ESCUELA POLITÉCNICA NACIONAL

FACULTAD INGENIERÍA DE SISTEMAS

UNIDAD DE TITULACIÓN

DESARROLLO DE UN FRAMEWORK QUE IDENTIFICA, DESCRIBE Y ORGANIZA RECURSOS EDUCATIVOS DISPONIBLES EN PLATAFORMAS WEB DE UNIVERSIDADES MEDIANTE MINERÍA DE DATOS

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL GRADO DE MAGISTER EN SISTEMAS DE INFORMACIÓN

LUIS EDUARDO ROSERO CORREA

luis.rosero01@epn.edu.ec

DIRECTORA: Lorena Katherine Recalde Cerda

lorena.recalde@epn.edu.ec

CODIRECTORA: Rosa del Carmen Navarrete Rueda

rosa.navarrete@epn.edu.ec

Quito, septiembre 2021

APROBACIÓN DEL DIRECTOR

Como directora del trabajo de titulación DESARROLLO DE UN FRAMEWORK QUE IDENTIFICA, DESCRIBE Y ORGANIZA RECURSOS EDUCATIVOS DISPONIBLES EN PLATAFORMAS WEB DE UNIVERSIDADES MEDIANTE MINERÍA DE DATOS desarrollado por el señor Luis Eduardo Rosero Correa, con cédula de ciudadanía 1723621080, estudiante de la Maestría en Sistemas de Información Mención Inteligencia de Negocios y Analítica de Datos Masivos, habiendo supervisado la realización de este trabajo y realizado las correcciones correspondientes, doy por aprobada la redacción final del documento escrito para que prosiga con los trámites correspondientes a la sustentación de la Defensa Oral.

Dra. Lorena Katherine Recalde Cerda

DIRECTORA DEL PROYECTO

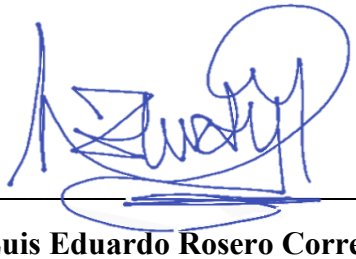
Dra. Rosa del Carmen Navarrete Rueda

CODIRECTORA DEL PROYECTO

DECLARACIÓN DE AUTORÍA

Yo, Luis Eduardo Rosero Correa, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

La Escuela Politécnica Nacional puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.



Luis Eduardo Rosero Correa

DEDICATORIA

A mi madre por darme la vida y la oportunidad de vivirla lleno de su inmenso amor, a mis hermanos por acompañarme siempre en cada paso, cada caída y cada nuevo comienzo, a mi hermana por permitirme ser su guía y motivarme siempre a ser mejor, a Rocío por su amor, apoyo y paciencia que me fortalecen cuando lo necesito.

Luis Eduardo Rosero Correa

AGRADECIMIENTO

Agradezco profundamente a la Dra. Lorena Katherine Recalde Cerda y la Dra. Rosa del Carmen Navarrete Rueda por todo cuanto supieron enseñarme, por la confianza, el tiempo, el apoyo y los valiosos aportes para este trabajo que, además de oportunos fueron faros que iluminaron el camino correcto a seguir, faros de una luz tan clara que solo puede nacer de profesionales llenas de sabiduría y ética como lo son ellas.

Luis Eduardo Rosero Correa

ÍNDICE DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE FRAGMENTOS

RESUMEN

Esta tesis de Maestría tiene la intención de analizar el grado de adopción de la web semántica con el vocabulario Schema.org y el formato JSON-LD, por parte de las universidades, para publicar y describir recursos educativos en sus sitios Web. Se basa en la fusión de las metodologías: Design Science Research (DSR) para guiar la creación de un artefacto y CRISP-DM para abordar el proceso de minería de datos. Puesto que estas metodologías no son excluyentes, se logró relacionar las diversas fases que las componen [1]. El resultado de este trabajo es un framework que consta de once componentes distribuidos en tres capas bien definidas: capa de acceso a datos, capa de servicio y capa de aplicación. Los componentes se desarrollan a lo largo de las fases combinadas de las dos metodologías y abordan el proceso de minería de datos desde la descarga hasta el análisis final de los datos para la obtención de información. En los resultados se puede evidenciar que las universidades sí utilizan el vocabulario Schema y el formato JSON-LD para publicar recursos, aunque no en la medida que se esperaría. Además, se notó la ausencia de propiedades del vocabulario Schema que permiten definir de forma específica a un recurso en un contexto educativo. La parte final de este trabajo propone algunas acciones para extender la funcionalidad del framework hacia otros contextos diferentes del educativo, así como también otros vocabularios o formatos.

Palabras clave: Marcado Estructurado Embebido. Recursos Educativos. Schema.org. JSON-LD. Raspado Web. CRISP-DM. DSR.

ABSTRACT

This Master's thesis intends to analyze the degree of adoption of the semantic web with the Schema.org vocabulary and the JSON-LD format by universities to publish and describe educational resources on their websites. It is based on the fusion of the methodologies: Design Science Research (DSR) to guide the creation of an artifact and CRISP-DM to address the data mining process. Since these methodologies are not exclusive, it was possible to relate the various phases that compose them [1]. The result of this work is a framework that consists of eleven components distributed in three well-defined layers: data access layer, service layer and application layer. The components are developed throughout the combined phases of the two methodologies and address the data mining process from download to final analysis of the data to obtain information. The results show that universities do use the Schema vocabulary and the JSON-LD format to publish resources, although not to the extent that would be expected. In addition, the absence of properties of the Schema vocabulary that allow a specific definition of a resource in an educational context was noted. The final part of this work proposes some actions to extend the functionality of the framework to other contexts different from the educational one, as well as other vocabularies or formats.

Keywords: Embedded Structured Markup. Educational resources. Schema.org. JSON-LD. Web Scraping. CRISP-DM. DSR.

1 INTRODUCCIÓN

1.1 Planteamiento del problema

La evolución de la tecnología se produce a ritmos acelerados, es así que tanto las tecnologías de la información, herramientas e incluso los paradigmas tecnológicos cambian de manera constante trayendo consigo tanto beneficios, como retos a la sociedad [2]. Estos hechos se pudieron evidenciar de forma clara con los acontecimientos suscitados desde el inicio de la pandemia del coronavirus. A nivel mundial, instituciones y personas tuvieron que adaptarse al cambio de forma repentina, para que aquellas tareas que se ejecutaban presencialmente se realizaran de forma remota. Un caso específico es el de los sistemas educativos, los cuales han enfrentado uno de sus mayores desafíos frente al COVID-19 puesto que, se tuvo que cambiar abruptamente de la instrucción convencional impartida en las aulas de clase a la enseñanza en línea y la educación virtual [3].

Para solventar las necesidades que surgieron dentro del sistema educativo, la opción de la educación virtual o e-learning se ha convertido en una modalidad esencial [4], lo cual implica el uso de plataformas educativas en la Web para facilitar así el acceso a los recursos educativos. El problema que se presenta con los recursos educativos en la Web es que estos no resultan fáciles de encontrar por parte de los usuarios cuando realizan búsquedas en Internet mediante motores de búsqueda como Google, Yahoo, Yandex o Bing. Esto, como se presenta en [5], se debe a que los recursos Web no se crean pensando en la visibilidad o facilidad de ser encontrados, por lo que no se agrega Metadata referente a dichos recursos que simplifique las tareas de búsqueda.

Una forma de mejorar la experiencia de los usuarios que navegan en la Web es entregándoles contenido más apropiado como respuesta a sus búsquedas [6], [7]. Esto se puede lograr mediante incrustaciones de marcado estructurado semántico en el código HTML de las páginas Web ya que los motores de búsqueda utilizan este marcado estructurado para "comprender" los recursos que se publican y así mejorar la precisión de los resultados de las búsquedas [5]; y por ejemplo en el caso de Google, presentar este contenido del marcado estructurado semántico como fragmentos enriquecidos o Rich Snnipets en el resultado de las búsquedas de los usuarios [8].

El uso de marcado estructurado semántico se puede aplicar en diversas áreas, por ejemplo, en [9], el marcado estructurado se usa para integrar anuncios de portales de empleo mediante una representación consolidada de un esquema común, el cual se obtiene a partir de la descripción del esquema en formato JSON-LD que proporcionan los portales Web de empleo. En [10], con base en el marcado estructurado, se desarrolla una herramienta para presentar información de recursos Web como páginas HTML, documentos Word o archivos PDF de diferentes sitios Web, en forma de fragmentos enriquecidos de forma similar a los resultados que presenta Google como respuesta a las búsquedas de los usuarios.

Con respecto al uso de marcado estructurado, también se han realizado varios trabajos que tienen relación con el ámbito educativo; por ejemplo, en [11], se realiza el etiquetado de recursos educativos relacionados con Data Science mediante técnicas de Machine Learning para describir el contenido de éstos. En [6] se propone el uso de las tecnologías de la Web semántica, que están orientadas a desarrollar una Web inteligente en donde la información puede ser comprendida por

las máquinas, esto con el fin de mejorar la interoperabilidad entre los sistemas e-Learning como Moodle. En [2] se presenta un sistema para indexación semántica de documentos relacionados con la ciberseguridad haciendo uso de técnicas de procesamiento de lenguaje natural (NLP) con el propósito de facilitar el proceso de documentación del tópico de ciberseguridad. En [12], los autores proponen el desarrollo del prototipo de un sistema de recomendación de recursos educativos abiertos (REA) basado en los metadatos que estos contienen, para apoyar el desarrollo de habilidades de los estudiantes de acuerdo con la información real existente sobre el mercado laboral relacionado con la ciencia de datos.

Como se puede observar, existen trabajos que buscan hacer uso de las incrustaciones de marcado estructurado semántico con el propósito de mejorar la experiencia de los usuarios con resultados más acertados para sus búsquedas en Internet. Por otro lado, se puede notar que, de forma general, no existe un uso adecuado del marcado estructurado semántico en las páginas Web. Esto se puede evidenciar por ejemplo en [5], en donde se presenta un estudio sobre el uso de incrustaciones de marcado estructurado semántico para describir los recursos educativos mediante los formatos Microdata y JSON-LD, en un conjunto de datos de 2018 obtenido a partir de técnicas de Web Crawling. En este estudio los resultados muestran que el uso de marcado estructurado embebido para describir recursos Web no es una práctica muy común, especialmente en el ámbito educativo, aunque se debe tener en cuenta que el conjunto de datos correspondía a un contexto general [5].

El hecho de que no se haga un uso adecuado de marcado estructurado embebido semántico para la descripción de recursos educativos en la Web, resulta preocupante si se toma en cuenta que la educación se orienta cada vez más hacia la Web. Si a esto le sumamos la premura con la que la pandemia producto del COVID-19, obligó a profesores y alumnos a continuar con las actividades académicas en un formato virtual, el que los recursos educativos en la Web sean difíciles de encontrar mediante los motores de búsqueda más populares, nos da un indicio claro de que al publicar recursos educativos en la Web es primordial el uso adecuado de marcado estructurado embebido, por ejemplo, en formato JSON-LD.

Con estos antecedentes, el propósito de este proyecto es el desarrollo de un framework que permita realizar un análisis de los sitios web de las universidades que se encuentran en el top de ranking internacionales, centrado en el uso de incrustaciones de marcado estructurado semántico con la sintaxis JSON-LD y el vocabulario de Schema. Para lo cual se recopilará un listado a nivel mundial de las universidades que forman parte del top de ranking internacionales, para luego mediante técnicas de Web Scraping analizar los sitios Web en busca de recursos educativos basados en el marcado estructurado embebido. Finalmente, se utilizará técnicas de minería de datos para describir y organizar los recursos educativos obtenidos.

La contribución del presente trabajo de titulación es el análisis del uso de marcado estructurado embebido con el vocabulario Schema y el formato JSON-LD en los sitios Web de universidades, ya que hasta el momento los trabajos realizados no se enfocan específicamente en el ámbito educativo o no usan un conjunto de datos concreto dentro de este contexto. Se contribuye también con la especificación de un framework que permite realizar este tipo de análisis sobre el uso de marcado estructurado embebido desde una fase de recolección de datos hasta la obtención de resultados e indicadores sobre los datos. Adicionalmente, el trabajo aporta con un conjunto de

datos conformado por las páginas HTML de los sitios Web de las universidades que se puede utilizar para nuevos análisis.

1.2 Objetivo general

Desarrollar un framework que identifica, describe y organiza recursos educativos que presentan marcado estructurado embebido, disponibles en plataformas Web de universidades, mediante la aplicación de técnicas de minería de datos.

1.3 Objetivos específicos

- Realizar una revisión de literatura sobre trabajos enfocados en el marcado estructurado embebido y su relación con los recursos educativos.
- Recolectar a nivel global una lista de las universidades ubicadas en el top de ranking internacionales, que publican recursos educativos y los sitios web correspondientes.
- Diseñar una estrategia de Web Scraping que permita indexar y analizar los sitios web de los recursos educativos con base en el marcado estructurado embebido.
- Emplear técnicas de minería de texto para describir y organizar los recursos educativos obtenidos.
- Evaluar el framework propuesto y su aplicabilidad en cuanto a la identificación, descripción y organización de recursos educativos en el dataset generado por el Web Scraping.

1.4 Alcance

Debido a que no existen estudios previos referentes al uso de marcado estructurado embebido, que manejen el vocabulario Schema junto con el formato JSON-LD en el ámbito educativo, este trabajo propone una solución que abarca desde la recolección de los datos hasta el análisis de resultados, todo esto gestionado mediante un framework, que es el objetivo principal del proyecto.

Algunos de los inconvenientes a superar son: obtener datos que se puedan utilizar para el desarrollo del proyecto, gestionar la descarga y el almacenamiento de los datos, desarrollar programas específicos para el tratamiento y manipulación de los datos, planificar y distribuir de manera correcta las actividades del proyecto para cumplir con el cronograma de trabajo.

Debido a estos inconvenientes, se debe limitar las tareas que se llevarán a cabo para la ejecución del proyecto, las mismas que se detallan a continuación:

El análisis se realizará sobre los sitios Web de universidades que se encuentran en el top de ranking internacionales. Como a nivel mundial existe una vasta cantidad de universidades, se seleccionará las 100 primeras universidades del top publicado por la página Top Universities[1] para el año 2021.

Los datos se descargarán mediante técnicas de Web Scraping, para esto se utilizará Python como lenguaje de programación, el framework Scrapy para gestionar el acceso y descarga de las páginas Web y la librería Extruct para facilitar la extracción de datos a partir de las páginas Web.

[1] https://www.topuniversities.com/university-rankings/world-university-rankings/2021

El almacenamiento de los datos extraídos se realizará en MongoDB, una base de datos orientada a documentos, mientras que el código HTML de las páginas Web se almacenará en disco en forma de texto plano. Estos elementos, tanto los de la base de datos como los archivos HTML, serán compartidos con el propósito de cumplir el objetivo de crear un nuevo conjunto de datos relacionado al ámbito educativo.

Las actividades que no se consideran para el desarrollo del proyecto son las siguientes:

El análisis de marcado estructurado embebido se centrará solamente en el uso del vocabulario Schema junto con el formato JSON-LD, dado que este es el recomendado por el W3C[2] (World Wide Web Consortium).

El enfoque del proyecto es en el contexto educativo, por tal razón no se extenderá el análisis hacia otras áreas o contextos.

El objetivo del proyecto es el desarrollo de un framework, por lo que no se plantea la creación de un modelo para el procesamiento de los datos en términos de analítica predictiva.

No se creará un almacén de datos (datawarehouse) debido a que el análisis de los datos se realizará mediante consultas directas a la base de datos MongoDB. El repositorio se mantendrá con fines de consultas analíticas mas no operaciones transaccionales de tipo update/delete.

1.5 Marco teórico

1.5.1 Marcado estructurado

En los últimos años se han realizado grandes avances para la construcción de una Web de datos (Web of data) [8]. Un ejemplo de Web de datos es la Web semántica que consiste en enriquecer datos de la Web con datos semánticos para proporcionarles un significado capaz de ser entendido por los computadores [13]. La web semántica de acuerdo con [6] se compone de estándares, estructuras de datos y software. Los tres elementos trabajan en conjunto y mejoran la experiencia de navegación del usuario en la Web. La Web semántica se implementa mediante el uso de marcado estructurado embebido que se refiere a realizar incrustaciones semánticas a través de etiquetas en el código HTML de páginas web convencionales [5], [8].

El uso de marcado estructurado embebido para realizar anotaciones Web semánticas se ha incrementado en los últimos años. Esto se debe a que es una estrategia factible que se puede utilizar para mejorar la información que se muestra en los resultados de búsqueda [5], [7]. El consumidor principal del marcado estructurado embebido son las máquinas y no los humanos, es por eso que un usuario no visualiza el marcado estructurado de las páginas Web [13]; por el contrario, las máquinas mediante los motores de búsqueda como Google, Yahoo, Bing, Yandex, etc. utilizan el marcado estructurado embebido primero para clasificar e interpretar correctamente el contenido del sitio Web [13], y luego para facilitar a los usuarios la navegación en la Web, entregando resultados de búsquedas con alta precisión y velocidad [5], [13] y en algunos casos, presentar los resultados como fragmentos enriquecidos [5], [8] que pueden estar relacionados a personas, organizaciones, eventos, productos, multimedia, entre otros [7]. Un ejemplo de fragmento

[2] https://www.w3.org/

enriquecido se muestra en la **Figura 1** como resultado de la búsqueda de schema.org en el buscador de Google. Este fragmento se presenta gracias a que Wikipedia hace uso del protocolo Open Graph[3] para agregar marcado estructurado semántico en sus páginas Web como se puede observar en la **Figura 2**.

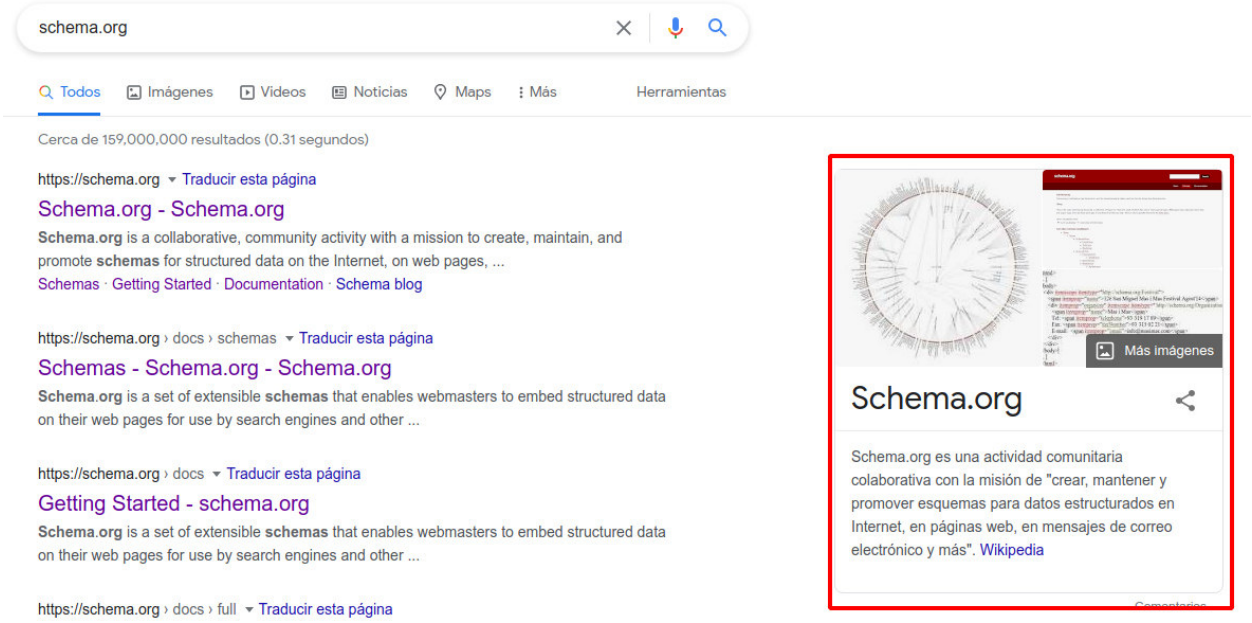


Figura 1. Ejemplo de fragmento enriquecido extraido por Wikipedia resultado de buscar schema.org en Google.

Elaborado por: Eduardo Rosero

```
<meta property="og:image" content="https://upload.wikimedia.org/wikipedia/commons/thumb/8/
8e/Schema.org_%285925660995%29.png/1200px-Schema.org_%285925660995%29.png">
<meta property="og:title" content="File:Schema.org (5925660995).png - Wikimedia Commons">
<meta property="og:type" content="website">
```

Figura 2. Uso del protocolo Open Graph en las páginas de Wikipedia.

Elaborado por: Eduardo Rosero

Las anotaciones semánticas para generar marcado estructurado embebido requieren de dos elementos: el vocabulario que se va a utilizar para describir los recursos Web (Schema.org, Dublin Core, OpenGraph); y el formato que se va a utilizar para representarlos (JSON-LD, microformat, microdata) [5], [13]. Una combinación común de estos elementos es el uso del vocabulario Schema con el formato JSON-LD.

[3] https://ogp.me/

1.5.2 Vocabulario Schema.org

Schema.org es una iniciativa comunitaria colaborativa que tiene la misión de crear, mantener y promover esquemas para datos estructurados en Internet, en páginas Web, mensajes de correo electrónico y otros [14]. En 2012, Schema.org se convirtió en el vocabulario estándar para producir un esquema de marcado de datos estructurado, esto debido al apoyo que obtuvo por parte de los principales motores de búsqueda (Google, Yahoo, Bing y Yandex) [5], [7].

En el vocabulario Schema se trabaja con tipos o clases que están ordenados en una jerarquía y además tienen asociado un conjunto de propiedades. La mayoría de las propiedades se pueden relacionar con varias clases para describirlas en asociación con conceptos. Por ejemplo, en la Figura 3 se puede visualizar que la clase Person se puede relacionar con las propiedades email, image, jobTitle, name, telephone, url, entre otras para describirla. Los elementos descriptivos que no se pueden utilizar como tipos se denominan términos o propiedades [5]. Actualmente el vocabulario consta de 729 clases y 1447 propiedades [14]. El propósito de Schema es proporcionar una amplia colección de términos destinados a describir conceptos y relaciones en un área de interés [5].

```
<script type="application/ld+json">
{
  "@context": "https://schema.org",
  "@type": "Person",
  "address": {
    "@type": "PostalAddress",
    "addressLocality": "Seattle",
    "addressRegion": "WA",
    "postalCode": "98052",
    "streetAddress": "20341 Whitworth Institute 405 N. Whitworth"
  },
  "colleague": [
    "http://www.xyz.edu/students/alicejones.html",
    "http://www.xyz.edu/students/bobsmith.html"
  ],
  "email": "mailto:jane-doe@xyz.edu",
  "image": "janedoe.jpg",
  "jobTitle": "Professor",
  "name": "Jane Doe",
  "telephone": "(425) 123-4567",
  "url": "http://www.janedoe.com"
}
</script>
```

Figura 3. Uso de propiedades para describir la clase Person de Schema.

Elaborado por: Eduardo Rosero

Schema cubre una amplia variedad de ámbitos, especialmente aquellos que tienen contenido con relevancia comercial directa o indirecta, como productos, lugares, eventos, libros, videos, películas, personas, organizaciones entre otros [5], [13]. Además, gracias a su constante evolución con la adopción de estándares y vocabularios para otros contextos, Schema ha permitido ampliar los campos en los que se puede aplicar como es el caso del ámbito educativo. En 2013 la especificación Learning Resource Metadata Initiative (LRMI) fue aceptada como una extensión

oficial de Schema.org, la cual se preocupa por extender y aplicar Schema.org a la descripción de propiedades relevantes de recursos educativos [5], [7], [15].

El vocabulario Schema se ha convertido en uno de los estándares más utilizados para estructurar datos en la Web [13], es así que en la actualidad más de 10 millones de sitios ocupan Schema.org para marcar sus páginas Web [14]. Esta amplia adopción se da gracias a que Schema.org es compacto y fácil de usar para describir recursos en la Web, para hacerlos interoperables y comprensibles [5], y porque trabaja con los formatos Microdata, RFDa o JSON-LD para codificarlo y agregar información al contenido web [5], [13], [16].

1.5.3 Formato JSON-LD

JSON-LD es un formato nuevo y además aprobado por el W3C como recomendación para la representación de datos estructurados en el marcado estructurado embebido [5]. JSON-LD hace uso del formato JSON (JavaScript Object Notation) para datos enlazados LD (Linked Data) y se trata de un esfuerzo de la comunidad por estandarizar un tipo de medio dirigido a la comunicación de máquina a máquina con soporte de hipermedia y semánticamente rico [17]. Este formato para datos enlazados se caracteriza por ser lo más simple posible, ligero, conciso y de fácil lectura y escritura [17], [18].

Un aspecto importante del formato JSON-LD es que proporciona una forma de serializar los datos en una sintaxis basada en la notación de objetos JavaScript (JSON) tradicional [19] con la que es 100% compatible [17], [20], a la vez que permite agregar semántica a dichos documentos JSON [17]. A diferencia de enfoques tradicionales que trabajan con una estructura denominada triple[4] o tripleta, JSON-LD sigue un enfoque basado en la entidad (orientado a objetos) [17], [20].

El modelo de datos de JSON-LD se basa en un grafo de datos enlazados en donde los nodos del grafo se denominan objetos o sujetos y las aristas se denominan propiedades. Un sujeto es aquel nodo que tiene al menos una arista saliente, mientras que un objeto es aquel nodo que tiene al menos una arista entrante. Existen nodos que pueden ser sujeto y objeto al mismo tiempo. Este modelo de datos se presenta en la **Figura 4**.

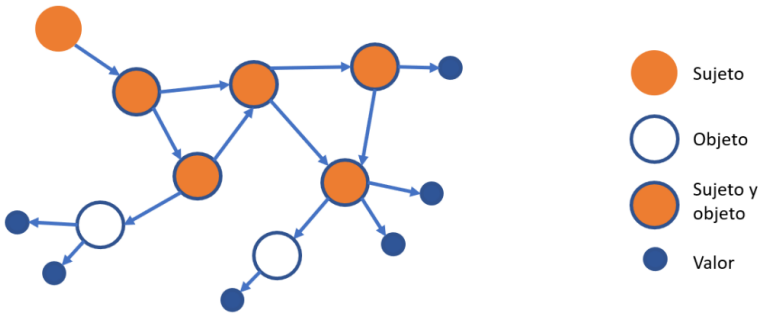


Figura 4. Grafo de datos enlazados del modelo de datos de JSON-LD.

