Framework para el soporte a la metodología de Linked Open Data y su aplicación sobre diversos casos de uso

José Segarra , José Ortiz

July 14, 2016

Abstract

Con el avanzar del tiempo un número mayor de empresas e instituciones se han ido sumando a la iniciativa de Linked Open Data, atraídos por las múltiples ventajas que esta tecnología aporta en el reusó y en el aprovechamiento de la información. Sin embargo, algunos factores como la necesidad de conocimiento avanzado acerca de esta temática y la falta de herramientas completas que faciliten su aplicación han ocasionado que esta propuesta se vea limitada y no pueda ser aplicada en la mayoría de escenarios disponibles o por un usuario común. Para hacer frente a la escases de herramientas que cubran el proceso completo de generación y publicación de Linked Open Data se ha propuesto un nuevo framework basado en el gestor de procesos ETL de Pentaho, para proveer un entorno unificado en donde usuarios con conocimientos básicos en la tecnología de Linked Data puedan generar datos enlazados de calidad sobre un conjunto amplio de dominios. En el presente documento se prueba la factibilidad de aplicación del framework para tres dominios específicos en los que constan repositorios digitales, base de datos bibliográficas y base de datos empresariales con los que se pueden apreciar la flexibilidad de la propuesta y que pueden dar las pautas para extenderla sobre escenarios adicionales.

I. Introducción

La iniciativa de Linked Open Data (LOD) propone un conjunto mecanismos o mejores prácticas para compartir información, de modo que puedan superar las barreras de aislamiento y heterogeneidad de los datos en la creación de una web más integrada y con significado. Por este motivo muchas instituciones o empresas que desean compartir su información lo han hecho en RDF[1], que permite ser aprovechado por un número más grande de interesados ya que no se ven limitados por el formato que estas disponen ni la forma en que se encuentra almacenada la información. Aun cuando en la actualidad se pueden encontrar mucha información en este formato y cada vez más instituciones se suman a este movimiento, aún existen algunas dificultades notables que impiden aplicar esta tecnología de forma más amplia sobre potenciales fuentes y para muchos más dominios. Algunas de la complicaciones que perjudican su mas extensa aplicación están relacionadas con el elevado conocimiento que es requerido para aplicar esta tecnología, así como falta de guías integrales y herramientas completas que ayuden al usuario durante el proceso de generación y publicación de LOD.

Para hacer frente a la necesidad de guías que permitan la aplicación estandarizada del proceso de generación y publicación de LOD se han realizado algunos adelantos en trabajos como [13] y [11]. En los cuales se describen una serie de fases o etapas para soportar el proceso de generación y publicación de LOD a través de su ciclo de vida. En los mismos trabajos además de dar lineamientos para la aplicación de LOD y su mantenimiento se dan referencias a herramientas con las cuales se pueden soportar total o parcialmente los procesos requeridos durante cada una de estas fases. Sin embargo, a pesar que la aplicación de herramientas desarrolladas pueden proveer muchas ventajas en cuanto a tiempo y costo debido a que permite una reutilización de herramientas ya probadas, aun requieren de un notable esfuerzo en su integración. El elevado esfuerzo de integración se debe a que la mayoria de propuestas nacen de forma individual, con tecnologías propias y que en ocasiones deben ser adaptadas sobre escenarios específicos en la búsqueda de soluciones completas para el manejo de LOD. En cuanto a las contadas soluciones que proponen un entorno más integrado con el cual trabajar los datos enlazados, se pueden apreciar algunas restricciones que tienen que ver con el dominio en el cual pueden ser integrados y algunas carencias en las fases del proceso de generación y publicación LOD. En este trabajo para superar las dificultades mencionadas se propone el Framework para soporte del proceso de Linked Open Data en base a la metodología presentada en [13].El framework descrito utiliza como base la herramienta Kettle[2] perteneciente a la suite de Pentaho Data Integration, para proveer de un entorno unificado y gráfico que permita a un usuario con conocimientos básicos en tecnologías semánticas, transformar y generar sus propios datos enlazados de calidad, es decir siguiendo los principios 5 estrellas propuesto en [1].

[1] Resource Description Framework

[2] http://community.pentaho.com/projects/data-integration/

II. Antecedentes y Trabajos Relacionados

I. Antecedentes

En esta seccion se describen brevememte los principales conceptos y tecnologías usuadas dentro del *framework*.

Datos enlazados

Es un concepto acuñado dentro del desarrollo de la Web semántica que se refiere a la publicación de datos en la Web en un formato estructurado y definiendo enlaces a fuentes de datos externas. En los últimos años este concepto ha sido usado ampliamente para integración de información a través de la Web. La publicación de datos enlazados se fundamenta en cuatro principios[2].

- Usar URIs[3] para identificar los recursos.
- Las URIs (a través de HTTP) deben permitir la consulta y localización de los recursos a los usuarios.
- Proveer información detallada de los recursos a través de formatos estructurados.
- Incluir enlaces a otras las URIs, de manera que se pueda navegar y descubrir nueva información.

Los lenguajes estándar para el intercambio de información publicada como Linked Data son RDF[4] y SPARQL. RDF es un lenguaje simple de representación de datos basado en tripletas (Sujeto, Predicado, Objeto). Adicionalmente, RDF es *schemaless* lo que lo hace flexible para el manejo de información heterogénea y cambiante en el tiempo. Por otro lado, SPARQL es el lenguaje de consulta sobre RDF, este permite consultar información usando una sintaxis parecida a SQL. Adicionalmente, SPARQL provee funcionalidades de federación de consultas que es uno de los mecanismos para el acceso distribuido a la información publicada como Linked Data. Generalmente la información publicada en estos formatos es manejada por motores de bases de datos especializados llamados Triple Stores[5].

Generación de RDF

Es el proceso por el cual los datos son convertidos a RDF desde sus formatos originales (CSV, XML, OAI, etc.). Es proceso de conversión se realiza con herramientas denominadas RDF-izers, que son programas de ETL específicos para transformación a RDF. Estas herramientas generalmente utilizan mappings, que establecen relaciones entre los campos de los formatos originales con el RDF usando una ontología como vocabulario para modelar la información.

Entre las características que debe cubrir la transformación a RDF tenemos[13]: Conversión total, que implica que todas las consultas posibles sobre las fuentes originales deben ser posibles también en la versión RDF. Ajustarse la ontología,

[3] Uniform Resource Identifier

[4] Resource Description Framework

[5] Un Triplestore o RDF store es una base de datos diseñada para el almacenamiento y recuperación de tripletas a través de consultas semánticas.

los datos generados como RDF deben concordar en lo posible con la estructura de la ontología usada para modelar los datos.

En el ámbito tecnológico se han desarrollado un gran numero herramientas para la generación de RDF. La mayoría de estas están centradas en formatos específicos y consideran una sola fuente de datos a la vez. A continuación, se provee una lista de algunas de las herramientas destacas, agrupadas por el formato de datos al que dan soporte.

- CSV: Open Refine, XLWrap, RDF123, NOR2O.
- XML: GRDDL, XSLT, TopBraid Composer ReDeFer.
- OAI: OAI2RDF.
- Bases de datos relacionales: D2R Server, ODEMapster, Triplify, Virtuoso RDF View, UtraWrap, R2RML.

De las herramientas mencionadas, el lenguaje R2RML se ha convertido probablemente en el estándar más usado, esto debido a su flexibilidad y a la popularidad de las fuentes de datos relacionales. Este formato permite establecer un archivo de mappings (siguiendo la especificación R2RML) sobre información relacional (tablas, vistas, consultas) y usando ontologías con el fin de poder generar una versión RDF de los datos. Una de las características más importantes de R2RML es que permite generar tanto SPARQL Endpoints directamente a partir de los datos originales (acceso virtual) así como Dumps como archivos RDF (materialización de la información) que pueden ser usados posteriormente.

Modelamiento ontologico

Una de las tareas más importante en la generación de datos enlazados es el modelamiento ontológico de las fuentes de datos. Esto debido a que la correcta selección de ontologías/vocabularios y los conceptos a ser usados para representar los datos garantizan la completitud de la representación de la información, así como reusabilidad de la misma a través de la Web.

Actualmente existen varias herramientas que facilitan la creación y manipulación de ontologías, las cuales asisten en el proceso de definir modelos ontológicos adecuados para la generación de datos enlazados. Entre las herramientas de soporte al modelamiento ontológico podemos destacar: Protege[6] y Neon[7]. Adicionalmente, con el fin de facilitar la estandarización y reusabilidad de los datos enlazados se han creado repositorios de ontologías que facilitan a los desarrolladores de LOD encontrar conceptos adecuados a ser usados para representar sus datos.

Existe un varios de estos repositorios disponibles en la Web que permiten registrar y buscar ontologías. A continuación, se detallan los repositorios más destacados.

- Prefix[8]: Una base de datos que contiene los prefijos de las URLs de los vocabularios.

[6] http://protege.stanford.edu/
[7] http://www.neon-project.org/
[8] https://prefix.cc/

- Lov[9]: Registro de las principales ontologías con sus conceptos.
- Buscadores de documentos semanticos: Swoogle[10], Watson[11], Falcons[12]

Además, hay que destacar que todos estos repositorios poseen APIs que permiten la consulta de sus ontologías y conceptos. Esto los convierte en fuentes de información para aplicaciones de creación de mappings ontológicos de fuentes de datos.

Descubrimiento de enlaces

El encontrar enlaces entre distintos repositorios de datos es una de las premisas de Linked Data. El poder navegar entre los repositorios a través de estos enlaces da un valor agregado a los datos, debido a que esto permite una integración a nivel Web. Adicionalmente, el inter relacionamiento de los datos permite descubrir nueva información a través de fuentes de información externas enlazadas y la aplicación de algoritmos de Data mining.

Actualmente se han desarrollado varias herramientas para el descubrimiento de diversos tipos de enlaces (owl:sameAs, owl:differentFrom, etc.) entre los recursos de los repositorios de datos enlazados. Entre las herramientas más usadas podemos destacar: Silk Workbench[13] y Limes[14]. Estas herramientas permiten definir reglas para identificar recursos iguales o similares entre repositorios independientes. Los mecanismos usados para detectar relaciones entre entidades dentro de estas herramientas son varios: medidas de similitud semántica, similitud sintáctica, etc. Y permiten automatizar en cierta medida las tareas detección y desambiguación. Los enlaces encontrados por estas herramientas generalmente son extraídos como un nuevo archivo RDF que posteriormente debe ser publicado para su explotación.

Hay que destacar que los desarrolladores de Silk Workbench en un esfuerzo de estandarización han creado un lenguaje de especificación de enlaces (LSL)[15]. Este lenguaje permite describir procesos de descubrimiento de enlaces en un archivo XML, permitiendo la reutilización de estos.

Publicación de datos enlazados

El hacer públicos los datos en la Web es un requerimiento básico dentro de Linked Data. Además, el publicar los datos usando un estándar estructurado es necesario para la interoperabilidad con otros repositorios de datos enlazados. Actualmente la mayor parte de los datos enlazados producidos en la Web hacen uso de servicios SPARQL (SPARQL Endppoints) para publicar la información, debido a la flexibilidad de consulta que estos ofrecen y su adopción por la comunidad de la Web Semantica como un estándar para intercambio de información.

Existe una gran variedad de herramientas (Triplestores) que permiten almacenar datos enlazados (RDF), manipularlos y publicarlos como SPARQL

[9] http://lov.okfn.org/dataset/lov/
[10] http://swoogle.umbc.edu/
[11] http://watson.kmi.open.ac.uk/WatsonWUI/
[12] http://ws.nju.edu.cn/falcons/objectsearch/index.jsp
[13] http://silkframework.org/
[14] http://aksw.org/Projects/LIMES.html
[15] https://app.assembla.com/wiki/show/silk/Link_Specification_Language

Endpoints. A continuación, se presenta una lista con las herramientas más destacadas.

- Fuseki[16]. Es un SPARQL Server ligero desarrollado sobre la plataforma Apache Jena, permite funcionalidades avanzadas como inferencia, indexación por texto, consultas federadas, etc.
- Virtuoso. Es un motor de base de datos hibrido que posee funcionalidades de indexación fulltext, triplestore, almacén XML, RDF, etc.
- Sesame[17]. Es framework y base de datos para la manipulación de RDF desarrollado en Java.
- KiWi[18]. Es un triplestore en proceso de desarrollo que puede configurarse sobre una base de datos relacional y actualmente es usado en la plataforma de Linked Data Apache Marmotta[19].

Explotación de datos enlazados

Una de las premisas de Linked Data es la disponibilidad de la información tanto para humanos como para procesos automáticos. Si bien los estándares RDF y SPARQL permiten almacenar y consultar la información de forma flexible, estos no están pensados para usuarios inexpertos. Es por esta razón que la implementación de mecanismos para la fácil visualización y consulta de datos enlazados resultan imprescindibles para que la información pueda ser aprovechada.

