MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA CARTOGRÁFICA

1º Ten GABRIEL BOURDON FERNANDES
1º Ten THIAGO MENCK PFEIFER MACEDO

CONVERSÃO DO MODELO EDGV 3.0 PARA O MODELO MGCP TRD4 V4.4

Rio de Janeiro
2020

INSTITUTO MILITAR DE ENGENHARIA

1º Ten GABRIEL BOURDON FERNANDES
1º Ten THIAGO MENCK PFEIFER MACEDO

CONVERSÃO DO MODELO EDGV 3.0 PARA O MODELO MGCP TRD4 V4.4

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia Cartográfica do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro Cartógrafo.

Orientador: Major Felipe Ferrari - M.Sc.

Rio de Janeiro
2020

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP 22290-270

004.69 Fernandes, Gabriel Bourdon
S586e Conversão do Modelo EDGV 3.0 para o Modelo MGCP TRD4 v4.4 / Gabriel Bourdon Fernandes, Thiago Menck Pfeifer Macedo, orientado por Felipe Ferrari - Rio de Janeiro: Instituto Militar de Engenharia, 2020.

63p.: il.

Projeto de Fim de Curso (graduação) - Instituto Militar de Engenharia, Rio de Janeiro, 2020.

1. Curso de Graduação em Engenharia Cartográfica - projeto de fim de curso. 1. Palavra 01. 2. Palavra 02. 3. Palavra 03. I. Ferrari, Felipe . II. Título. III. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

1º Ten GABRIEL BOURDON FERNANDES
1º Ten THIAGO MENCK PFEIFER MACEDO

CONVERSÃO DO MODELO EDGV 3.0 PARA O MODELO MGCP TRD4 V4.4

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia Cartográfica do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro de Cartógrafico.

Orientador: Major Felipe Ferrari - M.Sc.

Aprovado em 30 de Outubro de 2020 pela seguinte Banca Examinadora:

Major Felipe Ferrari - M.Sc. do IME

Major Ivanildo Barbosa - D.Sc. do IME

Capitão Philipe Borba - Eng. Crtf. da DSG

Rio de Janeiro
2020

Ao Instituto Militar de Engenharia, alicerce da minha formação e aperfeiçoamento.

AGRADECIMENTOS

Agradecemos primeiramente a Deus pelas nossas vidas e por todas as bençãos que recebemos todos os dias.

Ao nosso orientador Major Felipe Ferrari, e aos Capitães Diniz e Borba por sua disponibilidade e atenção, assim como ao Major Ivanildo, que se mostrou disponível mesmo nos corredores do IME.

Ao nosso amigo Bryan Santos, que tanto se esforçou e se mostrou presente para nos ajudar sempre que precisamos.

“Eu não fracassei. Só descobri 10 mil maneiras de não fazer uma lâmpada. ”

THOMAS EDISON

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE SIGLAS

EDGV	Especificações Técnicas para Estruturação de Dados Geoespaciais Vetoriais
MGCP	Multinational Geospatial Co-Production Program
CAD	Computer Aided Design
MTD	Mapoteca Topográfica Digital
TBCD	Tabela da Base Cartográfica Digital
DSG	Diretoria de Serviço Geográfico
IBGE	Instituto Brasileiro de Geografia e Estatística
CEPAD	Comitê Especializado para Estudo do Padrão de Intercâmbio de Dados Cartográficos Digitais
MMA	Ministério do Meio Ambiente
CONCAR	Comissão Nacional de Cartografia
CEMND	Comitê de Estruturação da Mapoteca Nacional Digital
EUA	Estados Unidos da América
NGA	National Geospatial Agency
IBM	International Business Machines Corporation
SIG	Sistema de Informação Geográfica
OSGeo	Open Source Geospatial Foundation
IDE	Integrated Development Environment
INDE	Infraestrutura Nacional de Dados Espaciais
SCN	Sistema Cartográfico Nacional
UML	Unified Modeling Language
JSON	JavaScript Object Notation

SQL	Structured Query Language

RESUMO

Atualmente no Brasil existem cartas representativas do território nacional que foram desenvolvidas com o padrão conhecido como EDGV (Especificação Técnica para a Estruturação de Dados Geoespaciais Vetoriais), na versão 2.1.3, e estão sendo atualizadas para a versão mais recente, a 3.0. Por outro lado, no cenário internacional, foi formado um grupo com diferentes nações participantes, que visa a cooperação para o mapeamento de todo o território do globo, e é conhecido como MGCP (Multinational Geospatial Coproduction Program). O Brasil, como participante deste grupo internacional, necessita produzir cartas que sigam o padrão estabelecido pelo MGCP para as classes e atributos, e para isso, este projeto visa o uso do plugin *EDGV Converter*, disponível no repositório do software QGIS, para realizar a conversão entre bancos de dados no formato da norma EDGV 3.0 para o TRD4 v4.4 do MGCP. Com isso, irá contribuir para que o país tenha participação relevante como membro do grupo.

ABSTRACT

Currently, in Brazil, there are charts representing the national territory that were developed in the standard known as EDGV (Technical Specification for Structuring Vector Geospatial Data), in version 2.1.3, and are being updated to the latest version, 3.0. On the other hand, in the international scene, a group with different participating nations, which aims to together cover the entire surface of the globe, and is known as MGCP (Multinational Geospatial Co-production Program). Brazil, as a participant in this international group, needs to produce charts that follow the standard established by MGCP for classes and attributes, and this project intends to use the plugin *EDGV Converter*, available in the QGIS repository, to perform the convertion amongst databases from the EDGV 3.0 standard to TRD4 v4.4 from MGCP. That will allow the country to have a more relevant participation as a member of the group.

1 INTRODUÇÃO

1.1 OBJETIVO

Este projeto tem como objetivo o uso do plugin *EDGV Converter*, disponível no repositório do *software* QGIS, para realizar a conversão entre classes e atributos do padrão ET-EDGV 3.0 (Especificações Técnicas para Estruturação de Dados GeoespaciaisVetoriais), para a TRD4 (Technical Reference Documentation) v4.4 do MGCP (Multinational Geospatial Co-Production Program). Esse mapeamento será realizado utilizando como auxílio recursos de computação e programação, disponíveis no QGIS.

1.2 ESPECIFICAÇÕES TÉCNICAS PARA ESTRUTURAÇÃO DE DADOS GEOESPACIAIS VETORIAIS (ET-EDGV)

No fim dos ano 80, todos os processos de produção cartográfica no Brasil, eram analógicos, e foi no início da década de 1990 que surgiram as primeiras automações destes processos baseados em CAD (Computer Aided Design), ao mesmo tempo em que estavam sendo adaptadas as normas técnicas para o meio digital para que os dados produzidos fossem validados, de acordo com às regras topológicas (CONCAR, 2010).

A MTD (Mapoteca Topográfica Digital), desenvolvida pelo IBGE (Instituto Brasileiro de Geografia e Estatística), e a TBCD (Tabela da Base Cartográfica Digital), desenvolvida pela DSG (Diretoria de Serviço Geográfico), deram início às atividades de estruturação de dados espaciais vetoriais, e com isso, cada órgão de mapeamento desenvolveu suas próprias estruturas para as atividades de mapeamento. Como consequência, as bases geradas pela DSG e IBGE nesse período estavam em estruturas diferentes, tornando necessária a conversão entre os padrões para que os produtos sejam utilizados em conjunto (CONCAR, 2010).

Essa diferença entre os produtos geoespaciais da DSG e IBGE gerou certa dificuldade de compatibilização para um padrão único dos dados, pois existiam diferentes categorias, semânticas para certas categorias, feições e diferentes atributos por feição (CONCAR, 2010).

Identificadas as diferenças e a necessidade de se criar um padrão único, em junho de 1997, criou-se o CEPAD (Comitê Especializado para Estudo do Padrão de Intercâmbio de Dados Cartográficos Digitais), que visava desenvolver um padrão para os dados

cartográficos digitais a serem gerados pelas organizações. As reuniões do CEPAD foram realizadas até o final do ano de 1998 (GALVÃO, W. P. 2017).

Com o fim das reuniões do CEPAD, os próprios órgãos DSG e IBGE realizaram reuniões com o objetivo de obter um padrão único para os modelos MTD e TBCD, porém, a falta de recursos financeiros, para as despesas com deslocamento de pessoal, impediram que o objetivo final fosse alcançado (CONCAR, 2010).

O MMA (Ministério do Meio Ambiente), em 2004, junto a CONCAR (Comissão Nacional de Cartografia), realizaram auditorias técnicas em bases digitais de empresas privadas, para verificar a adequação das mesmas ao SCN (Sistema Cartográfico Nacional). Essas auditorias visavam compor, com cartas na escala 1:100.000, a base cartográfica digital da Amazônia Legal (CONCAR, 2010).

A partir deste momento, iniciou-se a proposta da parceira do MMA com a DSG e IBGE, para que juntassem esforços e desenvolvessem um mesmo padrão de estrutura de dados (CONCAR, 2010). Esse padrão seria aplicável às bases digitais da Amazônia Legal, que na época, não seguiam padrão único.

Com o objetivo de definir um único padrão de estrutura de dados geoespaciais, usou-se como base a modelagem orientada a objetos desenvolvida pela DSG, a EGB2000. Nesta, foram incorporados conceitos da TBCD (Tabela da Base Cartográfica Digital) e MTD (Mapoteca Topográfica Digital), atendendo aos requisitos solicitados para a base cartográfica pretendida pelo MMA.

A estrutura desenvolvida foi proposta em 2005, para a estruturação da Mapoteca Nacional Digital. Em 2006, a CONCAR aprovou a versão 1.0 como norma provisória até que outra versão mais completa fosse desenvolvida, o que mais tarde, no ano de 2008, resultou na ET-EDGV 2.0 (CONCAR, 2010).

Com a publicação da nova versão da ET-EDGV, o SCN iniciou junto a seus órgãos, a adaptação dos dados geoespaciais para a versão 2.0. No mesmo período, a CEMND (Comitê de Estruturação da Mapoteca Nacional Digital) desenvolveu melhorias na estrutura de dados, no que culminou na versão 2.1 da ET-EDGV (CONCAR, 2010).

1.3 MULTINATIONAL GEOSPATIAL CO-PRODUCTION PROGRAM (MGCP)

O MGCP é um programa de mapeamento organizado pela NGA (National Geospatial Agency), tem por objetivo mapear todo o globo terrestre. Conta atualmente com 32 nações membros, que produzem cartas vetoriais nas escalas de 1:50.000 e 1:100.000, em células com dimensões de 1º x 1º (CASTRO, 2019).

Esse banco de dados GIS conta com a especificação das escalas de 1: 50.000 e 1: 100.000, que atende às necessidades nacionais e internacionais (C. JÓZSEF & M. OLÍVIA, 2009).

Para a produção, as fontes das ortofotografias feitas com base em fotografias aéreas e materiais de sensoriamento remoto disponíveis precisam ser recentes, e todas as nações participantes financiam a implementação das premissas a partir de seu próprio orçamento (C. JÓZSEF & M. OLÍVIA, 2009).

1.3.1 TECHNICAL REFERENCE DOCUMENTATION (TRD)

O TRD é uma especificação técnica do MGCP que visa a padronização de trabalho e metodologias para a produção de dados geoespaciais, e atualmente, é de responsabilidade da França. Como referência para uso no trabalho, foi utilizada a TRD4 v4.4, que é constituída de toda a documentação necessária para o mapeamento, assim como todas as estruturas em formato .gml/.xml (MGCP_XSD_TRD4), modelo semântico (*Semantic Information Model*), regras de aquisição (*Extraction Guide*), especificações de metadados (*Metadata Specification*), padrões de qualidade (*QA Cookbook*), entre outras (CASTRO, 2019).

1.4 MOTIVAÇÃO DA PESQUISA

O Brasil, atualmente, é um dos países com a posição de membro da MGCP, e para que continue fazendo parte do grupo, é necessário fornecer cartas, nas especificações previamente explicitadas (DSG, 2019).

Para que essas entregas sejam feitas, é necessário que as cartas existentes hoje sejam convertidas do modelo da ET-EDGV para o MGCP, e para isso, deve ser feito um trabalho de mapeamento entre classes e atributos de um para o outro.

O uso de um plugin com esta funcionalidade, proporcionará considerável economia de tempo para as equipes de cartografia do Exército Brasileiro, pois onde seria usada a mão de obra disponível para executar a conversão de cada carta a ser enviada, será feito de forma computacional.

A utilização desta solução proporcionará também participação mais ativa do Brasil dentro da comunidade do MGCP, pois simplificará a padronização das cartas, e com isso, ocorrerão mais entregas por parte do país.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 BANCOS DE DADOS

Um banco de dados é uma coleção de dados persistentes, usada pelos sistemas de aplicação de uma determinada empresa (DATE, 2004). Funciona como um local onde são armazenados dados para determinadas atividades, ou seja, um conjunto organizado de dados relacionados, criado com determinado objetivo e que atende uma comunidade de usuários (COSTA, 2011).

Pode ser descrito como uma coleção logicamente coerente de dados com algum significado inerente. Uma variedade aleatória de dados não pode ser corretamente chamada de banco de dados (ELMASRI & NAVATHE, 2011).

Um banco de dados é construído e projetado com dados para uma finalidade específica. Possui grupo definido de usuários e aplicações previamente concebidas, nas quais os usuários estão interessados (ELMASRI & NAVATHE, 2011).

Para que se entenda o conceito de Modelo Conceitual, é importante que se conheça primeiro os conceitos de Entidade, Atribuito e Relacionamento. A figura abaixo mostra a representação usual desses conceitos, que são explicados de nos subcapítulos subsequentes.

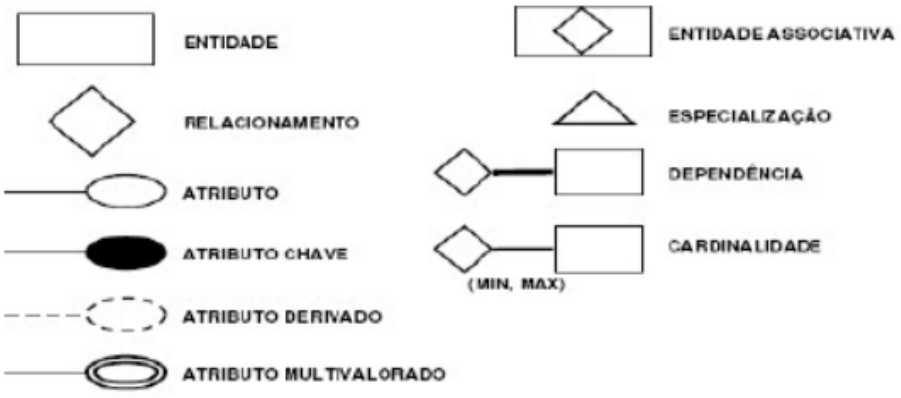


FIG. 2.1: Notações Utilizadas em Modelos de Bancos de Dados

2.1.1 ENTIDADES

Define-se entidade como um objeto que existe no mundo real com uma identificação distinta e com um significado próprio. São objetos e partes envolvidas em um domínio, e classificados de acordo com sua existência. As nomenclaturas adotadas para as entidades,

são substantivos concretos ou abstratos, que representem de forma clara sua função dentro do domínio (SIQUEIRA, 2016).

As entidades podem se classificar em 3 (três) grupos, de acordo com o motivo de sua existência, que são eles (BALAGUER, 2016):

- Entidades Fortes: aquelas que independem de outras, que sozinhas possuem sentido completo.
- Entidades Fracas: aquelas que dependem de outras, pois individualmente, não fazem possuem sentido completo.
- Entidades Associativas: quando há a necessidade de associar uma entidade a um relacionamento. Na modelagem Entidade-Relacionamento, se faz isso para tornar um relacionamento, uma entidade associativa, que se relaciona com outras entidades.

