GRADO EN INGENIERÍA INFORMÁTICA DE GESTIÓN Y SISTEMAS DE INFORMACIÓN

**TRABAJO FIN DE GRADO**

***gENERACión, evaluación y explotación de linked data basados en los datos publicados por open data euskadi***

**Curso:** 2017-2018

**Fecha:** <XXXX, día, mes, año>

**Alumno/Alumna**: Uchuari Vera Mishel

**Director/Directora:** Casquero Oyarzabal Oskar

**RESUMEN**

Diariamente distintas organizaciones tanto públicas como privadas publican Datos Abiertos y aumenta el número de ellas que eligen hacerlo a través de Datos Enlazados. Este nuevo método de publicación de información tiene como objetivo la interconexión, reutilización e integración de la información para crear una gran base de datos interconectada y distribuida que pueda ser consumida tanto por humanos como por máquinas.

Con este proyecto se pretende crear una conjunto de modelos de conversión de un conjunto de datos públicos publicados por Open Data Euskadi a Datos Enlazados, así como el proceso posterior de testeo de calidad de los datos generados y descubrimiento de enlaces relacionados.

**ÍNDICE**

[1. INTRODUCCIÓN 5](#_Toc500783577)

[1.1 Motivación 5](#_Toc500783578)

[1.2 Marco teórico: Introducción al RDF 5](#_Toc500783579)

[1.3 Descripción y situación del trabajo 7](#_Toc500783580)

[1.4 Contribución 7](#_Toc500783581)

[2. PLANTEAMIENTO INICIAL 8](#_Toc500783582)

[2.1 Objetivos 8](#_Toc500783583)

[2.2 Arquitectura 8](#_Toc500783584)

[2.3 Alcance 9](#_Toc500783585)

[2.3.1 Análisis de requisitos 12](#_Toc500783586)

[2.3.2 Diseño del sistema 12](#_Toc500783587)

[2.3.3 Diseño del programa 12](#_Toc500783588)

[2.3.3.1 Análisis de herramientas necesarias 12](#_Toc500783589)

[2.3.3.2.1 Búsqueda de ontologías relacionadas y creación ontologías propias 13](#_Toc500783590)

[2.3.3.2.2 Creación estructuras para la generación de RDF 13](#_Toc500783591)

[2.3.3.2.3 Creación algoritmos SPARQL endpoint y funcionalidades asociadas a la representación gráfica 14](#_Toc500783592)

[2.3.4 Codificación 14](#_Toc500783593)

[2.3.5 Pruebas 14](#_Toc500783594)

[2.3.6 Memoria 14](#_Toc500783595)

[2.4 Planificación temporal 15](#_Toc500783596)

[2.5 Herramientas 15](#_Toc500783597)

[2.5.1 Hardware 15](#_Toc500783598)

[2.5.2 Software 15](#_Toc500783599)

[2.6 Gestión de riesgos 16](#_Toc500783600)

[2.6.1 Error en la planificación temporal 16](#_Toc500783601)

[2.6.2 Desarrollo 17](#_Toc500783602)

[2.6.2.1 Falta de conocimiento 17](#_Toc500783603)

[2.6.2.2 Pérdida software desarrollado 17](#_Toc500783604)

[2.6.2.3 Enfermedad del personal de desarrollo 17](#_Toc500783605)

[2.6.2.4 Pérdida del equipo 17](#_Toc500783606)

[2.7 Evaluación económica 18](#_Toc500783607)

[3. ANÁLISIS DE ANTECEDENTES 19](#_Toc500783608)

[4. VERIFICACIÓN Y EVALUACIÓN 21](#_Toc500783609)

[4.1 Generación de RDF 22](#_Toc500783610)

[4.2 Testeo calidad RDF generado 22](#_Toc500783611)

[4.2.1 Testeo calidad RDF correspondiente a la calidad del aire 22](#_Toc500783612)

[4.2.1.1 Respecto a las observaciones 22](#_Toc500783613)

[4.2.1.2 Respecto a las mediciones con resultados de valores numéricos 23](#_Toc500783614)

[4.2.1.3 Respecto a las mediciones con resultados con valores literales 24](#_Toc500783615)

[4.2.2 Testeo calidad RDF correspondiente a las estaciones meteorológicas 24](#_Toc500783616)

[4.2.2.1 Respecto a las observaciones 24](#_Toc500783617)

[4.2.2.2 Respecto a las mediciones 25](#_Toc500783618)

[4.2.2.3 Testeo calidad RDF correspondiente a las retribuciones nominativas 25](#_Toc500783619)

[4.2.3 Testeo calidad RDF correspondiente a los contratos de trabajo 27](#_Toc500783620)

[4.3 SPARQL Endpoint 30](#_Toc500783621)

[4.4 Tabla 31](#_Toc500783622)

[4.5 Grafo 32](#_Toc500783623)

[5. CONCLUSIONES Y TRABAJO FUTURO 34](#_Toc500783624)

[6. Bibliografía 36](#_Toc500783625)

**ÍNDICE ILUSTRACIONES**

[Ilustración 1: Esquema tripleta RDF 5](#_Toc500782682)

[Ilustración 2: RDF en formato RDF/XML 5](#_Toc500782683)

[Ilustración 3: RDF en formato TURTLE haciendo uso de namespaces 6](#_Toc500782684)

[Ilustración 4: RDF en formato TURTLE sin namespace 6](#_Toc500782685)

[Ilustración 5: Query SPARQL 6](#_Toc500782686)

[Ilustración 6: Arquitectura pipeline del TFG 8](#_Toc500782687)

[Ilustración 7: Arquitectura cliente-servidor del TFG 8](#_Toc500782688)

[Ilustración 8: Paradigma desarrollo software cascada 9](#_Toc500782689)

[Ilustración 9: Diagrama EDT del TFG 10](#_Toc500782690)

**ÍNDICE TABLAS**

[Tabla 1: Análisis de requisitos 12](#_Toc500783144)

[Tabla 2: Diseño del sistema 12](#_Toc500783145)

[Tabla 3: Análisis de herramientas necesarias 13](#_Toc500783146)

[Tabla 4: Búsqueda de ontologías relacionadas y creación ontologías propias 13](#_Toc500783147)

[Tabla 5: Creación estructuras para la generación RDF 13](#_Toc500783148)

[Tabla 6: Creación algoritmos SPARQL endpoint y funcionalidades asociadas a la representación gráfica 14](#_Toc500783149)

[Tabla 7: Codificación 14](#_Toc500783150)

[Tabla 8: Pruebas 14](#_Toc500783151)

[Tabla 9: Memoria 15](#_Toc500783152)

[Tabla 10: Herramientas hardware 15](#_Toc500783153)

[Tabla 11: Herramientas software 16](#_Toc500783154)

[Tabla 12: Error en la planificación temporal 16](#_Toc500783155)

[Tabla 13: Error en el desarrollo por falta de conocimiento 17](#_Toc500783156)

[Tabla 14: Error en el desarrollo por pérdida software desarrollado 17](#_Toc500783157)

[Tabla 15: Error en el desarrollo por enfermedad del personal del desarrollo 17](#_Toc500783158)

[Tabla 16: Error en el desarrollo por pérdida del equipo utilizado 18](#_Toc500783159)

[Tabla 17: Pruebas en la generación de RDF 22](#_Toc500783160)

[Tabla 18: Pruebas en el testeo de la calidad del aire respecto a las observaciones 23](#_Toc500783161)

[Tabla 19: Pruebas en el testeo del RDF correspondiente a la calidad del aire respecto a las mediciones con resultados numéricos 23](#_Toc500783162)

[Tabla 20: Pruebas en el testeo del RDF correspondiente a la calidad del aire respecto a las mediciones con resultados con valores literales 24](#_Toc500783163)

[Tabla 21: Pruebas en el testeo del RDF correspondiente a las estaciones meteorológicas respecto a las observaciones 24](#_Toc500783164)

[Tabla 22: Pruebas en el testeo del RDF correspondiente a las estaciones meteorológicas respecto a las mediciones 25](#_Toc500783165)

[Tabla 23: Pruebas en el testeo del RDF correspondiente a las retribuciones nominativas 27](#_Toc500783166)

[Tabla 24: ruebas en el testeo del RDF referente a los contratos laborales 29](#_Toc500783167)

[Tabla 25: Pruebas sobre el SPARQL endpoint 30](#_Toc500783168)

[Tabla 26: Pruebas sobre la tabla que se genera al realizar una consulta sobre el SPARQL endpoint 31](#_Toc500783169)

[Tabla 27: Pruebas sobre la tabla que se genera al realizar una consulta sobre el SPARQL endpoint 33](#_Toc500783170)

**Glosario**

TFG: Trabajo fin de grado

W3C: El consorcio WWW o World Wide Web Consortium es un consorcio internacional que genera recomendaciones y estándares para el crecimiento de la World Wide Web.