Varias herramientas se han desarrollado para la mejorar la explotación de datos enlazados en los diversos ámbitos que puede tener la información: Map4RDF[20] (información geográfica), CubeViz[21] (información estadística), etc. Si bien los mecanismos de explotación de datos enlazados pueden variar significativamente de acuerdo al ámbito de la información, el acceder a la información de los recursos mediante páginas Web (usando su URI) se ha vuelto el mecanismo de explotación de datos enlazados más elemental.

En este contexto herramientas como ELDA API[22] y Pubby[23], proveen un primer nivel de explotación de datos enlazados, permitiendo visualizar/filtrar/buscar información de los recursos a través de páginas web amigables accesibles por la URI de los recursos.

II. Trabajo relacionado

Dentro del proceso de generación y publicación de Linked Open Data se han desarrollado algunas herramientas, las cuales pretenden cubrir diferentes fases de este proceso, como: extracción de datos y generación de RDF (CSV Import[24],

[16] https://jena.apache.org/documentation/serving_data/
[17] http://rdf4j.org/
[18] http://marmotta.apache.org/kiwi/triplestore.html
[19] http://marmotta.apache.org/
[20] http://oeg-dev.dia.fi.upm.es/map4rdf/
[21] http://aksw.org/Projects/CubeViz.html
[22] http://www.epimorphics.com/web/tools/elda.html
[23] http://wifo5-03.informatik.uni-mannheim.de/pubby/
[24] `http://aksw.org/Projects/CSVImport.html`

R2R[25], D2R[26] , RML[27] ,Karma[28]), limpieza (LOD Refine[29]), publicación (Virtuoso[30], Fuseki[31]), etc. Esta variedad de herramientas aunque en conjunto pueden llegar a cubrir todas las etapas del proceso mencionado, aún requieren de un gran esfuerzo manual y de conocimiento para su integración dentro de soluciones completas de Linked Data. La heterogeneidad de las fuentes de datos (OAI, Bases de datos, CSV, Marc 21, etc.) presentan también complicaciones adicionales que deben considerarse cuando se pretende trabajar con algunos dominios y para lo cual se pueden encontrar soluciones específicas.

En el ámbito de generación de Linked Data se pueden encontrar algunas herramientas destacadas como RML Editor[5] y Karma[8], que mediante interfaces gráficas Web, lenguajes de *mapping* (RML[4] y R2RML[3]) y soporte para fuentes de datos heterogéneas pueden convertirse en soluciones genericas más accesibles debido a la capacidad de configuración gráfica que brinda al usuario. Sin embargo, a pesar de todas las funcionalidades disponibles a través de estas herramientas, aún éstas se mantienen enfocadas sobre una fase específica de la metodología (generación de RDF), por lo que continúan siendo solo una solución parcial al proceso de publicación de LOD.

RML

RML Generic Mapping Language es un mecanismo para la generación de RDF basada en la especificación R2RML, que permite la generación RDF de múltiples fuentes de datos a la vez. RML expande el soporte de R2RML a fuentes de datos diferentes de las bases de datos relacionales (específicamente CSV, XML y JSON) y permite realizar mappings de varias fuentes de datos a la vez, enlazando automáticamente la información de estas. Además, RML posee una herramienta grafica (RML Editor) para facilitar la creación de los mappings de las fuentes de datos, lo que facilita en gran medida su utilización.

Entre las principales limitaciones de RML podemos destacar las siguientes. RML soporta un limitado número de fuentes de datos, y la incorporación de nuevos implicaría un arduo trabajo de implementación sobre los procesadores de este lenguaje. Adicionalmente, RML se enfoca exclusivamente en la generación de RDF a partir de mappings directos de las fuentes de datos lo que limita sus capacidades de limpieza y normalización de los datos. Finalmente, RML requiere de la utilización de herramientas adicionales para la publicación, descubrimiento de enlaces y explotación de la información.

Karma

Karma es una herramienta completa de integración de datos que permite generar RDF de varias fuentes de datos. Karma al igual que RML basan su funcionamiento de R2RML, extendiendo esta especiación para varios formatos de datos (CSV, XML, REST Web Services, etc.). Sin embargo, a diferencia de RML, Karma no permite la generación simultanea de varias fuentes de datos,

[25] `http://wifo5-03.informatik.uni-mannheim.de/bizer/r2r/`
[26] `http://d2rq.org/`
[27] `http://rml.io/RMLeditor.html`
[28] `http://usc-isi-i2.github.io/karma/`
[29] `https://github.com/sparkica/LODRefine`
[30] `http://virtuoso.openlinksw.com/`
[31] `https://jena.apache.org/documentation/serving_data/`

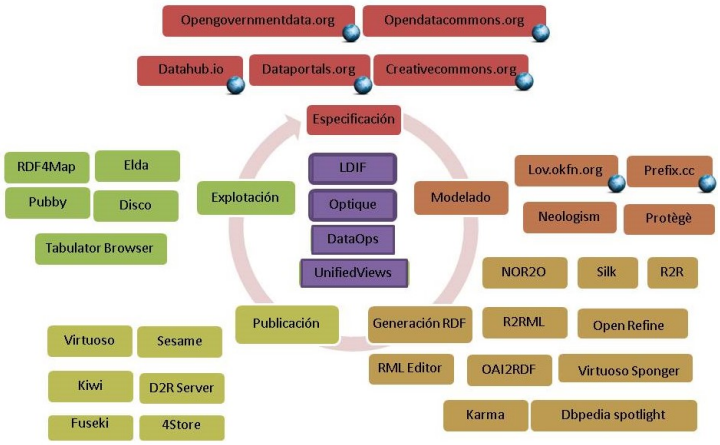

Figure 1: Distribución de herramientas LOD

dado que el proceso de generación se realiza de forma independiente para cada fuente. Por otro lado, una de las características destacadas de Karma es la utilización de algoritmos avanzados para sugerir mappings de las fuentes de datos basados en los vocabularios usados y las acciones previas del usuario.

Entre las principales debilidades esta herramienta podemos destacar las siguientes. Al igual que RML la extensión de Karma para soportar nuevas fuentes de datos requiriera de modificaciones importantes sobre la aplicación. Por otro lado, a pesar de que las actividades como limpieza y normalización de los datos pueden realizarse de una forma básica mediante los mecanismos disponibles en la herramienta (inserción de datos de ejemplo), los errores más complejos en los datos requieren de procesamiento externo para poder solucionarse. Finalmente, al ser una herramienta de generación de RDF no provee funcionalidades de publicación, enlace y explotación de datos enlazados.

En la figura 1 se muestra a modo de resumen, la distribución de las herramientas existentes según su aporte a las fases definidas en la metodología para aplicación de Linked Data presentada en [13]. El gran número de herramientas disponibles corresponde a la variedad de formatos de datos que deben ser traducidos a RDF, así como las otras acciones que deben realizarse (publicación, enlace, etc.) [12]. Las herramientas aisladas se presentan a la periferia del gráfico y alineadas con las etapas que cubren. Adicionalmente, en el centro de la figura se han posicionado a los proyectos de software que buscan dar una solución integral al proceso de aplicación de Linked Data, los cuales se tratarán a continuación.

Respecto a los trabajos enfocados en cubrir todo el proceso de publicación de Linked Data, los cuales tienen como objetivo superar las brechas generadas por la independencia de herramientas para brindar un medio de ejecución común. Estos enfoques se pueden dividir en dos categorías: gestores ETL con soporte integral para Linked Data y *frameworks* para gestión de Linked Data. A continuacion se describen las principales herramientas desarrolladas en cada caso.

UnifiedViews

Dentro de los gestores de procesos ETL con soporte para Linked Data, UnifiedViews[32] es uno de los trabajos más destacados[7] debido a que permite la definición de flujos de datos en una interfaz gráfica, donde se pueden realizar: extracción, transformación RDF y publicación de Linked Data. Adicionalmente, las funcionalidades de la herramienta pueden ser extendidas mediante el desarrollo de nuevas *Data Procesing Units* (DPU)[33]. Sin embargo, al ser una herramienta desarrollada desde cero su mayor limitación es el reducido stock de DPUs disponibles. Entre las principales carencias son destacables: soporte de fuentes de datos reducido (Bases de Datos, CSV, RDF y SPARQL), limpieza de datos limitada (no existen operaciones sobre cadenas, filtros, etc.) y funcionalidades para explotación de datos no disponibles.

Esta plataforma es en sí, es un gestor ETL desarrollado para dar soporte a tecnologías semántica como: RDF y SPARQL. UnifiedViews permite definir flujos de procesamiento de datos, con el fin de transfórmalos y publicarlos como Linked Data. Una de sus características es que soporta varios tipos de información (relacional, archivos, XML, RDF, tabular, etc), lo que la convierte en una buena herramienta para el desarrollo de soluciones de Linked Data. Adicionalmente, debido a su arquitectura modular (basada en DPUs) permite la fácil expansión con nuevas funcionalidades, soporte de fuentes de datos, publicación, etc.

A pesar de las ventajas de UnifiedViews, esta herramienta tiene varias limitaciones, las cuales se describen a continuación. UnifiedViews es una herramienta nueva, lo que implica que su comunidad desarrolladores aun no está consolidada y que el número de DPUs disponibles es aún reducido. La falta de desarrollo origina que UnifiedViews posea un reducido número fuentes de datos soportadas, que las funcionalidades de limpieza y normalización de datos sean reducidas, de la misma forma que las opciones de explotación de datos. Adicionalmente, UnifiedViews proporciona un nivel de asistencia básico en las actividades de *mapping* de fuentes de datos, lo que limita sus potencialidades de modelar ontológicamente las fuentes de datos.

Para ejemplificar las capacidades de esta herramienta a continuación se describe a breves rasgos un flujo típico que podría ser generado con UnifiedViews. En este ejemplo se cobren transformación y publicación información disponible en la Web hasta su publicación como datos enlazados en un triplestore.

- Descargar archivo ZIP (que contiene la información) desde un servidor FTP en Internet.
- Descomprimir el archivo y extraer un fichero XML con los datos.
- Transformación de los datos XML a RDF usando un template XSLT.
- Publicación del RDF en un TripleStore Virtuoso.
- Descubrimiento de enlaces entre los datos publicados y fuentes externas usando SILK.
- Actualización del Triplestore con los nuevos enlaces encontrados.

[32] `https://github.com/UnifiedViews/Core`
[33] DPU: Plugins adicionales al core de la herramienta

En el ejemplo planteado, UnifiedViews abordar la mayoría de las fases del ciclo de vida de LOD. Sin embargo, la limpieza, estandarización y explotación de datos no es considerada debido a la ausencia de DPUs especializadas para estos fines. Adicionalmente, el proceso de modelamiento de los datos (creación del template XSLT) es totalmente manual, es decir, no se proveen mecanismos de asistencia.

En cuanto a los Frameworks para la gestión de Linked Data, se han realizados varios esfuerzos por generar soluciones *All-In-One* para tratar con datos enlazado. Soluciones que permitan generar, publicar y explotar datos enlazados desde una única plataforma, la cual centraliza toda la información. Estas herramientas siguen varios enfoques para generar este tipo de soluciones en diversos ámbitos. A continuación, se detallan los trabajos más destacados en este contexto.

LDIF

Linked Data Integration Framework (LDIF) es un framework pensado para la integración de datos disponibles en la Web[10]. Esta herramienta soporta un reducido número de fuentes de datos (RDF, SPARQL, Web Crawler [34]) que en su mayoría están basadas en estándares semánticos, es decir, no cubre el problema de la integración de fuentes de datos heterogeneas, también permite la unificación y enlace entre ontologías (usando R2R y Silk[35]), sin embargo, no considera limpieza de datos. La publicación de los datos se realiza directamente en archivos RDF o SPARQL Endpoints y la explotación de datos no esta considerada dentro de la plataforma. Además, la aplicación no provee una interfaz gráfica y su configuración es compleja.

Para ejemplificar las capacidades esta herramienta a continuación se describe a breves rasgos el procedimiento básico seguido en esta herramienta para genera datos enlazados. En este ejemplo se extrae información desde páginas Web, se procesa y publica como datos enlazados sobre un triplestore.

- Configuración de un Crawler (LDSpider) para extraer Linked Data de la Web.
- Ejecución del crawler para extraer RDF.
- Normalización del vocabulario mediante R2R (archivo de mapping).
- Descubrimiento de enlaces entre los nuevos datos y preexistentes utilizando Silk.
- Evaluación de la calidad de los datos usando Sieve.
- Publicación los datos un SPARQL Endpoints existente o como un Dump RDF.

En el flujo presentado, LDIF abordar gran parte del problema de la generación de datos enlazados. Sin embargo, al ser una herramienta pensada para el procesamiento de información Web, no considera fuentes de datos de otros

[34] A Web crawler is an Internet bot which systematically browses the World Wide Web, typically for the purpose of Web indexing

[35] `http://silkframework.org/`

ámbitos (bases de datos, Marc 21, etc.). Adicionalmente, LDIF ofrece capacidades limitadas en la limpieza de errores y normalización de los datos, de la misma forma que la explotación de los datos no se considera dentro los módulos de la herramienta.

