1.2.1 Objetivo Geral 17

1.2.2 Objetivos Específicos 17

1.3 TRABALHOS RELACIONADOS 17

1.4 CONTRIBUIÇÕES 19

1.5 METODOLOGIA 20

1.6 ESTRUTURA E ORGANIZAÇÃO DO DOCUMENTO 21

2 FUNDAMENTAÇÃO TEÓRICA 22

2.1 WEB SERVICE 22

2.2 SERVIÇOS OGC 23

2.2.1 Web Map Service (WMS) 23

2.2.2 Web Feature Service (WFS) 27

2.2.3 Serviço de Catálogo (CSW) 32

2.3 APACHE SOLR 34

3 UM MOTOR DE BUSCA PARA INFRAESTRUTURAS DE DADOS ESPACIAIS 39

3.1 ANÁLISE 39

3.1.1 *Stakeholders* 39

3.1.2 Requisitos Funcionais 39

3.2 PROJETO ARQUITETURAL 42

3.3 MODELAGEM DO BANCO DE DADOS 43

3.4 IMPLEMENTAÇÃO 45

3.4.1 Módulo de Recuperação e Tratamento de Dados 45

3.4.2 Módulo de Consulta 49

3.4.3 Módulo de Consulta Temática 51
3.4.4 Módulo de Visão Web 52
4 CONCLUSÃO 55
REFERÊNCIAS 57

1 INTRODUÇÃO

As pesquisas na área da recuperação da informação proporcionaram nos últimos anos o desenvolvimento de novas e melhores práticas quanto a coleta, a manutenção e o compartilhamento de grandes conjuntos de dados em diversos domínios de aplicação. Esses avanços também estão sendo aplicados no domínio de dados espaciais. Cada vez mais, o surgimento de novas tecnologias proporciona uma melhor utilização de dados espaciais, que desempenham um papel fundamental em vários domínios de aplicação, tais como planejamento urbano, monitoramento ambiental e gerenciamento de desastres, e pode desempenhar um papel estratégico em processos de tomada de decisão de várias aplicações, tanto no setor público quanto no setor privado.

O crescimento na produção e na oferta de dados espaciais fez surgir a necessidade de se desenvolver soluções que permitissem o compartilhamento e a larga utilização desses dados. Uma importante solução para esses problemas são as infraestruturas de dados espaciais (IDE). Segundo Nebert (2004), uma IDE pode ser definida como uma base relevante de tecnologias, políticas e acordos institucionais que facilitam a disponibilidade e acesso a dados espaciais, oferecendo uma base para descoberta, avaliação e aplicação para usuários e provedores de todos os níveis de governo, do setor comercial, do setor sem fins lucrativos, da academia e por cidadãos em geral.

O desenvolvimento de IDEs pode proporcionar uma série de benefícios, como a facilidade para a localização de dados espaciais, a possibilidade de reuso (e a redução da duplicação de dados) a redução nos esforços e recursos necessários para a criação e manutenção desses dados (RAJABIFARD; WILLIAMSON, 2001). Uma IDE pode ser usada por qualquer pessoa comum ou jurídica e, ao mesmo tempo, esses dois grupos podem contribuir com novos dados para a IDE, aprimorando a qualidade e a quantidade dos mesmos.

Desde a sua proposição, muitas iniciativas foram criadas para o desenvolvimento de IDEs no mundo todo. Atualmente, existem IDEs de nível federal, regional e municipal. Além disso, existem iniciativas para o desenvolvimento de infraestruturas com abrangência territorial maior. Um exemplo desse tipo de iniciativa é a INSPIRE, que visa o desenvolvimento de um IDE com dados de toda a Europa (CRAGLIA; ANNONI, 2007).

Em 2008, o Decreto n o 6.666, de 27/11/2008 do governo federal instituiu a

construção e a implantação da Infraestrutura Nacional de Dados Espaciais (INDE), que é a IDE do governo federal brasileiro (BRASIL, 2008). Então, em 2010, a Comissão Nacional de Cartografia (CONCAR) elaborou um plano de descrevendo como seria a implementação dessa infraestrutura (CONCAR, 2010). O plano de ação da INDE estabeleceu três ciclos, com datas definidas para o desenvolvimento da infraestrutura de dados espaciais, tendo o seu último ciclo se encerrando em 2020.

Atualmente, a INDE[1] encontra-se disponível para o uso de qualquer cidadão, e disponibiliza acesso aos dados espaciais produzidos por dezenas de instituições, como a Agência Nacional de Águas (ANA), a Fundação Nacional do Índio (FUNAI), o Departamento Nacional de Infraestrutura de Transportes (DNIT), a Empresa Brasileira de Pesquisa Agropecuária (EMBRAPA) e o Instituto Brasileiro do Meio Ambiente (IBAMA).

Como a infraestrutura permite o acesso a dados fornecidos por diversas organizações, é necessário uma forma de padronizar esse acesso. Uma solução para esse problema consiste na utilização dos padrões definidos pelo Open Geographic Consortium (OGC), uma organização sem fins lucrativos formada por instituições acadêmicas e privadas que tem como objetivo a especificação de padrões para o uso e compartilhamento de dados espaciais. No tocante ao compartilhamento de dados, o OGC definiu uma série de serviços web que permitem que os dados disponibilizados por diferentes organizações possam ser acessados por meio de uma interface padronizada, sem que o cliente tenha que se preocupar com os detalhes de armazenamento desses dados.

1.1 MOTIVAÇÃO

Um dos objetivos primários de uma IDE consiste em facilitar a disseminação e o compartilhamento de dados espaciais. Para que esse objetivo possa ser alcançado, as infraestruturas atuais oferecem um serviço de catálogo, que é usado tanto por provedores quanto por clientes. Provedores de dados espaciais usam o serviço de catálogo para anunciar os dados que os mesmos disponibilizam. Os clientes, por sua vez, utilizam esse serviço para localizar os dados espaciais nos quais estão interessados.

Embora a disponibilização de um serviço de catálogo facilite a recuperação dos dados disponibilizados por uma IDE, ainda não é fácil para um usuário encontrar

[1] O PORTAL BRASILEIRO DE DADOS GEOESPACIAIS - SIG BRASIL. Disponível em: <https://inde.gov.br/>. Acesso em: 08 fev. 2020.

os dados de seu interesse. Um fator que dificulta a realização dessa tarefa é que esses serviços resolvem as consultas dos usuários utilizando como base apenas os metadados usados para a descrição dos conjuntos de dados disponíveis.

Como a quantidade de metadados fornecidos por cada provedor é normalmente pequena com relação à quantidade de dados oferecidos, a qualidade das consultas acaba sendo prejudicada. Além disso, nem todos os metadados que descrevem os dados são fornecidos na hora do registro, o que dificulta a realização das consultas com restrições temáticas, espaciais e temporais.

Por exemplo, no catálogo da INDE existe um recurso cujo título é “Perigo de Escorregamento para o Aglomerado Urbano de Piracicaba. IG, Dez/2017.”. Entretanto, caso o usuário realize consulta requisitando dados sobre escorregamentos (tema) ocorridos na cidade de Piracicaba (espaço) em 2017 (tempo), o motor de busca retorna um resultado vazio, indicando que não há no catálogo qualquer registro que satisfaça as restrições definidas na consulta.

Isso acontece porque o motor de busca verifica apenas os metadados dos registros cadastrados, como são poucas informações que descrevem esses registros, nenhum registro satisfaz todas as restrições enviadas. Outra questão que também contribui para esse problema são os metadados analisados. Para resolver as consultas, o motor de busca analisa somente os metadados relacionados aos registros, isso significa que todos os metadados que descrevem os serviços e os seus tipos de feição são perdidos, consequentemente diminuindo a chance de encontrar algum resultado que satisfaça a consulta enviada.

Com o intuito de amenizar essas limitações, este trabalho de conclusão de curso propõe o desenvolvimento de uma ferramenta de busca para infraestrutura de dados espaciais. Para melhorar a qualidade do processo de recuperação dos dados, a ferramenta proposta deverá resolver as suas consultas utilizando metadados mais precisos, extraídos a partir do conteúdo dos dados.

1.2 OBJETIVOS

Esta seção aborda os objetivos do trabalho, descrevendo tanto o objetivo geral quanto os objetivos específicos que devem ser alcançados.

1.2.1 Objetivo Geral

O objetivo geral deste trabalho de conclusão de curso consiste no desenvolvimento de um motor de busca para facilitar a recuperação de dados oferecidos por meio de infraestruturas de dados espaciais.

1.2.2 Objetivos Específicos

O trabalho proposto tem ainda os seguintes objetivos específicos:

- compreender como os dados são ofertados pelas infraestruturas de dados espaciais atuais;
- entender o funcionamento das ferramentas atuais utilizadas para a recuperação de dados nessas infraestruturas;
- identificar, a partir do conteúdo dos dados publicados em infraestruturas de dados espaciais, metadados que possam ser utilizados para melhorar o processo de recuperação da informação;
- gerar um banco de dados centralizado para a recuperação de dados espaciais;
- desenvolver uma ferramenta com interface web que permita a recuperação de dados espaciais por parte de qualquer usuário;
- aplicar a ferramenta desenvolvida a uma infraestrutura de dados espacial já existente, que será usada como estudo de caso.

1.3 TRABALHOS RELACIONADOS

No trabalho de Márquez et al. (2010) duas ontologias são definidas para descrever a semântica nos dados. A primeira é definida como *Divisões políticas* e diz respeito a nomes de cidades, províncias, estados, regiões e continentes. A segunda foi nomeada Divisões não políticas, e diz respeito aos nomes de domínio de conhecimentos como hidrografia, saúde, transporte e sociedade. Além das divisões de domínios dos dados utilizando a ontologia, o trabalho ainda adiciona uma ferramenta de *Knowledge Domain Matching* usada para resolver consultas quando a ontologia não encontra resultados.

A ferramenta usa uma base de taxonomias, representando uma relação genérica e específica entre as palavras para categorizar de diferentes formas a mesma consulta e encontrar os dados.

O sistema ainda conta com uma ferramenta de rastreamento (*crawler*) que busca na internet vários serviços OGC disponíveis e armazena todos os dados ofertados por cada serviço encontrado.

Marquez et al. (2011) proporam um ranking para consultas, levando em consideração os resultados mais prováveis e menos prováveis de acordo com a busca do usuário. O sistema analisa padrões OGC, principalmente WMS, para a análise e indexação dos dados, além de verificar se a principal fonte de informação do serviço vem de uma província.

A abordagem do sistema para classificação dos resultados, conforme uma dada palavra, é feita em tempo de indexação resultando em um processamento transparente para o usuário. O sistema conta com uma indexação por pontuação semântica (*Semantic Scoring*) e pontuação geográfica (*Geograpich Scoring*), que são utilizadas no momento das busca, para avaliar qual camada é a mais relacionada ao interesse do usuário.

A pontuação semântica é responsável por garantir que, por exemplo, em uma busca por dados sobre rios, também sejam recuperados recursos descritos com palavras como *hidrografia* e vice-versa. A pontuação é construída em tempo de indexação, com um valor entre 0 e 1 de pontuação para uma dada palavra e camada.

A pontuação Geográfica também é construída em tempo de indexação, e corresponde a um valor entre 0 e 1 para a pontuação entre as camadas ascendentes e descendentes já armazenadas. Isso significa que, em uma busca feita pelo usuário por uma camada *Hotéis em Paris* o serviço que oferece camadas de *Hotéis em Paris* terá uma pontuação maior, do que outro que oferece a mesma camada na França, já que os metadados do serviço que oferece as camadas presta mais informações sobre a determinada província. O sistema ainda conta com expansão do valor indexado, que cria mais valores relacionados a mesma palavra que está sendo armazenada.

O trabalho de Corti et al. (2018) estuda a IDE WorldMap e um motor de busca chamado Hypermap, criado para tornar mais eficiente as buscas feitas à IDE. O WorldMap é uma IDE de código aberto de Harvard que permite o compartilhamento de dados espaciais e o gerenciamento dos mesmos sem necessitar de conhecimentos técnicos específicos. A ferramenta permite ainda a sobreposição de camadas de

conjuntos de dados diferentes, além de manter dados de outros tipos como vídeos, imagens entre outros (GUAN et al., 2012).