API: La interfaz de programación de aplicaciones o Application Programming es la agrupación de procesos, métodos o procedimientos que ofrece una librería para su utilización por otro software como capa de abstracción.

XML: Extensible Markup Language o lenguaje de marcas extensible es un metalenguaje desarrollado por el W3C que se utiliza para almacenar datos de forma legible.

CSV: Valores separados por comas o comma separated values, este término hace referencia a un documento de formato libre utilizado para representar datos en forma tabular, siendo las columnas separadas por coma o punto y coma, y las filas por saltos de línea.

RDF: El Marco de Descripción de Recursos O Resource Description Framework es una familia de especificaciones de la W3C para el intercambio de datos en la WEB.

SPARQL: SPARQL Protocol and RDF Query Language o protocolo sparql y lenguaje de consulta sobre RDF, es el lenguaje de consulta RDF.

RDF/XML: Sintaxis normativa definida por el W3C para expresar un grafo RDF como XML.

JSON-LD: JSON-LD o JavaScript Object Notation for Linked Data, es una sintaxis para expresar RDF usando JSON.

Turtle: Turtle o Terse RDF Triple Language, es una sintaxis para representar RDF similar a SPARQL.

URI: Un identificador de recursos uniforme o uniform resource identifier es una cadena de caracteres que identifica los recursos de una red de forma unívoca.

Triple Store: Base de datos orientada a grafos.

SPARQL Endpoint: Interfaz web con una funcionalidad, ejecutar consultas SPARQL sobre una triple store.

Framework: Un framework o entorno de trabajo es un conjunto estandarizado de practicas o procedimientos para la realización de un determinado trabajo que sirve como precedente para resolver problemas de la misma naturaleza.

EDT: EDT o estructura de descomposición de trabajo es una representación sencilla y organizada de las tareas a realizar antes de concluir un proyecto.

Front-end: En desarrollo web hace referencia a la capa de presentación, a la parte del software con la que el usuario interactua.

Linked data fronted /servidor linked data¿?:

GREL: General Refine Expresion Language o lenguaje de expression general de Refine. Es el lenguaje utilizado por Open Refine para realizar transformaciones sobre los valores de las columnas de los CSV que manipula.

SHACL: Shapes Constraint Language, es un lenguaje para validar grafos RDF sobre un conjunto de condiciones. Estas condiciones son expresadas a su vez en forma de grafo RDF.

1. INTRODUCCIÓN

En este apartado se describirá la motivación inicial del proyecto, las contribuciones que se piensan aportar y una breve descripción de los datos utilizados en la realización de este TFG.

# Motivación

El concepto de Datos Abiertos, es una filosofía que tiene como objetivo poner a disposición de la sociedad, de forma accesible y reutilizable, determinados tipos de datos para favorecer su integración en la cadena de valor, garantizar la transparencia y facilitar la interoperabilidad.

Las administraciones públicas, quienes tienen control sobre grandes cantidades de datos de distinta naturaleza (geográfica, climática, científica, médica, artística, etc.) están continuamente publicando Datos Abiertos y, aunque todavía no es la forma más extendida, son cada vez más las que lo hacen utilizando la forma de Datos Enlazados.

Los Datos Enlazados se refieren a una forma de publicar datos que permite vincular datos distribuidos entre sí con el objetivo de construir la Nube de Datos Enlazados que pueda explorarse de forma automática por máquinas, de forma análoga a como la Web puede explorarse mediante enlaces por personas.

Esta vinculación entre datos distribuidos permite dar mayor significancia a los datos publicados dado que en este formato se puede extraer mucha más información y de mayor utilidad que de CSVs, XMLs y otras formas de almacenaje. Esto permite que los datos publicados de esta forma sean más accesibles, reusables e integrables, características que se alinean con la filosofía de Datos Abiertos y, por tanto, permiten pensar en los Datos Enlazados como una solución tecnológica adecuada para dar soporte a los Datos abiertos.

# Los datos

Los datos utilizados para la realización del proyecto son datos públicos publicados por Open Data Euskadi. Open Data Euskadi es una iniciativa del Gobierno Vasco en cuanto a la publicación abierta y libre de los datos que estén en su poder y que no tengan restricciones de privacidad, seguridad o propiedad.

Se utilizarán 4 datasets: calidad del aire, estaciones metereológicas: lecturas recogidas, relación de puestos de trabajo y evolución de las tablas retributivas de los miembros del gobierno, altos cargos y personal eventual. Dada la importancia del contenido de cadaset por cada uno de ellos se elaboró un modelo de dominio.

* + 1. Calidad del aire

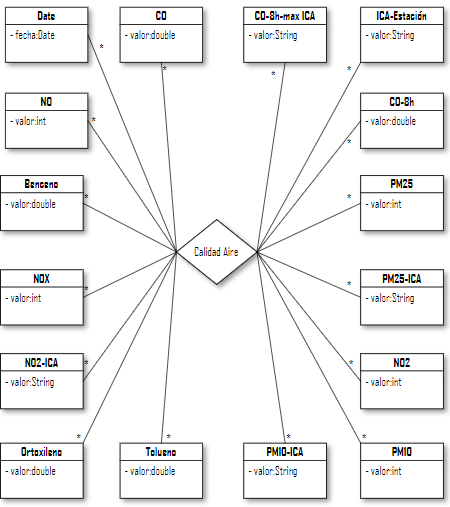
El departamento de Medio Ambiente y Politica Territorial del Gobierno Vasco es el encargado de la monitorización de los niveles de contaminación en la Comunidad Autónoma Vasca a través de la Red de Control de Calidad del Aire.

Esta red dispone de diversos analizadores y sensores que miden los contaminantes que marca la ley que deben cumplir las comunidades autónomas para el control de contaminación en su territorio.

“*La ley 34/2007, de 15 de noviembre, de calidad del aire y protección de la atmosfera define la evaluación como el resultado de aplicar cualquier método que permita medir, calcular, predecir o estimar las emisiones, los niveles o los efectos de la contaminación atmosférica.”*

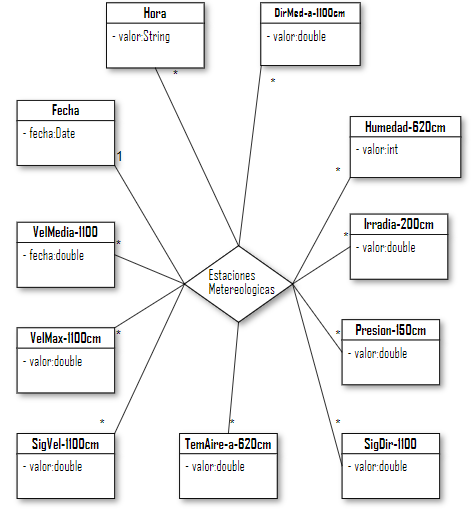
La Red de Control de calidad del Aire dispone de varias estaciones distribuidas a lo largo del país Vasco, actualmente hay 53 estaciones, de las cuales 14 pertenecen a actividades industriales.