DataOps

Information Workbench (Dataops) es un framework para la integración de datos empresariales a través de ontologías[9]. Este framework extrae y transforma a RDF información de diversas fuentes, sin embargo, se centra especialmente en fuentes semánticas (RDF, OWL, SPARQL) y un reducido número de formatos no semánticos (XML, CSV, Base de datos). Entre las características que contempla este framework están la limpieza de datos mediante Open Refine[36] y el enlace de datos usando Silk Workbench, no obstante, el framework se limita a la importación de datos desde dichas herramientas las cuales requieren ser manipuladas de forma independiente. Por otro lado, la publicación de los datos se realiza únicamente en el TripleStore[37] embebido, lo que le reduce su flexibilidad. Adicionalmente, la explotación de los datos en este framework requiere la utilización Widgets, los cuales necesitan configuraciones adicionales. Finalmente, esta herramienta posee una interfaz gráfica Web y su configuración es simple, aunque limitada cuando se requieren funcionalidades avanzadas.

Para ejemplificar las capacidades de DataOps, a continuación, se describe un proceso típico que sigue esta herramienta para generar y publicar datos enlazados. En este ejemplo se procesa información de un archivo CSV hasta convertirla en datos enlazados.

- La información proveniente del CSV es procesada en OpenRefine para encontrar y corregir errores.
- La información corregida es importada a DataOps usando la API REST de OpenRefine.
- Las columnas de los datos son mapeadas con los términos de una ontología dentro de DataOps.
- Los datos son transformados en RDF siguiendo los mappings especificados.
- Se descubren enlaces entre los nuevos datos en RDF con la información preexistente en el repositorio (Usando Silk).
- La información es publicada dentro del triplestore embebido dentro de Dataops.
- Se configuran mecanismos para la explotación de los datos usando widgets ya desarrollados.

Dataops permite cubrir todas las etapas de la generación de datos enlazados para el ejemplo propuesto. Sin embargo, parte del proceso requiere de la utilización de herramientas externas (OpenRefine), además, el proceso que se

[36] `http://openrefine.org/`

[37] A triplestore or RDF store is a purpose-built database for the storage and retrieval of triples through semantic queries.

realizan dentro de la herramienta no pueden ser modificados fácilmente para adaptarlos a casos particulares. Por otro lado, el agregar nuevas fuentes de datos a la herramienta puede significar un proceso de desarrollo extenso debido que su arquitectura es poco modular.

Optique

Optique es un framework de acceso a Big Data desarrollado sobre Information Workbench, que integra de forma virtual fuentes de datos relacionales a través de ontologías OWL 2[38] [6]. La extracción de datos se centra exclusivamente en bases de datos, ignorando otros tipos de fuentes mientras que la limpieza de datos no se considera puesto que se presume que están curados desde la fuente. El enlace de información se realiza a nivel de modelo de datos a través de mapeos entre esquemas relacionales y ontologías, los cuales pueden ser consultados posteriormente por medio de un SPARQL Endpoint, el cual permite la ejecución de consultas virtuales que son traducidas a los lenguajes nativos de las fuentes de datos (Publicación). Para la explotación de los datos se posee un generador de consultas gráfico cuyos resultados son presentados con ayuda de widgets. La herramienta posee una interfaz gráfica Web amigable con los usuarios y para su configuración se requieren conocimientos de OWL 2 y R2RML[39].

Para ejemplificar las capacidades de Optique, a continuación, se describe a breves rasgos un procedimiento de generación de datos enlazados con esta herramienta. Este ejemplo se generan datos enlazados a partir de una base de datos relacional, siguiendo un enfoque virtual de acceso a la información.

- Se registran dentro de la herramienta la base de datos desde la cual se genera LOD y ontologías para representar la información.
- Se procede a realizar mappings entre el esquema de los datos y los conceptos de las ontologías.
- Optique genera un SPARQL Endpoint virtual de los datos siguiendo los mappings definidos. Las consultas SPARQL son traducidas a SQL, y los datos retornados se transforman a RDF en tiempo de ejecución.
- Se configura la explotación de los datos mediante la utilización de widgets ya desarrollados de la herramienta como: generador gráfico de consultas, mapas, tablas, etc.

Optique permite generar soluciones de integración virtual de datos usando datos enlazados, las cuales cubren varias de las etapas del ciclo de vida de LOD. Sin embargo, Optique al tener un enfoque más bien empresarial, se centra en fuentes de datos populares en dicho ámbito (bases de datos), dejando de lado otros formatos. Adicionalmente, etapas como limpieza, normalización y enlace de datos no son considerados por esta herramienta. Por otro lado, el extender el funcionamiento de Optique para agregar las etapas faltantes o nuevas fuentes de datos sería una tarea compleja por la arquitectura de integración virtual que implementa esta plataforma.

[38] `https://www.w3.org/TR/owl2-overview/`
[39] `https://www.w3.org/TR/r2rml/`

Como se ha descrito en esta sección, las plataformas que actualmente dan soporte a la generación y publicación LOD aunque poseen caracteristicas destacables en su respectivo ámbito de uso, aún poseen carencias que les impiden cubrir la etapas de las guias metodológicas y buenas practicas exitentes en su totalidad. Las principales debilidades que se aprecian en estas herramientas son: acceso a un reducido número de fuentes de datos, limpieza de datos limitada o inexistente, difícil incorporación de nuevos componentes (a excepción de UnifiedViews) y configuración compleja de nuevos procesos de generación de LOD. Considerando las debilidades mencionadas, en la siguiente sección se propone una plataforma de generación y publicación LOD que esta diseñada para cubrir las diferentes fases de la metodología propuesta en [13].

III. Framework para el soporte del ciclo de vida LOD

La propuesta presentada en [13] brinda un marco de referencia claro y general para llevar a cabo el proceso de generación y publicación de LOD a traves de su ciclo de vida, por lo que ha sido utilizado como base en el desarrollo del framework propuesto. La propuesta mencionada consta de 5 etapas principales como se puede apreciar en la figura 2.

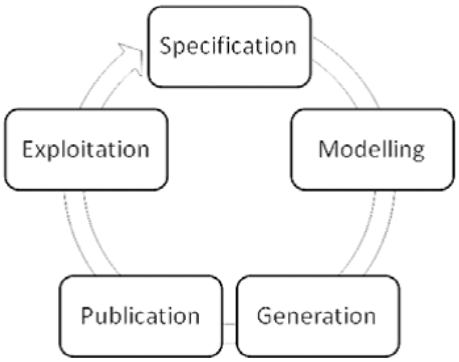

Figure 2: Fases de la metodología LOD [13]

A continuación se describe cada una de las fases mencionadas:

Especificación: Esta etapa se enfoca en definir las fuentes de la cual se extraerán los datos y las requerimientos necesarios para ejecutar el proceso de publicación de datos enlazados. También consta la definición de las URIs que se asignaran a los recursos y el licenciamiento que poseerán los datos que dentro de este contexto deben ser abiertos.

Modelado: En esta etapa se debe identificar, seleccionar o generar vocabularios que permitan describir semánticamente los datos de las fuentes disponibles de acuerdo a su dominio. Es aconsejable la reutilización de ontologías en lo posible, debido a que evita el desarrollo innecesario y facilita su interpretación si se encuentra en vocabularios conocidos.

Generación: Etapa central del proceso de Linked Data que consta de algunos procesos como extracción de datos, limpieza, generación de RDF y enlace (Linking). El objetivo de esta etapa es la conversión de datos en RDF considerando aspectos como la fiabilidad de los datos y el descubrimiento de relaciones entre recursos que permiten generar datos enlazados de calidad según la recomendación propuesta [1]

Publicación: Consiste en hacer públicos y accesibles los datos obtenidos de las etapa de generación para que puedan ser consumidos por las entidades interesadas. Dentro de esta etapa es común utilizar triple-stores como Virtuoso[40], Fuseki[41], Apache Marmota[42] , entre otros para almacenar la información y compartirla gracias a los servicios de Sparql Endpoint.

[40] `http://virtuoso.openlinksw.com/`
[41] `https://jena.apache.org/documentation/serving_data/`
[42] `http://marmotta.apache.org/`

Explotación: Esta etapa contempla el uso o desarrollo de herramientas que permitan realizar un mejor aprovechamiento de los datos. Por lo regular se orienta en brindar un acceso más amigable de la información generada para que pueda ser consumido por los usuarios.

El proceso descrito anteriormente presenta un conjunto de etapas que se puede usar como guía en el desarrollo ordenado y progresivo de publicación de LOD. Para la ejecución de estas etapas es posible aplicar actualmente algunas herramientas desarrolladas disponibles en la web que permiten soportar total o parcialmente los procesos involucrados en cada una de estas. Sin embargo al utilizar soluciones aisladas se debe tener presente que tareas como integración y ajustes sobre las herramientas son acciones requeridas en función de obtener una solución integral. Para evitar los problemas mencionados en la búsqueda de una solución integrada y extensible hacia diversos dominios, se ha desarrollado un framework para el soporte de la metodología de generación y publicación de Linked Data, utilizando como base la herramienta Kettle de Pentaho Data Integration.

Pentaho es una solución de integración de datos a nivel empresarial conformado por varias herramientas orientadas a la aplicación de BI (Inteligencia de Negocios) sobre los datos. Dentro de las herramientas que componen la suite de Pentaho se puede encontrar a Kettle como una de las destacadas, la cual es una herramienta de procesos ETL que mediante un conjunto de plugins gráficos permite generar complejos flujos de transformación de datos. Esta herramienta es muy utilizada en labores de limpieza e integración de datos en el ámbito empresarial. En este trabajo se utiliza la herramienta Kettle como base para el soporte de la metodología de LOD, sin embargo como Kettle es de ámbito general se ha desarrollado ciertos componentes en forma de plugins, que permitan completar las funcionalidades ofrecidas por los plugins propios de la herramienta y que posteriormente podrán ser usados de forma gráfica por el usuario. Los plugins desarrollados que forman parte del framework pueden ser obtenidos del repositorio Github del proyecto `https://github.com/santteegt/lodplatform`. Para su funcionamiento se requiere adicionar los plugins contenidos en el proyecto dentro de Pentaho Data Integration (Kettle) en la carpeta "Plugins".

En la figura 3 se muestra una gráfica con los componentes principales de la arquitectura propuesta en el framework, que dependiendo de los requerimientos y procesos existentes para cada una de las etapas son solventados con uno o más plugins. Para algunas de las fases como la de especificación dependiendo del formato de las fuentes puede requerirse utilizar algunos de los plugins propios de Kettle como el lector de hojas de cálculo o uno específico como el lector de recursos OAI.

A continuación se detalla como se brinda soporte al proceso de LOD mediante el framework y sus componentes.

Soporte a la etapa de especificación

Para el soporte a la etapa de especificación los plugins dentro del framework deben ser capaces de soportar la entrada de múltiples formatos, los cuales dependerán de las fuentes de datos disponibles como se puede ver en la parte inicial de la figura 3. Dependiendo del dominio en la mayoría de ocasiones los datos podrán ser leídos utilizando los plugins propios de la herramienta Kettle que

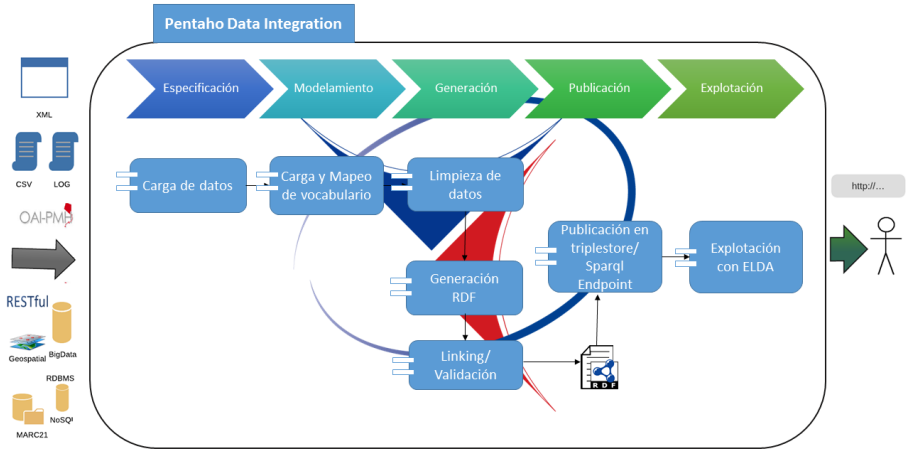

Figure 3: Arquitectura planteada por el framework

soportan una amplia gama de formatos (Csv, Excel, Base de datos, Servicios Web, etc), sin embargo, en caso de no ser posible gracias a la escalabilidad que permite la herramienta se puede extender las capacidades de lectura de formatos para ciertos dominios mediante el desarrollo de plugins específicos. Los plugins de esta etapa en concreto están diseñados para permitir la carga de datos de las distintas fuentes y presentarlos en la aplicación en forma de tabla que es la forma en que se maneja los datos en la herramienta.