Para resolver o problema da recuperação da informação na IDE WorldMap foi criado o Hypermap Registry (Hypermap). O Hypermap é uma ferramenta que gerencia os dados de uma IDE através dos serviços OGC, disponibiliza estatísticas sobre o gerenciamento dos dados coletados e permite uma busca mais flexível pelos dados se comparado a um catálogo OGC (CHEN et al., 2011).

O Hypermap possibilita melhoria significativa na busca pelos dados. A sua API limpa facilita a elaboração de filtros sobre os dados de uma forma mais simples se comparado a XML (*Extensible Markup Language*), além de sua resposta JSON, que também facilita o consumo do resultado por parte das aplicações.

A "lematização", palavras filtradas e sinônimos também estão incluídos no Hypermap facilitando a descoberta de dados a partir de metadados textuais. A ferramenta ainda traz uma organização dos resultados a partir de sua relevância, mostrando os principais resultados no topo. O usuário pode ainda selecionar resultados que são agrupados por palavras chaves indexadas como regiões ou pelo espaço temporal, o nome para esse recurso é denominado “facets”.

1.4 CONTRIBUIÇÕES

O trabalho traz as seguintes contribuições:

- O desenvolvimento de um banco de dados centralizado para a recuperação de dados espaciais;
- O desenvolvimento de um motor de busca capaz de realizar consultas em dois níveis;
- O desenvolvimento de um motor de busca capaz de realizar consultas espaciais, temporais, temáticas e multidimensionais;
- O desenvolvimento de um motor de busca capaz de criar um ranking a partir dos resultados das consultas.

1.5 METODOLOGIA

O desenvolvimento deste trabalho de conclusão de curso contou com o desenvolvimento das seguintes atividades:

- **Estudo sobre estado da arte (A1):** nessa atividade foi realizado um estudo mais aprofundado sobre como os dados espaciais são oferecidos e recuperados atualmente. Também foram pesquisadas ferramentas desenvolvidas com o intuito de melhorar a recuperação desses dados. A fim de manter o conhecimento sempre atualizado, essa atividade foi realizada durante todo o desenvolvimento do trabalho;
- **Análise e projeto (A2):** nessa atividade foram realizadas as atividades referentes à análise e projeto da ferramenta que foi implementada. Dentre essas atividades estão o levantamento dos requisitos funcionais, a definição da arquitetura e a elaboração do esquema do banco de dados;
- **Desenvolvimento do módulo de coleta de dados (A3):** nessa atividade foi implementado um módulo responsável por interagir com o serviço de catálogo da infraestrutura a ser usada como estudo de caso. Esse módulo é responsável por coletar os metadados de cada registro descrito no serviço de catálogo. Além disso, o módulo acessa os serviços descritos em cada registro para obter informações mais detalhadas sobre os dados disponíveis;
- **Desenvolvimento do módulo de extração de metadados (A4):** nessa atividade foi implementado um módulo responsável por processar as informações obtidas por meio do módulo de coleta de dados. Esse módulo é responsável por extrair os metadados que foram usados para melhorar o processo de recuperação da informação;
- **Desenvolvimento do motor de busca (A5):** nesta etapa foi implementado o motor de busca, que é responsável por receber as consultas dos usuários e resolvê-las com base nas informações extraídas pelo módulo de extração de metadados;
- **Elaboração do documento final de TCC (A6):** nesta etapa foi elaborado o documento de TCC. Ela também foi realizada ao longo de todo o desenvolvimento do trabalho.

1.6 ESTRUTURA E ORGANIZAÇÃO DO DOCUMENTO

O restante do documento é organizado em três capítulos:

- O Capítulo 2 descreve a fundamentação teórica, discutindo os principais conceitos básicos necessários para o entendimento da área de estudo e proposta do trabalho;
- O Capítulo 3 descreve a solução proposta por este trabalho, detalhando cada módulo presente na arquitetura desenvolvida;
- Finalmente, o Capítulo 4 apresenta as considerações finais e as possíveis contribuições futuras para o trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos, tecnologias e padrões que são necessários para o entendimento do presente trabalho, assim como da solução proposta. Inicialmente, o capítulo fala sobre a tecnologia de web services e a arquitetura orientada a serviços. Em seguida, são descritos os padrões definidos pelo OGC, que são utilizados para a implementação deste trabalho. Por fim, é abordado o funcionamento da ferramenta de busca textual utilizada neste trabalho.

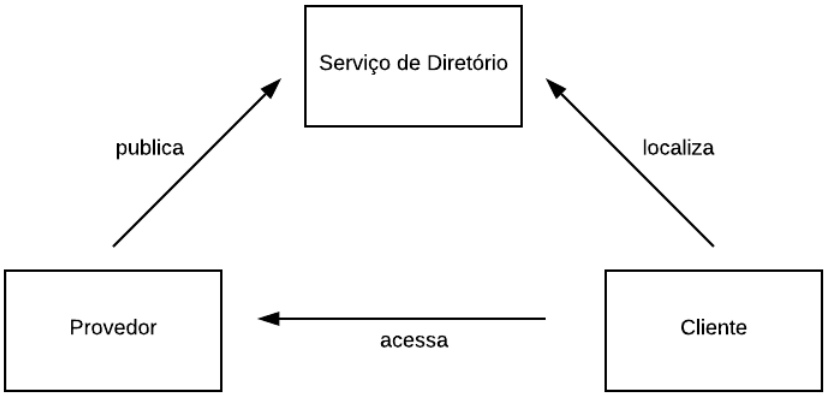
2.1 WEB SERVICE

Desde as últimas décadas, o padrão de arquitetura de software orientada a serviços (do inglês *Service Oriented Architecture* - SOA) (ERL, 2016) tem sido bastante utilizado no desenvolvimento de sistemas. Nesse tipo de arquitetura, as funcionalidades de um sistema são fornecidas como uma série de serviços que podem ser invocados por outros sistemas através da rede. Uma grande vantagem oferecida por uma arquitetura SOA é que os clientes que invocam os serviços não precisam conhecer os seus detalhes de implementação, aumentando, assim, o encapsulamento e a interoperabilidade entre diferentes aplicações.

A Figura 1 ilustra o funcionamento de uma arquitetura SOA, destacando os seus três componentes. O serviço de diretório corresponde a uma aplicação que é utilizada para a descoberta dos serviços disponíveis. Os provedores representam os usuários que tornam os seus serviços disponíveis através da *Internet*. Esses usuários usam o serviço de diretório para anunciar os serviços oferecidos, provendo, para cada serviço, informações como a *uniform resource locator* (URL) de acesso e os parâmetros requeridos para a sua execução. Os clientes representam os usuários que consomem os serviços disponibilizados. Esses clientes, por sua vez, utilizam o serviço de diretório para localizar o serviço de seu interesse. Depois de localizar o serviço, o cliente pode invocá-lo diretamente usando a URL na qual este serviço encontra-se disponível.

Uma nova implementação desta arquitetura, chamada *REST* (*Representational State Transfer*), foi proposta por Roy Fielding Fielding e Taylor (2000). O maior objetivo de seus criadores era garantir uma comunicação entre componentes através da rede com alta escalabilidade, segurança, interfaces genéricas e encapsulamento de sistemas legados.

Figura 1 – Arquitetura SOA



Fonte: Elaborado pelo autor

Desde a sua proposição, a arquitetura SOA tem sido amplamente usada em aplicações do domínio geoespacial. De forma a padronizar o acesso a dados espaciais, o OGC[1], consórcio que é formado por uma série de instituições privadas, instituições acadêmicas e outras sem fins lucrativos, tem definido diversos padrões para o uso de dados geoespaciais a partir de serviços web. Os serviços definidos pelo OGC utilizam a arquitetura *REST* para disponibilizar o acesso a dados espaciais através de chamadas HTTP (*Hypertext Transfer Protocol*).

2.2 SERVIÇOS OGC

Esta seção apresenta os padrões definidos pelo OGC que foram utilizados no desenvolvimento deste trabalho. Para cada um deles, são descritos o seu funcionamento e as suas operações principais.

2.2.1 Web Map Service (WMS)

Um dos padrões de serviço que o OGC criou para padronizar o acesso aos dados espaciais foi o *Web Map Service* (WMS) (BEAUJARDIERE, 2006), que tem como principal característica a recuperação de camadas vetoriais. Os dados são normalmente retornados pelo serviço já em um formato de apresentação, geralmente em um formato de imagem (embora o serviço dê suporte a outros tipos de formatos). A lista de camadas

[1] WELCOME to The Open Geospatial Consortium. Disponível em: <https://www.opengeospatial.org/>. Acesso em: 24 nov. 2019.

disponíveis e os formatos de apresentação disponíveis em um serviço WMS podem ser consultados a partir de uma requisição para o método *GetCapabilities*, que retorna um documento com as informações gerais do serviço, como uma descrição, contato, tipos de retorno das camadas, métodos HTTP disponíveis e camadas de dados oferecidas.

A Figura 2 mostra uma requisição para o método *GetCapabilities* de um serviço WMS oferecido no Geoportal do Exército Brasileiro. O serviço tem dados sobre limites municipais, estaduais, regiões com o mapeamento em andamento, entre outros. Nela, pode-se notar que, para realizar a requisição, é necessário informar alguns parâmetros. O parâmetro *service* define o tipo de serviço que se deseja consultar, neste caso, um serviço WMS. O parâmetro *version* define a versão do serviço que deve ser utilizada. E, por último, deve ser informado o tipo de requisição desejada. Nesse exemplo o parâmetro request define que se deseja consultar as capacidades do serviço informando o valor *GetCapabilities*.

Figura 2 – Exemplo de requisição *GetCapabilities*

```
1  http://bdgex.eb.mil.br/cgi-bin/geoportal?
2  service=wms&
3  version=1.1.1&
4  request=GetCapabilities
```

Fonte: Elaborado pelo autor

A resposta da requisição é um documento XML contendo as informações do serviço. Esse documento é dividido em várias seções. A primeira seção traz informações sobre o próprio serviço, como o nome, o título, a URL e as informações de contato, conforme é mostrado na Figura 3.

Figura 3 – Informações sobre o WMS buscado

```
11  <Service>
12    <Name>OGC:WMS</Name>
13    <Title>geoportal</Title>
14    <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
15     xlink:href="http://bdgex.eb.mil.br/cgi-bin/geoportal?"/>
16    <ContactInformation>
17    </ContactInformation>
18  </Service>
```

Fonte: Elaborado pelo autor

A segunda seção de informações descreve as operações disponíveis, como mostra a Figura 4. Para cada operação, são informados o tipo de retorno que pode ser

escolhido e as Uniform Resource Identifier (URI)s de acesso para cada método HTTP que pode ser usado para realizar a requisição.

Figura 4 – Seção de operações disponíveis em um WMS

```
<GetFeatureInfo>
    <Format>text/html</Format>
    <Format>application/vnd.ogc.gml</Format>
    <Format>text/plain</Format>
    <DCPType>
        <HTTP>
            <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
             xlink:href="http://bdgex.eb.mil.br/cgi-bin/geoportal?"/>
             </Get>
            <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
            xlink:href="http://bdgex.eb.mil.br/cgi-bin/geoportal?"/>
            </Post>
        </HTTP>
    </DCPType>
</GetFeatureInfo>
```

Fonte: Elaborado pelo autor

Logo em seguida, na Figura 5, pode-se perceber uma das seções mais importantes, que detalha os metadados das camadas de dados (tipos de feição) oferecidas pelo serviço. A figura mostra a descrição de uma das camadas oferecidas. Para cada camada, são retornadas informações como: o nome, que atua como o identificador da camada no serviço, o título, o sistema de coordenadas geográficas, a extensão geográfica e a escala mínima e máxima.