Elaborado por: Eduardo Rosero

[4] Estructura que permite que la información sea procesable por las máquinas. Se compone de tres elementos, sujeto, propiedad y objeto. https://www.w3.org/TR/rdfa-core/#triples

El uso de este formato para agregar marcado estructurado embebido en la Web es sencillo puesto que no se encuentra disperso por todo el documento HTML. De hecho, a diferencia de otros formatos, JSON-LD se coloca en el encabezado de la página Web entre una etiqueta script que contiene la propiedad type con el valor application/ld+json. La **Figura 5** muestra un ejemplo de marcado estructurado embebido en formato JSON-LD.

```
<script type="application/ld+json">
    {
        "@context": "https://schema.org",
        "@type": "CollegeOrUniversity",
        "name": "The Pennsylvania State University",
        "alternateName": "Penn State University",
        "url": "https://www.psu.edu/",
        "logo": "https://standard.psu.edu/images/uploads/psu-mark.svg",
        "address": {
            "@type": "PostalAddress",
            "addressLocality": "University Park",
            "addressRegion": "PA",
            "postalCode": "16802",
            "streetAddress": "201 Old Main",
            "description": "Penn State - The Pennsylvania State University"
        },
        "email": "admissions@psu.edu",
        "faxNumber": "814-863-7590",
        "telephone": "814-865-4700",
        "sameAs": [
            "https://www.facebook.com/pennstate",
            "https://twitter.com/penn_state",
            "https://www.instagram.com/pennstate/",
            "https://www.linkedin.com/school/penn-state-university/",
            "https://www.youtube.com/pennstate"
        ]
    }
</script>
```

Figura 5. Marcado estructurado en formato JSON-LD y con el vocabulario Schema para describir una universidad.

Elaborado por: Eduardo Rosero

1.5.4 Recursos educativos

La mayoría de los recursos educativos tradicionales (libros de texto, fotografía, reportajes de prensa, entre otros) están cayendo en desuso debido a que cada vez existen más recursos educativos digitales disponibles en la Web [21], es por eso que nos centraremos en estos últimos. Dentro de la terminología se puede encontrar los objetos de aprendizaje (LO), recursos educativos (RE) y los recursos educativos abiertos (REA) que se suelen usar como sinónimos, pero tienen sus diferencias, es así que los objetos de aprendizaje se refieren a cualquier recurso digital que se pueda utilizar para el aprendizaje, estos pueden ser abiertos o patentados y sujetos a derechos de autor [22]. Por otro lado, los recursos educativos abiertos son objetos de aprendizaje que están disponibles para su utilización, distribución y modificación bajo licencias abiertas, generalmente Creative Commons (CC) [23].

Es así que, los recursos educativos corresponden a contenidos digitales (imagen, video, audio, página web, documento, etc.) con fines de enseñanza, aprendizaje o investigación [7], que no necesariamente están disponibles de forma gratuita o cuentan con una licencia CC [23]. Estos recursos se almacenan en repositorios en línea y deben seguir algún patrón o estándares dictados por las reglas del repositorio.

Gracias al rápido avance del internet, los recursos educativos se generan y se publican de forma masiva a través de la Web [24], esto ha promovido en gran medida el aprendizaje a distancia, reconociendo la gobernanza y el intercambio de recursos educativos y el aprendizaje cooperativo [25].

1.5.5 Web Scraping

Con la explosión de datos disponibles en la Web, la tarea de recolectar y procesar estos datos es casi imposible de realizar de forma manual [26], para solventar este problema surge el *web scraping*, un proceso para la extracción de grandes volúmenes de datos de forma automática de páginas Web [27], [28], para procesarlos y almacenarlos en formatos útiles y fáciles de manejar como archivos csv, hojas de cálculo, bases de datos, etc [28]–[30]. Estos datos extraídos se pueden desplegar en un sistema local [31], para proveer datos a otras aplicaciones [27].

El proceso de web scraping involucra las tareas de rastreo (crawling) y análisis (parsing) por lo que un componente inherente del web scraping es un web crawler [30]. Los crawlers son robots web que se utilizan para realizar peticiones Web en la Wold Wide Web (WWW) [28], estos acceden a la URL objetivo, descargan el contenido de la página Web y siguen los enlaces del contenido de la página de acuerdo con la implementación que se haya realizado, por otro lado los analizadores (parsers) se encargan de procesar el contenido descargado para extraer datos [10], [26].

Documento HTML

```
<html>
▼<head>
    <title> Página de ejemplo </title>
  </head>
▼<body>
  ▼<div id="container">
      <div class="header"> Header </div>
      <div class="nav"> Menú </div>
      <div class="banner"> Banner </div>
      <hr>
    ▼<div class="content">
        <h1 itemprop="headline" class="title"> Página de ejemplo </h1>
        <h2 itemprop="summary" class="summary"> Resúmen </h2>
      ▼<div itemprop="articleBody" class="article">
          <p> Párrafo 1 </p>
          <p> Párrafo 2 </p>
          <p> Párrafo 3 </p>
        </div>
      </div>
      <div class="footer"> Footer </div>
    </div>
  </body>
</html>
```

Figura 6. Documento HTML de una página Web sencilla.

Elaborado por: Eduardo Rosero

De forma general, los sitios Web tienen cuatro componentes: HTML, CSS, JavaScript y recursos multimedia [26]. Los documentos HTML que son un conjunto de etiquetas como <html> <head> <title> determinadas por el W3C para organizar y representar el contenido de la página Web, son utilizados por los analizadores (parsers) mediante el árbol del Modelo de Objetos del Documento (DOM) que sirve para representar los elementos HTML y sus jerarquías [27]. La **Figura 6** muestra un ejemplo de un documento HTML mientras que la **Figura 7** presenta el respectivo árbol DOM.

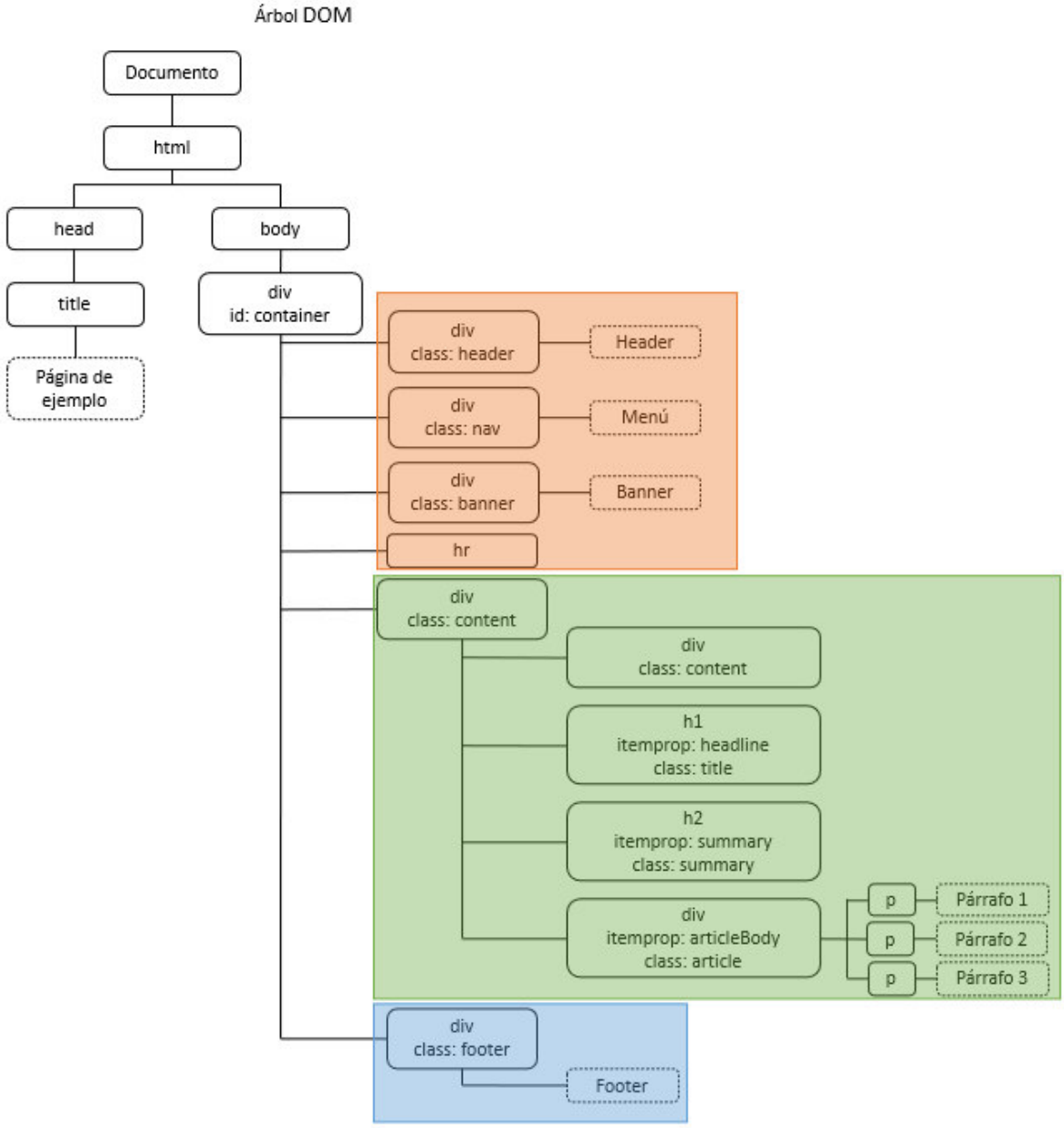


Figura 7. Representación del árbol DOM de un documento HTML.

Elaborado por: Eduardo Rosero

Para acceder a los elementos, los analizadores construyen el árbol DOM y se lo puede recorrer mediante funciones JavaScript como getElementById, getElementsByTagName, getElementsByClassName, querySelector, etc; o mediante el Xpath que se trata de un lenguaje que usa expresiones de ruta para navegar a través del árbol. Estas expresiones Xpath pueden ser utilizadas en varios lenguajes de programación como JavaScript, Java, Python y otros mediante diversas herramientas [27].

Para realizar el proceso de web scraping se pueden encontrar herramientas como librerías, complementos del navegador, servicios en línea, aplicaciones de escritorio, frameworks, que pueden ser de código abierto, gratuitas o comerciales [10]. Para el caso particular de Python, existe Scrapy[5], un framework gratuito y de código abierto que originalmente se creó como un scraper pero se pude utilizar también como un rastrador (crawler) o para extraer datos a partir de APIs [2]. Para crear una aplicación de web scraping con Scrapy, se requiere programación en Python y los datos de interés se pueden obtener a partir de selectores o Xpath. La **Figura 8** presenta un ejemplo simple del código HTML de una página, mientras que la **Figura 9** y la **Figura 10** presentan un ejemplo de la extracción de datos desde la consola de Google Chrome mediante Xpath y selectores CSS respectivamente.

Figura 8. Documento HTML sencillo que muestra un título y un mensaje.

Elaborado por: Eduardo Rosero

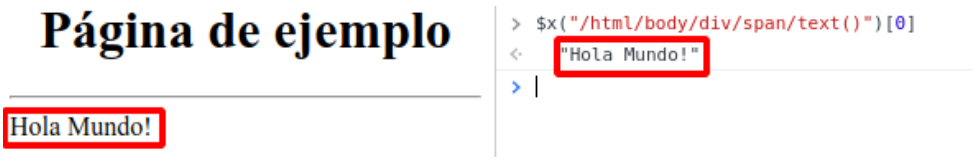


Figura 9. Extracción del texto Hola Mundo! del documento HTML mediante XPath en la consola de Google Chrome.

Elaborado por: Eduardo Rosero

[5] https://scrapy.org/

Figura 10. Extracción del texto Hola Mundo! del documento HTML mediante selectores CSS en la consola de Google Chrome.

Elaborado por: Eduardo Rosero

1.5.6 MongoDB

NoSQL es una nueva generación de bases de datos diseñadas para solventar algunas limitaciones de las bases de datos relacionales [32]. Además han sido denominadas como el futuro de la economía de los datos debido a sus características que se adecúan a aplicaciones de Big Data [33]. Las bases de datos NoSQL permiten manejar grandes volúmenes de datos y tienen una gran relevancia en el desarrollo de aplicaciones Web, aplicaciones móviles o análisis de datos, debido a que proveen características como la escalabilidad y alta disponibilidad [34], [35].

En los modelos de bases de datos relacionales las tablas son definidas bajo un esquema rígido, mientras que en NoSQL no se tiene un esquema predefinido, sino que son más bien flexibles por lo que pueden manejar datos estructurados, no estructurados y semi estructurados. Esta flexibilidad en su modelo de datos y el cambio de paradigma en su modelo de consulta genera un mejor rendimiento comparadas con bases de datos relacionales [36]. Los modelos de datos que manejan las bases de datos NoSQL permiten clasificarlas como: orientadas a documentos, familia de columnas, almacenes clave-valor y bases de datos de grafos [33]–[35]. La importancia de este tipo de bases de datos se puede evidenciar por la adopción de las mismas por parte de empresas como: Google, Facebook, Twitter, Amazon, Pinterest, Yahoo, y más [33], [34], [37], [38].

De la variedad de soluciones NoSQL disponibles actualmente, MongoDB es un sistema gestor de base de datos que ha ganado mucha popularidad [34], puesto que se ha desarrollado específicamente para manejar grandes volúmenes de datos especialmente no estructurados, lo que la ha llevado a reemplazar a las bases de datos relacionales en numerosas aplicaciones [33] y posicionarse en el primer lugar en el 2018 en el ranking de DB-Engines [36].

El sistema gestor de base de datos MongoDB es de código abierto y orientado a documentos [33], que proporciona un alto rendimiento, alta disponibilidad y escalamiento horizontal a la vez que reduce la necesidad de uniones y provee la facilidad de crear esquemas flexibles y dinámicos [32], [35] por lo que soporta todo tipo de datos, en cualquier estructura, formato y que cambian a menudo [39]. Otra de las ventajas de MongoDB es que los objetos de datos tienen el formato JSON, aunque estos se almacenan en BSON un formato binario más eficiente [40].

A diferencia del esquema fijo que manejan las bases de datos relacionales compuestas de filas y columnas, el esquema de almacenamiento en MongoDB es flexible y se basa en documentos y colecciones [34]. Un documento en MongoDB puede tener cualquier tipo de dato fundamental como date, array, string, number [33], aunque también se puede tener subdocumentos o documentos embebidos, lo que resulta muy útil para solventar la falta de uniones y relaciones en

la base de datos, simulándolas con menor latencia [33], [40]. Por otro lado, las colecciones consisten en conjuntos de documentos de estructura similar [34], [36].

MongoDB puede ser usada para construir bases de datos escalables capaces de trabajar de manera local con hardware básico (comodity hardware) o en la nube sin la necesidad de hardware complejo o software extra [39]. La escalabilidad de MongoDB se puede lograr mediante el escalamiento horizontal o sharding que consiste en la distribución de los documentos a través de múltiples nodos y la replicación que consiste en mantener varias copias de los datos para una alta disponibilidad y tolerancia a fallos [33], [34]. Esto además permite analizar datos de cualquier tipo de estructura directamente desde la base de datos logrando resultados analíticos sin ejecutar cargas de trabajo costosas en almacenes de datos (datawarehouses) [39]. En la **Figura 11** se puede visualizar el esquema de particionamiento y en la **Figura 12** el esquema de replicación.

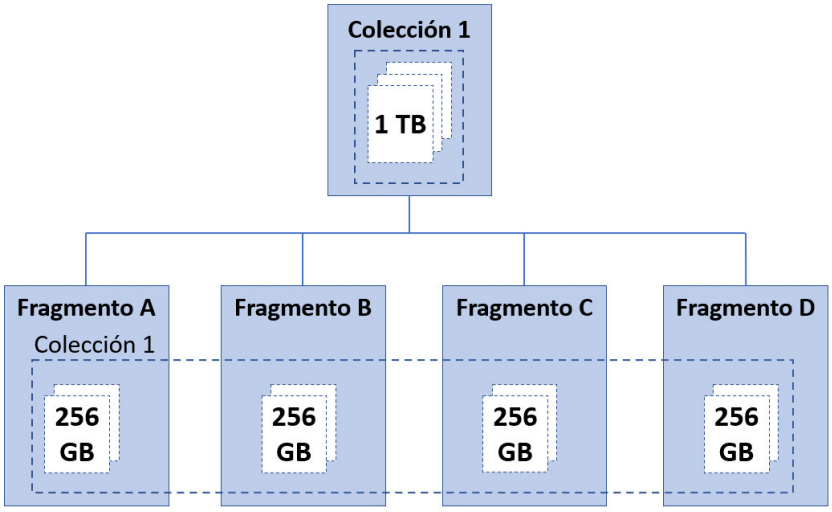


Figura 11. Esquema de particionamiento de datos de la base de datos MongoDB.

Elaborado por: Eduardo Rosero

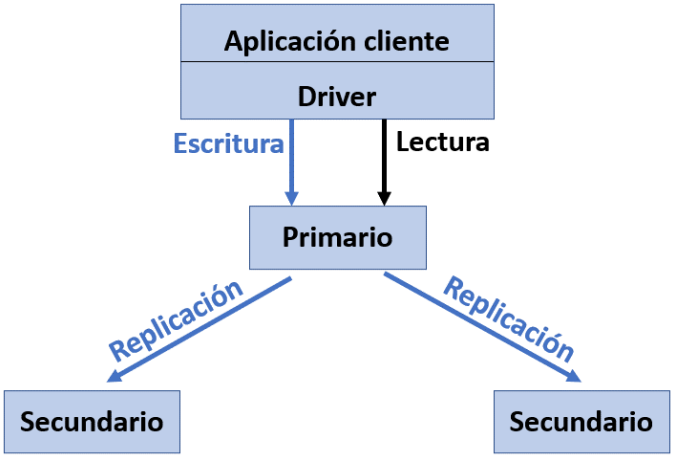


Figura 12. Esquema de replicación de datos de la base de datos MongoDB.

Elaborado por: Eduardo Rosero

Resumiendo lo expuesto, el capítulo presenta la problemática a ser abordada, los objetivos generales y específicos, el alcance del proyecto y el marco teórico de los aspectos más relevantes. Dentro de marco teórico se describe la web semántica la cual hace uso de marcado estructurado embebido para enriquecer el contenido de las páginas Web con metadatos. Adicionalmente, se detalla lo que son los recursos educativos, que junto con el vocabulario Schema para describirlos y el formato JSON-LD para representarlos, forman parte del objeto de estudio del proyecto.

Se aborda también lo correspondiente a web scraping debido a que representa un aspecto fundamental para la recolección de datos de la Web mediante el framework Scrapy. Se explica la aparición de nuevas bases de datos NoSQL con especial énfasis en MongoDB, una base de datos orientada a documentos cuyas características la convierten en la opción ideal para el almacenamiento de datos para el proyecto.

El siguiente capítulo detalla la metodología usada para el desarrollo y evaluación del framework que permitirá el análisis del uso de marcado estructurado embebido con el vocabulario Schema y el formato JSON-LD en los sitios Web de universidades.

2 METODOLOGÍA

Para el desarrollo de este trabajo se utilizó dos metodologías. En primer lugar, puesto que el objetivo principal del trabajo consiste en el desarrollo de un framework, se optó por la metodología de la investigación basada en la Ciencia del Diseño (Design Science Research, DSR) que está compuesta de un conjunto de seis actividades [41], [42] y que, mediante el conocimiento y la comprensión de un determinado problema, permite diseñar una solución construyendo un artefacto. Por otro lado, como el trabajo también aborda un proceso de minería de datos, se consideró el uso de la metodología CRISP-DM (Cross Industry Process for Data Mining), la cual se compone de seis fases, las mismas que conducen la ejecución de las actividades necesarias, para desarrollar un proyecto de minería de datos a través de todo su ciclo de vida [43].

Las metodologías utilizadas no son excluyentes y tienen varios aspectos en común, por lo que se hace una fusión de las dos metodologías de tal forma que los procesos comunes que existen entre ambas se realizan una sola vez. Además, puesto que las dos metodologías se utilizan de forma simultánea, es posible señalar las relaciones existentes entre las fases de CRISP-DM y DSR [1], como se muestra en la **Tabla 1**.

Tabla 1. Relaciones entre las fases de las metodologías CRISP-DM y DSR

<table>
<tr><th>Metodología</th><th colspan="7">DSR</th></tr>
<tr><td rowspan="7">CRISP-DM</td><td>Actividades</td><td>Identificar el problema y la motivación</td><td>Definir objetivos para la solución</td><td>Diseño y desarrollo</td><td>Evaluación</td><td>Demostración</td><td>Comunicación</td></tr>
<tr><td>Comprensión del negocio</td><td colspan="2">Fase 1</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Comprensión de los datos</td><td></td><td></td><td>Fase 2</td><td></td><td></td><td></td></tr>
<tr><td>Preparación de los datos</td><td></td><td></td><td>Fase 3</td><td></td><td></td><td></td></tr>
<tr><td>Modelado</td><td></td><td></td><td>Fase 4</td><td></td><td></td><td></td></tr>
<tr><td>Evaluación</td><td></td><td></td><td></td><td>Fase 5</td><td></td><td></td></tr>
<tr><td>Despliegue</td><td></td><td></td><td></td><td></td><td colspan="2">Fase 6</td></tr>
</table>

Elaborado por: Eduardo Rosero

El propósito de cada una de las fases se detalla a continuación:

- Fase 1: guiar en la comprensión del negocio, a la vez que se identifica el problema y la motivación del proyecto. Además, se establece los objetivos de la solución junto con un plan de ejecución.
- Fase 2: generar una familiarización con los datos mediante varias actividades: la descarga inicial de datos, la exploración y descripción de los datos, la generación de gráficas informativas sobre los datos, entre otras. Adicionalmente, en esta fase se obtiene la definición de la capa de acceso a datos y el servicio de descarga de la capa de servicios del

framework el cual se encarga de la orquestación de la descarga y almacenamiento de los datos. Además, se presenta el componente de ingreso de URL's que pertenece a la capa de aplicación del framework al igual que el servicio de ingreso de URL's perteneciente a la capa de servicios el framework.

- Fase 3: una vez que se conoce los datos se puede trabajar con ellos mediante tareas de limpieza y transformación, esto con el fin de facilitar su uso en tareas futuras. Esta fase conduce en la obtención del servicio de procesamiento de datos de la capa de servicios del framework el cual se encarga de procesar los datos para obtener un conjunto de estos que cumplan las características requeridas para el análisis.
- Fase 4: debido a que en el proyecto no se pretende desarrollar un modelo predictivo o prescriptivo, en esta fase se llevarán a cabo tareas de análisis de datos para la obtención de indicadores clave sobre estos. En esta fase se obtiene un aplicativo a manera de dashboard correspondiente a la capa de aplicación del framework el cual se encarga de la presentación de resultados del análisis de los datos mediante indicadores clave.
- Fase 5: dentro de esta fase se analizará los resultados obtenidos para conocer el nivel de adopción de la tecnología de la Web semántica y la viabilidad del framework para ser aplicado en otros contextos.
- Fase 6: la fase de despliegue de CRISP-DM que se relaciona con las fases de demostración y comunicación de DSR, se sustenta con la publicación del trabajo realizado para este proyecto, así como los resultados obtenidos.

Se puede ver un framework como una guía real o conceptual que sirve como soporte para la ejecución de tareas [44]. El framework propuesto en este trabajo se divide en varias capas las cuales a su vez agrupan varios componentes de acuerdo con el rol que cumplen dentro de este. Las capas definidas para el framework son las siguientes:

1. **Capa de acceso a datos:** esta capa agrupa aquellos componentes cuya funcionalidad es la de brindar un espacio y una forma de almacenamiento de datos. Los componentes que conforman esta capa son:
    a. MongoDB
    b. Redis
    c. Apache Kafka
    d. Disco duro.
2. **Capa de servicios:** esta capa se encarga de agrupar los componentes que ejecutan tareas correspondientes a la lectura, escritura y procesamiento de los datos, las mismas que servirán para prepararlos de tal forma que los demás componentes puedan hacer uso de ellos. Los componentes que conforman esta capa son:
    a. Servicio de ingreso de URL's
    b. Servicio de descarga, el cual consta de tres subcomponentes: scraper master, scraper worker y servicio de almacenamiento de HTML
    c. Servicio de procesamiento de datos
    d. Servicio de consulta de datos

3. **Capa de aplicación:** esta capa contiene los componentes que permiten interactuar con el framework. Estos componentes son:
    a. Aplicación de ingreso de URL´s
    b. Dashboard

Dado que para poner a DSR en acción se requiere de una metodología práctica para llevar a cabo el proyecto [1], y puesto que la metodología práctica escogida es CRISP-DM, en las subsecciones siguientes se describe a detalle la aplicación de dicha metodología para el desarrollo de este trabajo.

2.1 Fase 1

2.1.1 Comprensión del negocio

La fase de comprensión del negocio de CRISP-DM corresponde a la fase inicial compuesta de cuatro tareas y se enfoca en la comprensión de los objetivos y requisitos del proyecto desde una perspectiva comercial. Esta comprensión del negocio se traduce luego en un problema de minería de datos con un respectivo plan diseñado para lograr los objetivos propuestos [43].

2.1.1.1 Objetivos del negocio

En el contexto de la Web semántica y el análisis de datos de marcado estructurado embebido, no existe un proceso automatizado que destine sus esfuerzos a un análisis de ámbito netamente educativo, entendiéndose con esto que el análisis esté orientado a los sitios Web educativos como es el caso de los sitios de las universidades y que se analice específicamente el uso del marcado estructurado para describir recursos educativos publicados dentro de estos sitios Web. De forma aún más específica, no existe un proceso automatizado dentro de este mismo ámbito educativo que permita realizar el análisis del uso de marcado estructurado con el formato de JSON-LD y el vocabulario Schema.

Ante esta problemática, se propuso el desarrollo de un framework que permita comprender si las universidades publican recursos educativos en sus sitios Web haciendo uso de marcado estructurado embebido para facilitar que estos recursos sean encontrados mediante motores de búsqueda como Google, Bing, Yahoo o Yandex [6], [7]. De darse el caso, el framework explicará cómo se realiza el proceso de tal forma que los recursos educativos puedan ser descritos y organizados en la Web mediante marcado estructurado.

