UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

INSTITUTO DE MATEMÁTICA

CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JOÃO FELIPE CURCIO DA SILVA

ETL4LOD+: EVOLUÇÃO DO SUPORTE AO CICLO DE PUBLICAÇÃO DE DADOS CONECTADOS

RIO DE JANEIRO

2018

JOÃO FELIPE CURCIO DA SILVA

ETL4LOD+: EVOLUÇÃO DO SUPORTE AO CICLO DE PUBLICAÇÃO DE DADOS CONECTADOS

Trabalho de Conclusão de Curso de graduação apresentado ao Departamento de Ciência da Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Orientadora: Maria Luiza Machado Campos

RIO DE JANEIRO

2018

CIP - Catalogação na Publicação

S586e

Silva, João Felipe Curcio da
ETL4LOD+: evolução do suporte ao ciclo de publicação de dados conectados / João Felipe Curcio da Silva. -- Rio de Janeiro, 2018.
75 f.

Orientadora: Maria Luiza Machado Campos.
Trabalho de conclusão de curso (graduação) - Universidade Federal do Rio de Janeiro, Instituto de Matemática, Bacharel em Ciência da Computação, 2018.

1. Web semântica. 2. Triplificação. 3. ETL. 4. Linked Open Data. I. Campos, Maria Luiza Machado, orient. II. Titulo.

Elaborado pelo Sistema de Geração Automática da UFRJ com os dados fornecidos pelo(a) autor(a), sob a responsabilidade de Miguel Romeu Amorim Neto - CRB-7/6283.

JOAO FELIPE CURCIO DA SILVA

ETL4LOD+: EVOLUÇÃO DO SUPORTE AO CICLO DE PUBLICAÇÃO DE DADOS CONECTADOS

Trabalho de conclusão de curso de graduação apresentado ao Departamento de Ciência da Computação da Universidade Federal do Rio de Janeiro como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 14 de dezembro de 2018.

BANCA EXAMINADORA:

Maria Luiza Machado Campos, Ph.D. (UFRJ)

Giseli Rabello Lopes, D.Sc (UFRJ)

Fernanda Araujo Baião, D.Sc (PUC-RIO)

Letícia Dias Verona, M.Sc (UFRJ)

À minha mãe que me serviu de alicerce
em todos os momentos da minha educação
profissional.

AGRADECIMENTOS

À Maria Luiza pela orientação durante todo o processo de desenvolvimento deste trabalho e pela animação com os sucessos do projeto, que foi um grande motivador. À Kelli Cordeiro e ao Rogers Reiche que me ajudaram na compreensão de partes do projeto predecessor a este. E à Letícia Verona por me fornecer os dados usados na experimentação desde trabalho.

‘I have a dream for the Web [in which computers]
become capable of analyzing all the data on the Web
– the content, links, and transactions between
people and computers. A "Semantic Web",
which makes this possible, has yet to emerge,
but when it does, the day-to-day mechanisms of trade,
bureaucracy and our daily lives will be handled by machines
talking to machines. The "intelligent agents"
people have touted for ages will finally materialize.’

Tim Berners-Lee

RESUMO

O desenvolvimento do ETL4LOD, resultado de projeto do grupo GRECO junto à RNP (Rede Nacional de Pesquisa), foi um passo importante na criação de novas soluções para limpeza e triplificação de dados no contexto de dados abertos conectados (Linked Open Data) devido à sua facilidade de uso e de extensão para novas funcionalidades. Por diversas razões, como a falta de atualizações necessárias, o interesse no ETL4LOD foi diminuindo ao decorrer dos anos. Algumas ferramentas, como Karma, Open Refine e Any23, foram testadas como possíveis alternativas porém sem sucesso. Para preencher essa lacuna que o ETL4LOD deixou nas soluções de limpeza e triplificação, este trabalho desenvolveu o ETL4LOD+. O ETL4LOD+ é uma *framework* filha do ETL4LOD que estende as suas funcionalidades – permitindo buscar ontologias online e o trabalho com ontologias em arquivos, e adicionando a etapa de ligação automática – além de modernizar a sua estrutura base, que seria o seu conjunto de *plug-ins*, código e documentação.

Palavras-chave: Web Semântica. Triplificação. ETL. Linked Open Data.

ABSTRACT

The development of ETL4LOD, biproduct of a project from the group GRECO alongside RNP (Rede Nacional de Pesquisa), was an important step in terms of solutions for data cleaning and triplification of data in the context of Linked Open Data due to being user friendly and easy to extend its functionalities. Due to several reasons, including the lack of updates, the interest in ETL4LOD is dwindling over time. A few other tools, such as Karma, Open Refine and Any23, were tested as possible alternatives but to no avail. To fill this gap left by ETL4LOD, this paper developed ETL4LOD+, a child framework of ETL4LOD which expands its functionalities – adding the capability to search ontologies online and work with ontology dumps as well as adding an automated link discovery tool to its plugins – and modernizes its base framework, meaning its plugin ecosystem, code and documentation.

Keywords: Semantic Web. Triplification. ETL. Linked Open Data.

LISTA DE ILUSTRAÇÕES

Figura 1 - Grafo RDF com tripla sujeito, predicado e objeto 18
Figura 2 - Processo de publicação de dados em RDF 20
Figura 3 - Interface do ETL4LOD 22
Figura 4 - Fluxo de ETL no Kettle 23
Figura 5 - Classes que compõem um plug-in Kettle 23
Figura 6 - Página inicial do Karma. 25
Figura 7 - Funções de limpeza de dados do Open Refine 26
Figura 8 - Interface gráfica do Any23 27
Figura 9 - Ciclo de vida dos dados LOD 28
Figura 10 - Changelog das primeiras mudanças feitas nas dependências dos plug-ins 29
Figura 11 - Fluxo mais usado na triplificação de dados no ETL4LOD+ 31
Figura 12 - Fluxos de triplificação suportados pelo ETL4LOD+ 31
Figura 13 - Interface gráfica do Data Property Mapping 32
Figura 14 - Interface gráfica do Object Property Mapping 33
Figura 15 - Interface gráfica do NTriple Generator 34
Figura 16 - Exemplo de consulta aceita pelo SPARQL Endpoint 34
Figura 17 - Exemplo de consulta que pode ser feita no SPARQL Run Query 35
Figura 18 - Exemplo de configuração do SPARQL Run Query 35
Figura 19 - Exemplo de configuração do SPARQL Update Output 36
Figura 20 - Interface gráfica do Annotator 37
Figura 21 - Arquivo de mapeamento do Annotator 37
Figura 22 - Exemplo de configuração Graph Semantic Level Maker 38
Figura 23 - Configuração do arquivo XML de nível semântico 38
Figura 24 - Exemplo de consulta aceita pelo Graph SPARQL Endpoint 39
Figura 25 - Exemplo de configuração do Graph Triplify 40
Figura 26 - Exemplo de configuração Turtle Generator 40
Figura 27 - Exemplo de configuração Data Cube 41
Figura 28 - Busca de ontologias no Owl Input 42
Figura 29 - Seleção de termos Owl Input 43
Figura 30 - Exemplo de configuração Link Discovery Tool 44
Figura 31 - Excerto da lista suja 46

Figura 32 - Excerto das operações contidas na lista suja 47
Figura 33 - Operação 1 gerada manualmente em RDF/XML 48
Figura 34 - Operação 1 em formato N-Triples gerado pelo Any23 49
Figura 35 - Fluxo para triplificar as operações da lista suja 49
Figura 36 - Configuração do Owl Input 50
Figura 37 - Configuração do Object Property Mapping 50
Figura 38 - Configuração do Data Property Mapping 51
Figura 39 - Configuração do N-Triple Generator 51
Figura 40 - Triplas geradas pelo ETL4LOD+ 52
Figura 41 - Configuração do Sparql Update Output 52
Figura 42 - Excerto da primeira parte do dossiê da ABRASCO 53
Figura 43 - Fluxo de limpeza dos dados da ABRASCO 54
Figura 44 - Problema 1: rodapé com fonte dos dados 54
Figura 45 - Problema 2: múltiplos valores numa célula só 54
Figura 46 - Problema 2 após a limpeza 55
Figura 47 - Problema 3: dados fora do padrão 55
Figura 48 - Anotação dos sintomas agudos com a ontologia MEDDRA 56
Figura 49 - Mapeamento de triplas dos dados da ABRASCO 57
Figura 50 - Descrição de URIs usando labels 58
Figura 51 - Fluxo para triplificação dos dados da ABRASCO 58
Figura 52 - Triplas do agrotóxico Pentaciclorofenol 59
Figura 53 - Triplas do agrotóxico Pentaciclorofenol geradas pelo ETL4LOD+ 59
Figura 54 - Resumo das modificações feitas nos plug-ins 61

LISTA DE TABELAS

Tabela 1 - Lista de plug-ins sem retrocompatibilidade com o ETL4LOD+ 30

LISTA DE SIGLAS

API – Application Program Interface

CSV – Comma Separated Values

ETL – Extraction Transformation Load

HTTP – Hypertext Transfer Protocol

IRI – Internationalized Resource Identifier

LOD – Linked Open Data

LOV – Linked Open Vocabulary

OWL – Ontology Web Language

PDF – Portable Document Format

RDF – Resource Description Framework

REST – Representational State Transfer

SPARQL – SPARQL Protocol and RDF Query Language

TIFF – Tagged Image File Format

URI – Unique Resource Locators

WWW – World Wide Web

W3C – World Wide Web Consortium

XML – eXtensible Markup Language

SUMÁRIO

1 INTRODUÇÃO 14
1.1 MOTIVAÇÃO 14
1.2 OBJETIVO 15
1.3 ESTRUTURA 15
2 REVISÃO DA LITERATURA 17
2.1 WEB SEMÂNTICA E SUAS TECNOLOGIAS 17
2.2 PROCESSO DE PUBLICAÇÃO DE DADOS CONECTADOS 19
2.2.1 Serializações RDF 21
2.2.2 Ligação 21
2.3 TRABALHOS RELACIONADOS 22
2.3.1 ETL4LOD 22
2.3.2 Karma 25
2.3.3 Open Refine 26
2.3.4 Any23 27
3 ETL4LOD+ 28
3.1 PASSOS PARA CRIAÇÃO DO ETL4LOD+ 29
3.2 PLUG-INS 30
3.2.1 Data Property Mapping 32
3.2.2 Object Property Mapping 33
3.2.3 N-Triple Generator 33
3.2.4 Sparql Endpoint 34
3.2.5 Sparql Run Query 35
3.2.6 Sparql Update Output 36
3.2.7 ETL4LOD-Graph 36
3.2.8 Turtle Generator 40
3.2.9 Data Cube 41
3.2.10 Owl Input 42
3.2.11 Link Discovery Tool 43
3.2.12 Any23 45
4 EXEMPLO DE APLICAÇÃO 46
4.1 APLICAÇÃO DO ETL4LOD+ NA LISTA SUJA 46

4.2 APLICAÇÃO DO ETL4LOD+ NO DOSSIÊ DA ABRASCO 53
4.2.1 Limpeza dos dados 53
4.2.2 Triplificação dos dados limpos 55
4.3 CONSIDERAÇÕES FINAIS 59
5 CONCLUSÃO 61
5.1 DIFICULDADES ENCONTRADAS 62
5.1.1 Documentação 62
5.1.2 Testes 62
5.2 TRABALHOS FUTUROS 62
5.2.1 Link Discovery Tool 62
5.2.2 Owl Input 63
5.2.3 Data Cube 63
5.2.4 Plug-ins de grafo 63
5.2.5 Testes do ETL4LOD+ 63
5.2.6 Manutenção 64
REFERÊNCIAS 65
APÊNDICE A – Arquivo CSV de entrada para o *plug-in* Turtle Generator 68
APÊNDICE B – Arquivo RDF/Turtle gerado pelo *plug-in* Turtle Generator 69
APÊNDICE C – Exemplo de arquivo CSV usado como entrada para o *plug-in* Data Cube 70
APÊNDICE D – Arquivo RDF/Turtle gerado pelo *plug-in* Data Cube 71
APÊNDICE E – Esqueleto da criação do *plug-in* NTriple Generator 73

1 INTRODUÇÃO

Existe uma grande quantidade de dados governamentais disponíveis na Web, gerados a partir das muitas iniciativas de *e-gov* e *open-government*, que defendem a ampla divulgação de dados aos cidadãos e organizações. No entanto, o consumo conjunto e reutilização desses dados ainda é difícil, dada suas interfaces voltadas apenas para consulta ou extração ad-hoc, além dos altos custos e problemas envolvidos na sua análise (CORDEIRO et al., 2011).

Linked Data, ou dados conectados, pode amenizar esses problemas fornecendo uma série de padrões para representar e interligar dados, desta forma permitindo que dados sejam descobertos e usados por várias aplicações. Para gerar dados conectados precisamos que os dados estejam num formato que seja fácil de manipular e converter para RDF (processo conhecido como triplificação) (SUKHOBOK et al., 2016). Para facilitar este processo, foram desenvolvidas ferramentas que apoiem cada etapa de transformação dos dados fonte para dados conectados.

1.1 MOTIVAÇÃO

Em 2011, com o financiamento da RNP, a equipe do grupo GRECO (Grupo de Engenharia do Conhecimento) da UFRJ desenvolveu uma ferramenta para apoiar a limpeza, transformação e publicação de dados conectados, conhecida como ETL4LOD (EQUIPE GT, 2011), baseada em uma *framework* de ETL (Extraction, Transformation and Loading), o Kettle/PENTAHO[1]. A ferramenta fazia parte de uma plataforma maior, denominada LinkedDataBR, que foi pouco tempo depois descontinuada. Com o passar dos anos, após outras versões do Kettle serem lançadas, o ETL4LOD ficou defasado e o interesse em seu uso diminuiu.

Algumas funcionalidades importantes a triplificação de dados como a busca de ontologias e vocabulários, adição de arquivos de ontologias e automatização do processo de ligação estavam acopladas ao LinkedDataBR que, após ter sido descontinuado, deixou o ETL4LOD sem essas funcionalidades.

Com o objetivo de encontrar uma ferramenta atual mais adequada ao processo de limpeza e publicação de dados conectados, Dantas (2018) realizou um estudo comparativo das

[1] https://community.hitachivantara.com/docs/DOC-1009855-data-integration-kettle

ferramentas Karma, Any23 e Open Refine. Neste estudo, a ferramenta Karma se destacou positivamente por ser a mais robusta e apropriada para processos de triplificação.

Contudo, em estudo complementar realizado por Medeiros e Alves (2018), as ferramentas Open Refine e Karma se mostraram ineficazes para resolver alguns problemas de triplificação de dados, a exemplo da importação de algumas ontologias, e acabaram por ser pouco úteis em diversas situações.

Este trabalho apresenta o ETL4LOD+, uma nova *framework* criada com o ETL4LOD como base, que adiciona as funcionalidades antes apenas presentes no LinkedDataBR e que são fundamentais ao processo de triplificação.

1.2 OBJETIVO

O objetivo deste trabalho é criar uma *framework* de limpeza, triplificação e ligação de dados conectados – apelidada de **ETL4LOD+** –, usando o ETL4LOD como base. Neste trabalho, será mostrado como o ecossistema do ETL4LOD foi atualizado e expandido para gerar o ETL4LOD+. Além disso, serão apresentados exemplos de aplicação para evidenciar algumas funcionalidades disponibilizadas.

Em resumo, as modificações feitas ao projeto inicial foram:

1. Adaptação da ferramenta original para funcionar na versão 8.1 do Kettle;
2. Atualização das dependências de código para versões mais recentes;
3. Criação de uma documentação online de uso da ferramenta;
4. Adição de duas novas funcionalidades ao ETL4LOD:
    a. Busca e seleção de termos em ontologias e vocabulários;
    b. Ligação automática entre duas fontes de dados triplificadas.

1.3 ESTRUTURA

O Capítulo 1 contém uma introdução ao ETL4LOD e a motivação para a extensão dessa plataforma.

No Capítulo 2 são abordados os conceitos mais importantes de Web Semântica para embasar o trabalho e são apresentados trabalhos relacionados ao ETL4LOD+.

O Capítulo 3 contém todos os pormenores da criação do ETL4LOD+ e o que muda em relação a sua plataforma-pai: o ETL4LOD.

No Capítulo 4 são abordados os testes criados para o ETL4LOD+ e como a ferramenta se comportou durante os testes.

O Capítulo 5 contém as conclusões, dificuldades encontradas e trabalhos futuros relacionados ao projeto.

2 REVISÃO DA LITERATURA

2.1 WEB SEMÂNTICA E SUAS TECNOLOGIAS

Originalmente, a maioria dos dados e informações na Web foram desenhados para consumo humano, não para programas de computador manipularem de forma significativa. Computadores podem analisar páginas da Web e identificar seu layout – onde está o cabeçalho, onde está um link para outra página – mas em geral, computadores não possuem uma forma confiável de atribuir significado aos termos ou dados publicados (BERNERS-LEE; HENDLER; LASSILA, 2001), com a consequente dificuldade de resolver problemas de ambiguidade.

A Web Semântica, proposta por Tim Berners-Lee, James Hendler e Ora Lassila em artigo clássico “The Semantic Web” (BERNERS-LEE; HENDLER; LASSILA, 2001), vem para endereçar esse problema. Segundo Ismail e Shaikh (2016), a Web Semântica tenta classificar os dados na Web com base em diferentes tópicos e atribuir um significado a eles com o propósito de ajudar não só humanos a entender os dados, mas também máquinas e programas de computador a processá-los de forma menos ambígua.

É importante ressaltar que a Web Semântica não é uma Web separada mas uma extensão da World Wide Web, na qual à informação é atribuído algum significado, na forma de anotações em vocabulários ou ontologias, ou mesmo na forma de interligações com outros dados associados (BERNERS-LEE; HENDLER; LASSILA, 2001).

Para construir a Web Semântica alguns componentes básicos são necessários. Os dois primeiros componentes são XML e RDF. XML permite que usuários adicionem estrutura a documentos na Web, porém não diz nada sobre o que essa estrutura significa.

O significado vem do RDF, que é codificado em um conjunto de triplas parecidas com a estrutura de uma frase elementar com sujeito, objeto e predicado (BERNERS-LEE; HENDLER; LASSILA, 2001), ilustrado na Figura 1. Em RDF, um documento faz afirmações de que uma coisa (sujeito, e.g., uma pessoa) tem uma propriedade (predicado, e.g., é parente de) com certos valores (objeto, e.g., outra pessoa). E cada parte dessas afirmações podem ser identificadas por URIs assim como os *links* usados em uma página da Web.

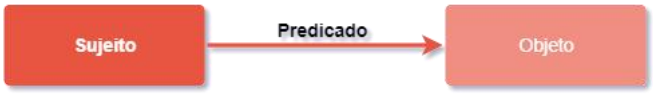


Figura 1 - Grafo RDF com tripla sujeito, predicado e objeto

Como RDF usa URIs para identificar as informações em documentos, isso garante que os conceitos expressos em RDF não sejam meras palavras em um documento mas estejam interligados a uma única definição que qualquer um pode encontrar na Web.

É claro que isto não é suficiente, porque duas fontes de dados podem possuir diferentes identificadores para o mesmo conceito. Um programa que quer comparar e combinar informações entre duas fontes de dados precisa saber que esses dois termos estão sendo usados para referenciar uma mesma coisa. Para solucionar esse problema, a Web Semântica possui um terceiro componente básico, uma coleção de termos e conceitos chamada ontologias (BERNERS-LEE; HENDLER; LASSILA, 2001), ou vocabulários.

Ontologias são usadas para modelar a semântica de conceitos dentro de um domínio particular e as relações entre esses conceitos (ALIYU; JUNAIDU; KANA, 2015). Isto é, uma ontologia é a especificação de conceitos usados com frequência e suas relações em formato que a Web Semântica consiga usar. Vocabulários são estruturas mais simples, mas que também servem de referência para resolver ambiguidades.

Por fim, precisamos de um último componente – uma linguagem padrão de consulta que permita que os dados dispostos na Web Semântica sejam acessados. Essa linguagem é o SPARQL. Com SPARQL é possível localizar informações específicas na Web Semântica, e a Web pode dessa forma ser vista como um grande banco de dados (WOOD et al., p. 241, 2014).

Uma vez que tenhamos esses componentes, estamos preparados para implementar e usar a Web Semântica idealizada por Berners-Lee, Hendler e Lassila.

Em resumo, a Web Semântica trata de adicionar metadados a dados na Web de forma a incluir algum significado a esses dados. Além disso, proporciona interligar esses dados de tal forma que uma pessoa ou máquina consiga explorar toda a Web de dados (BERNERS-LEE, 2006). Uma vez que esses dados estejam interligados, podemos entrar no conceito de Linked Open Data.

O termo dados conectados (Linked Data) se refere ao conjunto de melhores práticas para publicar e interligar dados estruturados na Web usando os padrões internacionais da W3C. Dados conectados usam RDF como modelo de dados e seus padrões, porém dados

conectados não são sinônimos de RDF. O que separa o RDF de dados conectados são quatro princípios (WOOD et al., 2014, p. 11) descritos por Berners-Lee (2006):

1. **Use URIs para nomear coisas**. Segundo a sintaxe do RDF 1.1 pela W3C, cada elemento de uma tripla RDF pode ser identificado por uma URI, um literal – um valor constante –, ou um nó em branco. Este princípio trata exatamente de como cada elemento dos dados conectados devem ser unicamente identificados – usando URIs.
2. **Use URIs HTTP para que pessoas possam procurar o significado dessas coisas.** Se suas URIs estiverem no formato HTTP então os navegadores a entenderão e será possível navegar nelas na Web.
3. **Quando alguém procurar uma URI, retorne informação útil, usando os padrões (RDF, SPARQL).** Quando um dado interligado é acessado, então uma informação útil, legível e bem estruturada deve ser retornada, a exemplo do que acontece no DBpedia[2].
4. **Inclua links para outras URIs para que novas coisas possam ser descobertas**. Os dados são conectados unicamente quando apontam para informações relacionadas.

Se os seus dados conectados estiverem disponíveis sobre uma licença aberta, que não impeça o reuso desses dados gratuitamente, então dizemos que esses são dados conectados abertos ou *Linked Open Data (LOD)* (BERNERS-LEE, 2010).