En este caso utilizaremos los datos pertenecientes a la estación de Av. Gasteiz recogidos durante el mes de febrero y marzo del año 2017. En la siguiente figura se pueden apreciar las distintas mediciones realizadas en esa estación.

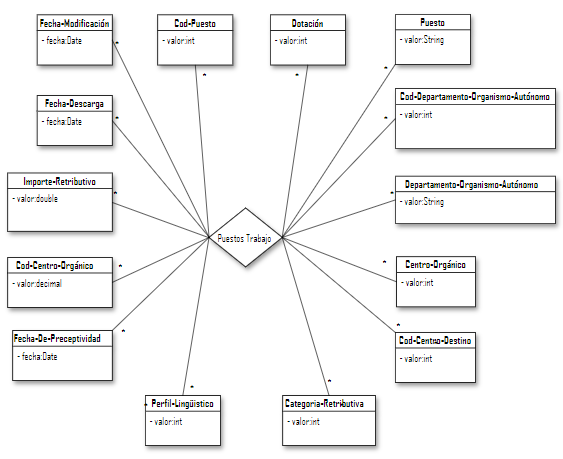


Como se puede observar en el modelo de dominio, el CSV podrá tener distintos valores en cada medición. Las mediciones se realizan sobre elementos como óxidos de nitrógeno, monóxido de carbono, benceno, partículas en suspensión, etc.

* + 1. Estaciones metereólogicas: lecturas recogidas

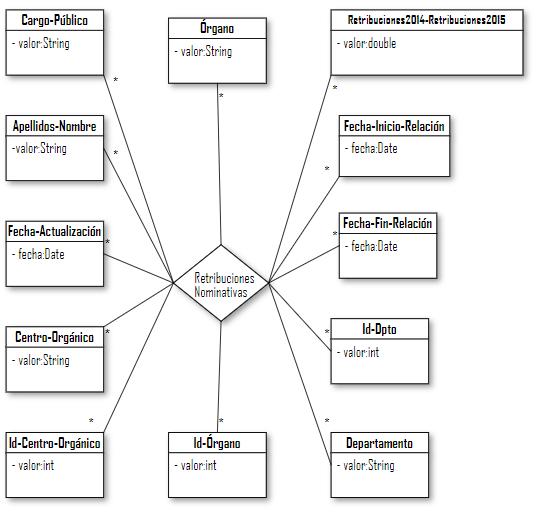


* + 1. Relación de puestos de trabajo



* + 1. Evolución de las tablas retributivas de los miembros del gobierno, altos cargos y personal eventual

Este dataset representa la evolución de las retribuciones del personal del Gobierno Vasco desde el año 2009 hasta la actualidad.



# Marco teórico: Introducción al RDF

Para la publicación e interconexión de los Datos Enlazados, se usa RDF y URI. Al realizar consultas sobre esa información se utiliza SPARQL.

RDF es un modelo estándar para el intercambio de información. Utiliza la estructura de enlaces de la Web, usa URIs para designar relaciones entre objetos o recursos y a estos mismos. A esta relación se la denomina triple.

Estos enlaces forman un grafo dirigido con etiquetas donde los nodos de cada enlace corresponden a recursos que mantienen una relación. Es decir, una tripleta se puede entender como atributos o características de los recursos.

El modelo RDF se puede relacionar con el diseño orientado a objeto donde un objeto tiene una serie de atributos. De este modo en un RDF se pueden distinguir tres tipos de objetos:

* **Recursos:** Cualquier objeto existente identificado por una URI.
* **Propiedades:** Son características o atributos que se usan para describir recursos. Se recogen en ontologías o vocabularios.
* **Objeto**: Corresponde al valor asociado al recurso para una determinada propiedad.

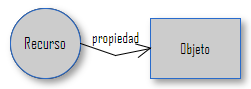


Ilustración 1: Esquema tripleta RDF

La sintaxis básica de RDF es XML, que da como resultado RDF/XML, pero también se puede representar en otros formatos como Turtle, JSON-LD, N-Quads, etc.

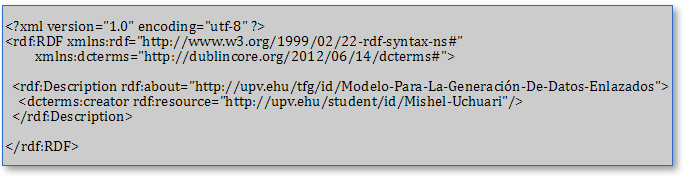


Ilustración 2: RDF en formato RDF/XML

Al representarse RDF se pueden usar namespaces para agrupar URIs.

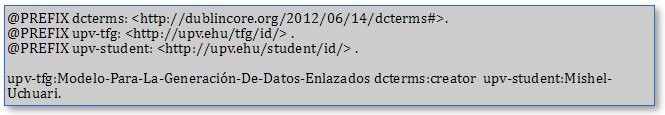


Ilustración 3: RDF en formato TURTLE haciendo uso de namespaces



Ilustración 4: RDF en formato TURTLE sin namespace

SPARQL por otro lado, es un lenguaje estandarizado de consulta sobre RDF.

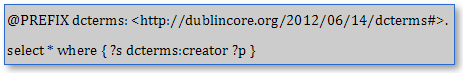


Ilustración 5: Query SPARQL

# Descripción y situación del trabajo

Este TFG consistirá básicamente en la creación de Datos Enlazados siguiendo las especificaciones de este paradigma de publicación de información. Se crearán un conjunto de RDFs partiendo de información publicada en Open Data Euskadi.

Partiendo de esta información contenida en CSVs, se creará una estructura para cada uno de ellos que permitirá generar RDF. Posteriormente se validará la calidad de los RDFs generados siguiendo unas directrices adecuadas a cada uno. Posteriormente, se examinará la calidad del RDF y se intentará conectar los datos generados con información relacionada existente en la Web mediante el descubrimiento de enlaces.

Para darle al usuario la posibilidad de consumir la información generada se creará una interfaz web que actuará de SPARQL endpoint en el que podrá realizar consultas sobre ellos. Esta interfaz también permitirá visualizar los datos a través de un gráfico dinámico en forma de grafo si se usa un tipo de consulta concreto. Además se proporcionará un servidor Linked Data en el que la información existente podrá ser consumida tanto por humanos como por máquinas mediante negociación de contenido.

# Contribución

Se presenta este Trabajo Fin de Grado, como parte de un proyecto más grande denominado ALDAPA (Assistant for Linked Data Production Automation) que tiene como objetivo el desarrollo de un API y un framework en Java para convertir distintos tipos de datos a Datos Enlazados mediante el uso de plugins de conversión a RDF y capacidad para trabajar con Triple Stores.

Por ello se estará satisfaciendo la necesidad real de un cliente, por lo que se usarán recursos, técnicas y procedimientos que van más allá de lo estudiado en el Grado, sino de las soluciones disponibles en el mercado para desarrollar este tipo de sistemas, lo que implica que se realizarán novedosos desarrollos a medida en el área de Linked Data.

1. PLANTEAMIENTO INICIAL

En este punto se presentarán los objetivos, el alcance, la planificación, la descripción de las distintas tareas que se llevarán a cabo, las herramientas que se utilizarán, la gestión de riesgos, los indicadores de calidad y la evaluación económica.

# Objetivos

El objetivo principal es la creación de un modelo para la generación de RDF basado en los datos publicados en Open Data Euskadi. Junto a ese objetivo, sus inmediatos complementarios, la revisión de la calidad del RDF generado así como el descubrimiento de enlaces de conexión con los creados anteriormente.