2.1.2 RELACIONAMENTO DE ENTIDADES

Para a montagem de um banco de dados, é preciso que sejam definidos os relacionamentos entre as entidades consideradas. De acordo com a quantidade de objetos envolvidos em cada lado do relacionamento, eles podem ser classificados de 3 (três) formas (BALAGUER, 2016):

- Relacionamento 1..1 (um para um): acontece quando uma das duas entidades envolvidas, referenciam uma, e apenas uma, unidade da outra.
- Relacionamento 1..n ou 1..* (um para muitos): é um relacionamento em que uma entidade referencia muitas unidades da outra, e essas várias, podem referenciar apenas uma unidade da origem.
- Relacionamento n..n ou *..* (muitos para muitos): cada entidade, de ambos os lados, podem referenciar múltiplas unidades da outra.

2.1.3 ATRIBUTOS

São interpretados como caracteristica de uma entidade que tem valor para a aplicação, que deve ser observada pelo usuário. A definição dos atributos devem ser discutidas antes de se começar o desenvolvimento, para que todo atributo que necessário, no presente ou no futuro, seja incluído. Estas discussões devem acontecer também, para que não se incluam dados desnecessários nos bancos de dados (SIQUEIRA, 2016).

Os atributos, assim como as entidades, podem ser classificados dependendo do tipo de informação que representam. São eles (SIQUEIRA, 2016):

- Atributo Monovalorado: assume valor único para cada elemento.
- Atributo Composto: formado por um ou mais sub-atributos. Nesse tipo de situação, é necessário que se analise a possibilidade de fragmentar este atributo em outros.
- Atributo Multivalorado: uma entidade possui diversos valores para este atributo.
- Atributo Determinante: identifica cada entidade de um conjunto (conhecido como atributo chave).

2.1.4 MODELO CONCEITUAL

A elaboração do desenho inicial do banco de dados, fase qual não se prende a detalhes da forma como será implementado, são utilizados modelos de dados. Um modelo de dados é uma coleção de conceitos que podem ser usados para descrever a estrutura de um banco de dados (COSTA, 2011).

Existem certos modelos conhecidos e amplamente utilizados, que são o Modelo Conceitual, o Modelo de Implementação e o Modelo Físico, sendo que para esta pesquisa, é importante que se entenda o conceito de modelagem conceitual. Este, é o mais próximo do usuário final. Entidades, atributos e relacionamentos são alguns dos conceitos utilizados (ELMASRI & NAVATHE, 2011).

Um exemplo de modelo conceitual, é o conhecido como Modelo Entidade-Relacionamento. Este, foi desenvolvido para simplificar o projeto de banco de dados, permitindo especificação e representação da estrutura e lógica geral de um banco de dados (SILBERSCHATZ & KORTH & SUDARSCHAN, 2006).

Peter Chen, em 1976, definiu o Modelo Entidade-Relacionamento, usando como base a teoria relacional criada por E.F.Cood em 1970. Segundo Chen, a visão da realidade se baseia no relacionamento entre os conceitos desta, que retratam os fatos dessa mesma realidade. Afirmou também que cada conceito (entidade ou relacionamento), pode possuir atributos (SIQUEIRA, 2016).

A figura abaixo ilustra uma aplicação dos conceitos da modelagem descrita neste capítulo, onde pessoas são relacionadas a projetos nos quais irão trabalhar. Neste, são aplicados as representações visuais contidas nas figura 2.1, e os conceitos dos subcapítulos 2.1.1, 2.1.2 e 2.1.3.

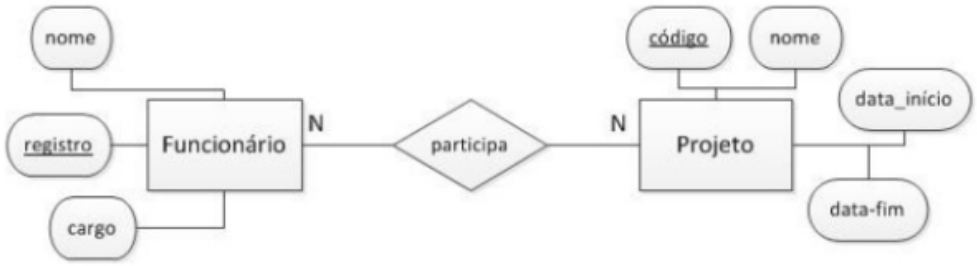


FIG. 2.2: Exemplo Modelo Entidade-Relacionamento (FIDELIX, 2019)

2.1.5 DIAGRAMA DE CLASSES

Um diagrama de classes é uma representação da estrutura e relações das classes elaboradas para o modelo. De acordo com (MELO, 2004), é um conjunto de objetos com as mesmas características, e assim, fica possível identificar objetos e agrupá-los, de forma a encontrar suas respectivas classes.

A UML (Unified Modeling Language), em diagrama de classes, é uma linguagem única, de modelagem de um software ou de uma aplicação. A UML gera artifícios visuais, que são diagramas que auxiliam na modelagem de uma classe. Sua representação se da por um retângulo com três divisões, que são: O nome da classe, seus atributos e por fim os métodos (MELO, 2004).

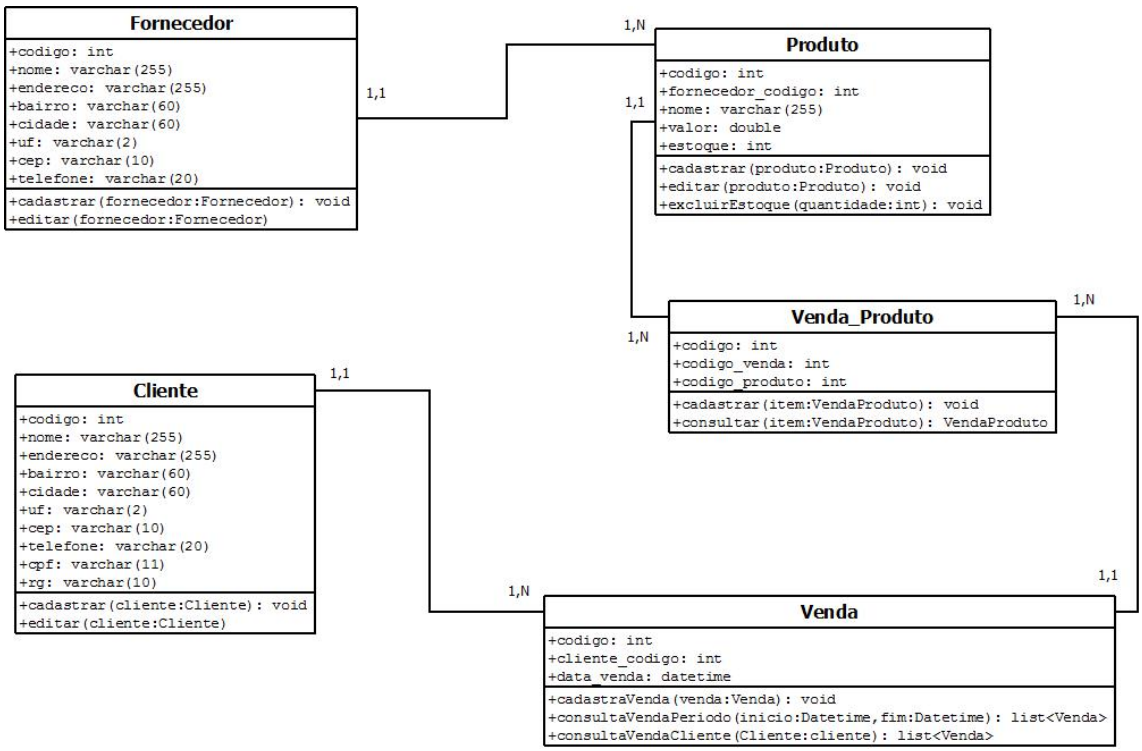


FIG. 2.3: Exemplo Aplicação UML (FONSECA, 2011)

Então, de acordo com (FONSECA, 2011), um Diagrama de Classes, pode ser definido como a representação de uma coleção de classes e seus inter-relacionamentos. A figura acima mostra um exemplo de uma aplicação do diagrama de classes UML. para o caso do caminho seguido por um produto, desde o fornecedor, até o cliente.

2.2 QGIS - MULTIPLATAFORMA DE SISTEMA DE INFORMAÇÃO GEOGRÁFICA

O software hoje chamado de QGIS, e anteriormente de Quantum GIS, é um *software* livre, que funciona como uma multiplaforma de SIG (Sistema de Informação Geográfica), que possui código-fonte aberto e licença pública geral. Foi projetado oficialmente pela OSGeo (Open Source Geospatial Foundation), e contém suporte para Linux, Unix, Mac OSX, Windows e Android (QGIS Project, 2020). É capaz de processar e interpretar inúmeros formatos de vetores, rasters e bases de dados.

O desenvolvimento inicial do programa começou no ano de 2002, e após 5 anos, passou por processo de incumbação por uma fundação ligada a projetos geoespaciais *open source*, o que possibilitou que a primeira versão fosse lançada em 2009 (QGIS Project, 2020).

Desde então, o software passou a ser ajustado para que sua interface fosse mais simples e intuitiva para o usuário. Outro objetivo adotado pelo projeto na época, foi a independência quanto ao sistema operacional utilizado.

Diferentemente dos outros *softwares* disponíveis no mercado, o uso do QGIS não requer um computador com configurações avançadas e processador de geração recente. O programa é menos exigente quanto à composição técnica das máquinas, quando comparado com similares (VITAL, 2019).

Uma das características mais marcantes do QGIS é a extensibilidade de seus recursos através de plugins. Estes, são desenvolvidos em ambiente de linguagem Python, podem possuir diversas funcionalidades de geoprocessamento e são elaborados para ter funções que o QGIS não supre nativamente.

2.2.1 DSG TOOLS - PLUGIN

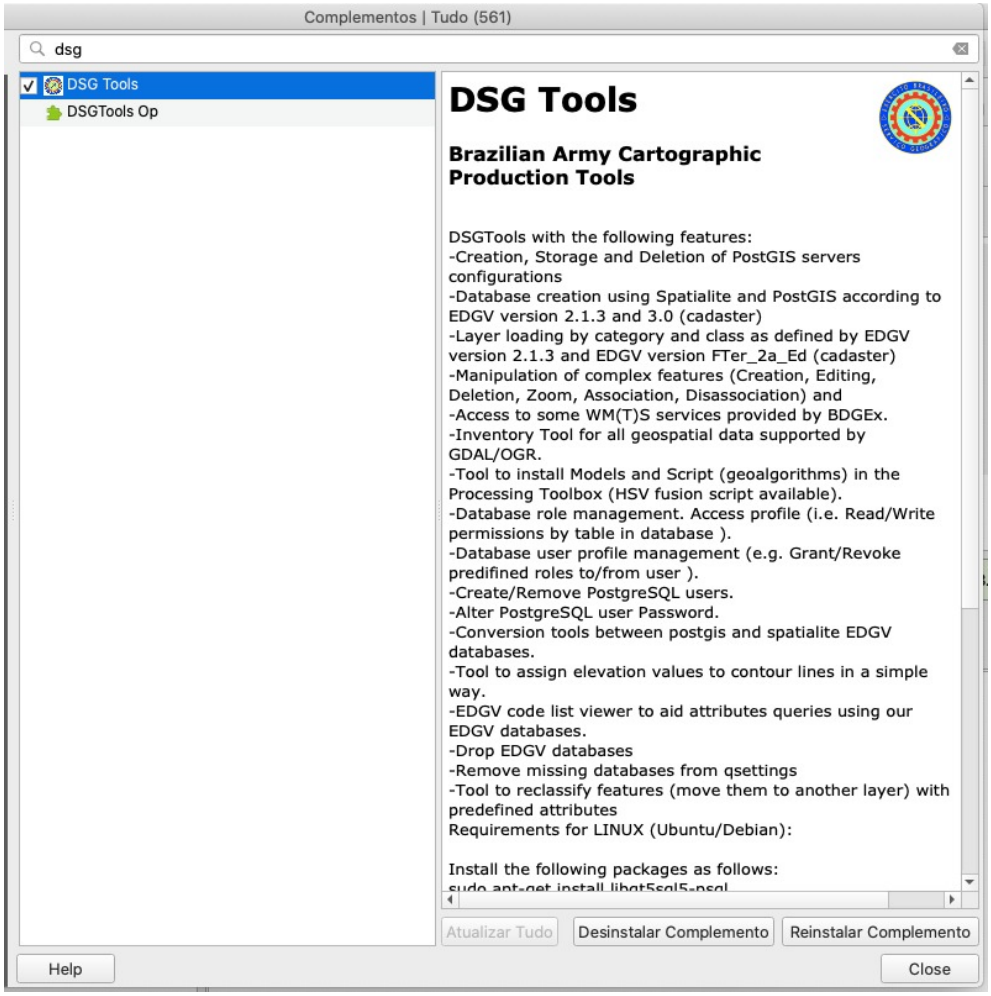


FIG. 2.4: Plugin DSG Tools no QGIS

O DSG Tools é um plugin hospedado no *software* QGIS, e é totalmente gratuito e aberto. Foi desenvolvido pela DSG, e permite a produção de dados geoespaciais conformes a INDE (Infraestrutura Nacional de Dados Espaciais). Visa atender não apenas o Exército Brasileiro, mas também produtores e usuários de geoinformação da sociedade.

O plugin possibilita as atividades de aquisição de feições vetoriais a partir de imagens matriciais. As ferramentas do plugin deverão receber atualizações para serem utilizadas em outras áreas, como por exemplo, no processamento digital de imagens, na vetorização semiautomática, na validação topológica e de atributos e na impressão de cartas (PRATES, 2015).

2.3 PYTHON

Em 1991, Guido Van Rossum lançara a primeira versão da linguagem de programação Python, e o primeiro intuito era desenvolver linguagem de scripts, aplicada ao sistema operacional Amoeba, desenvolvido pelo CWI, empresa onde trabalhava na época (SEVERANCE, 2015).

O Python é uma linguagem fracamente tipada, o que significa que não é necessário declarar tipos das variáveis. Pode ser utilizada tanto no ambiente desktop, web ou mobile (SILVA, 2019).

Para criar e desenvolver projetos, é necessário que se use programas com ambientes e funcionalidades próprias para tal função, e são conhecidos como IDE. Algumas das mais utilizadas estão: Spyder, PyCharm, Visual Studio, entre outras.

Sobre suas aplicações, onde o Python possui vantagens em relação a outras linguagens, são por exemplo, desenvolvimento Web, scripting e ciência de dados. Todos os exemplos citados, são área da computação amplamente explorados nos dias atuais, pois são aplicáveis à robótica, segurança da informação e outros assuntos relevantes para sociedade contemporânea (BORGES, 2010).

Como o objetivo do presente projeto é a utilização de um plugin disponível no *software* QGIS, e este foi desenvolvido em ambiente de programação *Python*, o conhecimento de suas funcionalidades e aplicações, é relevante para a realização do projeto.

2.4 ESTRUTURAS DE CLASSES E ATRIBUTOS DOS PADRÕES CONSIDERADOS

Conforme consta no objetivo deste documento, a finalidade realizar o mapeamento de classes e atributos do padrão EDGV 3.0 para o MGCP TRD4 v4.4. Para isso, nessa seção, serão apresentadas as formas que cada um desses padrões oferece para os usuários.

No caso da EDGV 3.0, os dados geoespaciais foram organizados nas chamadas Classes de Objetos, de acordo com semelhanças nas funcionalidades, e que são compostas por Atributos que as caracterizam. Quando agrupadas, essas classes formam Categorias de Informação (CONCAR, 2017).

A referida versão da EDGV, conta com 180 classes e 1706 atributos. Tais classes estão divididas em dois conjuntos: Mapeamento Topográfico para Pequenas Escalas (MapTopoPE) para escalas iguais ou menores que 1:25.000 e Mapeamento Topográfico para Grandes Escalas (MapTopoGE) para escalas entre 1:25.000 e 1:1.000 (CASTRO, 2019).

Como exemplo, na figura abaixo, está apresentado exemplo da categoria Hidrografia e suas classes contidas na ET-EDGV 3.0, e que possui um total de 20 (vinte) classes.