2.1.1.2 Evaluación de la situación actual

Dentro de esta tarea se detallan aspectos importantes relacionados con los recursos, limitaciones, suposiciones y otros factores importantes para el objetivo de la minería de datos y el plan del proyecto [43]. El estado de la situación actual se puede determinar respondiendo a las siguientes preguntas:

- **¿Cuál es el conocimiento previo disponible acerca del problema?**
  No se cuenta con un conjunto de datos específicos del ámbito educativo sobre los cuales realizar el análisis propuesto. Se ha trabajado en este tipo de análisis con conjuntos de datos existentes que no tienen un contexto específico [5], obteniendo resultados que indican que el uso de marcado estructurado embebido para describir recursos Web no es una práctica muy común. Si bien existen trabajos relacionados al ámbito educativo, estos suelen enfocarse en tópicos determinados, como es el caso de [11] en donde se presenta un índice

de recursos educativos abiertos sobre Data Science. Otro caso es el de [2] en donde se diseña una plataforma para acceder a documentos sobre ciberseguridad; o como en [9], [12] en donde se usa datos de sitios Web para crear sistemas de recomendación de cursos de formación y de recursos educativos abiertos sobre Data Science respectivamente. Además, se puede mencionar que los trabajos expuestos obtienen sus datos de toda la Web, es decir que no son recuperados de sitios Web educativos como es el caso de los sitios de las universidades.

- **¿Se cuenta con la cantidad de datos requerida para resolver el problema?**
  Actualmente no existe un conjunto de datos para llevar a cabo el análisis, pero se pretende, como parte de los objetivos, crear este conjunto de datos a partir de la extracción, mediante técnicas de Web Scraping, del contenido público de los sitios Web de las universidades que se encuentran en el top de ranking internacionales.

- **¿Cuáles son los requisitos, supuestos y restricciones del proyecto?**
  Se debe contar con una conexión estable de Internet para el proceso de recopilación de datos puesto que se obtendrán de los sitios Web de las universidades.

  Se espera que los sitios Web sean estáticos y cuenten con un sistema sencillo para la navegación entre páginas Web. Se considera que al acceder a contenido público en los sitios Web de las universidades, no se producirán restricciones de acceso a los mismos para poder extraer la información requerida.

  Se cuenta con que las universidades publiquen recursos educativos dentro de sus sitios Web describiéndolos mediante el uso de marcado estructurado embebido con el formato JSON-LD que es el recomendado como estándar por el W3C [5], [10] en conjunto con el vocabulario Schema.

2.1.1.3 Objetivos de la minería de datos

- Comprender el estado de adopción de marcado estructurado embebido en formato JSON-LD y con el vocabulario Schema en los sitios Web de las universidades.
- Determinar cuáles son los recursos educativos que más se publican.
- Identificar, describir y organizar recursos educativos de acuerdo con el marcado estructurado embebido en el texto (HTML).

2.1.1.4 Plan del proyecto

A continuación, se presenta la Tabla 2 en la que constan las distintas etapas y el tiempo estimado destinado a las mismas para llevar a cabo el desarrollo del proyecto considerando un tiempo total de 460 horas.

Tabla 2. Plan del proyecto con detalle de tareas y tiempo estimado

Etapa	Descripción	Tiempo estimado
1	Definir la estructura para el almacenamiento de los datos	10 horas
2	Selección de las tecnologías y herramientas para la obtención y el almacenamiento de los datos	20 horas
3	Selección de los sitios Web de los que se extraerá los datos	10 horas
4	Desarrollo de la estrategia para la recolección y el almacenamiento de datos	360 horas
5	Exploración de los datos	15 horas
6	Seleccionar la técnica de minería de datos para la extracción de información a partir del texto (HTML)	10 horas
7	Implementar la estrategia de minería de datos	20 horas
8	Análisis de resultados obtenidos, de ser el caso regresar a la etapa 3	10 horas
9	Presentación de resultados finales	5 horas

Elaborado por: Eduardo Rosero

2.2 Fase 2

2.2.1 Comprensión de los datos

Dentro de esta fase se realizaron actividades destinadas a comprender los datos en algunos ejes importantes como la calidad o la estructura, con el objetivo de familiarizarse con los datos y así lograr un primer acercamiento al conocimiento que se puede obtener a partir de estos. Para lograrlo se siguió un conjunto de tareas o sub-fases entre las que destacan la recolección inicial de los datos, un análisis exploratorio para describirlos y la verificación de la calidad de los datos [43].

2.2.1.1 Recolección de datos iniciales

Para la recolección de los datos, se desarrollaron varios componentes del framework en diversas capas de éste. Estos componentes se acoplan y complementan entre ellos para permitir realizar la descarga y el almacenamiento de los datos. A continuación, se detallan las herramientas utilizadas y los componentes desarrollados; además se explica el rol que desempeñan dentro del framework.

1. **Aplicación de ingreso de URL's**

Este componente forma parte de la capa de aplicación del framework. Consiste en una aplicación Web que permite ingresar las URL's de los sitios Web de las universidades de las que se desea descargar datos.

Uno de los objetivos del proyecto es la recopilación y el almacenamiento de los datos para su posterior análisis. Entonces, como paso inicial, previo a la recolección de los datos, se definió las fuentes de las cuales se va a hacer la extracción. Así, considerando que las universidades que se encuentran en top ranking internacionales deben estar a la vanguardia en cuestiones de tecnología, se determinó que los sitios web de cada una de estas universidades serían una buena fuente para la obtención de datos.

El listado de las 150 primeras universidades del top ranking se tomó de la página QS World University Rankings[6] seleccionando el año 2021 como filtro, de las cuales al final se conservará las 100 de las que más datos se haya descargado. Este listado se ingresó en un archivo de Excel en el cual posteriormente se agregó la URL del sitio Web de la universidad. La **Figura 13** muestra un extracto de siete universidades junto con su dirección web.

1	universidad	url
2	Massachusetts Institute of Technology (MIT)	https://www.mit.edu/
3	Stanford University	https://www.stanford.edu/
4	Harvard University	https://www.harvard.edu/
5	California Institute of Technology (Caltech)	https://www.caltech.edu/
6	University of Oxford	https://www.ox.ac.uk/
7	ETH Zurich - Swiss Federal Institute of Technology	https://ethz.ch/en.html
8	University of Cambridge	https://www.cam.ac.uk/

Figura 13. Extracto del archivo Excel que contiene los nombres de 7 universidades con la URL de su sitio Web.

Elaborado por: Eduardo Rosero

Con este archivo listo, se puede acceder a la aplicación de ingreso de URL's para cargar el archivo y enviarlo al servicio de ingreso de URL's. La forma de cargar el archivo es mediante un formulario Web como se presenta en la **Figura 14**.

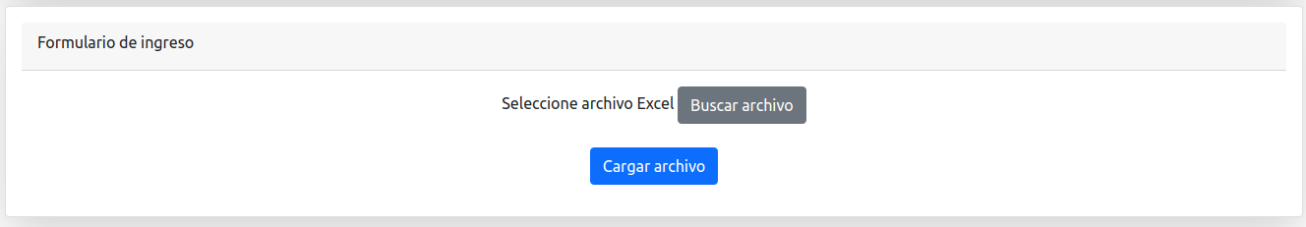


Figura 14. Formulario Web para la selección del archivo Excel que se enviará a procesar.

Elaborado por: Eduardo Rosero

Una vez que se envía el archivo, se redirige a una nueva página dependiendo del resultado: si el archivo se cargó y procesó con éxito, se muestra un mensaje indicando que se ejecutó correctamente, caso contrario, si ocurrió algún error al procesar el archivo se muestra un mensaje de error. La visualización del mensaje de éxito se puede apreciar en la **Figura 15** mientras que la visualización del mensaje de error se presenta en la **Figura 16**.

[6] https://www.topuniversities.com/university-rankings/world-university-rankings/2021

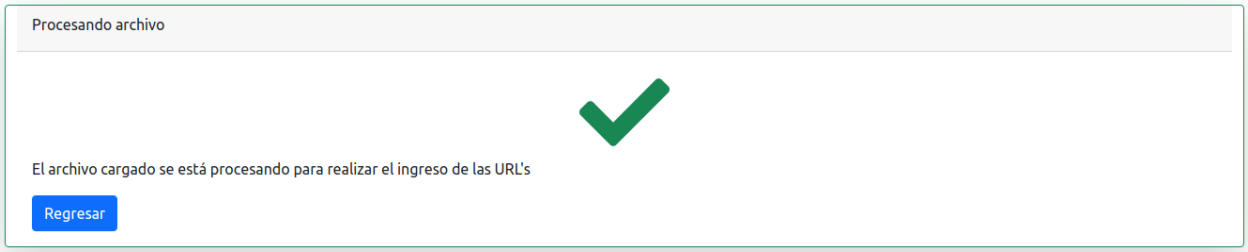


Figura 15. Mensaje de éxito que se visualiza luego del procesamiento del archivo Excel.

Elaborado por: Eduardo Rosero

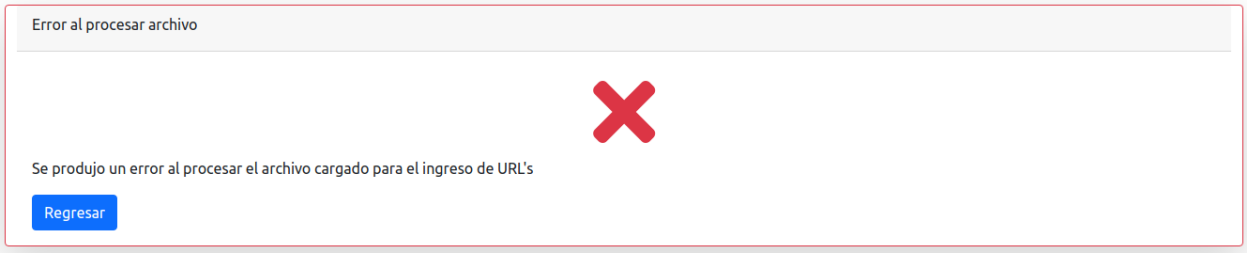


Figura 16. Mensaje de error que se presenta cuando no se puede procesar el archivo Excel.

Elaborado por: Eduardo Rosero

2. Servicio de ingreso de URL's

Este componente perteneciente a la capa de aplicación del framework, es el encargado de procesar el archivo cargado mediante el componente de ingreso de URL's. Una vez que se recibe el archivo se valida que tenga la estructura correcta, si la tiene, se procesa los registros que contiene y se los ingresa en la colección *universities* en MongoDB, la cual permite organizar los datos de acuerdo a esquemas dinámicos y que ofrece ciertas ventajas como: a) permitir la indexación de datos, b) realizar consultas flexibles hacia los documentos, c) facilitar la comunicación entre la base de datos y diferentes lenguajes de programación, y d) hallar soporte en una gran comunidad de desarrolladores ya que su uso es amplio, entre otras [2]. La **Tabla 3** detalla los campos que componen los documentos de la colección *universities*, mientras que el flujo de las tareas que se ejecutan en el servicio de ingreso de URL's se presenta en el diagrama de la **Figura 17**.

Tabla 3. Detalle de los campos de la colección universities

Campo	Tipo	Descripción
_id	String	Hash MD5 del dominio del sitio web de la universidad y que permite identificar el documento de manera única dentro de la colección.
name	String	Nombre de la universidad
url	String	URL del sitio Web de la universidad
processed	Boolean	Identifica si se ha realizado la descarga de datos para la URL contenida en el documento

Elaborado por: Eduardo Rosero

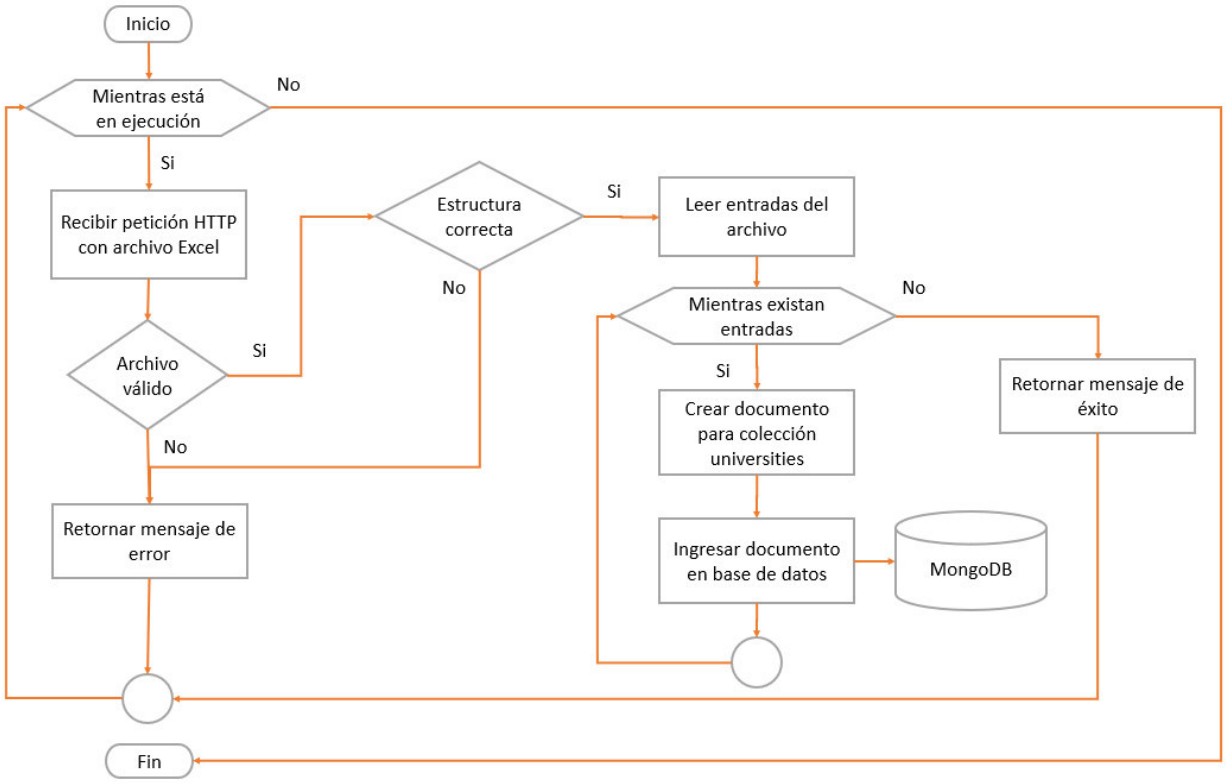


Figura 17. Diagrama de flujo del servicio de ingreso de URL's.

Elaborado por: Eduardo Rosero

Un ejemplo de un registro se muestra en la **Figura 18** en la que se visualiza la representación JSON del documento ingresado en la base de datos.

```
{
    "_id" : "615185c1398d6a3ee56d42b27781c6ef",
    "name" : "Massachusetts Institute of Technology (MIT) ",
    "url" : "https://www.mit.edu/",
    "processed" : false
}
```

Figura 18. Representación JSON de un documento de la colección universities en MongoDB.

Elaborado por: Eduardo Rosero

3. MongoDB

La base de datos orientada a documentos MongoDB que forma parte de la capa de acceso a datos, es usada por el servicio de descarga para la lectura y el almacenamiento de datos. Para la descarga de datos se requiere de la colección *universities* (**Tabla 3**), en la que se almacenan los registros de las universidades con su nombre y URL de la página principal del sitio Web y una nueva colección *scrapedPages*, en la que se almacenan los datos obtenidos a partir del Web Scraping de los dos

componentes: scraper master y scraper worker. La **Tabla 4** muestra la estructura de los documentos de la nueva colección con la descripción de cada uno de sus campos. Nótese el campo *valid* que determina si existe el formato JSON-LD en la página Web.

Tabla 4. Detalle de la estructura de los documentos de la colección scrapedPages

Campo	Tipo	Descripción
_id	String	Hash MD5 de la url de la página Web que se visitó en el sitio de la universidad y que permite identificar el documento de manera única dentro de la colección.
domain	String	URL inicial del sitio web de la universidad.
domainHash	String	Hash MD5 de la url inicial del sitio web de la universidad.
url	String	URL de la página Web que se visitó en el sitio de la universidad.
depth	Integer	Cantidad de enlaces que se siguieron hasta llegar a esta página.
processed	Boolean	Bandera que indica si el documento ya se ha procesado para analizar el contenido del código HTML correspondiente.
valid	Boolean	Identifica si el código HTML obtenido de una página Web contiene marcado estructurado en formato JSON-LD.
createdAt	Datetime	Fecha en la que se almacenó el documento en la base de datos.
processedAt	Datetime	Indica la fecha en la que se procesó el documento para analizar el contenido del código HTML correspondiente.
jsonld	Object	Campo de tipo lista que puede ser vacía o almacenar el marcado estructurado en formato JSON-LD obtenido del HTML de la página Web.

Elaborado por: Eduardo Rosero

La **Figura 19** muestra el esquema de la colección universities. Se puede apreciar que consta de cuatro campos. EL campo _id de tipo String representa el identificador único del documento dentro de la colección. Además, el campo _id está indexado para facilitar y acelerar los procesos de búsqueda.

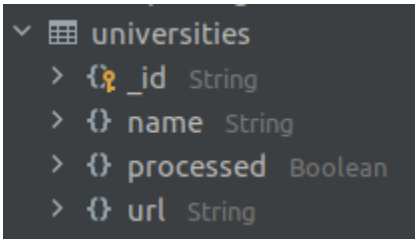


Figura 19. Esquema de la colección universities.

Elaborado por: Eduardo Rosero

La **Figura 20** muestra a su vez el esquema de la colección scrapedPages. En esta colección se almacenan los datos obtenidos a partir del Web Scraping. De forma similar, esta colección tiene un campo _id, el cual permite identificar al documento de forma única dentro de la colección. Este campo _id de la colección scrapedPages también está indexado.

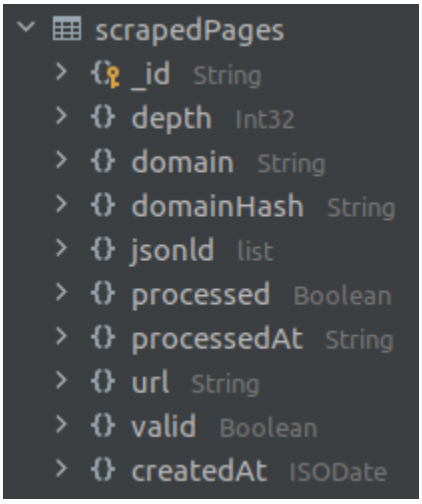


Figura 20. Esquema de la colección scrapedPages.

Elaborado por: Eduardo Rosero

El componente de almacenamiento de datos MongoDB interactúa con los siguientes módulos del servicio de descarga:

- Scraper master, para leer los registros de las universidades de la colección universities y para guardar datos en la colección scrapedPages.
- Scraper workers, para guardar datos en la colección scrapedPages.

4. Disco Duro

Este componente de la capa de acceso a datos es utilizado para almacenar el código HTML de las páginas descargadas. Para organizar los archivos se maneja la lógica de creación de un directorio para cada universidad de la que se va a descargar datos de su sitio Web. Para nombrar el directorio se usa el hash MD5 de la URL de la página de inicio del sitio Web. Dentro de este directorio se almacenan los archivos que contienen el código HTML de las páginas Web descargadas para cada universidad. Para nombrar los archivos también se utiliza el hash MD5 de la URL desde la que se accedió a la página Web, seguido de la extensión .html.

Para la asignación de los nombres de directorios y archivos, así como la definición de la estructura de almacenamiento, se tomó en cuenta dos criterios:

1. La organización de los archivos
2. La facilidad de acceso mediante la reconstrucción de la ruta relativa hacia el archivo

Este componente es utilizado por el servicio de almacenamiento de archivos HTML. La **Figura 21** muestra un ejemplo de nueve directorios creados para almacenar archivos; mientras que la **Figura 22** muestra un ejemplo de nueve archivos HTML almacenados con el hash MD5 como nombre y la respectiva extensión .html.

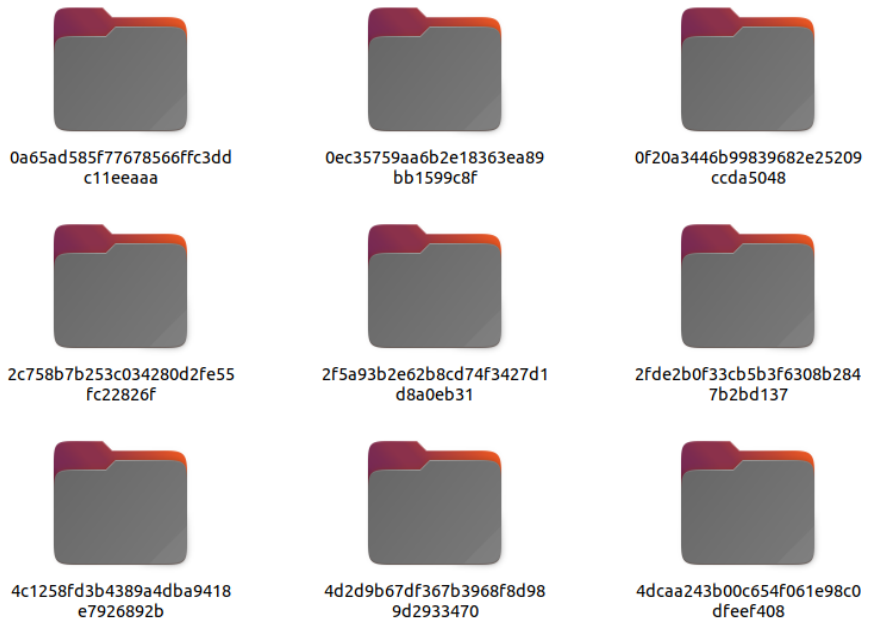


Figura 21. Directorios correspondientes a nueve sitios Web de universidades para almacenar documentos HTML.

Elaborado por: Eduardo Rosero

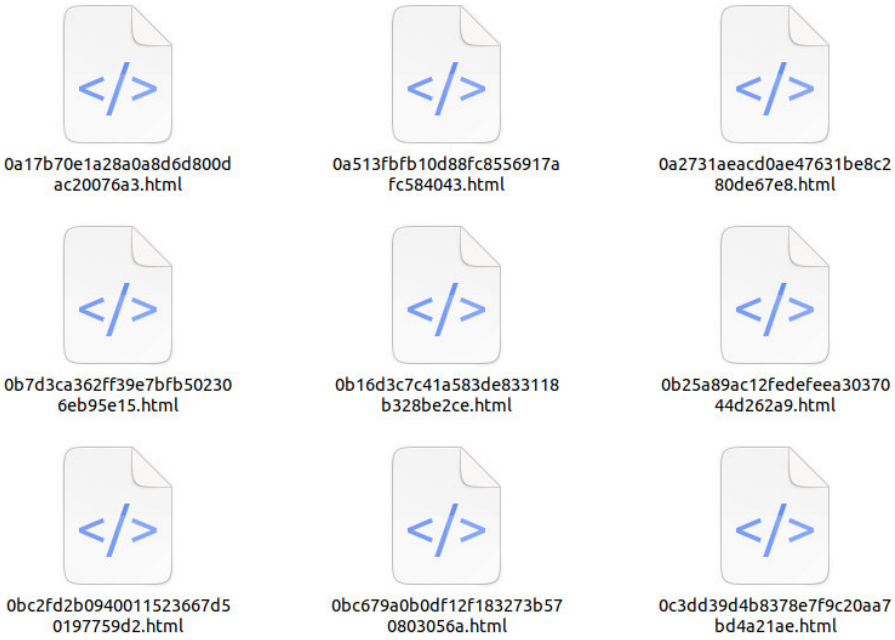


Figura 22. Documentos HTML correspondientes a nueve páginas Web del sitio Web de una universidad.

Elaborado por: Eduardo Rosero

5. Servicio de almacenamiento de archivos HTML

Este componente de almacenamiento de HTML forma parte del servicio de descarga de datos de la capa de servicios y se encarga de la gestión de directorios y archivos. La relación con los directorios se refiere a la creación y validación. En el caso de los archivos se refiere a la escritura de estos en los diferentes directorios.

El servicio de almacenamiento se relaciona de forma directa con el componente de Apache Kafka. Dentro del servicio de almacenamiento se ejecuta un bucle que consulta y recupera datos de un tópico de Kafka. Los datos recuperados se utilizan para validar la existencia del directorio en el que se va a crear el archivo o crearlo de ser el caso. Una vez definida la validez del directorio destino se escribe el archivo para guardarlo en disco.

El flujo de las actividades que ocurren en el servicio de almacenamiento de archivos HTML se presenta en la **Figura 23**.

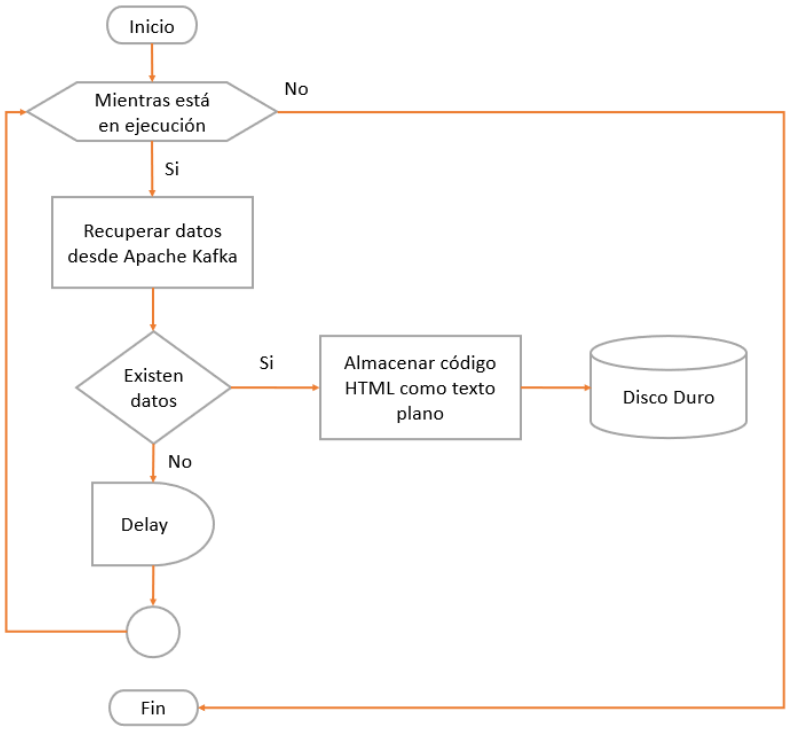


Figura 23. Diagrama de flujo del servicio de almacenamiento de archivos HTML.