2.2 PROCESSO DE PUBLICAÇÃO DE DADOS CONECTADOS

A *World Wide Web* é plena de dados. Dados são publicados em formatos como PDF, TIFF, CSV, Excel, em tabelas em *Word*, e várias outras formas de texto. Porém esse tipo de dado tem uma limitação: foram projetados para humanos consumirem. Não é fácil para ferramentas automáticas processarem, procurarem e/ou reusarem esses dados (WOOD et al., 2014, p. 4). A Web Semântica propõe uma solução para esse problema, fazendo com que os dados sejam disponibilizados no formato de dados conectados na Web. Isso reduz nosso problema a transformar dados não tratados em dados conectados em formato RDF na Web.

Contudo, a conversão de dados não tratados em dados conectados envolve mais do que simplesmente converter esses dados para RDF. Uma parte fundamental no processo de

[2] http://dbpedia.org/

publicação inclui a identificação ou criação de vocabulários e ontologias que provêm o modelo de dados por trás dos dados conectados (CORDEIRO et al., 2011).

Além disso, dados não tratados possuem vários problemas comuns de qualidade de dados, como valores faltantes, valores inválidos, linhas duplicadas, etc. Resolver problemas na qualidade de dados é especialmente importante para integrar fontes heterogêneas de dados (SUKHOBOK et al., 2016). Por isso, a primeira etapa na publicação de dados conectados, ilustrado na Figura 2, é submeter os dados não tratados a um pré-processamento conhecido como ETL. O processo de ETL é uma série de operações que permite dados não tratados serem sintaticamente e semanticamente harmonizados segundo a estrutura e terminologia de um modelo de dados alvo (ONG et al., 2017), resolvendo os problemas de qualidade de dados.

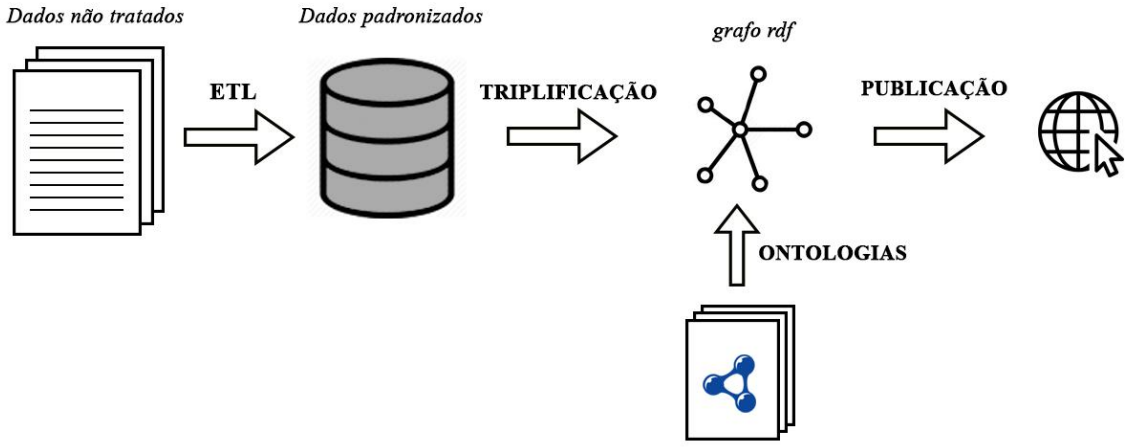


Figura 2 - Processo de publicação de dados em RDF

Uma vez que os problemas na qualidade dos dados sejam resolvidos, eles podem ser transformados em RDF num processo conhecido como ***triplificação*** (SUKHOBOK et al., 2016). Portanto, a segunda etapa do processo de publicação consiste em construir uma representação RDF a partir dos dados limpos de entrada. Depois que a representação em RDF do dado original é construída, são criados links para dados externos (VOLZ et al., 2009) num processo conhecido como ***ligação***.

O último passo consiste em publicar esses dados na Web para consumo. Dados conectados podem ficar disponíveis de três maneiras: (i) através de URIs derreferenciáveis; (ii) através de um endpoint SPARQL; e (iii) através de arquivos RDF (WOOD et al., 2014) serializados.

2.2.1 Serializações RDF

RDF é um modelo abstrato de como um arquivo deve ser criado, que pode ser representado de diversos formatos sintáticos conhecidos como serializações. XML é um desses formatos, e é a sintaxe recomendada pela W3C para escrever RDF, porém RDF não precisa estritamente ser escrito em XML (TOMASZUK, 2009).

RDF pode ser escrito em serializações como JSON, N-Triples, Turtle, TriG, N-Quads, RDFa, N3 para citar algumas. Todas essas serializações são formas válidas de escrever um arquivo RDF e possuem suas vantagens e desvantagens – algumas são mais apropriadas para o processamento de máquina (como a N-Triples), outras são mais legíveis para as pessoas (como a Turtle e JSON).

Neste trabalho apenas em três serializações serão utilizadas: XML – que é recomendada pela W3C –, N-Triples – usada para inserir dados num banco de triplas –, e Turtle – que é a mais legível das três para humanos.

2.2.2 Ligação

A última etapa no processo de publicação de dados conectados antes da publicação propriamente dita é a ligação. Ligação (ou *Link Discovery*) é o problema de encontrar *links* entre dois ou mais conjuntos de dados conectados baseado em alguma regra (KEJRIWAL; MIRANKER, 2014). Dois conjuntos de dados conectados, por exemplo, podem conter informações sobre a mesma entidade porém cada um usando uma URI única para representar esta entidade. A ligação poderia criar uma tripla com a regra *owl:sameAs* para dizer que essas duas URIs se referem a mesma coisa (KHATCHADOURIAN; CONSENS, 2010).

Várias *frameworks* de geração automática de links entre diferentes conjuntos de dados conectados estão disponíveis. Elas provêem uma linguagem declarativa para especificar que tipo de links RDF devem ser criados. Esses links são encontrados através de uma combinação de métricas de similaridade, que são agregadas retornando um valor de o quão similar duas URIs daquele tipo são (BIZER; HEATH; BERNES-LEE, 2009). O Silk é um exemplo desse tipo de *framework*. Ele usa uma linguagem declarativa conhecida como Silk-LSL onde pode ser especificado os tipos de links a serem descobertos de acordo com as regras de similaridades (VOLZ et al., 2009). Uma vez que os dados tenham passado por essa etapa de ligação, eles estarão prontos para serem publicados.

2.3 TRABALHOS RELACIONADOS

O processo de limpeza e tratamento de qualidade de dados é geralmente considerado um dos mais demorados e caros em todo o processo de publicação de dados conectados – de acordo com algumas fontes pode chegar até 80% do tempo de todo o processo (SUKHOBOK et al., 2016).

Portanto, precisamos de uma *framework* que unifique o processo de limpeza dos dados e triplificação que seja poderosa o suficiente para lidar com os problemas de limpeza de dados mais comuns e simples o suficiente que até não programadores consigam usá-la (SUKHOBOK et al., 2016).

Nesta seção apresentamos algumas soluções para este processo. Primeiramente, falaremos do ETL4LOD, em seguida abordaremos as soluções estudadas como possíveis alternativas do ETL4LOD – Karma, Open Refine e Any23.

2.3.1 ETL4LOD

Devido às similaridades entre ETL e o processo de publicação LOD, o projeto LinkedDataBR usou uma ferramenta de ETL, mais especificamente a Pentaho Data Integration ou Kettle, para implementar o fluxo de publicação (CORDEIRO et al., 2011) de dados conectados. A extensão do Kettle que implementa esse fluxo de publicação de LOD ficou apelidada de ETL4LOD (lê-se: ETL for *lódi*), que pode ser vista na Figura 3.

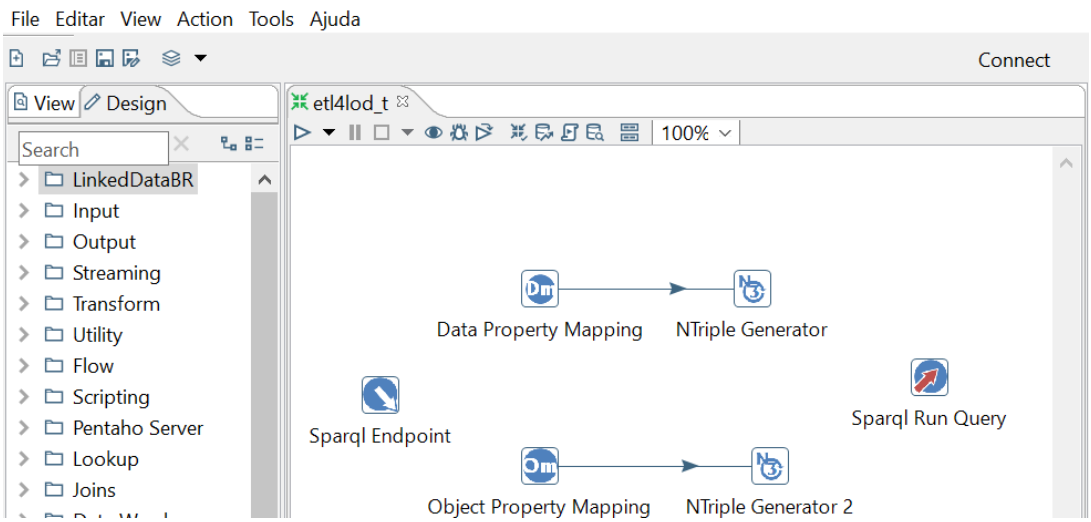


Figura 3 – Interface do ETL4LOD

O ETL4LOD foi implementado como um conjunto de *plug-ins* que estendem as funcionalidades do Kettle para trabalhar com LOD. Os motivos que levaram a escolha do Kettle como base foram:

Primeiro, o Kettle provê uma interface gráfica intuitiva baseada no modelo de *drag&drop* para criar fluxos de ETL, onde um fluxo de ETL do Kettle (Figura 4) é um grafo direcionado onde os nós são representados por *plug-ins* e as arestas indicam a ordem na qual esses *plug-ins* serão executados.

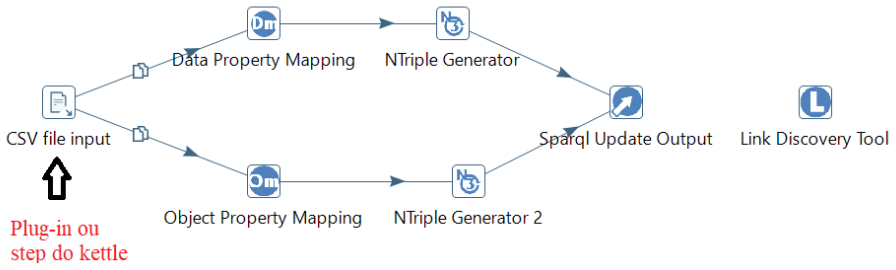


Figura 4 – Fluxo de ETL no Kettle

Segundo, o Kettle permite que suas funcionalidades sejam estendidas através da criação de *plug-ins* (CORDEIRO et al., 2011) usando a linguagem Java. A criação de um *plug-in* para o Kettle é bem documentada[3] e simples de ser feita. De forma simplificada, ela envolve a criação de apenas quatro arquivos Java (Figura 5) – um para definir a interface gráfica, um com a lógica do *plug-in* que define como a entrada será usada para produzir a saída, um com as configurações que serão salvas e, por fim, um que serve para salvar o estado do *plug-in* durante sua execução. O Apêndice E tem um esqueleto para criação de *plug-ins*.

Java Interface	Base Class	Main Responsibilities
`StepMetaInterface`	`BaseStepMeta`	• Save and load step settings
`StepDialogInterface`	`org.pentaho.di.ui.trans.step.BaseStepDialog`	• Step settings dialog
`StepInterface`	`BaseStep`	• Process rows
`StepDataInterface`	`BaseStepData`	• Store processing state, and to declare and serve as a place for field variables during row processing

Figura 5 – Classes que compõem um plug-in Kettle

[3] https://help.pentaho.com/Documentation/8.2/Developer_Center/PDI/Extend/000

Por último, pelo Kettle já ser uma ferramenta de ETL, as funções que tratam da qualidade dos dados de entrada já estão implementadas.

O ETL4LOD, em sua versão original, possuía o seguinte conjunto de *plug-ins*:

Any23 Converter: permite a conversão dos dados a serem publicados nos formatos aceitos pelo processo de LOD – N-Triples, RDF/XML e Turtle.

Sparql Endpoint: permite a construção de consultas SPARQL em endpoints específicos através do processo de publicação.

Sparql Update Insert: permite a manipulação dos dados armazenados em um endpoint específico através do processo de publicação.

Data Property Mapping; permite a anotação dos dados de um determinado domínio a partir dos conceitos presentes em uma ontologia de referência.

Object Property Mapping: permite a anotação dos relacionamentos existentes entre os dados de um determinado domínio a partir dos conceitos presentes em uma ontologia de referência.

Apesar de ser uma boa solução para implementar o fluxo de publicação de dados conectados, alguns problemas foram encontrados com o ETL4LOD nos testes realizados com a ferramenta. O principais problemas foram:

Versão defasada dos *plug-ins*. O ETL4LOD foi criado em 2011 quando o Kettle estava na versão 4. O Kettle atualmente se encontra na versão 8.1 e, devido as suas atualizações, não é mais compatível com os *plug-ins* criados em 2011.

Documentação insuficiente dos *plug-ins*. Apesar de um dos pontos fortes do Kettle ser a sua interface amigável ao usuário, seus *plug-ins* ainda possuem uma curva de aprendizado não trivial. A falta de uma documentação dos *plug-ins* do ETL4LOD tornou o uso da *framework* bem mais complicado do que deveria ser.

Algumas funcionalidades desejáveis não foram integradas. Não existe um *plug-in* para procurar ou adicionar ontologias ou para integrar o processo de ligação com dados na Web ao Kettle, sendo necessário o uso de funcionalidades implementadas na plataforma LinkedDataBR (não mais disponível) fora do Kettle.

O objetivo deste trabalho é resolver os problemas que foram encontrados no ETL4LOD e aumentar o ecossistema de *plug-ins* presente na ferramenta, criando assim o ETL4LOD+.

2.3.2 Karma

O Karma (Figura 6), desenvolvido por Knoblock et al (2012), é uma ferramenta de limpeza e triplificação de dados desenvolvida na *University of Southern California* que permite que usuários integrem uma variedade de fontes de dados. A interface gráfica aliada à facilidade de uso são os grandes atrativos da ferramenta (DANTAS, 2018).

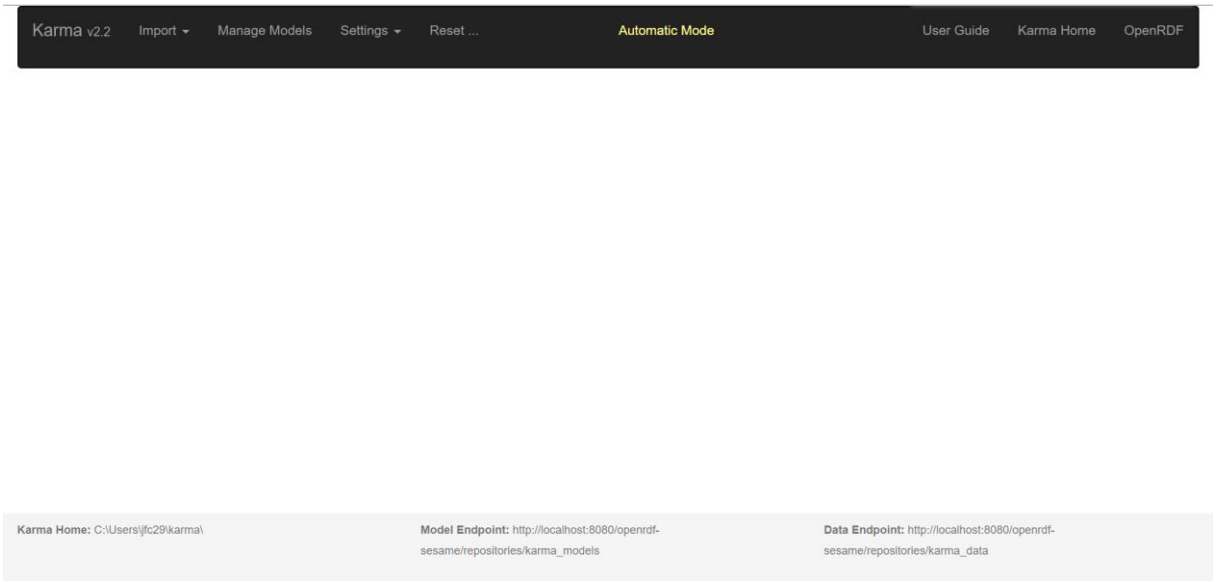


Figura 6 – Página inicial do Karma.

Segundo Dantas (2018), a experiência com o Karma foi positiva por ele ser de fácil instalação, ser amigável ao usuário e fazer integração com ontologias.

Medeiros e Alves (2018), no entanto, descrevem uma experiência menos positiva com uso do Karma para a triplificação dos dados testados devido a alguns problemas encontrados, como a falta de escalabilidade do Karma, a inabilidade de ele trabalhar com alguns tipos de ontologias e por não ser possível salvar um trabalho de modelagem incompleto.

O Karma também possui uma quantidade bem pequena de funções para limpeza de dados se comparado ao ETL4LOD, o que o torna uma solução bem menos poderosa para problemas que exijam grande pré-processamento de dados fonte. Por causa desses problemas

descritos por Medeiros e Alves (2018), o Karma não se mostrou ideal para diversos problemas de triplificação e limpeza de dados.

2.3.3 Open Refine

O Open Refine é uma ferramenta adquirida pela Google que, com a adição do *plug-in* de extensão RDF consegue gerar dados conectados com valor semântico (DANTAS, 2018).

De acordo com (2018), Dantas a limpeza no Open Refine foi a melhor encontrada entre as ferramentas testadas – Any23, Karma e Open Refine. Apesar de possuir uma considerável gama de funções para limpeza de dados, o Open Refine (Figura 7) não possui uma quantidade tão grande quanto a presente no Kettle, onde o ETL4LOD foi desenvolvido.

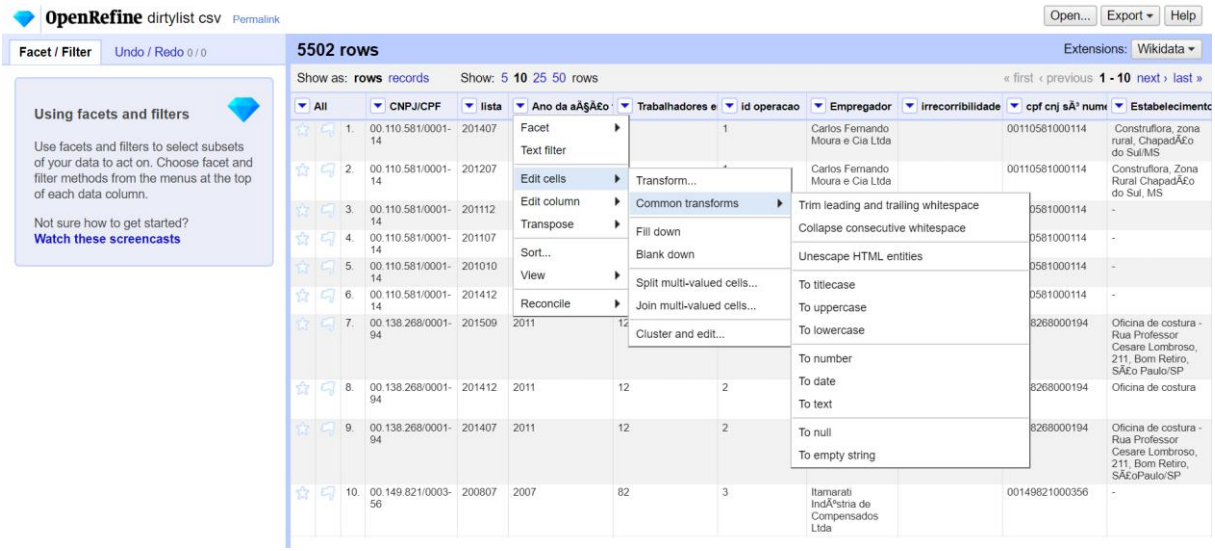


Figura 7 – Funções de limpeza de dados do Open Refine

O Open Refine cumpre seu papel em gerar RDF com valor semântico (DANTAS, 2018). Segundo MEDEIROS e ALVES (2018), porém, ele não possui a integração com nenhum tipo de API para o uso de ontologias que não estejam disponíveis em formato de arquivos. Isso é problemático para o caso de ontologias que sejam disponibilizadas somente como URIs, como é o caso da ontologia MEDDRA[4] usada por elas. Por este motivo, o Open Refine também se mostrou não ideal em diversos problemas de triplificação de dados.

[4] http://purl.bioontology.org/ontology/MEDDRA/10047862

2.3.4 Any23

O Any23 – *Anything to Triples* – é uma ferramenta de triplificação disponibilizada pela Apache Software Foundation e pode ser usada de diversas formas para converter dados em RDF (DANTAS, 2018). Ela é tão versátil que a próprio ETL4LOD costumava ter um *plug-in* que a integrava ao seu ecossistema.

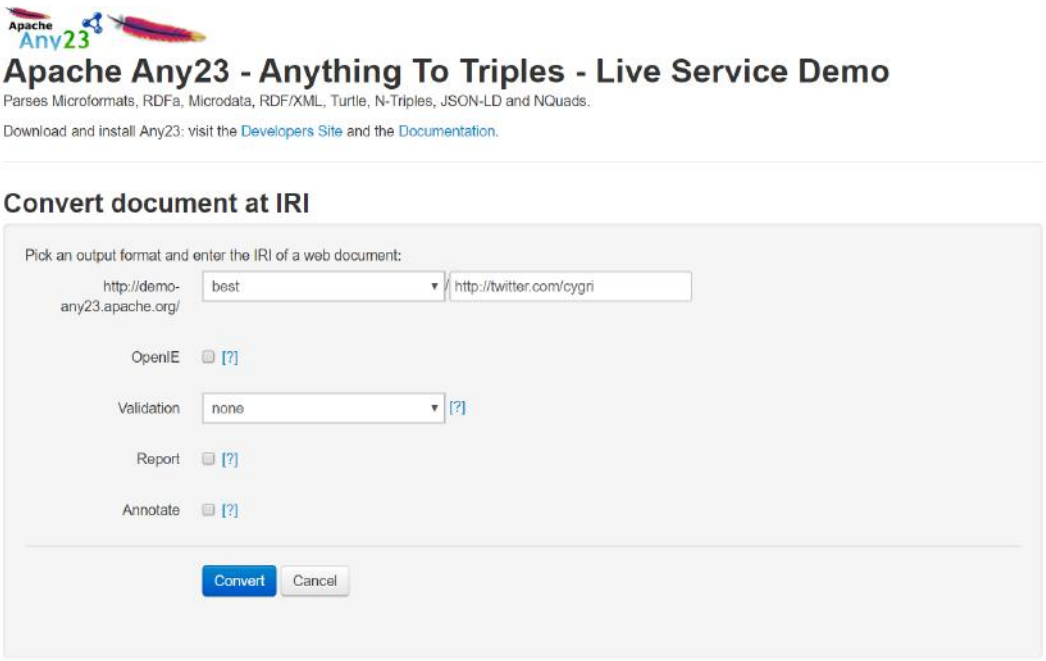


Figura 8 – Interface gráfica do Any23

Apesar da versatilidade, a utilização do Any23 (Figura 8) é menos intuitiva em relação ao Karma e no momento da triplificação não considera nenhuma ontologia, resultando em um RDF com pouco valor semântico (DANTAS, 2018). A ferramenta também não possui nenhuma funcionalidade de limpeza de dados, que é um passo fundamental no pré-processamento de dados não tratados. Portanto, o Any23 geralmente não pode ser usado sozinho para criação de dados conectados.