Como objetivos secundarios, la creación de un SPARQL endpoint de consulta y gestión de los datos en el que se representarán la información de distintas formas gráficas. Por último un servidor Linked Data con negociación de contenido en el que la información podrá ser consumida tanto por agente como por humanos.

# Arquitectura

Para la realización de este proyecto al tener éste diferentes módulos se han usado dos tipos de arquitecturas. La arquitectura de tipo pipeline y de tipo cliente-servidor.

Para toda la parte correspondiente a la creación del RDF, como su testeo de calidad y descubrimiento de enlaces posterior, se ha usado la arquitectura tipo pipeline. En este tipo de arquitectura, a partir de un conjunto de datos iniciales se los va transformando en un proceso comprendido por varias etapas en las que la entrada de cada fase es la salida del anterior. Se puede ver más claro la estructura que seguirán las fases del pipeline en la figura X.



Ilustración 6: Arquitectura pipeline del TFG

La arquitectura cliente-servidor se usa en las últimas fases del proyecto. En este caso se ofrece al usuario/agente un SPARQL endpoint contra el que realizar todo tipo de consultas SPARQL y un servidor Linked Data sobre el que se podrá realizar peticiones con negociación de contenido. La información usada por estos se encontrará alojada en una triple store.

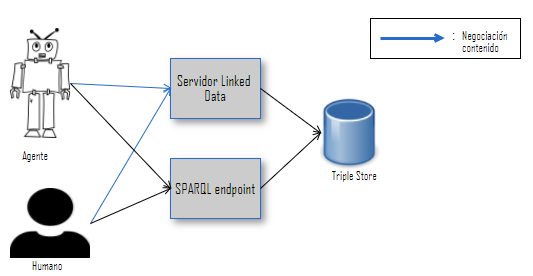


Ilustración 7: Arquitectura cliente-servidor del TFG

# Alcance

En este proyecto se ha decidido seguir el modelo de desarrollo en cascada. En este paradigma se sigue una serie de etapas sucesivas, la etapa siguiente comienza cuando termina la anterior.

Se ha elegido esta forma de desarrollo por la naturaleza de este proyecto. Al ser este un proyecto bien estructurado y con las fases bien definidas, la captura de requisitos que se expondrá posteriormente difícilmente sufrirá algún cambio.

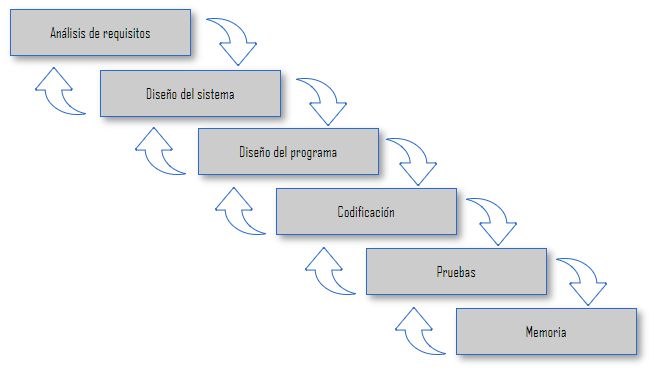


Ilustración 8: Paradigma desarrollo software cascada

El alcance de este proyecto se definirá mediante el siguiente diagrama EDT, donde se expondrán las fases que seguirá el desarrollo del mismo.

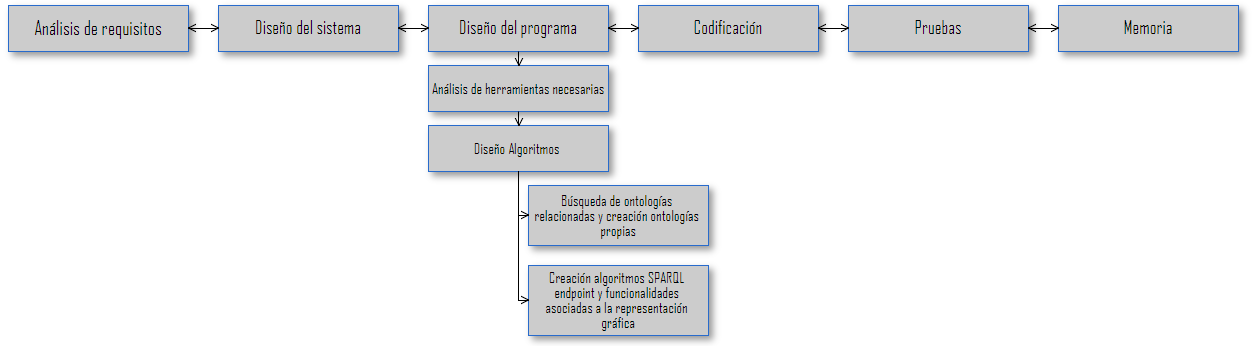


Ilustración 9: Diagrama EDT del TFG

## 2.3.1 Análisis de requisitos

|  |  |
| --- | --- |
| **Duración estimada** | 2 horas |
| **Descripción** | Durante esta fase se definirán los objetivos que debe cubrir el producto final. |
| **Entradas** | - |
| **Recursos necesarios** | Editor de textos, entrevista a profesional. |
| **Salidas/Entregables** | Documento con lista de objetivos a conseguir. |

Tabla 1: Análisis de requisitos

## 2.3.2 Diseño del sistema

|  |  |
| --- | --- |
| **Duración estimada** | 3 horas |
| **Descripción** | Se descompone el proyecto a desarrollar en subconjuntos. En este apartado se define la estructura del sistema a implementar y la especificación de cada uno de los subconjuntos por separado. |
| **Entradas** | Documento obtenido en la fase de Análisis de requisitos |
| **Recursos necesarios** | Editor de textos |
| **Salidas/Entregables** | Documento especificando los distintos subconjuntos en el que estará divido el trabajo a realizar. |

Tabla 2: Diseño del sistema

## 2.3.3 Diseño del programa

En esta fase se realizará el análisis de las herramientas necesarias para llevar a cabo la fase de codificación y se realizarán los algoritmos que seguirá nuestro proyecto.

### 2.3.3.1 Análisis de herramientas necesarias

|  |  |
| --- | --- |
| **Duración estimada** | 5 horas |
| **Descripción** | Se realiza una investigación exhaustiva sobre las recursos disponibles para llevar a cabo nuestro trabajo. |
| **Entradas** | - |
| **Recursos necesarios** | Internet, artículos, entrevista a profesional, editor textos. |
| **Salidas/Entregables** | Lista de herramientas a utilizar |

Tabla 3: Análisis de herramientas necesarias

* + - 1. **Diseño Algoritmos**

En este proyecto como norma general no se definirán algoritmos de programación como tal porque en la naturaleza de este proyecto no aplica, pero lo que si se deberá definir desde el principio es la estructura que seguirán los modelos de conversión para cada CSV. Por eso en esta fase separamos la parte correspondiente a la investigación de las ontologías a usar para cada RDF, o generación de una propia y la parte donde se genera el modelo que deberá seguirse en la parte de codificación para la creación de cada RDF.

En cambio para la parte correspondiente a la generación del SPARQL endpoint y la visualización en forma gráfica si se definirán algoritmos.