Elaborado por: Eduardo Rosero

6. Scraper master

El componente scraper master forma parte del servicio de descarga de la capa de servicios y es el punto de entrada del proceso de Web Scraping. Se lo denomina master porque a partir de este componente se genera el trabajo para los componentes scraper workers. Scraper master interactúa en primer lugar con el componente de almacenamiento de datos MongoDB. Se realiza la consulta

a la base de datos para recuperar los registros de las universidades de las que se va a descargar datos. Los registros se recuperan en bloques de 10 ítems cada vez.

Para cada uno de estos ítems se genera un servicio de Web Scraping dentro del mismo componente que realiza las siguientes actividades:

- Recuperar la URL de la página principal del sitio Web de la universidad
- Realizar la petición de la página Web
- Verificar la validez de la respuesta a la petición
- Recuperar el código HTML de la página Web
- Extraer datos del HTML
- Realizar el almacenamiento de los datos extraídos en la colección *scrapedPages* en MongoDB
- Enviar datos a un tópico en Apache Kafka para el almacenamiento del HTML de las páginas Web en su directorio correspondiente en el disco duro
- Verificar si los enlaces hacia otras páginas extraídos del HTML ya se han procesado antes
- Enviar datos para el procesamiento de la segunda fase del Web Scraping

La segunda interacción con la base de datos Mongo se realiza al momento del guardado de datos recuperados en el proceso de Web Scraping. También se realiza la interacción con el componente Apache Kafka: la primera vez cuando se envía las URL's a un tópico para que los scraper workers realicen la segunda fase de Web Scraping y la segunda vez cuando se envía el HTML a un tópico para su posterior descarga.

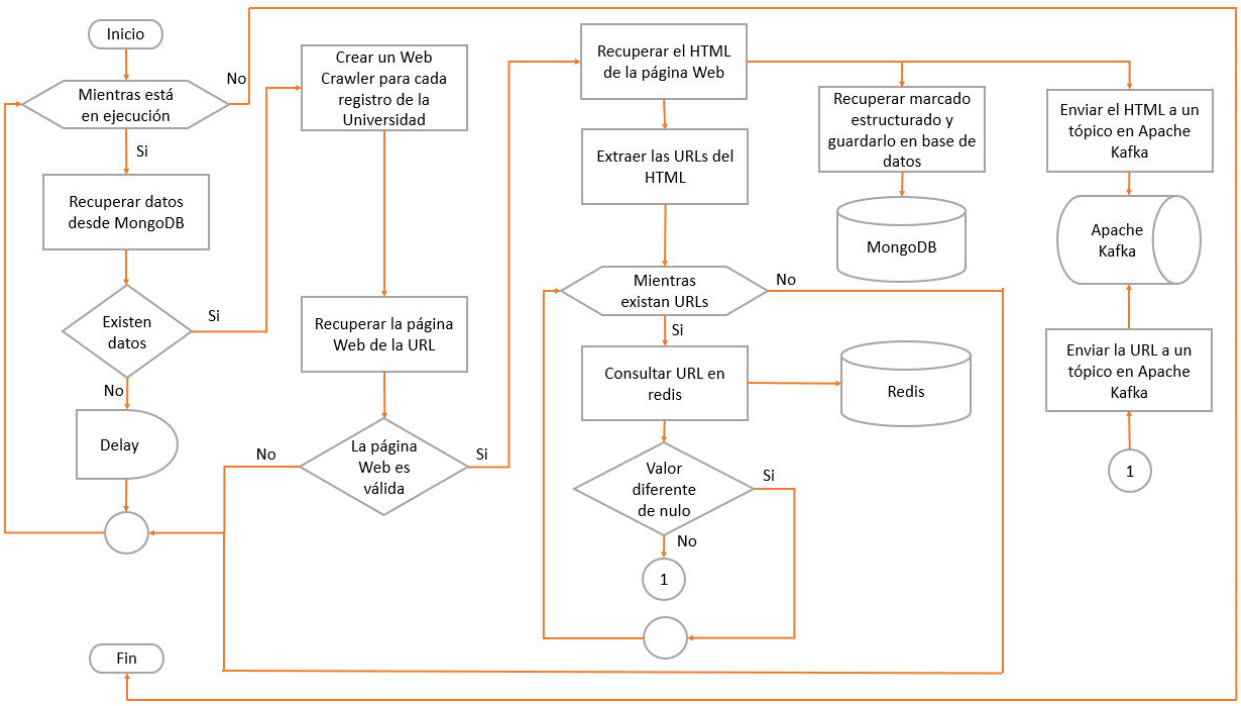


Figura 24. Diagrama de flujo del componente scraper master del servicio de descarga de datos.

Elaborado por: Eduardo Rosero

Otra interacción importante es la que tiene con el componente Redis. Cuando se recuperan las URL's del HTML antes de permitirles continuar con el flujo, se realiza una validación para comprobar si han sido procesadas antes, esto con el fin de evitar realizar procesamientos de descarga duplicados. Esta validación se realiza consultando a Redis por la existencia de la URL en su registro. Si existe la URL no se envía al procesamiento, si no existe se genera el registro en Redis y se envía la URL al procesamiento. El flujo de las actividades que ocurren en el componente scraper master se presentan en la **Figura 24**.

7. Scraper worker

Scraper worker es otro componente del servicio de descarga de datos de la capa de servicios y corresponde a la segunda fase de proceso de Web Scraping la cual se realiza a partir de las URL's obtenidas por el scraper master. Se creó esta segunda fase de Web Scraping para optimizar el proceso de descarga de datos ya que, al tener varias instancias, estas trabajan en paralelo generando mejores tiempos en la descarga de datos. El primer componente con el que interactúa es con Apache Kafka para recuperar los ítems que contienen las URL's, con los cuales se genera un proceso de Web Scraping para cada uno de ellos y se realizan las siguientes actividades:

- Recuperar la URL de una página del sitio Web de la universidad
- Realizar la petición de la página Web
- Verificar la validez de la respuesta a la petición
- Recuperar el código HTML de la página Web
- Extraer datos del código HTML de la página Web
- Realizar el almacenamiento de los datos extraídos en la colección *scrapedPages* en MongoDB
- Enviar datos a un tópico en Apache Kafka para el almacenamiento del archivo HTML de las páginas Web en su directorio correspondiente en el disco duro
- Verificar si los enlaces hacia otras páginas extraídos del código HTML de la página ya se han procesado antes
- Enviar datos para dentro del mismo servicio para continuar de forma iterativa con el proceso de Web Scraping hasta cumplir una condición de parada

La segunda interacción de los scraper workers con Apache Kafka, ocurre al momento de enviar el código HTML de las páginas a un tópico para que este sea almacenado en disco por el servicio correspondiente. También interactúan con la base de datos MongoDB cuando se realiza el almacenamiento de los datos extraídos mediante el Web Scraping. La interacción con el componente Redis tiene el mismo propósito que en el scraper master. Una vez recuperados los enlaces hacia otras páginas a partir del código HTML de la página actual, se verifican consultado a Redis para saber si ya han sido procesados antes. En caso de no haber sido procesados, se ejecutan dentro del mismo servicio de Web Scraping. Este proceso se repite de forma iterativa hasta alcanzar la condición de parada la cual está determinada por un límite en la profundidad de enlaces seguidos. La **Figura 25** muestra un esquema que explica la lógica de la profundidad de enlaces seguidos, además, el flujo de las actividades que ocurren en el componente scraper worker se presenta en la **Figura 26**.

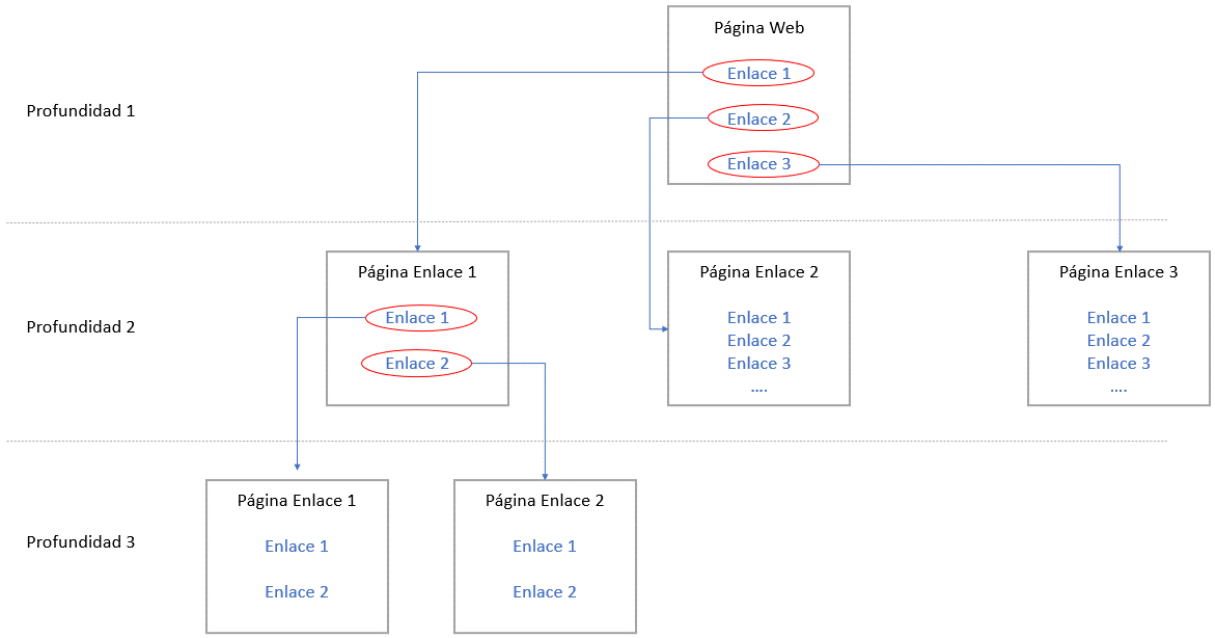


Figura 25. Representación de la navegación en un sitio Web mediante el seguimiento de enlaces.

Elaborado por: Eduardo Rosero

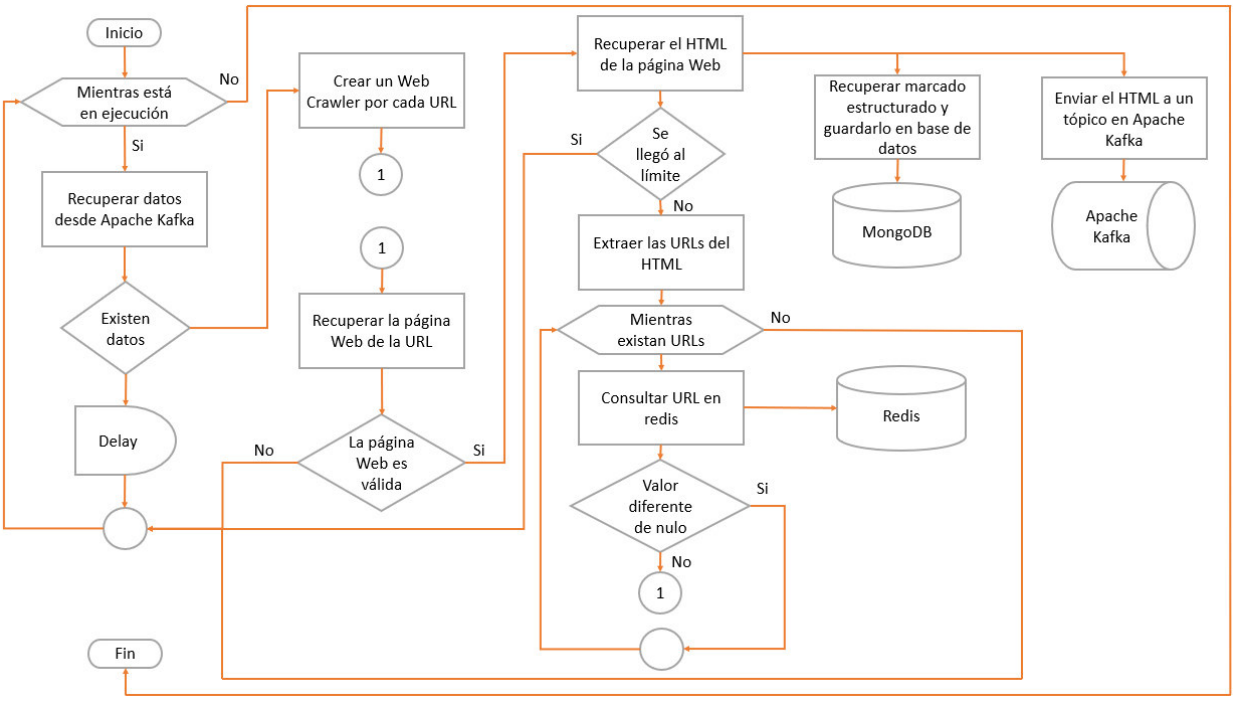


Figura 26. Diagrama de flujo del componente scraper worker del servicio de descarga de datos.

Elaborado por: Eduardo Rosero

8. Apache Kafka

El componente Apache Kafka que forma parte de la capa de acceso a datos, cumple la función de intermediario para la comunicación entre el resto de los componentes que intercambian datos entre ellos. Básicamente se manejan dos tópicos con la lógica de productor/consumidor, en donde el productor publica datos en el tópico y el consumidor los recupera para utilizarlos. El detalle de los tópicos creados en Apache Kafka para el servicio de descarga se presenta en la **Tabla 5**.

Tabla 5. Descripción de los tópicos utilizados en Apache Kafka

Tópico	Función	Características
to-save-files	Compartir los datos correspondientes al código HTML para almacenarlo en disco.	Se trata de un tópico simple con la lógica de productor/consumidor.
to-download-URL's	Compartir los datos correspondientes a las URL's extraídas por el componente scraper master para ser consumidas por el componente scraper worker.	Se trata de un tópico con la lógica de productor/consumidor que está dividido en particiones para distribuir los datos entre varios consumidores.

Elaborado por: Eduardo Rosero

9. Redis

El componente Redis de la capa de acceso a datos, se trata de una base de datos en memoria, se utiliza como caché para llevar un registro de las URL's que se han procesado mediante Web Scraping para la descarga de datos. Redis almacena los datos en una estructura clave-valor (estructura de datos tipo *hash table*) por lo cual, se almacenan las URL de las páginas Web como claves y el valor corresponde a las veces que se ha recuperado dicha URL en el proceso de Web Scraping para su procesamiento.

Los componentes scraper master y scraper worker son los que interactúan con Redis. Estos componentes realizan consultas para determinar si las URL's que extraen en el proceso de Web Scraping ya han sido procesadas o no. La consulta consiste en enviar la URL a Redis para recuperar el valor asociado, si el valor que retorna Redis es nulo, significa que la URL que se consultó aún no ha sido procesada. En este caso, esta URL se envía al tópico to-download-urls en Kafka si el que consulta es el scraper master, o se continua con el proceso iterativo de Web Scraping si el que consulta es el scraper worker. Por el contrario, si el valor que retorna Redis es diferente de nulo, es decir, retorna un número entero, este número indica la cantidad de veces que se encontró dicha URL al realizar el Web Scraping. Cuando se obtiene este resultado, tanto el scraper master como el scraper worker omiten la URL y continúan la ejecución con la siguiente.

Esta estrategia del uso de Redis como caché, beneficia al servicio de descarga porque evita que se efectúen descargas de datos redundantes, esto porque se filtra los datos y no se procesa la descarga de una misma URL dos o más veces.

2.2.1.2 Exploración de los datos

La tarea de exploración de datos como parte del proceso de minería de datos, consiste en realizar consultas a la base de datos y visualizaciones con los resultados obtenidos. Algunas de las cuestiones pueden ser: relaciones de atributos, resultados producidos a partir de agregaciones simples y análisis estadísticos simples [43]. Las consultas realizadas para el proceso exploratorio permitieron tener una visión más amplia y exacta de los datos para lograr una mejor descripción de estos. Así mismo se pudo comprender de mejor manera el estado en cuanto a la calidad de los datos [43]. Una de las consultas consiste en comprender la cantidad de páginas que se han descargado por dominio. Esto nos dará una idea del tamaño del sitio Web de la universidad. La consulta realizada a la base de datos para este caso se muestra en el **Fragmento 1** y el resultado obtenido se presenta en la **Figura 27**.

Fragmento 1. Consulta de la cantidad de páginas descargadas por dominio

```
db.scrapedPages.aggregate( [
  {
    $group: {
       _id: "$domain",
       count: {$sum: 1}
    }
  },
  {
       $match: {"count": {$gt: 1}}
     },
     {
    $sort: {count: -1}
  }
])
```

Elaborado por: Eduardo Rosero

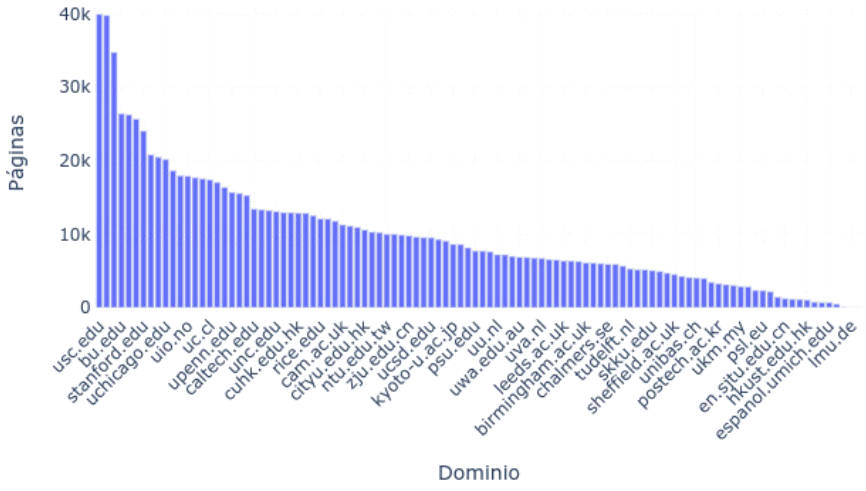


Figura 27. Cantidad de páginas Web descargadas para cada uno de los dominios de las universidades analizadas.

Elaborado por: Eduardo Rosero

Otra de las consultas que se realizó fue para comprender el crecimiento exponencial que se presenta en el proceso de Web Scraping. Mientras se extraen las URL's de las páginas para integrarlas al proceso de Web Scraping, la cantidad de páginas descargadas por nivel de profundidad aumenta. La consulta para este caso se muestra en el **Fragmento 2** y el resultado se presenta en la **Figura 28**.

Fragmento 2. Consulta de la cantidad de páginas descargadas por nivel de profundidad

```
db.scrapedPages.aggregate([
  {
    $group: {
       _id: "$depth",
       count: {$sum: 1}
    }
  },
  {
       $match: {"count": {$gt: 1}}
     },
     {
    $sort : {count: -1}
  }
])
```

Elaborado por: Eduardo Rosero

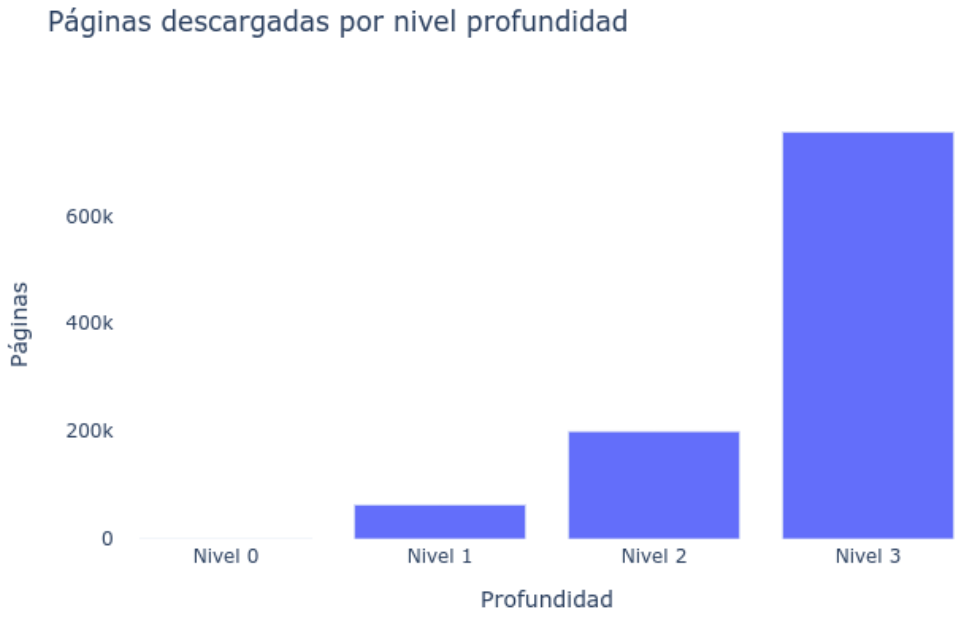


Figura 28. Cantidad de páginas Web descargadas por nivel de profundidad.

Elaborado por: Eduardo Rosero

Para comprender la relación existente entre páginas válidas y no válidas de forma general, tomando en cuenta que, para el contexto de este trabajo las páginas válidas son aquellas que hacen uso del vocabulario Schema y el formato JSON-LD en su marcado estructurado para describir recursos, se realizó la consulta que se muestra en el **Fragmento 3** y su resultado se presenta en la **Figura 29**.

Fragmento 3. Consulta de la cantidad de páginas válidas y no válidas descargadas

```
db.scrapedPages.aggregate([
    {
        "$group": {
           "_id": "$valid",
           "count": {"$sum": 1}
        }
    },
    {
        "$sort": {"count": -1}
    }
])
```

Elaborado por: Eduardo Rosero

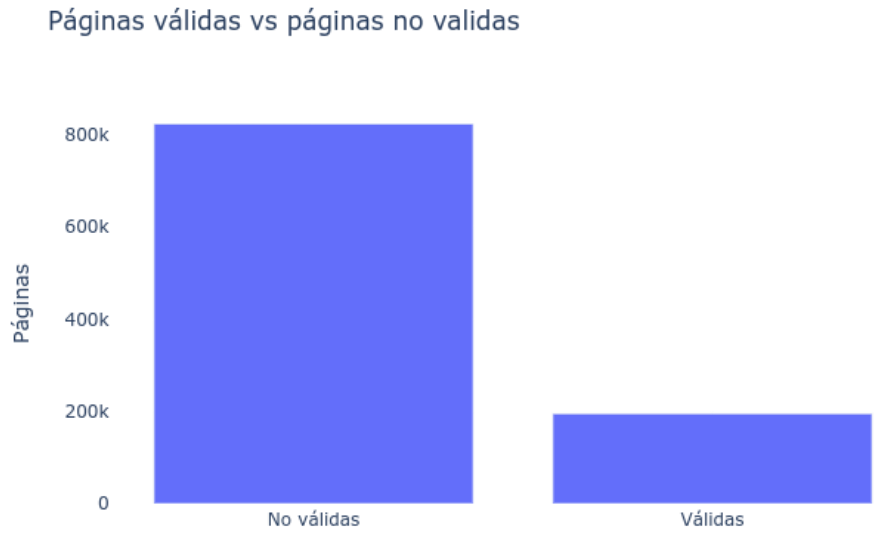


Figura 29. Cantidad de páginas Web válidas y no válidas descargadas.

Elaborado por: Eduardo Rosero

Para comprobar que esta relación se conserva también a nivel de dominio se realizó otra consulta. Para este caso se consideró la cantidad de páginas válidas y no válidas por los dominios de los cuales existen datos. La consulta realizada se muestra en el **Fragmento 4** y sus resultados en la **Figura 30**.

Fragmento 4. Consulta de la cantidad de páginas válidas y no válidas por dominio

```
db.scrapedPages.aggregate([
    {
        $group: {
            _id: {
                "domain": "$domain",
                "valid": "$valid",
            },
            count: {$sum: 1}
        }
    },
    {
        $match: {"count": {$gt: 1}}
    },
    {
        $project: {
            domain: "$_id.domain",
            valid: "$_id.valid",
            count: "$count",
            _id: 0
        }
    },
    {
        $sort: {domain: 1, count: -1}
    }
])
```

Elaborado por: Eduardo Rosero

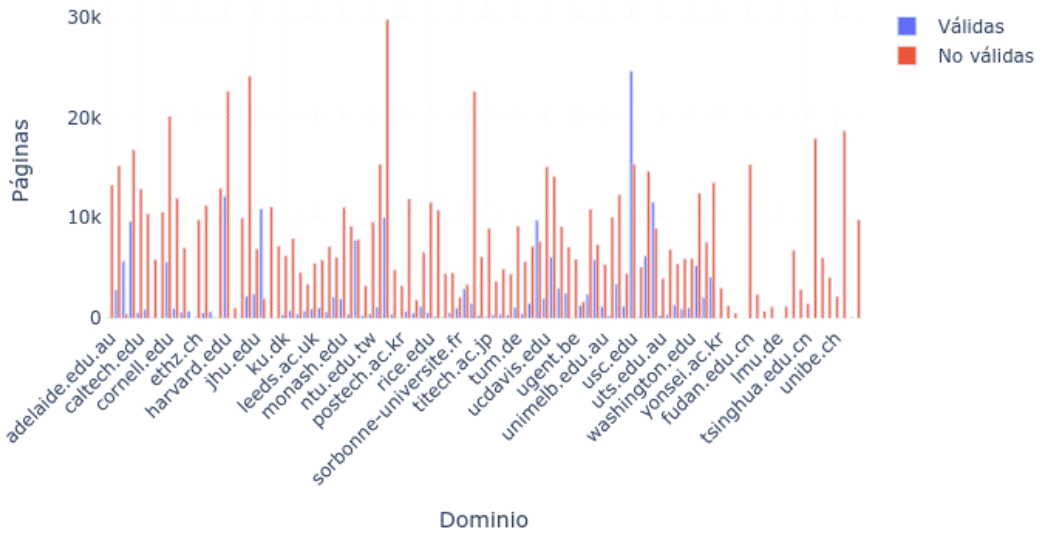


Figura 30. Cantidad de páginas Web válidas y no válidas descargadas para cada uno de los dominios de las universidades analizadas.

Elaborado por: Eduardo Rosero

La exploración de los datos permitió conocer características relevantes sobre los datos, entre las más importantes podemos mencionar:

- El marcado estructurado, específicamente con formato JSON-LD se encuentra con mayor frecuencia a medida que se profundiza en las páginas del sitio Web.
- Los tamaños de los sitios Web analizados varía notablemente en cuanto a la cantidad de páginas Web que se pudo descargar de cada uno.
- La cantidad de páginas Web que contienen marcado estructurado embebido en formato JSON-LD es pequeña comparada con la cantidad de páginas que no tienen marcado estructurado embebido o no se encuentra en el formato requerido.