03. Hidrografia

Classe Código na RCO	1:1.000	1:10.000	1:25.000	1:50.000	1:100.000	1:250.000
Area_Umida 1.3.1	X	X	X	X	X	X
Banco_Areia 1.3.2	X	X	X	X	X	X
Barragem 1.3.3	X	X	X	X	X	X
Canal 1.3.4	X	X	X	X	X	X
Canal_Vala 1.3.5	X	X	X	X	X	X
Comporta 1.3.6	X	X	X	-	-	-
Corredeira 1.3.7	X	X	X	X	X	-
Dique 1.3.8	X	X	X	X	X	X
Fonte_Dagua 1.3.9	X	X	X	X	X	-
Foz_Maritima 1.3.10	X	X	X	X	X	X
Ilha 1.3.11	X (1)	X (1)	X (1)	X (1)	X (1)	X (1)
Massa_Dagua 1.3.12	X	X	X	X	X	X
Quebramar_Molhe 1.3.13	X	X	X	X	X	X
Queda_Dagua 1.3.14	X	X	X	X	X	X
Recife 1.3.15	X	X	X	X	X	X
Rocha_Em_Agua (2) 1.3.16	X	X	X	X	-	-
Sumidouro_Vertedouro 1.3.17	X	X	X	X	X	X
Terreno_Sujeito_ Inundacao 1.3.18	X	X	X	X	X	X
Trecho_Drenagem 1.3.19	X	X	X	X	X	X
Vala 1.3.20	X	X	X	X	X	X

FIG. 2.5: Classes de Hidrografia na EDGV 3.0 (CONCAR, 2017)

De acordo com o contido nas especificações publicadas pelo CONCAR em 2017, observa-se que cada classe possui código numérico único, com 4 dígitos, que as individualiza, e os relaciona com sua respectiva categoria.

No caso da TRD, os dados geoespaciais são arranjados de forma similar, ainda em classes e atributos, porém com suas particularidades em relação à EDGV. A primeira grande diferença entre as duas é o idioma, pois o adotado para o programa foi o inglês, enquanto que, para o padrão brasileiro, o português. Como consequência deste fato, as classes e atributos acabam por possuir nomes totalmente diferentes, dificultando o relacionamento entre os modelos.

Uma segunda diferença relevante, é que assim como na EDGV, as classes possuem códigos, cada atributo também recebe uma codificação específica, porém os atributos possuem apenas três dígitos textuais, conforme pode ser observado na figura abaixo:

Beach Area Feature

Definition	On a shore, the area on which the waves break and over which shore debris (for example: sand, shingle, and/or pebbles) accumulate. (A beach includes backshore and foreshore.)
DFDD Code	BA050
Spatial representation	Area

LIST OF FEATURE ATTRIBUTES

INHERITED ATTRIBUTES (goto supertype)

DFDD [MGCP] Code	Name	Type	Unit
[GEOM]	Spatial representation of the feature	GM_Surface	
NAM	Name	CharacterString	
NFI	Named Feature Identifier	CharacterString	
NFN	Name Identifier	CharacterString	
SMC	Surface Material Type	CodeList	

FIG. 2.6: Classes e Atributos na MGCP (MGCP TRD4 v4.4, 2016)

A referida versão do MGCP, possui 196 classes e 93 atributos, sendo que este valor não leva em conta a primitiva geométrica de cada uma delas. Após analisar o catálogo de classes e atributos da TRD4, é importante que se entenda como as nomenclaturas foram estruturadas. O primeiro dígito textual determina a qual categoria a classe pertence. A divisão acontece da seguinte forma (CASTRO, 2019):

- Classes iniciadas com o dígito textual A são da categoria que representa construções que não envolvam hidrografia.
- Classes iniciadas com o dígito textual B são da categoria de hidrografia.
- Classes iniciadas com o dígito textual D são da categoria que representa elementos fisiográficos.
- Classes iniciadas com o dígito textual E são da categoria que representa vegetação.
- Classes iniciadas com o dígito textual F são da categoria que representa campos e estandes de tiro.
- Classes iniciadas com o dígito textual G são da categoria que representa objetos geográficos aeroespaciais.
- Classes iniciadas com o dígito textual S são da categoria que representa instalações militares.
- Classes iniciadas com o dígito textual Z são da categoria que representa vazio cartográfico e nome local.

Com essas informações acerca das estruturas de cada um dos padrões considerados, é possível perceber que existem diferentes números de atributos e classes entre estes, assim

como as escalas consideradas. Estas diferenças, influenciam diretamente na correpondência entre os mapeamentos considerados, podendo conter classes e atributos na EDGV 3.0 sem representação existente na MGCP TRD4 v4.4.

Segue abaixo uma tabela resumo de comparação entre os padrões considerados, que identifica as principais diferenças apontadas.

Informação	MGCP TRD4 v4.4	ET-EDGV 3.0
Idioma	Inglês	Português
# Classes	196	180
# Atributos	93	1706
Codificação de Classes	2 dígitos textuais + 3 dígitos numéricos	Código numérico de até 4 dígitos
Codificação dos Atributos	3 dígitos textuais	Sem regra específica
Escalas	1:50.000 e 1:100.000	Possui especificações para escalas grandes e pequenas

FIG. 2.7: Tabela de Comparação MGCP TRD4 v4.4 x ET-EDGV 3.0

2.5 CONVERSÕES DE MODELO

Ao longo da realização dessa pesquisa, foram utilizados produtos gerados em outros projetos como forma de embasamento e auxílio ao desenvolvimento da pesquisa. Os produtos utilizados foram um mapeamento de classes e atributos entre dois modelos, e um plugin de conversão, previamente desenvolvido.

O mapeamento de classes e atributos utilizado, foi o produto gerado na pesquisa "Mapeamento Entre as Especificações MGCP TRD4 v4.4 e ET-EDGV 3.0", elaborado por (CASTRO, 2019). A finalidade da pesquisa foi elaborar um arquivo de mapeamento que permita, de forma estruturada, a conversão entre as especificações da TRD4 v4.4 e a ET-EDGV 3.0 (CASTRO, 2019).

O plugin de conversão utilizado foi o *EDGV Converter*, desenvolvido por (SANTOS & WOLFGRAM & ALMEIDA, 2019), hospedado no *software* QGIS. Este plugin realiza a conversão de dados entre a EDGV versão 2.1.3 e a EDGV versão 3.0, de modo que o usuário possa realizar a conversão entre estas versões da EDGV em seu próprio banco de dados em um tempo reduzido e com o uso de softwares livres. A estrutura de código previamente existente no plugin foi usada como base para o desenvolvimento do proposto nesta pesquisa.

2.6 PLUGIN - PARTE COMPUTACIONAL

A parte computacional da presente pesquisa envolveu a utilização de um plugin, hospedado no repositório do QGIS, que realize a conversão de um banco de dados com classes e atributos no padrão da ET-EDGV 3.0, para o padrão adotado no TRD4 v4.4 da MGCP.

Para que não fosse necessário o desenvolvimento completo de um novo código, foi utilizado como base o plugin chamado *EDGV Converter*, desenvolvido por (SANTOS & WOLFGRAM & ALMEIDA, 2019). O código foi adaptado para interpretar os arquivos JSON de conversão entre os modelos EDGV 3.0 e TRD4 utilizado.

2.6.1 PLUGIN *EDGV CONVERTER*

2.6.1.1 INTERFACE DO PLUGIN

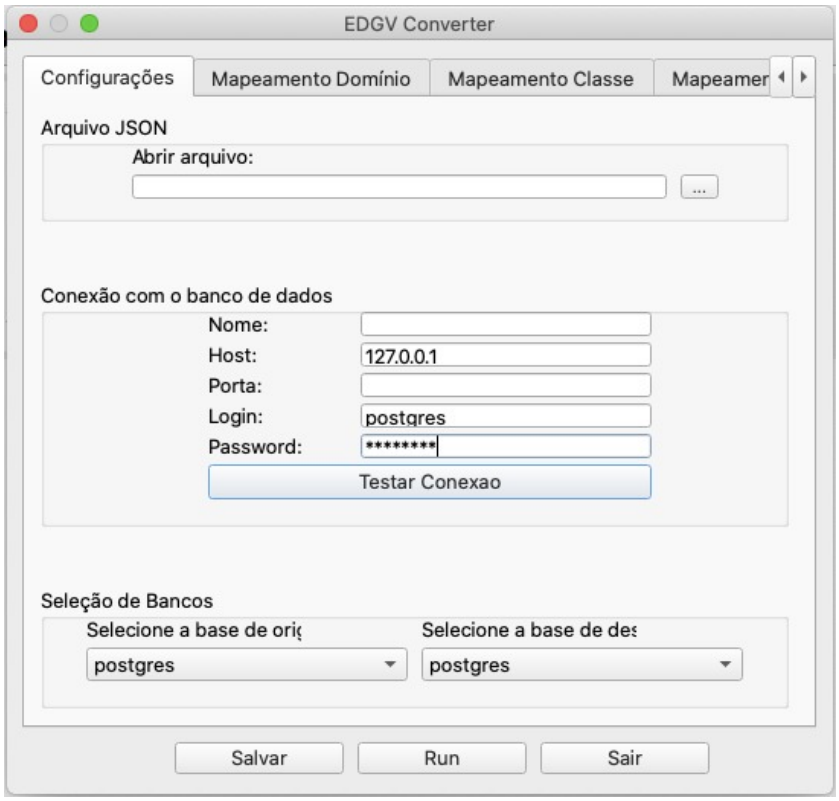


FIG. 2.8: Interface Plugin *EDGV Converter*

A interface deste plugin possui 4 (quatro) abas. Na aba de configurações, é possível incluir o arquivo JSON que será usado na conversão, assim como estabelecer a conexão com o banco de dados contendo as informações a serem convertidas.

Ainda na aba configurações, é possível escolher em qual padrão está a base de origem, e qual será a base de destino a ser considerada para a conversão.

As outras 3 (três) abas são destinadas à realização do mapeamento entre as classes e atributos das bases consideradas. No subcapítulo a seguir, estão explicados os funcionamentos das atividades relacionadas com cada aba.

2.6.1.2 FUNCIONAMENTO DO PLUGIN

Primeiramente, é importante que se saiba o que é necessário que o usuário tenha em posse para que seja possível a utilização do plugin. São 6 (seis) requisitos a serem atendidos (SANTOS & WOLFGRAM & ALMEIDA, 2019):

- *Software* QGIS versão 3.0 ou mais recente
- Python 3
- Versão 9.0 ou superior do PostgreSQL
- PostGIS
- Qt Designer
- Deve possuir 2 bancos de dados com as estruturas dos modelos a serem convertidos de forma automática. No caso do plugin original, a conversão é feita entre as versões 2.1.3 e 3.0 da ET-EDGV

Checados os 6 (seis) requisitos acima, é preciso entender como é o funcionamento do *plugin*, que obedece um sequenciamento de atividades que explicam o passo-a-passo para utilização deste.

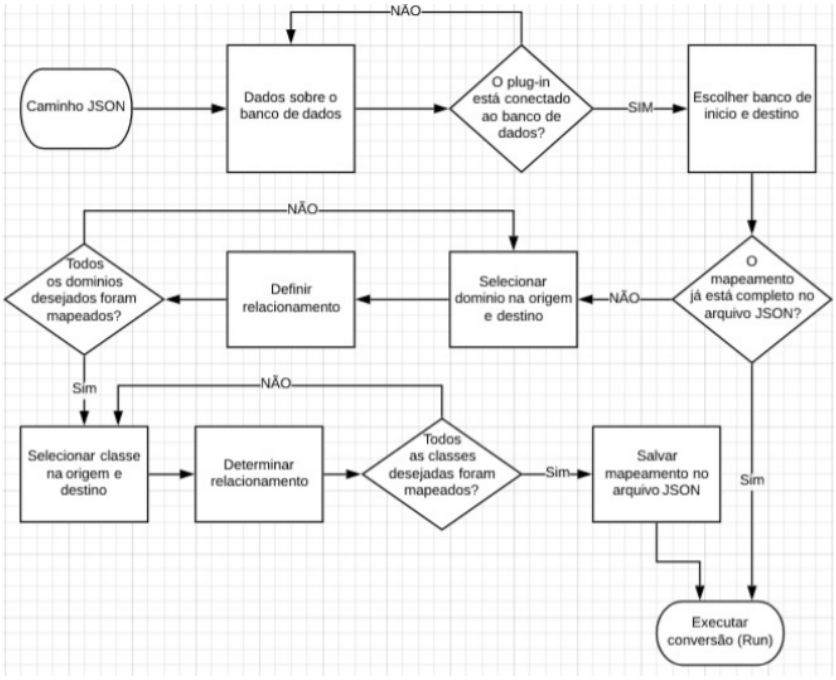


FIG. 2.9: Sequenciamento de atividades do *plug-in* (SANTOS & WOLFGRAM & ALMEIDA, 2019)

De acordo com a figura acima, é possível que o usuário utilize arquivos JSON com mapeamentos prontos, pois o *plugin* é capaz de fazer as alterações no arquivo fornecido, e torna possível fazer mapeamentos em partes, sem a necessidade de diversas versões do arquivo original. Esta funcionalidade facilita a realização de mapeamentos extensos.

A interface original, contém 4 (quatro) abas, onde cada uma possui funcionalidades a serem realizadas, que estão relacionadas abaixo (SANTOS & WOLFGRAM & ALMEIDA, 2019):

- A aba "Configurações"possui 3 atividades a serem realizadas, e são elas: seleção do arquivo JSON a ser utilizado, teste de conexão com banco de dados e determinar das bases de dados de origem e destino
- A aba "Mapeamento Domínio"possibilita mapear algo que não foi contemplado no arquivo JSON inserido.

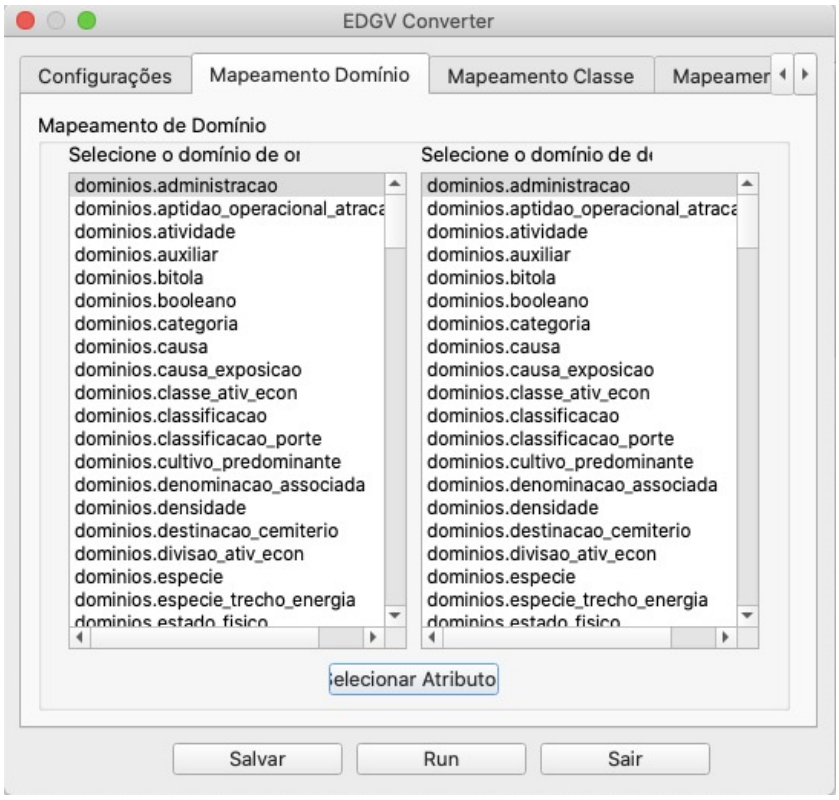


FIG. 2.10: Aba Mapeamento Domínio

Após o clique em "Selecionar Atributo", irá abrir uma janela chamada "Domínio", onde é possível que o usuário faça a relação entre os valores do domínio da base de origem para base de destino.

