# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA DE SISTEMAS

# DESARROLLO DE UN SISTEMA DE MANEJO DE REGLAMENTACIONES USANDO TECNOLOGÍAS DE LINKED DATA

**TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS INFORMÁTICOS Y DE COMPUTACIÓN**

**FELIPE DAVID BORJA TIPÁN**

felipe.borja@epn.edu.ec

**VÍCTOR ANDRÉS CUSME CASTRO**

victor.cusme@epn.edu.ec

**Director:** Dra. María Asunción Hallo Carrasco

maria.hallo@epn.edu.ec

**Quito, Diciembre 2017**

# CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Felipe David Borja Tipán y Víctor Andrés Cusme Castro, bajo mi supervisión.

_____

**Dra. María Asunción Hallo Carrasco**
**DIRECTORA DEL PROYECTO**

# DECLARACIÓN

Nosotros, Felipe David Borja Tipán y Víctor Andrés Cusme Castro, declaramos bajo juramento que el trabajo aquí escrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.

| | |
|---|---|
| **Felipe David Borja Tipan** | **Víctor Andrés Cusme Castro** |

# DEDICATORIA

*Dedico este proyecto a mis padres, mi hermano, mi esposa e hija, quienes forman los pilares que sostienen mi vida, ya que esta por sí sola no tiene sentido, sino que ustedes son quienes le han dado un rumbo y motivo para cada día despertar con la ilusión de "Ser más para servir mejor".*

Borja Tipán Felipe David

# DEDICATORIA

*A:*

*Toda mi familia, por ser el pilar fundamental tanto en mi formación personal como también profesional. También por brindarme todo su cariño y apoyo cuando más lo necesitaba.*

*Todos mis amigos, por formar parte de mi vida y por haber compartido tantos buenos y malos momentos juntos.*

Cusme Castro Víctor Andrés

# AGRADECIMIENTOS

Agradezco en primer lugar a Dios y mi Dolorosa, en quienes creo y me brindan la fortaleza diaria para hacer mejoras cosas y contribuir a que la parte del mundo en el que vivo sea mejor.

A mi familia que me ha visto crecer día a día y me ha dado el aliento necesario para seguir siempre para adelante, a través de sus consejos y apoyo incondicional.

A mi amigo Andrés, compañero en la elaboración de este proyecto, ya que hemos compartido una gran experiencia al trabajar en equipo y aprender nuevas tecnologías a través de la investigación y experimentación que dieron como fruto la conclusión de este proyecto.

A la Doctora María Hallo quien nos brindó su guía solventando las inquietudes que se plantearon a lo largo de la elaboración del proyecto.

A mis amigos, con quienes compartimos experiencias en nuestro crecimiento académico y profesional en la carrera.

Borja Tipán Felipe David

# AGRADECIMIENTOS

Agradezco a Dios por haberme dado la fortaleza para seguir adelante y cumplir con todo lo que me he propuesto, por ayudarme a superar los obstáculos que se me han presentado a lo largo de mi vida personal y estudiantil.

A mi familia por haberme guiado de la mejor forma durante todo este tiempo, por toda la ayuda que me han brindado y que gracias a su apoyo he podido levantarme de los más duros momentos.

A mi amigo Felipe, por ser mi compañero en la elaboración de este proyecto y también por ser un excelente amigo con el que he compartido muy gratos momentos a lo largo de mi vida estudiantil universitaria y también laboral.

A la Doctora María Hallo, por habernos guiado en la elaboración de este proyecto dándonos las pautas necesarias que requeríamos para lograr culminarlo con éxito.

A todos mis amigos que han estado siempre conmigo, gracias por su amistad y por su apoyo.

Cusme Castro Víctor Andrés

# Contenido

## LISTA DE TABLAS

## LISTA DE FIGURAS

# RESUMEN

Este proyecto tiene como propósito el desarrollar un sitio Web de manejo de reglamentaciones para la Escuela Politécnica Nacional, usando un administrador de contenidos (CMS) y tecnologías de *Linked Data* como es el caso de RDF y SPARQL, en vista que actualmente no se dispone de un repositorio de datos abiertos enlazados.

Para poder desarrollar el proyecto fue necesaria una primera etapa de investigación, para el entendimiento de las tecnologías de *Linked Data* existentes y la definición de cuáles eran necesarias aplicar para el desarrollo del proyecto.

La metodología de desarrollo de software usada en el proyecto fue SCRUM que permitió realizar la especificación de requerimientos, el trabajo en equipo y la entrega progresiva de resultados previos, para lo cual el proyecto fue dividido en seis *Sprints* en los cuáles se definieron las historias de usuario, las tareas y sus pruebas. Se adicionó funcionalidad al CMS a través de módulos externos al *Core* central.

Posterior se presentan los resultados frutos del desarrollo de cada uno de los *Sprints*, evidenciando el trabajo realizado para cumplir con cada una de las historias de usuario y los resultados que se obtuvieron, siendo estos los deseados, al tener un sitio Web que permite el manejo de modificaciones sobre Reglamentaciones, con un manejo del historial de cambios y el acceso a la información del sitio Web a través de consultas SPARQL.

Como punto final se tienen las conclusiones y recomendaciones obtenidas del desarrollo y la aplicación del proyecto conforme al uso de la metodología mencionada.

Palabras clave: CMS, *Linked Data,* RDF, *Scrum*, SPARQL.

# 1. INTRODUCCIÓN

## 1.1. Problema

En la Escuela Politécnica Nacional (EPN) no existe un sistema de manejo de información de reglamentos que responda vía Web a requerimientos especiales tales como consultar el historial de modificaciones de un reglamento; esto se debe a que el proceso de actualización de reglamentos en la Escuela Politécnica Nacional se lo realiza en forma manual; adicionalmente no se dispone de información publicada con datos abiertos, *Linked Data*, para ser consultada por agentes inteligentes. Por tal razón, este proyecto desarrolla un sistema de manejo de reglamentaciones que permite ofrecer una solución para el almacenamiento y recuperación de la información, la misma que estará enlazada a la Web semántica utilizando estándares de manejo documental con la utilización de herramientas de Web semántica como es el caso de un administrador de contenidos y RDF.

## 1.2. Administrador de contenidos (CMS)

Un administrador o gestor de contenidos nos permite crear sitios y aplicaciones web que necesitan flexibilidad al momento de ser desarrolladas. Se analizaron dos CMS que se detallan en la tabla 1, viables para el uso del proyecto:

*Tabla 1: Lista de software viable para su uso*

| Software viable de uso |
|---|
| OpenLink Virtuoso |
| Drupal |

***OpenLink Virtuoso***

OpenLink Virtuoso se trata de un CMS que tiene como ventaja el proporcionar una Web semántica que brinda un fácil acceso a la información manejada a través del manejo de data no solo SQL, por lo que permite el manejo RDF por defecto, al admitir la gestión de datos donde los tipos de relación entidad (datos) se representan como tablas relacionales o grafos RDF. Esto se puede evidenciar de mejor manera en la figura 1 donde se aprecia estructura de este software.

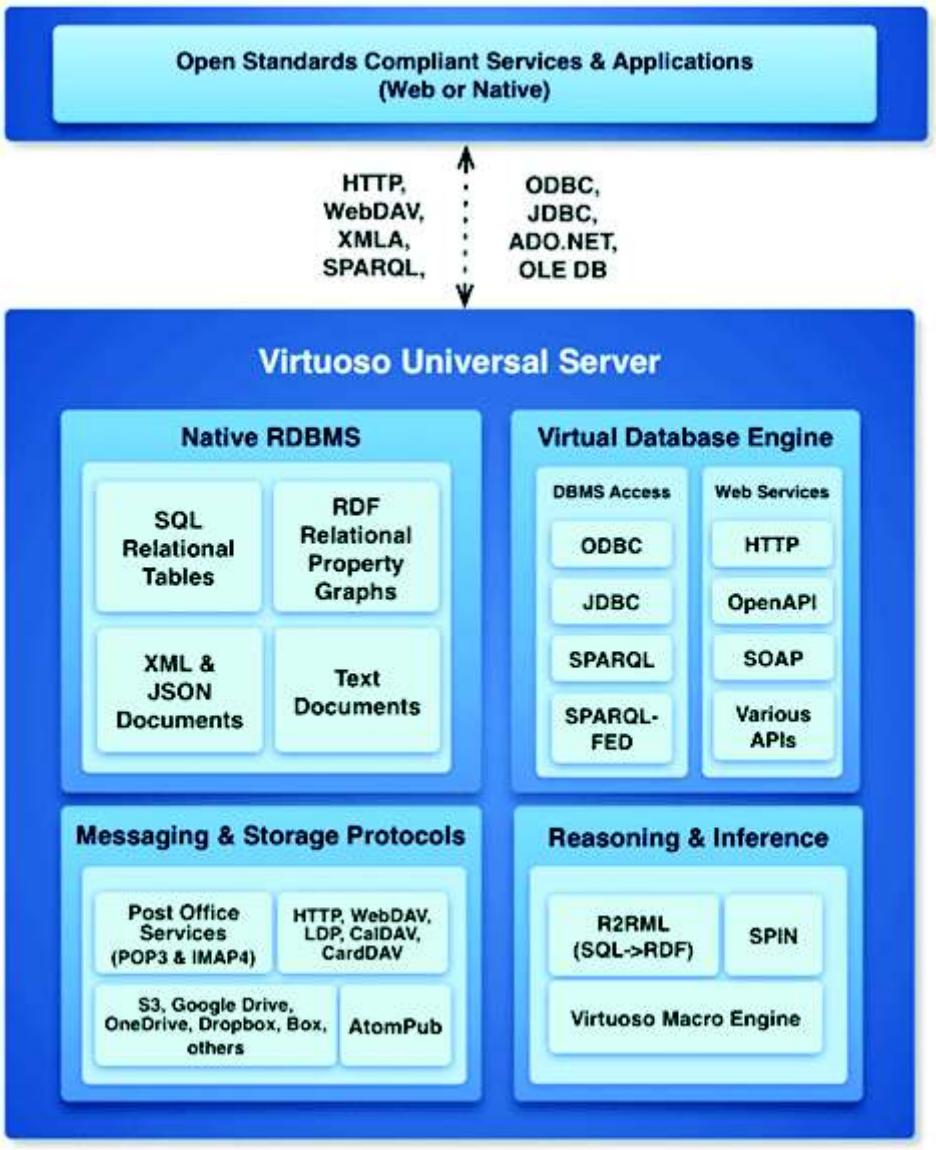


*Figura 1: Estructura de Software OpenLink Virtuoso [1]*

Con la estructura que maneja OpenLink Virtuoso se pensó en usarlo para el desarrollo del proyecto, pero ya que no existe una amplia documentación que nos permita usarlo de la mejor forma se tomó la decisión de dejarlo de lado para la implementación del sitio Web.