2.2.1.3 Verificación de la calidad de los datos

En la primera fase, los datos se generaron de forma manual en un archivo Excel para realizar la carga en la base de datos. En este caso se garantiza la calidad en la elaboración del archivo Excel y la posterior carga a la base de datos, la misma que se realizó de forma automatizada mediante un script en Python. Para el proceso de Web Scraping, los datos se generaron y se almacenaron de forma automática. Se consideró el procesamiento y almacenamiento del código HTML solo de las páginas para las que se obtuvo un *código 200* (success) al realizar la petición HTTP de la URL.

Para verificar que las páginas descargadas corresponden al sitio Web de la universidad, se implementaron dos controles: primero se valida que la URL de la que se va a recuperar la página Web corresponde al dominio del sitio Web de la universidad; el segundo control valida que la URL de la página Web descargada siga siendo parte del dominio del sitio Web de la universidad, para omitir páginas externas producto de redirecciones en las peticiones HTTP. Con estos controles se evita guardar páginas de error o que no posean contenido real referente a la universidad o al ámbito educativo, garantizando de esta forma la calidad de los datos descargados.

También se tuvo cuidado al momento de la recuperación del marcado estructurado embebido en el código HTML de las páginas Web. Para este caso se hizo uso de la biblioteca Extruct[7] de Python que permite realizar la extracción de metadatos de páginas HTML en diferentes formatos como JSON-LD, microdata o microformat.

2.2.2 Aportes de la fase

2.2.2.1 Capa de acceso a datos

Para el funcionamiento del framework, especialmente para la descarga de datos se requiere de varios tipos de almacenamiento. Estos componentes de almacenamiento como son MongoDB, Redis, Apache Kafka y disco duro corresponden a la capa de acceso a datos del framework. Además, los componentes de esta capa cumplen diferentes funciones, es así como: MongoDB y disco duro se utilizan para el almacenamiento de datos, mientras que Redis y Apache Kafka, aunque también almacenan datos, lo hacen con la idea de generar comunicación entre los componentes del servicio de descarga. La **Figura 31** muestra un esquema de la conformación de esta capa.

[7] Extruct es una librería de Python que permite extraer metadatos a partir del marcado estructurado embebido en el HTML de las páginas Web. https://pypi.org/project/extruct/

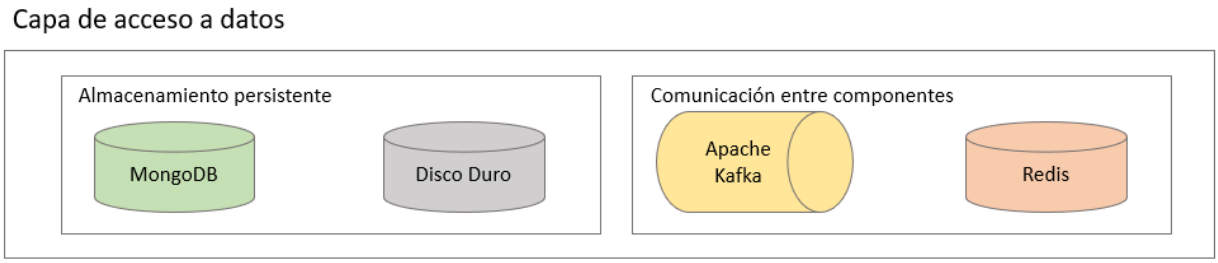


Figura 31. Esquema de la capa de acceso a datos del framework.

Elaborado por: Eduardo Rosero

2.2.2.2 Servicio de descarga

Este servicio forma parte de la capa de servicios del framework y consta de tres elementos: scraper master, scraper worker y servicio de almacenamiento de archivos HTML. Estos tres elementos se encargan de llevar a cabo todo el proceso de descarga y almacenamiento en los diferentes elementos de la capa de acceso a datos. La **Figura 32** presenta el esquema de comunicación entre los componentes del servicio de descarga de la capa de servicios y los componentes de la capa de acceso a datos.

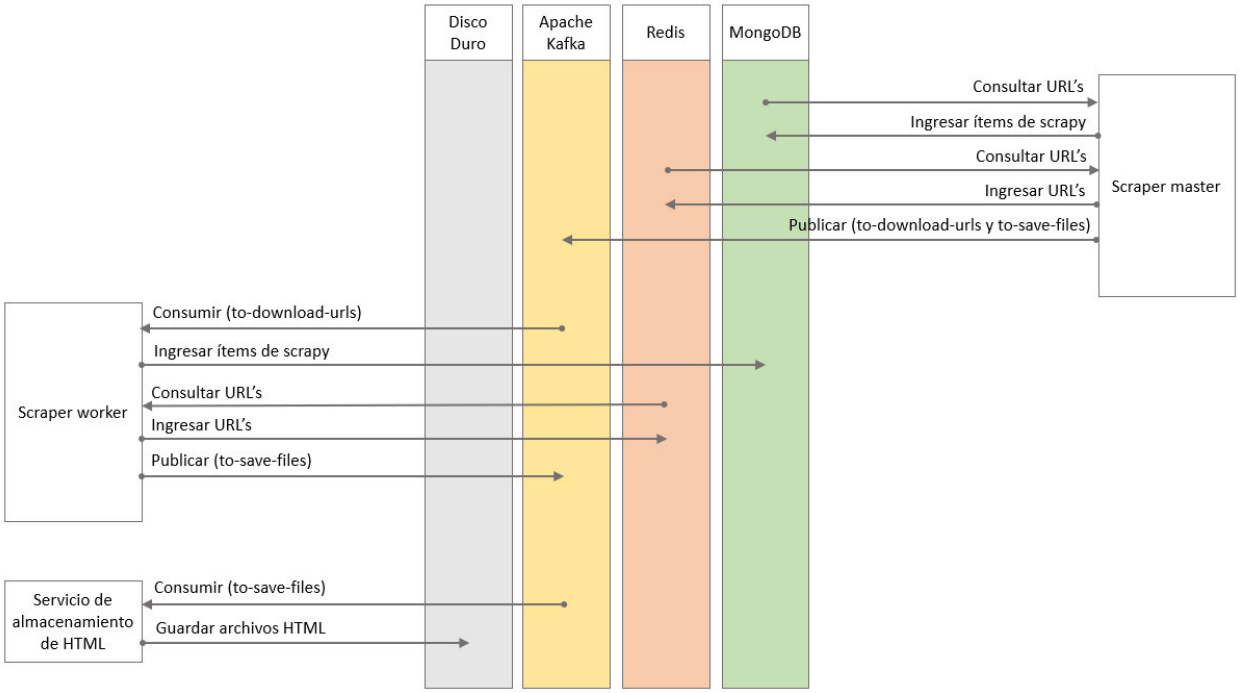


Figura 32. Interacción entre los componentes del servicio de descarga de datos y la capa de acceso a datos del framework.

Elaborado por: Eduardo Rosero

2.2.2.3 Servicio de ingreso de URL’s

Este componente forma parte de la capa de servicios y consta de un solo elemento encargado de realizar el ingreso de los datos de las universidades en base de datos. La **Figura 33** muestra la interacción que existe entre este servicio con la capa de acceso a datos del framework.

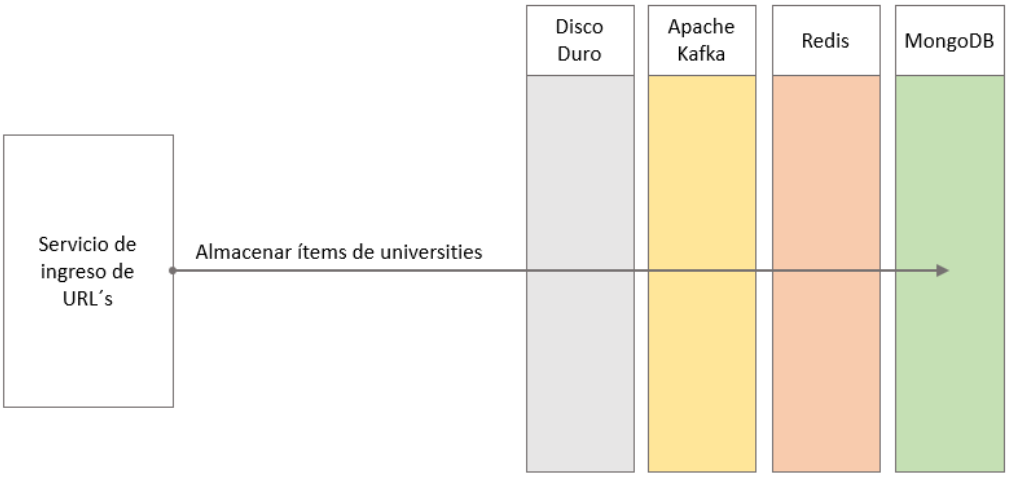


Figura 33. Interacción entre el servicio de ingreso de URL's y la capa de acceso a datos.

Elaborado por: Eduardo Rosero

2.2.2.4 Aplicación de ingreso de URL’s

Este componente de la capa de aplicación del framework permite la interacción para la carga de un archivo Excel con los datos de las universidades. Este elemento interactúa con el componente de ingreso de URL’s de la capa de servicios como se presenta en la **Figura 34**.

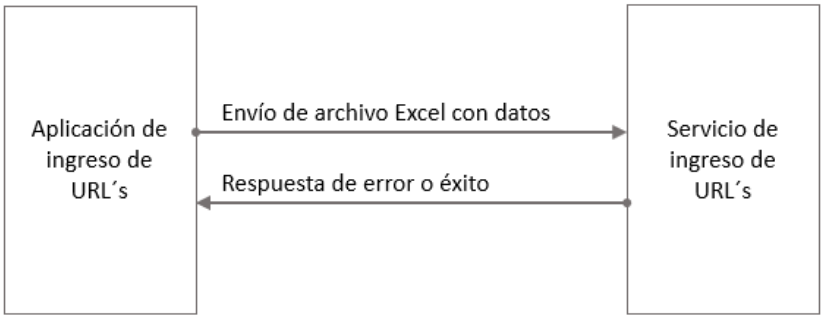


Figura 34. Interacción entre la aplicación de ingreso de URL's y el servicio de ingreso de URL's.

Elaborado por: Eduardo Rosero

2.3 Fase 3

2.3.1 Preparación de los datos

Esta fase permite trabajar con los datos con el fin de prepararlos para el trabajo de análisis principal del proyecto [43]. Los pasos a continuación describen la forma en que los datos fueron procesados para generar un conjunto de datos limpio.

2.3.1.1 Selección de los datos

Un primer paso para la selección de datos consiste en filtrar los válidos. En la colección scrapedPages se almacenan los datos recolectados por el proceso de Web Scraping. En esta colección se tiene el campo valid de tipo booleano que está relacionado con el campo jsonld de tal forma que, cuando el campo valid es verdadero se tiene datos en el campo jsonld. Puesto que este análisis se basa en los datos de marcado estructurado extraído de las páginas Web, se consideró solamente aquellos registros en los que el campo valid es verdadero. Dicho de otra manera, se filtró los registros para mantener aquellos en los que se pudo recuperar marcado estructurado en formato JSON-LD de las páginas Web. La **Tabla 6** detalla el proceso de selección de campos de la colección scrapedPages de acuerdo con su aporte al proyecto.

Tabla 6. Descripción del proceso de selección de campos de la colección scrapedPages para usarlos en data, la nueva colección data creada en MongoDB

Campo	Conservado	Detalle
_id	Sí	Identifica el registro de forma única, por lo tanto, se lo conserva.
domain	Sí	Dominio al cual pertenece el registro. Es relevante para cuestiones de relación y visualización de datos, por lo tanto, se lo conserva.
domainHash	Sí	Establece la relación con el registro de la universidad a la que pertenece en la colección universities. Se considera relevante en caso de que se quiera conocer el nombre de la universidad, por lo tanto, se lo conserva.
url	Sí	URL de la página mediante la cual se generó el registro. Se considera relevante en caso de que se quiera acceder a la página a través de la Web, por lo tanto, se lo conserva.
depth	Sí	Profundidad en la que se obtuvo la página Web. Se considera relevante en caso de que se quiera analizar la cantidad de enlaces que se siguió para llegar a la página Web, por lo tanto, se lo conserva.
processed	No	Determina si el registro ha pasado por un proceso de limpieza y transformación. Se lo usa como filtro de datos. No se considera relevante, por lo tanto, se lo descarta.
valid	No	Indica si el registro contiene datos de marcado estructurado. Se lo usa como filtro de datos. No se considera relevante, por lo tanto, se lo descarta.
createdAt	No	Fecha en la que se almacenó el registro en la base de datos. Se lo usa con propósitos informativos. No se lo considera relevante, por lo tanto, se lo descarta.
processedAt	No	Fecha en la que se realizó el proceso del registro para tareas de limpieza y transformación. Se lo usa con propósitos informativos. No se lo considera relevante, por lo tanto, se lo descarta.
jsonld	Sí	Datos correspondientes al marcado estructurado extraído de la página Web. Tiene especial importancia para el proyecto, por lo tanto, se lo conserva.

Elaborado por: Eduardo Rosero

2.3.1.2 Limpieza de los datos

Esta actividad se enfoca en realizar tareas que mejoren el nivel de calidad de los datos. Algunas acciones comunes que se realizan pueden ser la selección de subconjuntos de datos limpios o inserción de valores ya sea predeterminados o estimados mediante modelos [43].

Teniendo en cuenta que para la generación de datos se tomó varias medidas para asegurar la calidad de estos desde el momento de su extracción y almacenamiento, no se requiere tomar acciones para limpiarlos. No existen problemas de calidad pendientes de resolución, por lo tanto, esta fase puede ser omitida.

2.3.1.3 Construcción de los datos

Para llevar a cabo esta sub-fase es importante entender primero el valor que generará esta construcción de datos al desarrollo del proyecto. Si se determina que las tareas de construcción de datos serán de utilidad para el proyecto, se puede definir qué tipo de operaciones constructivas utilizar. Se puede construir atributos derivados, registros completamente nuevos o incluso transformar valores de atributos ya existentes [43].

Uno de los atributos más importantes de los datos descargados para el proyecto, es jsonld que se encuentra en la colección *scrapedPages*. Este campo se almacena en MongoDB como un objeto tipo lista debido a que en una página Web puede existir más de un elemento de marcado estructurado en formato JSON-LD. La cantidad mínima de elementos que puede contener la lista del atributo jsonld es uno, mientras que el valor máximo dentro de los datos descargados llegó hasta 1485 elementos. Un ejemplo en el cual el campo jsonld tiene dos elementos se muestra en la **Figura 35** en donde se puede visualizar un documento de la colección *scrapedPages*; aquí los campos son representados como claves y cada una de estas claves tiene un valor asociado de acuerdo con los detalles explicados en la **Tabla 4**. Los elementos del campo jsonld a su vez son documentos y cada uno varía de acuerdo con el recurso que describen y el sitio Web del que se extrajeron.

Para trabajar con el contenido del campo jsonld se requirió separar los elementos de la lista en elementos individuales; esta separación se llevó a cabo mediante la creación de un nuevo registro para cada elemento y su almacenamiento en una nueva colección (data) en MongoDB. Estos nuevos documentos se crearon con los campos seleccionados en la **Tabla 6**. Se realizaron dos cambios durante esta separación de elementos: el campo _id de la colección *scrapedPages* se lo almacenó con el nombre de pageId; se creó un nuevo campo _id generado de forma automática desde MongoDB al almacenar los documentos en la base de datos. Además, debido a que se va a analizar el marcado estructurado con el vocabulario Schema, se requirió conservar solamente aquellos registros que contengan dicho vocabulario, por lo cual se agregó una validación para determinar si el subdocumento en jsonld contiene el valor http://schema.org en el campo @context para no descartarlo. Este procedimiento se lo realizó mediante un script de Python el cual se presenta en el diagrama de la **Figura 36**. El ejemplo de la separación de los elementos del campo jsonld en nuevos documentos se presentan en la **Figura 37**. los mismos que fueron generados a partir de los datos presentados en la **Figura 35**.

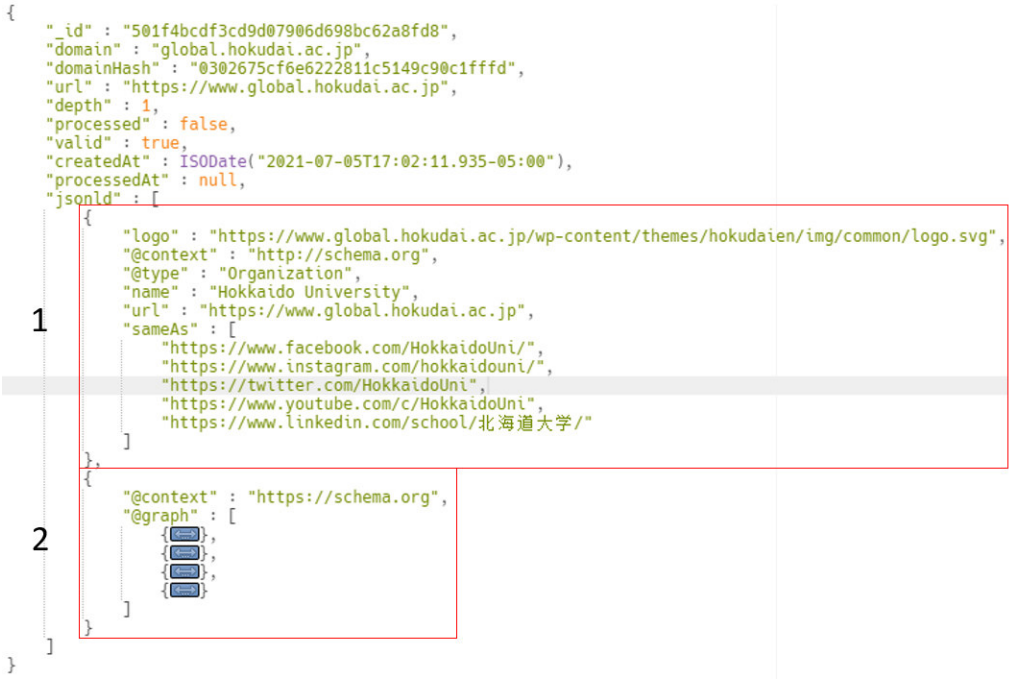


Figura 35. Documento de la colección scrapedPages en que el campo jsonld contiene dos elementos en la lista.

Elaborado por: Eduardo Rosero

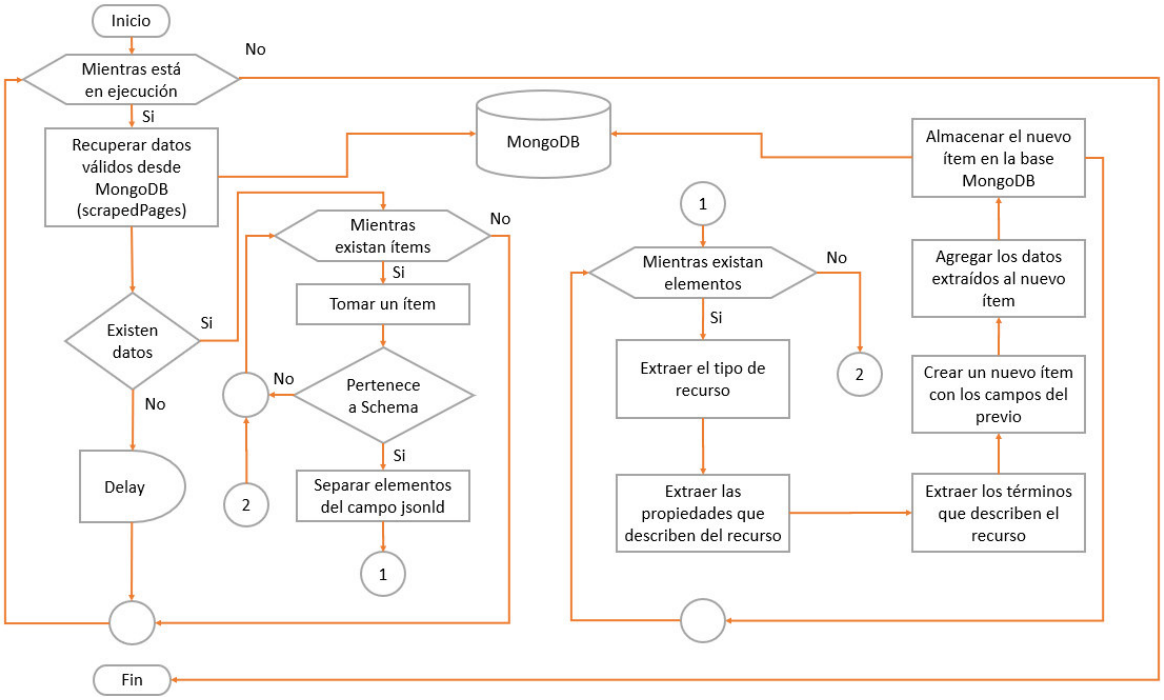


Figura 36. Diagrama de flujo de la separación de elementos de la lista del campo jsonld de la colección scrapedPages.

Elaborado por: Eduardo Rosero

1

```
/* 1 createdAt:7/13/2021, 9:15:43 PM*/
{
    "_id" : ObjectId("60ee48cf0f73eaa125487cea"),
    "pageId" : "501f4bcdf3cd9d07906d698bc62a8fd8",
    "domainHash" : "0302675cf6e6222811c5149c90c1fffd",
    "url" : "https://www.global.hokudai.ac.jp",
    "depth" : 1,
    "jsonld" : {
        "logo" : "https://www.global.hokudai.ac.jp/wp-content/themes/hokudaien/img/common/logo.svg",
        "@context" : "http://schema.org",
        "@type" : "Organization",
        "name" : "Hokkaido University",
        "url" : "https://www.global.hokudai.ac.jp",
        "sameAs" : [
            "https://www.facebook.com/HokkaidoUni/",
            "https://www.instagram.com/hokkaidouni/",
            "https://twitter.com/HokkaidoUni",
            "https://www.youtube.com/c/HokkaidoUni",
            "https://www.linkedin.com/school/北海道大学/"
        ]
    }
},
```

2

```
/* 2 createdAt:7/13/2021, 9:15:43 PM*/
{
    "_id" : ObjectId("60ee48cf0f73eaa125487ceb"),
    "pageId" : "501f4bcdf3cd9d07906d698bc62a8fd8",
    "domainHash" : "0302675cf6e6222811c5149c90c1fffd",
    "url" : "https://www.global.hokudai.ac.jp",
    "depth" : 1,
    "jsonld" : {
        "@context" : "https://schema.org",
        "@graph" : [
            {⟺},
            {⟺},
            {⟺},
            {⟺}
        ]
    }
}
```

Figura 37. Separación de los elementos de la lista del campo jsonld de la colección scrapedPages en documentos individuales en la colección data de MongoDB.

Elaborado por: Eduardo Rosero

A partir de esta separación del campo jsonld, se generaron tres nuevos campos para la colección *data*: el campo *itemType* para almacenar la clase del vocabulario Schema a la que se está haciendo referencia en el recurso descrito por el marcado estructurado; el campo *properties* en el cual se almacena una lista con los nombres de las propiedades utilizadas para describir el recurso; y el campo *terms* obtenido a partir de los las propiedades *name*, *title*, *description* y *headline* del recurso, se trata de la lista de las palabras utilizadas en dichas propiedades para describir el recurso. El diagrama de la **Figura 38** presenta el procedimiento seguido para la extracción de términos en el que se requiere de un listado de *stopwords*[8] tanto en el idioma inglés[9] como en español[10]. La **Figura 39** muestra el ejemplo de un documento completo incluidos los tres nuevos campos.

[8] Se conoce como stopwords a las palabras más comunes de un idioma como artículos, pronombres, preposiciones etc. Y son eliminadas en tareas de procesamiento de lenguaje natural (PNL) por tener poca relevancia en el análisis de texto. https://en.wikipedia.org/wiki/Stop_word

[9] https://gist.github.com/sebleier/554280

[10] https://github.com/Alir3z4/stop-words/blob/master/spanish.txt

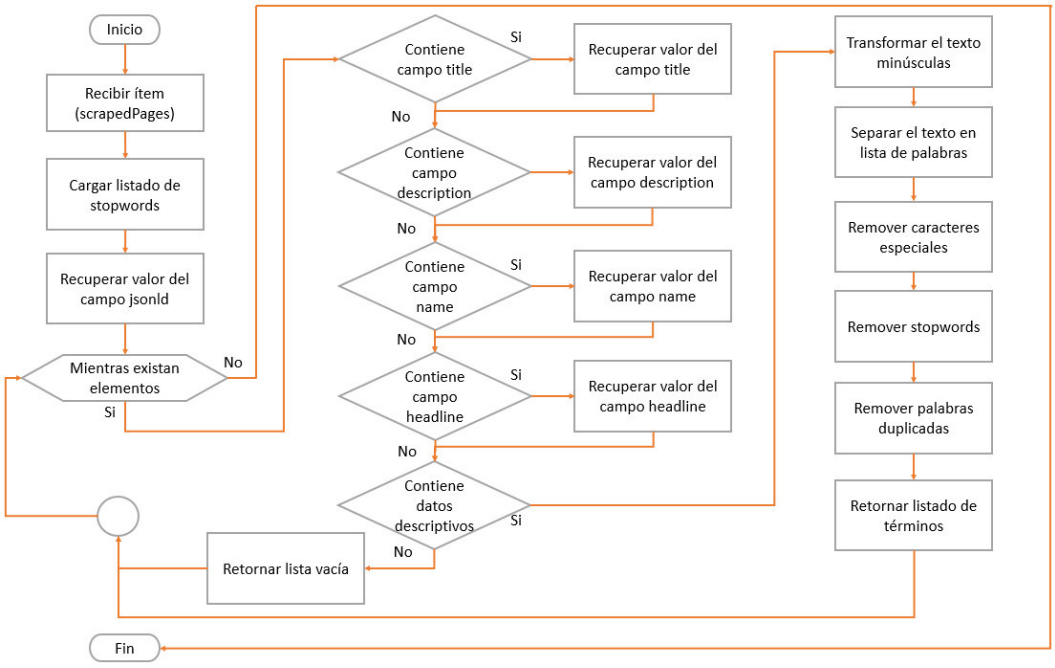


Figura 38. Procedimiento para la extracción de términos para generar el campo terms en la colección data.