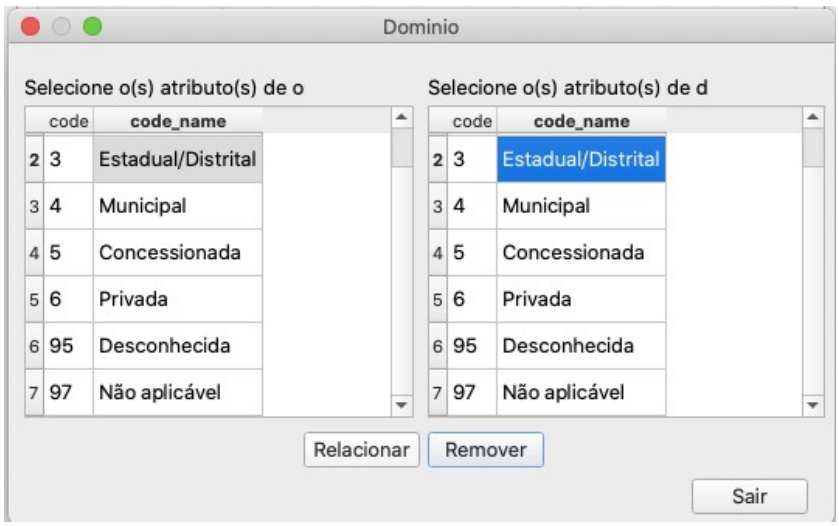


FIG. 2.11: Janela Domínio

- Na aba "Mapeamento de Classe", o usuário deverá fazer o relacionamento de classes, onde irá selecionar uma classe no campo de origem e sua correspondente no de

destino.

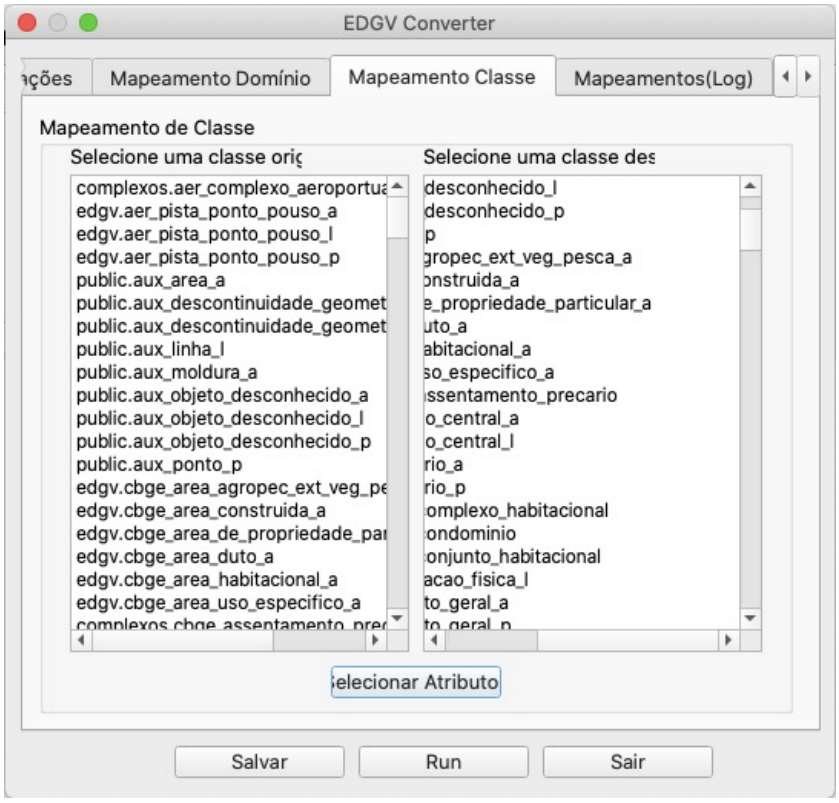


FIG. 2.12: Aba Mapeamento de Classe

Após clicar em "Selecionar Atributos", abrirá uma janela chamada "Atributos", onde serão relacionados os atributos que compõem as classes.

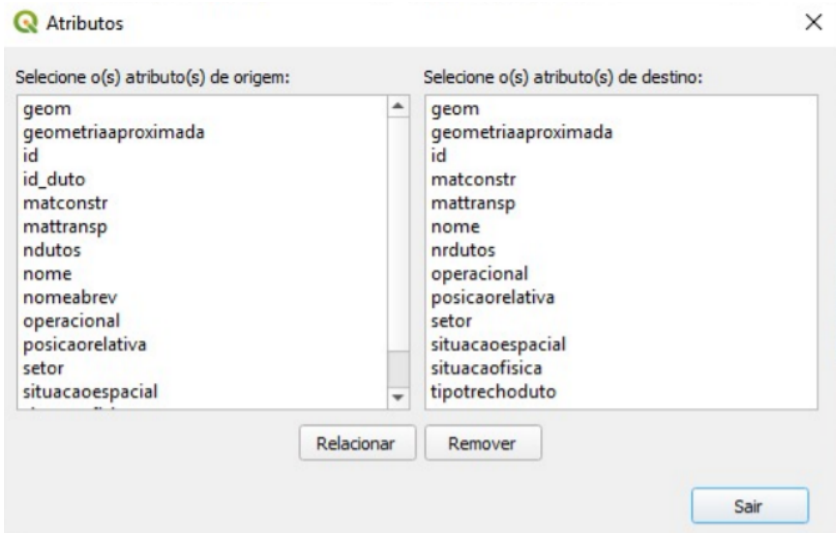


FIG. 2.13: Janela Atributos

■ A quarta e última aba é a chamada "Mapeamentos (Log)", que é a tela de registros,

onde o usuário pode checar os mapeamentos previamente executados. Essa aba auxilia na utilização do *plugin*.

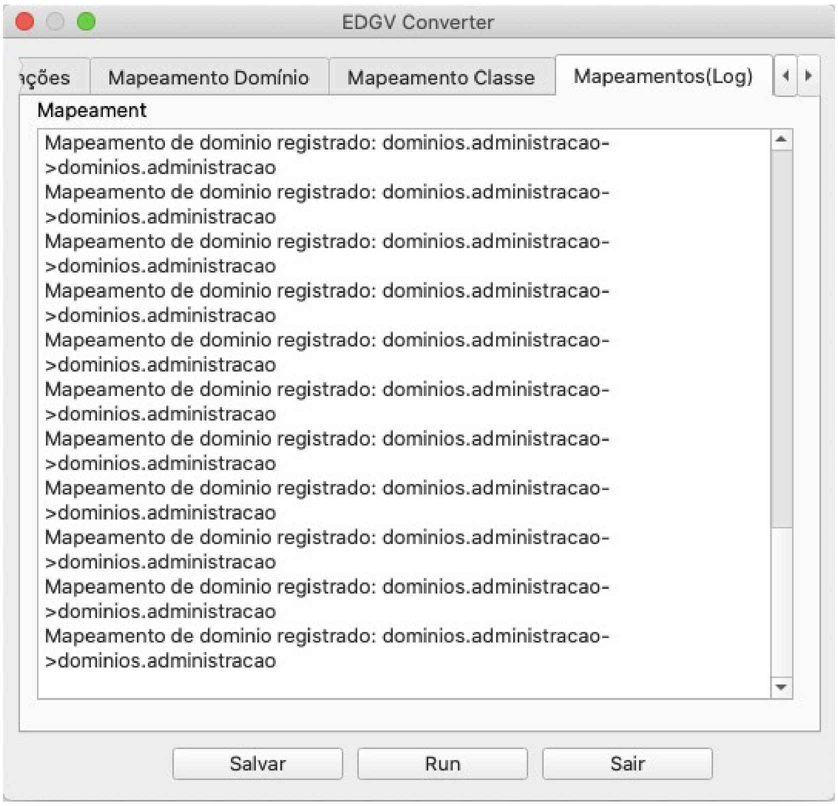


FIG. 2.14: Aba Mapeamentos (Log)

2.7 ARQUIVO JSON UTILIZADO

2.7.1 FORMATO JSON (*JAVASCRIPT OBJECT NOTATION*)

O JSON é um formato baseado em texto que possibilita a troca de dados independente da linguagem de programação utilizada (INTERNACIONAL, 2013). Quando se esta analisando troca de informações entre sistemas, que é o caso do presente projeto, onde foram convertidas classes e atributos no padrão da ET-EDGV 3.0 para o padrão TRD4, é evidente a importância da existência dessa tecnologia.

As estruturas deste formato, são flexíveis, e permitem trabalhar com dados sem a necessidade de se definir esquemas rígidos. No geral, um arquivo JSON obedece uma sequência de *tokens* do padrão *Unicode*. As estruturas existentes no formato são os colchetes ([ ]), chaves ({ }), dois pontos (:) e vírgula (,), além de *true*, *false* e *null* (MACHADO, 2017).

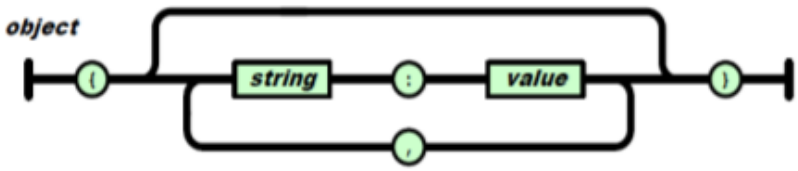


FIG. 2.15: Esquema de Funcionamento Formato JSON (INTERNACIONAL, 2013)

2.7.2 DESCRIÇÃO DO ARQUIVO

Para a utilização do plugin descrito na fundamentação teórica, foi usado um arquivo JSON que contém a conversão entre os padrões da EDGV 3.0 e TRD4 v4.4 do MGCP.

O arquivo no formato JSON foi desenvolvido por (DINIZ, 2019), disponível no plugin *DSG Tools*. Este, foi utilizado como base para a elaboração do mapeamento de classes e atributos dos padrões em questão, através do uso da funcionalidade "Mapeamento Manual"do plugin *EDGV Converter*.

A realização do mapeamento manual foi necessária, pois o plugin foi desenvolvido para receber arquivos JSON com estrutura distinta do produto gerado por (DINIZ, 2019). Com isso, o arquivo de saída do mapeamento manual possui a estrutura necessária para ser usado na etapa de testes, explicada nos capítulos subsequentes.

FIG. 2.16: Arquivo JSON utilizado

O arquivo completo pode ser encontrado no GitHub usado como repositório para os arquivos gerados por essa pesquisa, através do seguinte link: *https://github.com/thiago-mpm/pfc_2020_MB*.

O mapeamento é dividido em duas partes: mapeamento de classes e mapeamento de atributos. Sobre o de atributos, este permite mudar o nome deles, de forma independentemente das classes. Realiza também a tradução de valores dos atributos (DINIZ, 2019).

O arquivo JSON foi estruturado de forma que são definidas classes de ida e de volta, para que seja possível realizar a conversão entre as bases. São também definidos os atributos pertencentes a cada uma das classes a serem convertidas, conforme segue na figura abaixo:

```
"classe_ida": "al020",
"classe_volta": "lml_area_densamente_edificada",
"filtro_ida": {
  "$and": [
    {
      "$or": [
        {
          "nome_atributo": "fuc",
          "valor": 0
        },
        {
          "nome_atributo": "fuc",
          "valor": 4
        },
        {
          "nome_atributo": "fuc",
          "valor": 19
        }
      ]
    },
    {
      "nome_atributo": "$GEOM_TYPE",
      "valor": "POLYGON"
    }
  ]
},
"atributos_default_volta": [
  {
    "nome_atributo": "bac",
    "valor": 0
  },
  {
    "nome_atributo": "fuc",
    "valor": 0
  },
  {
    "nome_atributo": "fun",
    "valor": 0
  },
  {
    "nome_atributo": "ord",
    "valor": 0
  }
```


FIG. 2.17: Classes e atributos de ida e volta no arquivo JSON

Determinada a classe a qual será convertida a que está sendo analisada, são então preenchidos os atributos destas, e o arquivo em JSON contém o mapeamento entre as nomenclaturas e valores de suas características.

```
"classe_ida": "bd120",
"classe_volta": "hid_recife",
"mapeamento_atributos": [
  {
    "attr_ida": "wle",
    "attr_volta": "situacaoemagua",
    "traducao": [
      {
        "valor_ida": 1,
        "valor_volta": 0,
        "sentido": "ida"
      },
      {
        "valor_ida": 2,
        "valor_volta": 4
      },
      {
        "valor_ida": 3,
        "valor_volta": 5
      },
      {
        "valor_ida": 4,
        "valor_volta": 7
      },
      {
        "valor_ida": 1,
        "valor_volta": 0,
        "sentido": "ida"
      },
      {
        "valor_ida": 8,
        "valor_volta": 0,
        "sentido": "ida"
      },
      {
        "valor_ida": 998,
        "valor_volta": 0,
        "sentido": "ida"
      },
      {
        "valor_ida": 999,
        "valor_volta": 0,
        "sentido": "ida"
      }
    ]
```


FIG. 2.18: Tradução de atributos no arquivo JSON

Conforme previamente abordado neste texto, os padrões da EDGV 3.0 e TRD4 possuem inúmeras diferenças, e portanto, é necessário tratar dos atributos que não possuem conversão direta.

2.7.2.1 CHAVEAMENTOS RELEVANTES

Dentro do arquivo JSON, existe o chaveamento chamado de *"atributo_ default"*, que trata dos atributos que não estão no mapeamento, ou seja, que não trocam de valor. Para cada

um dos sentidos da conversão, de ida ou de volta, há uma listagem dos atributos com os respectivos valores que são preenchidos por *default* quando executada a conversão.

```
{
  "__comment": "BB190 BERTHING_STRUCTURE <-> hdv_atracadouro_terminal",
  "classe_ida": "bb190",
  "classe_volta": "hdv_atracadouro_terminal",
  "atributos_default_ida": [
    {
      "nome_atributo": "administracao",
      "valor": 0
    },
    {
      "nome_atributo": "matconstr",
      "valor": 0
    },
    {
      "nome_atributo": "operacional",
      "valor": 0
    },
    {
      "nome_atributo": "aptidaooperacional",
      "valor": 9999
    }
  ],
  "atributos_default_volta": [
    {
      "nome_atributo": "fac",
      "valor": 0
    },
    {
      "nome_atributo": "wid",
      "valor": -32767.0
    },
    {
      "nome_atributo": "wle",
      "valor": 0
    }
  ],
```

FIG. 2.19: Atributos Default

São tratados também os mapeamentos múltiplos, ou seja, quando "n"atributos influenciam o preenchimento de "m", sendo "n"ou "m"números diferentes de 1. No caso do exemplo abaixo, o atributo "ppo"com valor 86, implica no preenchimento dos atributos “tipoproduto” e “nome” com os valores de “99” e “quartzo” respectivamente.

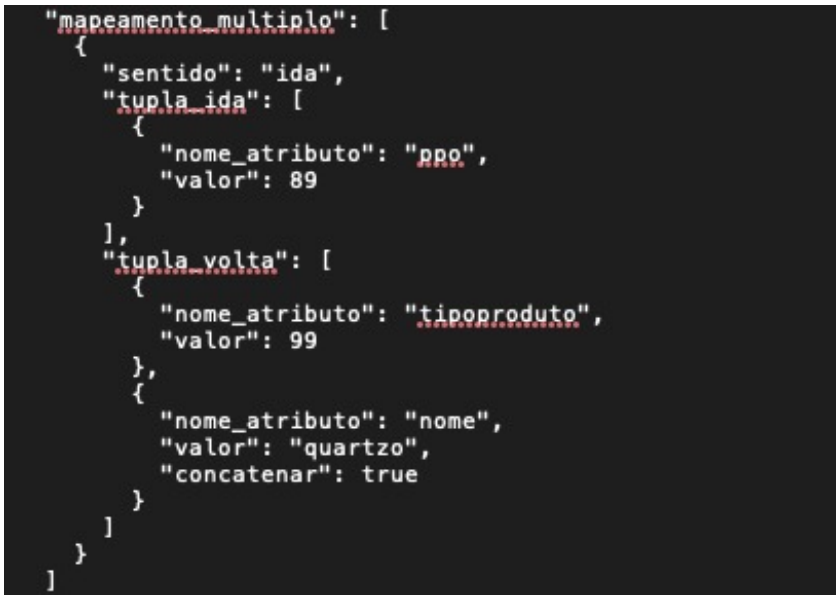


FIG. 2.20: Mapeamento Múltiplo

Os chaveamentos *"filtro_ida/filtro_volta"*, são chaves que fazem o mapeamento de um subconjunto dos dados a serem convertidos, filtrando informações no sentido indicado no filtro.