***Drupal***

Se trata de un CMS de código abierto modular y escalable, razón por la cual se realizó una investigación acerca de las bondades y uso del mismo, y sus diferentes versiones, buscando la versión que más compatibilidad tenga con el manejo de Web semántica, siendo esta Drupal 7, la cual tiene una gran flexibilidad al poder implementar varios módulos propios del *Core* o de contribuciones al proyecto Drupal, teniendo una estructura de capas tal como se presenta en la figura 2:

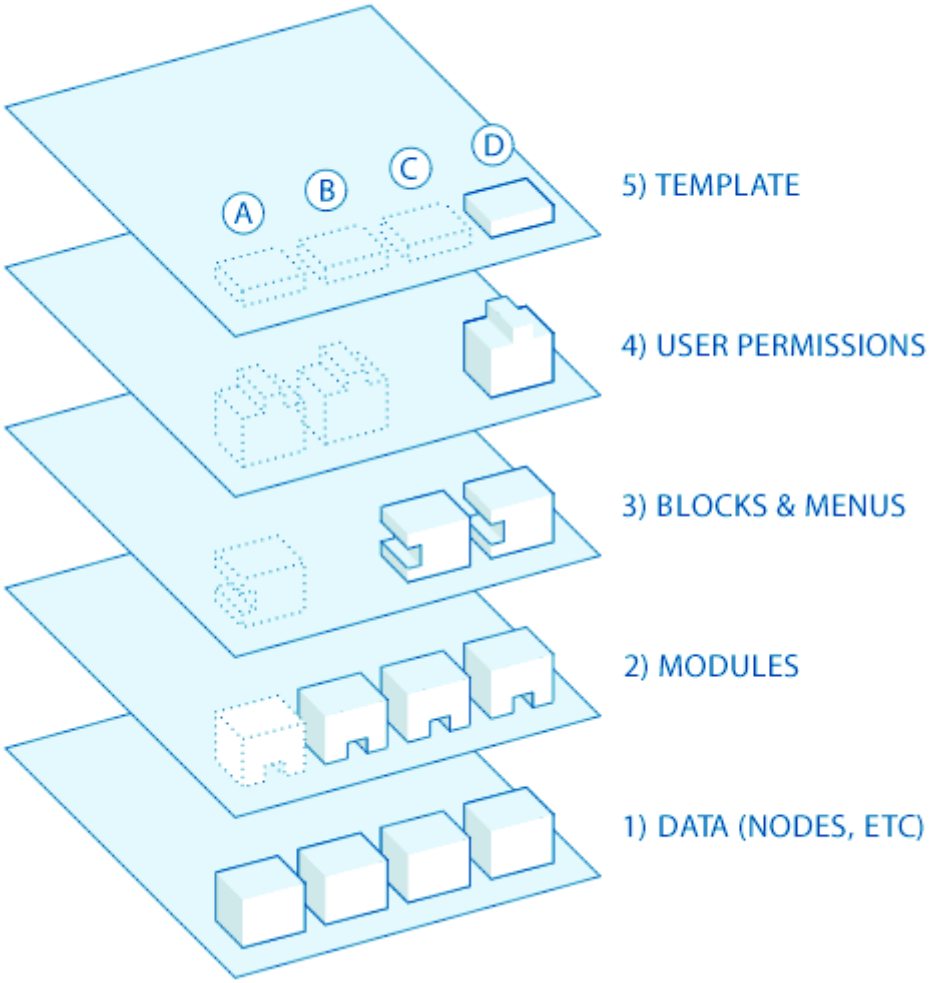


*Figura 2: Estructura de CMS Drupal [2]*

En la primera capa se tiene una colección de nodos que son el conjunto de datos que debe ser una entrada de datos para que pueda ser mostrada en el sitio. La siguiente capa es donde se encuentran los módulos que son un complemento a la funcionalidad base de este CMS. La tercera capa alberga a bloques para poder colocarlos en varias regiones dentro del CMS para su presentación visual y además alberga menús que son los links internos que proporcionan los enlaces a todas las páginas que hayan sido creadas con Drupal. La siguiente capa tiene los permisos de usuario que deben ser definidos y otorgados por el administrador y finalmente la capa superior que tiene el tema del sitio que se compone de XHTML y CSS principalmente [2]; gracias a la modularidad [3] se lo eligió para el desarrollo del proyecto.

## 1.3. Web Semántica

Es una Web extendida, que es dotada de mayor significado, semántica, en la que cualquier usuario dentro de Internet podrá encontrar respuestas a varias preguntas de forma más sencilla, al brindar mayor facilidad para la búsqueda de información gracias al uso de una

infraestructura común que puede ser entendida por todas las entidades que estén dentro de un dominio [4].

Debido a que la transferencia de información en la actualidad es continua y se tiene acceso a millones de recursos a lo largo de la Web, se han originado problemas referentes a la sobrecarga de información y heterogeneidad de fuentes de información [4], razón por la cual la Web semántica soluciona el problema al hacer que el software se vuelva capaz de procesar su contenido, entenderlo, y hacer deducciones lógicas para resolver varios problemas.

Esta solución se logra gracias a que se añaden metadatos semánticos y ontológicos al contenido a la *Word Wide Web*, lo que permite una mejor organización de la información, asegurando que las búsquedas sean más precisas por su significado antes que por un contenido textual.

El concepto de Web semántica es prácticamente nuevo en el desarrollo de sitios Web, por lo que presenta inconvenientes para quienes quieran hacer uso del mismo, ya que se vuelve costoso y laborioso el adaptar los documentos de Internet para que puedan ser procesados semánticamente además de ser necesario unificar los estándares semánticos para proveer relaciones de equivalencia entre conceptos, es decir que termine siendo independiente del lenguaje natural, sea este inglés, español u otros [5]. La ayuda que nos brinda es que el sitio Web adquiere un mayor contenido semántico, de fácil comprensión para la máquina.

## 1.4. *Resource Description Framework* (RDF)

RDF es un modelo estándar para para el intercambio de datos a través de la Web, cuyas especificaciones son publicadas por la *World Wide Web Consortium* (W3C) [6]. Permite además ampliar la estructura de enlace en la Web ya que hace uso de “triples RDF [7]”, con la finalidad de que crear una relación entre cosas para que puedan ser expuestas y consumidas desde diferentes aplicaciones [6]. Provee de gran facilidad para descomponer el conocimiento, guardar la información y su estructura, algo similar al lenguaje humano.

RDF es similar al XLM en su estructura, pero con la diferencia que RDF fue concebido para la representación de conocimiento, no únicamente de datos como XML. Con la ayuda de tres entidades, RDF puede hacer referencia hacia un hecho, y al colocar varios hechos juntos se puede llegar al conocimiento. Otro aspecto importante es que RDF tiene gran compatibilidad para trabajar con información distribuida, es decir que no necesariamente toma el conocimiento de un archivo específico, sino que puede tomar dicho conocimiento de varios archivos que no necesariamente estén en un mismo sitio sino que estén a lo largo de la Web a través del enlace de documentos que usen un vocabulario en común o permitiendo que cualquier documento tenga la facilidad de usar cualquier vocabulario; como se puede notar

esto permite adquirir y usar conocimiento de varias fuentes de información, haciendo que esta sea relacionado con seguridad y además presente mayor flexibilidad.

*Triples de Conocimiento*

RDF permite la descomposición del conocimiento en pequeñas partes, a las cuales se las denomina triples o tripletas que guardan reglas para mantener semántica de dichas partes descompuestas [8].

El objetivo de la descomposición es generar un grafo dirigido y etiquetado, en el que las aristas representan un hecho o la relación entre dos cosas. Para mantener la semántica es importante identificar un sujeto, un predicado y un objeto [8].

- Sujeto: es todo aquello que está en una arista,
- Predicado: es el tipo de la arista o su etiqueta.
- Objeto: es todo aquello que está al final de la arista.

Para expresar información en forma de tripletas se puede hacer un símil con el vocabulario humano, como se muestra en la tabla 2 de la siguiente forma:

*Tabla 2: Información en tripletas.*

| *Sujeto* | *Predicado* | *Objeto* |
|---|---|---|
| Yo | Tengo | Mi_casa |
| Mi_casa | Tiene | Mi_cuarto |
| Mi_casa | Tiene | Mi_ropa |
| Mi_casa | Está_en | Quito |

Cada una de las oraciones expresan un hecho, a esto se lo conoce como tripleta, así como ocurre en el lenguaje humano, cada parte de la oración o nombres no tiene significado por sí solos, por lo que no es importante los nombres que se seleccionen para expresar algo, mientras no se tenga constancia en su uso; a dichos nombres se los conoce como recursos. Como el objetivo de RDF es ser publicar las tripletas de información no tiene sentido el publicar nombres como “yo” o “mi casa”, por el contrario, deben ser nombres globales, por lo que formalmente los nombres para sujetos, predicados y objetos deben ser Identificadores Uniformes de Recursos (URIs). Las URIs pueden tener el mismo formato de una dirección Web por lo cual no debe ser extraño encontrar a los recursos como una dirección Web extensa.

A través del manejo de RDF se permite usar una ontología que nos facilite limitar el dominio del problema, definiendo etiquetas propias para reglamentaciones que sean relacionadas al contenido ingresado en el CMS usado, de tal modo que sobre esta información sea posible ejecutar consulta de los datos a través de consultas SPARQL.

## 1.5. Lenguaje de Ontologías Web (OWL)

El Lenguaje de Ontologías Web (OWL) nos permite describir el significado de cada uno de los términos que son usados en la Web [9]. Una ontología permite definir la terminología que será usada para representar una cierta área de conocimiento [10]. Las definiciones de la terminología y las relaciones existentes entre cada uno de los términos serán usadas por computadoras con la finalidad de que puedan ser interpretados y así poder realizar tareas sobre ellos.

OWL proporciona tres lenguajes, en donde cada uno de ellos tiene mayor expresividad sobre el otro. Dichos lenguajes son los siguientes [9]:

- OWL Lite: Permite representar necesidades básicas en la ontología, la cual tenga una organización jerárquica y que contenga relaciones simples. Permite establecer valores cardinales de 0 ó 1.
- OWL DL: Permite tener máxima expresividad garantizando que todas las conclusiones sean computables y que sean devueltas en un tiempo finito. Además, incluye todas las construcciones que son parte de OWL, pero que también posee ciertas restricciones. Es una extensión de OWL Lite.
- OWL Full: Brinda máxima expresividad y libertad sintáctica de RDF pero sin ofrecer garantías computacionales. Es una extensión de OWL DL.

## 1.6. Trabajos previos

El primer trabajo que se tomó en cuenta hace referencia a la propuesta de procesamiento de información para la transformación de catálogos de bibliotecas en *Linked Data* realizado en el año 2014 [11], la cual nos indica el proceso para la publicación de metadatos de bibliotecas como datos vinculados, al representar los datos en RDF y publicarlos usando un *endpoint* para consultas SPARQL.

El siguiente trabajo revisado propone el modelamiento de una ontología adecuada para el manejo de documentación legislativa el cual fue realizado en el año 2015 [12]. Esta ontología puede ser aplicada para documentación jurídica del Ecuador a través del manejo adecuado

de etiquetas XML que permite tener metadatos para describir el contenido de documentos legislativos, sus partes y los enlaces a los cambios realizados.