Figura 5 – Seção de camada de um serviço WMS

```
<Layer queryable="1" opaque="0" cascaded="0">
    <Name>Articulacao_Projeto_Amapa</Name>
    <Title>Articulacao_Projeto_Amapa</Title>
    <SRS>EPSG:4326</SRS>
    <LatLonBoundingBox minx="-125" miny="-55" maxx="0" maxy="15" />
    <BoundingBox SRS="EPSG:4326"
                minx="-125" miny="-55" maxx="0" maxy="15" />
    <ScaleHint min="0" max="748354272.644456" />
</Layer>
```

Fonte: Elaborado pelo autor

Outra operação oferecida pelo WMS é a *GetMap*, que pode ser usada para recuperar uma ou mais camadas disponibilizadas pelo serviço. A Figura 6 mostra um exemplo de requisição para a operação *GetMap*, na qual é recuperada a camada

municipios. Alguns parâmetros devem ser informados na requisição para que a operação tenha êxito. Os parâmetros são: o serviço que será utilizado, a operação (nesse caso *GetMap*), a versão do serviço, os nomes das camadas de interesse, a extensão geográfica, o sistema de coordenadas, a largura da imagem a ser retornada, a altura e formato de retorno.

Figura 6 – Requisição *GetMap* para um serviço WMS

```
1  http://bdgex.eb.mil.br/cgi-bin/geoportal?
2  service=WMS&
3  request=GetMap&
4  version=1.0.0&
5  layers=municipios&
6  bbox=-125,-55,0,15&
7  srs=EPSG:4326&
8  width=1000&
9  height=1000&
10 format=image/png
```

Fonte: Elaborado pelo autor

O resultado da requisição *GetMap* é normalmente uma imagem, no formato passado no parâmetro *format*, com as camadas solicitadas no parâmetro *layers*. A Figura 7 mostra a imagem retornada a partir da URL mostrada na Figura 6.

Figura 7 – Resultado da requisição *GetMap* do exemplo

Fonte: Elaborado pelo autor

Além das operações já mostradas, o serviço WMS também oferece a operação *GetFeatureInfo*, que recupera informações adicionais como a própria geometria da camada, *GetLegendGraphic*, que recupera uma legenda gerada para o mapa, e *DescribeLayer*, que traz informações adicionais sobre a camada independente do serviço.

2.2.2 Web Feature Service (WFS)

Outro padrão criado pelo OGC foi o *Web Feature Service* (WFS) Vretanos et al. (2016), que permite o acesso aos dados espaciais na forma bruta . Isso significa que os dados retornados pelo serviço podem ser processados pela aplicação que fez a invocação. O resultado de uma requisição para um serviço WFS pode retornar arquivos em formatos como *shapefile* ou *Geography Markup Language* (GML), que corresponde a uma linguagem baseada em XML, também proposta pelo OGC, para a descrição de dados espaciais. Todos os tipos de feição (*feature types*) disponibilizados por um serviço WFS podem ser consultados a partir de uma requisição REST, para

URI do serviço invocando a operação *GetCapabilities*, que retorna um documento XML contendo informações que descrevem o serviço, suas operações e os tipos de feição que se encontram disponíveis.

A figura 8 mostra um exemplo de uma requisição para a operação *GetCapabilities* de um serviço WFS disponibilizado no Geoportal do Exército Brasileiro. Alguns parâmetros precisam ser informados para que a requisição tenha êxito, como o serviço que deverá ser acessado, a operação a ser invocada e a versão do serviço.

Figura 8 – Exemplo de requisição *GetCapabilities* para um serviço WFS

```
1  http://bdgex.eb.mil.br/cgi-bin/mapaindice?
2  service=WFS&
3  request=GetCapabilities&
4  version=1.0.0
```

Fonte: Elaborado pelo autor

A resposta da requisição é um documento XML, que é dividido em várias seções. A primeira seção é referente à descrição do serviço. A Figura 9 mostra essa seção para o documento retornado, por meio da URL mostrada na Figura 8. As informações mostradas na figura representam, respectivamente, o nome do serviço, o título e os recursos disponíveis. Dependendo do serviço que está sendo acessado, outras informações podem ser oferecidas, como as palavras-chave e a descrição do mesmo.

Figura 9 – Seção que descreve o serviço WFS

```
12  <Service>
13    <Name>MapServer WFS</Name>
14    <Title>Mapas_Indice_do_BDGEx</Title>
15    <OnlineResource>
16      http://www.geoportal.eb.mil.br/cgi-bin/mapaindice?
17    </OnlineResource>
18  </Service>
```

Fonte: Elaborado pelo autor

A segunda seção do documento descreve as informações a respeito das operações que são fornecidas pelo serviço. A Figura 10 apresenta a descrição da operação *GetCapabilities*. Para cada operação, são retornadas informações como os formatos de retorno disponíveis e as URLs de acesso para cada método HTTP suportado pela operação.

Figura 10 – Seção que descreve as requisições disponíveis do serviço WFS

```
20  <Request>
21    <GetCapabilities>
22      <Format>application/vnd.ogc.wms_xml</Format>
23      <DCPType>
24        <HTTP>
25          <Get><OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
26           xlink:href="http://bdgex.eb.mil.br/cgi-bin/geoportal?"/></Get>
27          <Post><OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
28           xlink:href="http://bdgex.eb.mil.br/cgi-bin/geoportal?"/></Post>
29        </HTTP>
30      </DCPType>
31    </GetCapabilities>
```

Fonte: Elaborado pelo autor

Para cada operação, pode existir ainda uma seção indicando as exceções que o serviço pode retornar. A Figura 11 mostra um exemplo de descrição dessas exceções.

Figura 11 – Seção que descreve as exceções do serviço WFS

```
91   <Exception>
92     <Format>application/vnd.ogc.se_xml</Format>
93     <Format>application/vnd.ogc.se_inimage</Format>
94     <Format>application/vnd.ogc.se_blank</Format>
95   </Exception>
```

Fonte: Elaborado pelo autor

Uma das seções mais importantes do documento retornado pela operação *GetCapabilities* de um serviço WFS descreve as camadas (tipos de feição) disponíveis. Essa seção descreve todas as camadas oferecidas pelo serviço. A Figura 12 mostra a descrição de uma das camadas descritas no documento retornado pela URL da Figura 8. Nela, percebe-se que, para cada camada, são informados o nome, que atua como o identificador da camada no serviço, o título, o sistema de coordenadas geográficas, e a extensão espacial.

Figura 12 – Seção de camada de um serviço WFS

```
116     <Layer queryable="1" opaque="0" cascaded="0">
117         <Name>basebrasil</Name>
118         <Title>basebrasil</Title>
119         <SRS>EPSG:4326</SRS>
120         <LatLonBoundingBox minx="-125" miny="-55" maxx="0" maxy="15" />
121         <BoundingBox SRS="EPSG:4326"
122                       minx="-125" miny="-55" maxx="0" maxy="15" />
123     </Layer>
```

Fonte: Elaborado pelo autor

A operação usada para se obter os dados fontes de uma camada é a operação *GetFeature*, que traz todas as informações direto da base de dados para um tratamento do lado do cliente. Mas, para compreender quais dados e tipos serão retornados a respeito de uma única camada, é necessário antes invocar a operação *DescribeFeatureType*, que retorna um documento XML com informações a respeito do esquema da camada, descrevendo os seus atributos com os seus respectivos tipos.

A Figura 13 mostra um exemplo de requisição para a operação *DescribeFeatureType* da camada descrita na Figura. Para se invocar esta operação, deve-se passar como parâmetros o serviço, o tipo de requisição, a versão do serviço e os nomes das camadas desejadas.

Figura 13 – Exemplo de requisição *DescribeFeatureType* para um serviço WFS

```
1  http://bdgex.eb.mil.br/cgi-bin/mapaindice?
2  service=WFS&
3  request=DescribeFeatureType&
4  version=1.0.0&
5  typenames=basebrasil
```

Fonte: Elaborado pelo autor

O resultado da requisição é um documento XML contendo os elementos que descrevem o nome de cada atributo e o seu tipo. A Figura 14 mostra a descrição da camada requisitada. Nela, pode-se perceber que cada instância da camada é descrita por atributos como *gid, nome, temvetorial, datavetorial, estado*, entre outros.

Figura 14 – Resultado da requisição *DescribeFeatureType* para um serviço WFS

```
<element name="msGeometry" type="gml:GeometryPropertyType" minOccurs="0" maxOccurs="1"/>
<element name="gid" type="string"/>
<element name="inom" type="string"/>
<element name="asc_" type="string"/>
<element name="mi" type="string"/>
<element name="nome" type="string"/>
<element name="temvetorial" type="string"/>
<element name="temmatricial" type="string"/>
<element name="datavetorial" type="string"/>
<element name="datamatricial" type="string"/>
<element name="uuidvetorial" type="string"/>
<element name="uuidmatricial" type="string"/>
<element name="temmds" type="string"/>
<element name="temortoimagem" type="string"/>
<element name="datamds" type="string"/>
<element name="dataortoimagem" type="string"/>
<element name="uuidmds" type="string"/>
<element name="uuidortoimagem" type="string"/>
<element name="estado" type="string"/>
```

Fonte: Elaborado pelo autor

Depois que o esquema da camada é conhecido, o cliente pode usar a operação *GetFeature* para recuperar os dados da camada selecionada. A Figura 15 mostra um exemplo de uma requisição para essa operação. Para invocá-la, são informados os seguintes parâmetros: o serviço a ser invocado, a operação, a versão e os nomes das camadas que devem ser recuperadas.

Figura 15 – Exemplo de requisição *GetFeature* para um serviço WFS

```
http://bdgex.eb.mil.br/cgi-bin/geoportal?
service=WFS&
request=GetFeature&
version=1.0.0&
TypeName=Articulacao_Projeto_Amapa
```

Fonte: Elaborado pelo autor

A Figura 16 mostra parte da resposta em GML da requisição à feature Articulacao_Projeto_Amapa. Nela, percebe-se um campo chamado "*Articulacao_Projeto_Amapa*" que compreende todas as informações da feição. Entre as informações estão atributos como *Objetctid*, que representa a identificação da função, *Polygon*, que descreve a geometria do objeto, e *Escala*, que descreve a escala usada para a geração do mapa.

Figura 16 – Parte da resposta da requisição *GetFeature*

```
<gml:boundedBy>
  <gml:Box srsName="EPSG:4326">
    <gml:coordinates>-55.000000,-1.250000 -49.750000,4.500000</gml:coordinates>
  </gml:Box>
</gml:boundedBy>
<gml:featureMember>
  <ms:Articulacao_Projeto_Amapa>
    <gml:boundedBy>
      <gml:Box srsName="EPSG:4326">
        <gml:coordinates>-52.250000,-0.500000 -52.125000,-0.375000</gml:coordinates>
      </gml:Box>
    </gml:boundedBy>
    <ms:msGeometry>
      <gml:Polygon srsName="EPSG:4326">
        <gml:outerBoundaryIs>
          <gml:LinearRing>
            <gml:coordinates>-52.250000,-0.375000
             -52.208330,-0.375000 -52.166670,-0.375000
             -52.125000,-0.375000 -52.125000,-0.416670
             -52.125000,-0.458330 -52.125000,-0.500000
             -52.166670,-0.500000 -52.208330,-0.500000
             -52.250000,-0.500000 -52.250000,-0.458330
             -52.250000,-0.416670 -52.250000,-0.375000 </gml:coordinates>
          </gml:LinearRing>
        </gml:outerBoundaryIs>
      </gml:Polygon>
    </ms:msGeometry>
    <ms:OBJECTID>8642</ms:OBJECTID>
    <ms:INOM>SA-22-V-B-I-4-SO</ms:INOM>
    <ms:MI_1>0284-4-SO</ms:MI_1>
    <ms:Bloco>B</ms:Bloco>
    <ms:Escala>25000</ms:Escala>
  </ms:Articulacao_Projeto_Amapa>
</gml:featureMember>
```

Fonte: Elaborado pelo autor

2.2.3 Serviço de Catálogo (CSW)

Para facilitar a localização dos dados espaciais que se encontram disponíveis, o OGC definiu o *Catalogue Service for the Web* (CSW) Nebert et al. (2007), que atua como um serviço de catálogo de metadados, desempenhando o papel do serviço de diretório de uma arquitetura SOA. Os metadados correspondem a um conjunto de informações a respeito dos dados espaciais, que permitem ao cliente localizar os dados do seu interesse. A especificação do serviço CSW define uma operação chamada *GetRecords*, que permite ao cliente obter uma descrição de todos os registros (*records*)

ofertados.