FIG. 2.21: Filtros de Ida e de Volta

3 METODOLOGIA

3.1 TENTATIVA INICIAL

Como ponto de partida, com o intuito de compreender como acontece o relacionamento entre as classes e atributos da EDGV 3.0 e TRD4 v4.4 do MGCP, contruiu-se um arquivo *Excel* com o mapeamento de atributos destes padrões.

Neste arquivo foram criadas 8 (oito) colunas, que possibilitam compreender a conversão direta de atributos entre os padrões considerados. Segue descrição das colunas contidas no arquivo:

- Código Classe EDGV: contém o código de identificação das classes da EDGV
- Classe EDGV: contém o nome das classes da EDGV
- Código Atributo: contém o código de identificação dos atributos da EDGV
- Atributo EDGV: contém o nome dos atributos da EDGV
- Tipo EDGV: contém a definição do tipo dos atributos da EDGV
- Código Classe MGCP: contém o código de identificação das classes do MGCP
- Classe MGCP: contém o nome das classes do MGCP
- Atributo MGCP: contém o nome dos atributos do MGCP

Código Classe EDGV	Classe EDGV	Código Atributo	Atributo EDGV	Tipo EDGV	Código Classe MGCP	Classe MGCP	Atributo MGCP
1.02	curso_dagua	1.02.1	nome	Alfanumérico	BH140	River Feature	NAM
1.05	Trecho_massa_dagua	1.05.4	regime	Alfanumérico	BH140	River Feature	HYP
1.05	Trecho_massa_dagua	1.05.1	nome	Alfanumérico	BH140	River Feature	NAM
1.06	Limite_massa_dagua	1.06.3	materialPredominante	Alfanumérico	BA010	Land Water Boudary Line Feature	SLT
1.09	Barragem	1.09.1	nome	Alfanumérico	BI020	Dam	NAM
1.09	Barragem	1.09.3	matConstr	Alfanumérico	BI020	Dam	MCC
1.09	Barragem	1.09.5/1.09.6	operacional/situacaoFisica	Alfanumérico	BI020	Dam	FUN
1.11	Semidouro_Vertedouro	1.11.4	causa	Alfanumérico	BH145	Vanishing Point Feature	WST
1.12	Queda_dagua	1.12.1	nome	Alfanumérico	BH180	Waterfall Feature	NAM
1.17	Corredeira	1.17.1	nome	Alfanumérico	BH120	Rapids Line Feature	NAM
1.17	Corredeira	1.17.2	geometriaAproximada	Booleano	BH120	Rapids Line Feature	GEOM
1.19	Ilha	1.19.1	nome	Alfanumérico	BA030	Island Area Feature	NAM
1.19	Ilha	1.19.2	geometriaAproximada	Booleano	BA030	Island Area Feature	GEOM
1.20	Rocha_Em_Agua	1.20.1	nome	Alfanumérico	BD130	Hazardous Rock Point Feature	NAM
1.20	Rocha_Em_Agua	1.20.2	geometriaAproximada	Booleano	BD130	Hazardous Rock Point Feature	GEOM
1.20	Rocha_Em_Agua	1.20.3	situacaoEmAgua	Alfanumérico	BD130	Hazardous Rock Point Feature	WLE
1.21	Recife	1.21.1	nome	Alfanumérico	BD120	Reef Area Feature	NAM
1.21	Recife	1.21.2	geometriaAproximada	Booleano	BD120	Reef Area Feature	GEOM
1.21	Recife	1.21.4	situaMare	Alfanumérico	BD120	Reef Area Feature	WLE
1.21	Recife	1.21.4	situacaoCosta	Alfanumérico	BD120	Reef Area Feature	COD

FIG. 3.1: Arquivo *Excel* com mapeamento de atributos

Esta primeira tentativa de mapeamento mostrou ser importante para levantar questionamentos acerca de atributos que não possuem conversão direta, como foram tratados no arquivo JSON de referência e qual o impacto destes nos resultados dos testes realizados.

3.2 BANCOS DE DADOS UTILIZADOS

3.2.1 BANCO DE DADOS TRD4 V4.4

O banco de dados da TRD4 v4.4 utilizado, foi desenvolvido pela DSG em SQL (Structured Query Language), que é uma linguagem computacional destinada a desenvolvimento de bancos de dados. As tabelas necessárias foram inseridas conforme segue abaixo:

```
CREATE TABLE domains.tipo_veg (
        code smallint NOT NULL,
        code_name text NOT NULL,
        CONSTRAINT tipo_veg_pk PRIMARY KEY (code)
);

INSERT INTO domains.tipo_veg (code,code_name) VALUES (101,'Extração mineral');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (102,'Brejo (sem árvores, pouca humidade)');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (103,'Brejo (sem árvores, muita humidade)');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (104,'Pântano (com árvores, muita humidade)');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (105,'Arroz');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (106,'Terreno exposto');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (107,'Vegetação cultivada');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (108,'Pomar');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (109,'Vinhedo');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (110,'Lúpulo');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (111,'Campo');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (112,'Cana');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (113,'Arbustos');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (114,'Floresta');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (115,'Desmatamento');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (116,'Mangue');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (117,'Clareira');
INSERT INTO domains.tipo_veg (code,code_name) VALUES (9999,'A SER PREENCHIDO');
```

FIG. 3.2: Parte do Código do Banco de Dados TRD4 v4.4 Utilizado

Para que se possa utilizar este no plugin, é preciso que seja feito o *upload* no pgAdmin, que é um software gráfico para administração de banco de dados.

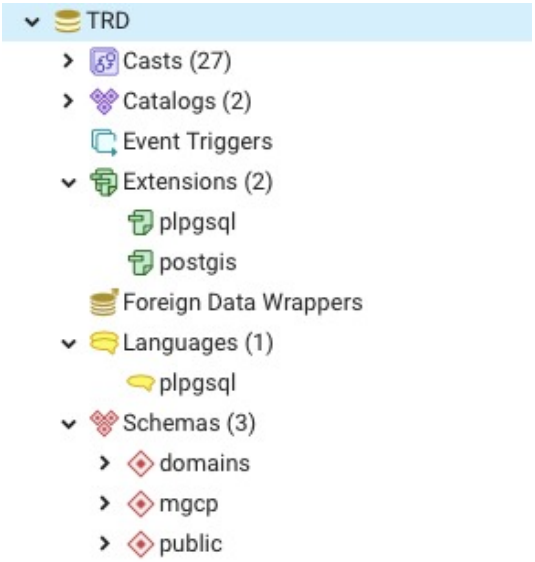


FIG. 3.3: Upload do Banco de Dados no PgAdmin

Após feito o *upload* do banco de dados no pgAdmin, é possível fazer o login através do *plugin*, e realizar o carregamento do banco de dados.

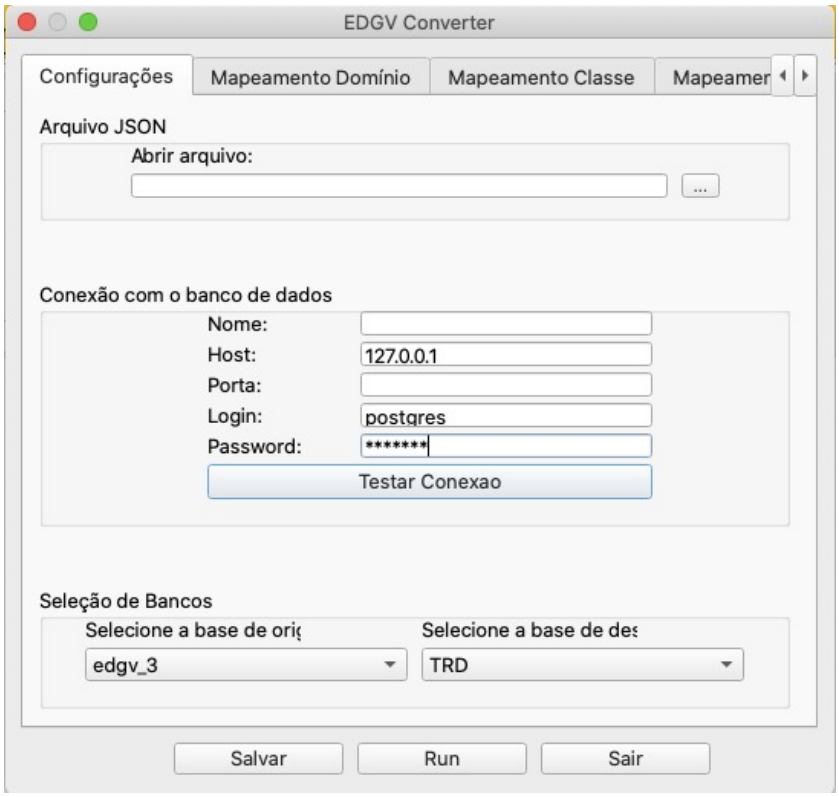


FIG. 3.4: Banco de Dados TRD4 Carregado no Plugin

Para o desenvolvimento deste banco de dados, foram criadas 392 tabelas, separadas em 3 *schemas*, e está disponivel no repositório do GitHub, no link *https://github.com/thiago-*

mpm/pfc_2020_MB.

3.2.2 BANCO DE DADOS ET-EDGV 3.0

Inicialmente, foi criado um banco de dados em EDGV 3.0, que estava inicialmente vazio, com o uso do plugin *DSG Tools*, e carregado no pgAdmin. Durante a realização das atividades, foi constatada a existência de um erro na estrutura desse banco de dados.

Como exemplificado abaixo, atributos como "matconstr", constam no arquivo JSON de referência, entretanto, não estavam contidos no SQL do banco de dados, o que dificulta a identificação do domínio.

Quando comparado com o banco de dados obtido a posteriori com a DSG, todos os atributos descritos no aqruivo JSON puderam ser encontrados.


```
"__comment": "BI020 DAM <-> hid_barragem",
"classe_ida": "bi020",
"classe_volta": "hid_barragem",
"atributos_default_ida": [
  {
    "nome_atributo": "usoprincipal",
    "valor": 0
  },
  {
    "nome_atributo": "operacional",
    "valor": 0
  }
],
"atributos_default_volta": [
  {
    "nome_atributo": "hgt",
    "valor": -32767.0
  },
  {
    "nome_atributo": "trs",
    "valor": 0
  },
  {
    "nome_atributo": "wid",
    "valor": -32767.0
  }
],
"mapeamento_atributos": [
  {
    "attr_ida": "mcc",
    "attr_volta": "matconstr",
    "traducao": [
      {
        "valor_ida": 0,
        "valor_volta": 0
```


FIG. 3.5: Código SQL do Banco de Dados Obtido com a DSG

Após realizar a conexão com o banco de dados através do pgAdmin, é possível visualizar informações acerca deste:

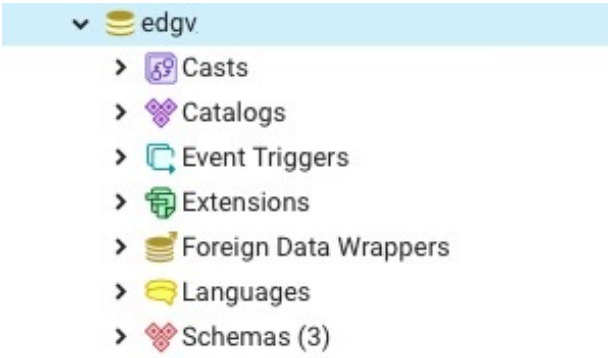


FIG. 3.6: Upload do Banco de Dados no PgAdmin

Este conta com 432 tabelas, separadas em 3 *schemas*. Após realizar a conexão, é possível seleciona-lo como base de origem ou destino no plugin. O banco de dados está disponível no repositório do GitHub, no link *https://github.com/thiago-mpm/pfc_2020_MB*.

3.3 MAPEAMENTO DE ATRIBUTOS

Para realizar o mapeamento de atributos, foram definidas duas linhas de estratégia, sendo a primeira chamada de “automática” e a segunda de “manual”.

Na linha automática, o intuito é adaptar o código do plugin de forma que se possível a utilização do arquivo JSON no seu formato original. Para isso, foi analisada a estrutura do código do plugin *"EDGV Converter"*, e notou-se que este cria inúmeros arquivos JSON, e depois o carregava. Se faz necessário criar arquivos JSON com a conversão relativa a cada classe existente, e para adaptar o arquivo JSON de referência, iria requerer a separação em outros menores.

O método “manual”, consiste em realizar o mapeamento pelo próprio plugin. Na aba "Configurações", são selecionados os banco de dados nos formatos desejados, que no caso da pesquisa, "EDGV 3.0"no campo de base de origem, e "TRD"na base de destino. Desta forma, existem dois mapeamentos a serem feitos, o Mapeamento de Domínio e o de Classe.

Para o Mapeamento de Classe, foram realizados os seguintes passos:

- Encontra-se uma conversão entre duas classes indicada no JSON de referência

```
"__comment": "BD120 REEF <-> hid_recife",
"classe_ida": "bd120",
"classe_volta": "hid_recife",
"mapeamento_atributos": [
  {
    "attr_ida": "wle",
    "attr_volta": "situacaoemagua",
    "traducao": [
      {
        "valor_ida": 1,
        "valor_volta": 0,
        "sentido": "ida"
      },
      {
        "valor_ida": 2,
        "valor_volta": 4
      },
      {
        "valor_ida": 3,
        "valor_volta": 5
      },
      {
```


FIG. 3.7: Conversão de Classes no Arquivo

■ Seleção das classes na aba "Mapeamento Classe"

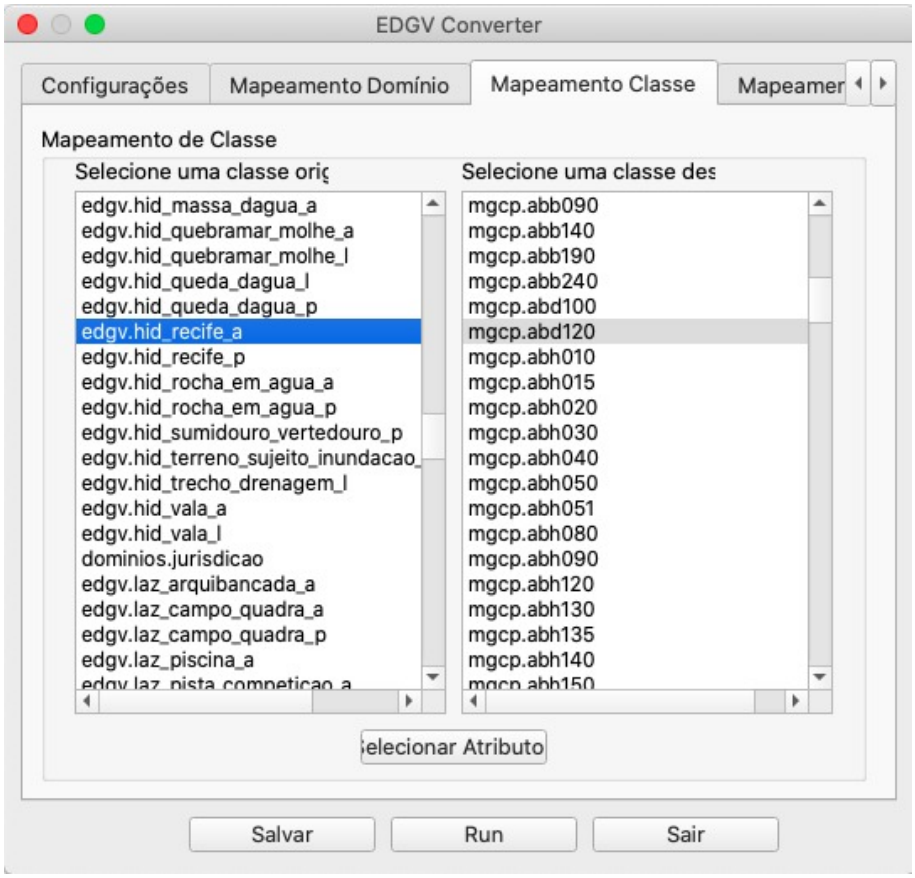


FIG. 3.8: Mapeamento de Classes no Plugin

■ Relacionar os atributos contidos na chave "mapeamento_atributos" do arquivo JSON

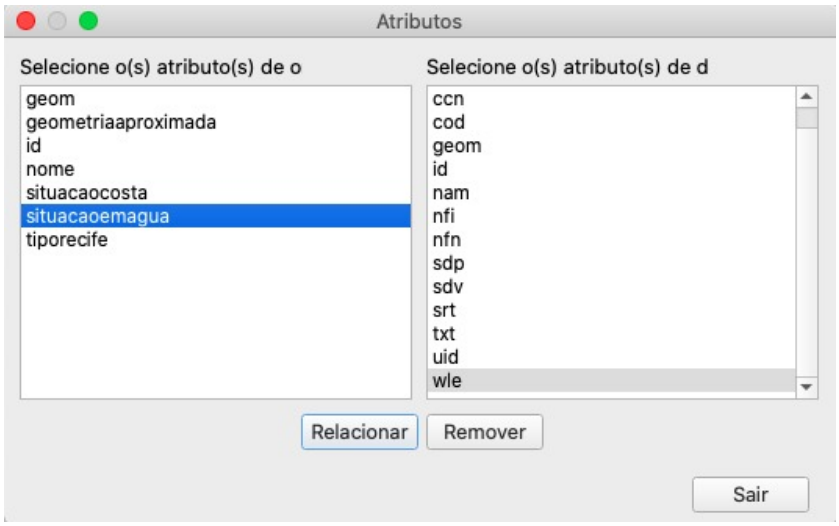


FIG. 3.9: Mapeamento de Atributos no Plugin

Durante esse processo, criou-se o arquivo .txt "Mapeamentos Encontrados", onde encontram-se as classes que possuem mapeamento direto entre os padrões EDGV 3.0 e TRD4 v4.4. Encontra-se disponivel no repositório do GitHub, no link *https://github.com/thiago-mpm/pfc_2020_MB*.