3 ETL4LOD+

Em virtude da necessidade de investirmos em uma ferramenta de escopo amplo para apoio ao processo de tratamento, triplificação e publicação de dados conectados, o ETL4LOD+ foi criado com o ETL4LOD como base.

O ETL4LOD+ é uma extensão do ETL4LOD cujo objetivo é fornecer uma *framework* amigável ao usuário que permita cobrir as atividades fundamentais do ciclo de vida de dados do tipo LOD (Figura 9).

Figura 9 – Ciclo de vida dos dados LOD

O ciclo de vida dos dados LOD começa com o planejamento, onde são especificadas as informações que serão publicadas e as fontes de dados que serão durante a etapa de publicação. A etapa de planejamento é dividida em: análise de requisitos, análise do ambiente e modelagem dos dados. As primeiras duas subetapas tratam, respectivamente, dessa especificação de informações e fonte de dados a serem usados. A modelagem de dados cria uma representação gráfica do esquema das fontes de dados.

As fontes de dados especificadas na etapa anterior são, em seguida, usadas na etapa de publicação. A publicação é dividida em três etapas: pré-processamento, triplificação e ligação. Os *plug-ins* criados para o ETL4LOD+ se encaixam em sua maioria nas etapas de triplificação e ligação, enquanto que os já existentes no Kettle na etapa de pré-processamento.

Uma vez que os dados passem pela ligação, eles estão prontos para serem consumidos na Web. A última etapa no ciclo trata de como os dados ficarão disponíveis para consumo.

3.1 PASSOS PARA CRIAÇÃO DO ETL4LOD+

A primeira etapa na criação do ETL4LOD+ foi a centralização de todos os *plug-ins* que tinham sido criados para o ETL4LOD em um único repositório, garantindo maior facilidade para padronização do código e futura liberação de novas versões dos *plug-ins*.

A centralização dos *plug-ins* foi seguida da atualização de todas as dependências utilizadas por eles (Figura 10) garantindo assim que possíveis *bugs* e falhas de segurança presentes nas versões desatualizadas das dependências não afetassem as novas versões dos *plug-ins*. A atualização das dependências forçou mudanças no código para substituir funções que tivessem entrado em desuso ou simplesmente não existissem mais. Detalhes sobre as mudanças feitas podem ser vistas no *changelog*[5] presente na documentação do ETL4LOD+.

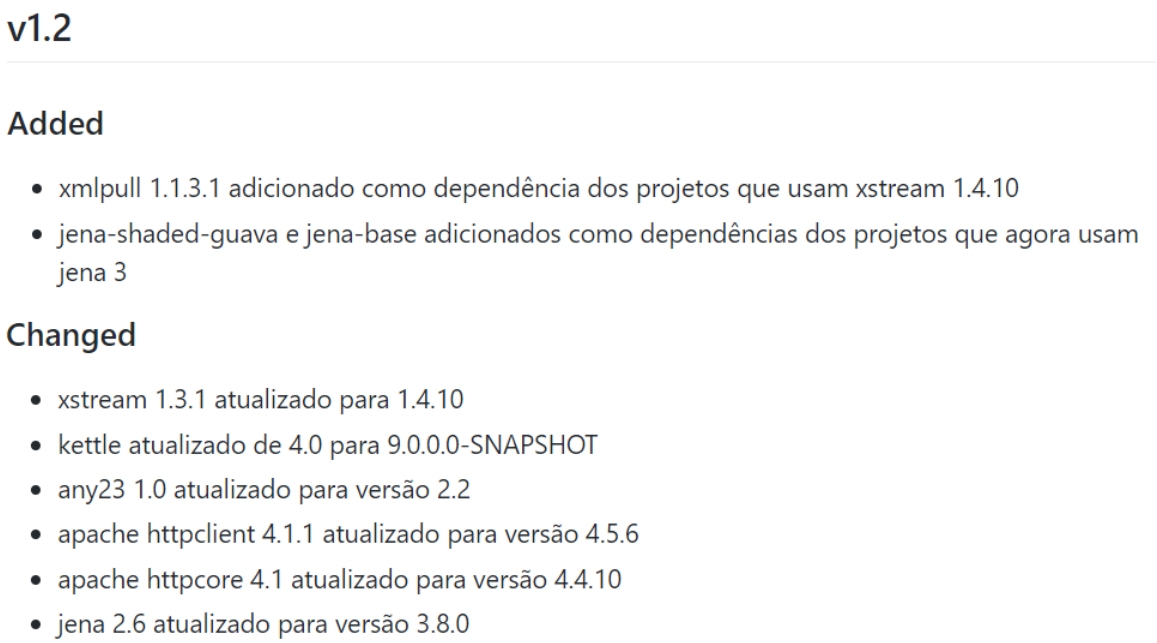


Figura 10 – *Changelog* das primeiras mudanças feitas nas dependências dos *plug-ins*

Em seguida, o código de alguns dos *plug-ins* foi alterado para resolver *bugs* de funcionalidade e para melhorar a legibilidade e manutenção do código. O principal problema no código encontrava-se na forma como era feita a serialização, que era incompatível com o Kettle 8.1. Por isso, foi necessária a alteração na estrutura do XML usado para abrir e salvar os arquivos do Kettle que contivessem *plug-ins* do ETL4LOD. Como consequência, alguns *plug-ins* do ETL4LOD+ não oferecem retrocompatibilidade com o ETL4LOD (Tabela 1).

[5] https://github.com/johncurcio/ETL4LODPlus/wiki/Changelog

Tabela 1: lista de *plug-ins* sem retrocompatibilidade com o ETL4LOD

PLUG-IN	MOTIVO DA PERDA DE COMPATIBILIDADE
Data Property Mapping	A forma de serializar os arquivos contendo esse *plug-in* foi reestruturada. O arquivo de configuração teve seu id alterado.
Object Property Mapping	A forma de serializar os arquivos contendo esse *plug-in* foi reestruturada.
Sparql Endpoint	A forma de serializar os arquivos contendo esse *plug-in* foi reestruturada.
Graph Sparql Endpoint	A forma de serializar os arquivos contendo esse *plug-in* foi reestruturada.

Para mitigar o problema da retrocompatibilidade, uma versão v1.0-alpha[6] dos *plug-ins* foi liberada com o intuito de permitir que arquivos antigos do ETL4LOD sejam abertos e lidos no Kettle 8.1, porém eles não podem ser modificados.

Após isso, todos os *plug-ins* foram internacionalizados em inglês e português, permitindo que eles sejam vistos na linguagem definida no Kettle.

Por último, novos *plug-ins* – para busca e seleção de termos de ontologias, Owl Input, e para ligação de fontes de dados, Link Discovery Tool – foram adicionados ao ecossistema do ETL4LOD+ e uma documentação foi criada no GitHub[7].

3.2 PLUG-INS

Os *plug-ins* mais utilizados do ETL4LOD+ durante qualquer tarefa de triplificação de dados são o Data Property Mapping e Object Property Mapping – para realizar o mapeamento dos dados de entrada em triplas –, N-Triple generator – para serializar as triplas em N-Triples –, e Sparql Update Output – para publicar essas triplas num *endpoint* SPARQL. Esses *plug-ins* juntos formam um fluxo que se repete com frequência durante a triplificação com o ETL4LOD+ (Figura 11).

[6] https://github.com/johncurcio/ETL4LODPlus/releases/tag/v1.0-alpha

[7] https://github.com/johncurcio/ETL4LODPlus/wiki

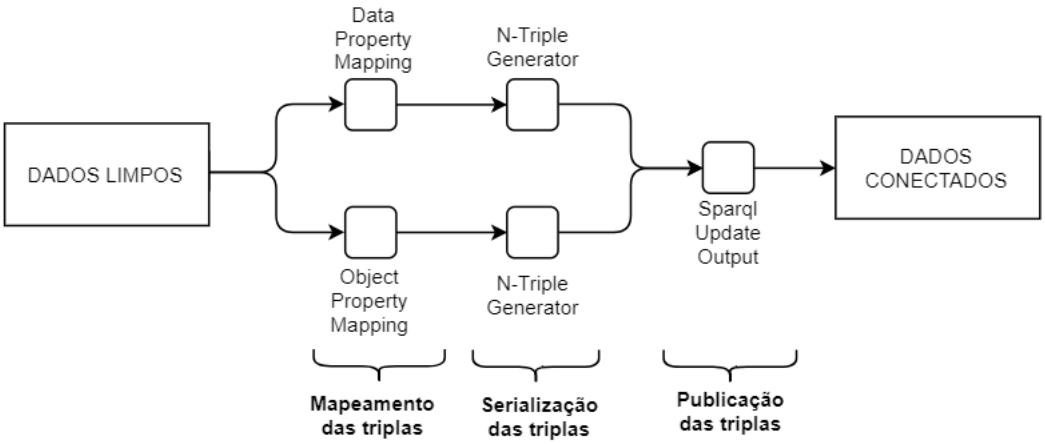


Figura 11 – Fluxo mais usado na triplificação de dados no ETL4LOD+

Adicionalmente esse fluxo pode conter a agregação de ontologias em arquivos provido pelo Owl Input ou um processo de ligação usando o Link Discovery Tool. O Sparql Update Output pode, com o uso de *plug-ins* adicionais do Kettle, ser substituído pelo Sparql Run Query. Em casos mais específicos, esse fluxo principal inteiro pode ser substituído por algum outro grupo de *plug-ins*. A Figura 12 representa os fluxos suportados pelo ETL4LOD+ na triplificação de dados.

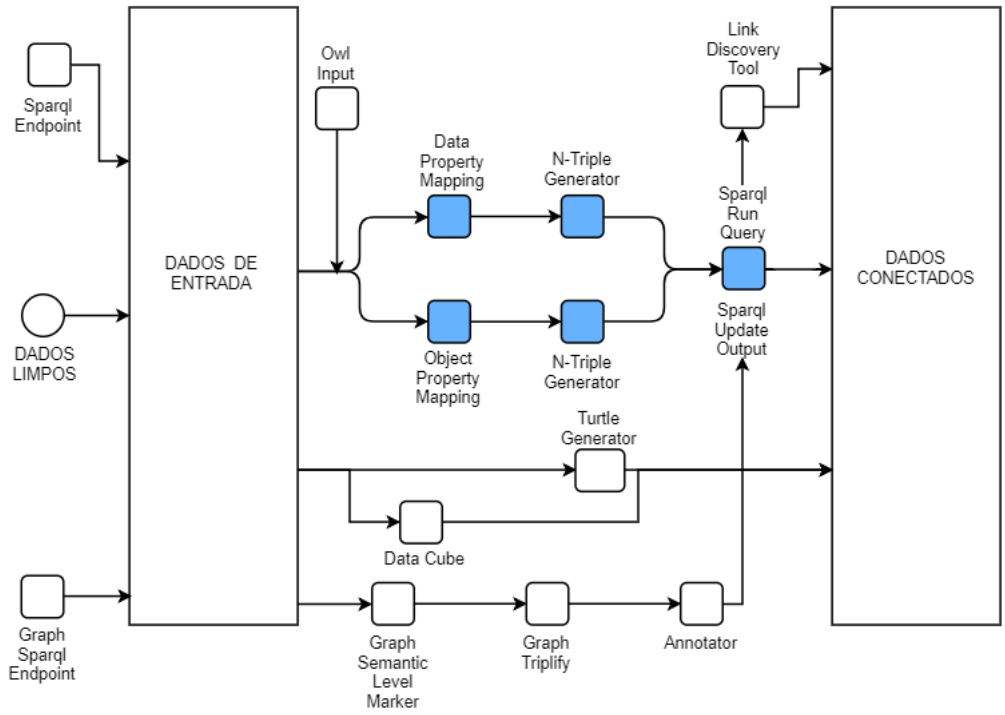


Figura 12 – Fluxos de triplificação suportados pelo ETL4LOD+

Uma descrição detalhada da forma de uso e cada um dos *plug-ins* presentes no ETL4LOD+ pode ser encontrada na documentação da ferramenta disponível em *https://github.com/johncurcio/ETL4LODPlus/wiki*.

3.2.1 Data Property Mapping

O Data Property Mapping (Figura 13) fornece a habilidade de mapear os campos de entrada em triplas RDF, indicando quem é o sujeito, objeto e predicado da tripla quando o objeto é um literal. Também permite que esse literal seja anotado com informações do tipo de dado e da linguagem usada.

Para o ETL4LOD+ o Data Property Mapping foi modificado para aceitar predicados que viessem tanto de um *plug-in* anterior a ele no fluxo do Kettle quanto de um texto digitado pelo usuário no campo de predicado.

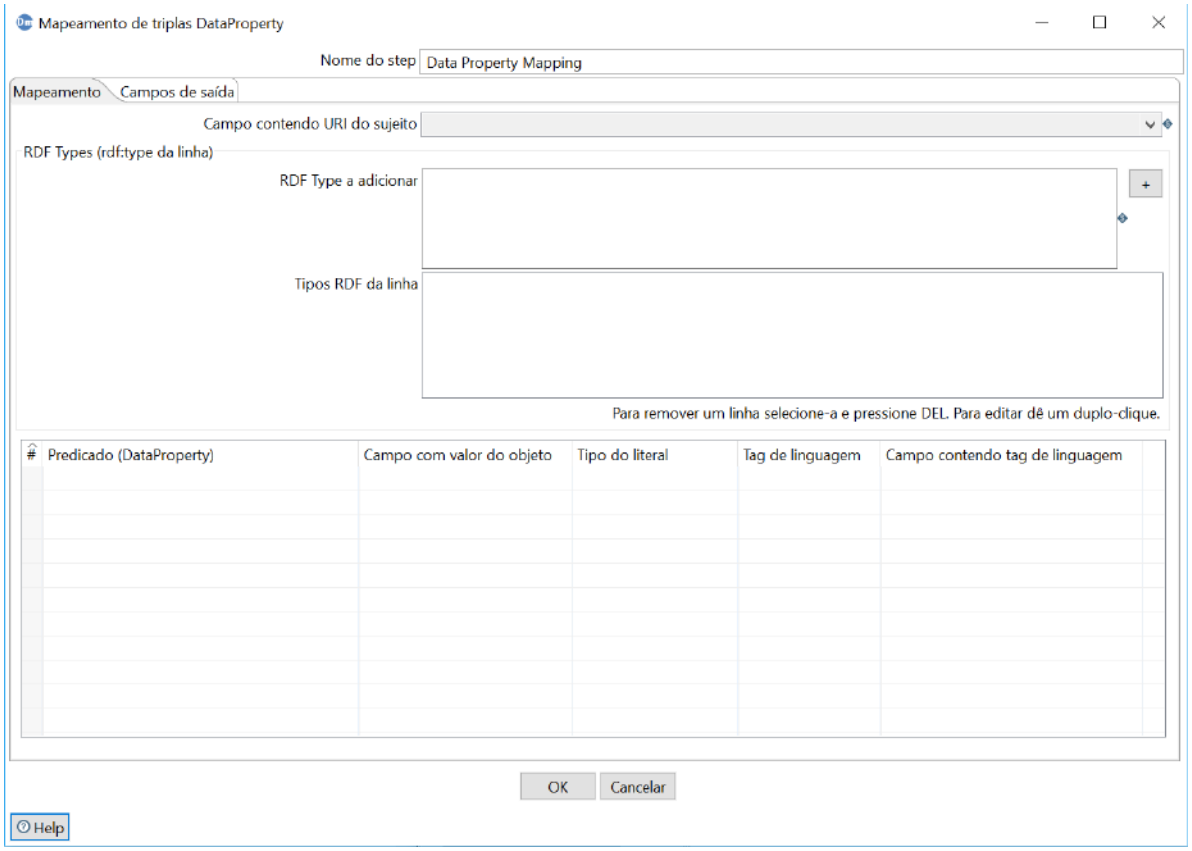


Figura 13 – Interface gráfica do Data Property Mapping

3.2.2 Object Property Mapping

O Object Property Mapping (Figura 14) possui um funcionamento similar ao do Data Property Mapping, porém os objetos mapeados no Object Property Mapping são URIs ao invés de literais.

Para o ETL4LOD+ o Object Property Mapping também foi modificado para aceitar predicados que viessem tanto de um *plug-in* anterior a ele no fluxo do Kettle quanto de um texto digitado pelo usuário no campo de predicado.

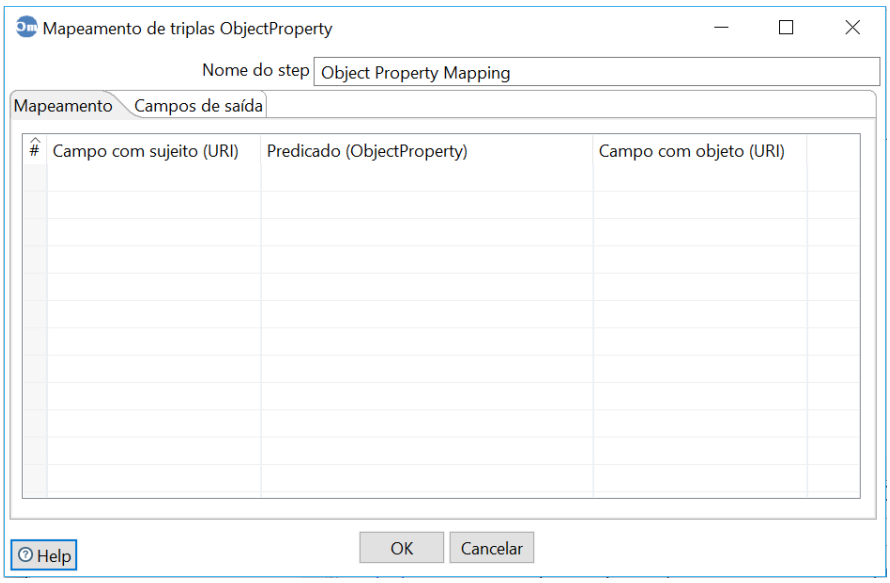


Figura 14 – Interface gráfica do Object Property Mapping

3.2.3 N-Triple Generator

O N-Triple Generator (Figura 15) é um *plug-in* que provê a funcionalidade de transformar os dados de entrada em sentenças RDF no formato N-Triples, do tipo <*sujeito*> <*predicado*> <*objeto*>, que podem ser posteriormente usadas pelo Sparql Update Output para inserir esses dados em um *endpoint* SPARQL. Ele espera receber de entrada os campos contendo o sujeito, predicado e objeto (adicionalmente o tipo e linguagem do objeto se ele for um literal) e retorna um único campo com a serialização N-Triples correspondente.

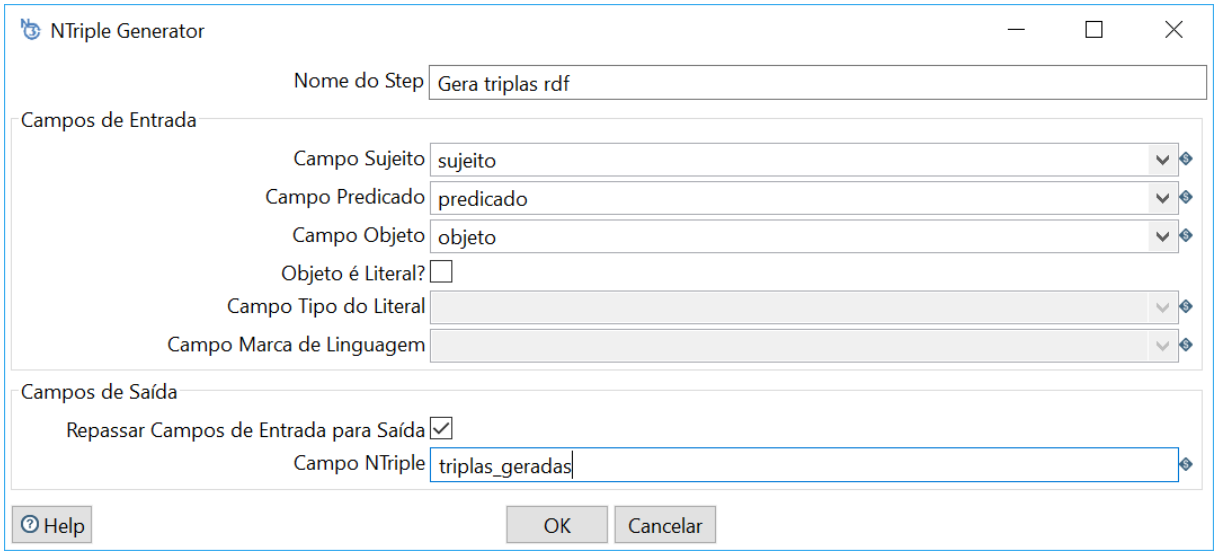


Figura 15 – Interface gráfica do NTriple Generator

3.2.4 Sparql Endpoint

O Sparql Enpoint permite extrair dados de um banco de triplas através de uma consulta SPARQL do tipo *SELECT*. Ele retorna todas as variáveis que sejam definidas na consulta para a saída do Kettle. No exemplo da Figura 16, serão retonados os campos s, p e o extraídos do grafo *http://www.lodbr.ufrj.br/lista_suja*.