#### Búsqueda de ontologías relacionadas y creación ontologías propias

|  |  |
| --- | --- |
| **Duración estimada** | 12 horas |
| **Descripción** | Búsqueda de ontologías relacionadas con las temáticas de nuestros datos de partida y creación de ontologías propias siguiendo el esquema establecido por la W3C. |
| **Entradas** | - |
| **Recursos necesarios** | Internet, artículos, entrevista a profesional, editor textos. |
| **Salidas/Entregables** | Lista de ontologías a usar para cada temática. |

Tabla 4: Búsqueda de ontologías relacionadas y creación ontologías propias

#### Creación estructuras para la generación de RDF

|  |  |
| --- | --- |
| **Duración estimada** | 10 horas |
| **Descripción** | Creación de estructura que se seguirá para la generación de RDF para cada CSV adecuado a sus respectivos datos y naturaleza. |
| **Entradas** | Lista de ontologías creada anteriormente. |
| **Recursos necesarios** | Editor textos, entrevista a profesional. |
| **Salidas/Entregables** | Estructura definida para CSV. |

Tabla 5: Creación estructuras para la generación RDF

#### Creación algoritmos SPARQL endpoint y funcionalidades asociadas a la representación gráfica

|  |  |
| --- | --- |
| **Duración estimada** | 8 horas |
| **Descripción** | Se diseñaran los algoritmos que seguirán la implementación del SPARQL endpoint, así como los que seguirá el grafico que se generará a partir de los resultados obtenidos. |
| **Entradas** | - |
| **Recursos necesarios** | Internet, editor textos. |
| **Salidas/Entregables** | Algoritmos |

Tabla 6: Creación algoritmos SPARQL endpoint y funcionalidades asociadas a la representación gráfica

## Codificación

|  |  |
| --- | --- |
| **Duración estimada** | 300 h |
| **Descripción** | Durante esta etapa se lleva a cabo la codificación de todos los algoritmos o sus equivalentes definidos en la etapa anterior. |
| **Entradas** | Algoritmos, estructuras definidas anteriormente. |
| **Recursos necesarios** | Eclipse, Counter Clock Wise como extensión para eclipse para desarrollar en Clojure. |
| **Salidas/Entregables** | - |

Tabla 7: Codificación

## Pruebas

|  |  |
| --- | --- |
| **Duración estimada** | 20 h |
| **Descripción** | Una vez finalizada la implementación de los requerimientos se diseña un conjunto de pruebas y se llevan a cabo. |
| **Entradas** | Código en funcionamiento |
| **Recursos necesarios** | Eclipse, Editor textos |
| **Salidas/Entregables** | - |

Tabla 8: Pruebas

## Memoria

|  |  |
| --- | --- |
| **Duración estimada** | 40 h |
| **Descripción** | Por último se elaborará un documento que reflejará los objetivos finales del TFG, así como el resultado obtenido y la forma o metodologías usadas para llegar a ese punto. |
| **Entradas** | - |
| **Recursos necesarios** | Editor textos, artículos |
| **Salidas/Entregables** | Memoria final proyecto |

Tabla 9: Memoria

# Planificación temporal

En la siguiente figura se muestra en detalle la planificación temporal del proyecto siendo esta representada por un diagrama de Gantt. Como podemos observar en el grafico la estimación total para la realización del proyecto es de 400 horas.

# Herramientas

## Hardware

|  |  |
| --- | --- |
| **Portátil** | Hacer E5 571, 64 bits |

Tabla 10: Herramientas hardware

## Software

|  |  |
| --- | --- |
| **Eclipse** | Entorno de programación para desarrollo en Java |
| **CounterClockWise** | Extensión de Eclipse para el desarrollo en Clojure |
| **Grafter** | Librería de Clojure para la producción de Linked Data |
| **Maven** | Herramienta para la gestión y construcción de proyectos Java |
| **Github** | Herramienta para alojar proyectos usando el sistema de control de versiones de Git |
| **Microsoft Office** | Herramienta de ofimática |
| **Cacoo** | Herramienta online para la creación de gráficos tales como casos de uso, modelo de dominio, EDT, etc. |
| **Gantt Project** | Interfaz para Linked Data que permite interactuar con SPARQL endpoints |
| **Insomnia** | Cliente REST para la realización de peticiones HTTP |
| **Pubby** | Linked Data Frontend para SPARQL endpoints |
| **D3** | Librería JavaScript para la producción de infogramas dinámicos |
| **Silk** | Framework para la integración de fuentes de datos y descubrimiento de enlaces |
| **TopBraid** | Api Java para llevar a comprobaciones de restricciones definidas en Shacl |
| **Yasqe** | Librería JavaScript para la creación de editores de querys SPARQL |
| **GraphDB** | Base de datos orientada a grafos |
| **RDF4J** | Framework de Java para el procesamiento de RDF |

Tabla 11: Herramientas software

# Gestión de riesgos

La realización de un proyecto de cualquier naturaleza acarrea unos riesgos por lo cual conviene realizar un análisis previo de estos para intentar evitarlos, así como un plan de contingencia que recoja el protocolo a seguir en caso de que se produjese alguno. A continuación se presenta un listado de riesgos que se prevé podrían ocurrir durante la realización de este trabajo.

## Error en la planificación temporal

|  |  |
| --- | --- |
| **Descripción** | La duración del proyecto es bastante larga y es conveniente tener en cuenta la posibilidad de que el tiempo estimado sea erróneo y se necesiten más o menos horas para su realización |
| **Prevención** | Se intentará crear una estimación temporal lo más detallada posible teniendo en cuenta la experiencia adquirida en la realización de proyectos similares |
| **Plan de contingencia** | Dejar margen de error en la estimación temporal |
| **Probabilidad** | Muy alta |
| **Impacto** | Medio |

Tabla 12: Error en la planificación temporal

## Desarrollo

### Falta de conocimiento

|  |  |
| --- | --- |
| **Descripción** | Al usar nuevas tecnologías es posible encontrarse con problemas de implementación por falta de conocimiento y experiencia con ellas |
| **Prevención** | Formación antes y durante la realización del proyecto |
| **Plan de contingencia** | Uso de foros de consulta y documentaciones oficiales |
| **Probabilidad** | Muy alta |
| **Impacto** | Alto |

Tabla 13: Error en el desarrollo por falta de conocimiento

### Pérdida software desarrollado

|  |  |
| --- | --- |
| **Descripción** | Perdida del software ya sea por virus o pérdidas. |
| **Prevención** | Realización de copias de seguridad regulares |
| **Plan de contingencia** | Implantación de copia de seguridad previa |
| **Probabilidad** | Baja |
| **Impacto** | Muy alta |

Tabla 14: Error en el desarrollo por pérdida software desarrollado

### Enfermedad del personal de desarrollo

|  |  |
| --- | --- |
| **Descripción** | Incapacidad de trabajar en el proyecto por un lapso de tiempo |
| **Prevención** | No se puede prever este tipo de riesgos |
| **Plan de contingencia** | Dejar margen de error en la estimación temporal |
| **Probabilidad** | Media |
| **Impacto** | Medio |

Tabla 15: Error en el desarrollo por enfermedad del personal del desarrollo

### Pérdida del equipo

|  |  |
| --- | --- |
| **Descripción** | Pérdida del equipo utilizado ya sea por deterioro, pérdida o robo |
| **Prevención** | No se puede prever este tipo de riesgos |
| **Plan de contingencia** | Adquirir nuevo equipo en el que se realizará un volcado de las copias de seguridad |
| **Probabilidad** | Baja |
| **Impacto** | Medio |

Tabla 16: Error en el desarrollo por pérdida del equipo utilizado

# Evaluación económica

Cabe recalcar que al ser este un proyecto de índole académico no se tienen pretensiones económicas, pero pese a ello se calcularán los gastos que se hubiesen dado en su ejecución. Para realizar este cálculo nos focalizaremos en los gastos de manos de obra y en los derivados por el uso de hardware ya que en este caso todo el software utilizado era libre.

Como mano de obra tenemos al estudiante que en determinadas ocasiones asumirá el rol de analista y en otras de programador. Del análisis inicial se prevé que en análisis se invertirán 30 horas y en codificación y trabajos derivados de esa tarea se invertirán 330 horas. En este caso consideraremos el sueldo de un analista 30€/hora y el de un programador 21€/hora.