O procedimento para o Mapeamento de Domínios é similar, onde usam-se o mesmo arquivo JSON e o banco de dados EDGV 3.0 que o Mapeamento de Classes. Os passos foram os seguintes:

- Identificar a classe no chaveamento "mapeamento_atributos" a se realizar mapeamento de domínios de cada um de seus atributos
- Localizar os domínios dos atributos da classe selecionada no código SQL através do pgAdmin

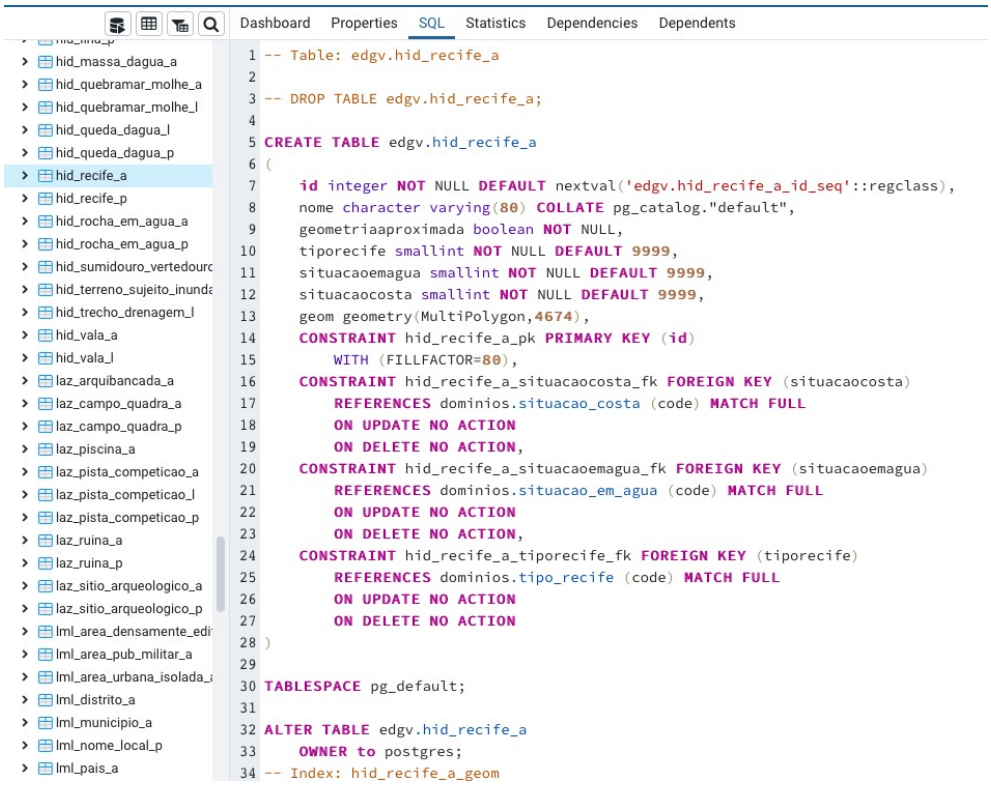


FIG. 3.10: Domínios dos Atributos no pgAdmin

■ Seleção dos domínios na aba "Mapeamento Domínio"do plugin

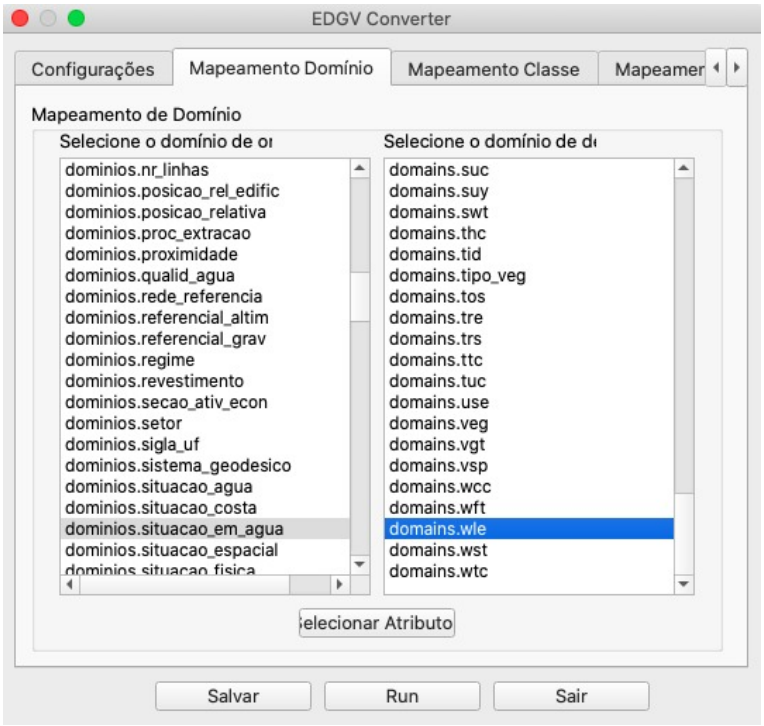


FIG. 3.11: Seleção de Domínios

- Seleção dos atributos que se convertem

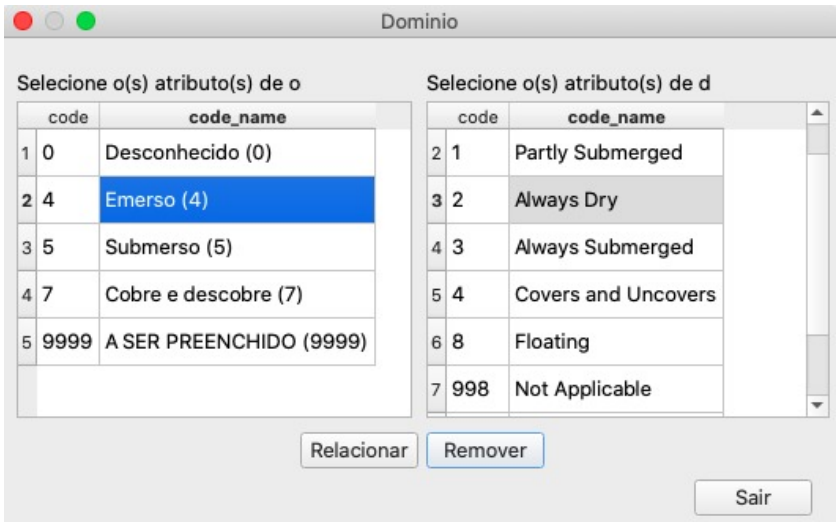


FIG. 3.12: Seleção de Atributos

Durante esse processo, criou-se o arquivo .txt "Mapeamentos de Domínios", onde encontram-se os domínios mapeados. Encontra-se disponivel no repositório do GitHub, no link *https://github.com/thiago-mpm/pfc_2020_MB*.

Após concluídas essas etapas, um arquivo JSON no formato de utilização do plugin foi gerado, de forma que é possível utilizá-lo para realizar conversões.

3.4 TESTES PLANEJADOS E MÉTRICAS DE AVALIAÇÃO

3.4.1 TESTES PLANEJADOS

Para a execução dos testes, utilizou-se um computador, no qual estavam instalados: Python 3, o *software* QGIS, PostgreSQL e PostGIS. Foram elaborados 3 (três) testes, com objetivos distintos. São eles:

- Teste com Camadas Escolhidas: trata-se de um teste mais direto
- Teste com Todas as Feições: trata-se de um teste mais forte, pois abrange todas as classes e atributos da EDGV 3.0
- Teste com Carta: trate-se de teste prático

Nos subcapítulos a seguir, serão detalhados cada um destes, com imagens e fluxogramas para maior compreensão acerca dos objetivos propostos dos testes.

3.4.1.1 TESTE COM CAMADAS ESCOLHIDAS

Neste, o foco são as camadas e atributos efetivamente mapeadas de maneira direta, criando feições somente com atributos e classes que constam no mapeamento direto. Após realizado o mapeamento, são destacadas as classes e seus atributos e domínios mapeados diretamente.

Conecta-se o QGIS com o pgAdmin, de forma que seja possível a importação de informações de banco de dados. Após criada a conexão, no lado esquerdo são selecionadas a opção destacada, então, e o banco de dados desejado (no caso, "Teste"). A seguir, escolhe-se o "schema" e por último carrega-se a camada desejada com um duplo clique.

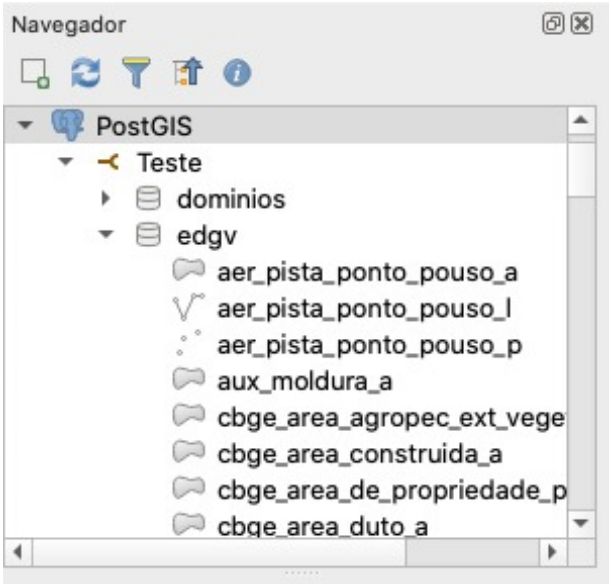


FIG. 3.13: Conexão do Banco de Dados com o QGIS

A partir deste ponto, basta editar as camadas escolhidas adicionando feições e escolhendo os atributos

Pelo mapeamento de atributos, faz-se um controle de toda a parte criada e os valores deles. Após executada a conversão no plugin, é realizado processo similar para o carregamento das camadas de destino previstas e a análise quanto á presença de todas as feições carregadas para a conversão. São então estudados os resultados da conversão de modo a identificar e contabilizar as saídas.

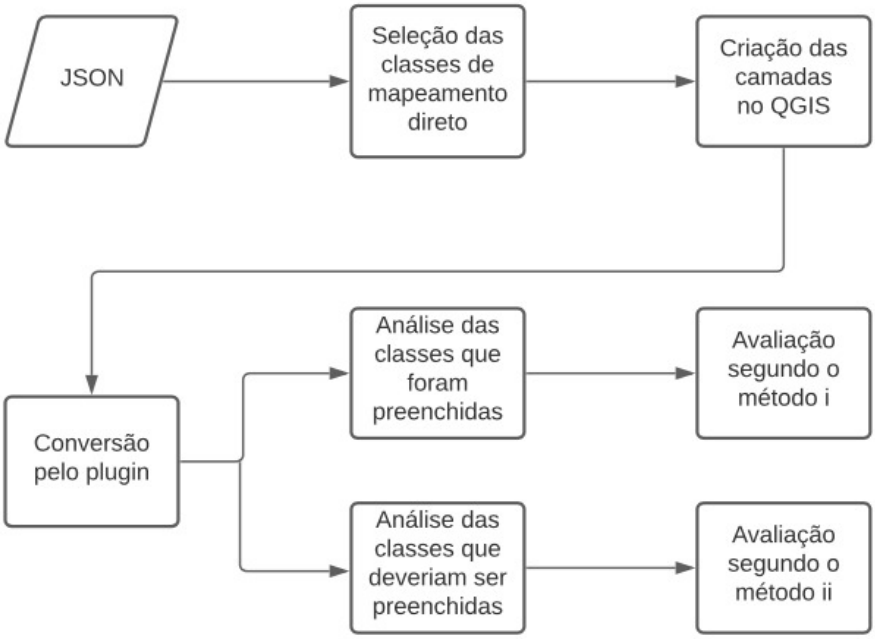


FIG. 3.14: Fluxograma Teste com Camadas Escolhidas

3.4.1.2 TESTE COM TODAS AS FEIÇÕES

Abrange a inserção de feições para todas as classes da EDGV, considerando tanto as classes e atributos contemplados com mapeamento direto quanto as não mapeadas. Adotando processo similar ao descrito no teste anterior, são carregadas as camadas do banco de dados de origem e criadas feições com seus atributos. Adota-se a mesma lógica e procedimento de métrica de qualidade descritas anteriormente.

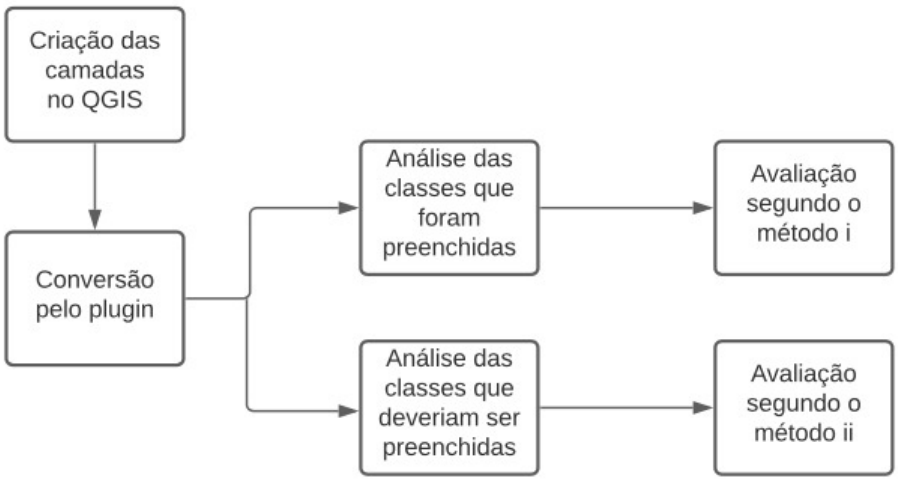


FIG. 3.15: Fluxograma Teste com Todas as Feições

3.4.1.3 TESTE COM CARTA

Para a realização deste teste, é utilizada uma carta existente, com as feições carregadas no banco de dados no formato da EDGV 3.0. Uma vez realizada a conversão, o banco de dados de destino é carregado no QGIS por procedimento similar ao descrito no teste anterior, sendo posteriormente efetuada a análise dos resultados.