A Figura 17 mostra um exemplo de uma requisição *GetRecords* para a URI do serviço de catálogo da INDE para obter os registros disponíveis. Nela, pode-se perceber que, logo após a URI do serviço de catálogo, devem ser passados alguns parâmetros que definem, respectivamente, as seguintes propriedades: o tipo de serviço que se deseja acessar, o tipo de requisição, a versão do serviço, o nível de descrição do registro e o tipo de resultado esperado.

Figura 17 – Exemplo de requisição *GetRecords* para um serviço CSW

```
1  http://www.metadados.inde.gov.br/geonetwork/srv/por/csw?
2  service=CSW&
3  request=GetRecords&
4  version=2.0.2&
5  elementSetName=full&
6  resultType=results
```

Fonte: Elaborado pelo autor

A Figura 18 mostra o documento XML retornado pelo serviço de catálogo como resultado da requisição. Como o parâmetro *elementSetName* da requisição foi usado o valor *full*, o serviço retorna a descrição completa de cada registro. Na figura, pode-se perceber a descrição de um dos registros retornados. Para esse registro, é possível visualizar algumas informações, como o identificador do recurso, a data, o título, o tema, a descrição e o idioma. A Figura 19 mostra mais algumas informações acerca do registro, tais como a sua extensão espacial e as URIs, a partir das quais o recurso pode ser acessado.

Figura 18 – Descrição do registro como resultado da requisição *GetRecords*

```
2    <csw:Record xmlns:dc="http://purl.org/dc/elements/1.1/"
3        xmlns:ows="http://www.opengis.net/ows"
4        xmlns:geonet="http://www.fao.org/geonetwork"
5        xmlns:dct="http://purl.org/dc/terms/">
6        <dc:identifier>4f127382-ca65-4104-a76a-dfe974f3f492</dc:identifi
7        <dc:date>2011-06-26T09:45:23</dc:date>
8        <dc:title>CARTA IMAGEM AERONÁUTICA DE PILOTAGEM - CIAP 9188 - SE
9        <dc:subject>Carta Aeronáutica de Pilotagem</dc:subject>
10       <dc:subject>transportation</dc:subject>
11       <dct:abstract>A Carta-Imagem Aeronáutica de Pilotagem CIAP - 1:2
         em referências visuais do terreno.…
13       <dc:description>A Carta-Imagem Aeronáutica de Pilotagem CIAP - 1
         em referências visuais do terreno.…
15       <dc:language>por</dc:language>
```

Fonte: Elaborado pelo autor

Figura 19 – URI disponíveis de um *Record* CSW

```
<ows:BoundingBox crs="urn:ogc:def:crs:::WGS 84">
    <ows:LowerCorner>-64,5 -7</ows:LowerCorner>
    <ows:UpperCorner>-66 -6</ows:UpperCorner>
</ows:BoundingBox>
<dc:URI protocol="image/png" name="thumbnail">resources.get?id=2000&amp;fname=9188_s.png&amp;access=public</dc:URI>
<dc:URI protocol="image/png" name="large_thumbnail">resources.get?id=2000&amp;fname=9188.png&amp;access=public</dc:URI>
<dc:URI protocol="WWW:LINK-1.0-http--link" name="">http://www.aisweb.aer.mil.br/arquivos/cartas/visuais/ciap/9188_fev03
<dc:URI protocol="WWW:DOWNLOAD-1.0-http--download" name="">http://200.255.9.14:80/geonetwork/srv/en/resources.get?id=20
<dc:URI protocol="OGC:WMS-1.1.1-http-get-map" />
```

Fonte: Elaborado pelo autor

Além da operação *GetRecords*, o CSW oferece outras operações, como *GetCapabilities*, que retorna uma série de metadados sobre o serviço de catálogo, como, por exemplo, quais operações são suportadas, o título do catálogo, sua descrição, palavras-chaves, a organização que provê o catálogo, o contato da organização e possibilidades de filtro de registros. Os serviços de catálogo também oferecem as operações *GetRecordById*, *GetDomain* e *DescribeRecord*.

2.3 APACHE SOLR

O Apache *Solr* é implementado sobre outra ferramenta, que é o Apache Lucene. O *Apache Lucene* é uma biblioteca de busca que fornece um conjunto de funcionalidades via API para consultas textuais, indexação, *ranking*, análise de linguagem e outros (SMILEY et al., 2015). Porém, desenvolver um aplicativo que se comunique diretamente com o Apache Lucene e tenha alto desempenho é uma tarefa complexa. Além disso, a API só dá suporte à linguagem de programação java, o que dificulta a sua utilização em várias aplicações.

O *Apache Solr* foi desenvolvido como uma solução para resolver essas limitações. Ele é um servidor de busca construído sobre o *Apache Lucene* adicionando algumas funcionalidades como a disponibilização de uma API que permite a comunicação com outros sistemas através de XML ou JSON, arquivos de configuração de esquema para indexação e análise, uma interface web de administração da ferramenta, suporte a busca distribuída, replicação de índices e configuração de cluster utilizando o Zookeeper para coordenação (BIAŁECKI et al., 2012; GRAINGER; POTTER, 2014). O *Solr* disponibiliza um web service capaz de realizar operações de consulta e indexação de documentos, provendo alto desempenho e facilitando a comunicação com aplicações desenvolvidas em outras linguagens de programação.

Para que o *Solr* funcione é necessário fazer algumas configurações que defi-

nem a forma como os dados serão armazenados e recuperados. Essas configurações são vinculadas a um conceito chamado *core*. Um *core* é uma instância de indexação de todas as configurações feitas a partir dos arquivos de configurações. Assim, uma vez criada a aplicação *Solr*, vários cores podem ser criados com diferentes configurações e diferentes dados armazenados. Para cada *core* existem dois arquivos principais, o arquivo de esquema (schema.xml) e o arquivo de configuração do *Solr* (solrconfig.xml).

Como mostra a Figura 20, o arquivo de esquema define os campos e suas configurações para os documentos que serão indexados e salvos no *core*. Ao se analisar a figura, percebe-se que existem os campos *id* e *service_metadata*. Para esses dois campos alguns atributos são informados. No campo id: a) *type* define que o tipo do valor armazenado é uma *string*; b) *indexed* define que os documentos podem ser recuperados a partir de consultas com esse campo; c) *stored* define que o valor do campo pode ser recuperado em consultas e; d) *required* define que o campo é obrigatório.

O campo *service_metadata* tem os mesmos atributos do campo *id*. Nele: a) *type* é definido com o valor *text_general*, um tipo dinâmico criado especificamente para este trabalho, para indicar os dados textuais que serão usados para consulta; c) *termVectors* e *termPositions* informam que o servidor deve manter todo o vetor do documento e adicionar dados sobre posições, ocorrências entre outros. Esses valores permitem ao servidor fazer buscas mais eficientes, porém aumentando o tempo de indexação e espaço consumido para o armazenamento dos índices.

Figura 20 – Principal seção do arquivo de esquema do *Solr*

```
4    <field name="id" type="string" indexed="true"
5    stored="true" required="true" />
6    <field name="service_metadata" type="text_general"
7    indexed="true" stored="true" termVectors="true"
8    termPositions="true"/>
```

Fonte: Elaborado pelo autor

O *Solr* permite a criação de tipos de campos para definir como os dados serão indexados ou recuperados. A Figura 21 mostra um exemplo de um tipo de campo que foi criado para o desenvolvimento deste trabalho. Esse campo define o tipo de campo *text_general* que foi usado na Figura 20.

Primeiro algumas configurações gerais são definidas, como o nome do tipo de

campo *text_general*. Em seguida são definidos dois *analyzers* diferentes. O primeiro deles cuidará da indexação, com seu contexto *index*, enquanto que o segundo será responsável pela consulta, com o contexto *query*.

Os *analyzers* são responsáveis por ler os textos de entrada e transformá-los em um fluxo de *tokens*, que são usados para armazenar ou consultar, dependendo de seu contexto. Porém, nesse caso, o parâmetro *class* não foi informado ao *analyzer*. Isso significa que a geração dos *tokens* será responsabilidade do conjunto de elementos que estão aninhados ao *analyzer*, ou seja, o *tokenizer* e os filtros.

Como o *analyzer* não fará a geração dos *tokens*, o *tokenizer* será o responsável por fazer esse trabalho e repassar o resultado para o fluxo. Em seguida, os *tokens* passam por uma série de filtros que adicionam, modificam ou removem os *tokens*.

A principal diferença entre um *filter* e um *tokenizer* é a entrada que eles recebem. Enquanto o *tokenizer* recebe um texto e retorna os *tokens* gerados a partir do mesmo, o *filter* recebe um *token* e o modifica, passando o mesmo adiante ou não. No final, o *token* resultante do último filtro será o valor armazenado no índice, e, consequentemente, utilizado durante a resolução das consultas.

Figura 21 – Tipo de campo *text_general*

```
53    <fieldType name="text_general" class="solr.TextField"
54      positionIncrementGap="100" multiValued="true">
55      <analyzer type="index">
56        <tokenizer class="solr.LetterTokenizerFactory"/>
57        <filter class="solr.StopFilterFactory" ignoreCase="true"
58          words="./lang/stopwords_pt.txt" />
59        <filter class="solr.LowerCaseFilterFactory"/>
60        <filter class="solr.ASCIIFoldingFilterFactory"/>
61      </analyzer>
62      <analyzer type="query">
63        <tokenizer class="solr.OpenNLPTokenizerFactory"
64          sentenceModel="lemmatizer/pt-sent.bin"
65          tokenizerModel="lemmatizer/pt-token.bin"/>
66        <filter class="solr.LowerCaseFilterFactory"/>
67        <filter class="solr.OpenNLPPOSFilterFactory"
68          posTaggerModel="lemmatizer/pt-pos-maxent.bin"/>
69        <filter class="solr.OpenNLPLemmatizerFilterFactory"
70          dictionary="lemmatizer/lemmas-pt.dict" />
71        <filter class="solr.ASCIIFoldingFilterFactory"/>
72      </analyzer>
73    </fieldType>
```

Fonte: Elaborado pelo autor

O principal filtro para o contexto de indexação, que pode ser observado ainda na Figura 21, é o filtro que remove os *tokens* a partir de uma lista de palavras, que é o filtro da classe “*solr.StopFilterFactory*”. Isso ajuda a evitar palavras que não são relevantes para a resolução de consultas, como artigos, pronomes e outras.

Para o contexto de consulta, o filtro que se destaca para garantir maior eficiência à ferramenta é o filtro da classe “*solr.OpenNLPPOSFilterFactory*”, que é responsável por criar os "*lemas*"das palavras de entrada.

A “*lematização*” é o processo usado para se extrair a raiz de uma palavra, porém o objetivo é apenas alterar o sufixo para obter a palavra normalizada (PLISSON et al., 2004). Normalizar as palavras é uma etapa crucial de um pré-processamento, que ajuda aplicações que realizam a recuperação de informações, principalmente em linguagens complexas e com muitas variações, a encontrar resultados com maior relevância (MANJAVACAS et al., 2019). Esse processo melhora as informações de entrada para as restrições temáticas, já que transforma os termos como: i) “criou” em “criar”; ii) “rodovias” em “rodovia”; iii) “edificações” em “edificação” e; “barragens” em “barragem”.

O *Solr* fornece uma API REST para realizar todas as suas operações. A Figura 22 mostra um exemplo de URL utilizada para realizar uma consulta aos documentos de metadados do serviço.

Figura 22 – Consulta via API REST do *Solr*

```
1  http://localhost:8983/solr/inde/select?
2  fl=*,score&q=service_metadata:preservação&rows=10
```

Fonte: Elaborado pelo autor

Como mostra a Figura 22, alguns parâmetros são informados para a consulta. O parâmetro *fl* define quais campos devem ser retornados. Nesse caso, o valor * significa que todos os campos definidos no esquema devem ser retornados. Já o campo *score* é criado em tempo de consulta pelo *Solr* para retornar o *ranking* de cada documento recuperado. O parâmetro *q* define a consulta que será executada. Primeiro é informado o campo que deve ser analisado, nesse caso o campo *service_metadata*. Após os dois pontos, o texto usado como critério para a consulta é informado.

No exemplo mostrado na figura, será realizada uma consulta por documentos que falem sobre preservação. O último parâmetro, chamado *rows*, informa uma quantidade máxima de documentos que deve ser retornada. O *Solr* ainda conta com muitos

outros parâmetros que ajudam na forma de resposta e filtro da consulta. Uma descrição detalhada desses parâmetros pode ser encontrada na documentação do solr[2].