El hardware utilizado ha sido un ordenador portátil Acer E5-57 que posee una vida útil estimada en 8 años y un coste original de 750 €. Este será utilizado un tiempo estimado a priori de 400 horas durante 5 meses.

Por lo tanto el coste total ascendería a 7896.67 €.

1. ANÁLISIS DE ANTECEDENTES

La generación de Datos Enlazados y por consiguiente RDF es una tarea que necesita supervisión humana y no se puede automatizar completamente. Tanto las fases de análisis previo en el que se crea la estructura que seguirá el RDF a crear como la posterior de descubrimiento de enlaces necesitan a una persona que las supervise. Por ello es difícil encontrar alguna aplicación o software desarrollado anteriormente que aune todas las características que este TFG ofrece.

Por lo mencionado con anterioridad, en este trabajo analizamos herramientas diferentes más o menos para cada fase. Se tuvo que buscar herramientas similares para la generación de datos enlazados como es el caso de Open Refine, otras que nos permitieran la creación de un servidor Linked Data como Elda y herramientas que actuarán como SPARQL endpoints.

Open Refine es una herramienta que permite al usuario importar sus datasets y realizar alteraciones sobre ellos tanto de forma manual como usando funciones Grel. A partir de estas funcionalidades se puede generar RDF como descubrir enlaces existentes en la Web que guarden relación con los datos importados. Esta herramienta tiene muchas funcionalidades útiles pero se considera limitada a ocasiones en las que la cantidad de RDF a generar es pequeño, ya que al ser un herramienta de escritorio con la que hay que interactuar continuamente durante el proceso no se puede programar para transformar un gran conjunto de datos que siguen la misma estructura, es decir, si por ejemplo se plantease la idea de transformar a RDF mil datasets que siguen la misma estructura referentes a la calidad del aire, habría que ir cargándolos uno a uno para su posterior conversión a RDF.

Elda es una implementación en Java de Linked Data API, que provee un medio configurable para acceder a datos RDF usando RESTful URLs que son traducidas a querys RDF que se ejecutan sobre un SPARQL endpoint. Provee al usuario de una interfaz gráfica en la que la información en RDF se puede presentar en distintos formatos.

Buscando referencias sobre las que partir para la creación de nuestro SPARQL endpoint se decidió examinar la triple store que estamos usando a lo largo de todo este proyecto para el almacenamiento de información, GraphDB.

GraphDB es una base de datos orientada a grafos. Además de actuar como SPARQL endpoint sobre el que realizar querys definidas en SPARQL sobre nuestros datos, ofrece una herramienta similar a Open Refine, OntoRefine.

OntoRefine ofrece de forma similar a OpenRefine la posibilidad de aplicar funciones sobre los datos originales en Grel y mediante una query SPARQL que sigue unas directrices, se genera RDF. Al igual que Open Refine tiene las mismas limitaciones en cuanto a que en el supuesto caso que se quiere convertir un conjunto grande de datasets se tendría que hacer todo de forma manual.

1. VERIFICACIÓN Y EVALUACIÓN

Al empezar a definir la captura de requisitos y los objetivos del proyecto se planteó como uno de ellos la validación del RDF. Pero avanzando en la realización de este documento que actúa como memoria del trabajo realizado surgió la duda de si no sería adecuado tomar esta funcionalidad como pruebas simplemente en vez de ser tomado como un objetivo a cumplir.

El objetivo de la validación de RDF consiste en analizar cada RDF generado para ver si tiene unas características que se consideran necesarias para afirmar que ese RDF es de calidad. Para llevarlo a cabo se construye un documento en SHACL, que es un lenguaje para la validación de RDF contra unas condiciones a definir.

Es decir, para cada RDF generado se debía construir un documento en SHACL para ver si este cumplía las condiciones definidas en él. Al final, por la naturaleza del trabajo que conllevaba la definición de los documentos SHACL y su posterior ejecución se mantuvo como funcionalidad.

Pese a ello se decidió añadir en esta fase lo concerniente a la explicación de lo que los documentos en SHACL probaban en cada dataset.

De esta forma en este apartado separaremos diferentes tipos de pruebas correspondientes a cada una de las fases del proyecto.

# Generación de RDF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | Generación de RDF para celdas con valores vacíos | Se genera RDF para las celdas que poseen valor | No se genera RDF para ninguno de los elementos de las columnas que posean valores vacíos | **ERROR\*** |

Tabla 17: Pruebas en la generación de RDF

\*Fallo en el identificador 1: El error se corrigió eliminando los valores vacíos del dataset antes de la generación de RDF.

# Testeo calidad RDF generado

## Testeo calidad RDF correspondiente a la calidad del aire

### Respecto a las observaciones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La observación debe tener asociada una fecha | La observación tiene asociada una fecha | Resultado esperado | **CORRECTO** |
| 2 | La observación debería tener asociada una geo-localización | La observación tiene asociada una geo localización | Resultado esperado | **CORRECTO** |
| 3 | La observación debería tener asociada diferentes mediciones | La observación tiene asociadas diferentes mediciones | Resultado esperado | **CORRECTO** |

Tabla 18: Pruebas en el testeo de la calidad del aire respecto a las observaciones

### Respecto a las mediciones con resultados de valores numéricos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La medición de CO debería tener asociado un label | La medición de CO tiene asociado un label | Resultado Esperado | **CORRECTO** |
| 2 | La medición debería tener un valor numérico como resultado asociado | La medición tiene un valor numérico como resultado asociado | Resultado Esperado | **CORRECTO** |
| 3 | El resultado de la medición debería tener asociado una unidad de medida | El resultado de la medición tiene asociado una unidad de medida | Resultado Esperado | **CORRECTO** |

Tabla 19: Pruebas en el testeo del RDF correspondiente a la calidad del aire respecto a las mediciones con resultados numéricos

### Respecto a las mediciones con resultados con valores literales

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La medición de CO debería tener asociado un label | La medición de CO tiene asociado un label | Resultado Esperado | **CORRECTO** |
| 2 | La medición debería tener un valor numérico como resultado asociado | La medición tiene un valor numérico como resultado asociado | Resultado Esperado | **CORRECTO** |

Tabla 20: Pruebas en el testeo del RDF correspondiente a la calidad del aire respecto a las mediciones con resultados con valores literales

## Testeo calidad RDF correspondiente a las estaciones meteorológicas

### Respecto a las observaciones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La observación debe tener asociada una fecha | La observación tiene asociada una fecha | Resultado esperado | **CORRECTO** |
| 2 | La observación debería tener asociada una geo-localización | La observación tiene asociada una geo localización | Resultado esperado | **CORRECTO** |
| 3 | La observación debería tener asociada diferentes mediciones | La observación tiene asociadas diferentes mediciones | Resultado esperado | **CORRECTO** |

Tabla 21: Pruebas en el testeo del RDF correspondiente a las estaciones meteorológicas respecto a las observaciones

### Respecto a las mediciones

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La medición de CO debería tener asociado un label | La medición de CO tiene asociado un label | Resultado Esperado | **CORRECTO** |
| 2 | La medición debería tener un valor numérico como resultado asociado | La medición tiene un valor numérico como resultado asociado | Resultado Esperado | **CORRECTO** |
| 3 | El resultado de la medición debería tener asociado una unidad de medida | El resultado de la medición tiene asociado una unidad de medida | Resultado Esperado | **CORRECTO** |

Tabla 22: Pruebas en el testeo del RDF correspondiente a las estaciones meteorológicas respecto a las mediciones