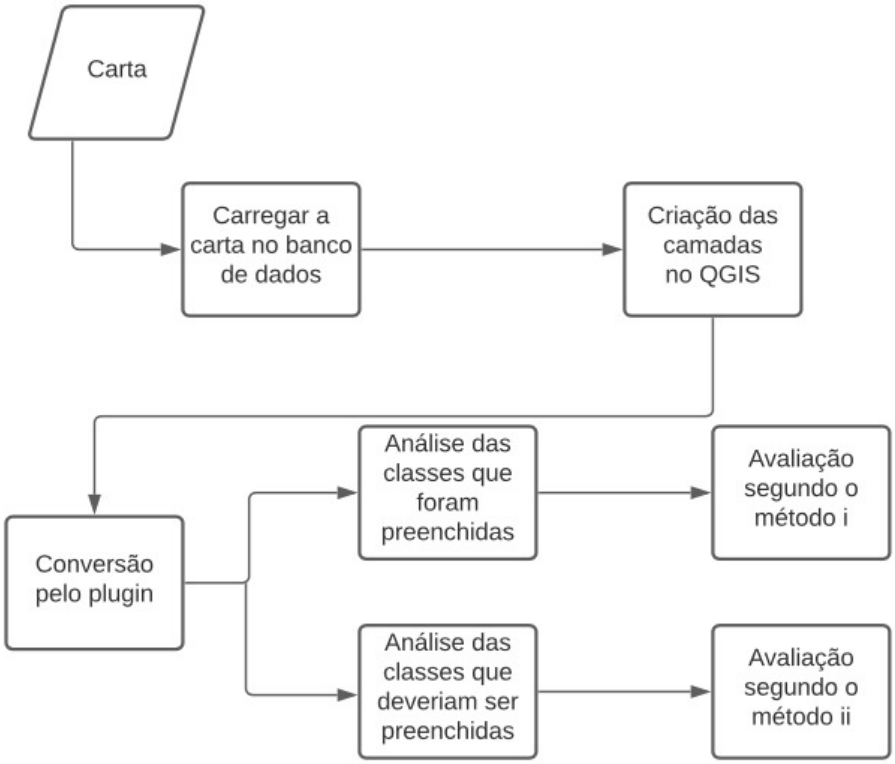


FIG. 3.16: Fluxograma Teste com Carta

3.4.2 MÉTRICAS DE AVALIAÇÃO

Para realizar a avaliação dos resultados e medir a qualidade de conversão, foram elaboradas 2 (duas) métricas:

- i. Quantidade de feições com atributos convertidas (FC) pela quantidade de feições com atributos inseridas para a conversão (FI)

$$\frac{FC}{FI}$$

- ii. Quantidade de feições com atributos convertidas corretamente (LC) por quantidade de feições com atributos inseridos para a conversão (FI)

$$\frac{LC}{FI}$$

Para definir indicadores numéricos que auxiliem na análise dos resultados, foi levado em consideração o impacto em se ter uma conversão com erro. Cartas a serem utilizadas no plugin, podem ter finalidades das mais diversas, como por exemplo missões de paz no exterior (onde seria necessário o uso de cartas estrangeiras), ações conjuntas internacionais, e estudos cartográficos estratégicos. Sob essa perspectiva, acredita-se que se ter 10% ou mais de erro na conversão de cartas para usos semelhantes aos listados acima, inviabiliza o uso destas para esses fins. Desta forma, definiu-se os valores de 95% de acerto para a razão "i" e de 90% para "ii"

3.5 CRONOGRAMA SEGUIDO NA PESQUISA

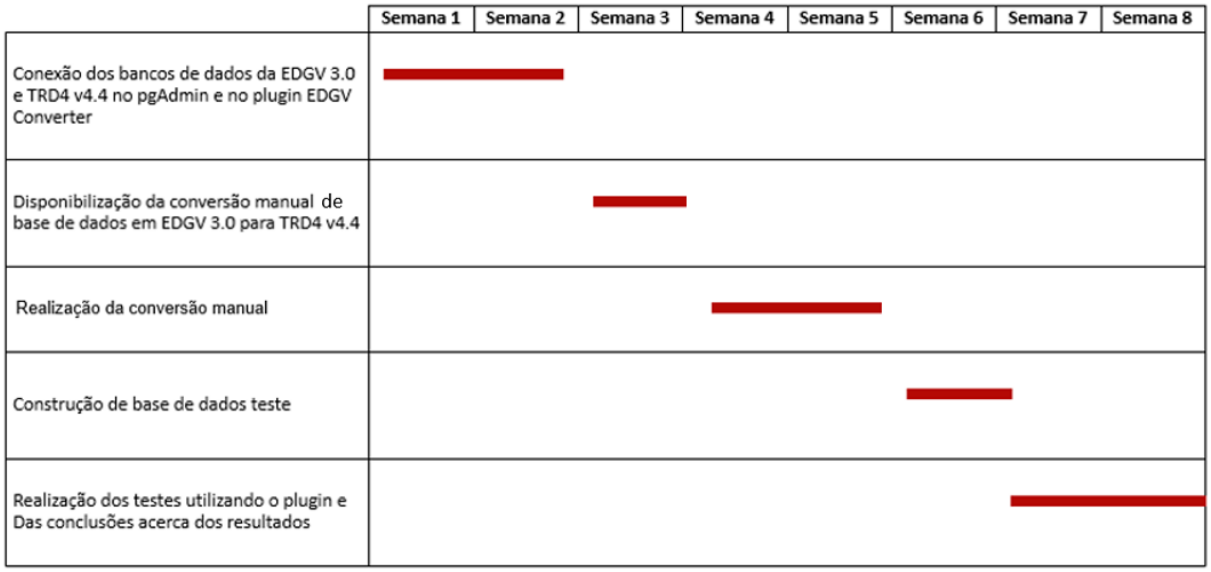


FIG. 3.17: Cronograma Seguido na Pesquisa

De acordo com o cronograma acima, foram utilizadas 08 (oito) semanas para a realização da pesquisa. Abaixo estão descritas as atividades contidas neste:

- Conexão com os bancos de dados da EDGV 3.0 e TRD4 v4.4 no pgAdmin e no plugin *EDGV Converter* (semanas 1 e 2): Atividades descritas no capítulo 3.2 deste texto
- Disponibilização de conversão manual para base de dados em EDGV 3.0 (semana 3): Atividade descrita no capítulo 2.6.1.2 deste texto
- Realização da conversão manual, usando como base o arquivo JSON com o mapeamento entre classes e atributos da EDGV 3.0 e TRD4 (semanas 4 e 5): Atividade descrita no capítulo 3.3 deste texto

- Construção da base de dados teste (semana 6): Atividade descrita no capítulo 3.4 deste texto
- Realização dos testes utilizando o plugin e das conclusões acerca dos resultados (semanas 7 e 8): Atividades realizadas e descritas nos capítulos 4 e 5 deste texto

4 RESULTADOS E DISCUSSÕES

4.1 RESULTADOS DOS TESTES REALIZADOS

O Teste com Camadas Escolhidas, foi realizado 2 (duas) vezes, sendo uma com feições criadas no banco de dados da DSG tools, e outra com as feições criadas no banco de dados recebido pela DSG.

Durante a criação de feições no primeiro caso, encontraram-se erros no código SQL do banco, de forma que certos atributos não estavam criados, e portanto não poderiam ser preenchidos. Mesmo assim, continuou-se a atividade, de forma que não foram criadas feições em atributos em que esse fenômeno acontecia.

Ao tentar realizar a execução do plugin no primeiro caso, ocorreu erro relacionado à conversão dos atributos que não puderam ser preenchidos.

Desta forma, foi carregado o banco de dados recebido pela DSG para a realização de uma segunda tentativa. As feições foram criadas de maneira similar ao primeiro caso, porém, sem a ocorrência dos erros destacados anteriormente.

A etapa de criação de camadas ocorreu, portanto, conforme o planejado, sendo geradas as feições. Dessa forma, foram realizados os passos de funcionamento do plugin, explicitados no capítulo 2.6.1.2.

Ao tentar executar a conversão através do plugin, surgiu a mensagem de erro conforme figura abaixo:

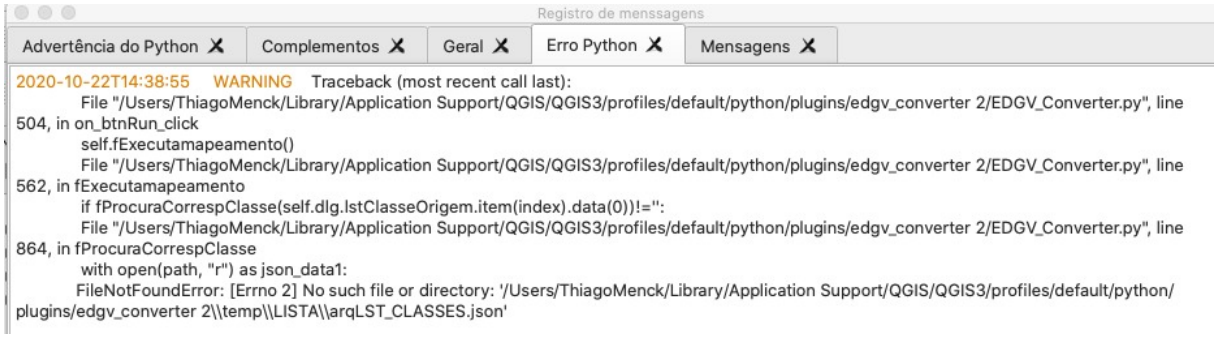


FIG. 4.1: Mensagem de Erro no Plugin *EDGV Converter*

De acordo com a mensagem, notou-se que o erro referia-se a inexistência do arquivo "arqST_CLASSES.json". Prontamente, verificou-se que mesmo quando presente na pasta indicada, o mesmo erro era acusado, e portanto, não sendo possível realizar a conversão.

Foram realizadas conferências no nome do arquivo, assim como a presença de espaços e caracteres especiais, para confirmar que o caminho chamado no plugin se referia ao arquivo explicitado.

4.2 DISCUSSÕES

Foram identificados limitadores quanto ao uso do plugin *EDGV Converter* para realizar conversões semelhantes à proposta. A ferramenta em questão, realiza conversões onde existem associações de atributos que acontecem de forma direta.

Para o mapeamento de atributos no padrão EDGV 3.0 para TRD4 v4.4 do MGCP, constam chaves no arquivo JSON de referência que não tratam de mapeamentos diretos, como as tratadas no capítulo 2.7.2.1.

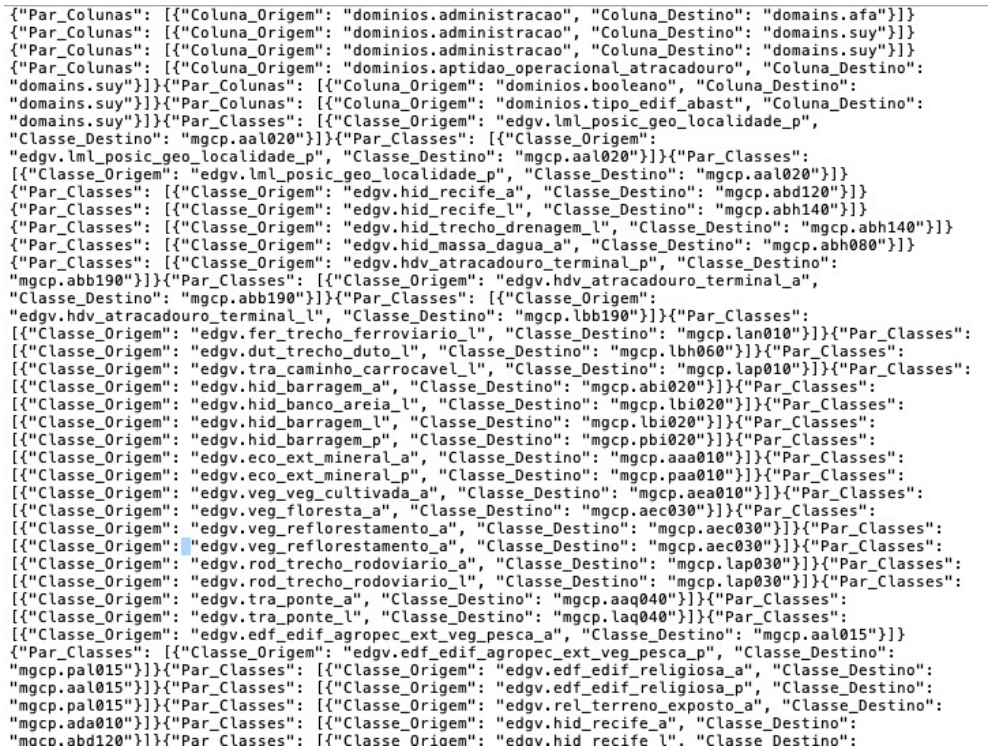
Com isso, neste arquivo, somente 13% dos mapeamentos contém o chaveamento "mapeamento_atributos", que é voltado para atributos com relacionamento direto entre os padrões considerados no trabalho.

4.3 PRODUTOS DA PESQUISA

Todos os produtos e arquivos citados abaixo, podem ser encontrados completos e disponíveis para download no repositório do GitHub criado para a presente pesquisa, no link: *https://github.com/thiago-mpm/pfc_2020_MB*.

4.3.1 ARQUIVO JSON - MAPEAMENTO EDGV 3.0 EM TRD4 V4.4

Conforme descrito no capítulo "Metodologia", um dos produtos gerados no projeto, foi um arquivo JSON com o mapeamento que contém a correpondência dos atributos do padrão EDGV 3.0, na TRD4 v4.4 da MGCP. Este arquivo está no formato necessário para que o plugin *EDGV Converter* possa utilizá-lo para realizar conversões de forma automática entre bancos de dados nos padrões considerados. O nome deste arquivo é "arq_selected".



```
{"Par_Colunas": [{"Coluna_Origem": "dominios.administracao", "Coluna_Destino": "domains.afa"}]}
{"Par_Colunas": [{"Coluna_Origem": "dominios.administracao", "Coluna_Destino": "domains.suy"}]}
{"Par_Colunas": [{"Coluna_Origem": "dominios.administracao", "Coluna_Destino": "domains.suy"}]}
{"Par_Colunas": [{"Coluna_Origem": "dominios.aptidao_operacional_atracadouro", "Coluna_Destino":
"domains.suy"}]}{"Par_Colunas": [{"Coluna_Origem": "dominios.booleano", "Coluna_Destino":
"domains.suy"}]}{"Par_Colunas": [{"Coluna_Origem": "dominios.tipo_edif_abast", "Coluna_Destino":
"domains.suy"}]}{"Par_Classes": [{"Classe_Origem": "edgv.lml_posic_geo_localidade_p",
"Classe_Destino": "mgcp.aal020"}]}{"Par_Classes": [{"Classe_Origem":
"edgv.lml_posic_geo_localidade_p", "Classe_Destino": "mgcp.aal020"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.lml_posic_geo_localidade_p", "Classe_Destino": "mgcp.aal020"}]}
{"Par_Classes": [{"Classe_Origem": "edgv.hid_recife_a", "Classe_Destino": "mgcp.abd120"}]}
{"Par_Classes": [{"Classe_Origem": "edgv.hid_recife_l", "Classe_Destino": "mgcp.abh140"}]}
{"Par_Classes": [{"Classe_Origem": "edgv.hid_trecho_drenagem_l", "Classe_Destino": "mgcp.abh140"}]}
{"Par_Classes": [{"Classe_Origem": "edgv.hid_massa_dagua_a", "Classe_Destino": "mgcp.abh080"}]}
{"Par_Classes": [{"Classe_Origem": "edgv.hdv_atracadouro_terminal_p", "Classe_Destino":
"mgcp.abb190"}]}{"Par_Classes": [{"Classe_Origem": "edgv.hdv_atracadouro_terminal_a",
"Classe_Destino": "mgcp.abb190"}]}{"Par_Classes": [{"Classe_Origem":
"edgv.hdv_atracadouro_terminal_l", "Classe_Destino": "mgcp.lbb190"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.fer_trecho_ferroviario_l", "Classe_Destino": "mgcp.lan010"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.dut_trecho_duto_l", "Classe_Destino": "mgcp.lbh060"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.tra_caminho_carrocavel_l", "Classe_Destino": "mgcp.lap010"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.hid_barragem_a", "Classe_Destino": "mgcp.abi020"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.hid_banco_areia_l", "Classe_Destino": "mgcp.lbi020"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.hid_barragem_l", "Classe_Destino": "mgcp.lbi020"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.hid_barragem_p", "Classe_Destino": "mgcp.pbi020"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.eco_ext_mineral_a", "Classe_Destino": "mgcp.aaa010"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.eco_ext_mineral_p", "Classe_Destino": "mgcp.paa010"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.veg_veg_cultivada_a", "Classe_Destino": "mgcp.aea010"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.veg_floresta_a", "Classe_Destino": "mgcp.aec030"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.veg_reflorestamento_a", "Classe_Destino": "mgcp.aec030"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.veg_reflorestamento_a", "Classe_Destino": "mgcp.aec030"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.rod_trecho_rodoviario_a", "Classe_Destino": "mgcp.lap030"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.rod_trecho_rodoviario_l", "Classe_Destino": "mgcp.lap030"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.tra_ponte_a", "Classe_Destino": "mgcp.aaq040"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.tra_ponte_l", "Classe_Destino": "mgcp.laq040"}]}{"Par_Classes":
[{"Classe_Origem": "edgv.edf_edif_agropec_ext_veg_pesca_a", "Classe_Destino": "mgcp.aal015"}]}
{"Par_Classes": [{"Classe_Origem": "edgv.edf_edif_agropec_ext_veg_pesca_p", "Classe_Destino":
"mgcp.pal015"}]}{"Par_Classes": [{"Classe_Origem": "edgv.edf_edif_religiosa_a", "Classe_Destino":
"mgcp.aal015"}]}{"Par_Classes": [{"Classe_Origem": "edgv.edf_edif_religiosa_p", "Classe_Destino":
"mgcp.pal015"}]}{"Par_Classes": [{"Classe_Origem": "edgv.rel_terreno_exposto_a", "Classe_Destino":
"mgcp.ada010"}]}{"Par_Classes": [{"Classe_Origem": "edgv.hid_recife_a", "Classe_Destino":
"mgcp.abd120"}]}{"Par_Classes": [{"Classe_Origem": "edgv.hid_recife_l", "Classe_Destino":
```