A Figura 23 mostra o resultado da consulta executada a partir da figura 22. Alguns metadados da consulta, como o número de documentos encontrados, a pontuação máxima, o tempo e os parâmetros utilizados são retornados, além dos próprios documentos.

Figura 23 – Resultado da consulta feita ao *Solr*


```
{
  "responseHeader":{
    "status":0,
    "QTime":491,
    "params":{
      "q":"service_metadata:preservação",
      "fl":"*,score",
      "rows":"10",
      "_":"1607447700475"}},
  "response":{"numFound":1,"start":0,"maxScore":0.18047042,"docs":[
      {
        "id":"438ccd08-a994-4dd9-bf2f-726aad05b638",
        "_version_":1685529183121309696,
        "service_metadata":[" ÁREA DE PONDERAÇÃO DO CENSO 2010 BELO HORIZONT
        "score":0.18047042}]
  }}
```


Fonte: Elaborado pelo autor

[2] Parâmetros comuns de consulta. Disponível em: <https://lucene.apache.org/solr/guide/8_5/common-query-parameters.html>. Acesso em: 10 jan. 2021.

3 UM MOTOR DE BUSCA PARA INFRAESTRUTURAS DE DADOS ESPACIAIS

Este capítulo descreve a implementação da ferramenta proposta por este TCC. O capítulo é dividido em quatro seções. A seção 3.1 descreve o processo de análise do sistema proposto. A seção 3.2 descreve o projeto arquitetural elaborado para cumprir com os requisitos levantados. A seção 3.3 apresenta a modelagem do banco de dados elaborada para o armazenamento dos dados recuperados. Finalmente, a seção 3.4 específica detalhes de implementação dos módulos que compõem a arquitetura.

3.1 ANÁLISE

Este tópico apresenta o processo de análise do sistema proposto, descrevendo os seus requisitos funcionais e os *stakeholders*.

3.1.1 *Stakeholders*

Todas as pessoas interessadas e que influenciam diretamente ou indiretamente nos requisitos do sistema devem fazer parte da atividade de elicitação de requisitos. O usuário final do sistema também pode fazer parte desse grupo de pessoas, que são chamadas de *stakeholders* e devem ter total atenção para que todos os requisitos, serviços e restrições do sistema sejam descritos de forma correta (SOMMERVILLE, 2011).

Os *stakeholders* envolvidos como usuário final do sistema proposto são representados por qualquer usuário que deseje obter dados geoespaciais de uma IDE, de maneira a trazer resultados com maior relevância para as pesquisas do mesmo.

3.1.2 Requisitos Funcionais

Durante a etapa de análise da ferramenta foram definidos os seguintes requisitos funcionais para a implementação do sistema:

- **Resolução de consultas em dois níveis (R1):** a ferramenta proposta neste TCC

deve resolver consultas tanto em serviços, quanto em nível de tipo de feição. No primeiro tipo de consulta, devem ser recuperados todos os serviços que ofereçam pelo menos um tipo de feição, que satisfaça os critérios definidos na consulta. No segundo tipo de consulta, a ferramenta deve selecionar todos os tipos de feição, que satisfaçam os critérios definidos na consulta do usuário;

- **Resolução de consultas espaciais (R2):** a ferramenta deve ser capaz de resolver consultas com restrições espaciais. Nesse tipo de consulta, o usuário deve fornecer o nome de um lugar ou selecionar uma região geográfica de seu interesse, e a ferramenta deve retornar todos os recursos, cujo conteúdo tenha algum dado sobre a localização desejada;
- **Resolução de consultas temporais (R3):** a ferramenta deve ser capaz de resolver consultas com restrições temporais. Nesse tipo de consulta, o usuário deve fornecer um intervalo de tempo de seu interesse, e a ferramenta deve retornar todos os recursos, cujo conteúdo tenha algum dado sobre a o período desejado;
- **Resolução de consultas temáticas (R4):** a ferramenta deve ser capaz de resolver consultas com restrições temáticas. Nesse tipo de consulta, o usuário deve fornecer uma ou mais palavras-chaves, correspondentes aos temas de seu interesse, e a ferramenta deve retornar todos os recursos cujo conteúdo tenha algum dado sobre o tema desejado;
- **Resolução de consultas multidimensionais (R5):**a ferramenta deve ser capaz de resolver consultas com mais de um tipo de restrição. Nesse tipo de consulta, o usuário deve fornecer a região espacial, o período de tempo e o tema de seu interesse, e a ferramenta deve selecionar todos os recursos que tenham algum dado que satisfaça todas as restrições especificadas.

O Quadro 1 mostra as características deste trabalho com os trabalhos relacionados abordados.

Quadro 1 – características deste trabalho com os trabalhos relacionados abordados.

Característica / Trabalho	1. Márquez et al (2010)	2. Márquez et al. (2011)	3. Corti et al. (2018)	Este trabalho
Lematização	**não**	**sim**	**sim**	**sim**
Crawler	**sim**	**sim**	**não**	**não**
Busca espacial	**sim**	**sim**	**sim**	**sim**
Busca temporal	**não**	**não**	**não**	**não**
Busca temática	**sim**	**sim**	**sim**	**sim**
Pontuação semântica	**não**	**sim**	**não**	**não**
Ordenação pela pontuação	**sim**	**sim**	**sim**	**não**
Expansão feita em tempo de indexação	**sim**	**sim**	**não**	**não**
Agrupamento dos resultados	**não**	**não**	**sim**	**não**
Resolução de consultas em dois níveis	**não**	**não**	**não**	**sim**
Resolução de consultas multidimensionais	**sim**	**sim**	**sim**	**sim**

Fonte: Elaborado pelo autor

Embora o trabalho de Márquez et al. (2010) não use "*lematização*", o mesmo usa a taxonomia através de um domínio de conhecimento para expandir a busca textual transformando palavras em hipônimo (sentido mais abrangente) e merônimo (parte do significado).

Apesar deste trabalho não ter um módulo de crawler para busca de serviços na *Internet*, o mesmo deixa a adição de novos catálogos de forma simples, para análise e extração de novos dados. O trabalho de Corti et al. (2018) conta com um sistema de sincronização, onde a cada nova atualização de dados no WorldMap uma nova tarefa é gerada pelo backend para sincronizar os novos dados no Hypermap.

Para a busca espacial, além deste trabalho permitir criar restrições a partir de nomes de cidades, estados e regiões, também é possível, utilizando o módulo de consultas que disponibiliza uma API, fazer buscas a partir de um *bounding box* especificado ou recuperar registros com geometria semelhante a um registro informado que pertence a base de dados.

Em relação à busca temporal, somente este trabalho e o trabalho de Corti et al.

(2018) permitem realizar consultas com restrições temporais. Para a restrição temática todos os trabalhos relacionados e este trabalho disponibilizam formas de restrições temáticas.

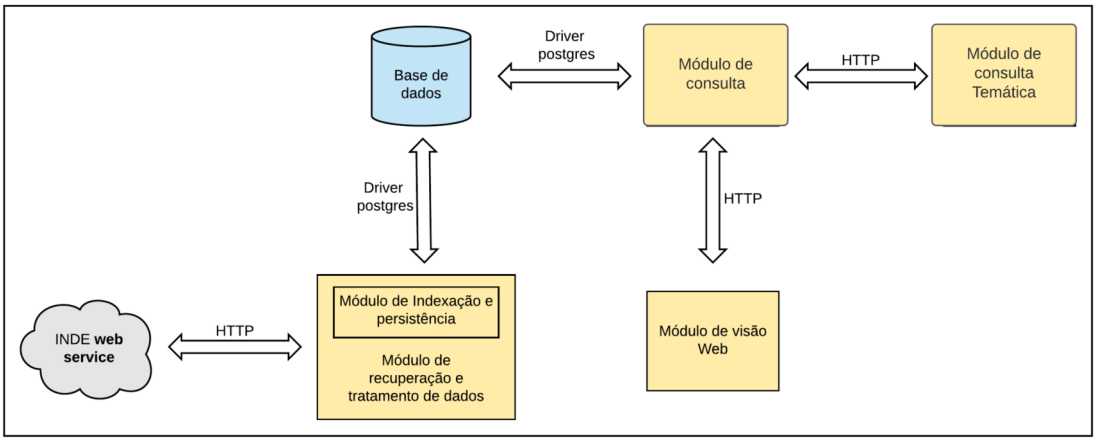
O principal objetivo do trabalho de Marquez et al. (2011) foi criar um pontuação semântica, que pontua a similaridade de um conceito expandido com o metadado real do registro, dessa forma, os resultados com uma maior pontuação tem uma relação mais condizente com o conteúdo do registro. Todo o processo de pontuação das palavras e conceitos é feito em tempo de indexação.

A expansão das palavras em sinônimos, lemas e outros em alguns dos trabalhos relacionados foram feitas em tempo de indexação, para tornar o processamento transparente para o usuário e deixar as consultas mais rápidas. Contudo, neste trabalho não foi notado um aumento no tempo de realização da consulta ao adicionar a expansão em tempo de consulta.

3.2 PROJETO ARQUITETURAL

Esta seção apresenta a arquitetura da ferramenta proposta, descrevendo cada módulo e suas interações com os demais. A arquitetura usada para a implementação é mostrada na Figura 24. Nela, observa-se que a ferramenta é dividida em cinco partes: i) módulo de recuperação e tratamento de dados; ii) módulo de consulta; iii) módulo de consulta temática; iv) módulo de visão web e; v) banco de dados centralizado para armazenamento dos dados.

Figura 24 – Projeto Arquitetural



Fonte: Elaborado pelo autor

O módulo de recuperação de dados é responsável por interagir com o serviço de catálogo da IDE para a coleta de metadados. Para isso, ele realiza requisições HTTP para o serviço de catálogo da IDE, utilizando o padrão CSW do OGC. Logo em seguida, é feito o tratamento dos metadados retornados pelo catálogo para selecionar apenas as informações que ajudem na futura recuperação de dados. Esse módulo possui um módulo de indexação e persistência, que é responsável por armazenar os dados depois de seu tratamento no banco de dados local do sistema.

O módulo de consulta é responsável por resolver as consultas enviadas pelos usuários do sistema. Para isso, o módulo consulta os metadados armazenados no banco de dados do sistema. O módulo de consulta temática também é responsável por resolver consultas. A sua diferença está na responsabilidade de resolver apenas consultas com restrições temáticas. O módulo de visão web é responsável por interagir com os usuários finais do sistema, tanto para receber as requisições, quanto para exibir os resultados das consultas. Essa interação deve ser feita por meio de uma interface gráfica.