### Testeo calidad RDF correspondiente a las retribuciones nominativas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | El contrato debería tener asociado un label | El contrato tiene asociado un label | Resultado Esperado | **ERROR** |
| 2 | El contrato debería tener asociado un empleado | El contrato tiene un empleado asociado | Resultado Esperado | **CORRECTO** |
| 3 | El contrato debería tener asociada la fecha en la que se formalizo | El contrato tiene asociada la fecha en la que se formalizo | Resultado Esperado | **CORRECTO** |
| 4 | El contrato debería tener asociada la fecha en la que terminará | El contrato tiene asociada la fecha en la que terminará | Resultado Esperado | **CORRECTO** |
| 5 | El contrato debería tener asociado el código del departamento en el que se encontrará el puesto | El contrato tiene asociado el código del departamento en el que se encontrará el puesto | Resultado Esperado | **CORRECTO** |
| 6 | El contrato debería tener asociados datos respecto a su departamento gerente | El contrato tiene asociados datos respecto a su departamento gerente | Resultado Esperado | **CORRECTO** |
| 7 | El contrato debería tener asociado el nombre del centro orgánico del puesto | El contrato tiene asociado el nombre del centro orgánico del puesto | Resultado Esperado | **CORRECTO** |
| 8 | El contrato debería tener asociado el identificador del centro orgánico del puesto | El contrato tiene asociado el identificador del centro orgánico del puesto | Resultado Esperado | **CORRECTO** |
| 9 | Sólo si el contrato ha sufrido alguna modificación se debería almacenar la fecha en la que se realizo | Si el contrato ha sido modificado se almacena la fecha en la que se realizo | Resultado Esperado | **CORRECTO** |
| 10 | El contrato debería tener asociado un sueldo correspondiente al puesto | El contrato tiene asociado un sueldo correspondiente al sueldo | Resultado Esperado | **CORRECTO** |
| 11 | El contrato debería tener una 'descripción' del puesto al que se refiere | El contrato tiene asociada una 'descripción' del puesto al que se refiere | Resultado Esperado | **CORRECTO** |

Tabla 23: Pruebas en el testeo del RDF correspondiente a las retribuciones nominativas

## Testeo calidad RDF correspondiente a los contratos de trabajo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | El contrato debería tener asociado un label | El contrato tiene asociado un label | Resultado Esperado | **ERROR** |
| 2 | Sólo si el contrato ha sufrido alguna modificación se debería almacenar la fecha en la que se realizo | Si el contrato ha sido modificado se almacena la fecha en la que se realizo | Resultado Esperado | **CORRECTO** |
| 3 | El contrato debería tener asociado el código del puesto al que se refiere | El contrato debería tener asociado el código del puesto al que se refiere | Resultado Esperado | **CORRECTO** |
| 4 | El contrato debería tener almacenada la fecha en la que se realizó la descarga de datos | El contrato tiene almacenada la fecha en la que se realizó la descarga de datos | Resultado Esperado | **CORRECTO** |
| 5 | El contrato debería tener asociado una dotación en referencia al puesto al que se refiere | El contrato tiene asociado una dotación en referencia al puesto al que se refiere | Resultado Esperado | **CORRECTO** |
| 6 | El contrato debería tener asociado el código del departamento en el que se encontrará el puesto | El contrato tiene asociado el código del departamento en el que se encontrará el puesto | Resultado Esperado | **CORRECTO** |
| 7 | El contrato debería tener asociado el código del centro de destino del puesto | El contrato tiene asociado el código del centro de destino del puesto | Resultado Esperado | **CORRECTO** |
| 8 | El contrato debería tener asociado el código del centro orgánico del puesto | El contrato tiene asociado el código del centro orgánico del puesto | Resultado Esperado | **CORRECTO** |
| 9 | El contrato debería tener asociado el nombre del departamento gerente del puesto | El contrato tiene asociado el nombre del departamento gerente del puesto | Resultado Esperado | **CORRECTO** |
| 10 | El contrato debería tener asociado el nombre del centro orgánico del puesto | El contrato tiene asociado el nombre del centro orgánico del puesto | Resultado Esperado | **CORRECTO** |
| 11 | El contrato debería tener asociado el nombre del centro destino del puesto | El contrato tiene asociado el nombre del centro destino del puesto | Resultado Esperado | **CORRECTO** |
| 12 | El contrato debería tener asociado el código de la categoría retributiva del puesto | El contrato tiene asociado el código de la categoría retributiva del puesto | Resultado Esperado | **CORRECTO** |
| 13 | El contrato debería tener asociado el código del perfil lingüístico del puesto | El contrato tiene asociado el código del perfil lingüístico del puesto | Resultado Esperado | **CORRECTO** |
| 14 | El contrato debería tener asociado un sueldo | El contrato tiene asociado un sueldo | Resultado Esperado | **CORRECTO** |

Tabla 24: Pruebas en el testeo del RDF referente a los contratos laborales

# SPARQL Endpoint

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La query es sintácticamente correcta | Se ejecuta la consulta que devolverá o no resultados | Resultado esperado | **CORRECTO** |
| 2 | La query es sintácticamente incorrecta | Mensaje de error avisando de que la query tiene un error de sintaxis | Resultado esperado | **CORRECTO** |
| 3 | La query no es de tipo CONSTRUCT ni SELECT | Mensaje de error avisando de que sólo se pueden ejecutar querys de tipo CONSTRUCT o SELECT | Resultado esperado | **CORRECTO** |

Tabla 25: Pruebas sobre el SPARQL endpoint

# Tabla

Estas pruebas se realizan sobre la funcionalidad presente en la interfaz web en la que se genera una tabla representando los resultados obtenidos de la query ejecutada en el SPARQL endpoint cuando esta es de tipo CONSTRUCT o SELECT.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La consulta es de tipo CONSTRUCT o SELECT | Se genera la tabla a partir de los resultados obtenidos de la query | Resultado esperado | **CORRECTO** |
| 2 | La consulta no es de tipo CONSTRUCT ni SELECT | Mensaje de error, no se genera la tabla | Resultado esperado | **CORRECTO** |
| 3 | La query no genera resultados | Mensaje de error, no se genera la tabla | Resultado esperado | **CORRECTO** |
| 4 | La query genera resultados | Se genera la tabla | Resultado esperado | **CORRECTO** |

Tabla 26: Pruebas sobre la tabla que se genera al realizar una consulta sobre el SPARQL endpoint

# Grafo

Estas pruebas se realizan sobre la funcionalidad presente en la interfaz web en la que se genera un grafo representando los datos cuando la query ejecutada en el SPARQL endpoint es de tipo CONSTRUCT.

En este caso no se probará si se generará el grafo cuando la query no haya obtenido resultados ya que ese error se coteja y se soluciona en el caso de prueba 3 de la tabla ya que la funcionalidad que genera el grafo imposibilita al usuario de solicitar la representación en forma de grafo hasta que los datos hayan sido representados en forma de tabla.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Descripción | Resultado Esperado | Resultado Obtenido | Funcionamiento |
| 1 | La consulta no es de tipo CONSTRUCT | Mensaje de error, no se genera el grafo y se mantienen los resultados representados en forma tabular | Resultado esperado | **CORRECTO** |
| 2 | Se ejecuta una query construct que devuelve un resultado con tamaño adecuado | Se genera el grafo para los resultados obtenidos de la query | Resultado Esperado | **CORRECTO** |
| 3 | Se ejecuta una query construct cuyo resultado es demasiado grande para ser representado en forma de grafo | Mensaje de error, no se genera el grafo y se mantienen los resultados representados en forma tabular | Resultado esperado | **CORRECTO** |
| 4 | Al hacer clic en un elemento para obtener información sólo de ese recurso, la query automática que se ejecuta no obtiene resultados | Mensaje de error y se mantiene el grafo representado hasta el momento | Resultado esperado | **CORRECTO** |
| 5 | Al pasar el ratón sobre un nodo para focalizar el grafo en él y sus adyacentes el identificador del nodo contiene caracteres especiales | Se focaliza la parte del grafo correspondiente al nodo y sus adyacentes | No se produce la focalización correctamente | **ERROR\*** |
| 6 | Al pasar el ratón sobre un nodo para focalizar el grafo en él y sus adyacentes el identificador del nodo no contiene caracteres especiales | Se focaliza la parte del grafo correspondiente al nodo y sus adyacentes | Resultado Esperado | **CORRECTO** |

Tabla 27: Pruebas sobre la tabla que se genera al realizar una consulta sobre el SPARQL endpoint

\*Fallo en identificador 4: Este error se producía porque la función creada que realizaba las focalizaciones estaba diseñada de tal manera que no soportaba realizar búsquedas de elementos HTMLs que contuviesen identificadores con caracteres especiales. Se corrigió pasando un filtro a los identificadores que eliminase esos caracteres especiales para que la función funcionase correctamente.