A continuación se presentan dos de los plugins desarrollados dentro del framework, destinados a la extracción y lectura de datos de fuentes bibliográficas y de repositorios digitales:

Plugin de lectura de repositorios digitales

En la figura 4 se muestra la interfaz de configuración para uno los plugins desarrollados, que puede ser utilizado en la lectura de repositorios digitales mediante el protocolo OAI-PMH[43]. Entre las opciones que permite definir el plugin "OAI-PMH Loader" se encuentra:

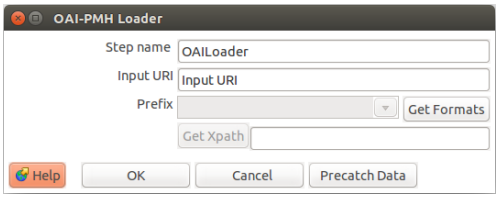

Figure 4: Configuraciones del plugin de carga (OAILOADER)

- Input URI: Ruta de acceso al servicio OAI-PMH perteneciente al repositorio .

[43] `https://www.openarchives.org/pmh/`

- Prefix: Especificación del formato de lectura que puede ser extraído con el servicio, por ejemplo OAI-DC o XOAI .
- GetXPATH: Ruta desde la cual se empezara a leer la información.

Plugin de lectura para archivos bibliotecarios

En la figura 5 se puede apreciar la interfaz de uno de los plugins desarrollados para la lectura de archivos de fuentes bibliográficas almacenados bajo el formato Marc 21. Los archivos almacenados bajo el formato Marc 21 constan con una organización especial de almacenamiento, donde los campos que contienen la información se encuentran de forma secuencial y deben ser extraidos siguiendo la información que proveen unos descriptores, razón por la cual fue necesario el desarrollo de un plugin específico para este formato. Entre las opciones que se pueden configurar en el plugin para la lectura de archivos Marc 21 se encuentran:

- Batch: Opción de ejecutar los archivos por lotes.
- Name of the marcfile: Ubicación del archivo con extensión Marc 21.
- Generate Marc XML: Opción para generar la información como XML .
- Field to load separate with @: Campos que seran extraídos del fichero Marc 21 separados mediante @.

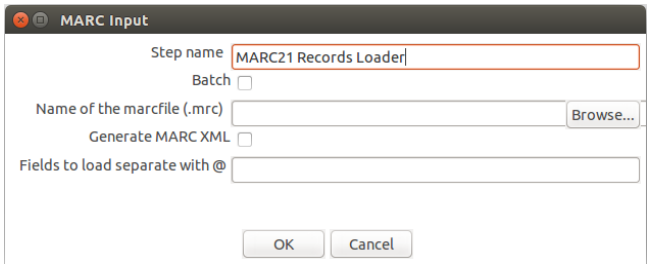

Figure 5: Configuraciones del plugin de carga (Marc21 Input)

La descripción de plugins adicionales para la lectura de datos (csv, hojas de calculo, etc.) pueden encontrarse en la documentación propio de la herramienta Kettle.

Soporte a la etapa de modelamiento

En esta etapa se selecciona y carga los vocabularios de ontologías que serán utilizados en el modelo de datos semántico. Dentro del framework propuesto para soportar la carga de ontologías se ha generado un plugin especializado “Get Properties OWL”, el cual requiere ya sea del prefijo de la ontología o su archivo para cargar sus elementos (clases y propiedades) en el flujo. Los elementos del modelo ontológico dentro del flujo pueden ser utilizados posteriormente en la etapa de generación de documentos RDF al mapearse con los datos obtenidos de las fuentes. En la figura 6 se puede ver la interfaz del plugin, en la cual se dispone de los siguientes opciones:

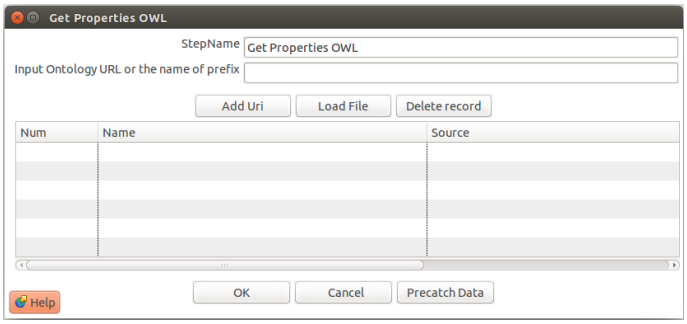

Figure 6: Configuraciones del plugin de carga del modelo Ontológico (GetPropertiesOWL).

- Input URL or the Name of Prefix: En este campo se puede ingresar la URL completa de la ontología o su prefijo[44].
- Add URI: Mediante este botón se puede cargar la ontología definida en el campo anterior.
- Load File: Despliega un ventana en la cual se puede ingresar la ruta del archivo que contiene la ontología.
- Delete Record: Permite borrar una ontología seleccionada en la grilla inferior.
- Precatch Data: Con esta opción se puede precargar el vocabulario en el framework y facilitar el acceso a los atributos de la ontología en la declaración del mapeo.

Soporte a la etapa de Generación

La etapa de generación es una de las etapas centrales del proceso y consta de los siguientes procesos:

Limpieza

Una de los principales procesos dentro de la etapa de generación es la Limpieza de datos, que tiene como objetivo asegurar la calidad mínima de los datos y su estandarización. Para la etapa de limpieza Pentaho dispone de un conjunto muy completo de plugins que están diseñados para soportar la transformación y manipulación de datos, haciendo de dicha funcionalidad una de sus características más destacadas. En este caso el usuario puede utilizar los plugins predefinidos en la generación de tareas de filtrado y transformación segun las errores encontrados en los datos. Los plugins comúnmente utilizados en labores de limpieza se muestran en la tabla 1:

Un paso requerido una vez realizado el proceso de limpieza dentro del framework es el almacenamiento de los datos dentro de una base de datos temporal o

[44]En caso de ingresar el prefijo se buscara dentro de un servicio la ruta completa de la ontología para su carga

Plugin	Descripción
String operations	Permite realizar operaciones con cadenas de caracteres, como remover números, eliminar caracteres especiales, quitar espacios en blanco entre otros.
Replace in string	Brinda la posibilidad de realizar reemplazos sobre cadenas de caracteres, con lo cual se puede eliminar caracteres desconocidos o erróneos.
Value Mapper	Con este plugin se pueden mapear valores de los campos con otros, lo cual es útil cuando se quiere realizar estandarizaciones.
Split Fields	Se puede utilizar cuando se tiene más de un tipo de información en un campo y se requiere separarla.

Table 1: Principales plugins para Limpieza disponibles en Kettle

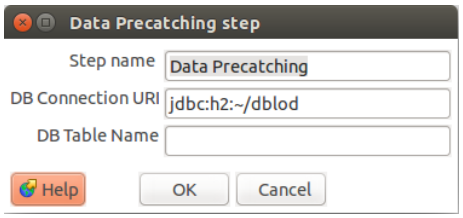

Figure 7: Configuración del plugin de cache (Data Precatching)

cache, lo cual se lo realiza mediante el plugin "Data Precatching". Es necesario realizar este proceso intermedio debido a que la cantidad de datos que se maneja para algunas fuentes es elevada y presentaría un impacto negativo si toda la información se la maneja únicamente en memoria. En la figura 7 se presenta el plugin mencionado, donde las configuraciones a realizarse se limitan a declarar una ruta de conexión con la base de datos, que por lo general se puede dejar con un valor por defecto.

Generación

Una vez que los datos están estandarizados y libre de errores se procede a la trasformación de los datos a formato estándar de intercambio de información en Linked Data como es RDF. Este proceso consiste básicamente en el mapeo del vocabulario definido con los datos extraídos y procesados previamente de la fuente. Para realizar dicha tarea se utiliza el plugin de mapeo desarrollado (Ontology Mapping) como se muestra en la figura 8, en donde es posible configurar y especificar la relaciones existentes de los datos extraídos con el vocabulario seleccionado. Las principales opciones que se pueden definir dentro del plugin son:

- Ontology Step: En este campo se debe definir la denominación con la que se encuentra el plugin de mapeo "GetPropertiesOWL".
- Data Step: En este campo se define la denominación con la que se encuentra el plugin de cache "Data Precatching".

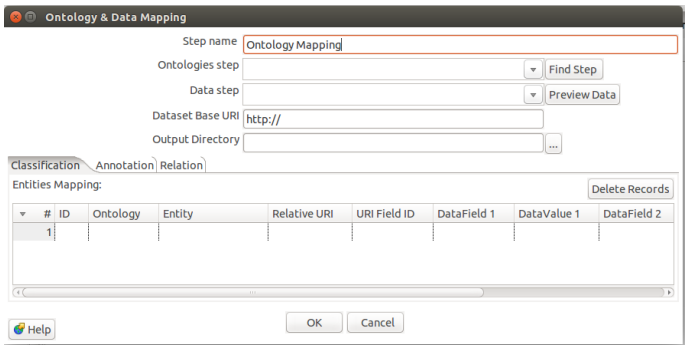

Figure 8: Configuración del plugin de mapeo (Ontology&DataMapping)

- Data Base URI: Definición de la URI absoluta con la que se generaran los nuevos recursos. Por lo general apunta a una dirección web en la que se puede encontrar descripción del recurso.
- Output Directory: Ruta en la cual se almacenara el archivo de mapeo generado R2RML.

En la parte inferior se dispone de una tabla que según las relaciones que pueden requerirse, permite declarar diferentes tipos de mapeos. Los mapeos que se pueden configurar son:

Mapeos por Clasificación o Tipo: En este se definen los registros o datos como un tipo específico de recurso. Dentro de la tabla para declarar los mapeos por clasificación se disponen de los siguientes campos.

- ID: Un identificador que se genera automáticamente para identificar el mapeo definido de entidades.
- Ontology/ Entity: Nombre de la ontología y el vocabulario específico con el cual se relacionara un registro para definirlo como recurso. Ejemplo: foaf/foaf:person
- Relative URI: URI relativa que se complementara con la URI absoluta para formar la URI del recurso. Por ejemplo (persona/)
- URI Field ID: Campo de los registros dentro del flujo que pasara a convertirse en el identificador único de cada recurso. Por ejemplo Data: Nombre.
- Data Field/Data Value : Campo y valor que debe tener un registro para que sea considerado en el mapeo. Por ejemplo Field/Autor

Mapeos de anotación o propiedades: Permite relacionar las propiedades de un recurso definido con un vocabulario. Los campos que se dispone son:

- ID: Un identificador que se genera automáticamente para identificar el mapeo definido para propiedades.

- EntityClassID: Permite definir el ID de la entidad mapeada con la cual se relacionaran las propiedades declaradas.
- Ontology/ Property: En estos campos se definen la ontología y el vocabulario que se usa para representar la relación de propiedad. Ejemplo: foaf/foaf:name
- Extraction Field: Campo del registro del cual tomara el valor la propiedad. Por ejemplo "Data: Antonio Ramirez."
- Data Field/Data Value : Campo y valor que debe cumplir el registro para aplicarse la regla de mapeo por propiedad. Ejemplo: Field /Nombre del autor.
- Data Type: Definición del tipo de dato que representa la propiedad. Por ejemplo: String.

Mapeo de Relación: Los mapeos de relaciones permiten definir semánticamente las relaciones existentes entre los recursos encontrados. Para declara las relaciones se dispone de los siguientes campos:

- ID: Un identificador que se genera automáticamente para identificar el mapeo definido para definir las relaciones entre recursos.
- EntityClassID 1 / EntityClassID 2 : Permite definir el ID de la primera y segunda entidad que van a ser relacionadas.
- Ontology/ Property: En estos campos se definen la ontología y el vocabulario con el cual se identificara la relación. Ejemplo (dcterms/dcterm:contributor)

Las relaciones definidas por el plugin son procesadas y mapeadas en un archivo de configuración R2RML[45] que servirá como entrada a la siguiente plugin, que será el encargado de generar el archivo en el formato esperado (RDF o Turtle).

Para el funcionamiento del plugin "RDF Generation" se utiliza el archivo definido en R2RML obtenido anteriormente para convertir los datos almacenados temporalmente en una base de datos en archivos RDF automáticamente. La interfaz de configuración se puede apreciar en la figura 9. Entre los campos que se pueden definir en el plugin se encuentran:

- R2RML File: Permite definir la ruta del archivo R2RML generado en el plugin anterior.
- SQL Vendor: Con esta opción se puede seleccionar el proveedor de la base de datos. Por defecto es H2.
- Data Base URL: Ruta de la base de datos donde se encuentra los datos transformados en cache.
- Data Base Schema: Esquema de la base de datos.
- Username/Password: Credenciales para acceder a la base de datos.