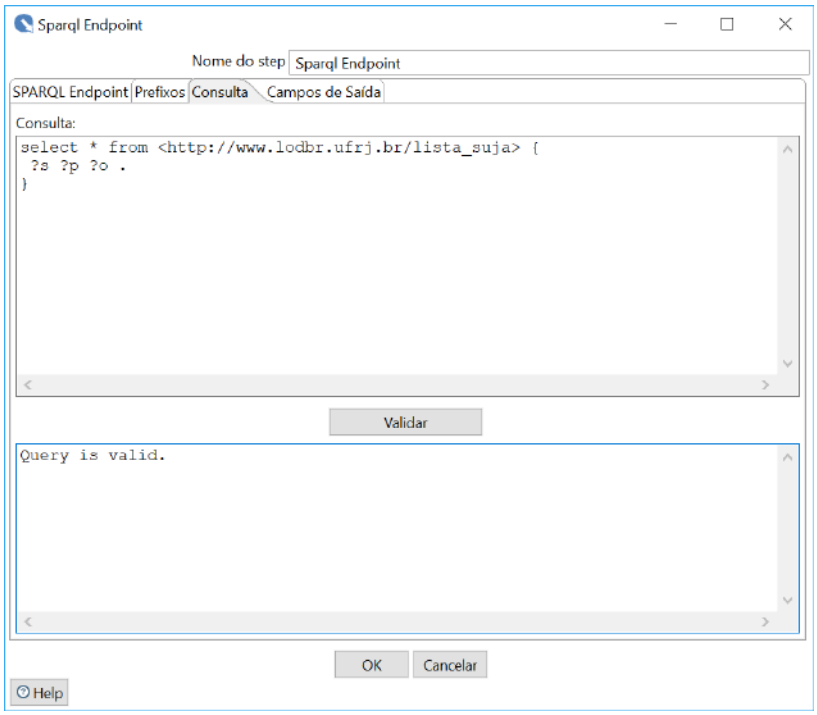


Figura 16 – Exemplo de consulta aceita pelo SPARQL Endpoint

O Sparql Endpoint precisou ter grande parte do seu código modificado quando a dependência Jena foi atualizada da versão 2 para a 3, isso foi devido à atualização do Jena ter removido algumas funções que eram utilizadas pelo *plug-in* que precisaram ser substituídas.

3.2.5 Sparql Run Query

O Sparql Run Query permite executar uma consulta para atualizar um conjunto de dados, tal como *UPDATE*, *DELETE* e *DROP*. Essa consulta precisa estar definida em um campo vindo de um *plug-in* anterior e cada linha desse campo precisa se uma consulta inteira a ser executada. A consulta da Figura 17 pode ser armazenada, por exemplo, em um campo *queries* no fluxo do Kettle.

```
DELETE DATA FROM <http://lodbr.ufrj.br/lista_suja> {
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/atividade> "Corte de
Cana" }
```

Figura 17 – Exemplo de consulta que pode ser feita no SPARQL Run Query

Dessa forma, a configuração apresentada na Figura 18 executa essa consulta no *endpoint* disponível em *http://localhost:8890/sparql*, retornando uma mensagem de sucesso ou de erro.

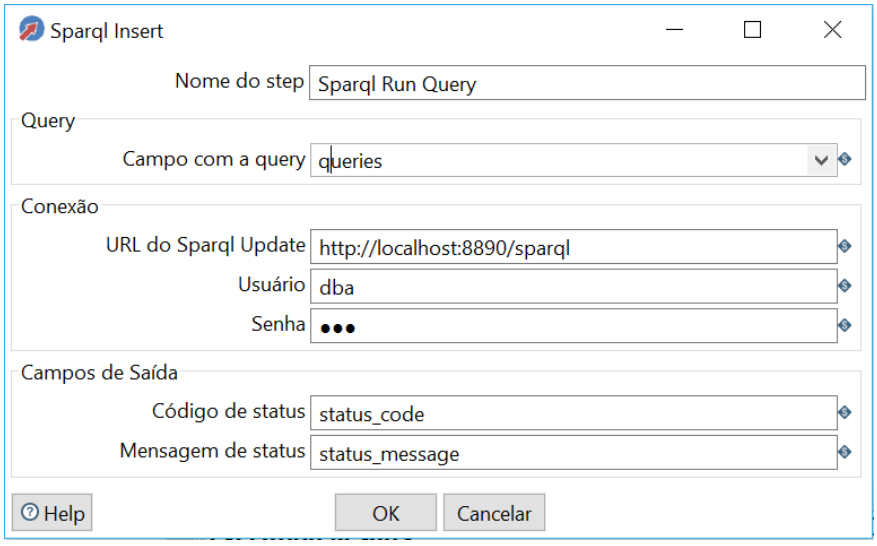


Figura 18 – Exemplo de configuração do SPARQL Run Query

3.2.6 Sparql Update Output

O Sparql Update Output permite inserir e atualizar triplas RDF num banco de triplas usando um campo vindo do *plug-in* anterior com triplas no formato N-Triples.

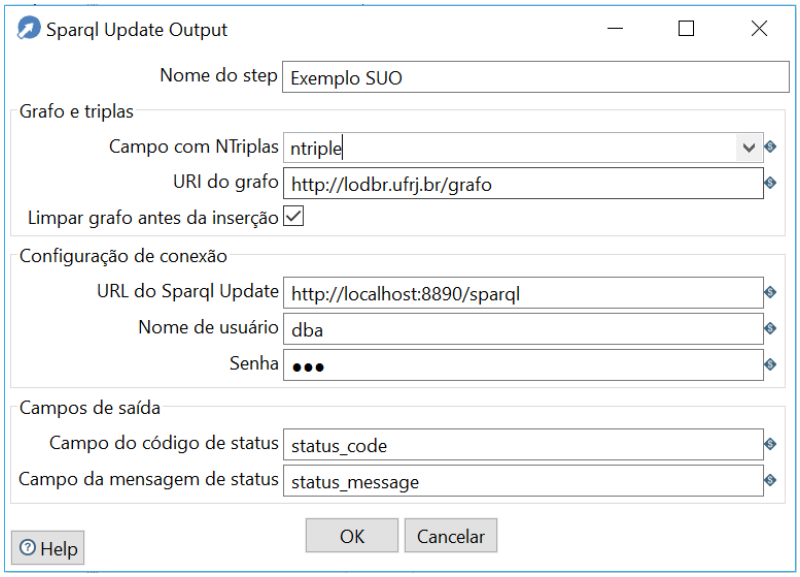


Figura 19 – Exemplo de configuração do SPARQL Update Output

Na Figura 19, o campo *ntriple* vindo de um NTriple Generator contém um conjunto de triplas serializadas em RDF/Triples que serão inseridas no grafo *http://lodbr.ufrj.br/grafo* em um *endpoint* disponível em *http://localhost:8890/sparql*.

3.2.7 ETL4LOD-Graph

No ETL4LOD, cada linha do fluxo representa uma única tripla com sujeito, predicado e objeto. Em ambientes complexos e dinâmicos, como o explorado pelo aDApTA, um recurso pode ser descrito por diferentes propriedades, mesmo quando essas propriedades descrevem o mesmo conceito. Nesses casos, representar os dados como triplas sujeito, predicado e objeto não é suficiente, porque é necessário trabalhar com várias triplas em uma única linha do fluxo – portanto uma linha precisa representar um subgrafo RDF. Para isso, uma extensão conhecida como ETL4LOD-Graph foi criada no projeto aDApTA, que permite processar grafos RDF (CORDEIRO et al., 2011).

Cada um dos *plug-ins* criados para o ETL4LOD-Graph são descritos abaixo. Para este trabalho, apenas o *plug-in* Graph Sparql Endpoint sofreu alterações maiores que as descritas em 3.1.

Annotator: O Annotator (Figura 20) permite anotar uma tripla RDF com termos de uma ontologia de acordo com um mapeamento *de-para* definido em um arquivo XML.

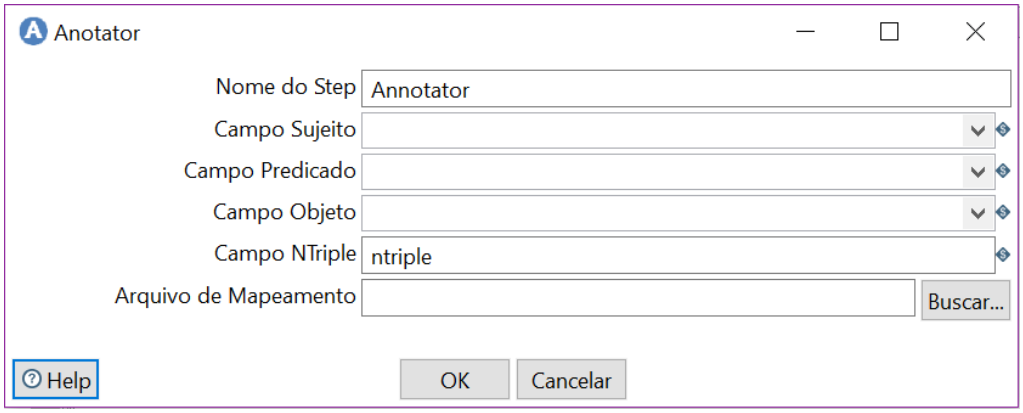


Figura 20 – Interface gráfica do Annotator

Na Figura 21, tem um exemplo de arquivo de mapeamento que pode ser usado pelo Annotator.

```
<map id="1">
  <from>ToWarehouse/Facility/Office</from>
  <to>laiid:sentTo</to>
</map>
<map id="2">
  <from>Name</from>
  <to>rdfs:label</to>
</map>
```

Figura 21 – Arquivo de mapeamento do Annotator

O Annotator procura se o texto escrito na *tag* <*from*> ocorre em algum dos campos definidos como sujeito, predicado e objeto. A ocorrência desse texto é substituída com o texto descrito na *tag* <*to*>.

Graph Semantic Level Marker: permite avaliar o nível de expressividade semântica de um grafo RDF, criando uma nova tripla com o nível marcado nela. Na Figura 22, o *plug-in* recebe de entrada um grafo RDF *resultado* e retorna triplas com sujeito, predicado e objeto marcadas de acordo com o arquivo *SemanticLevelFrameworkRules.xml*.

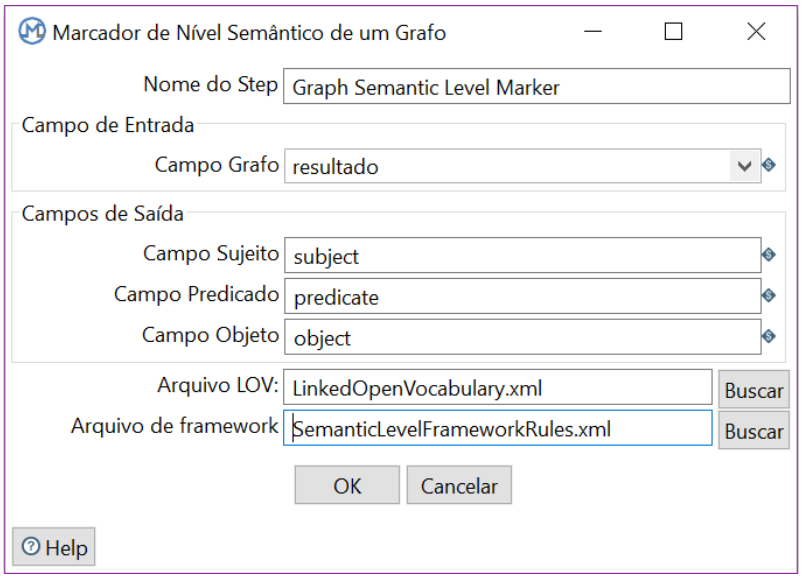


Figura 22 – Exemplo de configuração Graph Semantic Level Maker

O XML *SemanticLevelFrameworkRules* (Figura 23) define o nível de expressividade de um subgrafo como **baixo** se os seus nós são literais, **médio** se seus nós possuem vocabulários, **alto** se seus nós possuem ontologias.

```
<SemanticLevelFramework>
  <Frame id="1">
    <Rule>s.getLiteral() != null</Rule>
    <LevelValue>1</LevelValue>
    <LevelDescription>sstamp:low</LevelDescription>
  </Frame>
  <Frame id="2">
    <Rule>isVocabulary</Rule>
    <LevelValue>2</LevelValue>
    <LevelDescription>sstamp:medium</LevelDescription>
  </Frame>
  <Frame id="3">
    <Rule>isOntology</Rule>
    <LevelValue>3</LevelValue>
    <LevelDescription>sstamp:high</LevelDescription>
  </Frame>
</SemanticLevelFramework>
```

Figura 23 – Configuração do arquivo XML de nível semântico

Se o nível de expressividade for baixo, o Annotator pode ser usado para anotar as triplas com ontologias e aumentar o nível de expressividade do subgrafo.

Graph Sparql Endpoint: permite extrair um grafo RDF de um *endpoint* SPARQL usando os comandos *DESCRIBE* ou *CONSTRUCT*. Este *plug-in* é quase idêntico ao Sparql Endpoint, a diferença está no tipo de consulta executada e no resultado que é um grafo RDF ao invés de dados tabulares.

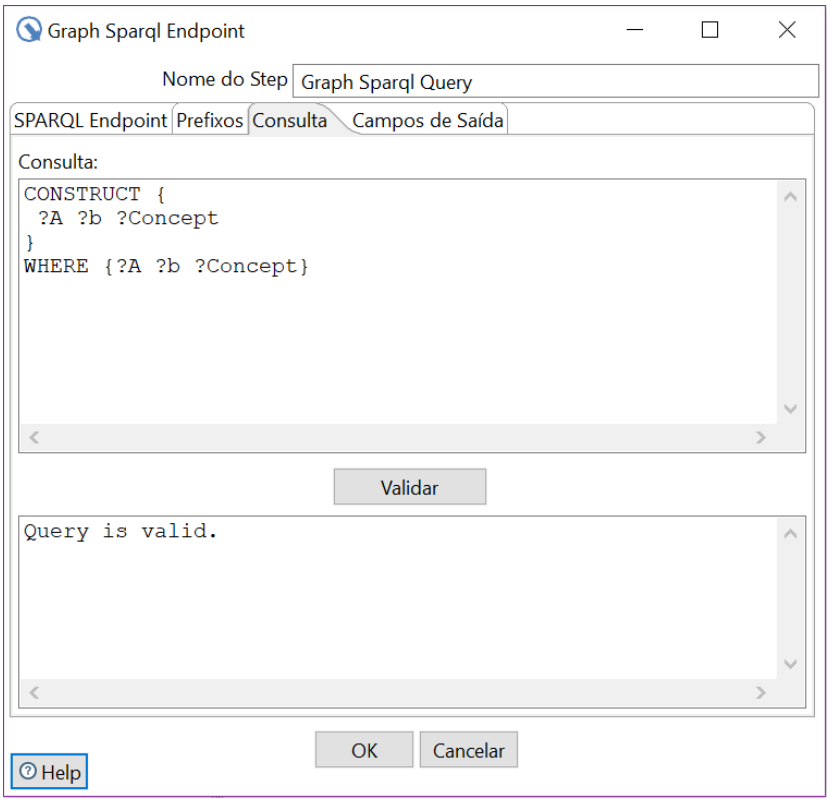


Figura 24 – Exemplo de consulta aceita pelo Graph SPARQL Endpoint

Na Figura 24, encontra-se um exemplo de consulta *CONSTRUCT*, que extrai um grafo RDF com os conceitos presentes no *endpoint* SPARQL.

O Graph Sparql Endpoint sofreu as mesmas modificações descritas para o Sparql Endpoint.

Graph Triplify: em momentos durante o processo de triplificação é necessário trabalhar com cada tripla em separado (ao invés de trabalhar com um subgrafo), este *plug-in* foi desenvolvido para converter um subgrafo RDF em triplas do tipo sujeito, predicado e objeto que podem ser trabalhadas individualmente. A Figura 25 mostra um exemplo de configuração deste *plug-in*.

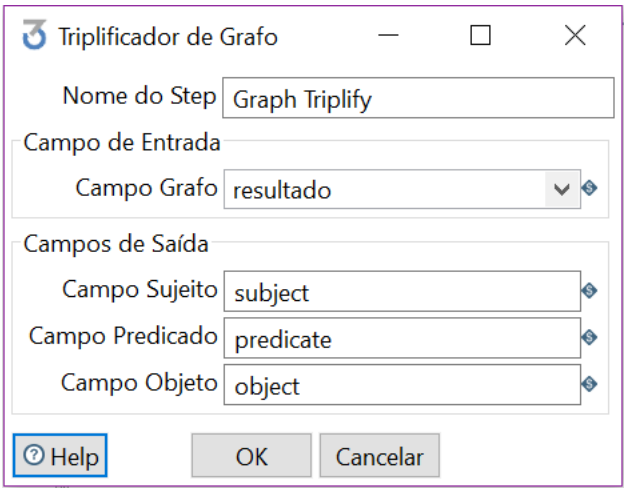


Figura 25 – Exemplo de configuração do Graph Triplify

3.2.8 Turtle Generator

O Turtle Generator permite transformar um arquivo CSV em RDF/Turtle. Para isso, é preciso definir um mapeamento nas configurações do *plug-in* definindo o que cada coluna no arquivo CSV (Apêndice A) representa. Na Figura 26, cada propriedade é uma coluna do arquivo CSV de entrada, o *label* é um texto que descreve essa coluna e a URI é um *link* que será usado para anotar essa propriedade. No fim da execução, o Turtle Generator retorna a triplificação em formato RDF/Turtle (Apêndice B).

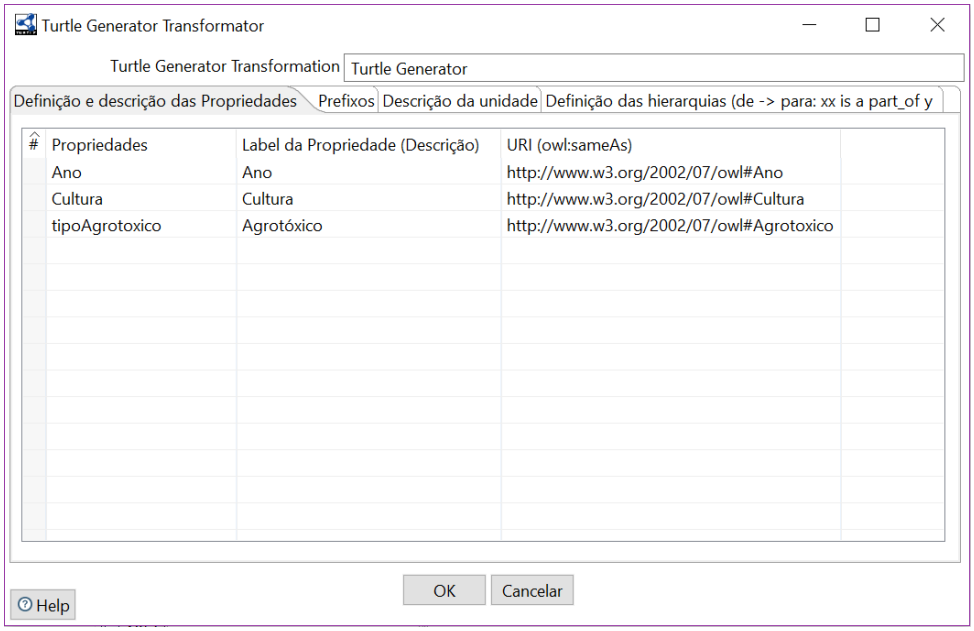


Figura 26 – Exemplo de configuração Turtle Generator

Para o Turtle Generator houve uma completa refatoração do código realizada para padronizar com o código dos outros *plug-ins*.

3.2.9 Data Cube

O Data Cube permite criar arquivos RDF/Turtle usando o vocabulário Data Cube[8] a partir de um arquivo CSV (ou de uma tabela no banco de dados) que contenha uma modelagem multidimensional. O vocabulário Data Cube consiste num conjunto de termos feitos para padronizar a publicação de dados multidimensionais na Web.

Em sua primeira versão liberada para o ETL4LOD, o Data Cube permitia a adição de exatamente 9 dimensões para triplificação. Além das atualizações indicadas em 3.1, o Data Cube sofreu uma mudança completa na estrutura do seu código para que pudesse comportar *n* dimensões no ETL4LOD+.

Na Figura 27, um arquivo CSV com as informações populacionais de uma região em determinado ano (Apêndice C) é dado como entrada para o *plug-in*, que gera um arquivo no formato RDF/Turtle usando o vocabulário Data Cube (Apêndice D).

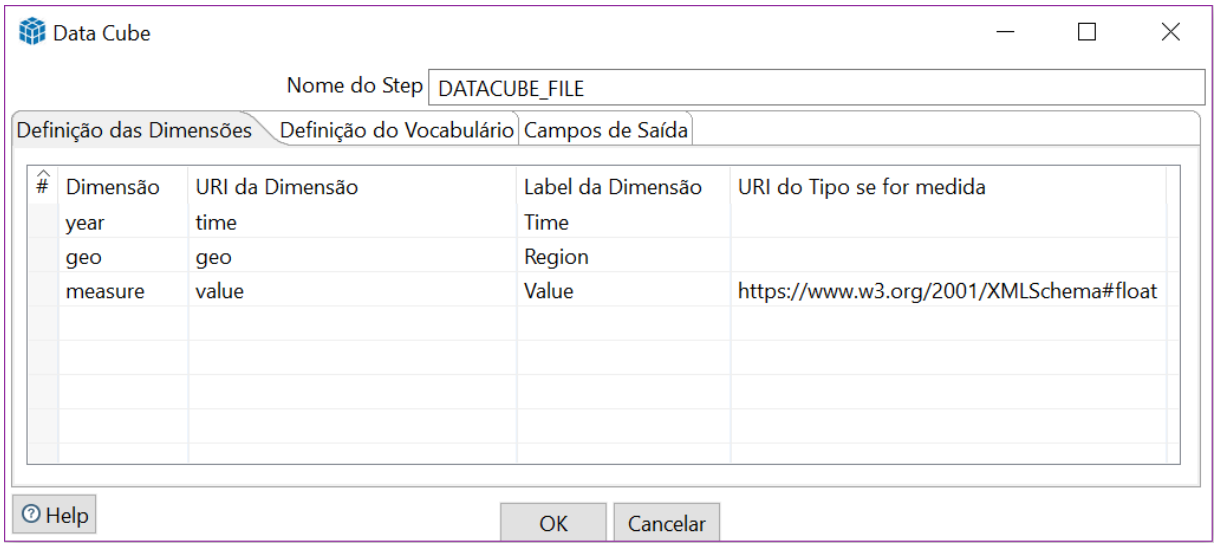


Figura 27 – Exemplo de configuração Data Cube

[8] https://www.w3.org/TR/vocab-data-cube/

3.2.10 Owl Input

O Owl Input é um *plug-in* criado especificamente para o ETL4LOD+ que permite a adição de arquivos de ontologias ou a busca de vocabulários no LOV[9]. Ele foi criado para substituir a funcionalidade de gerenciamento de ontologias presente na plataforma LinkedDataBR e integrá-la ao ETL4LOD+. O Owl Input busca os termos dos vocabulários do LOV ou dentro das ontologias e retorna isso para o usuário. Na Figura 28, o vocabulário *foaf* é procurado no LOV e adicionado a lista do Owl Input clicando no botão Adicionar.