3.3 MODELAGEM DO BANCO DE DADOS

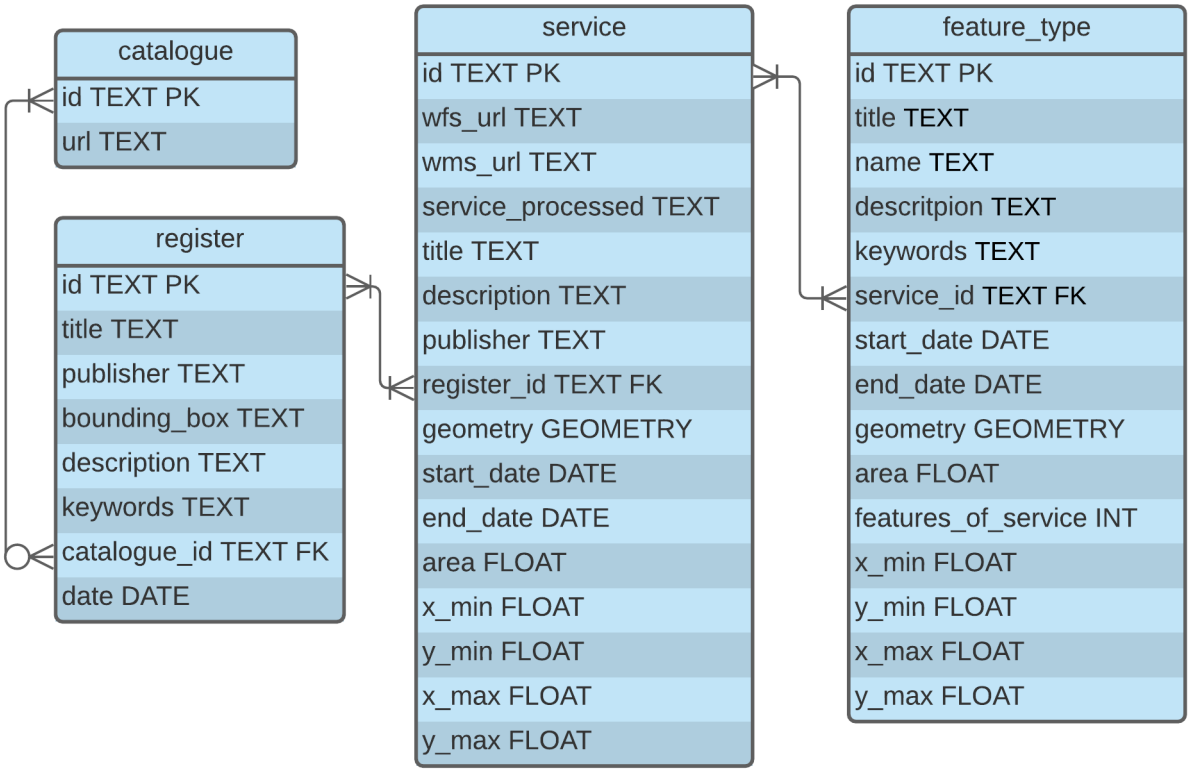
O sistema proposto conta com um banco de dados que é responsável pelo armazenamento persistente dos dados usados para a resolução das consultas. A Figura 25 mostra o esquema lógico utilizado para a implementação desse banco de dados. Ao se analisar a figura, percebe-se que o banco de dados está organizado em quatro tabelas.

A tabela *catalogue* armazena as informações dos catálogos já processados, e é usada para permitir à ferramenta manipular dados de várias infraestruturas. Para cada catálogo, são armazenados a sua identificação e a sua URL.

A tabela *register* é responsável por armazenar as informações dos registros encontrados em cada catálogo. Ela é composta pelas seguintes colunas: a coluna de identificação do registro (definido como id), responsável (*publisher*), data de atualização (date), título (*title*), extensão espacial (*bounding_box*), descrição (*description*) e palavras-chave (keywords). Por fim, a tabela ainda conta com uma chave estrangeira que faz referência para o serviço de catálogo a partir do qual o registro foi coletado.

A tabela *service* é responsável pelo armazenamento dos dados dos serviços que foram encontrados a partir dos registros coletados no catálogo da IDE. Para cada

Figura 25 – Esquema Lógico do Banco de Dados



Fonte: Elaborado pelo autor

serviço são armazenadas as seguintes informações: id, para a sua identificação única, a url do serviço WFS (wfs_url), a url do serviço WMS (wms_url), e qual das URL foi processada para recuperar os dados (*service_processed*), o título (*title*), a descrição (*description*), o responsável (*publisher*), a geometria (*geometry*), sua extensão espacial (x_min, y_min, x_max, y_max), a data de início do conteúdo (*start_date*), a data de fim do conteúdo (*end_date*) e, por último, uma chave estrangeira que representa a referência ao registro a partir do qual o serviço foi identificado.

A última tabela, chamada *feature_type*, armazena as informações a respeito dos tipos de feição oferecidos por cada serviço. Para cada tipo de feição, são armazenados: o *id*, para a identificação de cada tipo de feição (*feature type*), o título (*title*), o nome (*name*), que é um atributo usado para recuperação do tipo de feição no serviço, a sua extensão espacial (x_min, y_min, x_max, y_max), a sua geometria (*geometry*), a data de início do conteúdo (*start_date*), a data de fim do conteúdo (*end_date*), a quantidade de feature types presente no serviço na qual o feature type esta contido (*features_of_service*), a sua área (*area*), a sua descrição (*description*) e as palavras-chave (*keywords*). A tabela *feature_type* ainda conta com uma chave estrangeira, que

permite ao banco de dados identificar o serviço que oferece o tipo de feição.

3.4 IMPLEMENTAÇÃO

As próximas seções descrevem os detalhes de implementação de cada módulo do sistema.

3.4.1 Módulo de Recuperação e Tratamento de Dados

O módulo de recuperação e tratamento de dados tem a função de interagir com o serviço de catálogo de uma IDE para iniciar o processo de coleta, processamento e armazenamento dos mesmos. O módulo é responsável por fazer as requisições HTTP ao catálogo de dados de uma IDE para obter todos os seus registros disponíveis. Para a implementação dessa tarefa foi utilizado o pacote OWSLib[1], um pacote implementado na linguagem de programação *Python* que provê interfaces para acesso aos serviços OGC.

A maior vantagem de se utilizar o pacote OWSLib é que o mesmo realiza a manipulação dos arquivos XML que são retornados depois de cada operação. A Figura 26 mostra as etapas feitas pelo módulo de recuperação e tratamento de dados para encontrar e persistir os serviços e *feature types* encontrados.

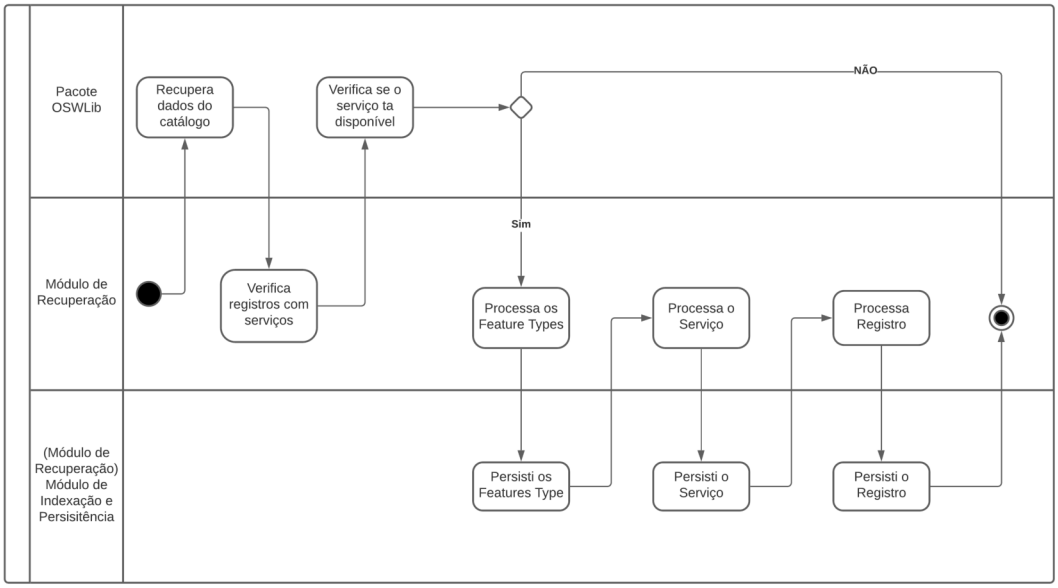
Depois de recuperar os dados disponíveis, o módulo processa os registros recuperados com o intuito de identificar os registros, cujos dados sejam ofertados por meio dos serviços WFS ou WMS. Essa verificação é feita a partir dos metadados do registro, por meio do valor da URI. Caso o registro tenha essas informações, o módulo verifica a disponibilidade do novo serviço encontrado e, caso este seja ativo, o registro e o serviço são tratados e armazenados.

Logo após, é iniciado o processo de recuperação dos metadados do novo serviço. No tratamento dos metadados do registro, as informações mais relevantes como o título, o responsável, a extensão espacial, a data de criação, a descrição e as palavras-chaves são selecionadas para armazenamento. Já no tratamento dos metadados do serviço é selecionado apenas a sua URL, o tipo de serviço reconhecido

[1] Documentação OWSLib 0.20.0. Disponível em: <https://geopython.github.io/OWSLib/>. Acesso em: 10 jan. 2021.

no momento da identificação do registro, o título, a descrição, a extensão espacial, a data de início do conteúdo, a data de fim do conteúdo e o responsável.

Figura 26 – Diagrama de atividade para os passos feito pelo módulo de recuperação e tratamento



Fonte: Elaborado pelo autor

Para cada novo serviço encontrado no registro, é feito o processo de recuperação dos tipos de feição que o mesmo oferece. Para isso, o módulo acessa o serviço por meio de sua URL. Após a recuperação dos dados, o sistema começa o processo de tratamento, no qual uma parte dos metadados recuperados de cada feição (*feature type*) são selecionados como o título, o valor de identificação, a extensão espacial, a descrição, a data de início do conteúdo, a data de fim do conteúdo e as palavras chaves. Logo em seguida, o módulo armazena os dados dos tipos de feição no banco de dados da ferramenta.

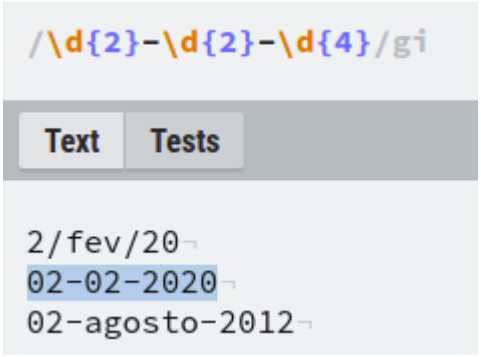
Para a criação da geometria da feição a partir dos metadados, o módulo recupera o *bounding box* informado nos metadados e utiliza a função *ST_MakeEnvelope* do *postgis* que cria uma geometria a partir de quatro valores decimais de entrada: x mínimo, y mínimo, x máximo e y máximo. Já na criação da geometria do serviço, todas as geometrias das feições pertencentes ao serviço são consideradas. Dessa forma, a geometria do serviço é o *envelop* dos municípios, armazenados na tabela auxiliar *place*, que intersectam com as feições que pertencem ao serviço. Para criar uma geometria a partir da junção de outras geometrias foi utilizado a função *ST_Extent* do *postgis*.

Para a identificação da data de início e data de fim do conteúdo do serviço e da feição, os metadados textuais passam por um processo de análise utilizando um regex

para identificar uma data ou um período de datas. *RegEx* ou expressões regulares, são padrões utilizados para selecionar caracteres em uma *string*.

O *regex* da Figura 27 mostra um exemplo para encontrar combinações de datas separadas por hífen e uma combinação que atendeu ao *regex*. A combinação de "\" mais "d" adiciona uma nova regra ao *regex* informando que é esperado uma quantidade *n* de caracteres, a quantidade é definida dentro das chaves após o caractere "d". Como o caractere "-" foi inserido sem nenhum caractere de escape, é adicionado uma regra ao *regex* que é esperado exatamente o mesmo caractere.

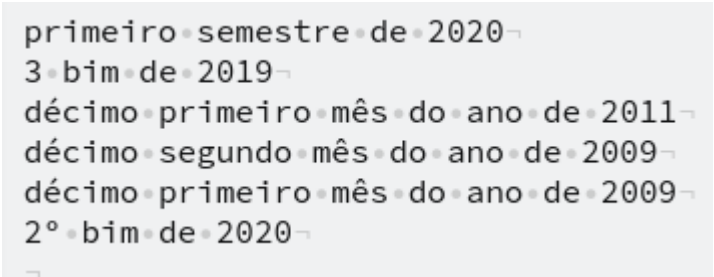
Figura 27 – Exemplo de *regex* para identificar datas



Fonte: Elaborado pelo autor

Foram implementados três *regex* para este trabalho. O primeiro é utilizado para encontrar padrões de datas com mês, bimestres, trimestres e semestres, como mostra os exemplos da Figura 28.

Figura 28 – Exemplo de datas com semestre que serão capturadas pelo *regex*



Fonte: Elaborado pelo autor

Caso nenhuma data seja encontrada, é utilizado o segundo *regex* para tentar encontrar padrões de datas que contenham textos e números juntos, como é mostrado na Figura 29.

Figura 29 – Exemplo de datas com textos que serão capturadas pelo *regex*

```
25 de março 2010
10 de outubro do ano de 2015
primeiro de janeiro de 2010
15 de ago de 2010
```

Fonte: Elaborado pelo autor

Por último, caso não seja encontrada nenhuma data, é utilizado o *regex* para encontrar padrões de datas que estejam entre barras ou hífen, como mostra a Figura 30.