Finalmente se revisó un trabajo que ejecutó un proyecto para el manejo de la información jurídica a través del lenguaje de marcado XML realizado en el 2009 [13], cuyo nombre se denomina SIDRA y actualmente es un sistema integral de información documental en red ya consolidado que está soportado por una única base de datos y en una única aplicación de gestión documental, que permite el tratamiento, la recuperación, la distribución y el préstamo de todo tipo de documentación o información especializada, pública o restringida, necesaria para el ejercicio de las funciones de la Administración del Principado de Asturias [13].

Es importante anotar que se realizó un trabajo de investigación previa de los conceptos de Web semántica, *Linked Data* y de las herramientas que nos permitieron hacer uso de todos los conceptos investigados, de tal modo que el desarrollo del proyecto se lo enfocó acorde a solucionar el problema planteado en un inicio haciendo uso adecuado de tecnologías de *Linked Data*.

El sitio Web desarrollado tiene como característica fundamental que aplica conceptos de Web semántica, como es el manejo de la información a través de etiquetado RDF dentro un administrador de contenidos (CMS) que permite mantener un registro histórico de los cambios que se realicen sobre un documento, en este caso un reglamento, además de permitir la comparación del contenido que ha sufrido alguna modificación.

## 1.7. Métodos para el manejo de información legislativa por medio de la Web

Se realizó un breve estudio de los métodos que existen para el manejo de información legislativa por medio de la Web, de los cuales se escogieron los siguientes:

### 1.7.1. Manejo a través de XML

El uso del metalenguaje de marcado XLM propone un formato en el que se intercalan marcas en el texto de los documentos con el objeto de distinguir las partes o elementos estructurales del mismo. Existen dos organizaciones a nivel mundial que fueron pioneras en la estandarización de la información legislativa [13]:

- LegalXML en Estados Unidos (sección de OASIS, Organization for the Advancement of Structured Information Standards). [13]
- LEXML en Europa. [13]

Ambas organizaciones persiguen la creación de un grupo de DTD para determinados tipos de documentos jurídicos. Con base a la estandarización de la información legislativa se desarrolló el proyecto SIDRA con tecnología estándar universal XML bajo la plataforma de Tamino que es un sistema de gestión de bases de datos XML nativo de alto rendimiento capaz de almacenar, gestionar, publicar e intercambiar documentos XML en su formato original, basado en tecnologías de Internet de estándar abierto [14] y Dogma que es la aplicación de gestión documental del SIDRA, soportada por Tamino, que almacena de forma nativa los documentos en formato XML. [13]. A través de SIDRA se puede definir de forma abierta y flexible varios modelos de datos y asociarlos a registros de todo tipo de documentos originales como archivos textuales, video, imagen, etc. para garantizar su localización inmediata y la recuperación de su contenido gracias a que dichos registros se almacenan en formato XML. [13]

### 1.7.2. Manejo a través de *Linked Data*

El manejo de información legislativa a través de *Linked Data* tiene como base la publicación de datos que puedan ser consultados a través de consultas SPARQL a través del manejo de la metadata de la información en los siguientes pasos [11]:

- Análisis de fuente de información: implica el análisis de la estructura que maneja el documento, definiendo sus partes como cabeceras, cuerpo, etc. [11]
- Extracción de metadatos: se trata de tener definidos los metadatos con los que se va a realizar el manejo de la información. [11]
- Modelado: se basa en buscar un vocabulario existente o crear uno nuevo acorde a la estructura del documento, de los metadatos; en el caso de crear un nuevo vocabulario se debe: [11]
    - Desarrollar vocabulario y documentarlo.
    - Validar el vocabulario que esté acorde a la metadata que va a manejar.
    - Publicación del set de datos con la metadata.
- Generación RDF: implica seleccionar la tecnología RDF que va a hacer uso, como el caso de las tripletas de información, para lo cual se debe: [11]
    - Mapear el set de datos a formato RDF.
    - Transformar la data a RDF.

- Publicar datos: una vez que se tiene la información en formato RDF se la debe publicar para que pueda ser consultada desde cualquier *endpoint* SPARQL. [11]

# 2. METODOLOGÍA

Existen varias metodologías para el desarrollo de software, pero no existen aquellas que se enfoquen directamente en el desarrollo de un sistema en específico, es decir, en este caso no existen metodologías creadas para el desarrollo de sistemas legislativos. Por tal razón, hemos analizado tanto metodologías ágiles como también las tradicionales para la realización de nuestro proyecto, de las cuales nos hemos enfocado en una metodología ágil antes que en una tradicional, ya que lo que se desea es tener resultados visibles a corto plazo, además de que debe ser adaptable a requerimientos que se vayan dando en conformidad al avance del proyecto.

En dicho análisis se tomaron como referencia las metodologías RUP, MSF, XP y SCRUM para tener una visión de las características básicas que puedan o no tener, las cuales se definierion en función del Módulo 0 del Programa Master de Dirección de Proyectos de la Universidad de Alcalá. [15]

En la tabla 3, se tiene un cuadro comparativo que contiene las características más relevantes de las metodologías antes mencionadas y que nos sirvió para definir cuál de éstas se adapta mejor para utilizarla en el desarrollo del proyecto.

*Tabla 3: Matriz comparativa de metodologías [16].*

| Característica | RUP | MSF | XP | SCRUM |
|---|---|---|---|---|
| Heredan modelos | Sí | Sí | - | - |
| Independiente de tecnologías | - | Sí | - | Si |
| Documentación estricta | Sí | Sí | - | - |
| Estrictamente sistemático | Sí | - | Sí | - |
| Más enfocado en los procesos | Sí | Sí | - | - |
| Más enfocado en las personas | - | - | Si | Sí |
| Resultados rápidos | - | - | Sí | Sí |
| Cliente activo | - | - | Sí | Sí |
| Manejo del tiempo | Sí | Sí | Sí | Sí |
| Refactorización de código | - | - | Sí | - |
| Iterativo | Sí | Sí | Sí | Sí |
| Respuesta a los cambios | - | - | Sí | Sí |

Las características mostradas en la tabla anterior representan los diferentes contextos en que estas metodologías pueden ser desarrolladas, los niveles de riesgos que puede tener cada proyecto, tanto en los diferentes niveles de calidad, así como también en los diferentes tipos de clientes que puedan tener [16]. Al revisar esta matriz podemos concluir que las metodologías que mejor se adaptan a nuestro proyecto son las ágiles, XP y *Scrum*, pero en vista que la base de XP es la programación en pares teniendo a un desarrollador *senior* y a un *junior*, que en nuestro caso no aplica porque ninguno de los desarrolladores puede considerarse senior por el poco conocimiento inicial que se tuvo en el tema y las herramientas a ser usada, se optó por metodología SCRUM en vista a que nos va a permitir obtener resultados rápidos y está enfocado en las personas, es decir en el equipo de trabajo y principalmente que tiene una buena respuesta a los cambios. Los cambios que puedan afectar en el desarrollo del proyecto se centran en el uso de las herramientas para el manejo de tecnologías de *Linked Data* [3] debido a que el conocimiento de las mismas es gradual ya que cuando se empieza a hacer uso de dichas tecnologías verdaderamente se puede apreciar si algún requerimiento no está completo en su descripción o si hace falta incluirlo.

## 2.1. SCRUM

Esta metodología se trata de un marco de trabajo para la correcta gestión del desarrollo de proyectos ágiles, en el que se definen roles y prácticas como principio fundamental para la ejecución de un proyecto software de tal modo que el trabajo colaborativo permite obtener resultados rápidos que agregar valor al negocio. En *Scrum* se realizan entregas parciales de

lo que después de se convertirá en el producto final, tal como se muestra en la figura 3 que ilustra el ciclo de vida de la metodología en mención.

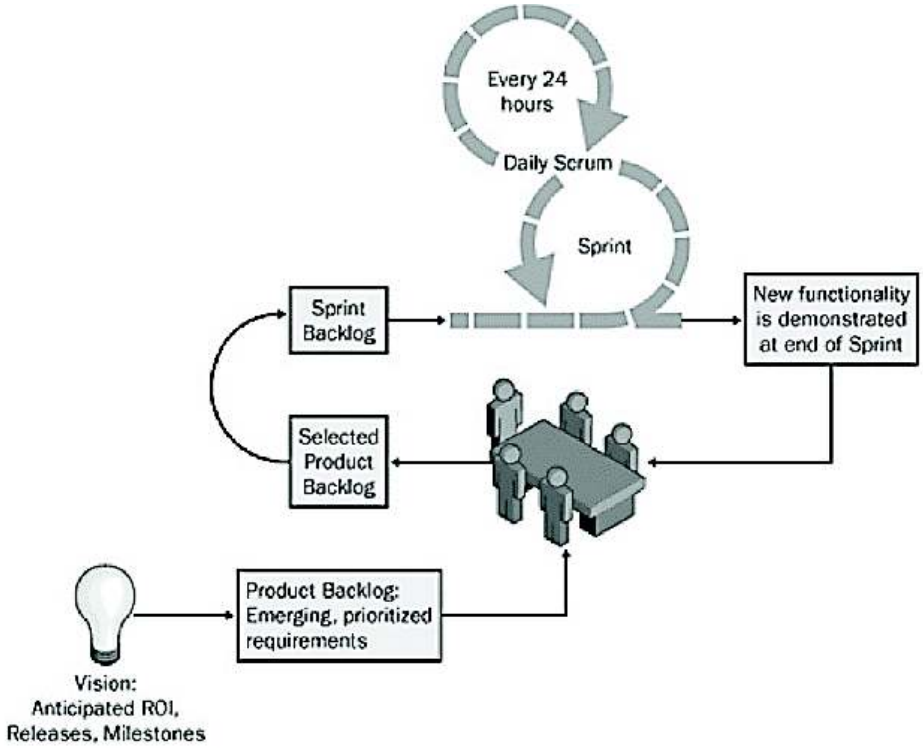


*Figura 3: Ciclo de la metodología Scrum [17]*

Una de las principales ventajas de usar *Scrum* es su rápido aprendizaje ya que los elementos que lo componen pueden ser adaptados para cualquier proyecto software que necesite ver resultados inmediatos.

Las entregas parciales se las hace al final de cada *Sprint*, que corresponde a un ciclo o iteración que tiene como resultado un entregable funcional y de utilidad para el cliente [18]; estos ciclos deben ser cortos, para que se cumpla su objetivo, siendo así que es recomendable que los mismos no tengan una duración mayor a cuatro semanas.

## 2.2. Especificación de requerimientos

Para la especificación de requerimientos de acuerdo a la metodología *Scrum* se van a hacer uso de los elementos que se describen a continuación.

### 2.2.1. Roles

En *Scrum* se pueden definir varios roles que se hacen uso de acuerdo al tamaño del proyecto, de los cuales es necesario contar con los siguientes roles para el presente proyecto: *Product Owner*, *Scrum Master* y *ScrumTeam*, siendo parte de este último los *Developers*.

En la tabla 4 se detallan los roles *Scrum* que participan en el desarrollo de este proyecto, junto con la persona que cumple dicho rol.