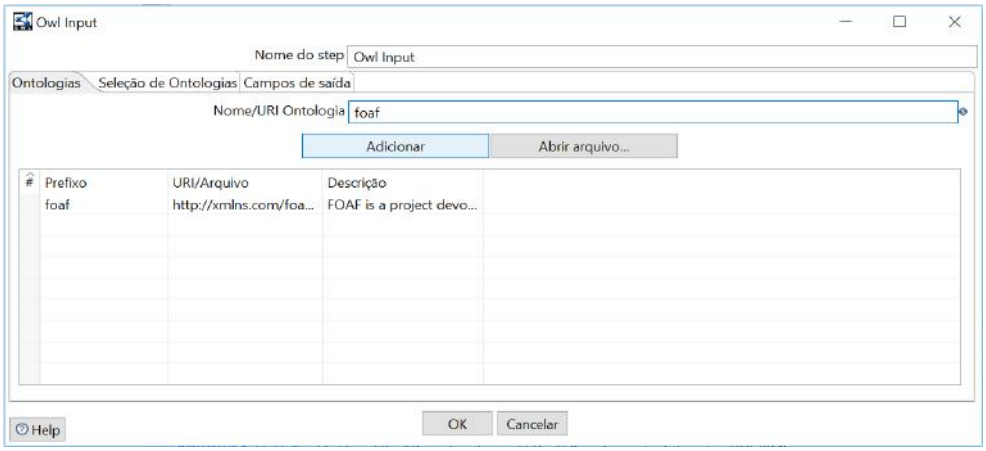


Figura 28 – Busca de ontologias no Owl Input

Ele também possui a funcionalidade de seleção de alguns dos termos para a saída (Figura 29), caso nem todos os termos de um vocabulário sejam necessários para a triplificação, o que geralmente é o caso. Além disso, caso um termo não seja encontrado, ele pode ser manualmente adicionado pelo usuário nesta seção de seleção de vocabulários.

[9] http://lov.linkeddata.es

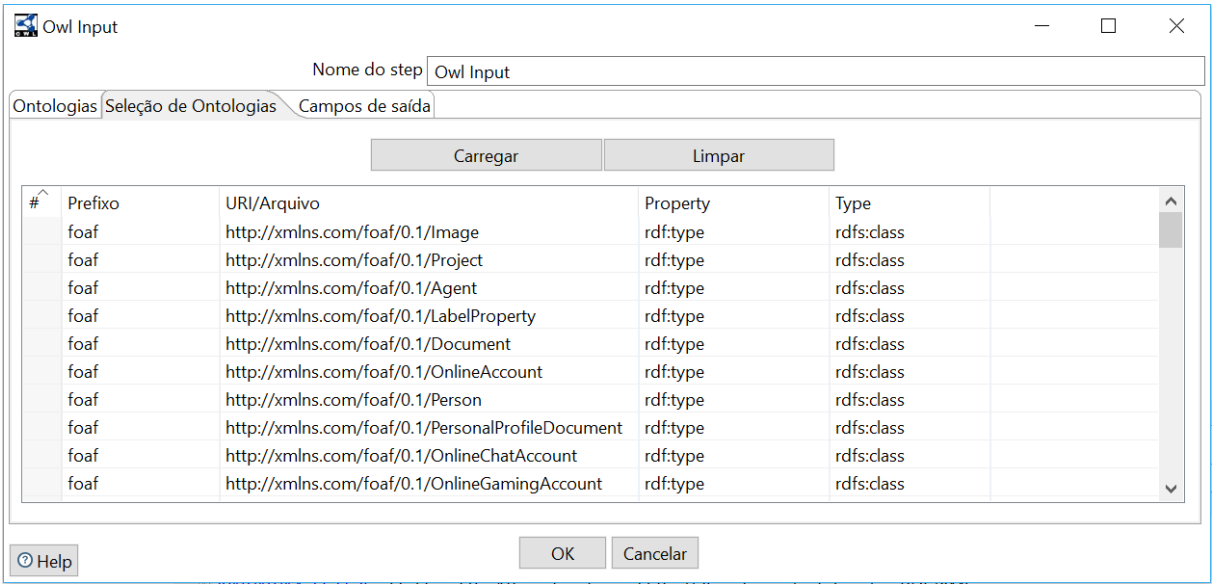


Figura 29 – Seleção de termos Owl Input

Atualmente ele só busca vocabulários presentes no LOV, portanto qualquer vocabulário ou ontologia não indexada pelo LOV precisa ser adicionada através de arquivos.

3.2.11 Link Discovery Tool

O Link Discovery Tool é um *plug-in* criado especificamente para o ETL4LOD+ com o intuito de adicionar a etapa de **ligação** ao Kettle, que antes era feita separadamente no Silk[10]. A intenção do Link Discovery Tool não é substituir o Silk, muito pelo contrário, o *plug-in* usa o Silk para gerar a sua saída. Com a adição do Link Discovery Tool, todas as etapas do ciclo de vida de dados LOD agora podem ser feitas dentro do Kettle.

Na Figura 30, um exemplo de configuração para reproduzir o alinhamento de vocabulários disponível em *https://joinup.ec.europa.eu/document/tutorial-use-silk-aligning-controlled-vocabularies* pode ser vista.

[10] http://silkframework.org

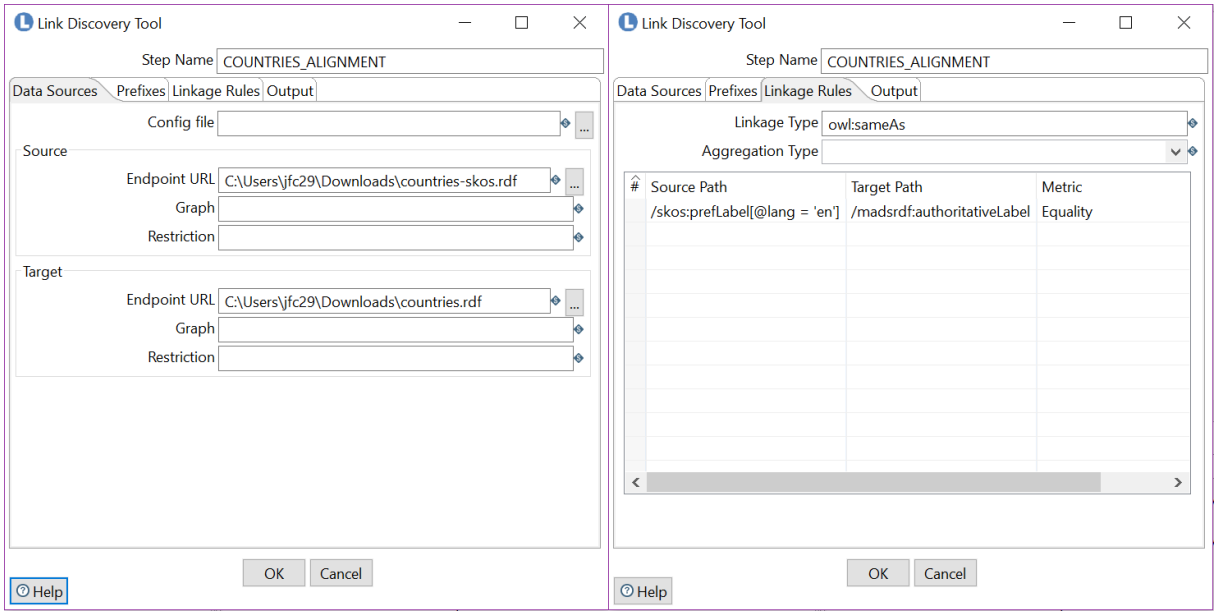


Figura 30 – Exemplo de configuração Link Discovery Tool

O Link Discovery Tool possui algumas limitações conhecidas que não puderam ser concluídas no escopo deste trabalho:

1. **Apenas uma agregação pode ser feita por vez.** Isso acontece devido a uma limitação no frontend do kettle. Idealmente uma GUI para gerar grafos como o jGraphx[11] seria incluída nesse *plug-in*, porém tentativas de usar o jGraphx para integrar capacidades de criar grafos ao Kettle não foram bem sucedidas. Esta limitação é a que possui maior impacto na ligação, porque torna o *plug-in* menos poderoso que o Silk. Para remediar o problema, a capacidade de adicionar um arquivo de configuração manualmente foi adicionada ao *plug-in*.
2. **O *plug-in* executa o Silk pela linha de comando dentro do Java.** Como consequência disso, não é possível retornar a saída gerada pelo Silk para o fluxo do Kettle. A saída precisa ser salva em um arquivo N-Triples ou em um *endpoint SPARQL*. Como a ligação é geralmente a última etapa da geração de dados conectados (salvo a publicação), essa não é uma limitação grande.

O *plug-in* Link Discovery Tool que deveria trabalhar também com o LIMES[12], que é uma ferramenta de funcionalidade similar a do Silk, porém devido a uma série de problemas

[11] https://github.com/jgraph/jgraphx

[12] aksw.org/Projects/LIMES

enfrentados tentando integrar o LIMES ao ETL4LOD+ – incluindo *bugs* para salvar arquivos de configuração e impossibilidade de executar um arquivo de configuração pela linha de comando –, apenas o Silk foi utilizado.

3.2.12 Any23

O *plug-in* que fazia a integração do Any23 ao ETL4LOD foi descontinuado antes de o ETL4LOD+ começar a ser desenvolvido e, por isso, não se encontra no ETL4LOD+.

4 EXEMPLO DE APLICAÇÃO

Para testar o uso do ETL4LOD+ na manutenção do ciclo de publicação de dados LOD e verificar o funcionamento dos *plug-ins* apresentados no capítulo anterior, dois conjuntos de dados – a "lista suja" do trabalho escravo no Brasil e o dossiê da ABRASCO sobre o uso de agrotóxicos no país – foram selecionados para serem triplificados. Ambos os conjuntos de dados foram selecionados por serem dados não triviais de triplificar e por já terem sido triplificados usando outras ferramentas, já que são alvos de outras pesquisas do grupo GRECO da UFRJ, permitindo criar um comparativo das triplas geradas pelo ETL4LOD+ com as geradas por outras ferramentas.

Na primeira parte deste capítulo, a triplificação da lista suja do Ministério do Trabalho é explorada, sendo seguida da triplificação da primeira parte do dossiê da ABRASCO. Por fim, comentários gerais sobre a experiência de triplificação no ETL4LOD+ são apresentados.

4.1 APLICAÇÃO DO ETL4LOD+ NA LISTA SUJA

Anualmente, o Ministério do Trabalho divulga uma "lista suja" do trabalho escravo, no qual denuncia empresas pela prática do crime no Brasil. O documento, representado em parte na Figura 31, contém um cadastro de empregadores que submeteram trabalhadores a condições análogas a do trabalho escravo – trabalho forçado, com condições degradantes, com jornada exaustiva e/ou com servidão por dívida (INPACTO, 2016).

Ministério do Trabalho

Cadastro de Empregadores que tenham submetido trabalhadores a condições análogas à de escravo
(Portaria Interministerial nº 4, de 11 de maio de 2016)

Atualização periódica de 6/10/2017. Cadastro atualizado em 21/11/2017.

I- PUBLICAÇÃO DA RELAÇÃO DE EMPREGADORES PREVISTA NO ARTIGO 2º, CAPUT, DA PORTARIA INTERMINISTERIAL Nº 4, DE 11 DE MAIO DE 2016

	Ano da ação fiscal	Empregador	CNPJ/CPF	Estabelecimento	Trabalhadores envolvidos	CNAE	Decisão administrativa de procedência - irrecorribilidade
1	2016	Adalberto Braz de Souza	884.400.954-49	Rod. BR 386, bairro Olarias/Conventos, Lajeado/RS	17	4789-0/99	13/04/17
2	2014	Ademir Andrade de Oliveira	705.704.936-68	Fazenda Santa Helena/Chácara Vargem Bonita - zona rural, Ibiraci/MG	11	0134-2/00	05/02/15
3	2015	AEV Empreendimentos Imobiliários SPE Ltda	20.288.137/0001-09	Obra Residencial American Garden I - Rua Lindolfo de Azevodo, 1.184, Jardim América, Belo Horizonte/MG	9	4110-7/00	07/07/16
4	2014	Agenor Tibúrcio da Silva	375.056.961-49	Fazenda Bagre - Região do Marimbondo, zona rural, Caldas Novas/MG	3	0899-1/99	05/05/15
5	2014	Airton Luiz Cobalchini	828.271.339-20	Viveiro de Mudas e Serraria - Rua Pedro Damo, 87, Formosa, Campo Erê/SC	1	0210-1/03	03/12/14
6	2016	Alex Teixeira de Oliveira Santos	949.176.121-87	Rua 47, Quadra 116, Lote 3, Jardim Tiradentes, Aparecida de Goiânia/GO	11	3212-4/00	27/02/17
7	2014	Alexandre Vieira Lins	360.426.924-53	Fazena Sara - Rod. BR 135, km 122, Miranda do Norte/MA	4	0151-2/01	10/12/14
8	2014	Amândio Celestino Cogo	120.299.399-00	Fazenda Perseverança - Ramal Jorge Kalume, km 16, Rio Branco/AC	3	0151-2/01	28/07/15
9	2016	Ancelmo Gomes Gonçalves	819.832.803-30	Área de extração de carnaúba - Povoado Areal, zona rural, Santa Cruz do Piauí/PI	18	0220-9/99	07/04/17

Figura 31 – Excerto da lista suja

No trabalho de Verona (2018), a lista suja foi combinada a dados governamentais para descobrir a influência política em agentes contemporâneos de escravidão, ou seja, conectar políticos a empresas que praticam escravidão moderna. Para isso, os dados fonte da lista suja disponibilizados pelo Ministério do Trabalho foram processados, triplificados e publicados como dados conectados. Por conter tanto os dados fonte antes da triplificação como o resultado após a triplificação, a lista suja já trabalhada foi o conjunto de dados perfeito para testar a capacidade do ETL4LOD+ de automatizar a triplificação de dados conectados.

Os dados fonte disponibilizados já se encontravam limpos, portanto não foi necessário fazer um trabalho de pré-processamento e foi possível ir direto para a triplificação. Três triplificações foram feitas para o trabalho original:

1. **Dados das Operações**: dados de cada uma das operações realizadas na lista suja;
2. **Dados dos Agentes**: dados sobre as pessoas ou empresas que aparecem na lista;
3. **Dados da Lista**: dados sobre onde as listas sujas de cada ano foram encontradas.

O subconjunto de dados escolhido para ser triplificado no ETL4LOD+ foi o das operações realizadas para descobrir a inadimplência das empresas. Este subconjunto foi escolhido por ser o mais complexo e por usar todos os dados da lista suja em sua triplificação.

CNPJ/CPF	lista	Ano da ação fiscal	Trabalhadores	id operacao	Empregador	Estabelecimento	CNAE	atividade	cod ativ
00.110.581/0001-14	201407	2009	14	1	Carlos Fernando	Construflora, zona rura	0230-6		
00.110.581/0001-14	201207	2009	14	1	Carlos Fernando	Construflora, Zona Rural	Chapadão	Corte de eucalipto	REFLO
00.110.581/0001-14	201112	2009	14	1	Carlos Fernando	-		Corte de eucalipto	REFLO
00.110.581/0001-14	201107	2009	14	1	Carlos Fernando	-		Corte de eucalipto	REFLO
00.110.581/0001-14	201010	2009	14	1	Carlos Fernando	-		Corte de eucalipto	REFLO
00.110.581/0001-14	201412	2009	14	1	Carlos Fernando	-		Corte de eucalipto	
00.138.268/0001-94	201509	2011	12	2	La-Fee Confecçõ	Oficina de costura - Rua Professor Cesare Lombroso, 211, Bom Retirc			
00.138.268/0001-94	201412	2011	12	2	La-Fee Confecçõ	Oficina de costura		Setor têxtil	
00.138.268/0001-94	201407	2011	12	2	La-Fee Confecçõ	Oficina de costura - Rua	1412-6/01		

Figura 32 – Excerto das operações contidas na lista suja

Cada linha da lista suja representa uma operação que foi realizada para descobrir empregadores vinculados ao trabalho escravo (Figura 32), linhas repetidas para empregadores indicam que um mesmo empregador foi descoberto com trabalhadores em condições similares a da escravidão múltiplas vezes.

Para triplificar esses dados, foi necessário conectar cada uma das operações (os sujeitos) a todas as outras informações na tabela (os objetos) usando o cabeçalho para dizer o que cada objeto representa (os predicados).

Usando a lista suja, dados conectados em formato RDF tinham sido gerados manualmente para cada uma das operações. A serialização escolhida para representar os dados conectados foi a RDF/XML (Figura 33), que não é a melhor serialização para trabalhar com o ETL4LOD+. Portanto, apesar de os dados originais terem sido serializados em RDF/XML, esta experimentação usa a serialização N-Triples para triplificar os dados das operações.

```
<rdf:Description rdf:about="http://lodbr.ufrj.br/Operacao1">
    <rdf:type rdf:resource="http://lodbr.ufrj.br/OntoSlavery#OperacaoMPT"/>
    <lodufrj:ano_acao_fiscal>2009</lodufrj:ano_acao_fiscal>
    <lodufrj:empregador_nome>Carlos Fernando Moura e Cia Ltda</lodufrj:empregador_nome>
    <lodufrj:uf>MS</lodufrj:uf>
    <lodufrj:uf_geonames rdf:resource="http://www.geonames.org/3457415/" />
    <lodufrj:empregador_cpf_cnpj>00.110.581/0001-14</lodufrj:empregador_cpf_cnpj>
    <lodufrj:estabelecimento> Construflora, zona rural, Chapadão do Sul/MS</lodufrj:estabelecimento>
    <lodufrj:trabalhadores_envolvidos>14</lodufrj:trabalhadores_envolvidos>
    <lodufrj:cnae>0230-6</lodufrj:cnae>
    <lodufrj:empregador rdf:resource="http://lodbr.ufrj.br/Empresa_00110581000114"/>
    <dbpedia:member rdf:resource="http://lodbr.ufrj.br/listasuja201407"/>
</rdf:Description>
<rdf:Description rdf:about="http://lodbr.ufrj.br/Operacao1">
    <dbpedia:member rdf:resource="http://lodbr.ufrj.br/listasuja201207"/>
    <lodufrj:atividade>Corte de eucalipto</lodufrj:atividade>
    <lodufrj:cod_atividade>REFLO</lodufrj:cod_atividade>
    <lodufrj:localizacao>inserido no nome da empresa</lodufrj:localizacao>
</rdf:Description>
<rdf:Description rdf:about="http://lodbr.ufrj.br/Operacao1">
    <dbpedia:member rdf:resource="http://lodbr.ufrj.br/listasuja201112"/>
    <lodufrj:atividade>Corte de eucalipto</lodufrj:atividade>
    <lodufrj:cod_atividade>REFLO</lodufrj:cod_atividade>
    <lodufrj:localizacao>Construflora, Zona Rural</lodufrj:localizacao>
</rdf:Description>
<rdf:Description rdf:about="http://lodbr.ufrj.br/Operacao1">
    <dbpedia:member rdf:resource="http://lodbr.ufrj.br/listasuja201107"/>
    <lodufrj:atividade>Corte de eucalipto</lodufrj:atividade>
    <lodufrj:cod_atividade>REFLO</lodufrj:cod_atividade>
    <lodufrj:localizacao>Construflora, Zona Rural</lodufrj:localizacao>
</rdf:Description>
<rdf:Description rdf:about="http://lodbr.ufrj.br/Operacao1">
    <dbpedia:member rdf:resource="http://lodbr.ufrj.br/listasuja201010"/>
    <lodufrj:atividade>Corte de eucalipto</lodufrj:atividade>
    <lodufrj:cod_atividade>REFLO</lodufrj:cod_atividade>
    <lodufrj:localizacao>Construflora, Zona Rural</lodufrj:localizacao>
</rdf:Description>
<rdf:Description rdf:about="http://lodbr.ufrj.br/Operacao1">
    <dbpedia:member rdf:resource="http://lodbr.ufrj.br/listasuja201412"/>
    <lodufrj:municipio>Chapadão do Sul</lodufrj:municipio>
    <lodufrj:atividade>Corte de eucalipto</lodufrj:atividade>
    <lodufrj:localizacao>Construflora, Zona Rural</lodufrj:localizacao>
</rdf:Description>
```

Figura 33 – Operação 1 gerada manualmente em RDF/XML

O primeiro passo dos testes do ETL4LOD+ foi o uso da ferramenta Any23 para converter os dados no formato RDF/XML para N-Triples, criando assim uma forma de verificar se as triplas geradas pelo ETL4LOD+ condiziam com as geradas manualmente (Figura 34).

```
<http://lodbr.ufrj.br/Operacao1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://lodbr.ufrj.br/OntoSlavery#OperacaoMPT> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/ano_acao_fiscal> "2009" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/empregador_nome> "Carlos Fernando Moura e Cia Ltda" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/uf> "MS" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/uf_geonames> <http://www.geonames.org/3457415/> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/empregador_cpf_cnpj> "00.110.581/0001-14" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/estabelecimento> " Construflora, zona rural, Chapad\u00E3o do Sul/MS" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/trabalhadores_envolvidos> "14" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/cnae> "0230-6" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/empregador> <http://lodbr.ufrj.br/Empresa_00110581000114> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201407> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201207> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201112> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201107> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201010> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201412> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/atividade> "Corte de eucalipto" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/cod_atividade> "REFLO" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/localizacao> "inserido no nome da empresa" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/localizacao> "Construflora, Zona Rural" .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/municipio> "Chapadão do Sul" .
```

Figura 34 – Operação 1 em formato N-Triples gerado pelo Any23

Após isso, o ETL4LOD+ foi utilizado para criar o fluxo necessário para triplificar os dados fonte para o formato N-Triples.