Figura 30 – Exemplo de datas com barras ou hífen que serão capturadas pelo *regex*

```
10/10/2020
10/05/15
02/agosto/20
2/fev/20
02-02-2020
02-agosto-2012
```

Fonte: Elaborado pelo autor

Ao identificar uma data, o módulo irá preencher as datas com o intervalo encontrado. Caso tenha sido encontrado apenas o ano, 2019 por exemplo, o módulo irá definir a data de início como 01/01/2019 e a data de fim como 31/12/2019. Caso o padrão, 09/2019 ou setembro de 2019, seja encontrado o módulo irá definir a data de início como 01/09/2019 e a data de fim como 30/09/2019. No caso do *regex* encontrar duas ou mais datas nos metadados textuais, a menor data e a maior serão utilizadas como data de início e data de fim respectivamente.

Para os dados temáticos, os metadados textuais mais relevantes são recuperados pelo módulo e armazenados. No caso dos serviços, além de seu título, descrição e palavras-chaves os metadados das feições (título, descrição e palavras-chaves), que pertencem ao mesmo também são armazenados. De forma semelhante, os metadados do serviço ao qual a feição pertence também são adicionados a mesma e armazenados. Isso permite que a ferramenta encontre serviços, a partir de uma busca temática com dados de entrada que estão presentes nas feições do serviço. Da mesma maneira, as feições podem ser encontradas, caso o dado de entrada da busca temática esteja presente em um serviço ao qual a feição pertence.

3.4.2 Módulo de Consulta

O módulo de consulta é responsável por resolver as consultas feitas pelo módulo de visão web ou por outro cliente através de requisições REST. Esse módulo recebe todas as restrições da consulta, porém somente as restrições espacial e temporal são executadas de fato por ele. Para a restrição temática, as informações são passadas para o módulo de consulta temática, resolvidas e por fim, retornadas para o módulo de consulta, que é responsável por combinar os resultados e devolver como resposta da requisição.

Para relacionar os resultados das restrições solicitadas, somente os identificadores únicos dos *feature types* ou *services* são retornados por cada restrição. Dessa forma, após a execução, somente os resultados com identificadores incluídos no total de restrições feitas são retornados. Ao invés de recuperar todos os dados do recurso como geometria, dados textuais e temporais, é selecionado apenas o seu identificador A dinâmica de retorno apenas dos identificadores dos recursos favorece o tempo em que a consulta demora para ser feita.

Para recuperação dos dados de cada recurso, o módulo de consulta disponibiliza uma funcionalidade que recebe os identificadores e retorna todos os dados disponíveis dos recursos solicitados.

Na execução de uma consulta espacial, o módulo de consulta recebe o nome do local solicitado pelo cliente. Para realizar a consulta, primeiro o módulo recupera as informações espaciais acerca do local enviado através de uma tabela auxiliar *place*. A tabela *place* é uma tabela criada a partir dos *shapefiles* disponibilizados pelo Instituto Brasileiro de Geografia e Estatística (IBGE)[2] contendo o nome do lugar, o tipo do lugar (município, estado ou região) e sua geometria.

Depois que as informações espaciais do local são identificadas, o módulo seleciona todos os recursos cuja extensão espacial intersecta o local definido na consulta do usuário, que pode ser um serviço ou um tipo de feição. Para aumentar a eficiência da consulta, o módulo utiliza o *bounding-box* das geometrias que representam a localidade solicitada pelo usuário e a região coberta por cada recurso.

Para a realização dessa tarefa foram criados dois procedimentos armazenados. O primeiro obtém o *bounding-box* da área da interseção entre a localidade definida na consulta e a região coberta pelo recurso. O segundo procedimento utiliza o primeiro

[2] Download | IBGE. Disponível em: <https://www.ibge.gov.br/geociencias/downloads-geociencias.html>. Acesso em: 17 jan. 2021.

procedimento armazenado para calcular a área da interseção e a similaridade entre a região definida na consulta e a região coberta pelo recurso. O objetivo dessa ação consiste em priorizar recursos, cuja extensão espacial seja mais parecida com a região definida na consulta. Para o cálculo da similaridade entre duas regiões foi usada uma adaptação da equação de Tversky, ao domínio de dados espaciais Tversky (1977). Essa equação, que é mostrada na Equação 1, calcula a similaridade entre dois objetos considerando tanto as características que eles têm em comum, quanto as suas diferenças.

$$Tversky(A, B) = \frac{A \cap B}{A \cap B + \alpha(A - B) + \beta(B - A)} \tag{1a}$$

Equação 1

Para o cálculo da similaridade entre duas regiões geográficas, os elementos da equação foram considerados da seguinte forma:

- *A* representa o *bounding-box* da região referente ao local informado pelo cliente;
- *B* representa o *bounding-box* da região coberta pelo recurso que está sendo avaliado. O recurso pode ser um serviço ou um tipo de feição;
- *a* interseção entre A e B representa a área da interseção entre A e B;
- o complemento A - B representa a área de A que não intersecta B;
- o complemento B - A representa a área de B que não intersecta A;
- as constantes α e β definem o peso de cada complemento no cálculo final da similaridade. No trabalho foi utilizado o valor *0,5* para ambas as constantes. O resultado final é um valor entre *0* e *1*. Quanto mais próximo de *1* mais as extensões espaciais comparadas são similares, logo, quanto mais próximo de *0* menor é a similaridade.

Na execução de uma consulta temporal, o módulo de consulta recebe um intervalo de tempo que representa o período de interesse do usuário. Esse intervalo é composto por duas datas: uma data inicial e uma data final. A partir do intervalo enviado, o módulo de consulta recupera todos os recursos (tipo de feição ou serviço), cuja extensão temporal tem alguma intersecção com o intervalo requisitado.

Para cada recurso selecionado, o módulo calcula a similaridade entre o intervalo requisitado e a extensão temporal coberta pelo recurso. Essa similaridade também é calculada por meio da Equação 1. Entretanto, em uma consulta temporal, os dados de entrada são o intervalo temporal solicitado na requisição e o intervalo temporal coberto pelo recurso que está sendo analisado. A interseção e o complemento desses intervalos é calculado em meses.

Para resolver as consultas de forma mais rápida, o módulo de consulta realiza todas as consultas espacial e temporal em paralelo. Caso a consulta recebida tenha alguma restrição temática, ele a envia ao módulo de consulta temática e aguarda o resultado do seu processamento, que também acontece em um processo paralelo.

Quando todas as consultas são executadas, os identificadores são retornados para cada restrição feita. Caso a consulta contemple as três restrições, espacial, temporal e temática, três listas de identificadores são retornados para uma última filtragem. Cada elemento da lista tem o identificador do recurso e uma pontuação que define a similaridade do recurso e o dado de entrada em relação à restrição.

Os identificadores que estão presentes nas três listas são selecionados como uma nova lista de resposta. Os elementos da lista de resposta são compostos pelo seu identificador, igual nas listas, e pela sua pontuação final, que é definida a partir da média aritmética da sua pontuação em cada lista.

Caso apenas duas restrições sejam feitas na consulta, apenas duas listas de resultados serão retornadas para filtragem. Os identificadores que pertencem às duas listas são retornados como uma lista de resposta e a pontuação de cada elemento é a média aritmética de sua pontuação em cada uma das duas listas. Por fim, caso apenas uma restrição seja feita, a lista de resposta será a própria lista resultante do processo sem a necessidade de uma média.

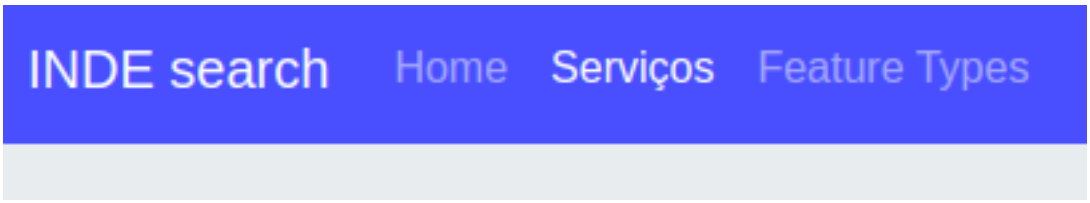
3.4.3 Módulo de Consulta Temática

O módulo de consulta temática é responsável por resolver as consultas temáticas e repassar o resultado para o módulo de consulta. Para resolver essas consultas, o módulo de consulta temática utiliza o *Solr*. Ao receber a consulta, o módulo realiza uma consulta no *Solr* usando como base o texto que foi enviado como parâmetro da consulta. Após executar a consulta, o *Solr* retorna o *id* do elemento e o *scoring* alcançado de acordo com os próprios critérios de busca do mesmo.

3.4.4 Módulo de Visão Web

O Módulo de Visão Web é responsável por disponibilizar uma interface web que permite ao usuário realizar consultas por dados espaciais. A Figura 31 mostra dois menus de acesso nos quais o usuário pode escolher qual o tipo de recurso que deseja recuperar.

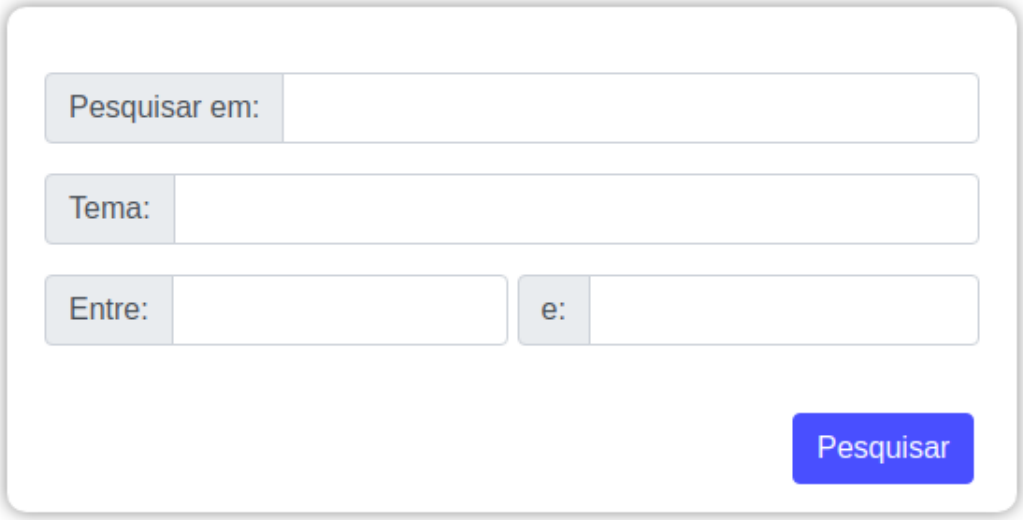
Figura 31 – Menu de acesso aos recursos disponíveis para busca



Fonte: Elaborado pelo autor

A Figura 32 mostra o formulário no qual o usuário pode inserir as restrições que devem ser usadas para a consulta.

Figura 32 – Formulário de pesquisa do módulo de visão web



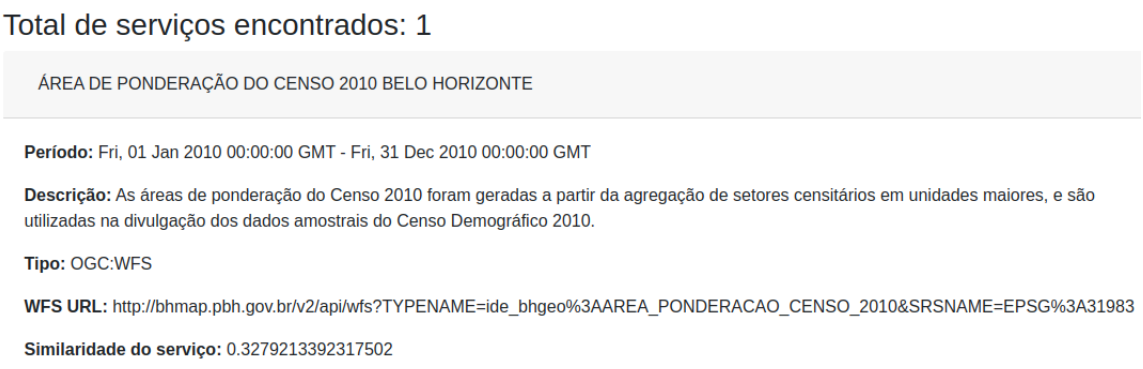
Fonte: Elaborado pelo autor

No campo "*pesquisar em*" o usuário pode inserir o nome de uma cidade, estado ou região e assim adicionar a restrição espacial. No campo "*tema*" o usuário pode colocar uma ou mais palavras-chaves para adicionar a restrição temática. Por fim, o usuário ainda pode informar um intervalo de tempo nos campos "*entre*", definindo uma data inicial e uma data final, para adicionar uma restrição temporal.