Elaborado por: Eduardo Rosero


```
{
    "_id" : ObjectId("610850611105b10643e7f8e2"),
    "pageId" : "715ed2a231565bf9a1bc485998145d9a",
    "domain" : "usc.edu",
    "domainHash" : "0f20a3446b99839682e25209ccda5048",
    "url" : "https://michelson.usc.edu/science-engineering-come-together-improve-lives/",
    "depth" : 1,
    "itemType" : "Article",
    "properties" : [
        "isPartOf",
        "author",
        "headline",
        "datePublished",
        "dateModified",
        "mainEntityOfPage",
        "publisher",
        "articleSection",
        "inLanguage"
    ],
    "terms" : [
        "science",
        "and",
        "engineering",
        "come",
        "together",
        "to",
        "improve",
        "lives"
    ],
    "jsonld" : {
        "@type" : "Article",
        "@id" : "https://michelson.usc.edu/science-engineering-come-together-improve-lives/#article",
        "isPartOf" : {...},
        "author" : {...},
        "headline" : "Science and engineering come together to improve lives",
        "datePublished" : "2017-10-27T21:43:49+00:00",
        "dateModified" : "2017-10-27T22:46:26+00:00",
        "mainEntityOfPage" : {...},
        "publisher" : {...},
        "articleSection" : "Cornerstone",
        "inLanguage" : "en-US",
        "@context" : "https://schema.org"
    }
},
```


Figura 39. Documento completo de la colección data de MongoDB.

Elaborado por: Eduardo Rosero

2.3.2 Aportes de la fase

2.3.2.1 Servicio de procesamiento de datos

El servicio de procesamiento de datos pertenece a la capa de servicios del framework, este permite realizar tareas de limpieza y procesamiento de datos pata prepararlos para su análisis. La **Figura 40** presenta el esquema de comunicación entre el componente del servicio de procesamiento de datos y los componentes de la capa de acceso a datos.

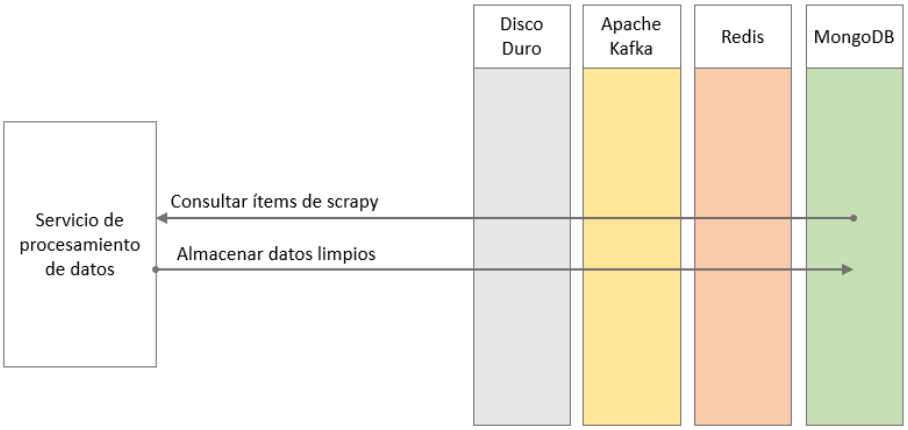


Figura 40. Interacción entre el servicio de procesamiento de datos y la capa de acceso a datos del framework.

Elaborado por: Eduardo Rosero

2.4 Fase 4

2.4.1 Modelado

Normalmente en esta fase se realiza la selección y posterior aplicación de una o varias técnicas de modelado. El problema para resolver puede ser abordado desde diferentes perspectivas por lo que varios modelos pueden ser aplicables para su resolución, la idea es optimizar los modelos seleccionados a fin de generar buenos resultados [43].

En este caso particular no se realizará un modelo, sino que más bien se aplicará técnicas de consulta y agregación de datos para la obtención de indicadores clave, los mismos que se presentarán en un dashboard para una mejor visualización y entendimiento de resultados.

1. **Dashboard**

Puesto que en el trabajo se pretende analizar el uso de marcado estructurado embebido para la descripción de recursos educativos, es importante seleccionar las clases del vocabulario Schema que se asocian con dichos recursos educativos. De acuerdo con [5], los tipos superiores que se usan para describir recursos educativos son: *CreativeWork*, *Book*, *WebPage*, *WebSite*, *Article* y *Course*. A este listado se puede añadir también las clases: *ImageObject*, *VideoObject* y *Thesis* ya que también se usan para representar recursos educativos. El componente dashboard forma parte de la capa de aplicación del framework y se trata de una aplicación Web que permite analizar los datos recolectados. Dicho análisis se realiza mediante indicadores clave y gráficos que resumen los datos y proporcionan información sobre estos. La información que se puede obtener mediante el

dashboard es referente a la descarga general de los datos y al uso de marcado estructurado en los sitios Web de las universidades. Por esta razón el dashboard se divide en dos secciones: la primera corresponde a describir los datos que se han descargado para las universidades seleccionadas, y la segunda se encarga de la descripción del uso de marcado estructurado en los sitios Web de las universidades.

Estos datos pueden ser examinados más a detalle mediante la aplicación de filtros: en la sección de la descarga de los datos se puede analizar los datos de una universidad de forma individual, un grupo de universidades o todo el conjunto de datos. Por otro lado, en la sección del uso de marcado estructurado se puede analizar más a detalle los datos aplicando el filtro de universidades y adicionalmente seleccionando una, varias o todas las clases seleccionadas que se usan para describir recursos educativos. Estos filtros generan cambios en los gráficos de acuerdo con los datos proporcionados por el servicio de consulta de datos.

2. Servicio de consulta de datos

Este componente pertenece a la capa de servicios del framework y es el encargado de comunicarse con la base de datos para realizar consultas y proporcionar datos al dashboard. Una vez que el servicio recibe la petición crea la estructura de una consulta general, luego realiza validaciones para determinar si las consultas a realizar incluirán todos los datos o se deben aplicar filtros. En el caso de que la solicitud no incluya filtros se ejecuta la consulta y se retorna los datos, en caso contrario se modifica la consulta general creada para agregar las sentencias de filtrado. Una vez hecho esto, se realiza la consulta y se retorna los datos. La **Figura 41** muestra el diagrama de las tareas que se ejecutan en el servicio de consulta.

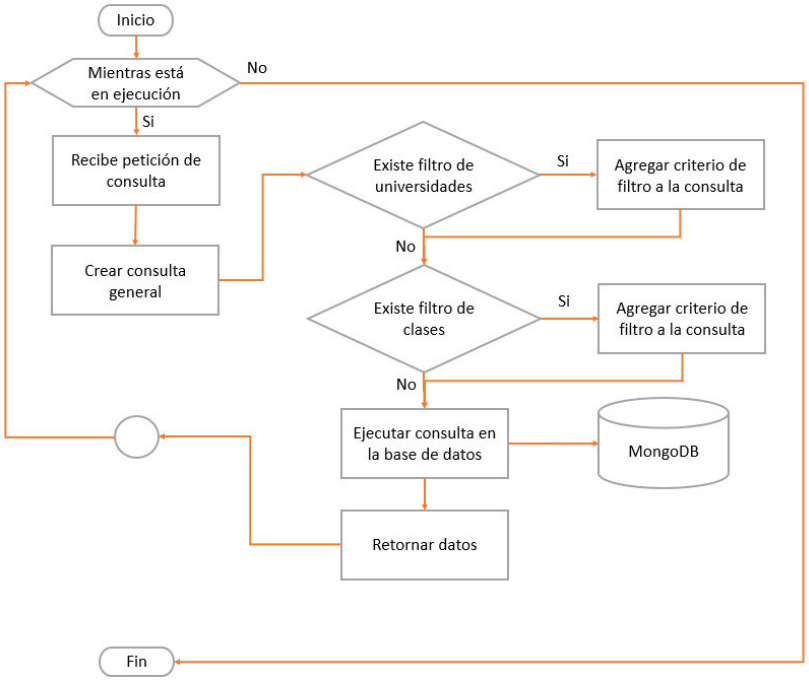


Figura 41. Diagrama de flujo del servicio de consulta de datos.

Elaborado por: Eduardo Rosero

2.4.2 Aportes de la fase

2.4.2.1 Servicio de consulta de datos

El servicio de consulta de datos pertenece a la capa de servicios del framework, este se encarga de realizar las consultas a MongoDB y enviar los resultados al dashboard para su presentación. La **Figura 42** presenta el esquema de comunicación entre el componente del servicio de consulta de datos y los componentes de la capa de acceso a datos.

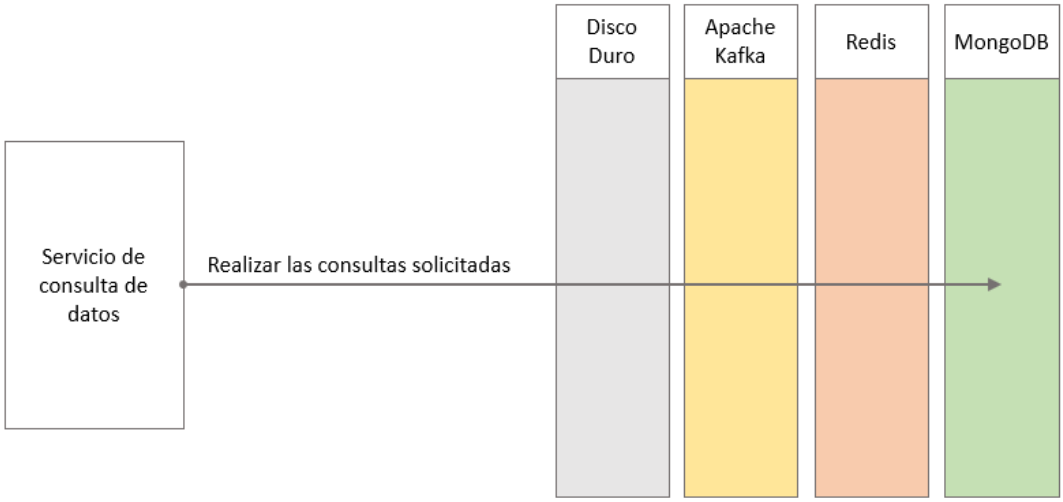


Figura 42. Interacción entre el servicio de consulta de datos y la capa de acceso a datos del framework.

Elaborado por: Eduardo Rosero

2.4.2.2 Dashboard

Este componente pertenece a la capa de aplicación y permite analizar los datos de forma interactiva mediante la aplicación de filtros. Este componente interactúa con el servicio de consulta de datos, el diagrama de la interacción de los componentes se presenta en la **Figura 43**.

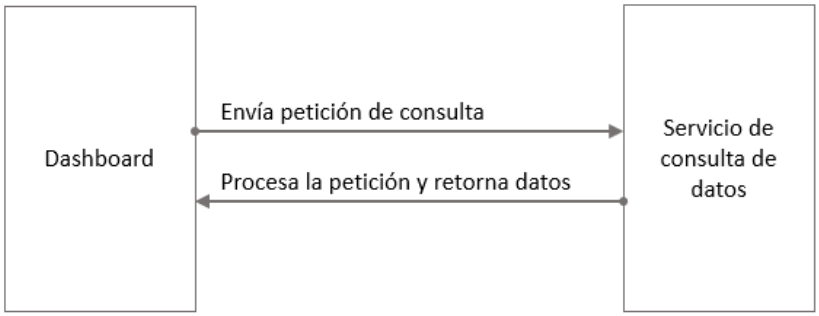


Figura 43. Interacción entre la aplicación del dashboard y el servicio de consulta de datos.

Elaborado por: Eduardo Rosero

2.5 Fase 5

2.5.1 Evaluación

Esta etapa está orientada al análisis de los resultados obtenidos para conocer su utilidad y viabilidad desde un enfoque de análisis de datos, de tal forma que se cumplan los objetivos del proyecto con buenos criterios de calidad [43].

En este caso particular la evaluación no se realiza sobre un modelo sino sobre un framework, el cual se desarrolló en las fases previas, y cuyos componentes se obtuvo a partir de cada una de ellas. Este framework se divide en tres capas: la capa de acceso a datos que comprende los cuatro diferentes componentes de almacenamiento utilizados, la capa de servicios que se compone de los servicios de registro de URL's, descarga de datos, procesamiento de datos y consulta de datos, y la capa de aplicación que tiene el componente de ingreso de URL's y la aplicación del dashboard que permite realizar el análisis de los datos. El esquema del framework se presenta en la **Figura 44**.

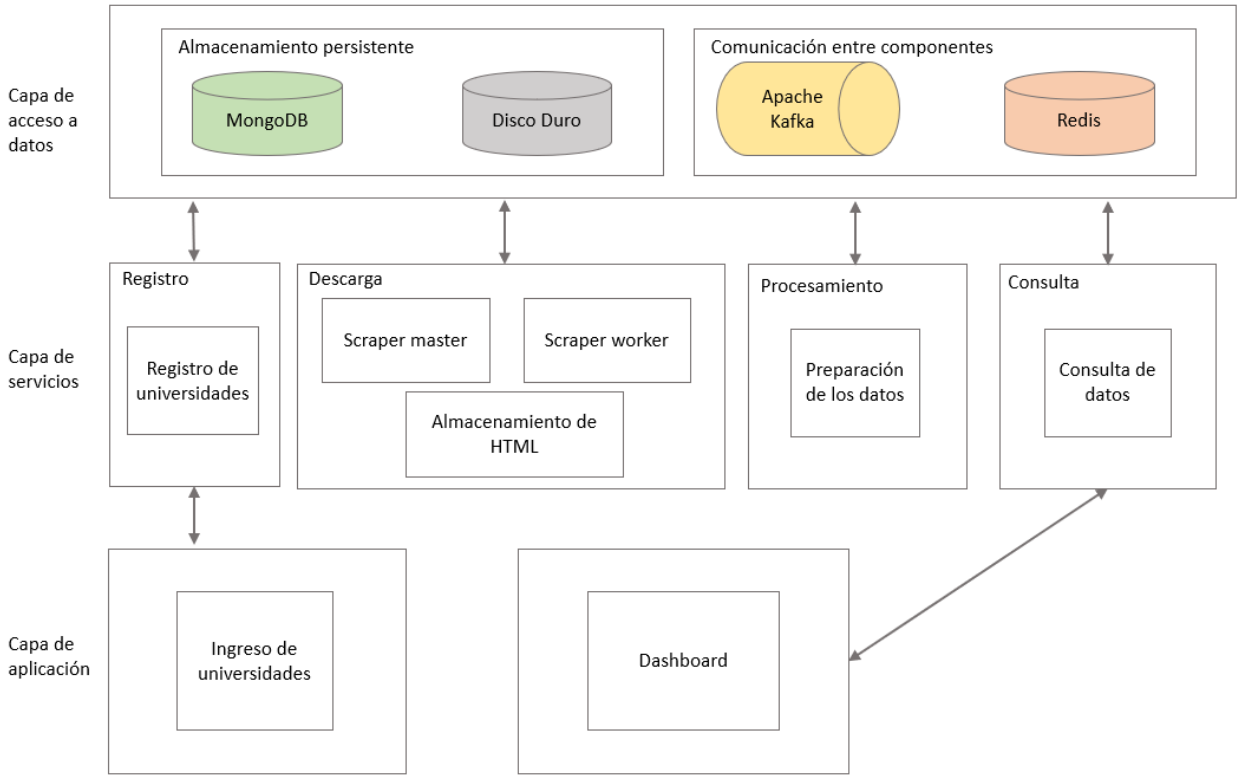


Figura 44. Esquema completo del framework que comprende doce componentes agrupados en tres capas.

Elaborado por: Eduardo Rosero

Para realizar la evaluación del framework, se diseñó un plan de ejecución con el fin de verificar el funcionamiento de cada uno de los componentes que lo conforman. Los criterios de funcionalidad tomados en cuenta para evaluar del framework se presentan en la **Tabla 7**.

Tabla 7. Criterios de evaluación de la funcionalidad del framework

Componente	Tarea	Se cumplió	Observaciones
Aplicación de ingreso de URL's	Acceder a la aplicación	Sí	Sin Novedad
	Cargar archivo	Sí	Sin Novedad
	Enviar archivo a procesamiento	Sí	Sin Novedad
	Visualizar mensaje éxito/error	Sí	Sin Novedad
Servicio de ingreso de URL's	Validar archivo Excel	Sí	Sin Novedad
	Realizar ingreso de datos a MongoDB	Sí	Sin Novedad
Servicio de descarga de datos - Scraper master	Recuperar datos de universidades de MongoDB	Sí	Sin Novedad
	Iniciar proceso de web scraping para las universidades	Sí	Sin Novedad
	Enviar URL's a tópico to-download-urls de Apache Kafka	Sí	Sin Novedad
	Enviar código HTML a tópico to-save-files de Apache Kafka	Sí	Sin Novedad
	Comunicación con Redis	Sí	Sin Novedad
	Marcar ítems de la colección universities como procesados	Sí	Sin Novedad
	Almacenar datos en colección scrapedPages de MongoDB	Sí	Sin Novedad
Servicio de descarga de datos - Scraper worker	Recuperar datos de URL's del tópico to-download-urls de Apache Kafka	Sí	Sin Novedad
	Iniciar proceso de web scraping para las URL's recuperadas	Sí	Sin Novedad
	Enviar código HTML a tópico to-save-files de Apache Kafka	Sí	Sin Novedad
	Establecer comunicación con Redis	Sí	Sin Novedad
	Almacenar datos en colección scrapedPages de MongoDB	Sí	Sin Novedad
Servicio de descarga de datos - Almacenamiento de HTML	Leer datos del tópico to-save-files de Apache Kafka	Sí	Sin Novedad
	Gestionar lógica de creación de directorios en disco duro	Sí	Sin Novedad
	Almacenar los archivos HTML en disco duro	Sí	Sin Novedad
Servicio de procesamiento de datos	Leer datos de la colección scrapedPages de MongoDB	Sí	Sin Novedad

	Generar nuevos datos y almacenarlos en la colección data de MongoDB	Sí	Sin Novedad
	Marcar los ítems de la colección scrapedPages como procesados	Sí	Sin Novedad
Servicio de consulta de datos	Leer datos de las colecciones scrapedPages y data de MongoDB	Sí	Sin Novedad
	Recibir las solicitudes de consulta de datos	Sí	Sin Novedad
	Aplicar filtros si se requiere	Sí	Sin Novedad
	Realizar las consultas y retornar los datos	Sí	Sin Novedad
Dashboard	Acceder a la aplicación	Sí	Sin Novedad
	Visualización de los gráficos	Sí	Sin Novedad
	Gestionar la selección de filtros	Sí	Sin Novedad
MongoDB	Contener la colección universities con datos ingresados	Sí	Sin Novedad
	Contener la colección scrapedPages con datos ingresados	Sí	Sin Novedad
	Contener la colección data con datos ingresados	Sí	Sin Novedad
Apache Kafka	Gestionar tópicos	Sí	Sin Novedad
	Gestionar flujo de datos	Sí	Sin Novedad
Redis	Gestionar ingresos y consultas de datos	Sí	Sin Novedad

Elaborado por: Eduardo Rosero

Para determinar si se cumplen los criterios de funcionalidad para la evaluación del framework, se realizó una valoración completa de este con datos de prueba. A continuación, se presenta el detalle del proceso de ejecución del framework y la validación de sus componentes.

Como primeros pasos se procedió a crear la nueva base de datos *universitiesTest* en MongoDB para almacenar los datos de prueba, se realizó la limpieza de los datos almacenados en Redis y se creó un directorio nuevo en el disco duro para almacenar los archivos HTML de las páginas descargadas. Luego se procedió a configurar los componentes de la capa de servicio que se comunican con la base de datos MongoDB de la capa de acceso a datos para que se conecten a la base de datos de pruebas *universitiesTest*. La **Figura 45** muestra la nueva base de datos *universitiesTest*, la cual está en un inicio vacía sin colecciones ni datos.

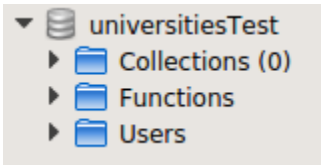


Figura 45. Base de datos vacía creada en MongoDB para pruebas.

Elaborado por: Eduardo Rosero

A pesar de que los componentes pueden trabajar de forma autónoma, estos requieren los datos base para realizar sus diferentes actividades, por lo que la evaluación del framework se realizará desde el punto de inicio el cual se considera el ingreso de las URL's de los sitios Web a analizar, hasta la visualización de los resultados en el dashboard. La secuencia de ejecución y evaluación del framework se detalla a continuación:

1. Preparar el archivo Excel

Para ingresar los datos de las universidades de las cuales se va a descargar datos para la evaluación se preparó el archivo Excel con los datos que se presentan en la **Figura 46**, en donde se puede apreciar que existen tres registros correspondientes a las universidades ecuatorianas: Escuela Politécnica Nacional, Universidad de las Fuerzas Armadas y Universidad Andina Simón Bolívar.

A	B
universidad	url
Escuela Politécnica Nacional	https://www.epn.edu.ec/
Universidad de las Fuerzas Armadas	https://www.espe.edu.ec/
Universidad Andina Simón Bolívar	https://www.uasb.edu.ec/

Figura 46. Esquema del archivo Excel con los datos de tres universidades para las pruebas del framework.

Elaborado por: Eduardo Rosero

2. Carga y procesamiento del archivo

Una vez listo el archivo con los datos de las universidades a analizar, este se utiliza en la aplicación de ingreso de URL's que corresponde a un componente de la capa de aplicación del framework. Dentro de la aplicación se realizó la carga del archivo y el envío para su procesamiento. La selección del archivo se puede apreciar en la **Figura 47**, mientras que la **Figura 48** presenta el detalle del archivo cargado y listo para enviarlo a procesar.

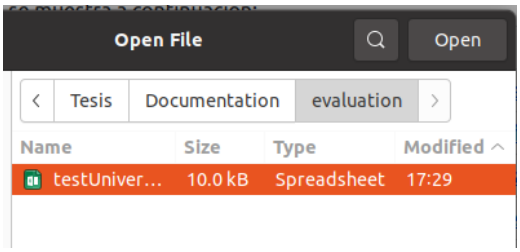


Figura 47. Selección del archivo de Excel con los datos de prueba.

Elaborado por: Eduardo Rosero

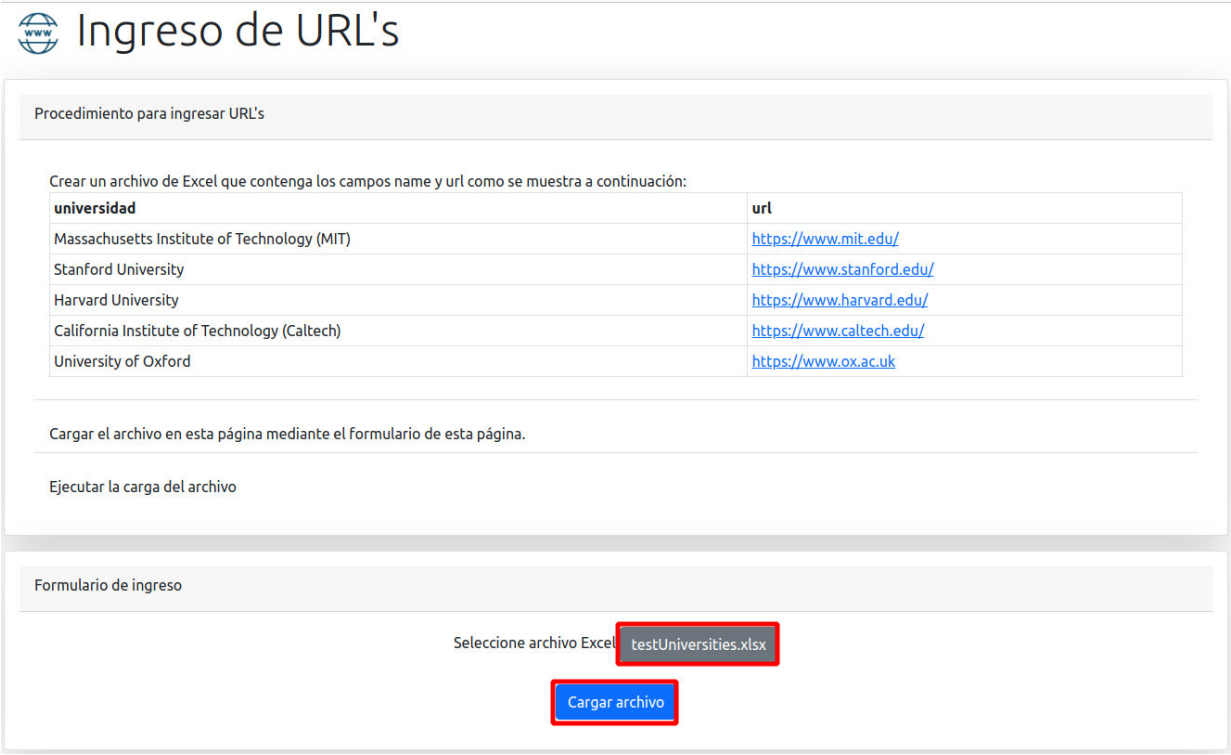


Figura 48. Archivo de Excel cargado en el formulario Web para enviarlo a procesar.

Elaborado por: Eduardo Rosero

El procesamiento se realizó en el servicio de ingreso de URL's, componente de la capa de servicio del framework, en el cual se agregaron logs[11] para visualizar parte del procesamiento, de modo que se pueda informar sobre la cantidad de registros que se ingresarán en la base de datos y la estructura que estos van a tener como se muestra en la **Figura 49**.

[11] https://es.wikipedia.org/wiki/Log_(inform%C3%A1tica)

```
[18-Aug-21 17:51:15.524] - [693363-UniversitiesService] - INFO - Se van a ingresar [3] registros de universidades
[18-Aug-21 17:51:15.524] - [693363-UniversitiesService] - INFO -
{
    "_id": "f9944cb4cd507d2d19020f4c6154a63b",
    "name": "Escuela Polit\u00e9cnica Nacional",
    "url": "https://epn.edu.ec/",
    "processed": false
}
[18-Aug-21 17:51:15.552] - [693363-UniversitiesService] - INFO -
{
    "_id": "e5d1c94d20a16edc5ca5fdc5f4b4caea",
    "name": "Universidad de las Fuerzas Armadas ",
    "url": "https://www.espe.edu.ec/",
    "processed": false
}
[18-Aug-21 17:51:15.554] - [693363-UniversitiesService] - INFO -
{
    "_id": "97a02fc7fb2dee99e462e262d8b43a43",
    "name": "Universidad Andina Sim\u00f3n Bol\u00edvar",
    "url": "https://www.uasb.edu.ec/",
    "processed": false
}
```


Figura 49. Logs del servicio de ingreso de URL's agregados con fines informativos y de monitoreo.

Elaborado por: Eduardo Rosero

Puesto que el archivo tenía la estructura correcta se obtuvo el mensaje de éxito en el procesamiento de este el cual se puede apreciar en la **Figura 50**.