[45]Lenguaje para el mapeo personalizado de base relacionales a RDF

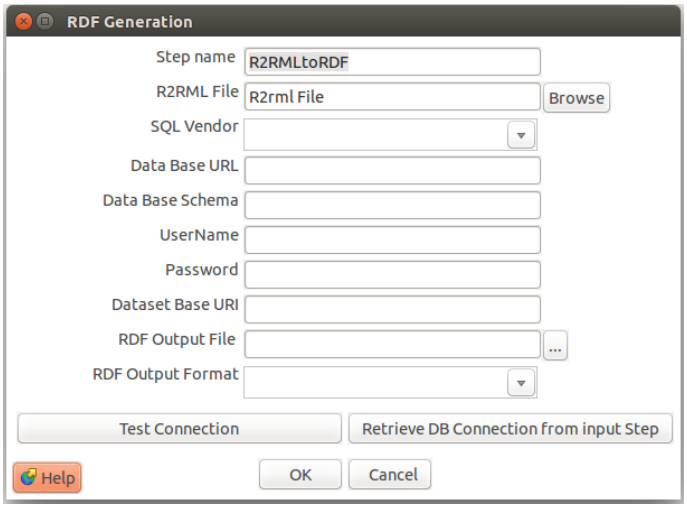

Figure 9: Configuración plugin para generación RDF (RDFGeneration)

- Data base URI: URI absoluta para los recursos.
- RDF output File: Ruta de salida con el archivo RDF.
- RDF output Format: Formato específico en el cual se generara el RDF. (Disponible XML y TTL).
- Retrieve DB connection from input step: Con este botón podemos recuperar las configuraciones de base de datos realizadas en el plugin "Data Precatching".

Con el objetivo de seguir los principios recomendados en la generación de LOD, se procede al enlace de los datos generados con fuentes externas (*Linking*) tal como consta en la metodología. Para el enlace de datos se ha desarrollado un plugin específico (Silk Plugin) que permite utilizar la potencialidad de SILK Workbench[46] de forma integrada y sencilla para encontrar los recursos similares entre dos fuentes. Se necesita reconocer los recursos similares entre diferentes fuentes debido a que este paso brinda la característica de datos enlazados que se busca al aplicar LOD con el cual se puede aumentar la información que se puede aprovechar. En la figura 10 se puede ver las configuraciones que se puede realizar en el plugin para la ejecución del proceso de *Linking* entre dos fuentes.

Los campos que contiene este plugin son los siguientes:

- Insert First Endpoint/Graph: Sirve para definir el primer endpoint o endpoint base y su grafo con el cual se ejecutara el proceso de *Linking*.
- Insert Second Endpoint/Graph: Sirve para definir el segundo endpoint o endpoint objetivo y su grafo con el cual se ejecutara el proceso de *Linking*.
- File SLS (Opcional) : Con esta opción se puede cargar directamente un archivos SLS de SLIK que contenga las configuraciones que se desean ejecutar.

[46] http://silkframework.org/

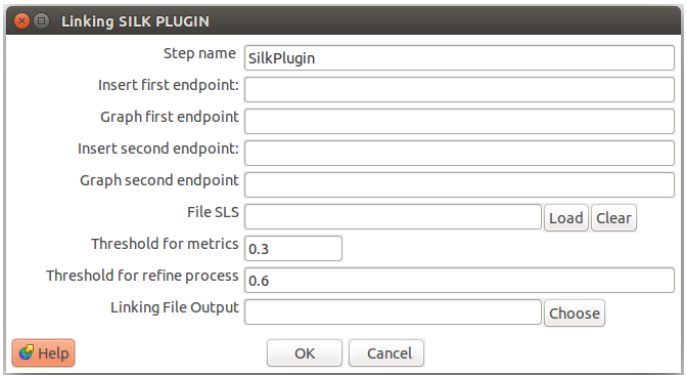

Figure 10: Configuración del plugin para enlace de datos.

- Threshold for metrics: Permite definir el umbral de la métrica de similitud empleado en el proceso de encontrar recursos similares.
- Threshold for refine process: Permite definir los umbrales para el proceso de validación empleando metricas semánticas.
- Linking File Output: En este campo se define la ruta con los resultados del proceso de enlace.

Como se puede apreciar en las opciones anteriores, para ejecutar el proceso de enlace de datos (*Linking*) dentro del plugin se dispone de dos opciones:

- Ingresar un archivo de configuración en Silk Link Specification Language (Silk-LSL).
- Configurar los parámetros principales de las fuentes a enlazarse para que el plugin genere un archivo de enlace automáticamente.

Independientemente de la forma en la que se ingrese las configuraciones al plugin, para realizar el proceso de *Linking* deben encontrarse los datos a enlazarse accesibles mediante SPARQL Endpoint. Debido al requerimiento mencionado esta actividad se recomienda realizar posteriormente a la etapa de publicación, aunque con motivo de conservar su orden respecto a la propuesta sea tratado en la etapa de generación. Entre otros de los aspectos que deben constar en las configuraciones se encuentran también los umbrales que utilizarán las métricas en el reconocimiento de enlaces de similitud entre recursos los cuales pueden definirse como *owl:SameAs*, *skos:closeMatch*, *skos:nearMatch*, *owl:differentFrom*, etc. así como en el proceso de validación.

Una de las características destacadas del desarrollo del plugin, es que cuenta con una versión de SILK modificada, que permite aparte de encontrar enlaces similares mediante métricas sintácticas, realizar validación de los enlaces encontrados mediante relación semántica de atributos relacionados.

Soporte a la etapa de publicación

Con los datos generados, se provee un acceso a estos mismos mediante su publicación en un triplestore. Para esta etapa se ha desarrollado un plugin (Fusek-

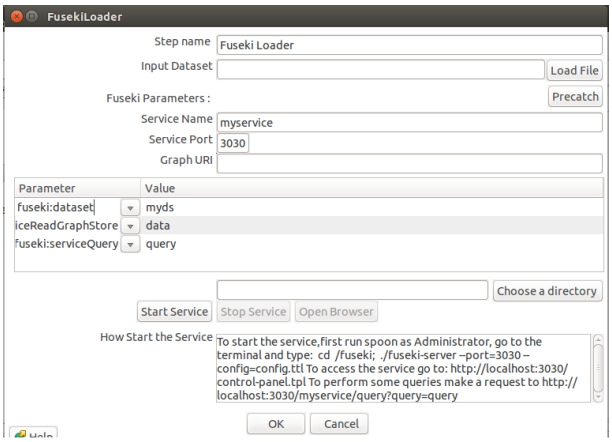

Figure 11: Configuración del plugin de publicación RDF (FusekiLoader)

iLoader) que permite el despliegue del triple store fuseki[47] que puede emplearse como medio de almacenamiento, consulta y de interacción con los datos gracias a su SPARQL endpoint. Entre las ventajas que posee dicho triple-store frente a otras tecnologías similares como Virtuoso[48] o Apache Marmota[49] esta su reducido tamaño y la capacidad de soportar algunas características necesarias como consultas federadas e indexación por texto.

En la figura 11 se puede apreciar la interfaz del plugin con la cual se puede acceder a las funcionalidades mencionadas mediante su configuración. Entre los campos que posee el plugin para su configuración se encuentran:

- Input Dataset: En este campo se define la ruta del archivo que se desea almacenar y desplegar en fuseki.
- Service Name: Campo opcional en donde se define el nombre que se le puede dar al servicio.
- Service Port: Puerto por el cual se puede acceder al servicio.
- Graph URI: Sirve para definir un grafo en el cual almacenar la información dentro del triple-store.
- Choose Directory : Permite definir la ruta de salida de la aplicación Fuseki.
- Grilla de configuración : En esta sección se pueden definir reglas que se aplicaran al almacenar los datos en el triple-store Fuseki.
  - Dataset: Nombre de la variable que representa al dataset a publicarse dentro de las configuraciones de Fuseki.
  - ServiceQuery/Upload/Update/ReadGraphStore/ReadWriteGraphStore: Se deben definir estas variables dependiendo del nivel de permisos que se desee habilitar en el endpoint. Por ejemplo Lectura, Escritura, Actualización, etc.

[47] https://jena.apache.org/documentation/fuseki2/
[48] http://virtuoso.openlinksw.com/
[49] http://marmotta.apache.org/

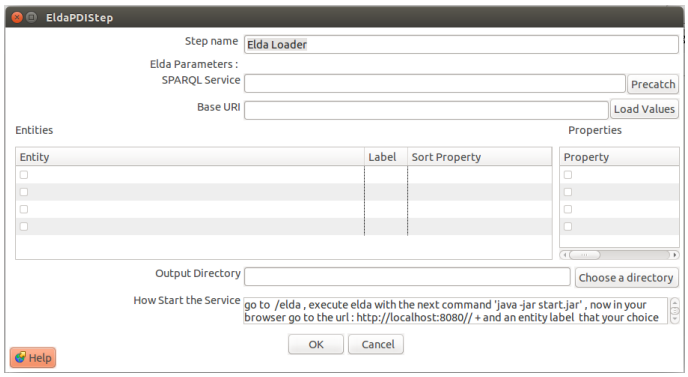

Figure 12: Configuración del plugin de explotación ELDA (Elda Loader)

– Lucene: Permite definir los campos que se indexaran por texto. Para definir los campos a indexar se debe colocar la URI completa y si hay varios separarlos por ";".

Soporte a la etapa de explotación

En el ámbito de explotación el framework ofrece la posibilidad de consumir y navegar por la información generada mediante un API SPARQL conocido como ELDA[50]. La configuración y los datos necesarios para el despliegue de esta interfaz web se encuentran dentro del mismo plugin y requiere únicamente su configuración mediante la interfaz presentada en la figura 12. Las configuraciones que se pueden realizar sobre esta interfaz son:

- Sparql Service: Endpoint Sparql del cual se tomara los datos.
- Base URI: URI del grafo en el cual se encuentran los datos que se desean explotar.
- Load Values: Carga los datos en la grilla de configuración.
- Output directory: Ubicación de salida del software ELDA configurado y listo para ser desplegado.
- Grillas de configuración: En esta sección se pueden seleccionar las elementos encontrados en el endpoint (entidades y propiedades) que se desean visualizar mediante ELDA. Además es posible renombrar en la columna "Label" los elementos seleccionados para facilitar su interpretación por el usuario.

Como resumen en la figura 13, se puede observar un esquema con cada uno de los plugins tratados y como en conjunto llegan a cubrir cada una de las diferentes etapas en la publicación de Linked Data.

[50]http://www.epimorphics.com/web/tools/elda.html

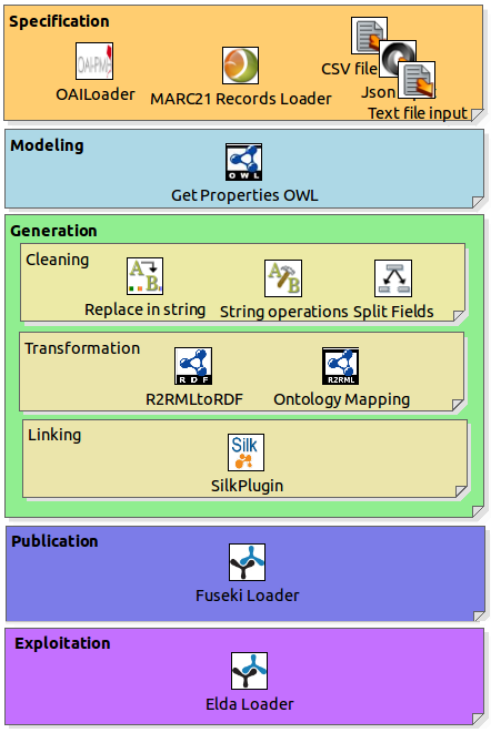

Figure 13: Plugins del framework que soportan LOD

IV. Casos de Uso

Para demostrar la validez del framework propuesto sobre diversos escenarios, se ha aplicado el proceso metodológico de publicación de datos enlazados sobre tres ámbitos diferentes: Repositorios digitales, recursos bibliotecarios y recursos de una base de datos de índole empresarial. Estos casos de uso se describen a continuación:

I. Casos de uso Repositorios Digitales

Una de las primeras pruebas realizadas con el framework desarrollado, fue la transformación y publicación como Linked Data de los datos pertenecientes a repositorios digitales Dspace. Específicamente los datos tratados pertenecen a los repositorios encontrados en los Dspaces de diversas universidades del Ecuador.