FIG. 4.2: Arquivo arq_selected.json

4.3.2 ATTGEN - ATRIBUTOS COM CONVERSÃO GENÉRICA

O AttGen é um recorte do JSON de referência, composto pela parte de "mepeamento_classes", onde constam os mapementos das classes, os filtros, e as chaves de default, e pela parte de "mapeamento_atributos", onde estão os atributos genéricos, ou seja, que pertencem a duas ou mais classes. O nome dado para o arquivo foi "AttGenericosconversao_pg-trd-44_pg-edgv-300-.json". São exemplos, "geometriaaproximada", que é convertido para "acc", e "nome" para "nam".

```
"mapeamento_atributos": [
  {
    "__comment": "ACC Horizontal Accuracy Category <-> geometriaaproximada",
    "attr_ida": "acc",
    "attr_volta": "geometriaaproximada",
    "traducao": [
      {
        "valor_ida": 1,
        "valor_volta": false
      },
      {
        "valor_ida": 2,
        "valor_volta": true
      }
    ]
  },
  {
    "__comment": "NAM Name <-> nome",
    "attr_ida": "nam",
    "attr_volta": "nome",
    "traducao":[
```


FIG. 4.3: Exemplos de Atributos com Conversão Genérica

4.3.3 MAPEAMENTOS ENCONTRADOS

O arquivo "Mapeamentos Encontrados" contém partes do JSON de referência. Foram selecionadas apenas os casos em que a chave "mapeamento_atributo" está presente, e acrescentados ao arquivo as informações do chaveamento das classes, os filtros e o mapeamento de atributos com o mapeamento de domínio descrito.


```
"__comment": "AL015 GENERAL BUILDING |  Place of Worship  & Religious Activities <-> edf_edif_religiosa",
  "classe_ida": "al015",
  "classe_volta": "edf_edif_religiosa",
  "filtro_ida": {
   "$or": [
    {
     "nome_atributo": "ffn",
     "valor": 931
    },
    {
     "nome_atributo": "ffn",
     "valor": 930
    }
   ]
  },
  ,
  "mapeamento_atributos": [
   {
    "attr_ida": "hwt",
    "attr_volta": "tipoedifrelig",
    "traducao": [
     {
      "valor_ida": 4,
      "valor_volta": 1
     },
     {
      "valor_ida": 16,
      "valor_volta": 2
```


FIG. 4.4: Arquivo Mapeamentos Encontrados - mapeamento_atributos

Logo após cada mapeamento de atributos, foi registrado com que classe da EDGV 3.0 o foi realizado, de acordo com a forma "RELACIONOU COM edgv.xxx_xxxxx". São escritos a seguir os atributos efetivamente relacionados. São eles aqueles constantes na chave "mapemento_atributo" e todos os atributos gerais constantes em AttGen e que estiverem presentes também na classe em questão.

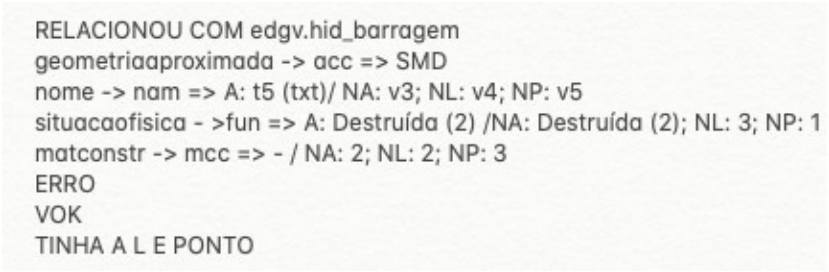


FIG. 4.5: Arquivo Mapeamentos Encontrados - Registro Classes

Após a seta com traço duplo, estão listados os valores dos atributos escolhidos no teste. Há classes que devem ser preenchidas com feições de mais de um tipo, não se restringindo a ponto, linha ou área somente (por exemplo "edgv.edif_edif_religiosa", que possui "edgv.edif_edif_religiosa_a" e "edgv.edif_edif_religiosa_p").

As feições criadas no banco de dados utilizado inicialmente, foram nomeadas por "t1", "t2", "t3", etc; enquanto que as criadas no banco de dados enviado pela DSG, por "v1", "v2", "v3", etc. São usadas as letras "A", "L" e "P" para identificar o valor do atributo preenchido na feiçao do tipo área, linha ou ponto, respectivamente.

Para feições criadas em ambos os bancos de dados, os valores usados no segundo teste são caracterizados na forma "NA", "NL" ou "NP", para nova área, nova linha e novo ponto respectivamente. Nesses casos, há uma barra "/" de separação entre os valores preenchidos nos bancos.

Dessa forma, as linhas logo abaixo do registro da classe relacionada, terão, de maneira geral, o formato "attedgv -> attmgcp => P: x; A: y; L: z/ NP: a; NA: b; NL: c", de acordo com a (FIG 4.4). Por vezes, há o número do atributo e o seu nome, "Destruída (2)", por exemplo.

A presença da palavra "ERRO", indica que não foi possível criar a feição no banco de dados considerado inicialmente, e a palavra "NOK", indica sucesso na criação da feição no recebido pela DSG.

4.3.4 DESCRITIVO DAS FEIÇÕES CRIADAS NO QGIS

Durante o povoamento dos bancos de dados para realizar o teste com camadas escolhidas, foram criados 2 (dois) arquivos para registrar os atributos preenchidos:

- Pontos Marcados para Realizar o Teste.txt - contém as feições criadas no banco de dados inicial
- Novos Pontos Marcados.txt - contém informações das feições criadas no banco de dados recebido pela DSG

5 CONCLUSÕES

No decorrer do presente trabalho, foram analisadas as normas quanto a estrutura, e foram percebidas similaridades e diferenças associadas. Procedeu-se o desenvolvimento de um relacionamento de atributos entre as duas normas em um arquivo *Excel*. Este se mostrou ser importante para levantar questionamentos acerca de atributos que não possuem conversão direta, como foram tratados no arquivo JSON de referência e qual o impacto destes nos resultados dos testes realizados.

Foi estudado o material da DSG relacionado ao assunto, que abrangia o arquivo JSON de referência, o caminho do repositório do GitHub da DSG, e contato com outros trabalhos e produtos relevantes. Com isso, foram elaboradas estratégias de trabalho e escolha dos caminhos ao longo da realização do trabalho.

Devido a presença de inúmeras chaves de mapeamentos não-diretos ("atributos_default", "mapeamento_multiplo", etc.), e ao baixo número de mapeamentos diretos, apenas uma parcela pequena das feições poderiam ser convertidas pelo plugin *EDGV Converter*.

Pensando em um universo de conversão que abranja grande parte das classes e atributos contidos na EDGV 3.0, ainda que o plugin apresentasse valor de "ii"de 100%, mesmo assim todas as feições convertidas formaria um espaço amostral extremamente pequeno em relação ao que deveria ser convertido.

Com isso, para viabilizar a conversão entre um banco de dados preenchido no padrão EDGV 3.0 para um em TRD v4.4 do MGCP através do plugin *EDGV Converter*, é necessário realizar adaptações no mesmo, ou o desenvolvimento de um plugin específico para atividade.

O trabalho desprendido para realizar as adaptações seria equivalente, ou maior, ao de desenvolver um novo, sendo este, o caminho mais adequado.

6 REFERÊNCIAS BIBLIOGRÁFICAS

GALVÃO, W.P; LIMA, N.C.C (2015). **Avaliação do Impacto da Utilização da ET-EDGV Defesa Fter Na Linha de Produção Cartográfica.** Disponível em: https://bdex.eb.mil.br/jspui/handle/1/847 [capturado em 24 ago 2020]

COMISSÃO NACIONAL DE CARTOGRAFIA - CONCAR (2010). **Especificação Técnica para a Estruturação de Dados Geoespaciais Vetoriais**. Disponível em: http://www.geoportal.eb.mil.br/index.php/inde2?id=139 [capturado em 13 jul 2020]

JÓZSEF, C.; OLÍVIA, M. (2009). **Multinational geospatial co-production program**. Disponível em: https://www.researchgate.net/publication/321700982 [capturado em 13 jul 2020]

DIRETORIA DE SERVIÇOS GEOGRÁFICOS - DSG. **Participação do 1 CGEO em evento Internacional de Defesa.** Disponível em: http://www.dsg.eb.mil.br/index.php/ultimas-noticias/55-ultimas-noticias-da-dsg/124-participacao-do-1-cgeo-em-evento-internacional-de-defesa [capturado em 25 ago 2020]

DATE, C.J.; **Introdução a Sistemas de Bancos de Dados.** 8 ed. São Paulo: GEN LTC, 2004.

OLIVEIRA, S.S. (2014). **Bancos de Dados Não-Relacionais: Um Novo Paradigma para Armazenamento de Dados em Sistemas de Ensino Colaborativo**. Disponível em: https://www2.unifap.br/oliveira/files/2016/02/35-124-1-PB.pdf [capturado em 25 ago 2020].

QGIS Project, c2020. Sobre o QGIS. Disponível em: https://www.qgis.org/ptBR/site/about/index.html. [Capturado em 25 ago 2020].

VITAL, Marcos. SIG em Código Aberto: A História do QGIS. [Entrevista Concedida a] QGIS Project. **QGIS Project**, 2019. Disponivel em: https://www.youtube.com/watch?v=fbIea0o3RWU [capturado em 25 ago 2020].

CASTRO, B.P.S. (2019). **Mapeamento Entre as Especificações MGCP TRD4 v4.4 e ET-EDGV 3.0** [capturado em 26 ago 2020].

SILVA, D.M.. Python: História e Ascendência. **Revista Programar**, ed. 59, 2018, página 96. Disponível em: https://www.revista-programar.info/edicoes/edicao-59/ [capturado em 26 ago 2020].

BORGES, L.E.. **Python para desenvolvedores**. 2ª Edição, Rio de Janeiro. 2010. Disponível em: https://ark4n.files.wordpress.com/2010/01/ python_para_desenvolvedores $_2$*ed.pdf* [*capturadoem*26*ago*2020].

COMISSÃO NACIONAL DE CARTOGRAFIA - CONCAR (2017). **Especificação Técnica para a Estruturação de Dados Geoespaciais Vetoriais (ET-EDGV 3.0).** Disponível em: https://www.concar.gov.br/temp/365@ET-EDGV_versao_3.0_2018_05_20.pdf [capturado em 26 ago 2020].

Multinational Geospatial Co-production Program (2016). **TRD4 v4.4 MGCP Technical Reference Documentation** [capturado em 26 ago 2020].

DIRETORIA DE SERVIÇOS GEOGRÁFICOS - DSG (2014). **Certificado de Registro de Programa de Computador.** Disponível em: http://www.dsg.eb.mil.br/images/certificado_dsg_tools-tarjapreta.pdf [capturado em 27 ago 2020].

PRATES, I. (2015). **DSG Tools**. Disponível em: https://mundogeo.com/2015/04/15/dsg-tools/ [capturado em 27 ago 2020].

GALVÃO, W.P.; FREITAS, F. L.; LIMA, N. C. C. (2017). **O Impacto do Versionamento da Estrutura de Dados Espaciais Vetoriais na Etapa de Reambulação, no Âmbito da Diretoria de Serviço Geográfico do Exército**. Disponível em: http://www.cartografia.org.br/cbc/2017/trabalhos/3/fullpaper/CT03-22_1506790445.pdf [capturado em 31 ago 2020].

COSTA, E.R. (2011). **Bancos de Dados Relacionais.** Disponível em: http://www.fatecsp.br/dti/tcc/tcc0025.pdf [capturado em 03 set 2020].

SIQUEIRA, F. (2016); **Modelo Entidade e Relacionamentos.** Disponível em: https://sites.google.com/site/uniplibancodedados1/aulas/aula-4—modelo-entidade-e-relacionamentos [capturado em 03 set 2020].

BALAGUER, A. (2016); **Modelo Entidade Relacionamento (MER) e Diagrama Entidade-Relacionamento (DER).** Disponível em: https://professor.adrianobalaguer.com/2016/10/modelo-entidade-relacionamento-mer-e.html [capturado em 03 set 2020].

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. (2006). **Sistema de banco de dados.** 5. ed. Rio de Janeiro: Elsevier, Campus, 2006. 781 p. [capturado em 03 set 2020].

FIDELIX, M.C. (2019); **Modelo Entidade e Relacionamento (MER).** Disponível em: https://www.slideshare.net/CrisFidelix/3-modelo-entidade-relacionamento [capturado em 03 set 2020].

MELO, A.C. (2004); **Desenvolvimento aplicações com UML 2.0: do conceitual à implementação.** 2. Ed. Rio de Janeiro: Brasport, 2004 [capturado em 03 set 2020].

FONSECA, G. (2011); **Os principais diagramas da UML.** Disponível em: https://www.profissionaisti.com.br/os-principais-diagramas-da-uml-resumo-rapido/ [capturado em 03 set 2020].

MACHADO, F.T.S. (2017); **Um Processo para Extração de Esquemas Conceituais em Fontes de Dados JSON Baseado em Técnicas de Similaridade de Texto** [capturado em 03 set 2020].

INTERNATIONAL, E. (2013); **The JSON Data Interchange Format.** Disponível em: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf [capturado em 03 set 2020].

SANTOS, B.A.; ALMEIDA, F.C.; WOLFGRAM, V.A. (2019); **Ferramenta de**

Conversão de Dados Geoespaciais Vetoriais [capturado em 10 set 2020].

DINIZ, F.C. (2019); **Relatório Técnico Nº 04/2019 – DGEO/1ºCGEO, Padrão de Mapeamento Entre Modelagens.**

SEVERANCE, C. (2015); **Guido van Rossum: The Early Years of Python.** Disponível em: https://www.computer.org/csdl/magazine/co/2015/02/mco2015020007/13rRUy3gmYB [capturado em 15 out 2020].