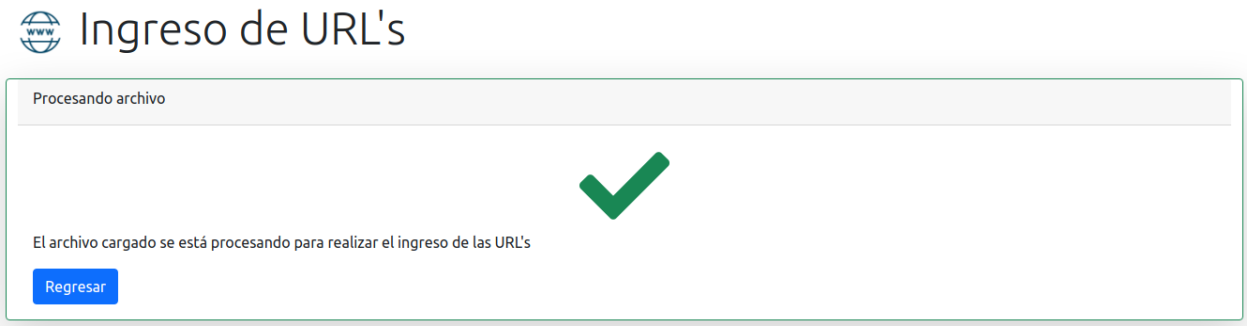


Figura 50. Mensaje de éxito recibido luego de procesar el archivo Excel con datos de prueba.

Elaborado por: Eduardo Rosero

3. Verificar el ingreso de datos en MongoDB

Una vez que se completa el proceso de ingreso de URL's tanto en la capa de aplicación como en la capa de servicios, para verificar que todo se haya ejecutado de manera correcta, se puede verificar que en la nueva base de datos se ha creado la colección *universities* que contiene tres documentos, correspondientes a los datos de las universidades ingresadas. La **Figura 51** muestra la nueva colección y los tres registros en la base de datos.

Figura 51. Colección universities de la base de datos de pruebas de MongoDB con los datos de las universidades de prueba.

Elaborado por: Eduardo Rosero

4. Verificar la ejecución del subcomponente scraper master

Para comprobar que este subcomponente se ejecuta de forma correcta, también se agregó logs que permitan visualizar las tareas que se están ejecutando. Es así como podemos visualizar cuando inicia el proceso de web scraping para los datos de las universidades ingresadas como se visualiza en la **Figura 52**.

```
[19-Aug-21 01:03:57.402] - [12-ScrapyLeaderService] - INFO - Se recuperaron [3] universidades para procesar
[19-Aug-21 01:03:57.449] - [12-ScrapyLeaderService] - INFO - INICIO DEL PROCESO DE SCRAPY
[19-Aug-21 01:03:57.450] - [12-ScrapyLeaderService] - INFO - Programando scrapy para [https://www.espe.edu.ec]
[19-Aug-21 01:03:57.450] - [12-ScrapyLeaderService] - INFO - Programando scrapy para [https://epn.edu.ec]
[19-Aug-21 01:03:57.450] - [12-ScrapyLeaderService] - INFO - Programando scrapy para [https://www.uasb.edu.ec]
[19-Aug-21 01:03:57.466] - [16-scrapy.utils.log] - INFO - Scrapy 2.5.0 started (bot: tuteggito)
```

Figura 52. Logs de la ejecución del componente scraper master del servicio de descarga de datos agregados con fines informativos y de monitoreo.

Elaborado por: Eduardo Rosero

Una vez iniciado este proceso de descarga de datos, se puede observar también que se crea la colección *scrapedPages* en la base de datos MongoDB en la que se almacenan los datos descargados tanto por el componente scraper master y el componente scraper worker. La **Figura 53** muestra la base de datos con la colección *scrapedPages* y los primeros datos ingresados.

Figura 53. Colección scrapedPages de la base de datos de pruebas de MongoDB con los datos de las páginas Web descargadas.

Elaborado por: Eduardo Rosero

5. Verificar la ejecución del subcomponente scraper worker

Este subcomponente que forma parte del servicio de descarga de datos requiere del componente Apache Kafka y de que existan datos en el tópico *to-download-urls*, el cual es alimentado con datos a partir del componente scraper master. La **Figura 54** muestra el estado del tópico *to-download-urls* con tres particiones cada una con un componente scraper worker conectado. Se puede apreciar también la cantidad de datos asignada a cada partición en la columna *LOG-END-OFFSSET* y la cantidad de datos por leer en la columna *LAG*.

```
Every 2,0s: kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group g1-scrapy

GROUP           TOPIC            PARTITION  CURRENT-OFFSET  LOG-END-OFFSET  LAG             CONSUMER-ID                                     HOST            CLIENT-ID
g1-scrapy       to-download-urls 1          0               71              71              scrapy-m1-2-c070e88b-239c-4582-b3d8-13b3f2ced612 /172.19.0.1     scrapy-m1-2
g1-scrapy       to-download-urls 2          0               52              52              scrapy-m1-3-85d2e360-5ce9-498b-b757-ec9bb6c370d1 /172.19.0.1     scrapy-m1-3
g1-scrapy       to-download-urls 0          0               67              67              scrapy-m1-1-2e5b4d6d-570e-49da-882e-e581692932c2 /172.19.0.1     scrapy-m1-1
```

Figura 54. Datos informativos sobre el estado del tópico to-download-urls de Apache Kafka.

Elaborado por: Eduardo Rosero

Cada componente scraper worker de los tres existentes en este caso, lee datos del tópico de Apache Kafka y a su vez comienza un nuevo proceso de web scraping. La ejecución de esta tarea se puede visualizar también gracias a los logs generados por el componente como se aprecia en la **Figura 55**, en donde se indica que se recuperaron 7 URL's todas correspondientes al dominio uasb.edu.ec.

```
tuteggito@ragnarok:~$ docker logs -f uni-worker-1
[19-Aug-21 01:59:49.363] - [1-AppConfiguration] - INFO - Perfil [workerEval] seleccionado como activo desde la variable de entorno [PROFILE_ACTIVE]
[19-Aug-21 01:59:51.231] - [1-ScrapyWorkerService] - INFO - Iniciando servicio de scrapy
[19-Aug-21 01:59:56.849] - [1-ScrapyWorkerService] - INFO - Se recuperaron [7] urls para procesar
[19-Aug-21 01:59:21.368] - [1-ScrapyWorkerService] - INFO - INICIO DEL PROCESO DE SCRAPY
[19-Aug-21 01:59:21.371] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/noticias]
[19-Aug-21 01:59:21.371] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/biblioteca]
[19-Aug-21 01:59:21.371] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/casa-andina]
[19-Aug-21 01:59:21.371] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/universidad/centros-y-programas]
[19-Aug-21 01:59:21.371] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/admisiones/posgrados-2021-segunda-llamada]
[19-Aug-21 01:59:21.371] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/admisiones/costos-y-ayudas-financieras]
[19-Aug-21 01:59:21.372] - [1-ScrapyWorkerService] - INFO - Programando scrapy para [https://www.uasb.edu.ec/publicacion]
[19-Aug-21 01:59:21.441] - [8-scrapy.utils.log] - INFO - Scrapy 2.5.0 started (bot: tuteggito)
```

Figura 55. Logs de la ejecución del componente scraper worker del servicio de descarga de datos agregados con fines informativos y de monitoreo

Elaborado por: Eduardo Rosero

6. Verificar ejecución del servicio de almacenamiento de archivos HTML

Este subcomponente perteneciente al servicio de descarga también interactúa con Apache Kafka debido a que lee datos del tópico *to-save-files* al cual se envían los datos a partir de los subcomponentes scraper master y scraper worker. La **Figura 56** muestra el estado del tópico *to-save-files*, en donde se puede apreciar que se han procesado 3 registros para el almacenamiento de archivos HTML de acuerdo con la columna *CURRENT-OFFSET* y que aún no existen más datos para procesar según la columna *LAG*.

```
Every 2,0s: kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group g1-files

GROUP           TOPIC           PARTITION  CURRENT-OFFSET  LOG-END-OFFSET  LAG             CONSUMER-ID                                     HOST            CLIENT-ID
g1-files        to-save-files   0          3               3               0               scrapy-m1-1-f9e994e9-4676-4014-8211-d8f5c1cba0ce /172.19.0.1     scrapy-m1-1
```

Figura 56. Datos informativos sobre el estado del tópico to-save-files de Apache Kafka

Elaborado por: Eduardo Rosero

Para visualizar la ejecución de la tarea de almacenamiento dentro del subcomponente que almacena los archivos HTML, se agregaron logs que indican la cantidad de elementos que se recuperaron de Apache Kafka para procesarlos. Estos logs se muestran en la **Figura 57**.

```
[19-Aug-21 02:03:26.548] - [12-Main-Process] - INFO - Se recuperaron [1] items para guardar el archivo
[19-Aug-21 02:03:26.571] - [12-Main-Process] - INFO - Se recuperaron [11] items para guardar el archivo
[19-Aug-21 02:03:26.596] - [12-Main-Process] - INFO - Esperando items para guardar archivos
[19-Aug-21 02:03:31.601] - [12-Main-Process] - INFO - Se recuperaron [1] items para guardar el archivo
[19-Aug-21 02:03:31.616] - [12-Main-Process] - INFO - Se recuperaron [11] items para guardar el archivo
[19-Aug-21 02:03:31.640] - [12-Main-Process] - INFO - Se recuperaron [8] items para guardar el archivo
[19-Aug-21 02:03:31.669] - [12-Main-Process] - INFO - Se recuperaron [11] items para guardar el archivo
[19-Aug-21 02:03:31.702] - [12-Main-Process] - INFO - Se recuperaron [11] items para guardar el archivo
[19-Aug-21 02:03:31.726] - [12-Main-Process] - INFO - Se recuperaron [2] items para guardar el archivo
[19-Aug-21 02:03:31.734] - [12-Main-Process] - INFO - Esperando items para guardar archivos
[19-Aug-21 02:03:36.737] - [12-Main-Process] - INFO - Se recuperaron [1] items para guardar el archivo
[19-Aug-21 02:03:36.751] - [12-Main-Process] - INFO - Se recuperaron [11] items para guardar el archivo
[19-Aug-21 02:03:36.785] - [12-Main-Process] - INFO - Se recuperaron [7] items para guardar el archivo
[19-Aug-21 02:03:36.812] - [12-Main-Process] - INFO - Se recuperaron [1] items para guardar el archivo
[19-Aug-21 02:03:36.820] - [12-Main-Process] - INFO - Esperando items para guardar archivos
```

Figura 57. Logs del servicio de almacenamiento de archivos HTML agregados con fines informativos y de monitoreo.

Elaborado por: Eduardo Rosero

Este subcomponente de almacenamiento también interactúa con el disco duro puesto que almacena los datos en forma de archivos. La **Figura 58** muestra los tres directorios creados, correspondientes a las universidades seleccionadas para el análisis, mientras que la **Figura 59** muestra algunos ejemplos de archivos almacenados de una de estas universidades.

Figura 58. Directorios creados para el almacenamiento de documentos HTML de los sitios Web de las tres universidades de prueba.

Elaborado por: Eduardo Rosero

7. Verificar la ejecución del servicio de procesamiento de datos

Este componente de la capa de servicios del framework requiere de los datos de la colección *scrapedPages* en MongoDB la cual se llena con datos a partir de los componentes scraper master y scraper worker. Para la visualización de las tareas que se ejecutan en este componente se agregaron logs informativos como se puede apreciar en la **Figura 60**.

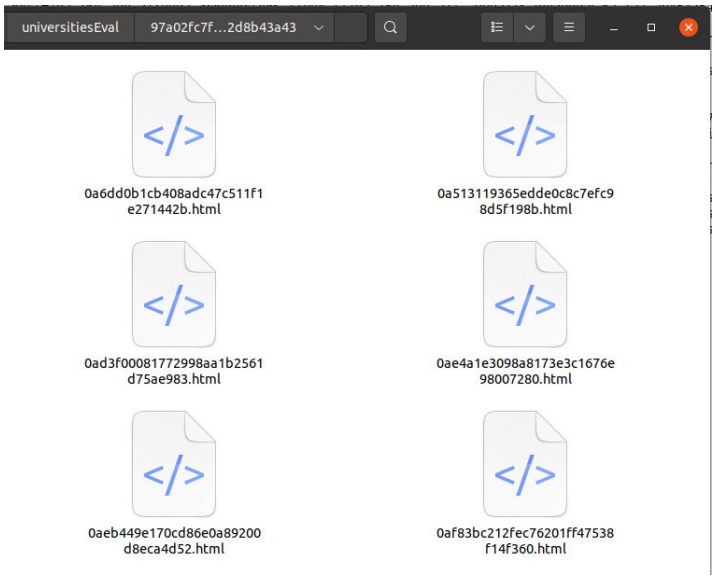


Figura 59. Documentos HTML almacenados para una de las universidades de prueba.

Elaborado por: Eduardo Rosero

```
[18-Aug-21 21:08:29.123] - [847030-ProcessItemsService] - INFO - Se recuperaron [500] para procesar
Se recuperaron [1] elementos para el id [14a0454ba4d05f599fb139206b93fb27]
Se recuperaron [1] elementos para el id [24bdbadbb07c1aac1c1fc4ae362e500e]
Se recuperaron [1] elementos para el id [6fccdfd751d877aaf6c38c0794f3ca82]
Se recuperaron [1] elementos para el id [59431bfa3464cf63644f20bd4a050414]
Se recuperaron [1] elementos para el id [34999ec15b9d6291a2015df5634caafa]
```

Figura 60. Logs del servicio de procesamiento de datos agregados con fines informativos y de monitoreo.

Elaborado por: Eduardo Rosero

Este componente además crea la colección *data* en la base de datos luego de procesar los datos de la colección *scrapedPages*. Esta nueva colección con un ejemplo de los datos que contiene se puede visualizar en la **Figura 61**.

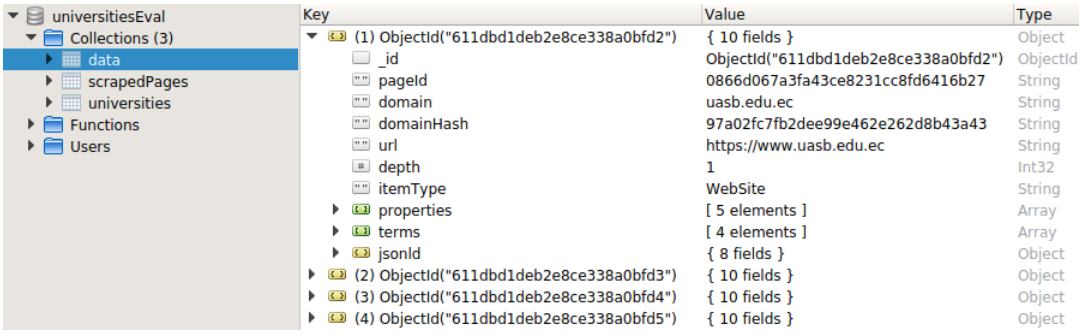


Figura 61. Colección data de la base de datos de pruebas de MongoDB con los datos generados a partir del componente de procesamiento de datos.

Elaborado por: Eduardo Rosero

8. Verificar la ejecución del dashboard

Este componente de la capa de aplicación del framework requiere del servicio de consulta de datos. Para verificar que el servicio de consulta de datos trabaja de forma adecuada basta con revisar el funcionamiento del dashboard, ya que este último requiere de datos para generar las visualizaciones, es decir que, si el servicio de consulta no se ejecuta de forma adecuada, no se tendría datos para presentar el dashboard. La **Figura 62** muestra la sección del dashboard correspondiente a los aspectos generales de los datos descargados generada a partir de los datos de prueba descargados, mientras que la **Figura 63** muestra la sección del dashboard correspondiente al análisis del uso del vocabulario Schema con el formato JSON-LD generada a partir del procesamiento de los datos descargados.

Figura 62. Visualización de la sección del dashboard correspondiente a los datos generales sobre los datos descargados.

Elaborado por: Eduardo Rosero

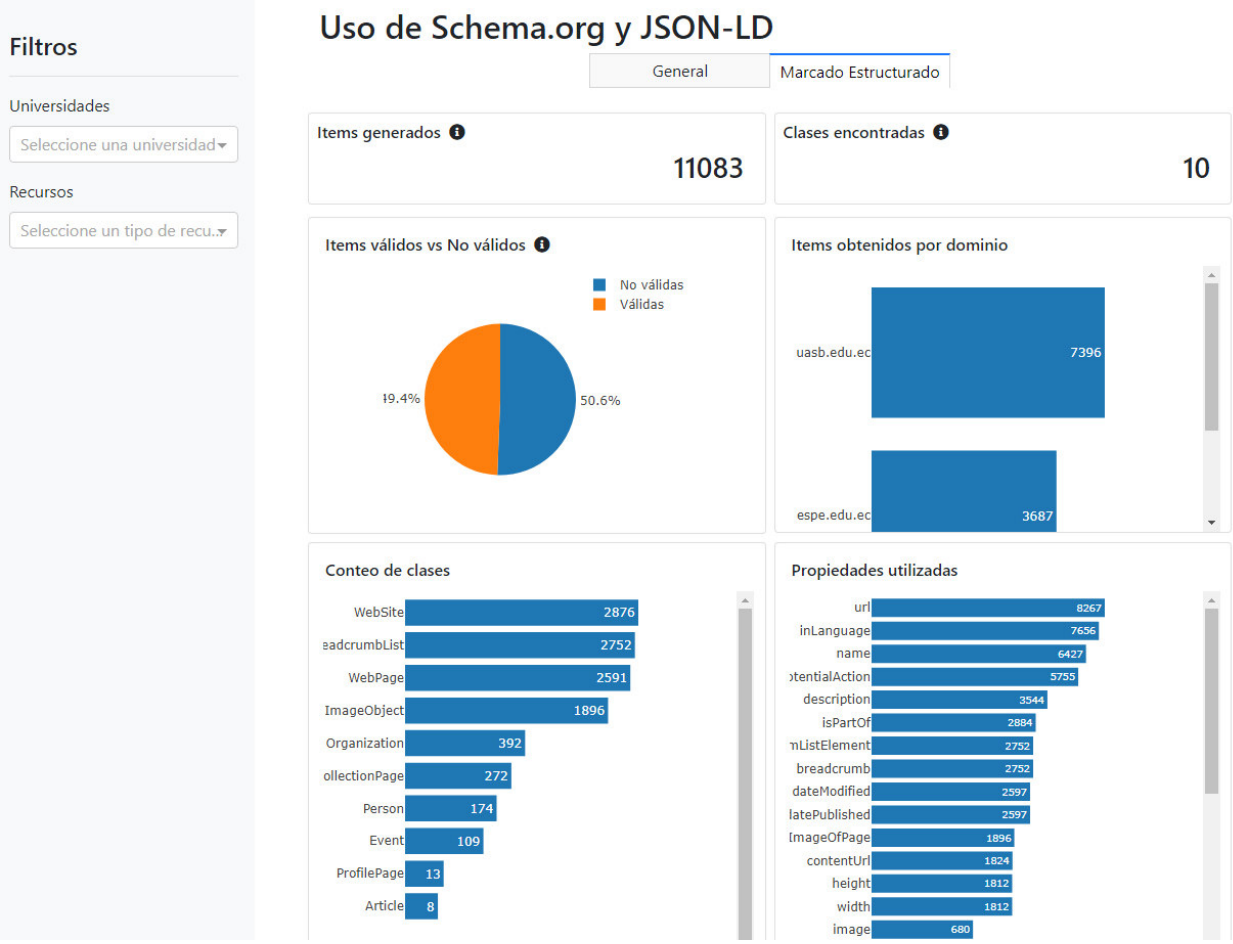


Figura 63. Visualización de la sección del dashboard correspondiente al uso del vocabulario Schema con el formato JSON-LD.

Elaborado por: Eduardo Rosero

3 RESULTADOS Y DISCUSIÓN

Una vez presentado el diseño y desarrollo del framework así como su evaluación, se puede mencionar aspectos relevantes del proceso de acuerdo con los resultados obtenidos. Estos resultados corresponden a la información generada sobre la base del análisis de los datos conseguidos a partir del framework.

El análisis se realizó mediante un dashboard sencillo diseñado para presentar información específica, tanto aquella relacionada con los datos descargados como aquella que se enfoca en el uso de marcado estructurado embebido con el vocabulario Schema y el formato JSON-LD. La aplicación Web del dashboard se compone de dos secciones, una de filtros que permite la interacción del usuario y otra de visualizaciones que resume y presenta detalles de los datos analizados.

3.1 Descarga general de datos

Para el proyecto se consideró el análisis de los sitios Web de 100 universidades de las 150 seleccionadas del top ranking de universidades mundiales proporcionado por QS World University Rankings, conservando aquellas de las que se extrajo una mayor cantidad de datos. De estas universidades seleccionadas se logró descargar 1019268 páginas Web de las cuales 195098 correspondientes al 19.1% del total de páginas, se las considera como válidas puesto que hacen uso del vocabulario Schema y el formato JSON-LD para la descripción de recursos. El 80.9% restante corresponde a 824170 páginas que no usan Schema o JSON-LD por lo que se las considera como no válidas para el análisis. La **Figura 64** presenta el gráfico de estos resultados.

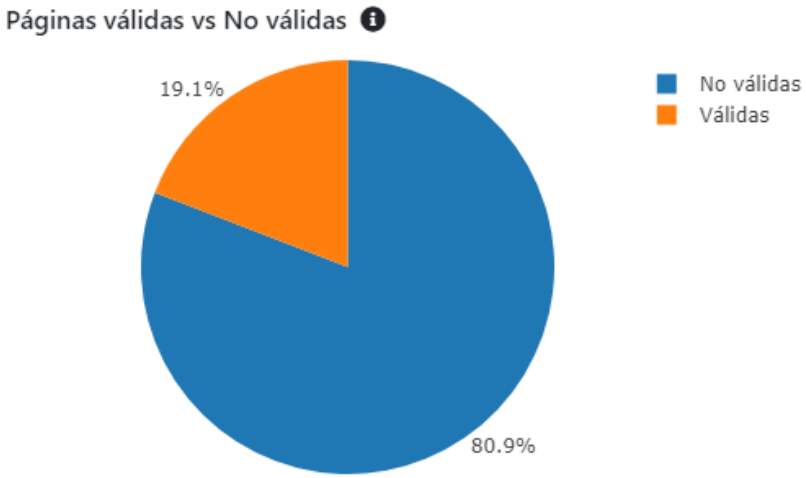


Figura 64. Porcentaje de páginas válidas y no válidas descargadas.

Elaborado por: Eduardo Rosero

Existe variación de la cantidad de páginas Web descargadas de cada sitio, de esto se puede destacar que los tres sitios Web de las universidades de los que se descargó mayor cantidad de páginas fueron:

- University of Southern California (https://www.usc.edu/) con 39950 páginas
- University of Oxford (https://www.ox.ac.uk/) con 39791 páginas
- Harvard University (https://www.harvard.edu/) con 34761 páginas

Debido a que algunos sitios Web gestionan sus páginas Web con una estructura más dinámica basada en JavaScript, se presenta la dificultad en la recuperación de enlaces hacia otras páginas. Por estas y otras razones, se tiene casos en los que solamente se logró descargar una página Web, correspondiente a la página principal del sitio Web. La **Figura 65** muestra el ejemplo de la página principal del sitio Web de Osaka University, en la que se puede apreciar que el cuerpo del documento HTML está compuesto solamente de etiquetas *script* las mismas que contienen el código JavaScript que se ejecuta para mostrar el contenido.

```
<!DOCTYPE html> == $0
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
▶<head>…</head>
▼<body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  ▶<div id="root">…</div>
  ▶<script>…</script>
  ▶<script>…</script>
  <script src="https://www.osaka-u.ac.jp/++resource++ou_www_en_resource/static/js/2.f3578c37.chunk.js"></script>
  <script src="https://www.osaka-u.ac.jp/++resource++ou_www_en_resource/static/js/main.9e05b4b8.chunk.js"></script>
  ▶<script type="text/javascript">…</script>
  <script src="https://ssl.google-analytics.com/ga.js" type="text/javascript"></script>
  ▶<script type="text/javascript">…</script>
</body>
</html>
```

Figura 65. Documento HTML que hace uso de JavaScript para mostrar el contenido de forma dinámica.

Elaborado por: Eduardo Rosero

Con estos precedentes, los tres sitios Web de los cuales se obtuvo solamente una página en el proceso de descarga fueron:

- Osaka University (https://www.osaka-u.ac.jp/en)
- Universidad de Buenos Aires (https://www.uba.ar/)
- University of Science and Technology of China (http://en.ustc.edu.cn/)

3.2 Uso de Schema y JSON-LD para agregar marcado estructurado

De los datos generales descargados se tomó aquellos válidos, es decir 195098 documentos para su procesamiento y análisis. A partir de estos datos, luego del procesamiento para recuperar los recursos descritos usando Schema y JSON-LD se obtuvieron 645613 elementos los cuales están descritos por 47 clases del vocabulario Schema.

Tomando en cuenta que para el análisis de este trabajo se consideraron las clases: CreativeWork, WebSite, Article, Course, Book, WebPage, ImageObject, VideoObject y Thesis como válidos para la descripción de recursos educativos, se tiene que la cantidad de elementos válidos son 260705 correspondiente al 40.4%, mientras que aquellos recursos diferentes a las clases mencionadas son 384908 correspondiente al 59.6%. La **Figura 66** estos datos de forma gráfica.

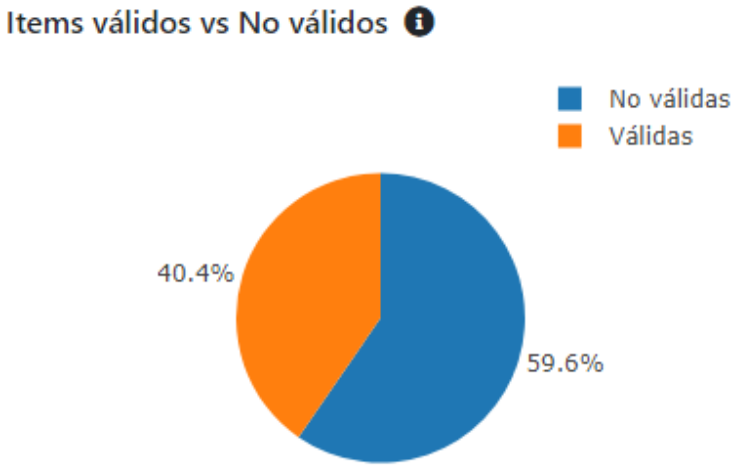


Figura 66. Porcentaje de recursos válidos y no válidos obtenidos a partir de los datos descargados.

Elaborado por: Eduardo Rosero

De estas clases válidas, la que se publica en mayor cantidad corresponde a WebSite mientras que Thesis es la que menos se publica como se puede observar en la **Figura 67**.