Especificación

Para la lectura de los datos de las fuentes Dspace se utilizó el estándar OAI-PMH, el cual es un mecanismo reconocido para cosecha de metadatos en el ámbito de repositorios digitales y que brinda la posibilidad de acceso a través de peticiones HTTP. Aunque Kettle posee un gran conjunto de plugins para la lectura de diferentes formatos, en este caso debido a características propias

de la llamadas al servicio fue requerida la generación de un plugin especializado, con el fin de asegurar la fiabilidad en la lectura y carga de estos tipos de información. Hay que destacar que en plataformas de generación LOD similares no dan soporte a este tipo de datos específico (OAI-PMH). Además, la incorporación de este formato dentro de dichas plataformas implicaría un arduo proceso de desarrollo debido a que en general sus arquitecturas son poco extensibles, una excepción es UnifiedViews en la cual se pudo haber desarrollado una DPU (Pluging) para OAI-PMH.

En la figura 14 se presenta un ejemplo de configuracion para el plugin mencionado para el repositorio de la universidad de Cuenca.

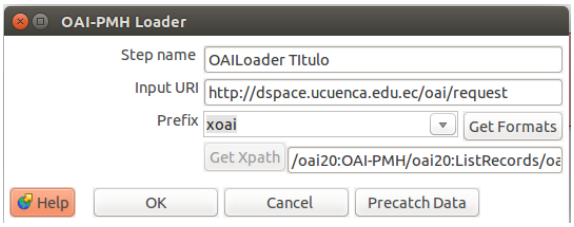

Figure 14: Configuración del plugin para carga de datos mediante OAI

El diseño de las URIs que fue aplicado sobre estos ejemplos poseen el siguiente formato:

http://DireccionServidorElda/FuentedeOrigen /TipodeRecurso/NombredelRecurso

Donde la primera parte de la URI esta conformada por la dirección del servicio del API ELDA, lo que ofrece la ventaja de ser direccionadas a una página de descripción del recurso cuando es accedido, funcionalidad que forma parte de la etapa de explotación. El resto de elementos presentes en la URI sirven como identificador del recurso y su fuente de procedencia.

Un ejemplo de una URI generada acerca de una persona que es autor o/y contribuyente del algún recurso se muestra a continuación:

`http://190.15.141.66:8899/ucuenca/contribuyente/ESPINOZA_MEJIA__JORGE_MAURICIO`

Modelamiento

Dentro de esta etapa, tomando en consideración que los recursos extraídos son de tipo bibliográficos, se han seleccionado vocabularios ontológicos que cubren este dominio, como **bibo**[51], **dcterms**[52]. Adicionalmente para datos que no cubren los vocabularios mencionados como personas se han seleccionado ontologías adicionales como **foaf**[53] que permite identificar semánticamente a los autores o contribuyentes de los recursos bibliográficos y **RDAA**[54] para relaciones especiales entre recursos (*a contribuido en*, *a creado* , *es parte de*). Para cargar los vocabularios mencionados dentro del flujo, se utilizó el plugin "GetPropertiesOWL" correspondiente a la etapa de modelamiento, mediante la declaración

[51] http://bibliontology.com/
[52] http://dublincore.org/
[53] http://xmlns.com/foaf/spec/
[54] http://www.rdaregistry.info/

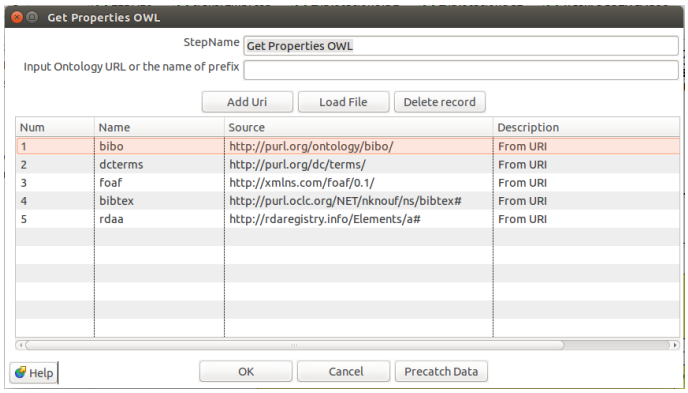

Figure 15: Configuración del plugin para la carga de ontologías.

de los prefijos de las ontologías seleccionadas. Respecto a las herramientas de LOD disponibles, solamente la plataforma Optique y las herramientas de generación de RDF: RML Editor y Karma proveen mecanismos para la asistencia dentro del modelamiento ontológico de las fuentes de datos. Estas herramientas permiten pre cargar vocabularios y sugieren conceptos en el proceso de generación de *mappings*. No obstante, las demás herramientas ignoran esta etapa y requirieran de cambios significativos dentro de su diseño para la incorporación de esta funcionalidad.

La configuración del plugin se puede apreciar en la figura 15.

Generación

Como primer paso dentro de la etapa de generación fue necesario realizar la revisión y análisis del estado de las fuentes. Dicha actividad se realizó con el objetivo de asegurar la calidad de los datos y poder ejecutar las correcciones necesarias previas al proceso de transformación. Una vez reconocidos los problemas se desarrolló flujos de limpieza utilizando para esto varios de los plugins nativos de la herramienta Kettle de Pentaho como se puede ver en la figura 16. Entre algunos de los problemas más comunes encontrados en las fuentes están:

Datos adicionales no correspondientes a los nombre propios de autores, por ejemplo:

- Recalde Moreno, Dr. Celso
- Torres Arroba, Fernando Javier Ph.D

Dos nombres pertenecientes a diferentes autores en un solo campo:

- Castro Muñoz, Celia María;Chang Gómez, José Vicente
- Rivas Tarazona, RossanaOsorio Llanos, Ana Cecilia

Nombres completos sin separador entre nombres y apellidos.

- Poma Salazar Mónica Eulalia
- Hidalgo David

Problemas en la codificación de algunos nombres

- Land??zuri V., Carolina
- Ca??izares Aguilar Aurelio Ernesto

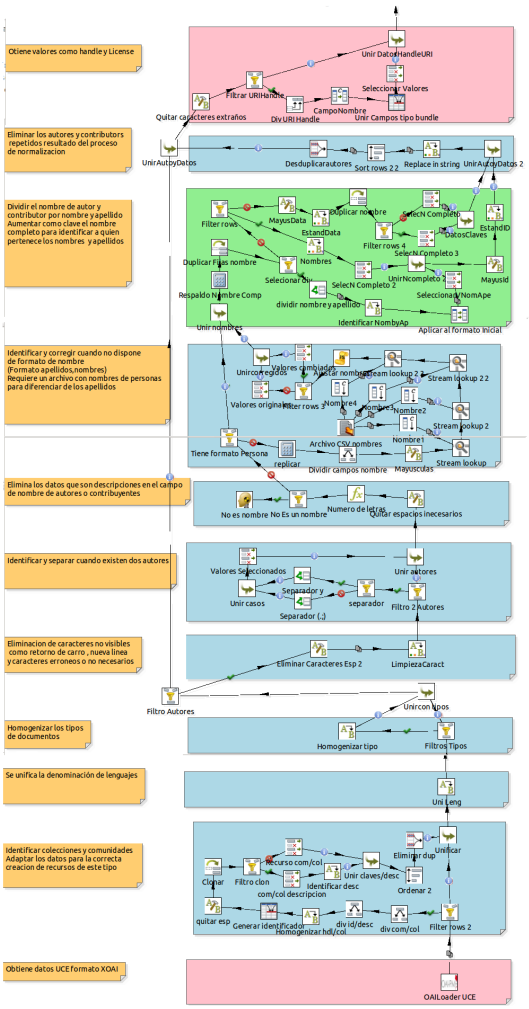

Figure 16: Parte del flujo de transformación en la limpieza de datos.

La posibilidad de definir flujos completos para la limpieza y normalización de los datos antes de la generación como RDF y además la reutilización de funcionalidades propias de un gestor ETL (Kettle) son características importantes del framework planteado. Esto debido a que las demás plataformas de LOD disponibles actualmente ignoran estas actividades o cuando más permiten realizar normalización de los datos a un nivel básico (plataforma Karma). Si bien plataformas como Unified Views podrían ser extendidas para realizar estas labores mediante el desarrollo de nuevas DPUs (Operaciones con cadenas de texto,

división de campos, etc.), esto requiriere de un gran esfuerzo de implementación debido principalmente a que los datos estan orientados a ser manipulados una vez se encuentran como RDF.

Posteriormente a la etapa de limpieza, para la trasformación de los datos disponibles de las fuentes Dspace a RDF, se definió un conjunto de reglas de mapeo basados en el flujo de datos disponible y las ontologías cargadas, utilizando para esto el plugin "Ontology Mapping" descrito en la sección anterior. Dado que los datos están cargados dentro del flujo, su definición consiste básicamente en reconocer campos como nombres de autores, título, resumen, etc. que contienen las fuentes, para definir su relación con su respectivo vocabulario semántico.

Los recursos o entidades que se identificaron durante esta etapa, fueron Personas que pueden representar autores o contribuyentes de alguna obra, Documentos que representan a los recursos bibliográficos y Colecciones a las cuales pertenecen dichos documentos. Un ejemplo de los mapeos realizados para los recursos bibliográficos se muestran en la tabla 2. En la parte izquierda de la tabla se define el campo encontrado en la fuente que se asociara con el vocabulario semántico presentado en la derecha.

Documentos Bibliográficos

Tipo: `http://purl.org/ontology/bibo/Document`

Campos	**Vocabulario semántico**
Date/Accessioned	`dcterms:dateSubmitted`
Date/Available	`dcterms:available`
Date/issued	`dcterms:issued`
Abstract	`bibo:abstract`
Provenance	`dcterms:provenance`
Subject	`dcterms:subject`
Title	`dcterms:title`
Language	`dcterms:language`
License	`dcterms:license`
URI	`bibo:uri`
Handle	`bibo:handle`

Table 2: Mapeo de las propiedades de un documento

El plugin con los mapeos definidos para todos los recursos se puede ver en la figura 17.

Publicación

Para la publicación de los datos obtenidos como documentos RDF en la etapa de trasformación, se configuró el plugin "Fuseki Loader" para cada repositorio fuente. Las configuraciones que se realizaron fueron la definición de grafos para cada endpoint, permisos exclusivos lectura para precautelar modificaciones no autorizadas y la indexación de las propiedades más relevantes en cada recurso, como Título, Asbtract, Nombre de Autores, Nombres de colecciones, etc. que se utiliza en la etapa de explotación. Las configuraciones mencionadas se puede apreciar en la figura 18.

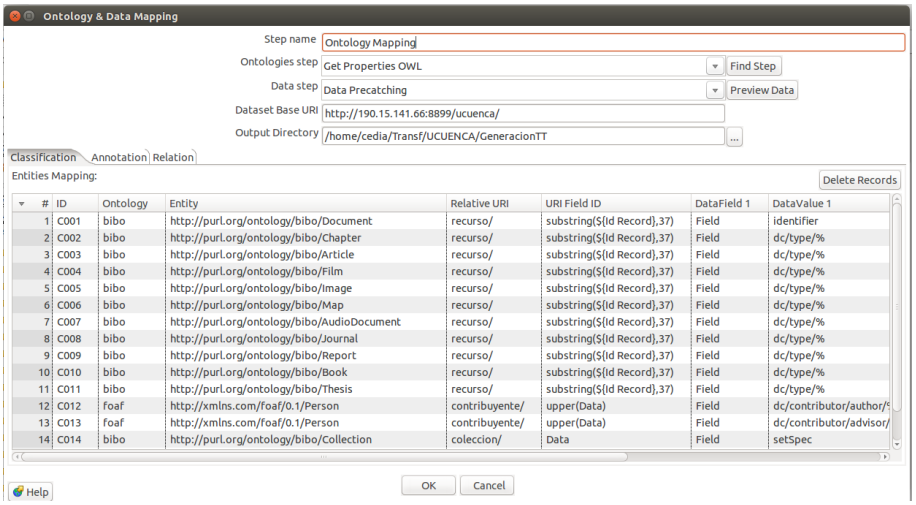

Figure 17: Definicion de mapping semántico para la universidad de Cuenca.

Enlace de datos (Linking)

El proceso de Linking como se ha definido anteriormente se ha realizado despues de la etapa de publicación, aprovechando de esta manera los datos disponibles en los endpoints que requiere el plugin desarrollado. Para la configuración del proceso de enlace se ha decidido usar las configuraciones por defecto, ingresando únicamente las direcciones sparql de los endpoints a enlazar, sus grafos y los umbrales de comparación y validación, como se muestra en la figura 19. Como ejemplo se ha realizado el proceso de Linking para los datos obtenidos a partir del repositorio de la universidad de Cuenca y Cedia.

Los autores encontrados y validados por el plugin de enlace de datos entre los repositorios de ejemplo se resumen en la tabla 3.