*Tabla 4: Roles Scrum para el proyecto*

| Rol | Persona encargada | Descripción |
|---|---|---|
| Propietario del Producto *(Product Owner)* | Dra. María Hallo | Se encarga de transmitir al equipo la perspectiva del negocio. |
| Facilitador Scrum *(Scrum Master)* | Dra. María Hallo | Es quien garantiza que el equipo cuente con lo necesario para desarrollar el proyecto. |
| Grupo Scrum *(Scrum Team)* | Felipe Borja<br>Andrés Cusme | Son los responsables del desarrollo del proyecto. |

## 2.3. Historias de Usuario

Las historias de usuario son escritas por el *Product Owner* y son una representación de un requisito del cliente, es decir de una funcionalidad requerida [19], la cual es escrita en lenguaje natural, no técnico; además debe ser acotada, es decir que debe tratar de ser precisa y no abarcar demasiada funcionalidad, ya que debe ser independiente de otras historias.

Cada historia de usuario debe estar acompañada de sus criterios de aceptación, es decir que deben cumplir sus requerimientos funcionales. Un criterio de aceptación permitirá al *Product Owner* para evaluar si la historia de usuario fue desarrollada según la expectativa del proyecto y se si puede considerar que ha sido realizada satisfactoriamente, además que contribuye para que el *Scrum Team* tenga una visión de hacia dónde debe enfocarse el desarrollo ya que una vez que se concluya con el mismo, debe ser probado de acuerdo a dichos criterios para ser aceptada.

En vista a la funcionalidad deseada en el proyecto se han definido las siguientes historias de usuario, colocadas en una plantilla adecuada [20], que se muestran en la tabla 5, que se lee de izquierda a derecha por cada fila, de acuerdo al enunciado de la historia y sus criterios de aceptación.

*Tabla 5: Historias de usuario y criterios de aceptación para el proyecto*

| Enunciado de la historia | | | | | Criterios de aceptación | | | |
|---|---|---|---|---|---|---|---|---|
| **Identificador (ID) de la historia** | Rol | Característica / Funcionalidad | Razón / Resultado | Número (#) de escenario | Criterio de aceptación (Título) | Contexto | Evento | Resultado / Comportamiento esperado |
| **LMS-0001** | Como usuario final. | Necesito poder acceder al sitio web de Manejo de Reglamentaciones. | Con la finalidad de consultar los reglamentos de la EPN ingresados en el sitio Web. | 1 | URL válida. | En caso que exista el sitio web publicado para el Manejo de Reglamentaciones. | Ingresar al sitio web. | Se presentará una pantalla inicial de bienvenida. |
| | | | | | | | | |
| **LMS-0002** | Como usuario final. | Necesito visualizar un reglamento. | Con la finalidad de revisar su contenido. | 1 | Visualización en un CMS. | En caso que el reglamento esté en el listado. | Seleccionar el enlace. | Se presentará el contenido del reglamento. |
| | | | | | | | | |
| **LMS-0003** | Como usuario final. | Necesito ver las modificaciones del reglamento. | Con la finalidad de revisar el historial de cambios. | 1 | Listado de revisiones del reglamento. | En caso que existan modificaciones. | Seleccionar el enlace de revisiones. | Se presentará una lista de todos los cambios realizados en el reglamento. |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **LMS-0004** | Como usuario final. | Necesito comparar las revisiones del reglamento. | Con la finalidad de revisar los cambios realizados. | 1 | Cambios por inserción. | En caso que existan cambios por inserción de contenido. | Seleccionar las revisiones a comparar. | Se presentará una descripción de la revisión. |
| | | | | 2 | Cambios por modificación. | En caso que existan cambios por modificación de contenido. | Seleccionar las revisiones a comparar. | Se presentará un contraste entre el contenido anterior y el actual junto con la descripción de la revisión. |
| | | | | 3 | Cambios por eliminación. | En caso que existan cambios por eliminación de contenido. | Seleccionar las revisiones a comparar. | Se presentará una descripción de la revisión. |
| | | | | | | | | |
| **LMS-0005** | Como usuario final. | Necesito la versión actual del reglamento en formato PDF. | Con la finalidad de tener un respaldo. | 1 | Visualización del enlace para generar el reglamento en formato PDF. | En caso que se muestre el reglamento. | Visualizar el enlace para generar el PDF. | Se presentará el enlace para generar el reglamento en formato PDF. |
| | | | | 2 | Generación del reglamento en formato PDF. | En caso que se muestre el reglamento. | Seleccionar el enlace para generar el PDF. | Se generará el reglamento en formato PDF. |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **LMS-0006** | Como usuario administrador. | Necesito ingresar un nuevo reglamento. | Con la finalidad de crear nuevos contenidos en el CMS. | 1 | Inserción de un nuevo reglamento como borrador. | En caso de que no exista el reglamento. | Agregar nuevo contenido. | Se presentará el formulario para el ingreso del nuevo reglamento y no será publicado en el sitio Web. |
| | | | | 2 | Inserción de un nuevo reglamento para ser publicado. | En caso de que no exista el reglamento. | Agregar nuevo contenido. | Se presentará el formulario para el ingreso del nuevo reglamento. |
| | | | | | | | | |
| **LMS-0007** | Como usuario administrador. | Necesito hacer modificaciones al reglamento. | Con la finalidad de actualizar su contenido. | 1 | Inserción de nuevo contenido. | En caso que se requiera agregar nuevo contenido. | Editar el reglamento. | Se permitirá ingresar el nuevo contenido junto con la justificación del cambio. |
| | | | | 2 | Modificación de contenido existente. | En caso que se requiera modificar contenido existente. | Editar el reglamento. | Se permitirá modificar el contenido junto con la justificación del cambio. |
| | | | | 3 | Eliminación de contenido existente. | En caso que se requiera eliminar contenido existente. | Editar el reglamento. | Se permitirá eliminar el contenido junto con la justificación del cambio. |
| | | | | | | | | |
| **LMS-0008** | Como usuario administrador. | Necesito eliminar un reglamento existente. | Con la finalidad de eliminar reglamentos del CMS. | 1 | Eliminación de un reglamento. | En caso de que exista el reglamento. | Eliminar contenido. | Se eliminará el reglamento. |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **LMS-0009** | Como usuario administrador. | Necesito tener un *endpoint*. | Con la finalidad de realizar consultas SPARQL. | 1 | *Endpoint* para SPARQL. | En caso que exista un *endpoint* para consultas SPARQL. | Cuando el *endpoint* esté publicado. | Se permitirá realizar consultas SPARQL. |
| | | | | | | | | |
| **LMS-0010** | Como usuario administrador. | Necesito mapear el contenido con etiquetas RDF. | Con la finalidad de dotar al sitio Web de contenido semántico. | 1 | Agregar etiquetas RDF al contenido. | En caso de crear tipos de contenido para el manejo de reglamentaciones. | Mapear los tipos de contenido con etiquetas RDF. | Se permitirá mapear los tipos de contenido con diferentes etiquetas RDF. |
| | | | | | | | | |
| **LMS-0011** | Como usuario administrador. | Necesito contar con documentación. | Con la finalidad de poder usar fácilmente el Sistema de Manejo de Reglamentaciones. | 1 | Manual de instalación y configuración. | En caso de instalar y configurar el CMS. | Abrir el manual de instalación y configuración. | Se mostrará la documentación necesaria para la instalación y configuración. |
| | | | | | | | | |
| **LMS-0012** | Como usuario final y administrador. | Necesito contar con documentación. | Con la finalidad de poder usar fácilmente el Sistema de Manejo de Reglamentaciones. | 1 | Manual de usuario. | En caso de querer revisar la documentación del proyecto. | Abrir el manual de usuario. | Se mostrará la documentación necesaria para el uso del sitio Web. |

## 2.4. Historias Técnicas de Usuario

Las historias de usuario deben ser divididas en tareas pequeñas que permitan la ejecución de las mismas, donde se define la prioridad para el negocio, el riesgo en su desarrollo, la iteración asignada y los responsables de su desarrollo.

Las historias de usuario técnicas [20] propuestas para este proyecto se encuentran a continuación:

**Definir arquitectura**

En esta historia técnica nos vamos a enfocar en la arquitectura necesaria para la implementación adecuada del CMS, así como en los complementos o módulos que son necesarios para el trabajo con RDF, lo cual se encuentra detallado en la tabla 6.

*Tabla 6: Historia técnica de usuario – Definir arquitectura*

<table>
<tr><th colspan="3">Historia Técnica de Usuario</th></tr>
<tr><td>Número: 1</td><td colspan="2">Usuario: Usuario Administrador.</td></tr>
<tr><td colspan="3">Nombre historia: Definir arquitectura.</td></tr>
<tr><td>Prioridad en negocio: Alta.</td><td colspan="2">Riesgo en desarrollo: Alto.</td></tr>
<tr><td>Iteración asignada: 1</td><td colspan="2">Responsables: Felipe Borja, Andrés Cusme.</td></tr>
<tr><td colspan="3">Descripción:<br><br>El usuario administrador hace uso de un sitio Web implementado con un CMS que le permite ampliar su funcionalidad a través de módulos o complementos.<br><br>Descripción técnica:<br><br>1. Definir base de datos.<br>2. Definir servidor de aplicaciones.<br>3. Definir CMS.<br>4. Definir módulos o complementos.</td></tr>
<tr><td colspan="3">Observaciones: El diseño se establecerá de acuerdo a un CMS que sea compatible con RDF.</td></tr>
</table>

**Implementar arquitectura**

Una vez que la arquitectura ha sido definida es necesaria su implementación, tomando en cuenta los prerrequisitos que existan al momento de instalar cada uno de los componentes necesarios para optimizar al CMS, es decir, su base de datos, servidor de aplicaciones y módulos externos, lo cual se encuentra detallado en la tabla 7.

*Tabla 7: Historia técnica de usuario – Implementar arquitectura*

<table>
<tr><th colspan="3">Historia Técnica de Usuario</th></tr>
<tr><td>Número: 2</td><td colspan="2">Usuario: Usuario Administrador.</td></tr>
<tr><td colspan="3">Nombre historia: Implementar arquitectura.</td></tr>
<tr><td>Prioridad en negocio: Alta.</td><td colspan="2">Riesgo en desarrollo: Alto.</td></tr>
<tr><td>Iteración asignada: 1</td><td colspan="2">Responsables: Felipe Borja, Andrés Cusme.</td></tr>
<tr><td colspan="3">Descripción:<br><br>El usuario administrador hace uso de un sitio Web implementado con el CMS Drupal 7, ampliando su funcionalidad a través de módulos o complementos.<br><br>Descripción técnica:<br><br>1. Instalar base de datos MySQL.<br>2. Instalar servidor de aplicaciones Apache.<br>3. Instalar PHP 5.6.<br>4. Instalar Drupal 7.<br>5. Instalar módulos o complementos.</td></tr>
<tr><td colspan="3">Observaciones: La instalación se realiza sobre un ambiente Linux.</td></tr>
</table>

**Configurar módulos**

Esta historia técnica va enfocada a la configuración de cada uno de los módulos instalados, debido a que dichos módulos deben ser instalados en un cierto orden, además de tener que seguir pasos propios de configuración por cada uno de ellos, lo cual se encuentra detallado en la tabla 8.