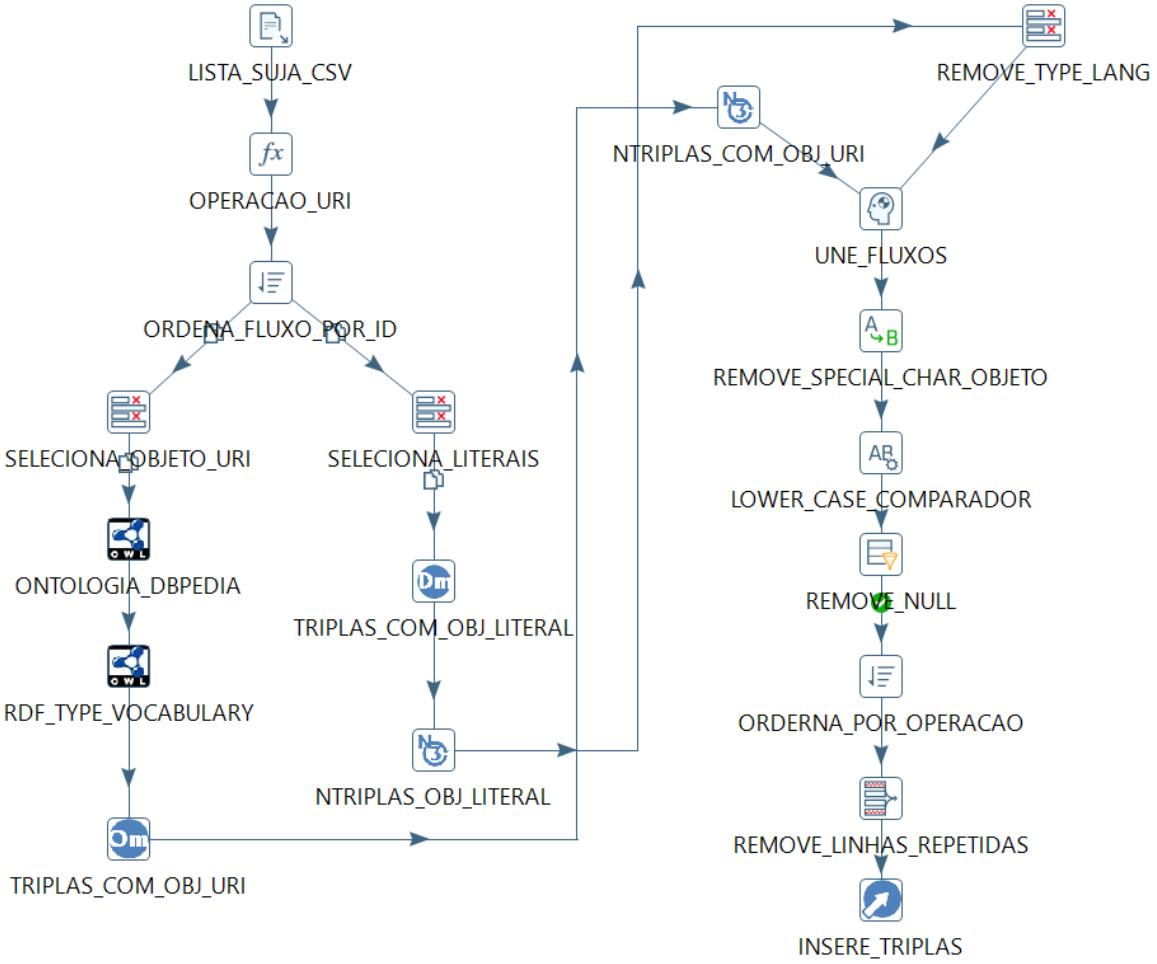


Figura 35 – Fluxo para triplificar as operações da lista suja

Para gerar os dados com fidelidade, eles precisaram ser separados em dois fluxos paralelos (Figura 35): um para lidar com os objetos literais e outro para lidar com os objetos

que fossem URIs. Para integrar os termos *member* da ontologia do DBpedia e *type* da sintaxe do RDF, o *plug-in* Owl Input foi utilizado com a configuração disposta na Figura 36.

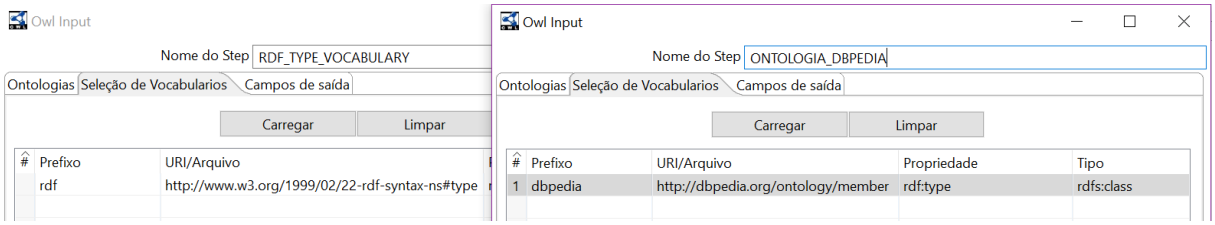


Figura 36 – Configuração do Owl Input

Uma vez que os termos da ontologia estivessem no fluxo, o Object Property Mapping foi usado para criar o mapeamento para todos os objetos que fossem URIs (Figura 37).

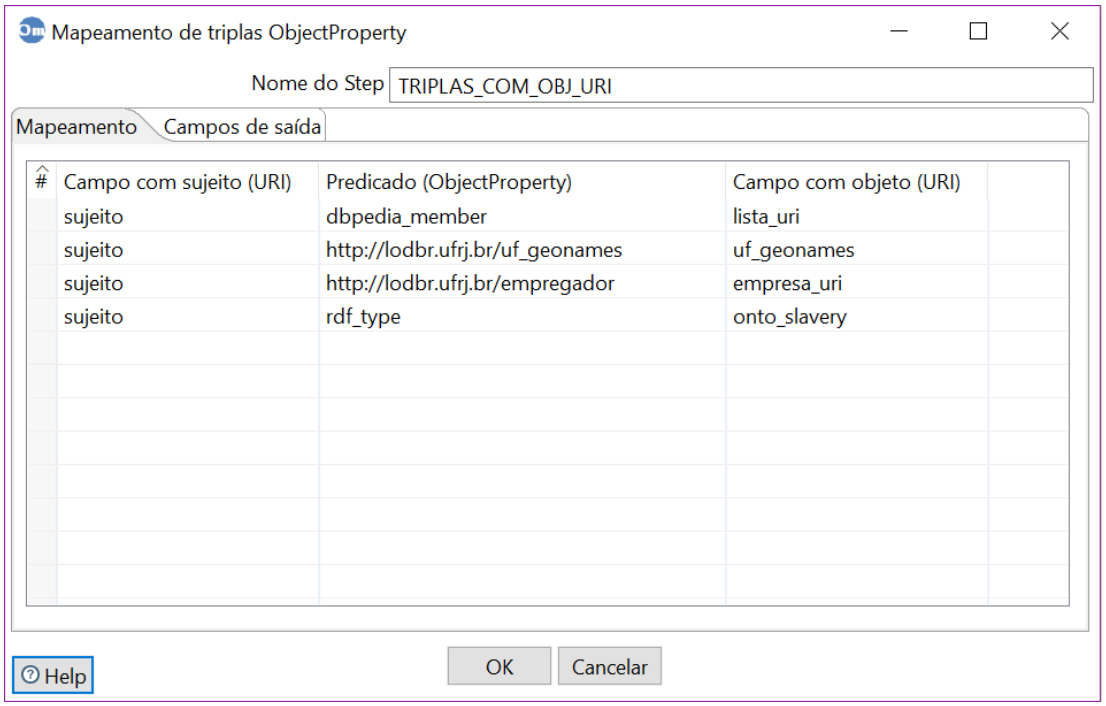


Figura 37 – Configuração do Object Property Mapping

O Data Property Mapping, em contrapartida, foi usado para o mapeamento de todos os objetos que fossem literais (Figura 38).

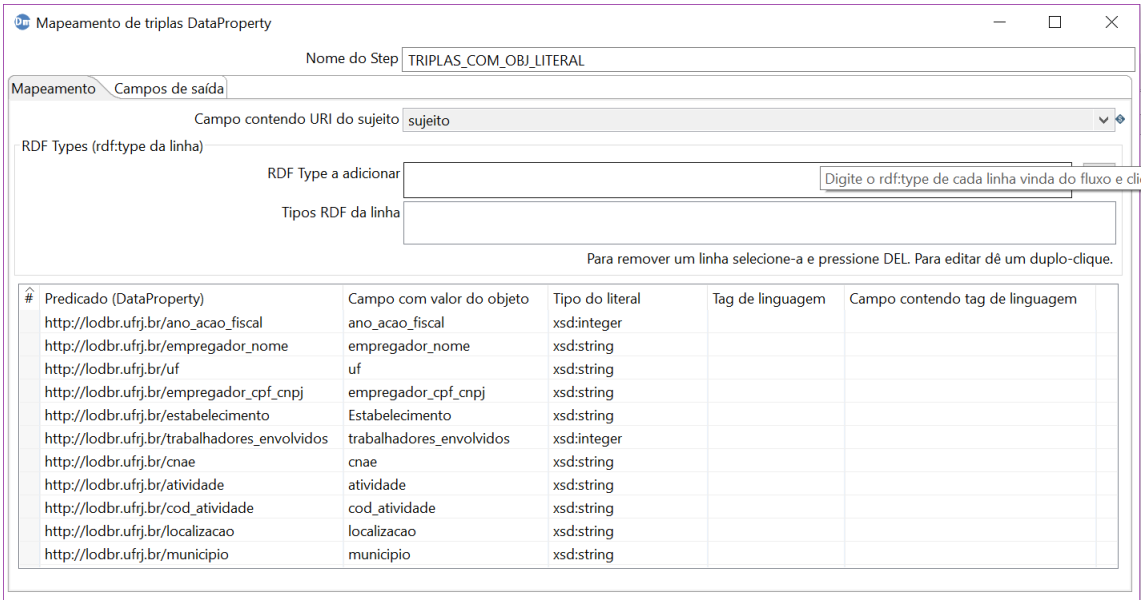


Figura 38 – Configuração do Data Property Mapping

Depois disso, o NTriple Generator foi usado para gerar sentenças no formato N-Triples para ambos os fluxos (Figura 39).

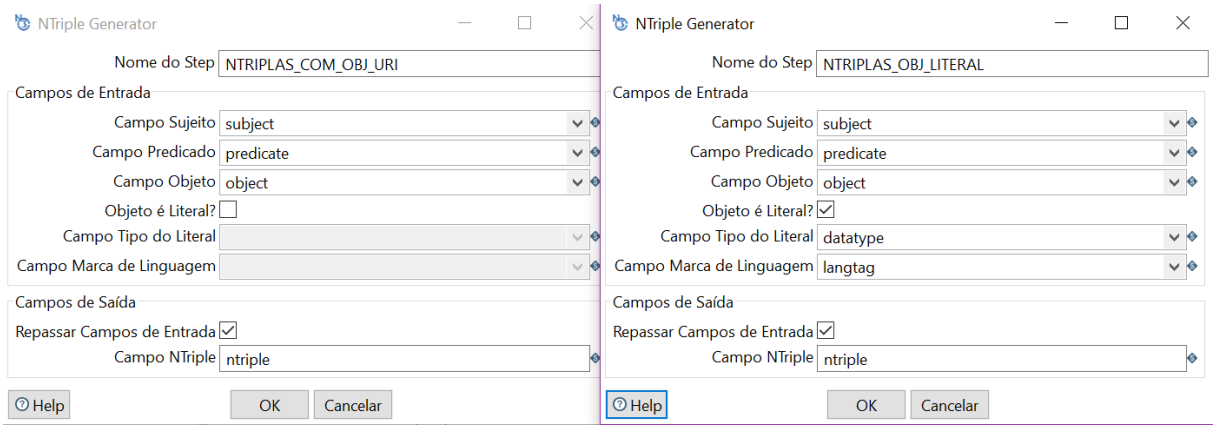


Figura 39 – Configuração do N-Triple Generator

Os fluxos gerados foram unidos num fluxo só e linhas repetidas foram removidas, gerando o conjunto de triplas apresentado na Figura 40.

```
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201010> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201107> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201112> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201207> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201407> .
<http://lodbr.ufrj.br/Operacao1> <http://dbpedia.org/ontology/member> <http://lodbr.ufrj.br/listasuja201412> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/ano_acao_fiscal> "2009"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/atividade> "Corte de eucalipto"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/cnae> "0230-6"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/cod_atividade> "REFLO"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/empregador> <http://lodbr.ufrj.br/Empresa_110581000114> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/empregador_cpf_cnpj> "00.110.581/0001-14"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/empregador_nome> "Carlos Fernando Moura e Cia Ltda"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/estabelecimento> "Construflora, zona rural, Chapadão do Sul/MS"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/localizacao> "Construflora, Zona Rural"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/localizacao> "inserido no nome da empresa"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/municipio> "Chapadão do Sul"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/trabalhadores_envolvidos> "14"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/uf> "MS"^^<http://www.w3.org/2001/XMLSchema#string> .
<http://lodbr.ufrj.br/Operacao1> <http://lodbr.ufrj.br/uf_geonames> <http://www.geonames.org/3457415/> .
<http://lodbr.ufrj.br/Operacao1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://lodbr.ufrj.br/OntoSlavery#OperacaoMPT> .
```

Figura 40 – Triplas geradas pelo ETL4LOD+

Esse conjunto de triplas foi então adicionado a um *endpoint* SPARQL local usando o Sparql Update Output (Figura 41).

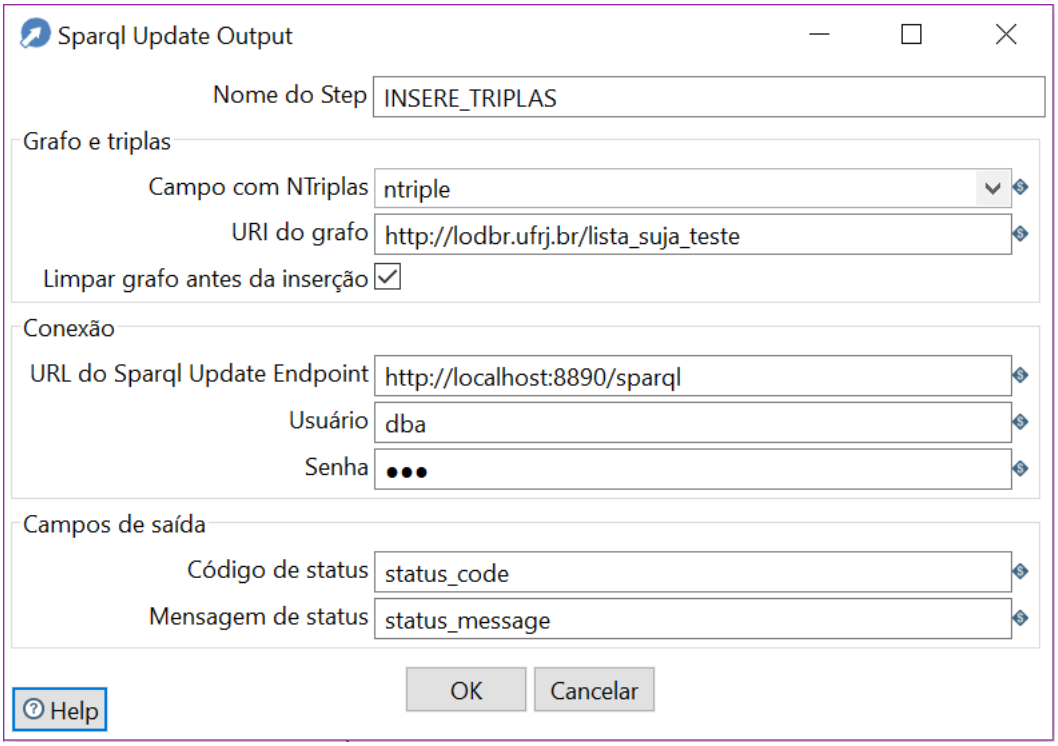


Figura 41 – Configuração do Sparql Update Output

Como pode ser observado na Figura 34, o conjunto de triplas geradas para as operações pelo ETL4LOD+ é quase idêntica ao conjunto convertido pelo Any23, a única

diferença está no fato de a versão gerada pelo ETL4LOD+ ter anotado os literais com o tipo do dado, agregando maior valor semântico às triplas.

4.2 APLICAÇÃO DO ETL4LOD+ NO DOSSIÊ DA ABRASCO

A Associação Brasileira de Saúde Coletiva (ABRASCO) elaborou um dossiê para registrar e difundir a preocupação de pesquisadores, professores e profissionais da área com o aumento no uso de agrotóxicos no país e a resultante contaminação das pessoas e meio ambiente (MEDEIROS; ALVES, 2018). A primeira parte desse dossiê relaciona os tipos de agrotóxicos usados com os sintomas agudos e crônicos que esse agrotóxico causa (Figura 42).

	A	B	C	D
1	Grupo Químico	Classe	Sintomas de intoxicação aguda	Sintomas de intoxicação crônica
2	Organofosforados e carbamatos	Inseticidas	Fraqueza, cólicas abdominais, vômitos, espasmos musculares e convulsões	Efeitos neurotóxicos retardados, alterações cromossomiais e dermatites de c
3	Organoclorados	Inseticidas	Náuseas, vômitos, contrações musculares involuntárias	Lesões hepáticas, arritmias cardíacas, lesões renais e neuropatias periféricas
4	Piretroides Sintéticos	Inseticidas	Irritações das conjuntivas, espirros, excitação, convulsões	Alergias, asma brônquica, irritações nas mucosas, hiper-sensibilidade
5	Ditiocarbamatos	Fungicidas	Tonteiras, vômitos, tremores musculares, dor de cabeça	Alergias respiratórias, dermatites, doença de Parkinson, cânceres
6	Fentalamidas	Fungicidas		Teratogêneses
7	Dinitroferóis e pentaciclorofenol	Herbicidas	Dificuldade respiratória, hipertermia, convulsões	Cânceres (PCP-formação de dioxinas), cloroacnes
8	Fenoxiacéticos	Herbicidas	Perda de apetite, enjoo, vômitos, fasciculação muscular	Indução da produção de enzimas hepáticas, cânceres, teratogeneses
9	Dipiridilos	Herbicidas	Sangramento nasal, fraqueza, desmaios, conjuntivites	Lesões hepáticas, dermatites de contato, fibrose pulmonar
10				
11	Fonte:OPAS/OMS – ORGANIZAÇÃO PANAMERICANA DA SAÚDE/ ORGANIZAÇÃO MUNDIAL DA SAÚDE. Representação do Brasil. Manual de vigilância da saúde de populações expostas a agrotóxicos. Brasí			

Figura 42 – Excerto da primeira parte do dossiê da ABRASCO

Em Medeiros e Alves (2018), a primeira parte do dossiê da ABRASCO foi triplificada e anotada com as ontologias MEDDRA[13] e CHEBI[14]. Por conter tanto os dados fonte como o resultado esperado da triplificação, este conjunto de dados foi usado para a segunda aplicação do ETL4LOD+.

4.2.1 Limpeza dos dados

Diferente da lista suja, os dados fontes da primeira parte do dossiê da ABRASCO não se encontravam limpos e em formato adequado para triplificação. Por esta razão, a primeira etapa da triplificação dos dados envolveu a limpeza dos dados fonte usando os *plug-ins* já existentes no Kettle.

Os dados fonte possuíam 3 problemas principais que foram tratados durante a limpeza. A resolução desses problemas foi implementada no fluxo do Kettle apresentado na Figura 43.

[13] https://bioportal.biontology.org/ontologies/MEDDRA

[14] http://purl.obolibrary.org/obo/CHEBI

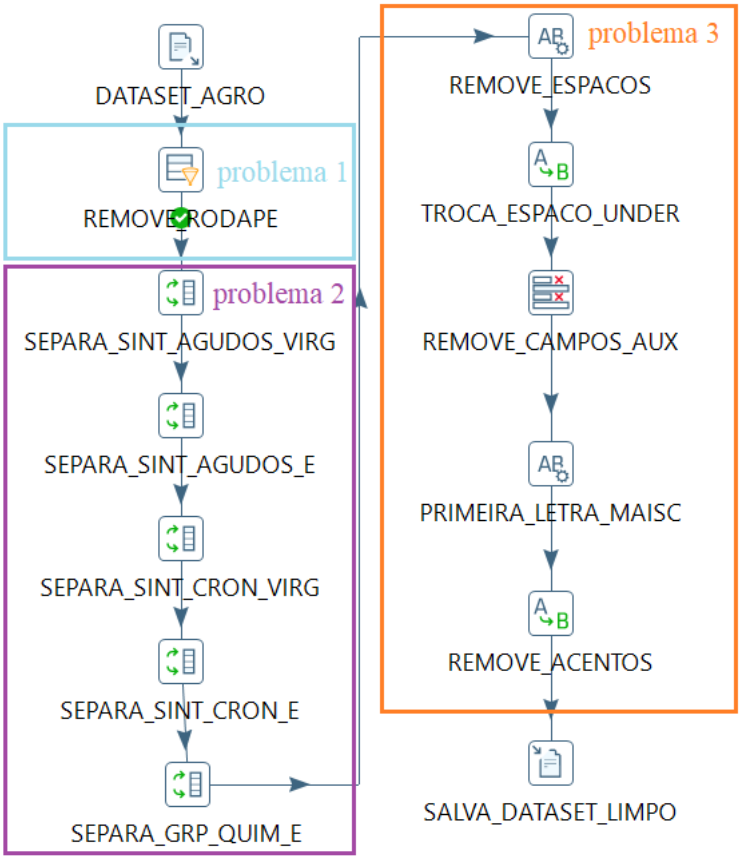


Figura 43 – Fluxo de limpeza dos dados da ABRASCO

O primeiro problema tratado foi a remoção do rodapé na linha 11 (Figura 44) do conjunto de dados usando o *plug-in* Filter Rows do Kettle.

10	
11	Fonte:OPAS/OMS – ORGANIZAÇÃO PANAMERICANA DA SAÚDE/ ORGANIZAÇÃO MUNDIAL DA SAÚDE. Representação do Brasil. Manual de vigilância da saúde de populações expostas a agrot

Figura 44 – Problema 1: rodapé com fonte dos dados

O segundo problema tratado foi quebrar em várias linhas os valores que estavam agrupados numa célula só separados por vírgula ou *e* (Figura 45). O *plug-in* Split Field To Rows do Kettle foi usado para atingir esse objetivo (Figura 46).