Universidad de Cuenca	**Cedia**
Delgado Suconota, María Fernanda	Delgado, María Fernanda
Illescas Riera, Raquel Guadalupe	Illescas Riera, Raquel
Ortíz Segarra, José Ignacio	Ortíz Segarra, José
Andrade, Gabriela	Andrade, Gabriela
Espinoza, Mauricio	Espinoza, Mauricio
Saquicela, Víctor	Saquicela, Víctor

Table 3: Resultados del proceso de *Linking* entre Universidad de Cuenca y CEDIA

Explotación

Para la explotación de los datos pertenecientes a los Dspace se configuró y utilizó el plugin “Elda Loader”, el cual permite generar una instancia de ELDA

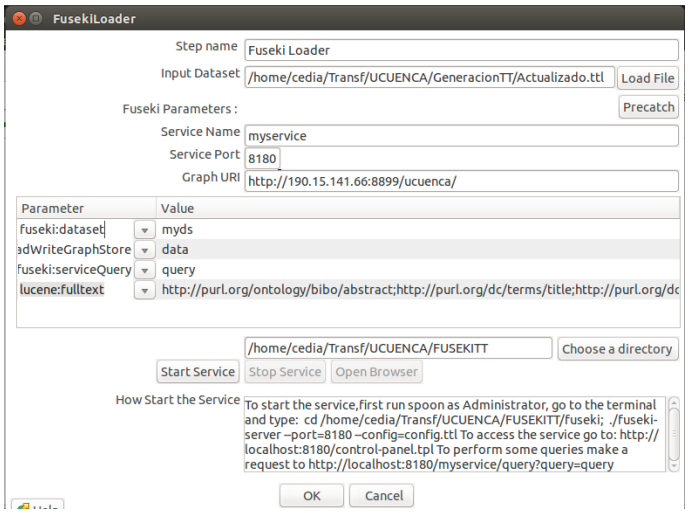

Figure 18: Configuraciones para la publicación de los datos de la universidad de Cuenca.

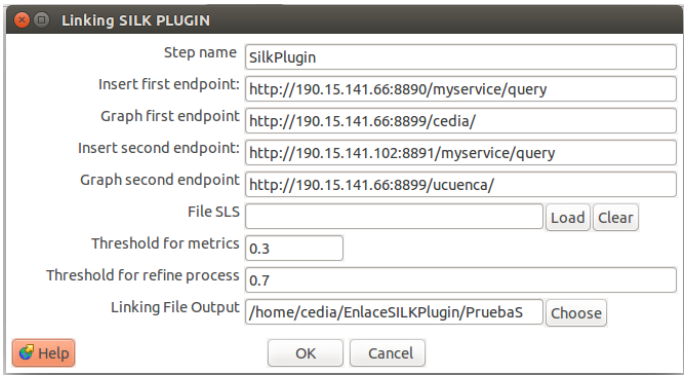

Figure 19: Configuración de ejemplo dentro del plugin SILK

para visualizar los recursos de forma amigable al usuario como se presenta en la figura 20 . Al igual que en la etapa de publicación se generó una configuración para cada fuente Dspace, con la diferencia que al final todos los archivos fueron concatenados en uno solo y levantados en un solo servicio ELDA con el objetivo de optimizar recursos.

Como otra forma de aprovechar los recursos y el indexado sobre los datos definido, se realizo un buscador de recursos para faciltar la localización de los información de los Dspaces frente a los usuarios, como se puede ver en la figura 21.

II. Casos de Uso Bibliotecas

De forma similar a los procesos de publicación de datos enlazados de repositorios Dspace, se procedió a utilizar el framework sobre datos bibliotecarios para probar su validez en este ámbito. Los datos de la biblioteca procesados en este

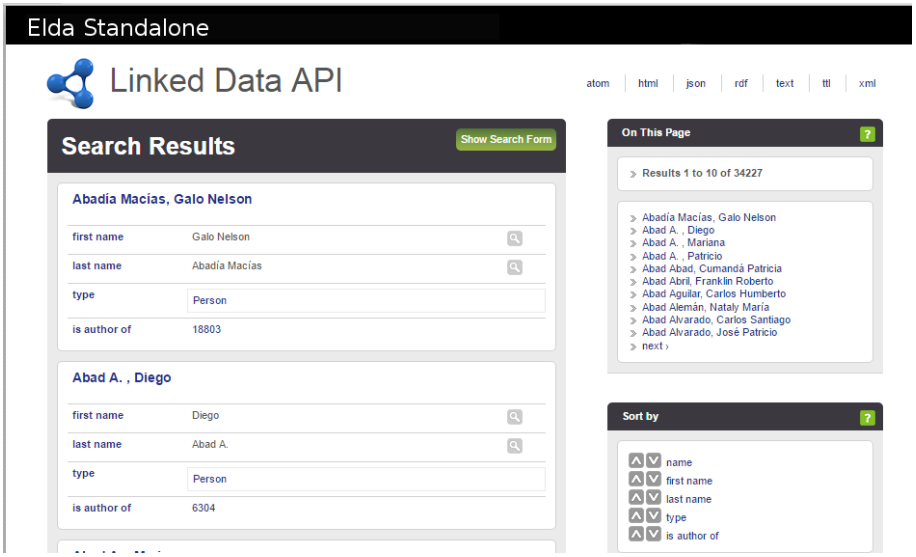

Figure 20: Interfaz web Elda para visualización de Personas

caso de uso correspondieron a los manejados por la Biblioteca Juan Bautista Vásquez[55] en su base de datos. Dado que el sistema que utiliza la biblioteca no tiene un servicio específico para exportación y cosecha de sus datos se requirió un procesamiento previo para almacenarlos bajo el formato Marc 21.

Especificación

Los datos que se trataron en esta etapa como se ha mencionado anteriormente fueron tomados desde fichero con formato Marc 21 para una mejor estandarización. El formato Marc 21 es un formato de almacenamiento e intercambio bibliotecario ampliamente usado en este ámbito por lo que representa apropiadamente las necesidades de este caso.

Para la lectura de Marc 21 se utilizo el plugin desarrollado "Marc Input" y se configuro tal como se muestra en la figura 22. Entre los campos que se definieron se encuentran la ruta del archivo con extensión marc 21, los campos que se extraerán del archivo y si se requiere exportar los datos a XML. Para el diseño de las URIs se siguió el mismo formato que el utilizado en el caso de los Dspace.

Modelamiento

Para la etapa de modelamiento de recursos bibliotecarios se tomaron las mismas ontologías utilizadas para el caso de uso de los Dspace como **bibo** y **dcterms**. Se utilizaron las mismas ontologías debido a que en ambos casos cubren el vocabulario relacionado con recursos bibliográficos. De la misma forma para autores se ha utilizado la ontología **foaf** que permite cubrir el vocabulario entorno a las personas o entidades. La mayor diferencia detectada con el caso del Dspace con la extracción de información de fuentes bibliotecarias, es que las últimas poseen

[55]http://www.ucuenca.edu.ec/recursos-servicios/biblioteca/la-biblioteca

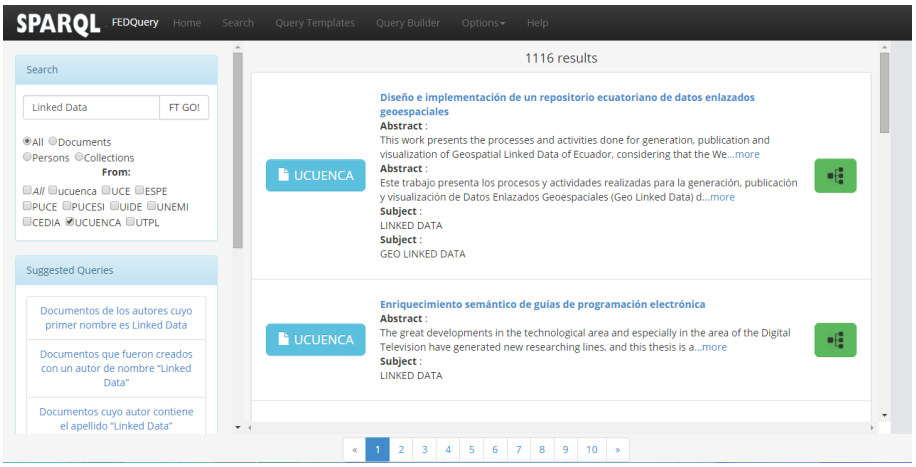

Figure 21: Buscador de recursos enlazados de repositorios Dspace

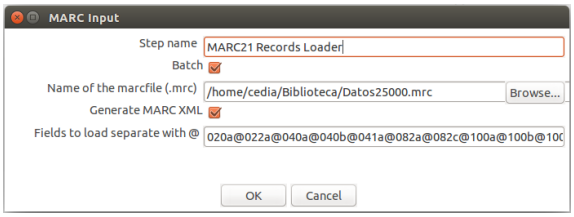

Figure 22: Panel de configuración de plugin Marc21

datos más detallados respecto a los recursos, como ubicación del recurso, número de páginas, editorial entre otros que son útiles para catalogación de los recursos tanto físicos como digitales. La carga de las ontologías mencionadas se realizó mediante el plugin (*GetPropertiesOWL*) configurandolo de forma similar a como se muestra figura 15.

Generación

En esta etapa, de forma similar al caso de uso anterior se ha comenzado con la revisión y análisis de las fuentes de datos. Una de las principales características que se puede apreciar en la fuente de tipo bibliográfico es que sus campos son identificados por códigos alfanuméricos propios de Marc 21, por lo que fue necesario mapear dichos valores por conceptos más intuitivos. Adicionalmente, a partir del análisis se detectó patrones de problemas muy similares a los de los recursos bibliográficos de los Dspace como:

Información adicional a los autores que no son propios de su nombre

- Beryl [et al.]

Varios Autores en un mismo campo de autor.

- Curtius, Ernst Robert. Frenk Alatorre, Margit, traductor

Valores de otros campos dentro de autores.

- 1572-1631

Los patrones de errores encontrados sirvieron para realizar las labores de limpieza mediante flujos y estandarizaciones previas a la transformación. Posteriormente dentro de la etapa de transformación se realizó el reconocimiento y el enlace de los campos más relevantes dentro de los recursos bibliotecarios con sus correspondencias con el vocabulario ontológico. Entre las clases que se reconocieron a partir de datos bibliotecarios se tiene de forma similar al caso de uso anterior, personas que representan a los autores principales o secundarios, los recursos bibliográficos que son los documentos catalogados por el centro bibliotecario, las colecciones a las que pertenecen los documentos y el centro catalogador o biblioteca. En la tabla 4 se presenta a más detalle las propiedades y campos que se mapearon para la generación de Linked Data a partir de recursos bibliográficos.

Campos Marc 21	**Descripción**	**Vocabulario semántico**
20 a	ISBN	`bibo:isbn`
41 a	Lenguaje	`dcterms:language`
245 a	Título	`dcterms:title`
250 a	Edición	`bibo:edition`
260 b	Editor	`bibo:editor`
300 a	Número de Paginas	`bibo:numPages`
362 a	Fecha de publicación	`dcterms:issued`
520 a	Resumen	`bibo:abstract`
653 a	Descriptores	`dcterms:subject`
856 u	Ubicación URI	`bibo:uri`
900 a	Número de Volumen	`bibo:volume`
900 o	Localización	`dcterms:Location`

Table 4: Mapeo de propiedades de recursos bibliotecarios

Publicación

Para la publicación de los datos se generó una instancia de fuseki mediante el plugin *FUSEKI LOADER* al igual que en el caso de repositorios Dspace.

Enlace de datos (Linking)

Al reconocer que existe una estrecha relación entre la biblioteca Juan Bautista Vásquez tratada y la universidad de Cuenca, debido a que la biblioteca pertenece a la universidad, se quizo comprobar si existen enlaces de autores entre sus repositorios. Para comprobar los enlaces existentes y validarlos como en el caso de uso anterior, se utilizo el plugin desarrollado SILK. Como resultado del enlace se encontraron 41 enlaces, una muestra de los enlaces mencionados se presentan en la tabla 5.

Explotación

Dentro de la etapa de explotación se utilizó el plugin *ELDA LOADER*, para generar un servicio que permita la visualización y búsqueda a través de los datos

Biblioteca JVB	Universida de Cuenca
de la Charles, Marie	Charles, Marie de la
Villaroel, Gaspar de	Villaroel, Gaspar de
Rocafuerte, Vicente	Rocafuerte, Vicente
Quezada Q., Fausto C	Quezada Q., Fausto C
Barrera A., T	Barrera A., T

Table 5: Muestra del resultado del proceso de *Linking* entre la biblioteca JVB y Universidad de Cuenca

generados de fuentes bibliotecarias. En la figura 23 se muestra la interfaz que el API ELDA permite visualizar, en este caso presenta recursos bibliotecarios que se han generado como LOD.