*Tabla 8: Historia técnica de usuario – Configurar módulos*

| **Historia Técnica de Usuario** | |
|---|---|
| **Número:** 3 | **Usuario:** Usuario Administrador. |
| **Nombre historia:** Configurar módulos. | |
| **Prioridad en negocio:** Alta. | **Riesgo en desarrollo:** Alto. |
| **Iteración asignada:** 2 | **Responsables:** Felipe Borja, Andrés Cusme**.** |
| **Descripción:**<br><br>El usuario administrador hace uso de un sitio Web implementado con el CMS Drupal 7, ampliando su funcionalidad a través de módulos o complementos. Cada uno de estos módulos tiene una configuración específica.<br><br>**Descripción técnica:**<br><br>1. Configurar módulo: Arc2 store.<br>2. Configurar módulo: CKEditor - WYSIWYG HTML editor.<br>3. Configurar módulo: Conditional fields.<br>4. Configurar módulo: Chaos tool suite.<br>5. Configurar módulo: Date.<br>6. Configurar módulo: Devel.<br>7. Configurar módulo: Diff.<br>8. Configurar módulo: Diff different.<br>9. Configurar módulo: Entity reference.<br>10. Configurar módulo: Entity Api.<br>11. Configurar módulo: Field as block.<br>12. Configurar módulo: Inline entity form.<br>13. Configurar módulo: Libraries Api.<br>14. Configurar módulo: Pathauto.<br>15. Configurar módulo: Printer, email and PDF versions.<br>16. Configurar módulo: RDF indexer.<br>17. Configurar módulo: RDF Extensions.<br>18. Configurar módulo: SPARQL.<br>19. Configurar módulo: Token.<br>20. Configurar módulo: Transliteration.<br>21. Configurar módulo: Views.<br>22. Configurar módulo: Context. | |
| **Observaciones:** Los módulos usados están escritos en PHP y son parte de las colaboraciones realizadas al proyecto Drupal. | |

**Diseñar e implementar tipos de contenido**

En esta historia se define la estructura que va a manejar el sitio Web, de acuerdo a la reglamentación que se desea implementar, para lo cual es necesaria la creación de diferentes tipos de contenido dentro del CMS que nos permita en lo posterior ingresar información acorde a cada tipo de contenido creado, lo cual se encuentra detallado en la tabla 9.

*Tabla 9: Historia técnica de usuario – Diseñar e implementar tipo de contenido*

<table>
<tr><th colspan="3">Historia Técnica de Usuario</th></tr>
<tr><td>Número: 4</td><td colspan="2">Usuario: Usuario Administrador.</td></tr>
<tr><td colspan="3">Nombre historia: Diseñar e implementar tipos de contenido.</td></tr>
<tr><td>Prioridad en negocio: Alta.</td><td colspan="2">Riesgo en desarrollo: Alto.</td></tr>
<tr><td>Iteración asignada: 3</td><td colspan="2">Responsables: Felipe Borja, Andrés Cusme.</td></tr>
<tr><td colspan="3">Descripción:<br><br>El usuario administrador crea tipos de contenido acorde a la estructura legislativa de un reglamento.<br><br>Descripción técnica:<br><br>1. Diseñar estructura del reglamento.<br>2. Crear tipo de contenido en base a la estructura del reglamento.<br>3. Configurar el tipo de contenido en base a la estructura del reglamento.</td></tr>
<tr><td colspan="3">Observaciones: El diseño se realiza en base los reglamentos de la EPN.</td></tr>
</table>

**Diseñar e implementar Ontología Legislativa**

Esta historia abarca el estudio, diseño e implementación de una ontología legislativa, que se adapte a las necesidades del proyecto para el ingreso de información y etiquetado de las secciones acorde al tipo de contenido creado; esta ontología debe ser publicada para que pueda ser usada en lo posterior por el CMS, lo cual se encuentra detallado en la tabla 10.

*Tabla 10: Historia técnica de usuario – Diseñar e implementar ontología legislativa*

<table>
<tr><th colspan="3">Historia Técnica de Usuario</th></tr>
<tr><td>Número: 5</td><td colspan="2">Usuario: Usuario Administrador.</td></tr>
<tr><td colspan="3">Nombre historia: Diseñar e implementar ontología legislativa.</td></tr>
<tr><td colspan="2">Prioridad en negocio: Media.</td><td>Riesgo en desarrollo: Media.</td></tr>
<tr><td colspan="2">Iteración asignada: 3</td><td>Responsables: Felipe Borja, Andrés Cusme.</td></tr>
<tr><td colspan="3">Descripción:<br><br>El usuario administrador añade la ontología desarrollada al CMS Drupal 7, para ser mapeado a los tipos de contenido creados.<br><br>Descripción técnica:<br><br>1. Diseñar ontología legislativa.<br>2. Implementar ontología legislativa.<br>3. Publicar ontología legislativa.<br>4. Usar la ontología legislativa en el CMS Drupal 7.</td></tr>
<tr><td colspan="3">Observaciones: El diseño se realiza en base a la estructura de los reglamentos de la EPN.</td></tr>
</table>

**Crear Base de Datos RDF**

La presente historia técnica tiene por objetivo la creación de una base de datos dentro de MySql, que permita el almacenamiento de la información en forma de tripletas, que permitan en lo posterior realizar consultas SPARQL sobre la información almacenada en la misma, lo cual se encuentra detallado en la tabla 11.

*Tabla 11: Historia técnica de usuario – Crear base de datos RDF*

<table>
<tr><th colspan="3">Historia Técnica de Usuario</th></tr>
<tr><td>Número: 6</td><td colspan="2">Usuario: Usuario Administrador.</td></tr>
<tr><td colspan="3">Nombre historia: Crear Base de Datos RDF.</td></tr>
<tr><td colspan="2">Prioridad en negocio: Alta.</td><td>Riesgo en desarrollo: Alto.</td></tr>
<tr><td colspan="2">Iteración asignada: 4</td><td>Responsables: Felipe Borja, Andrés Cusme.</td></tr>
<tr><td colspan="3">Descripción:<br><br>El usuario administrador añade reglamentos al CMS Drupal 7, que serán almacenados en una base de datos RDF.<br><br>Descripción técnica:<br><br>1. Crear la base de datos RDF.<br>2. Configurar la base de datos e indexar el sitio Web.</td></tr>
<tr><td colspan="3">Observaciones: La creación de la base de datos RDF se realizará sobre MySQL.</td></tr>
</table>

**Mapear tipos de contenido**

En esta historia se va a realizar la relación o mapeo entre cada uno de los tipos de contenido creados con las etiquetas RDF propias de la ontología, que van a permitir dar semántica al sitio Web, lo cual se encuentra detallado en la tabla 12.

*Tabla 12: Historia técnica de usuario – Mapear tipos de contenido*

| **Historia Técnica de Usuario** | |
|---|---|
| **Número:** 7 | **Usuario:** Usuario Administrador. |
| **Nombre historia:** Mapear tipos de contenido. | |
| **Prioridad en negocio:** Media. | **Riesgo en desarrollo:** Bajo. |
| **Iteración asignada:** 4 | **Responsables:** Felipe Borja, Andrés Cusme**.** |
| **Descripción:**<br><br>El usuario administrador añade tipos de contenido al CMS Drupal 7, que necesitan ser mapeados con etiquetas RDF.<br><br>**Descripción técnica:**<br><br>1. Mapear tipos de contenido. | |
| **Observaciones:** El mapeo se realizará sobre los tipos de contenido creados anteriormente. | |

**Ingresar contenido**

Esta historia técnica está enfocada a darle uso al CMS, en el sentido de ingresar información dentro de cada uno de los tipos de contenido creados y mapeados con las etiquetas RDF, con el fin de tener datos con los cuales validar posteriormente la funcionalidad completa del sitio Web, lo cual se encuentra detallado en la tabla 13.

*Tabla 13: Historia técnica de usuario – Ingresar contenido*

<table>
<tr><th colspan="3">Historia Técnica de Usuario</th></tr>
<tr><td>Número: 8</td><td colspan="2">Usuario: Usuario Administrador.</td></tr>
<tr><td colspan="3">Nombre historia: Ingresar contenido.</td></tr>
<tr><td colspan="2">Prioridad en negocio: Media.</td><td>Riesgo en desarrollo: Bajo.</td></tr>
<tr><td colspan="2">Iteración asignada: 4</td><td>Responsables: Felipe Borja, Andrés Cusme.</td></tr>
<tr><td colspan="3">Descripción:<br><br>El usuario administrador añade información a cada uno de los tipos de contenido que forman parte de la estructura del reglamento al CMS Drupal 7.<br><br>Descripción técnica:<br><br>1. Agregar reglamentos al CMS Drupal 7.</td></tr>
<tr><td colspan="3">Observaciones: El reglamento referencial es el “Reglamento General de Elecciones” [21] de la EPN.</td></tr>
</table>

**Publicar sitio Web como Endpoint**

Esta historia técnica tiene el propósito de realizar una verificación de la funcionalidad del e*ndpoint* para en lo posterior poder hacer consultas SPARQL, esto debido a que la funcionalidad por defecto en Windows como en Linux no es completamente igual, para lo cual se tendrá que hacer los ajustes necesarios en la configuración del CMS, lo cual se encuentra detallado en la tabla 14.

*Tabla 14: Historia técnica de usuario – Verificar endpoint de Sitio Web*

| **Historia Técnica de Usuario** | |
|---|---|
| **Número:** 9 | **Usuario:** Usuario Administrador. |
| **Nombre historia:** Verificar *endpoint* de sitio Web. | |
| **Prioridad en negocio:** Media. | **Riesgo en desarrollo:** Bajo. |
| **Iteración asignada:** 5 | **Responsables:** Felipe Borja, Andrés Cusme**.** |
| **Descripción:**<br><br>El usuario administrador haciendo uso del CMS debe verificar que la funcionalidad del *endpPoint* del CMS esté disponible para poder consultar la información que se encuentra en la base de datos, de tal modo que al realizar las consultas SPARQL se retorne la información adecuada.<br><br>**Descripción técnica:**<br><br>1. Verificar disponibilidad de *Endpoint* SPARQL del CMS. | |
| **Observaciones:** Es importante tener en cuenta que el módulo SPARQL debe proveer el *endpoint* del sitio Web. | |

**Documentación acerca de la instalación, configuración y uso del sistema Web**

Esta historia técnica va enfocada al diseño e implementación de pruebas a ser realizadas sobre el sitio para verificar su funcionamiento y desempeño, y en caso de ser necesario corregir algún inconveniente que se presente, lo cual se encuentra detallado en la tabla 15.

*Tabla 15: Historia técnica de usuario – Diseñar e implementar pruebas*

| **Historia Técnica de Usuario** | |
|---|---|
| **Número:** 10 | **Usuario:** Usuario Final. |
| **Nombre historia:** Diseñar e implementar pruebas. | |
| **Prioridad en negocio:** Alta. | **Riesgo en desarrollo:** Bajo. |
| **Iteración asignada:** 5 | **Responsables:** Felipe Borja, Andrés Cusme**.** |
| **Descripción:**<br><br>El usuario final hace uso del sitio Web creado con el CMS Drupal 7 para verificar la funcionalidad de manejo de revisiones históricas de los reglamentos de la EPN.<br><br>**Descripción técnica:**<br><br>1. Diseño de pruebas.<br>2. Ejecución de pruebas. | |
| **Observaciones:** Las pruebas se realizarán sobre reglamentos que hayan sido modificados. | |

**Diseñar e implementar consultas SPARQL**

En esta historia técnica vamos a diseñar posibles consultas SPARQL e implementarlas con el fin de poder acceder a la información que brinda nuestro sitio Web desde un punto externo que debe acceder al *endpoint* que está provisto en el sitio Web, lo cual se encuentra detallado en la tabla 16.