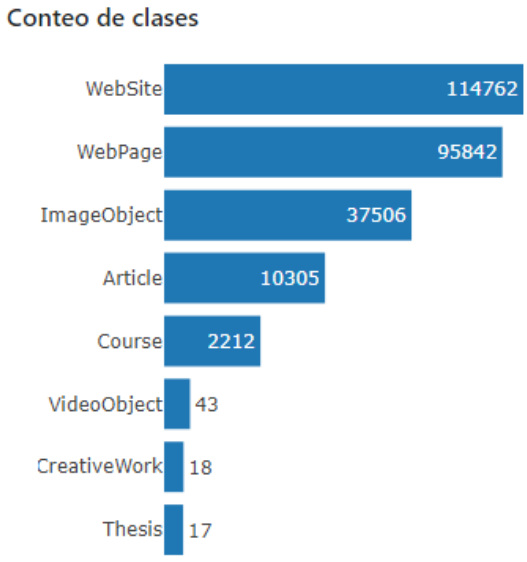


Figura 67. Cantidad total de recursos válidos obtenidos a partir de los datos descargados.

Elaborado por: Eduardo Rosero

La **Tabla 8** presenta el detalle de las propiedades utilizadas para describir los recursos asociados a las clases seleccionadas para este análisis. Se puede observar que la propiedad *url* es la que se usa con mayor frecuencia en la descripción de recursos educativos.

Tabla 8. Propiedades utilizadas para la descripción de recursos

Propiedad	**Cantidad de recursos**	**Porcentaje**
url	245948	15,029%
name	214718	13,120%
inLanguage	204371	12,488%
potentialAction	167489	10,234%
description	125014	7,639%
datePublished	103003	6,294%
dateModified	102989	6,293%
isPartOf	102465	6,261%
breadcrumb	59719	3,649%
publisher	57983	3,543%
primaryImageOfPage	48058	2,937%
height	32986	2,016%
width	32986	2,016%
author	28126	1,719%
contentUrl	21274	1,300%
image	19044	1,164%
caption	18098	1,106%
mainEntityOfPage	11137	0,681%
headline	10535	0,644%
articleSection	8545	0,522%
creator	4782	0,292%
wordCount	3037	0,186%
keywords	2925	0,179%
commentCount	2595	0,159%
thumbnailUrl	2594	0,159%
provider	1275	0,078%
courseCode	1063	0,065%
occupationalCredentialAwarded	779	0,048%
numberOfCredits	779	0,048%
about	625	0,038%
coursePrerequisites	196	0,012%
isAccessibleForFree	189	0,012%
Provider	158	0,010%
identifier	146	0,009%
encoding	141	0,009%
educationalCredentialAwarded	125	0,008%
educationalLevel	125	0,008%
articleBody	110	0,007%

copyrightHolder	103	0,006%
speakable	58	0,004%
uploadDate	43	0,003%
text	42	0,003%
mainContentOfPage	37	0,002%
duration	34	0,002%
abstract	18	0,001%
dateCreated	17	0,001%
workExample	17	0,001%
hasCourseInstance	12	0,001%
embedUrl	7	0,000%
alternateName	6	0,000%
mainEntity	6	0,000%

Elaborado por: Eduardo Rosero

3.3 Discusión

De acuerdo con las pruebas realizadas se puede evidenciar que los resultados obtenidos se alinean con los objetivos del proyecto. El framework y los componentes que lo conforman permitieron realizar la descarga de los datos, su procesamiento y análisis.

Por otro lado, luego de analizar el uso del vocabulario Schema y JSON-LD en los datos obtenidos, se puede evidenciar que sí se hace uso de estas tecnologías, aunque no en la medida que se esperaría. Además, aunque se encontró las clases de Schema que comúnmente se usan para la descripción de recursos educativos, las propiedades generales con que se los describe no aportan los datos necesarios que definan la semántica adecuada para un recurso educativo.

Existen tres ejes que ayudan a definir de forma más precisa un recurso educativo: el valor educativo, la licencia y la accesibilidad [5]. Cada uno de estos ejes cuenta con sus propiedades como se describe en la **Tabla 9**. Si bien estas propiedades son más específicas de un contexto educativo, no se evidenció su uso en los recursos encontrados en los sitios Web de las universidades analizadas.

Tabla 9. Ejes y propiedades que definen de forma específica un recurso educativo

Eje	Propiedad	Descripción
Valor educativo	educationalAlignment	La alineación de materiales con estándares educativos.
	educationalUse	El contexto pedagógico.
	timeRequired	El tiempo promedio para consumir el recurso.
	typicalAgeRange	La edad común recomendada para consumir el recurso.
	learningResourceType	El tipo de recurso predominante.
	interactivityType	La interacción esperada del alumno con el recurso.
	targetDescription	La descripción de un nodo en un marco educativo definido.

	targetName	El nombre de un nodo en un marco educativo definido.
	targetURL	La URL de un nodo en un marco educativo definido.
Licencia	license	Documento de licencia que se aplica a este contenido, normalmente indicado por URL.
Accesibilidad	accessibilityAPI	Indica que el recurso es compatible con la API de accesibilidad referenciada.
	accessibilityControl	Identifica los métodos de entrada que son suficientes para controlar completamente el recurso descrito.
	accessibilityFeature	Características de contenido del recurso, como medios accesibles, alternativas y mejoras admitidas para la accesibilidad.
	accessibilityHazard	Una característica del recurso descrito que es fisiológicamente peligrosa para algunos usuarios.
	accessMode	El sistema de percepción sensorial humano o la facultad cognitiva a través del cual una persona puede procesar o percibir información. Los valores esperados incluyen: auditivo, táctil, textual, visual.
	accessModeSufficient	Una lista de modos de acceso únicos o combinados que son suficientes para comprender todo el contenido intelectual de un recurso.
	accessibilitySummary	Un resumen legible por humanos de características o deficiencias de accesibilidad específicas.

Elaborado por: Eduardo Rosero

La ausencia de estas propiedades en la descripción de los recursos publicados en los sitios Web de las universidades analizadas, ya sea por el desconocimiento de su existencia o forma de utilización, o porque a su vez se utiliza otro vocabulario o formato para describirlos, resulta en una desventaja. Por un lado, el no utilizar las propiedades específicas para la descripción de recursos educativos hace que se pierdan características que aportan a la semántica de dichos recursos. En el caso que se utilice otro vocabulario o formato para la descripción, implica que no se siga las recomendaciones de entes reconocidos a nivel mundial como el W3C, quien promueve el uso del vocabulario JSON-LD como formato para la representación de datos estructurados en el marcado estructurado embebido.

Aunque las propiedades generales del vocabulario Schema permiten tener una buena descripción de los recursos publicados, en el caso de los recursos educativos resultaría muy práctica la utilización de propiedades más específicas del contexto que permitan identificar a los recursos educativos como tal y no como cualquier otro recurso de la Web.

Además, el uso de propiedades de licencia y accesibilidad aportarían mucha información sobre el recurso, de tal forma que los usuarios puedan decidir si será útil ingresar a un sitio a revisar el contenido, ya que por lo general se utilizan motores de búsqueda para realizar consultas y estos presentan como resultado un listado de fragmentos enriquecidos con datos sobre recursos que se alinean o coinciden con las búsquedas de los usuarios.

4 CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

El uso de la metodología CRISP-DM, permitió obtener información relevante sobre los datos referentes al uso de marcado estructurado embebido que serían de utilidad al momento de diseñar algunos de los componentes del framework destinados al análisis de datos y generación de resultados. Además, la metodología permitió que el desarrollo del proyecto se realice de una manera sistemática y ordenada.

El uso de la metodología DSR, permitió enfocar los esfuerzos del trabajo en diseñar un framework como solución para la obtención, procesamiento y análisis de datos para conocer la adopción de la tecnología de la web semántica con el vocabulario Schema y el formato JSON-LD por parte de las universidades para la publicación y descripción de recursos educativos en sus sitios Web.

La aplicación conjunta de las metodologías CRISP-DM y DSR de las cuales se relacionó las diversas fases que las componen, permitió desarrollar de forma conjunta la lógica de análisis de datos y el diseño de los componentes del framework. Esta fusión de metodologías trajo consigo la ventaja de obtener un artefacto funcional para un proceso completo de análisis de datos basado en buenas prácticas y apoyado por las bases de ambas metodologías.

Al definir la estructura del framework en tres capas claramente identificadas, se logró tener una visión detallada de los componentes que lo conforman, así como la interacción existente entre ellos. Esto facilitó el desarrollo de tales componentes, puesto que cada uno estaba orientado a cumplir una tarea concreta, apoyado por la funcionalidad de uno o varios de los demás componentes.

La estrategia diseñada para la descarga de datos, la cual contempla un nodo principal y varios nodos secundarios que se comunican mediante un sistema gestor de colas, permitió agilizar el proceso por medio de la paralelización al tener varios componentes encargados de acceder a los sitios Web de las universidades para recuperar las páginas Web una única vez y procesarlas para la extracción de datos, evitando así la redundancia en el acceso a las páginas Web lo que a su vez optimiza los tiempos de descarga.

Con base en los resultados obtenidos a partir del análisis de datos, se puede concluir que las universidades no hacen uso del marcado estructurado embebido con el vocabulario Schema y el formato JSON-LD en la medida que se esperaría al ser este el estándar aceptado por el W3C. De hecho, a pesar de que en el 88% de las universidades analizadas se hace uso de marcado estructurado embebido con Schema y JSON-LD, los resultados muestran que del total de páginas Web descargadas solo el 19.1% contiene el dicho marcado estructurado, es decir que el 80.9%, correspondiente a 824170 páginas Web, no contienen marcado estructurado con Schema y JSON-LD. Además, aunque se encontró el uso de clases de Schema que se pueden asociar con recursos educativos, como son: WebSite, WebPage, ImageObject, Article, Course, VideoObject, CreativeWork, Thesis, no se pudo evidenciar el uso de propiedades específicas como: educationalAlignment o educationalUse, que los definan en un contexto educativo.

Gracias a la estructura modular y en capas del framework, se puede argumentar que resultaría sencillo realizar pequeñas modificaciones para enfocar el análisis de datos en otros contextos diferentes del educativo, así mismo, se podría realizar pequeñas modificaciones para variar el objetivo del análisis para que no sea estrictamente basado en el vocabulario Schema y el formato JSON-LD. Entonces, se puede concluir que el framework es extensible y fácilmente generalizable.

Las pruebas de funcionalidad ejecutadas al framework, gracias su diseño controlado, permitieron validar que cada uno de sus componentes cumple de forma correcta con la tarea para la cual fue concebido. Esto se pudo evidenciar mediante el seguimiento realizado a cada uno de los componentes monitorizando el flujo de entradas y salidas durante la fase de evaluación.

La elaboración de un dashboard sencillo permitió que los resultados respecto al uso de JSON-LD en los recursos educativos publicados por las universidades en estudio sean claros de entender y visualizar. El uso de gráficos como pasteles y barras horizontales para representar los diferentes datos que se obtuvieron a partir del framework simplificaron la tarea de presentación de resultados. Además, con la opción de interacción mediante la aplicación de filtros se logró realizar un análisis más dinámico.

Gracias al componente de almacenamiento de HTML que trabaja en conjunto con los componentes del servicio de descarga, se pudo generar un conjunto de datos relacionado específicamente al ámbito académico, compuesto por el código HTML de las páginas de los sitios Web de las universidades que fueron objeto de análisis de este trabajo. Este dataset[12] resulta útil para replicar los análisis ejecutados o para realizar nuevos análisis sin la necesidad de descargar los datos nuevamente.

4.2 Recomendaciones

Para la descarga de datos es recomendable tener una buena conexión de Internet, puesto que este proceso requiere el acceso a las páginas Web de los sitios de las universidades, y como se lo realiza en paralelo con cada nodo y se lanzan varias peticiones simultaneas dentro de los nodos, el consumo del ancho de banda resulta afectado por la cantidad de solicitudes de páginas Web. Además, resultaría beneficioso que los nodos de descarga se ejecuten en diferentes máquinas para la optimización del consumo de recursos tanto de la máquina como del ancho de banda del Internet.

Debido a que el framework permite realizar la descarga de los documentos HTML de las páginas Web, se recomienda contar con un buen espacio de almacenamiento en disco ya que, dependiendo de la forma en que se diseñan los sitios Web, algunas de las páginas pueden resultar bastante pesadas porque incluyen fragmentos CSS para generar estilos y fragmentos JavaScript para proporcionar funcionalidad adicional, lo que causa que ocupen mucho espacio al ser almacenadas. Por ejemplo, la cantidad de disco usada para las páginas Web de las 100 universidades del estudio fue de algo más de 85 GB.

A pesar de que el framework está enfocado en analizar el uso de marcado estructurado embebido con el vocabulario Schema y el formato JSON-LD en los sitios Web de las universidades, se recomienda realizar pruebas con el framework para analizar otros sitios Web. Incluso con ciertas

[12] https://www.kaggle.com/eduardorosero/web-pages-of-university-websites

modificaciones en los componentes, es posible realizar el análisis del uso de marcado estructurado embebido con otro vocabulario o formato.

Si se quiere replicar o ampliar el análisis realizado en este trabajo, se recomienda utilizar el conjunto de datos obtenido a partir de la ejecución del framework, con la finalidad de evitar realizar el proceso de descarga de datos nuevamente. Este dataset se encuentra publicado en la plataforma kaggle.

Si a partir del framework se obtiene una cantidad muy grande de datos, se recomienda proporcionar buenos recursos, especialmente memoria y procesador para el sistema gestor de base de datos MongoDB, de tal forma que el rendimiento no se vea afectado, tanto en la base de datos como en los componentes que interactúan con esta, sobre todo el servicio de consulta de datos. Además, sería de gran ayuda la creación de índices en las colecciones para facilitar el proceso de búsqueda en la base de datos.

Si se desea conocer de una manera dinámica la creación de marcado estructurado con el vocabulario Schema y el formato JSON-LD, se recomienda el uso de la herramienta generatorjsonld[13] producto del trabajo [45] relacionado con el uso de JSON-LD, la cual permite generar los fragmentos de marcado estructurado para las clases de Schema que se utilizan para la descripción de recursos educativos. La herramienta consiste en un formulario Web en donde se agrega valores para las propiedades de cada clase de Schema mientras se tiene una visualización previa del resultado, el mismo que se puede copiar para embeberlo en el código HTML de una página Web.

[13] https://generatorjsonld.herokuapp.com/home

REFERENCIAS BIBLIOGRÁFICAS

[1] A. T. Ferreira, C. Fernandes, J. Vieira, and F. Portela, "Pervasive intelligent models to predict the outcome of COVID-19 patients," *Futur. Internet*, vol. 13, no. 4, 2021, doi: 10.3390/fi13040102.

[2] T. M. Gourgescu, "Machine Learning Based System for Semantic Indexing Documents Related to Cybersecurity," *Econ. Informatics J.*, vol. 19, no. 1/2019, pp. 5–13, 2019, doi: 10.12948/ei2019.01.01.

[3] S. J. Daniel, "Education and the COVID-19 pandemic," *Prospects*, vol. 49, no. 1–2, pp. 91–96, 2020, doi: 10.1007/s11125-020-09464-3.

[4] R. Radha, K. Mahalakshmi, V. S. Kumar, and A. R. Saravanakumar, "E-Learning during Lockdown of Covid-19 Pandemic: A Global Perspective," *Int. J. Control Autom.*, vol. 13, no. 4, pp. 1088–1099, 2020.

[5] R. Navarrete, L. Recalde, C. Montenegro, and S. Lujan-Mora, "Analyzing embedded semantic with JSON-LD and microdata for educational resources in large scale web datasets," *Proc. - 6th Annu. Conf. Comput. Sci. Comput. Intell. CSCI 2019*, pp. 1133–1138, 2019, doi: 10.1109/CSCI49370.2019.00214.

[6] A. Bakhouyi, R. Dehbi, M. Banane, and M. Talea, "A Semantic Web Solution for Enhancing the Interoperability of E-Learning Systems by Using Next Generation of SCORM Specifications," *Adv. Intell. Syst. Comput.*, vol. 1102 AISC, pp. 56–67, 2020, doi: 10.1007/978-3-030-36653-7_5.

[7] R. Navarrete and S. Luján-mora, "Microdatos con vocabulario de Schema," *2018 13th Iber. Conf. Inf. Syst. Technol.*, pp. 1–6.

[8] T. Ohshima and M. Toyama, "SDC: Structured data collection by yourself," *ACM Int. Conf. Proceeding Ser.*, no. March, pp. 16–18, 2018, doi: 10.1145/3200842.3200849.

[9] V. A. Ngo, T. T. N. Doan, T. T. Le, T. H. Tran, and B. L. Do, "Exploration and Integration of Job Portals in Vietnam," *Proc. - 2020 RIVF Int. Conf. Comput. Commun. Technol. RIVF 2020*, 2020, doi: 10.1109/RIVF48685.2020.9140732.

[10] E. L. Kitaev and R. Y. Skornyakova, "Scraping on the fly of external web resources, driven by HTML page markup," *Keldysh Inst. Prepr.*, no. 20, pp. 1–31, 2019, doi: 10.20948/prepr-2019-20.

[11] J. L. Ambite, J. Gordon, L. Fierro, G. Burns, and J. Mathew, "Linking educational resources on data science," *33rd AAAI Conf. Artif. Intell. AAAI 2019, 31st Innov. Appl. Artif. Intell. Conf. IAAI 2019 9th AAAI Symp. Educ. Adv. Artif. Intell. EAAI 2019*, pp. 9404–9409, 2019, doi: 10.1609/aaai.v33i01.33019404.

[12] M. Tavakoli, A. Faraji, S. T. Mol, and G. Kismihok, "OER Recommendations to Support Career Development," *Proc. - Front. Educ. Conf. FIE*, vol. 2020-Octob, 2020, doi:

10.1109/FIE44824.2020.9274175.

[13] C. Steinberger and J. Frießer, “Why and how to capture the semantics of web user interfaces,” *18th Int. Conf. WWW/Internet 2019*, no. February, pp. 35–42, 2019, doi: 10.33965/icwi2019_2019131005.

[14] “Schema.org.” https://schema.org/ (accessed Jul. 17, 2021).

[15] R. Navarrete and S. Lujan-Mora, “Use of Linked Data to enhance Open Educational Resources,” *2015 Int. Conf. Inf. Technol. Based High. Educ. Training, ITHET 2015*, 2015, doi: 10.1109/ITHET.2015.7218017.

[16] E. Rajabi and W. Greller, “Exposing social data as linked data in education,” *Int. J. Semant. Web Inf. Syst.*, vol. 15, no. 2, pp. 92–106, 2019, doi: 10.4018/IJSWIS.2019040105.

[17] M. Lanthaler and C. Gütl, “On using JSON-LD to create evolvable RESTful services,” *ACM Int. Conf. Proceeding Ser.*, no. April, pp. 25–32, 2012, doi: 10.1145/2307819.2307827.

[18] J. Wang, A. Aryani, B. Evans, and L. Wyborn, “Providing research graph data in JSON-LD using schema.org,” *26th Int. World Wide Web Conf. 2017, WWW 2017 Companion*, pp. 1213–1218, 2017, doi: 10.1145/3041021.3053052.

[19] M. Lanthaler and C. Gütl, “Model your application domain, not your JSON structures,” *WWW 2013 Companion - Proc. 22nd Int. Conf. World Wide Web*, pp. 1415–1420, 2013, doi: 10.1145/2487788.2488184.

[20] M. Lanthaler, “Creating 3rd generation web APIs with hydra,” *WWW 2013 Companion - Proc. 22nd Int. Conf. World Wide Web*, pp. 35–37, 2013, doi: 10.1145/2487788.2487799.

[21] A. T. Pires and G. L. Miranda, “Digital educational resources production in a Digital Tv studio: Training course for teachers,” pp. 1–7, 2016, doi: 10.1109/cisti.2016.7521540.

[22] V. Rodés-Paragarino, A. Gewerc-Barujel, and M. Llamas-Nistal, “Use of Repositories of Digital Educational Resources: State-of-the-Art Review,” *Rev. Iberoam. Tecnol. del Aprendiz.*, vol. 11, no. 2, pp. 73–78, 2016, doi: 10.1109/RITA.2016.2554000.

[23] S. Scheunemann, A. Brandao, and D. Brauner, “Towards defining quality criteria for digital educational resources in distance learning,” *EDUNINE 2018 - 2nd IEEE World Eng. Educ. Conf. Role Prof. Assoc. Contemp. Eng. Careers, Proc.*, pp. 1–4, 2018, doi: 10.1109/EDUNINE.2018.8450968.

[24] J. Xiong, Y. Liu, and W. Liu, “Ontology-based integration and sharing of big data educational resources,” *Proc. - 11th Web Inf. Syst. Appl. Conf. WISA 2014*, pp. 245–248, 2014, doi: 10.1109/WISA.2014.51.

[25] C. Gang, “Sharing mechanism of educational resource system based on grid,” *Proc. - 2008 Int. Conf. Multimed. Inf. Technol. MMIT 2008*, pp. 133–135, 2008, doi: 10.1109/MMIT.2008.82.

[26] H. Phan, “Building Application Powered by Web Scraping,” no. March, 2019, [Online]. Available: https://www.theseus.fi/handle/10024/166489

[27] E. Uzun, “A Novel Web Scraping Approach Using the Additional Information Obtained

from Web Pages," *IEEE Access*, vol. 8, pp. 61726–61740, 2020, doi: 10.1109/ACCESS.2020.2984503.

[28] S. Goel, M. Bansal, A. K. Srivastava, and N. Arora, "Web Crawling-based Search Engine using Python," *Proc. 3rd Int. Conf. Electron. Commun. Aerosp. Technol. ICECA 2019*, pp. 436–438, 2019, doi: 10.1109/ICECA.2019.8821866.

[29] R. Diouf, E. N. Sarr, O. Sall, B. Birregah, M. Bousso, and S. N. Mbaye, "Web Scraping: State-of-the-Art and Areas of Application," *Proc. - 2019 IEEE Int. Conf. Big Data, Big Data 2019*, pp. 6040–6042, 2019, doi: 10.1109/BigData47090.2019.9005594.

[30] K. Turk, S. Pastrana, and B. Collier, "A tight scrape: Methodological approaches to cybercrime research data collection in adversarial environments," *Proc. - 5th IEEE Eur. Symp. Secur. Priv. Work. Euro S PW 2020*, no. June, pp. 428–437, 2020, doi: 10.1109/EuroSPW51379.2020.00064.

[31] S. Mehak, R. Zafar, S. Aslam, and S. M. Bhatti, "Exploiting filtering approach with web scrapping for smart online shopping : PPenny Wise: A wise tool for online shopping," *2019 2nd Int. Conf. Comput. Math. Eng. Technol. iCoMET 2019*, pp. 1–5, 2019, doi: 10.1109/ICOMET.2019.8673399.

[32] E. S. Pramukantoro, D. Primanita Kartikasari, and R. A. Siregar, "Performance evaluation of MongoDB, cassandra, and HBase for heterogenous IoT data storage," *Proc. ICAITI 2019 - 2nd Int. Conf. Appl. Inf. Technol. Innov. Explor. Futur. Technol. Appl. Inf. Technol. Innov.*, pp. 203–206, 2019, doi: 10.1109/ICAITI48442.2019.8982159.

[33] K. Chaudhary, M. Gupta, and P. Kaur, "Analyzing IPL dataset with MongoDB," *Proc. 9th Int. Conf. Cloud Comput. Data Sci. Eng. Conflu. 2019*, pp. 212–216, 2019, doi: 10.1109/CONFLUENCE.2019.8776979.

[34] A. S. Chaudhary, K. Singh, S. Kalra, and P. Kaur, "An empirical comparison of mongoDB and hive," *2018 4th Int. Conf. Comput. Commun. Autom. ICCCA 2018*, pp. 1–4, 2018, doi: 10.1109/CCAA.2018.8777525.

[35] M. Sharma, V. D. Sharma, and M. M. Bundele, "Performance Analysis of RDBMS and No SQL Databases: PostgreSQL, MongoDB and Neo4j," *3rd Int. Conf. Work. Recent Adv. Innov. Eng. ICRAIE 2018*, vol. 2018, no. November, pp. 1–5, 2018, doi: 10.1109/ICRAIE.2018.8710439.

[36] C. F. Andor, V. Varga, and C. Sacarea, "A graph based knowledge and reasoning representation approach for modeling mongodb data structure and query," *2019 27th Int. Conf. Software, Telecommun. Comput. Networks, SoftCOM 2019*, no. Section II, 2019, doi: 10.23919/SOFTCOM.2019.8903854.

[37] L. Rocha, F. Vale, E. Cirilo, D. Barbosa, and F. Mourão, "A framework for migrating relational datasets to NoSQL," *Procedia Comput. Sci.*, vol. 51, no. 1, pp. 2593–2602, 2015, doi: 10.1016/j.procs.2015.05.367.

[38] F. Lu, T. Fang, Z. Zhang, S. Li, J. Chen, H. An, and W. Han, "Improving the performance of mongodb with RDMA," *Proc. - 21st IEEE Int. Conf. High Perform. Comput. Commun. 17th IEEE Int. Conf. Smart City 5th IEEE Int. Conf. Data Sci. Syst. HPCC/SmartCity/DSS*

2019, pp. 1004–1010, 2019, doi: 10.1109/HPCC/SmartCity/DSS.2019.00144.

[39] R. Aluvalu and M. A. Jabbar, "Handling data analytics on unstructured data using MongoDB," *IET Conf. Publ.*, vol. 2018, no. CP747, 2018, doi: 10.14419/ijet.v7i2.12.11320.

[40] P. W. Jordaan and J. E. W. Holm, "Reflection on MongoDB Database Logical and Physical Modeling," *IEEE AFRICON Conf.*, vol. 2019-Septe, 2019, doi: 10.1109/AFRICON46755.2019.9133972.

[41] Ł. Ostrowski, M. Helfert, and S. Xie, "A conceptual framework to construct an artefact for meta-abstract design knowledge in design science research," *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, pp. 4074–4081, 2012, doi: 10.1109/HICSS.2012.51.

[42] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, 2007, doi: 10.2753/MIS0742-1222240302.

[43] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, W. Rudiger, "Crisp-Dm 1.0," *Cris. Consort.*, p. 76, 2000.

[44] V. Vaishnavi, B. Kuechler, and S. Petter, "Design Science in Information Systems," *Assoc. Manag. Softw.*, vol. 28, no. 1, pp. 75–105, 2004, [Online]. Available: http://www.desrist.org/design-research-in-information-systems/

[45] L. Recalde, R. Navarrete, and F. Pogo, "Making Open Educational Resources Discoverable: A JSON-LD Generator for OER Semantic Annotation," *Eight Int. Conf. eDemocracy eGovernment*, 2021.