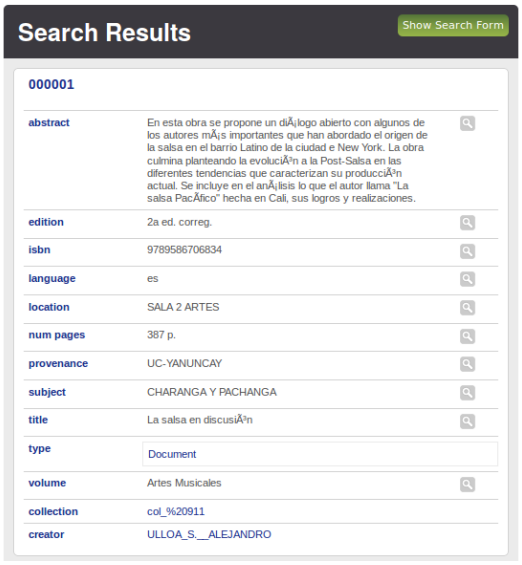

Figure 23: Interfaz API ELDA que presenta recursos bibliográficos

III. Base de datos de una empresa

Un último caso de uso que se ha tratado para mostrar la validez de la plataforma sobre múltiples ámbitos, es la de publicación de LOD a partir de una base de datos. Este caso se ha considerado relevante pues mucha de la información que puede requerirse ser convertida bajo los principios de LOD generalmente pueden venir de una fuente de almacenamiento común como lo son las base de datos. La base de datos utilizada en este caso se encuentra en MySql[56] y ha sido generada como muestra, basandose en algunos de los datos que puede tener una empresa con respecto a sus empleados.

[56]https://www.mysql.com/

Especificación

El esquema tomado para la aplicación de este caso de uso se puede ver en la figura 24. En este esquema constan de forma simplificada las tablas con los datos acerca de empleados, sucursales y empresas que pueden ser parte de los datos comunes contenidos en aplicaciones empresariales.

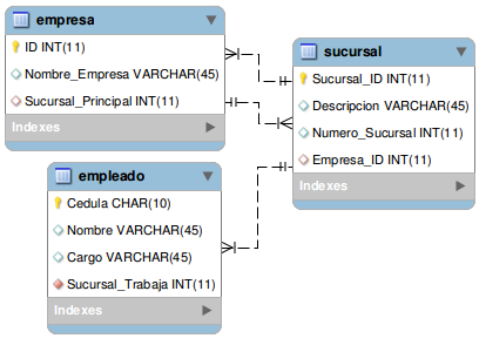

Figure 24: Diagrama con el esquema de base de datos

Para la obtención de los datos contenidos en la base de datos se utilizó uno de los plugins propios de la plataforma llamado "*Table Input*", el cual permite la consulta de tablas de una base de datos mediante la generación de una conexión y que puede ser refinadas con una consulta SQL. Como ejemplo en la figura 25 se muestra la configuración realizada para la obtención de los datos de la tabla empleados.

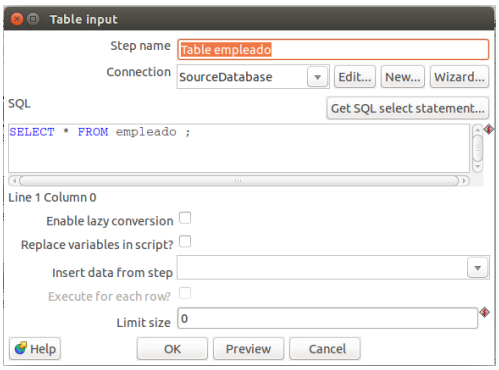

Figure 25: Configuración del plugin sobre la tabla empleados

El modelamiento de la URI de forma similar a como se ha venido realizando contiene el siguiente formato dependiendo del tipo de recurso:

`http://{Dirección_Elda}/empresa/{Persona|Organización|Sucursal}/{Cédula|Código}`

Modelamiento

Los datos obtenidos en este caso están enfocados al ámbito empresarial, por lo que las ontologías han sido seleccionadas según este requerimiento. Entre las

ontologías seleccionadas se encuentran **foaf** que como se ha mencionado cubre el ámbito relacionado con descripción de personas e instituciones y **Schema.org**[57], la cual es una ontología muy completa y de ámbito general enfocado principalmente para descripción de páginas web pero se ha comprobado se extiende muy bien en el ámbito empresarial y de empleados que se requería en este caso de uso. Las ontologías han sido cargadas como en casos anteriores mediante el plugin *GetPropertiesOWL*.

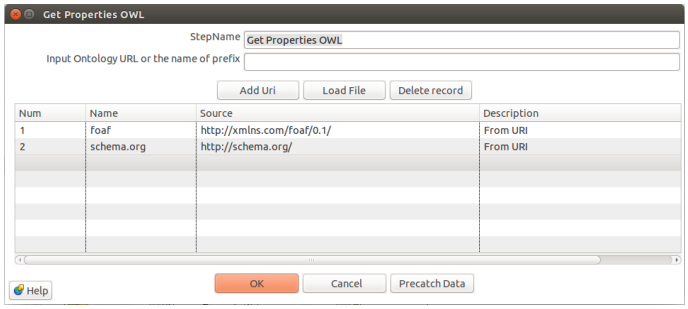

Figure 26: Normalización de la tabla empleados

Generación

Para este caso de uso, la etapa de generación se ha simplificado en gran medida pues al conocerse el estado de la fuente de datos, se ha evitado realizar actividades como análisis y limpieza de datos que en casos anteriores se han realizado. Sin embargo, labores de estandarización y adaptación de los datos se han mantenido, pues es necesario convertir las tablas de las fuentes de datos en tripletas para su correcta manipulación por el framework. Una vez que los datos han sido adaptados se ha realizado el proceso de mapping correspondiente mediante el plugin *Ontology Mapping*, siguiendo las relaciones definidas en la tablas 6, 7, 8 para los diferentes recursos encontrados.

Personas Tipo : http://xmlns.com/foaf/0.1/Person

Campo	**Vocabulario Semántico**
Nombre	foaf:name
Cargo	schema:jobTitle
Sucursal	schema:memberOf

Table 6: Relaciones definidas entre los datos de empleados y el vocabulario.

Sucursal Tipo : http://schema.org/Organization

Campo	**Vocabulario Semántico**
Descripción	foaf:name
Teléfono	foaf:phone
Empresa	schema:parentOrganization

Table 7: Relaciones definidas entre los datos de sucursales y el vocabulario.

[57]http://schema.org/

Empresa Tipo : http://schema.org/Organization

Campo	Vocabulario Semántico
Nombre	foaf:name

Table 8: Relaciones definidas entre los datos de la empresa y el vocabulario.

Publicación

El proceso para esta etapa ha sido el mismo que en casos anteriores, configurando el plugin FUSEKI LOADER para cargar y proveer un Sparql Endpoint del archivo RDF generado.

Enlace de datos (Linking)

Para este caso de uso no se han encontrado fuentes de datos comunes con las cuales realizar enlace de datos, sin embargo en caso de que existieran podría usarse el plugin de enlace de datos SILK tal como en casos anteriores. En el ámbito empresarial con facilidades como frameworks para generación de RDF, iran aumentando las posibilidad de encontrar fuentes de datos con los cuales pueda enlazarse una empresa, como proveedores o consumidores que también dispusieran de sus datos en RDF.

Explotación

Al igual que en casos de uso anteriores se utilizó ELDA LOADER para generar una instancia de ELDA que permite tener una visualización más amigable de los datos generados con la plataforma.

V. Conclusión

En este trabajo se presenta una plataforma para la generación de LOD, que sigue los lineamientos metodológicos existentes en el estado del arte para el proceso de publicación de LOD. La plataforma ha sido implementada sobre el popular gestor de procesos ETL Pentaho Data Integration (Kettle) y reutiliza la mayoría de sus componentes disponibles. Todos estos elementos se han integrado en un ambiente común que agilita la generación de soluciones LOD completas. Adicionalmente, los componentes/plugins de la plataforma presentada pueden ser usados de forma independiente para cubrir necesidades específicas dentro del ciclo de vida de LOD. Todas estas características superan las limitaciones de los frameworks actuales que o cubren solo ciertas etapas de las metodologías, o sus componentes no pueden usarse de forma independiente.

Una de las características destacables del framework expuesto ádemas, se encuentra en su extensibilidad para soportar diversos dominios, aspecto que muy pocas plataformas se han orientado, ya que en su mayoria fueron diseñados para solucionar problemas específicos (integración a nivel empresarial, manipulación de datos enlazados, etc). Para demostrar estas capacidades se ha presentado 3 casos de uso sobre 3 dominios diferentes que llegan a demostrar su aplicación practica sobre problemas reales y marcan las pautas para llegar a abarcar dominios similares.

El trabajo futuro se centrará en expandir el ámbito de la plataforma a nuevos dominios de la información, a través del desarrollo de nuevos plugins. Adicionalmente, se pretende desarrollar un módulo para recomendación de ontologías y conceptos dentro de la fase de mapeo semántico de datos. De igual manera, se agregarán componentes para el soporte nativo de tecnologías semánticas dentro del gestor ETL, específicamente: lectura/manipulación/almacenamiento de RDF-SPARQL.

Finalmente, la plataforma se probará con casos de uso reales adicionales con el objetivo de aprovechar el conocimiento que proveen las particularidades de cada caso. Es importante comentar el éxito de la plataforma presentada reside en la constante adquisición de experiencia de los aplicaciones realizadas y surge de la necesidad de aplicación del proceso de publicación de LOD sobre casos reales.

Bibliography

[1] Tim Berners-Lee. Linked data, 2006.

[2] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[3] Souripriya Das, Seema Sundara, and Richard Cyganiak. R2rml: Rdb to rdf mapping language (w3c working draft), 2011.

[4] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In *Proceedings of the 7th Workshop on Linked Data on the Web*, April 2014.

[5] Pieter Heyvaert, Anastasia Dimou, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, Erik Mannens, and Rik Van de Walle. RMLEditor: a graph-based mapping editor for Linked Data mappings. In Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, May 2016.

[6] Evgeny Kharlamov, Ernesto Jiménez-Ruiz, Dmitriy Zheleznyakov, Dimitris Bilidas, Martin Giese, Peter Haase, Ian Horrocks, Herald Kllapi, Manolis Koubarakis, Özgür Özçep, Mariano Rodríguez-Muro, Riccardo Rosati, Michael Schmidt, Rudolf Schlatte, Ahmet Soylu, and Arild Waaler. Optique: Towards obda systems for industry. In Philipp Cimiano, Miriam Fernández, Vanessa Lopez, Stefan Schlobach, and Johanna Völker, editors, *The Semantic Web: ESWC 2013 Satellite Events*, volume 7955 of *Lecture Notes in Computer Science*, pages 125–140. Springer Berlin Heidelberg, 2013.

[7] Tomáš Knap, Maria Kukhar, Bohuslav Macháč, Petr Škoda, Jiří Tomeš, and Ján Vojt. Unifiedviews: An etl framework for sustainable rdf data processing. In Valentina Presutti, Eva Blomqvist, Raphael Troncy, Harald Sack, Ioannis Papadakis, and Anna Tordai, editors, *The Semantic Web: ESWC 2014 Satellite Events*, volume 8798 of *Lecture Notes in Computer Science*, pages 379–383. Springer International Publishing, 2014.

[8] Craig A. Knoblock, Pedro Szekely, Jose Luis Ambite, Shubham Gupta, Aman Goel, Maria Muslea, Kristina Lerman, Mohsen Taheriyan, and Parag Mallick. Semi-automatically mapping structured sources into the semantic

web. In *Proceedings of the Extended Semantic Web Conference*, Crete, Greece, 2012.

[9] Christoph Pinkel, Andreas Schwarte, Johannes Trame, Andriy Nikolov, AnaSasa Bastinos, and Tobias Zeuch. Dataops: Seamless end-to-end anything-to-rdf data integration. In Fabien Gandon, Christophe Guéret, Serena Villata, John Breslin, Catherine Faron-Zucker, and Antoine Zimmermann, editors, *The Semantic Web: ESWC 2015 Satellite Events*, volume 9341 of *Lecture Notes in Computer Science*, pages 123–127. Springer International Publishing, 2015.

[10] Andreas Schultz, Andrea Matteini, Robert Isele, Christian Bizer, and Christian Becker. Ldif : Linked data integration framework. 2011.

[11] Bert Van Nuffelen, Valentina Janev, Michael Martin, Vuk Mijovic, and Sebastian Tramp. Supporting the linked data life cycle using an integrated tool stack. In Sören Auer, Volha Bryl, and Sebastian Tramp, editors, *Linked Open Data – Creating Knowledge Out of Interlinked Data*, volume 8661 of *Lecture Notes in Computer Science*, pages 108–129. Springer International Publishing, 2014.

[12] Bert Van Nuffelen, Valentina Janev, Michael Martin, Vuk Mijovic, and Sebastian Tramp. Supporting the linked data life cycle using an integrated tool stack. In Sören Auer, Volha Bryl, and Sebastian Tramp, editors, *Linked Open Data – Creating Knowledge Out of Interlinked Data*, volume 8661 of *Lecture Notes in Computer Science*, pages 108–129. Springer International Publishing, 2014.

[13] Boris Villazón-Terrazas, Luis M Vilches-Blázquez, Oscar Corcho, and Asunción Gómez-Pérez. Methodological guidelines for publishing government linked data. In *Linking government data*, pages 27–49. Springer, 2011.