*Tabla 16: Historia técnica de usuario – Diseñar e implementar consultas SPARQL*

| **Historia Técnica de Usuario** | |
|---|---|
| **Número:** 11 | **Usuario:** Usuario Final. |
| **Nombre historia:** Diseñar e implementar consultas SPARQL. | |
| **Prioridad en negocio:** Media. | **Riesgo en desarrollo:** Bajo. |
| **Iteración asignada:** 6 | **Responsables:** Felipe Borja, Andrés Cusme**.** |
| **Descripción:**<br><br>El usuario final hace uso de las consultas SPARQL para poder acceder a la información almacenada en el sitio Web creado con el CMS Drupal 7.<br><br>**Descripción técnica:**<br><br>1. Diseño de consultas SPARQL.<br>2. Ejecución de consultas SPARQL. | |
| **Observaciones:** Las consultas serán realizadas con lenguaje SPARQL y en base a la estructura del reglamento. | |

**Documentar instalación, configuración y uso del sitio Web**

En esta historia técnica vamos a documentar la instalación, configuración y uso del sitio Web lo cual se encuentra detallado en la tabla 17.

*Tabla 17: Historia técnica de usuario – Documentar instalación, configuración y uso del sitio Web*

| **Historia Técnica de Usuario** | |
|---|---|
| **Número:** 12 | **Usuario:** Usuario Final. |
| **Nombre historia:** Documentar instalación, configuración y uso del sitio Web. | |
| **Prioridad en negocio:** Media. | **Riesgo en desarrollo:** Bajo. |
| **Iteración asignada:** 6 | **Responsables:** Felipe Borja, Andrés Cusme**.** |
| **Descripción:**<br><br>El usuario final debe tener documentación acerca de la instalación, configuración y uso del sitio Web creado con el CMS Drupal 7.<br><br>**Descripción técnica:**<br><br>1. Realizar una guía de instalación y configuración del sitio Web.<br>2. Realizar un manual de uso del sitio Web. | |
| **Observaciones:** El documento de instalación y configuración se realizará en base a la instalación realizada en un servidor Linux local. | |

## 2.5. *Product Backlog*

El *Product Backlog* es una lista priorizada de las historias de usuario que representa la visión del cliente respecto al proyecto; esta lista contiene las historias de usuario con un identificador de historia, su enunciado, el posible riesgo, su prioridad y la iteración en la cual será desarrollada, lo cual se encuentra detallado en la tabla 18.

*Tabla 18: Product Backlog*

| N° | ID de la Historia | Enunciado de la Historia | Riesgo | Prioridad | Iteración |
|---|---|---|---|---|---|
| **1** | HU01 | Definir arquitectura. | Alto | Alta | 1 |
| **2** | HU02 | Implementar arquitectura. | Alto | Alta | 1 |
| **3** | HU03 | Configurar módulos. | Alto | Alta | 2 |
| **4** | HU04 | Diseñar e implementar tipos de contenido. | Alto | Alta | 3 |
| **5** | HU05 | Diseñar e implementar ontología legislativa. | Medio | Media | 3 |
| **6** | HU06 | Crear base de datos RDF. | Alto | Alta | 4 |
| **7** | HU07 | Mapear tipos de contenido. | Bajo | Media | 4 |
| **8** | HU08 | Ingresar contenido. | Bajo | Media | 4 |
| **9** | HU09 | Verificar *endpoint* del sitio Web. | Bajo | Media | 5 |
| **10** | HU10 | Diseñar e implementar pruebas. | Bajo | Alta | 5 |
| **11** | HU11 | Diseñar e implementar consultas SPARQL. | Bajo | Alta | 6 |
| **12** | HU12 | Documentar instalación, configuración y uso del sitio Web. | Bajo | Media | 6 |

## 2.6. *Sprints*

Los *sprint* han sido definidos en base al tiempo que se tiene para el desarrollo y entrega del proyecto, haciendo la relación de un sprint correspondiente a dos semanas; durante cada *sprint* se llevarán a cabo las ceremonias propias de *Scrum* manteniendo actualizado el *Product Release Plan* elaborado para el desarrollo del proyecto.

Para el arranque del proyecto se definió como fecha de inicio de 03 de julio de 2017, y se estima finalizar el 22 de septiembre de 2017, con un trabajo de lunes a viernes sin tomar en cuenta los fines de semana considerando un tiempo destinado para el desarrollo de cuatro horas diarias se estima finalizar en un plazo de 60 días.

### 2.6.1. Primer *Sprint*

Este sprint está conformado por las dos primeras tareas definidas en el *Product Backlog* inicial. Inicia el 03 de julio y se estima finalizar hasta el 16 de julio, con una duración aproximada de 10 días; además las reuniones de pies se las realiza diariamente a las 6 pm con una duración máxima de 15 minutos, en base a la disponibilidad del equipo de trabajo.
Este *sprint* tiene como objetivo el realizar un análisis de la arquitectura que deberá componer el proyecto, así como su implementación, con la instalación de los componentes necesarios para su desarrollo.
Para poder llevar un adecuado control de las tareas que son incluidas en el *sprint* así como su cumplimiento se hace uso de una plantilla [20] que permite la inclusión del tipo de tarea, su estado, el responsable de la tarea y esfuerzo de cada tarea de acuerdo a un calendario sujeto a las fechas planteadas para el cumplimiento de los objetivos del *sprint*, que se muestra en la figura 4.

| SPRINT | INICIO | DURACIÓN |
|---|---|---|
| 1 | 3-jul.-17 | 10 |

| | | | | | L | M | X | J | V | L | M | X | J | V |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | 3-jul. | 4-jul. | 5-jul. | 6-jul. | 7-jul. | 10-jul. | 11-jul. | 12-jul. | 13-jul. | 14-jul. |
| | | | | Tareas pendientes | 3 | 2 | 3 | 2 | 4 | 2 | 2 | 2 | 2 | 2 |
| | | | | Horas de trabajo pendientes | 36 | 32 | 28 | 24 | 21 | 17 | 14,5 | 10 | 5,5 | 1 |
| | PILA DEL SPRINT | | | | ESFUERZO | | | | | | | | | |
| Backlog | Tarea | Tipo | Estad | Responsal | | | | | | | | | | |
| | Definir base de datos | Análisis | Terminada | Equipo | 2 | | | | | | | | | |
| | Definir servidor de aplicaciones | Análisis | Terminada | Equipo | 1,5 | | | | | | | | | |
| | Definir CMS | Análisis | Terminada | Equipo | | 3,5 | 1,5 | | | | | | | |
| | Definir módulos o complementos | Análisis | Terminada | Equipo | | | 2 | 3,5 | | | | | | |
| | Instalar base de datos MySQL | Desarrollo | Terminada | Equipo | | | | | 1 | | | | | |
| | Instalar servidor de aplicaciones | Desarrollo | Terminada | Equipo | | | | | 1 | | | | | |
| | Instalar PHP 5.6 | Desarrollo | Terminada | Equipo | | | | | 0,5 | | | | | |
| | Instalar Drupal 7 | Desarrollo | Terminada | Equipo | | | | | | 3,5 | 2 | | | |
| | Instalar módulos o complementos | Desarrollo | Terminada | Equipo | | | | | | | | 4 | 4 | 4 |
| | Documentación | Desarrollo | Terminada | Equipo | 0,5 | 0,5 | 0,5 | 0,5 | 0,5 | 0,5 | 0,5 | 0,5 | 0,5 | 0,5 |

*Figura 4: Sprint Backlog – Primer Sprint*

### *Burn-down Chart*

La figura 5 muestra el Gráfico de esfuerzo vs Número de horas o *Burn-down Chart*, que se lo tomó al final del *sprint* con el fin de presentar cómo fue el desarrollo del primer *sprint*, pudiendo evidenciar que la curva descendente propia de este gráfico se cumple de una manera correcta, lo que nos indica que las estimaciones de los tiempos de las tareas a ser ejecutadas en el *sprint* fueron buenas y reales, ya que se las pudo concluir.

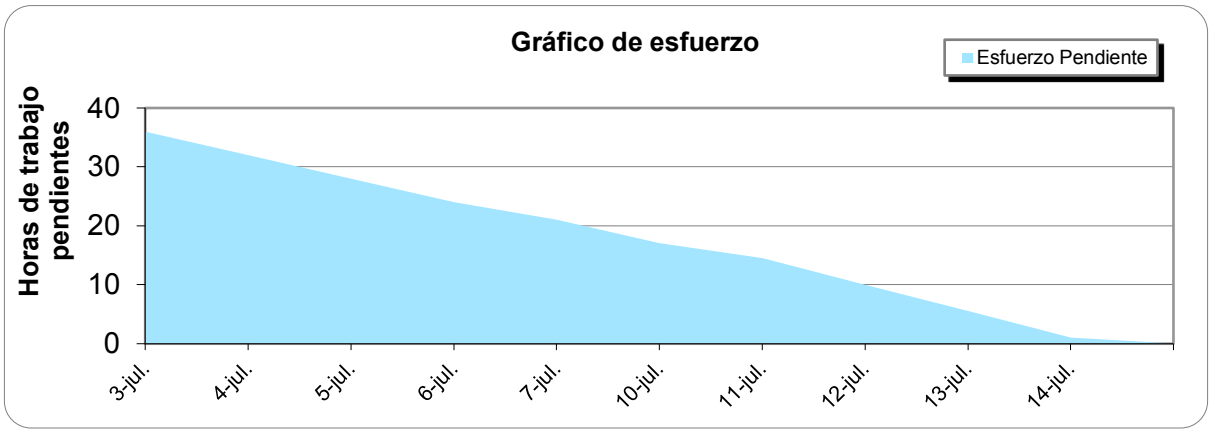


*Figura 5: Gráfico de esfuerzo vs Número de Horas – Primer Sprint*

### 2.6.2. Segundo *Sprint*

Este *sprint* está compuesto por una única historia de usuario que tiene por objetivo la configuración de los módulos que son necesarios dentro del CMS para el manejo adecuado de RDF y cumplir con el fin del proyecto. Este sprint se compone de aproximadamente 10 días de desarrollo, comenzando a partir de 17 de julio.

Para poder llevar un adecuado control de las tareas que son incluidas en el *sprint* así como su cumplimiento se hace uso de una plantilla [20] que permite la inclusión del tipo de tarea, su estado, el responsable de la tarea y esfuerzo de cada tarea de acuerdo a un calendario sujeto a las fechas planteadas para el cumplimiento de los objetivos del *sprint*.