Grupo Químico	Classe	Sintomas de intoxicação aguda
Organofosforados e carbamatos	Inseticidas	Fraqueza, cólicas abdominais, vômitos, espasmos musculares e convulsões

Figura 45 – Problema 2: múltiplos valores numa célula só

Grupo Químico	Classe	Sintomas de intoxicação aguda
Organofosforados	Inseticidas	Fraqueza
Organofosforados	Inseticidas	cólicas abdominais
Organofosforados	Inseticidas	vômitos
Organofosforados	Inseticidas	espasmos musculares
Organofosforados	Inseticidas	convulsões
carbamatos	Inseticidas	Fraqueza
carbamatos	Inseticidas	cólicas abdominais
carbamatos	Inseticidas	vômitos
carbamatos	Inseticidas	espasmos musculares
carbamatos	Inseticidas	convulsões

Figura 46 – Problema 2 após a limpeza

O último problema resolvido foi a falta de padronização dos dados (Figura 47) para criação de sufixos para URIs, que envolveu a remoção de acentos, substituição de espaços por '_' e capitalização da primeira letra de cada célula. Foi necessário uma combinação dos *plug-ins* String Operations e Replace in String do Kettle para resolver este problema.

Sintomas de intoxicação aguda	Sintomas_Agudos_URI
Fraqueza	Fraqueza
cólicas abdominais	Colicas_abdominais
vômitos	Vomitos
espasmos musculares	Espasmos_musculares
convulsões	Convulsoes

Figura 47 – Problema 3: dados fora do padrão

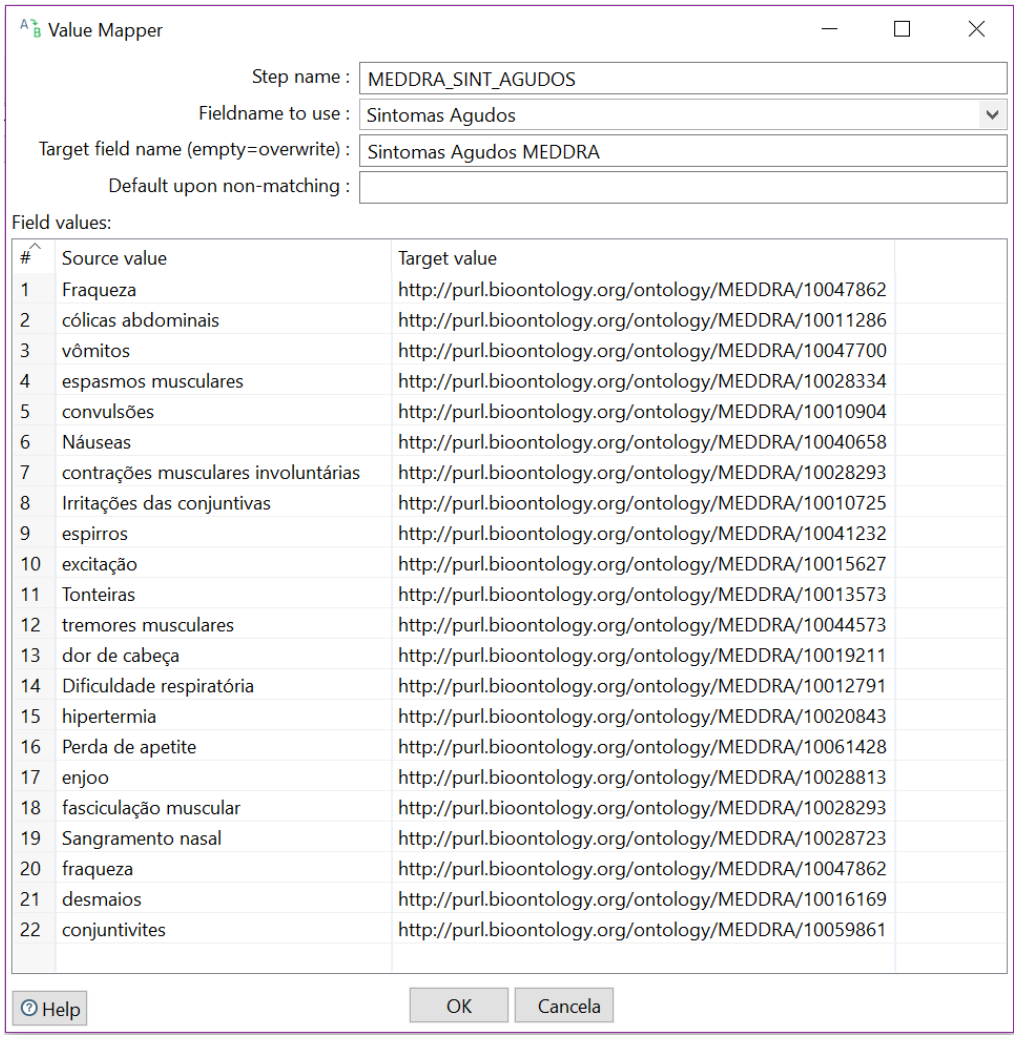
Por fim, os dados foram novamente salvos em CSV para serem triplificados após todo este trabalho de pré-processamento.

4.2.2 Triplificação dos dados limpos

Uma vez que os dados estejam limpos, o processo de triplificação pode ser iniciado. A primeira parte do processo de triplificação foi converter os campos padronizados para URIs com *http://lodbr.ufrj.br/* como prefixo.

Em seguida, os dados foram anotados com as ontologias MEDDRA e CHEBI. A ontologia MEDDRA foi utilizada para anotar os sintomas causados pelos agrotóxicos enquanto que a CHEBI foi usada para anotar os agrotóxicos em si. Como o ETL4LOD+ não possui integração com as APIs de nenhuma das duas ontologias, o processo de anotação

envolveu uma busca manual pelos termos necessários de cada ontologia. Outrossim, tanto os sintomas como os agrotóxicos estão em português no conjunto de dados. Para que a busca pudesse ser realizada automaticamente eles precisariam ser traduzidos para o termo em inglês usado para identificá-lo na ontologia. Como o trabalho de tradução seria tão custoso quanto o trabalho de busca manual, a busca manual dos termos foi feita.



#	Source value	Target value
1	Fraqueza	http://purl.bioontology.org/ontology/MEDDRA/10047862
2	cólicas abdominais	http://purl.bioontology.org/ontology/MEDDRA/10011286
3	vômitos	http://purl.bioontology.org/ontology/MEDDRA/10047700
4	espasmos musculares	http://purl.bioontology.org/ontology/MEDDRA/10028334
5	convulsões	http://purl.bioontology.org/ontology/MEDDRA/10010904
6	Náuseas	http://purl.bioontology.org/ontology/MEDDRA/10040658
7	contrações musculares involuntárias	http://purl.bioontology.org/ontology/MEDDRA/10028293
8	Irritações das conjuntivas	http://purl.bioontology.org/ontology/MEDDRA/10010725
9	espirros	http://purl.bioontology.org/ontology/MEDDRA/10041232
10	excitação	http://purl.bioontology.org/ontology/MEDDRA/10015627
11	Tonteiras	http://purl.bioontology.org/ontology/MEDDRA/10013573
12	tremores musculares	http://purl.bioontology.org/ontology/MEDDRA/10044573
13	dor de cabeça	http://purl.bioontology.org/ontology/MEDDRA/10019211
14	Dificuldade respiratória	http://purl.bioontology.org/ontology/MEDDRA/10012791
15	hipertermia	http://purl.bioontology.org/ontology/MEDDRA/10020843
16	Perda de apetite	http://purl.bioontology.org/ontology/MEDDRA/10061428
17	enjoo	http://purl.bioontology.org/ontology/MEDDRA/10028813
18	fasciculação muscular	http://purl.bioontology.org/ontology/MEDDRA/10028293
19	Sangramento nasal	http://purl.bioontology.org/ontology/MEDDRA/10028723
20	fraqueza	http://purl.bioontology.org/ontology/MEDDRA/10047862
21	desmaios	http://purl.bioontology.org/ontology/MEDDRA/10016169
22	conjuntivites	http://purl.bioontology.org/ontology/MEDDRA/10059861

Figura 48 – Anotação dos sintomas agudos com a ontologia MEDDRA

Para realizar a anotação, cada uma das colunas do conjunto de dados precisou de um *plug-in* Value Mapper como o da Figura 48 para associar os termos da ontologia aos seus respectivos valores nos dados.

Após isso, os sujeitos, predicados e objetos foram mapeados (Figura 49). Foram geradas triplas indicando qual a classe de cada agrotóxico, quais os sintomas causados por ele, e quais tipos de agrotóxicos são pesticidas. Os sintomas também foram anotados com o termo *Symptom* da DBPedia.

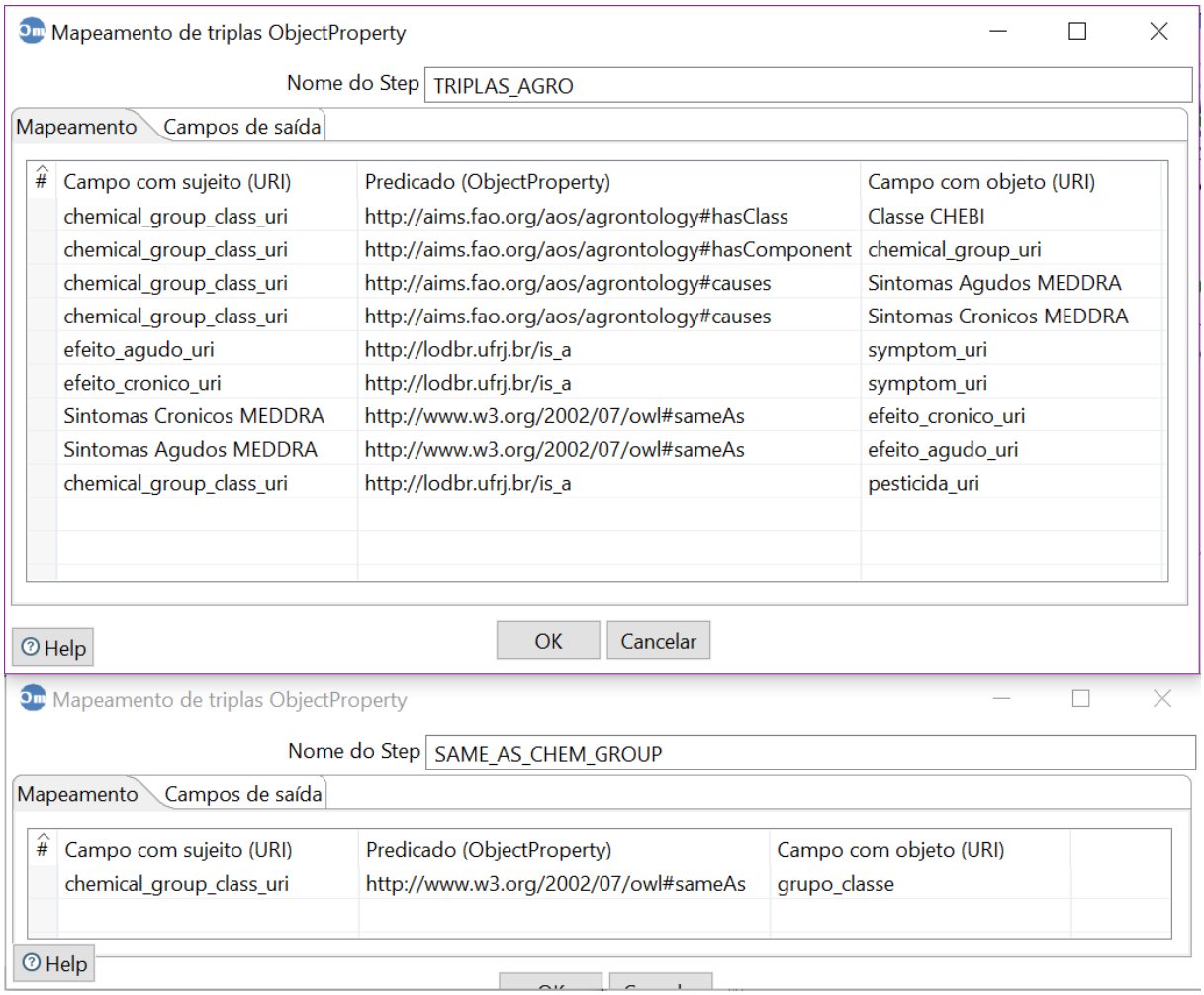


Figura 49 – Mapeamento de triplas dos dados da ABRASCO

Por fim, as URIs das classes e grupos químicos foram descritos por rótulos usando *rdfs:label* como predicado (Figura 50). Como os rótulos são valores literais, o Data Property Mapping precisou ser usado.

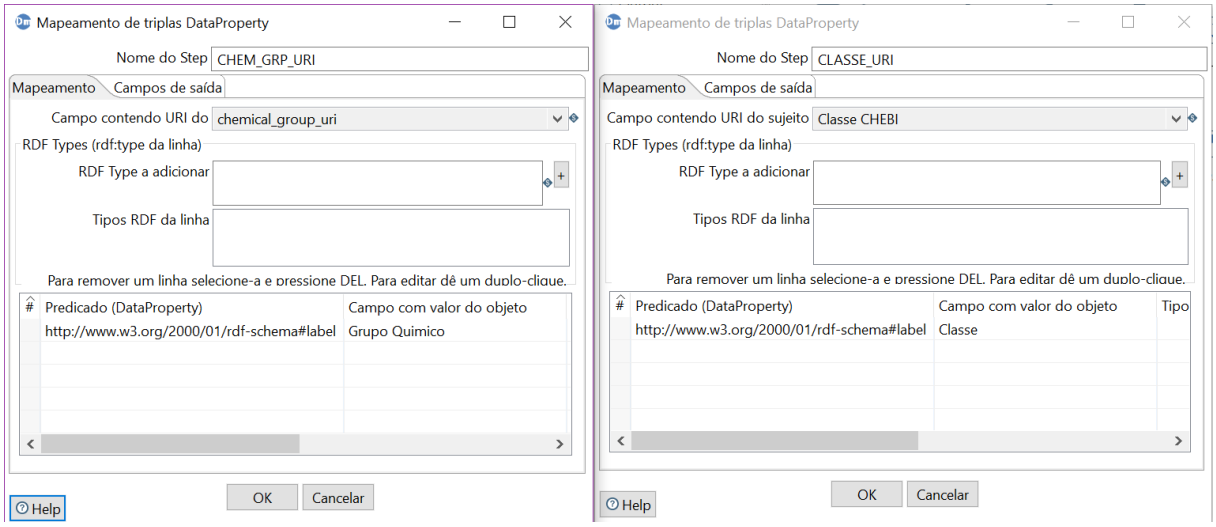


Figura 50 – Descrição de URIs usando labels

Em suma, para triplificar os dados da ABRASCO, foi criado o fluxo representado na Figura 51 no Kettle.

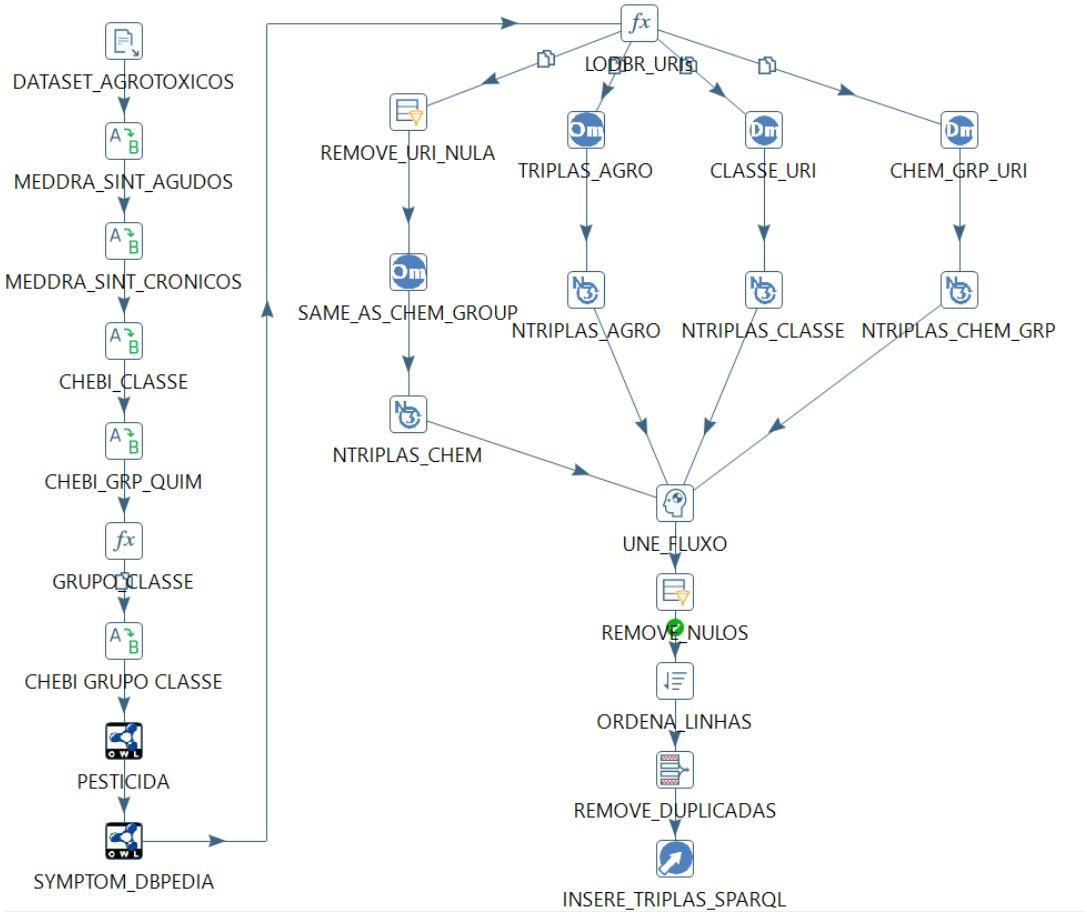


Figura 51 – Fluxo para triplificação dos dados da ABRASCO

O conjunto de triplas geradas para as operações pelo ETL4LOD+ é quase idêntico ao conjunto no qual essa aplicação foi baseada com algumas diferenças propositais, apresentadas na Figura 52 e Figura 53:

1. Os nomes dos agrotóxicos não foram traduzidos para o inglês na aplicação do ETL4LOD+.
2. O predicado *a* foi renomeado para *http://lodbr.ufrj.br/is_a*.
3. Por fim, os dados do ETL4LOD+ possuem uma etapa extra de ligação que associa alguns dos agrotóxicos locais à ontologia CHEBI usando o predicado *owl:sameAs*.

```
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#causes>
<http://purl.bioontology.org/ontology/MEDDRA/10020843>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#causes>
<http://purl.bioontology.org/ontology/MEDDRA/10010904>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#causes>
<http://purl.bioontology.org/ontology/MEDDRA/10012791>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#causes>
<http://purl.bioontology.org/ontology/MEDDRA/10008571>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#causes>
<http://purl.bioontology.org/ontology/MEDDRA/10007050>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#hasClass>
<http://purl.obolibrary.org/obo/CHEBI_24527>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> <http://aims.fao.org/aos/agrontology#hasComponent>
<http://lodbr.ufrj.br/chemical_group/Pentacyclorophenol>.
<http://lodbr.ufrj.br/chemical_group_class/Pentacyclorophenol_Herbicides> a <http://purl.obolibrary.org/obo/CHEBI_25944>.
```

Figura 52 – Triplas do agrotóxico Pentaciclorofenol

```
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#causes> <http://purl.bioontology.org/ontology/MEDDRA/10007050> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#causes> <http://purl.bioontology.org/ontology/MEDDRA/10008571> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#causes> <http://purl.bioontology.org/ontology/MEDDRA/10010904> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#causes> <http://purl.bioontology.org/ontology/MEDDRA/10012791> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#causes> <http://purl.bioontology.org/ontology/MEDDRA/10020843> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#hasClass> <http://purl.obolibrary.org/obo/CHEBI_24527> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://aims.fao.org/aos/agrontology#hasComponent> <http://lodbr.ufrj.br/chemical_group/Pentaciclorofenol> .
<http://lodbr.ufrj.br/chemical_group_class/Pentaciclorofenol_Herbicidas> <http://lodbr.ufrj.br/is_a> <http://purl.obolibrary.org/obo/CHEBI_25944> .
```

Figura 53 – Triplas do agrotóxico Pentaciclorofenol geradas pelo ETL4LOD+

4.3 CONSIDERAÇÕES FINAIS

As experimentações realizadas no ETL4LOD+ mostram que ela é uma ferramenta adequada para triplificar dados não triviais de forma simples, além de conter uma gama de funções para limpeza dos dados fonte já disponíveis no próprio Kettle. Essas funções, geralmente usadas para limpeza, se provaram úteis também na própria triplificação de dados, o que torna o ecossistema do Kettle bem rico no quesito funcionalidade.

As experimentações também mostraram que pela forma como o Kettle combina as suas colunas, pode acontecer uma quantidade grande de repetição de triplas, que podem ser facilmente removidas com o *plug-in* Unique Rows do Kettle.

O maior empecilho encontrado durante a triplificação dos exemplos acima foi o de anotar o conjunto de dados com termos de uma ontologia. Cada ontologia possui sua própria API REST de acesso e não existe um *plug-in* que se comunique com a API das ontologias utilizadas. Mesmo com a existência de tal *plug-in*, o ETL4LOD+ geralmente é usado com dados em português, portanto ainda existiria o trabalho manual de traduzir os campos para a linguagem usada na ontologia.

5 CONCLUSÃO

O principal propósito deste trabalho foi o de disponibilizar uma versão atualizada do ETL4LOD como plataforma de publicação de dados do tipo LOD e estender as suas funcionalidades para incluir a busca e seleção de termos de ontologias e a etapa de ligação de dados conectados.

A maior parte do trabalho foi focada em centralizar e atualizar todos os *plug-ins* já existentes no ETL4LOD que tinham sido feitos por diferentes pessoas e disponibilizados em diferentes lugares em uma única plataforma que ficou conhecida como ETL4LOD+.

Por fim, as duas novas funcionalidades agregadas ao ETL4LOD, a busca e seleção de termos de vocabulários e ontologias e a etapa de ligação, devem se provar úteis pois visam dispensar o uso de ferramentas externas ao ETL4LOD+ para a criação de dados conectados.

Em resumo, as modificações feitas nos *plug-ins* para este trabalho podem ser separadas em três categorias, como ilustrado na Figura 54: **atualizados** – *plug-ins* que foram atualizados para a versão 8.1 do Kettle, porém sua funcionalidade e código permaneceram inalteradas, **modificados** – *plug-ins* que foram atualizados e sofreram mudanças grandes no código ou adição de novas funcionalidades, e **criados** – *plug-ins* desenvolvidos especificamente neste trabalho.