A Figura 33 abaixo mostra um exemplo de um resultado para uma busca feita por dados sobre serviços que tenham dados de Belo Horizonte entre os anos de 2010

e 2011. Nesse exemplo, é realizada uma consulta espacial por dados sobre a cidade de Belo Horizonte, uma consulta temporal por dados sobre o período de “01/01/2010” a “01/01/2011”.

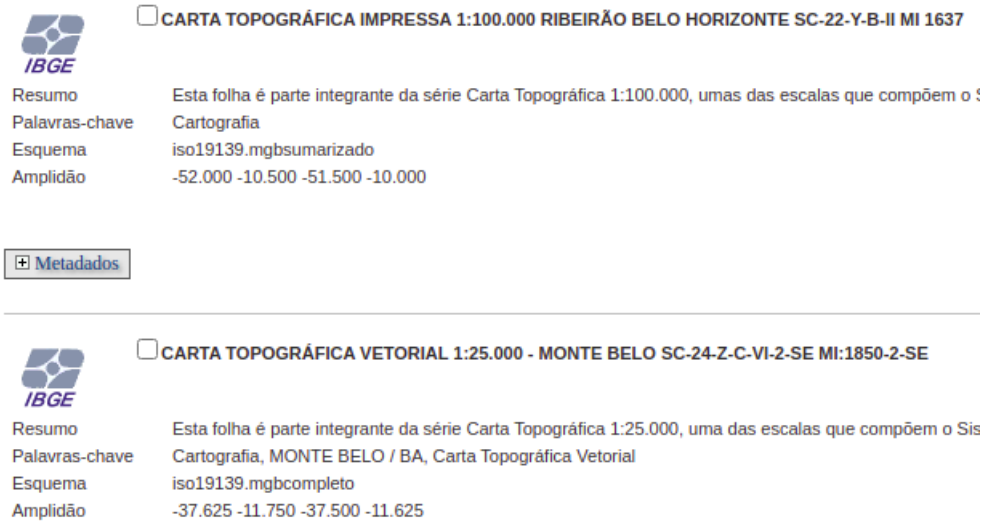
Figura 33 – Resposta para consulta "Belo Horizonte", “01/01/2010” a “01/01/2011”



Fonte: Elaborado pelo autor

Para a mesma consulta feita no portal da INDE, Figura 34, são retornados dois resultados, mas apenas um deles sendo referente a cidade de Belo Horizonte.

Figura 34 – Resposta para consulta "Belo Horizonte", “01/01/2010” a “01/01/2011” no portal da INDE



Fonte: Elaborado pelo autor

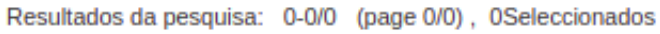
Uma segunda busca feita utilizou as três restrições. Para a restrição espacial foi inserido o valor “são paulo”, para a restrição temática o valor “alphaville” foi usado e por fim, para a restrição temporal “01/01/2013” e “31/12/2013” foram usados. Na ferramenta desenvolvida foi encontrado um resultado, assim como mostra a Figura 35.

Figura 35 – Resposta encontrada pela ferramenta desenvolvida para consulta “são paulo”, "alphaville", “01/01/2013” a “31/12/2013”

Fonte: Elaborado pelo autor

Já as mesmas restrições sendo inseridas no portal da INDE não geram nenhum resultado, como mostra a Figura 36.

Figura 36 – Nenhum resultado encontrado pelo portal da INDE para a consulta “são paulo”, "alphaville", “01/01/2013” a “31/12/2013”



Fonte: Elaborado pelo autor

4 CONCLUSÃO

Nos últimos anos, muitas infraestruturas de dados espaciais têm sido desenvolvidas no mundo todo, como uma forma de facilitar a disseminação e o reuso de dados espaciais. Desde a sua proposição, governos de diversos locais, como Estados Unidos, Canadá e Austrália têm implementado IDEs para a disseminação dos dados produzidos por suas agências. Outras iniciativas, como o INSPIRE, na Europa, visam criar IDEs de maior abrangência geográfica. No Brasil, também já existe uma infraestrutura disponível, que é chamada de INDE.

Com o intuito de facilitar a recuperação dos seus conjuntos de dados por diferentes tipos de usuário, as IDEs atuais oferecem serviços de catálogo. Os clientes podem usar esse serviço para localizar os dados nos quais estão interessados. Embora os serviços de catálogo facilitem a localização dos dados, eles possuem limitações importantes para a resolução de diversos tipos de consulta. Alguns problemas surgem porque os catálogos atuais resolvem suas consultas apenas com base nos metadados informados pelos provedores dos dados no momento do registro, que normalmente são pouco precisos.

De modo a resolver essas limitações, este trabalho de conclusão de curso propõe uma nova ferramenta de busca, que extrai informações mais precisas, em nível de tipos de feição, para melhorar a recuperação de dados oferecidos por IDEs.

Durante o desenvolvimento do trabalho percebeu-se que os dados das infraestruturas de dados espaciais são ofertados a partir de uma série de padrões definidos pela OGC. Desta forma, as aplicações se comunicam com as infraestruturas seguindo as especificações do padrão utilizado na comunicação. Vários dados e metadados podem ser retornados, dependendo do tipo de padrão OGC utilizado para se comunicar. Depois de analisar todos os metadados de resposta notou-se que alguns seriam mais relevantes para um processo de recuperação da informação. Para as consultas espaciais o metadado que descrevia sobre a área delimitadora do recurso foi utilizado. Para as consultas temáticas e temporais os metadados textuais que descreviam sobre o recurso foram utilizados. Com os metadados selecionados um banco de dados centralizado pode ser gerado e populado pelo processamento da infraestrutura nacional de dados espaciais (INDE).

O módulo de recuperação e tratamento de dados foi desenvolvido e já se comunica com a infraestrutura a partir dos padrões OGC salvando os metadados mais

relevantes no banco de dados. O módulo de consulta, responsável pelas consultas espaciais, temporais e repassar as consultas temáticas para o módulo de consulta temática, foi implementado e executa as consultas enviadas paralelamente. O módulo de consulta temática foi criado e executa todas as consultas temáticas enviadas pelo módulo de consulta. Por fim, o módulo de visão web foi desenvolvido e permite que as consultas sejam enviadas a partir de uma interface web.

Por meio dessas conclusões pode-se perceber que todos os objetivos do trabalho foram alcançados. Trabalhos futuros podem ser feitos para melhorar mais a ferramenta, tais como: a) permitir o agrupamento dos resultados nos mesmos níveis das restrições das consultas (espacial, temática e temporal); b) criar uma crawler que busque novas infraestruturas na internet e adicione a sua URI para a ferramenta processa-lo e adicioná-lo a base dados; c) melhorar a interface web e; d) adicionar um mapa interativo para as consultas espaciais.

REFERÊNCIAS

BEAUJARDIERE, J. de L. Opengis® web map server implementation specification. version 1.3. 0. Open Geospatial Consortium, 2006.

BIAŁECKI, A.; MUIR, R.; INGERSOLL, G.; IMAGINATION, L. Apache lucene 4. In: **SIGIR 2012 workshop on open source information retrieval**. [S.l.: s.n.], 2012. p. 17.

BRASIL. Decreto n o 6.666, de 27 de novembro de 2008. Institui, no âmbito do poder executivo federal, a infraestrutura nacional de dados espaciais – inde (considero interessante completar). **Diário Oficial da União, Poder Executivo**, 2008. Brasília, DF, 28 de nov. 2008. Seção 1, p. 57.

CHEN, N.; CHEN, Z.; HU, C.; DI, L. A capability matching and ontology reasoning method for high precision ogc web service discovery. **International Journal of Digital Earth**, Taylor & Francis, v. 4, n. 6, p. 449–470, 2011.

CONCAR. Plano de ação para a implantação da inde. **Comissão Nacional de Cartografia (CONCAR)**, 2010.

CORTI, P.; KRALIDIS, A. T.; LEWIS, B. Enhancing discovery in spatial data infrastructures using a search engine. **PeerJ Computer Science**, PeerJ Inc., v. 4, p. e152, 2018.

CRAGLIA, M.; ANNONI, A. Inspire: An innovative approach to the development of spatial data infrastructures in europe. **Research and theory in advancing spatial data infrastructure concepts**, ESRI Press: Redlands, CA, USA, p. 93–105, 2007.

ERL, T. **Service-oriented architecture: analysis and design for services and microservices**. [S.l.]: Prentice Hall Press, 2016.

FIELDING, R. T.; TAYLOR, R. N. **Architectural styles and the design of network-based software architectures**. [S.l.]: University of California, Irvine Irvine, 2000. v. 7.

GRAINGER, T.; POTTER, T. **Solr in action**. [S.l.]: Manning Publications Co., 2014.

GUAN, W. W.; BOL, P. K.; LEWIS, B. G.; BERTRAND, M.; BERMAN, M. L.; BLOSSOM, J. C. Worldmap–a geospatial framework for collaborative research. **Annals of GIS**, Taylor & Francis, v. 18, n. 2, p. 121–134, 2012.

MANJAVACAS, E.; KÁDÁR, Á.; KESTEMONT, M. Improving lemmatization of non-standard languages with joint learning. **arXiv preprint arXiv:1903.06939**, 2019.

MÁRQUEZ, J.; CÓRCOLES, J.; QUINTANILLA, A. A semantic index structure for integrating ogc services in a spatial search engine. In: IEEE. **2010 IEEE Conference on Open Systems (ICOS 2010)**. [S.l.], 2010. p. 103–108.

MARQUEZ, J.; CORCOLES, J. E.; QUINTANILLA, A. Scoring results in a geospatial services search engine according to geographic and semantic awareness. In: IEEE. **2011 7th International Conference on Next Generation Web Services Practices**. [S.l.], 2011. p. 238–243.

NEBERT, D. **Developing Spatial Data Infrastructures: The SDI Cookbook. Technical report, Global Spatial Data Infrastructure**. 2004. Disponível em: <http://www.gsdi.org/docs2004/Cookbook/cookbookV2.0.pdf>.

NEBERT, D.; WHITESIDE, A.; VRETANOS, P. Opengis-catalogue services specification (version: 2.0. 2). **Open Geospatial Consortium**, p. 115, 2007.

PLISSON, J.; LAVRAC, N.; MLADENIC, D. et al. A rule based approach to word lemmatization. In: **Proceedings of IS**. [S.l.: s.n.], 2004. v. 3, p. 83–86.

RAJABIFARD, A.; WILLIAMSON, I. P. Spatial data infrastructures: concept, sdi hierarchy and future directions. 2001.

SMILEY, D.; PUGH, E.; PARISA, K.; MITCHELL, M. **Apache Solr enterprise search server**. [S.l.]: Packt Publishing Ltd, 2015.

SOMMERVILLE, I. Engenharia de software, 9a. **São Palo, SP, Brasil**, 2011.

TVERSKY, A. Features of similarity. **Psychological review**, American Psychological Association, v. 84, n. 4, p. 327, 1977.

VRETANOS, P. et al. Web feature service implementation specification with corrigendum. version 1.1. 3. Open Geospatial Consortium, 2016.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras
Rua José Antônio da Silva, 300 - Bairro Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Restrito

Entrega de trabalho de conclusão de curso

Assunto: Entrega de trabalho de conclusão de curso
Assinado por: Leanderson Santos
Tipo do Documento: Anexo
Situação: Finalizado
Nível de Acesso: Restrito
Hipótese Legal: Informação Pessoal (Art. 31 da Lei no 12.527/2011)
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Leanderson Coelho dos Santos**, **ALUNO (201712010001) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 04/03/2021 13:26:19.

Este documento foi armazenado no SUAP em 04/03/2021. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 183795
Código de Autenticação: 503bd35e0c