1. CONCLUSIONES Y TRABAJO FUTURO

Una vez concluido el proyecto volviendo la vista atrás y analizando los objetivos planteados en sus inicios, se puede considerar que han sido cumplidos. Bien es cierto que cuando se planteó el proyecto y sus objetivos, solo se tenía la idea de lo que se quería conseguir, no los medios para llevarlo a cabo por lo que la mayoría de las herramientas necesarias para su ejecución se descubrieron durante las fases de análisis.

Se llegó a un punto en el que fue evidente que objetivos a los que se había dado mucho peso como objetivos principales por su relevancia en el proceso de generación de datos enlazados, no aportaban demasiada carga de trabajo al proyecto ya que las herramientas para ejecutarlos no necesitaban demasiada intervención de parte del alumno, y objetivos a los que sin embargo se los había tomado como secundarios o adicionales al proyecto tomaron más tiempo del esperado o se decidió invertir más tiempo en ellos.

Ese fue el caso del servidor Linked Data, en un principio se pensó que este objetivo conllevaría un tiempo considerable de atención, ya sea invertido en investigación o programándolo, pero más tarde cuando se llevó a cabo fue evidente que no era necesario dedicarle demasiadas horas dado que las herramientas disponibles necesitaban básicamente ser configuradas.

Además, al ser este un proyecto en el que el objetivo primordial era generar RDF de calidad y adecuado a los datasets originales, no se pensó demasiado en su parte visual. En la mayoría de las triples stores en las consultas que se ejecutan sobre ellas los datos se presentan de forma tabular, y esa era la idea inicial de lo que se planeaba construir cuando se planteaba al inicio del proyecto la idea de la creación de un SPARQL endpoint. En cambio, al descubrir las capacidades de la librería D3, que permite la generación de gráficos dinámicos, a este apartado se le dedico mucho más tiempo del esperado para llegar a conseguir lo que se presenta actualmente, que se considera mejora la experiencia del usuario en cuanto a que ve de forma gráfica los resultados de sus consultas y puede navegar a través del grafo. Esta es una funcionalidad que cada vez más triple stores intentan adaptar a sus servicios y nos complace haberlo podido realizar en este trabajo.

Al pensar en mejoras o trabajo a futuro, se considera interesante crear un asistente que partiendo de un banco de datos con ontologías ya existentes asesore al usuario en la creación de RDF. El usuario podría intentar definir la naturaleza de sus datos de partida y su contenido en sí para ayudar a que el asistente le haga sugerencias adecuadas. A partir de ahí, él podría elegir de entre las ontologías planteadas cuáles se adaptarían mejor a su información y empezar con el proceso de generación de RDF. Esta herramienta tendría gran utilidad ya que ahorraría en gran medida el proceso de investigación para la elección de ontologías.

Además, debido a la complejidad y la comúnmente poca experiencia en SPARQL, realizar consultas sobre datos representados en RDF puede resultar complicado y tedioso. Sería interesante también analizar la viabilidad de crear un “traductor” de lenguaje humano a SPARQL, definir unas directrices para que el usuario exprese en lenguaje de a pie siguiendo unas estructuras definidas por el programador los datos que quiere obtener y que el sistema de alguna forma transforme esa petición a SPARQL.

Para concluir, a modo de opinión personal, se considera que la realización de este proyecto ha supuesto un gran crecimiento profesional y ha aumentado las capacidades de resolución e iniciativa del alumno. No solo se ahondado en tecnologías inexploradas anteriormente, sino que se ha trabajado con varios lenguajes desconocidos hasta el momento por él.

Todas estas novedades en cuanto a tecnologías y lenguajes produjo en varias ocasiones atascos en la realización de este trabajo, lo que ocasionó retrasos en la finalización de ciertos apartados, pero como se comentaba anteriormente, esto promovió la resolución del alumno ya que éste no contaba con referentes expertos en ciertas tecnologías a los que consultar, sólo información en Internet o foros de ayuda y al ser los Datos Enlazados un tema novedoso, en ciertos casos, era escasa.

1. BIBLIOGRAFÍA

[1] Hallo, María Asunción. Martínez González, M. Mercedes; Fuente Redondo, Pablo de la. Las tecnologías de Linked Data y sus aplicaciones en el gobierno electrónico. // Scire. 18:1 (en.-jun. 2012) 49-61. ISSN 1135-3716

[2] Berners-Lee, Tim (2006). Design Note: Linked Data. http://www.w3.org/DesignIssues/LinkedData.html (2011- 07-02)

[3] Cyganiak, Richard; Bizer, Chistian (2011). Pubby. A Linked Data Frontend for SPARQL Endpoints. http://www4. wiwiss.fu-berlin.de/pubby/ (2011-07-02).

[6] Méndez Rodríguez, E. M. (1999). RDF: Un modelo de metadatos flexible para las bibliotecas digitales del próximo milenio. Jornades catalanes de documentació (7. 1999. Barcelona). Les biblioteques i els centres de documentació al segle XXI: peça clau de la societat de la informació. 4, 5 i 6 de novembre de 1999. Palau de Congressos de Barcelona. Barcelona: Col.legi Oficial de Bibliotecaris-Documentalistes de Catalunya, pp. 487-498.

[4] https://www.w3.org/RDF/

[5] <https://github.com/JoshData/rdfabout/blob/gh-pages/intro-to-rdf.md>

[6] <https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones> ->DEF API

[7] <https://es.wikipedia.org/wiki/Valores_separados_por_comas> -> def csv

<https://es.wikipedia.org/wiki/RDF/XML> -> DEF RDF/XML

<https://es.wikipedia.org/wiki/Turtle_(sintaxis)> 🡪 TURTLE

<https://es.wikipedia.org/wiki/JSON-LD> -> JSON LD

<https://es.wikipedia.org/wiki/Framework> -> FRAMEWORK

<https://es.wikipedia.org/wiki/Resource_Description_Framework> --> RDF

<https://es.wikipedia.org/wiki/World_Wide_Web_Consortium> -> W3C

<https://www.w3.org/TR/shacl/> -> SHACL

<http://opendata.euskadi.eus/catalogo/-/calidad-aire-en-euskadi-2017/> -> calidad del aire

<http://opendata.euskadi.eus/catalogo/-/estaciones-meteorologicas-lecturas-recogidas-en-2017/> -> estaciones metereologicas

<http://opendata.euskadi.eus/catalogo/-/relaciones-de-puestos-de-trabajo-de-los-departamentos-y-organismos-autonomos-de-la-administracion-de-la-comunidad-autonoma/>

🡪 relaciones puestos de trabajo

<http://opendata.euskadi.eus/catalogo/-/retribuciones-altos-cargos-y-personal-eventual-gobierno-vasco-y-organismos-autonomos-y-entes-publicos/> 🡪 retribuciones nominativas

<http://www.mapama.gob.es/es/calidad-y-evaluacion-ambiental/temas/atmosfera-y-calidad-del-aire/Cap3_Criterios_evaluacion_tcm7-419007.pdf> --> lo de la ley ambiental