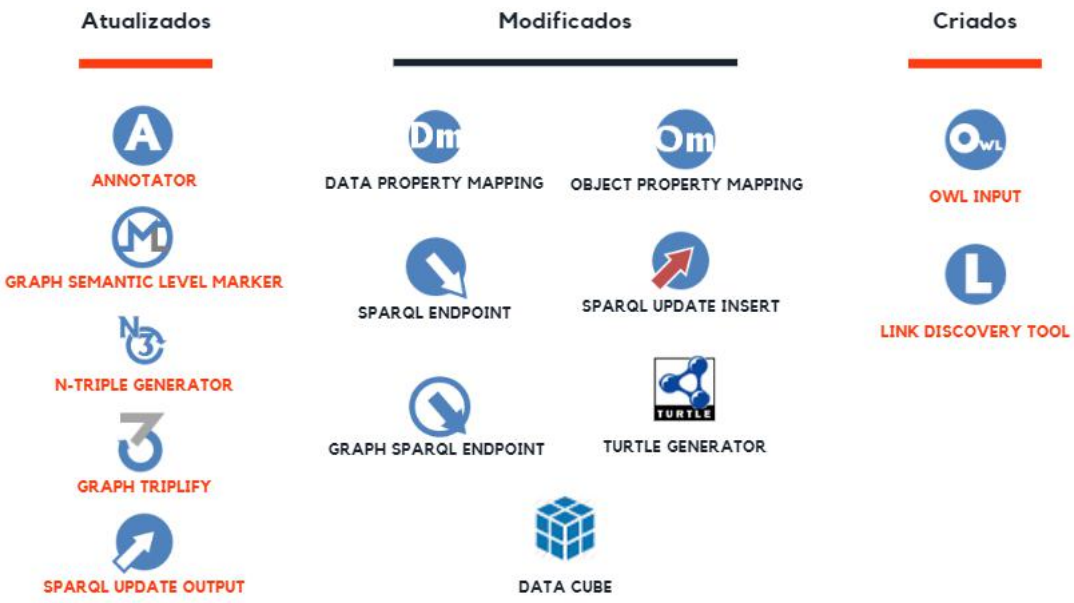


Figura 54 – Resumo das modificações feitas nos *plug-ins*

5.1 DIFICULDADES ENCONTRADAS

5.1.1 Documentação

A primeira dificuldade encontrada durante a criação do ETL4LOD+ foi a criação da documentação. Muitos dos *plug-ins* já existentes não possuíam exemplos de uso disponíveis ou descrições detalhadas de como funcionavam. Até mesmo os documentos da RNP que continham os detalhes do projeto quando foi idealizado não possuíam informações suficientes sobre os *plug-ins*. Foi preciso deduzir através de experimentação e leitura do código exatamente qual era a intenção por trás deles. Em alguns casos foi necessário pedir ao criador original do *plug-in* para explicar sua funcionalidade básica.

5.1.2 Testes

A segunda dificuldade encontrada foi a de testar os *plug-ins* atualizados. Exemplos de uso do ETL4LOD foram difíceis de conseguir e também difíceis de reproduzir no início do projeto. Testes pontuais foram realizados para todos os *plug-ins*, porém apenas dois conjuntos de dados foram triplificados do início ao fim – o que representa um volume de testes menor do que o desejado.

5.2 TRABALHOS FUTUROS

O ETL4LOD+ é uma boa opção para a criação de dados conectados, no entanto não pode ser considerada uma plataforma completa. Alguns *plug-ins* possuem limitações bem conhecidas que podem ser resolvidas em trabalhos futuros.

5.2.1 Link Discovery Tool

Tanto o Silk quanto o LIMES possuem a funcionalidade de criação de um grafo para indicar como e quais métricas e agregações estão sendo usadas para gerar aquelas ligações. Idealmente esta etapa poderia ser integralmente feita dentro do Kettle como uma configuração do Link Discovery Tool. Existem bibliotecas Java como o *jgraphx* e *graphstream* que

permitem a criação de grafos pelo usuário e poderiam ser adicionadas ao Link Discovery Tool.

Além disso, o LIMES poderia ser integrado ao Link Discovery Tool para que ele pudesse executar tanto o Silk quanto o LIMES.

5.2.2 Owl Input

O *plug-in* de gerenciamento de ontologias atualmente só aceita ontologias que estejam no LOV ou em arquivos. O *plug-in* também não faz a busca automática pelos valores encontrados em um campo do fluxo. O Owl Input poderia ser melhorado para aceitar novos tipos de ontologias disponíveis em diversos serviços do tipo REST e em mais formatos de arquivos, e para buscar URIs que representem os valores literais no fluxo.

5.2.3 Data Cube

O *plug-in* que trabalha com o vocabulário Data Cube é estruturado de uma forma muito rígida. Ele, por exemplo, não faz uso de novos prefixos adicionados às suas configurações. Diversas partes da sua saída como comentários, nome dos prefixos, e rótulos são fixas. Um trabalho de melhorar a forma com a qual o *plug-in* cria a sua saída final com Data Cube poderia ser feito.

5.2.4 Plug-ins de grafo

Os *plug-ins* Graph Sparql Enpoint e Sparql Enpoint assim como o Graph Triplify e N-Triple Generator são irmãos no quesito funcionalidade, porém um trabalha com grafos e o outro com triplas RDF. A funcionalidade desses *plug-ins* poderiam ser combinadas em um único *plug-in* que identificaria se a entrada é um grafo ou não.

5.2.5 Testes do ETL4LOD+

Os testes realizados no ETL4LOD+ foram infelizmente menos extensivos do que o desejado. Apenas dois conjuntos de dados foram triplificados do início ao fim e esses testes

não usavam todos os *plug-ins* disponíveis. Isso indica que possíveis *bugs* principalmente nas novas funcionalidades podem ter ficado escondidos.

Os testes também foram todos realizados por mim, que conhece o ETL4LOD+ o suficiente para não esbarrar em eventuais problemas com a ferramenta.

A realização de novos testes de usabilidade e funcionalidade da ferramenta são necessários e já estão planejados de serem feitos por participantes do grupo GRECO. O foco principal serão os dados do Portal de Dados contra o Uso de Agrotóxicos, projeto do grupo, que contém várias fontes de dados neste tema que precisam ser disponibilizadas também como dados conectados.

5.2.6 Manutenção

Como qualquer software, o ETL4LOD+ precisará de atualizações novamente, de correção de *bugs*, modificação de funcionalidades e atualizações na documentação. Para facilitar esse tipo de modificação, o projeto foi liberado como *open source* no GitHub e um manual de contribuição para o projeto foi criado[15], permitindo que qualquer um que possua interesse na ferramenta possa trabalhar sobre ela.

[15] https://github.com/johncurcio/ETL4LODPlus/

REFERÊNCIAS

ALIYU, S.; JUNAIDU, S. B.; KANA, A. F. A Category Theoretic Model of RDF Ontology. **International Journal of Web & Semantic Technology**, Dubai, v. 6, n. 3, jul. 2015.

BERNERS-LEE, T., HENDLER, J.; LASSILA, O. The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. **Scientific American**, Estados Unidos, v. 284, n. 5, 2001. Disponível em: https://www.scientificamerican.com/magazine/sa/2001/05-01/#article-the-semantic-web. Acesso em: 01 out. 2018.

BERNERS-LEE, Tim. **Linked Data**. Disponível em: https://www.w3.org/DesignIssues/LinkedData.html. Acesso em: 01 out. 2018.

BIZER, C., HEATH, T., BERNERS-LEE, T. *Linked Data: The Story so Far*. Semantic Services, Interoperability and Web Applications: Emerging Concepts. **IGI Global**, Hershey, PA., p. 205-227, 2011. Disponível em: https://www.igi-global.com/chapter/linked-data-story-far/55046. Acesso em: 10 de nov. 2018.

CORDEIRO, et al. An approach for managing and semantically enriching the publication of Linked Open Governmental Data. In: Workshop de Computação Aplicada em Governo Eletrônico (WCGE), 2011, Florianópolis. Workshop de Computação Aplicada em Governo Eletrônico (WCGE), 2011. v. 1.

CORDEIRO, Kelli. **ADApTA**: Adaptive approach for information integration to support decision making in complex environments. 2015. 148 f. Tese (Doutorado em Informática)-Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2015.

DANTAS, Adriano. **Estudo comparativo entre ferramentas de triplificação para dados interligados**. 2018. 65 f. (Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)-Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2018.

EQUIPE GT. **LinkedDataBR**: Exposição, Compartilhamento e Conexão de Recursos de Dados Abertos na Web (Linked Open Data). 2011. Documentação - Ciência da Computação, UFRJ, Rio de Janeiro, 2011. Disponível em: <https://memoria.rnp.br/pd/gts2010-2011/gt_linkeddatabr.html>. Acesso em: 12 nov. 2018.

_____. **Portal de Dados Abertos sobre Agrotóxicos**. Disponível em: http://dados.contraosagrotoxicos.org/. Acesso em: 26 de novembro de 2018.

ISMAIL, S.; Shaikh, T. A Literature Review on Semantic Web - Understanding the Pioneers' Perspective. In: Sixth International Conference on Computer Science, Engineering & Applications (ICCSEA), 11., 2016, Dubai. **Anais…** Dubai: AIRCC Publishing Corporation, 2016. v. 6 p. 15-28.

INPACTO. **Entenda a "Lista Suja".** 2016. Disponível em: http://www.inpacto.org.br/en/trabalho-escravo/lista-suja/. Acesso em: 11 nov. 2018.

KEJRIWAL, M., MIRANKER, D. On Linking Heterogeneous Dataset Collections. In: ISWC-PD'14 Proceedings of the 2014 International Conference on Posters & Demonstrations Track, 2014, Riva del Garda, Italy. **Proceedings...** ISWC-PD'14, 2014. v. 1272. p. 217-220.

KHATCHADOURIAN, S., CONSENS, M. ExpLOD: Summary-Based Exploration of Interlinking and RDF Usage in the Linked Open Data Cloud. In: ESWC'10 Proceedings of the 7th international conference on The Semantic Web: research and Applications, 2010, Heraklion, Grécia. **Proceedings...** ESWC Conference, 2010. v. II. p. 272-287.

KNOBLOCK, et al. Semi-Automatically Mapping Structured Sources into the Semantic Web. In: ESWC'12 Proceedings of the 9th international conference on The Semantic Web: research and Applications, 2012, Heraklion, Grécia. **Proceedings...** ESWC Conference, 2012. v. II. p. 272-287.

MEDEIROS, Amanda; ALVES, Julia Anne. **Triplificação de Dados Conectados**: uma experimentação de ferramentas no contexto do portal da campanha contra o uso de agrotóxicos. 2018. 108 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)-Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2018.

ONG, et al. Dynamic-ETL: a hybrid approach for health data extraction, transformation and loading. **BMC Med. Inf. & Decision Making**, Estados Unidos, v. 17, n. 1, 2017. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5598056/. Acesso em: 12 out. 2018.

Özsu, M. A survey of RDF data management systems. **Frontiers of Computer Science: Selected Publications from Chinese Universities**, Nova Iorque, v. 10, n. 3, jun. 2016. Disponível em: https://arxiv.org/abs/1601.00707. Acesso em: 13 out. 2018.

SUKHOBOK, et al. Tabular Data Cleaning and Linked Data Generation with Grafterizer. In: 13th ESWC Conference, 2016, Heraklion, Crete. **Proceedings**... ESWC Conference, 2016. v. 1.

TOMASZUK, Dominik. Flat triples approach to RDF graphs in JSON. Disponível em: https://www.w3.org/2009/12/rdf-ws/papers/ws02. Acesso em: 12 out. 2018.

VERONA, L., LOPES, G., CAMPOS MACHADO, M. L. Using government data to uncover political power and influence of contemporary slavery agents in Brazil. 2018. In: BIDU 2018: Workshop on Big Social Data and Urban Computing, Rio de Janeiro, Brasil. **Forthcoming Proceedings…**, BIDU 2018, v. 926

VERONA, L., OLIVEIRA, J., DA CUNHA HISSE, J., MACHADO CAMPOS, M. L. Metrics for network power based on Castells? Network Theory of Power: a case study on Brazilian elections. JOURNAL OF INTERNET SERVICES AND APPLICATIONS, v. 9, p. 1-16, 2018.

VERONA, Letícia; MACHADO CAMPOS, Maria Luiza. De dados governamentais para dados abertos interligados: revelando as correntes do trabalho escravo contemporâneo no Brasil.. **XI Reunião Científica Trabalho Escravo Contemporâneo e questões correlatas**, Minas Gerais, out. 2018.

VOLZ, et al. Silk – A Link Discovery Framework for the Web of Data. In: Proceedings of the 2nd Workshop on Linked Data on the Web, 2009, Madrid, Spain. **Proceedings...** LDOW 2009, 2009. v. 538.

WOOD, et al. **Linked Data: Structured data on the Web**. 1 ed. Nova Iorque: Manning Publications, 2014.

W3C. RDF 1.1 Concepts and Abstract Syntax. Disponível em: https://www.w3.org/TR/rdf11-concepts/. Acesso em: 13 out. 2018.

_____. RDF 1.1 Semantics. Disponível em: https://www.w3.org/TR/rdf11-mt/. Acesso em: 13 out. 2018.

APÊNDICE A – ARQUIVO CSV DE ENTRADA PARA O *PLUG-IN* TURTLE GENERATOR

```
Cultura,Ano,tipoAgrotoxico
Algodão,2000,produto comercial
Alho,2000,produto comercial
Amendoim,2000,produto comercial
Arroz Irrigado,2000,produto comercial
Arroz Sequeiro,2000,produto comercial
Banana,2000,produto commercial
```

APÊNDICE B – ARQUIVO RDF/TURTLE GERADO PELO *PLUG-IN* TURTLE GENERATOR

```
@base <http://meu.exemplo/>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix dc: <http://purl.org/dc/elements/1.1/>.
@prefix skos: <http://www.w3.org/2004/02/skos/core#>.
@prefix obo: <http://purl.obolibrary.org/obo#>.
@prefix ex: <http://meu.exemplo/>.
@prefix exProp: <http://meu.exemplo/properties/>.

#
# PROPERTIES DEFINITION
#

exProp:ano owl:sameAs <http://www.w3.org/2002/07/owl#Ano>;
       rdfs:label "Ano"@en .

exProp:cultura owl:sameAs <http://www.w3.org/2002/07/owl#Cultura>;
       rdfs:label "Cultura"@en .

exProp:tipoagrotoxico owl:sameAs <http://www.w3.org/2002/07/owl#Agrotoxico>;
       rdfs:label "Agrotóxico"@en .

#
# HIERARCHIES
#

ex:obj0 exProp:ano "2000";
       exProp:cultura "Algodão";
       exProp:tipoagrotoxico "produto comercial";
       rdfs:label "Agrotóxicos e suas culturas" .

ex:obj1 exProp:ano "2000";
       exProp:cultura "Alho";
       exProp:tipoagrotoxico "produto comercial";
       rdfs:label "Agrotóxicos e suas culturas" .

ex:obj2 exProp:ano "2000";
       exProp:cultura "Amendoim";
       exProp:tipoagrotoxico "produto comercial";
       rdfs:label "Agrotóxicos e suas culturas" .

ex:obj3 exProp:ano "2000";
       exProp:cultura "Arroz Irrigado";
       exProp:tipoagrotoxico "produto comercial";
       rdfs:label "Agrotóxicos e suas culturas" .

ex:obj4 exProp:ano "2000";
       exProp:cultura "Arroz Sequeiro";
       exProp:tipoagrotoxico "produto comercial";
       rdfs:label "Agrotóxicos e suas culturas" .

ex:obj5 exProp:ano "2000";
       exProp:cultura "Banana";
       exProp:tipoagrotoxico "produto comercial";
       rdfs:label "Agrotóxicos e suas culturas" .
```

APÊNDICE C – EXEMPLO DE ARQUIVO CSV USADO COMO ENTRADA PARA O *PLUG-IN* DATA CUBE

```
year,geo,population_in_mil
Y2001,DE,1500
Y2002,DE,2000
```

APÊNDICE D – ARQUIVO RDF/TURTLE GERADO PELO *PLUG-IN* DATA CUBE

```
@base <http://example.cubeviz.org/datacube/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .
@prefix sdmx_code: <http://purl.org/linked-data/sdmx/2009/code#> .
@prefix sdmx_dimension: <http://purl.org/linked-data/sdmx/2009/dimension#> .
@prefix cube: <http://purl.org/linked-data/cube#> .
@prefix ex: <http://meu.exemplo/datacube/> .
@prefix exProp: <http://meu.exemplo/datacube/properties/> .

<http://purl.org/linked-data/cube> a owl:Ontology ;
rdfs:label "Example DataCube Knowledge Base" ;
dc:description "This knowledgebase contains one Data Structure Definition with one
Data Set. This Data Set has a couple of Components and Observations." .

# Data Structure Definitions

ex:dsd a cube:DataStructureDefinition ;
    rdfs:label "A Data Structure Definition"@en ;
    rdfs:comment "Defines the structure of a DataSet or slice." ;
    cube:component  <time>,
      <geo>,
      <value> .

# Component Specifications

<time> a cube:ComponentSpecification ;
    rdfs:label "Time" ;
    cube: cube:dimension exProp:year .

<geo> a cube:ComponentSpecification ;
    rdfs:label "Region" ;
    cube: cube:dimension exProp:geo .

<value> a cube:ComponentSpecification ;
    rdfs:label "Value" ;
    cube: cube:dimension exProp:measure .

# Data Set
rdfs:label "A DataSet"^^<http://www.w3.org/2001/XMLSchema#string> ;
rdfs:comment "Represents a collection of observations and conforming to some common
dimensional structure." ;
cube:structure ex:dsd .

# Dimensions, Unit and Measure
exProp:year a cube:DimensionProperty ;
    rdfs:label "Time"@en .

exProp:geo a cube:DimensionProperty ;
    rdfs:label "Region"@en .

exProp:unit a cube:AttributeProperty ;
    exProp:measure a cube:MeasureProperty ;
    rdfs:label "Value"@en .

ex:ob0 a cube:Observation;
    cube:dataSet ex:dataset;
    exProp:year ex:Y2001;
    exProp:geo ex:DE;
    exProp:unit "Value";
    exProp:value "1500"^^https://www.w3.org/2001/XMLSchema#float;
    rdfs:label "".
```

```
ex:ob1 a cube:Observation;
    cube:dataSet ex:dataset;
    exProp:year ex:Y2002;
    exProp:geo ex:DE;
    exProp:unit "Value";
    exProp:value "2000"^^https://www.w3.org/2001/XMLSchema#float;
    rdfs:label "".
```

APÊNDICE E – ESQUELETO DA CRIAÇÃO DO *PLUG-IN* NTRIPLE GENERATOR

```
package br.ufrj.ppgi.greco.kettle;

// imports

public class NTripleGeneratorStepData extends BaseStepData implements
StepDataInterface {
	public RowMetaInterface outputRowMeta;
	// define campos que podem ser acessados durante o processamento
}


package br.ufrj.ppgi.greco.kettle;

// imports

public class NTripleGeneratorStep extends BaseStep implements StepInterface {
	// vars globais

	public NTripleGeneratorStep(StepMeta stepMeta, StepDataInterface
stepDataInterface, int copyNr, TransMeta transMeta,
			Trans trans) {
		super(stepMeta, stepDataInterface, copyNr, transMeta, trans);
	}

	@Override
	public boolean init(StepMetaInterface smi, StepDataInterface sdi) {
		return super.init(smi, sdi);
		// inicializa o processamento de linhas de entrada
	}

	@Override
	public void dispose(StepMetaInterface smi, StepDataInterface sdi) {
		super.dispose(smi, sdi);
	}

	public boolean processRow(StepMetaInterface smi, StepDataInterface sdi)
throws KettleException {
		// método executado para cada linha de input
		// definir o que o plugin faz com a entrada e como é gerada a saída
	}
}



package br.ufrj.ppgi.greco.kettle;

/*imports*/

public class NTripleGeneratorStepDialog extends BaseStepDialog implements
StepDialogInterface {

	// var globais

	public NTripleGeneratorStepDialog(Shell parent, Object stepMeta, TransMeta
transMeta, String stepname) {
		super(parent, (BaseStepMeta) stepMeta, transMeta, stepname);
		// inicializações
	}


	public String open() {
		// define toda a criação da tela, tamanho de campos e posição dos
mesmos
		// e os valores que devem ser inicializados quando a tela é aberta
	}
```

```
	protected void cancel() {
		// comportamento do botão cancelar
	}

	protected void ok() {
		// comportamento do botão ok, salva os valores inputados pelo usuário
	}
}


package br.ufrj.ppgi.greco.kettle;

/*imports*/

public class NTripleGeneratorStepMeta extends BaseStepMeta implements
StepMetaInterface {

	/*variáveis globais*/

	public NTripleGeneratorStepMeta() {
		setDefault();
	}

	@Override
	public void check(List<CheckResultInterface> remarks, TransMeta transMeta,
StepMeta stepMeta, RowMetaInterface prev,
			String[] input, String[] output, RowMetaInterface info) {
		//tratamento de erros
	}

	public StepInterface getStep(StepMeta stepMeta, StepDataInterface
stepDataInterface, int copyNr,
			TransMeta transMeta, Trans trans) {
		return new NTripleGeneratorStep(stepMeta, stepDataInterface, copyNr,
transMeta, trans);
	}

	public StepDataInterface getStepData() {
		return new NTripleGeneratorStepData();
	}

	@Override
	public String getDialogClassName() {
		return NTripleGeneratorStepDialog.class.getName();
	}

	// Carregar campos a partir do XML de um .ktr
	@Override
	public void loadXML(Node stepDomNode, List<DatabaseMeta> databases,
Map<String, Counter> sequenceCounters)
			throws KettleXMLException {
		//leitura do arquivo xml/ktr de entrada
	}

	// Gerar XML para salvar um .ktr
	@Override
	public String getXML() throws KettleException {
		// salvar o arquivo xml/ktr de configuração
	}

	public void setDefault() {
		//valores default das variáveis globais
	}

	@Override
	public void getFields(RowMetaInterface inputRowMeta, String name,
RowMetaInterface[] info, StepMeta nextStep,
```

```
                    VariableSpace space) throws KettleStepException {
              // define os campos de saída
        }

        //setters e getters
}
```