Una característica especial del gráfico del detalle de las tareas para este *sprint*, es que se ha descompuesto la tarea general de configuración de módulos en varias subtareas, es decir, que se dedica un tiempo específico para la configuración de cada uno de los módulos que son necesarios para el desarrollo del proyecto, además de una tarea de documentación para presentar la característica de configuración de los módulos, ya que cada uno de ellos tiene su particularidad, lo cual se detalla en los anexos finales, lo cual se puede ver en la figura 6.

| SPRINT | INICIO | DURACIÓN |
|---|---|---|
| 1 | 17-jul.-17 | 10 |

| | L 17-jul. | M 18-jul. | X 19-jul. | J 20-jul. | V 21-jul. | L 24-jul. | M 25-jul. | X 26-jul. | J 27-jul. | V 28-jul. |
|---|---|---|---|---|---|---|---|---|---|---|
| Tareas pendientes | 4 | 4 | 4 | 4 | 4 | 4 | 1 | 1 | 1 | 1 |
| Horas de trabajo pendientes | 72 | 64 | 56 | 48 | 40 | 32 | 28 | 24 | 20 | 16 |

| PILA DEL SPRINT | | | | | ESFUERZO | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Backlog | Tarea | Tipo | Estado | Responsal | | | | | | | | | | |
| | Configurar Arc2 store | Desarrollo | Terminada | Andrés | 2 | | | | | | | | | |
| | Configurar CKEditor | Desarrollo | Terminada | Andrés | 2 | | | | | | | | | |
| | Configurar Conditional fields | Desarrollo | Terminada | Andrés | | 2 | | | | | | | | |
| | Configurar Chaos tool suite | Desarrollo | Terminada | Andrés | | 2 | | | | | | | | |
| | Configurar Date | Desarrollo | Terminada | Andrés | | | 2 | | | | | | | |
| | Configurar Devel | Desarrollo | Terminada | Andrés | | | 2 | | | | | | | |
| | Configurar Diff | Desarrollo | Terminada | Andrés | | | | 2 | | | | | | |
| | Configurar Diff different | Desarrollo | Terminada | Andrés | | | | 2 | | | | | | |
| | Configurar Entity reference | Desarrollo | Terminada | Andrés | | | | | 2 | | | | | |
| | Configurar Entity Api | Desarrollo | Terminada | Andrés | | | | | 2 | 2 | | | | |
| | Configurar Field as block | Desarrollo | Terminada | Andrés | | | | | | 2 | | | | |
| | Configurar Inline entity form | Desarrollo | Terminada | Felipe | 2 | | | | | | | | | |
| | Configurar Libraries Api | Desarrollo | Terminada | Felipe | 2 | | | | | | | | | |
| | Configurar Pathauto | Desarrollo | Terminada | Felipe | | 2 | | | | | | | | |
| | Configurar Printer | Desarrollo | Terminada | Felipe | | 2 | | | | | | | | |
| | Configurar RDF indexer | Desarrollo | Terminada | Felipe | | | 2 | | | | | | | |
| | Configurar RDF Extensions | Desarrollo | Terminada | Felipe | | | 2 | | | | | | | |
| | Configurar SPARQL | Desarrollo | Terminada | Felipe | | | | 2 | | | | | | |
| | Configurar Token | Desarrollo | Terminada | Felipe | | | | 2 | 2 | | | | | |
| | Configurar Transliteration | Desarrollo | Terminada | Felipe | | | | | 2 | | | | | |
| | Configurar Views | Desarrollo | Terminada | Felipe | | | | | | 2 | | | | |
| | Configurar Context | Desarrollo | Terminada | Felipe | | | | | | 2 | | | | |
| | Documentación | Desarrollo | Terminada | Equipo | | | | | | | 4 | 4 | 4 | 4 |

*Figura 6: Sprint Backlog –Segundo Sprint*

## *Burn-down Chart*

La figura 7 muestra el Gráfico de esfuerzo vs Número de horas o *Burn-down Chart*, que se lo tomó al final del segundo *sprint* con el fin de presentar cómo fue el desarrollo del mismo, pudiendo evidenciar que la curva descendente propia de este gráfico se cumple de una manera correcta, lo que nos indica que las estimaciones de los tiempos de las tareas individuales de configuración a ser ejecutadas en el *sprint* fueron bien estimadas, ya que se las pudo concluir.

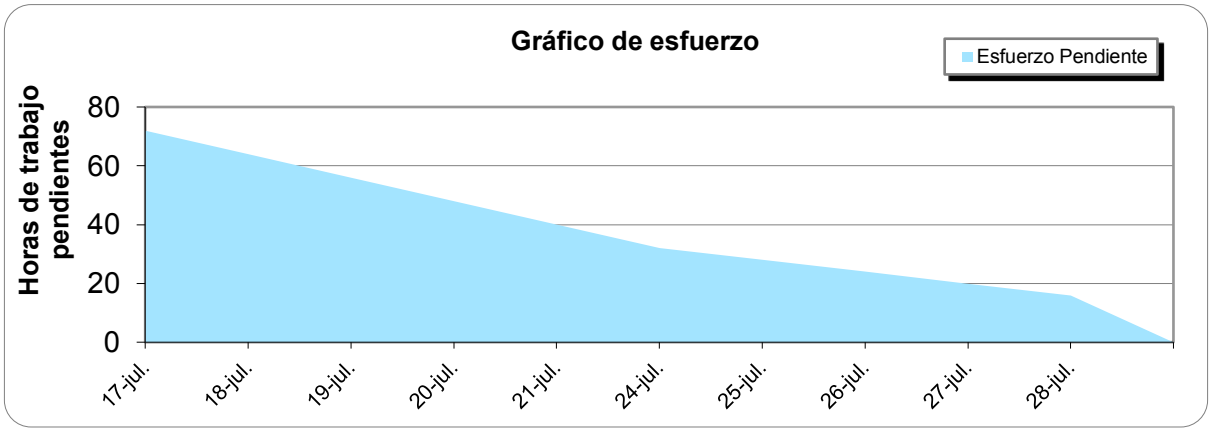


*Figura 7: Gráfico de esfuerzo vs Número de Horas – Segundo Sprint*

### 2.6.3. Tercer *Sprint*

Este *sprint* está compuesto por dos historias de usuarios que tienen por objetivo el diseño e implementación de tipos de contenido y una ontología Legislativa. Este *sprint* se compone de aproximadamente 10 días de desarrollo, a partir de 31 de julio. Para poder llevar un adecuado control de las tareas que son incluidas en el sprint así como su cumplimiento se hace uso de una plantilla [20] que permite la inclusión del tipo de tarea, su estado, el responsable de la tarea y esfuerzo de cada tarea de acuerdo a un calendario sujeto a las fechas planteadas para el cumplimiento de los objetivos del *sprint*, que se muestra en la figura 8.

| SPRINT | INICIO | DURACIÓN |
|---|---|---|
| 1 | 31-jul.-17 | 10 |

| | L | M | X | J | V | L | M | X | J | V |
|---|---|---|---|---|---|---|---|---|---|---|
| | 31-jul. | 1-ago. | 2-ago. | 3-ago. | 4-ago. | 7-ago. | 8-ago. | 9-ago. | 10-ago. | 11-ago. |
| Tareas pendientes | 2 | 2 | 2 | 1 | 1 | 1 | 2 | 2 | 2 | 2 |
| Horas de trabajo pendientes | 72 | 64 | 56 | 52 | 48 | 44 | 36 | 28 | 20 | 12 |

| PILA DEL SPRINT | | | | | ESFUERZO | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Backlog | Tarea | Tipo | Estado | Responsable | | | | | | | | | | |
| | Diseñar estructura del reglamento | Análisis | Terminada | Andrés | 4 | | | | | | | | | |
| | Crear tipo de contenido | Desarrollo | Terminada | Andrés | | 4 | | | | | | | | |
| | Configurar el tipo de contenido | Desarrollo | Terminada | Andrés | | | 4 | | | | | | | |
| | Diseñar Ontología Legislativa | Análisis | Terminada | Felipe | 4 | | | | | | | | | |
| | Implementar Ontología Legislativa | Desarrollo | Terminada | Felipe | | 4 | 4 | | | | | | | |
| | Publicar Ontología Legislativa | Desarrollo | Terminada | Felipe | | | | 4 | 4 | | | | | |
| | Usar Ontología Legislativa en CMS | Pruebas | Terminada | Equipo | | | | | | 4 | 4 | 4 | 4 | 4 |
| | Documentación | Desarrollo | Terminada | Equipo | | | | | | | 4 | 4 | 4 | 4 |

*Figura 8: Sprint Backlog – Tercer Sprint*

### *Burn-down Chart*

La figura 9 muestra el Gráfico de esfuerzo vs Número de horas o *Burn-down Chart*, que se lo tomó al final del tercer *sprint* con el fin de presentar cómo fue el desarrollo del mismo, pudiendo evidenciar que la curva descendente propia de este gráfico se cumple de una manera correcta, lo que nos indica que las estimaciones de los tiempos de las tareas individuales de configuración a ser ejecutadas en el *sprint* fueron bien estimadas.

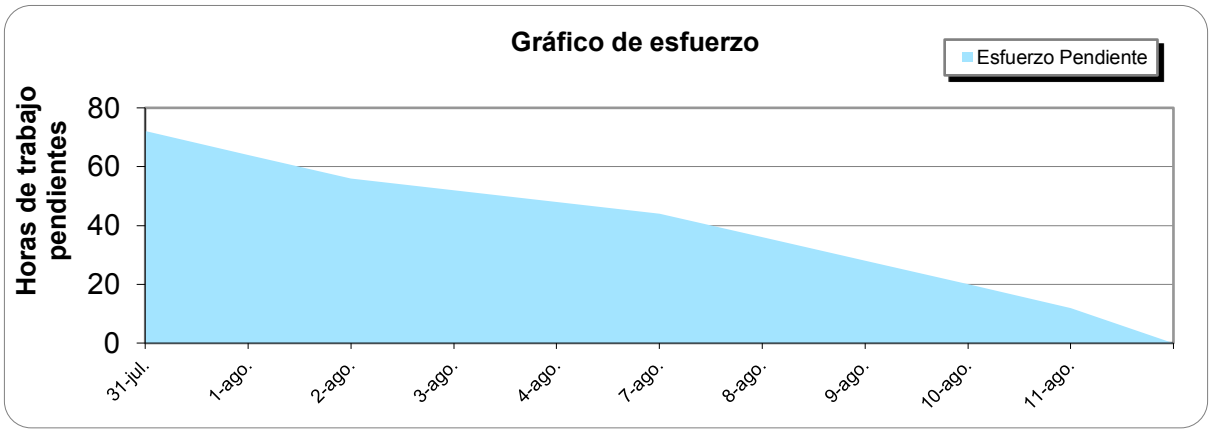


*Figura 9: Gráfico de esfuerzo vs Número de Horas – Tercer Sprint*

### 2.6.4. Cuarto *Sprint*

Este *sprint* se compone por tres historias de usuarios que tienen por objetivo la creación de una base de datos que permita trabajar con las etiquetas propias de RDF, el mapeo de los tipos de contenido con dichas etiquetas y para realizar una prueba del mapeo se introducirá contenido o información, es decir un reglamento. Este *sprint* se compone de aproximadamente 10 días de desarrollo, comenzando a partir del 14 de agosto. Para poder llevar un adecuado control de las tareas que son incluidas en el *sprint* así como su cumplimiento se hace uso de una plantilla [20] que permite la inclusión del tipo de tarea, su estado, el responsable de la tarea y esfuerzo de cada tarea, que se muestra en la figura 10.

| SPRINT | INICIO | DURACIÓN |
|---|---|---|
| 1 | 14-ago.-17 | 10 |

| | L 14-ago. | M 15-ago. | X 16-ago. | J 17-ago. | V 18-ago. | L 21-ago. | M 22-ago. | X 23-ago. | J 24-ago. | V 25-ago. |
|---|---|---|---|---|---|---|---|---|---|---|
| Tareas pendientes | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 |
| Horas de trabajo pendientes | 72 | 68 | 60 | 52 | 44 | 36 | 28 | 20 | 12 | 4 |

| PILA DEL SPRINT | | | | | ESFUERZO | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Backlog | Tarea | Tipo | Estado | Responsable | | | | | | | | | | |
| | Crear Base de Datos RDF | Desarrollo | Terminada | Andrés | 4 | | | | | | | | | |
| | Configurar acceso a Base de Datos | Desarrollo | Terminada | Felipe | | 4 | | | | | | | | |
| | Indexar el sitio Web | Desarrollo | Terminada | Felipe | 4 | | | | | | | | | |
| | Mapear tipos de contenido | Desarrollo | Terminada | Equipo | | | 4 | 4 | 4 | 4 | 4 | | | |
| | Ingresar contenido | Prueba | Terminada | Equipo | | | 4 | 4 | 4 | 4 | 4 | 4 | 4 | |
| | Documentación | Desarrollo | Terminada | Equipo | | | | | | | | 4 | 4 | 8 |

*Figura 10: Sprint Backlog – Cuarto Sprint*

### *Burn-down Chart*

La figura 11 muestra el Gráfico de esfuerzo vs Número de horas o *Burn-down Chart*, que se lo tomó al final del cuarto *sprint* con el fin de presentar cómo fue el desarrollo del mismo, pudiendo evidenciar que la curva descendente propia de este gráfico se cumple de una manera correcta, lo que nos indica que las estimaciones de los tiempos de las tareas individuales de configuración a ser ejecutadas en el *sprint* fueron bien estimadas.

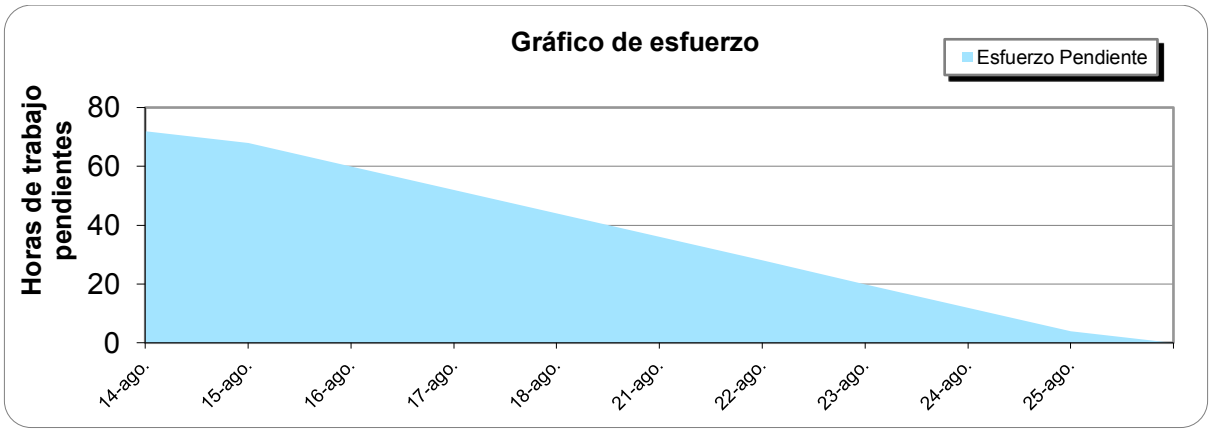


*Figura 11: Gráfico de esfuerzo vs Número de Horas – Cuarto Sprint*

### 2.6.5. Quinto *Sprint*

Este *sprint* se compone por dos historias de usuarios que tienen por objetivo la verificación del *endpoint* del sitio Web que permita en lo posterior, realizar consultas SPARQL sobre el mismo, además del diseño e implementación de pruebas de la funcionalidad del sitio Web. Este *sprint* se compone de aproximadamente 10 días de desarrollo, comenzando a partir del 28 de agosto. Para poder llevar un adecuado control de las tareas que son incluidas en el sprint así como su cumplimiento se hace uso de una plantilla [20] que permite la inclusión del tipo de tarea, su estado, el responsable de la tarea y esfuerzo de cada tarea de acuerdo a un calendario sujeto a las fechas planteadas para el cumplimiento de los objetivos del *sprint*, que se muestran en la figura12.

| SPRINT | INICIO | DURACIÓN |
|---|---|---|
| 1 | 28-ago.-17 | 10 |

| | L 28-ago. | M 29-ago. | X 30-ago. | J 31-ago. | V 1-sep. | L 4-sep. | M 5-sep. | X 6-sep. | J 7-sep. | V 8-sep. |
|---|---|---|---|---|---|---|---|---|---|---|
| Tareas pendientes | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Horas de trabajo pendientes | 72 | 64 | 56 | 48 | 40 | 32 | 24 | 16 | 8 | |

| PILA DEL SPRINT | | | | | ESFUERZO | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Backlog | Tarea | Tipo | Estado | Responsable | | | | | | | | | | |
| | Agregar reglamentos al CMS | Desarrollo | Terminada | Equipo | 4 | 4 | 4 | | | | | | | |
| | Diseño de pruebas | Análisis | Terminada | Felipe | 4 | 4 | 4 | | | | | | | |
| | Ejecución de pruebas | Desarrollo | Terminada | Andrés | | | | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| | Documentación | Desarrollo | Terminada | Equipo | | | | 4 | 4 | 4 | 4 | 4 | 4 | 4 |

*Figura 12: Sprint Backlog – Quinto Sprint*

### *Burn-down Chart*

La figura 13 muestra el Gráfico de esfuerzo vs Número de horas o *Burn-down Chart*, que se lo tomó al final del quinto *sprint* con el fin de presentar cómo fue el desarrollo del mismo, pudiendo evidenciar que la curva descendente propia de este gráfico se cumple de una manera correcta, lo que nos indica que las estimaciones de los tiempos de las tareas individuales de configuración a ser ejecutadas en el *sprint* fueron bien estimadas.

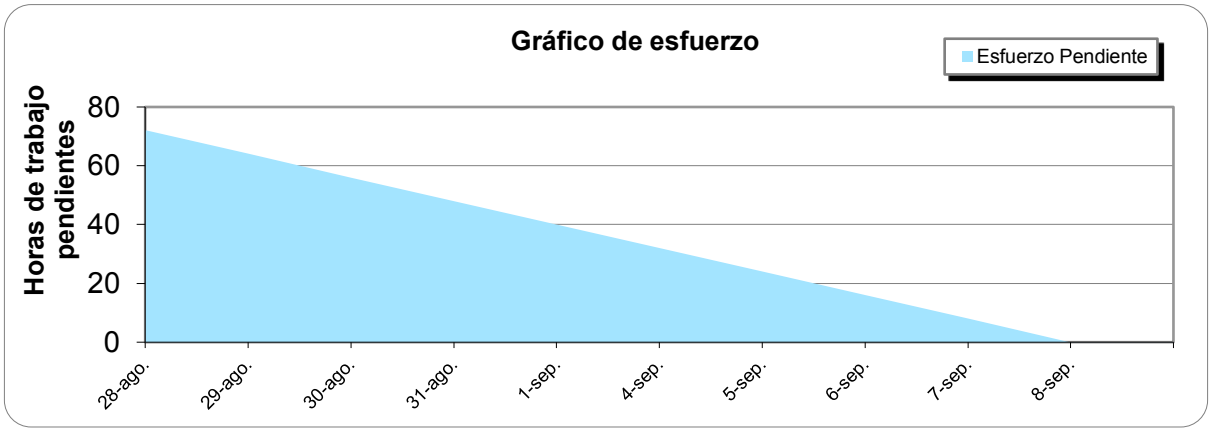


*Figura 13: Gráfico de esfuerzo vs Número de Horas – Quinto Sprint*

### 2.6.6. Sexto *Sprint*

Este *sprint* se compone por una única historia de usuario que tiene por objetivo el diseño y la implementación de consultas SPARQL que puedan ser realizadas sobre el sitio, con el fin de demostrar el contenido semántico que maneja el sitio Web, al poder ser consultado desde un *endpoint*. Este *sprint* se compone de aproximadamente 10 días de desarrollo, comenzando a partir del 11 de septiembre. Para poder llevar un adecuado control de las tareas que son incluidas en el sprint así como su cumplimiento se hace uso de una plantilla [20] que permite la inclusión del tipo de tarea, su estado, el responsable de la tarea y esfuerzo de cada tarea de acuerdo a un calendario sujeto a las fechas planteadas para el cumplimiento de los objetivos del *sprint*, como se muestra en la figura 14.

| SPRINT | INICIO | DURACIÓN |
|---|---|---|
| 1 | 11-sep.-17 | 10 |

| | L 11-sep. | M 12-sep. | X 13-sep. | J 14-sep. | V 15-sep. | L 18-sep. | M 19-sep. | X 20-sep. | J 21-sep. | V 22-sep. |
|---|---|---|---|---|---|---|---|---|---|---|
| Tareas pendientes | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Horas de trabajo pendientes | 72 | 64 | 56 | 48 | 40 | 32 | 24 | 16 | 8 | |

| PILA DEL SPRINT | | | | | ESFUERZO | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Backlog | Tarea | Tipo | Estado | Responsable | | | | | | | | | | |
| | Diseño de consultas SPARQL | Análisis | Terminada | Equipo | 4 | 4 | 4 | 4 | 4 | | | | | |
| | Revisión de consultas SPARQL | Análisis | Terminada | Equipo | 4 | 4 | 4 | 4 | 4 | | | | | |
| | Ejecución de consultas SPARQL | Desarrollo | Terminada | Equipo | | | | | | 4 | 4 | 4 | 4 | 4 |
| | Documentación | Desarrollo | Terminada | Equipo | | | | | | 4 | 4 | 4 | 4 | 4 |

*Figura 14: Sprint Backlog – Sexto Sprint*

### *Burn-down Chart*

La figura 15 muestra el Gráfico de esfuerzo vs Número de horas o *Burn-down Chart*, que se lo tomó al final del sexto *sprint* con el fin de presentar cómo fue el desarrollo del mismo, pudiendo evidenciar que la curva descendente propia de este gráfico se cumple de una manera correcta, lo que nos indica que las estimaciones de los tiempos de las tareas individuales de configuración a ser ejecutadas en el *sprint* fueron bien estimadas.

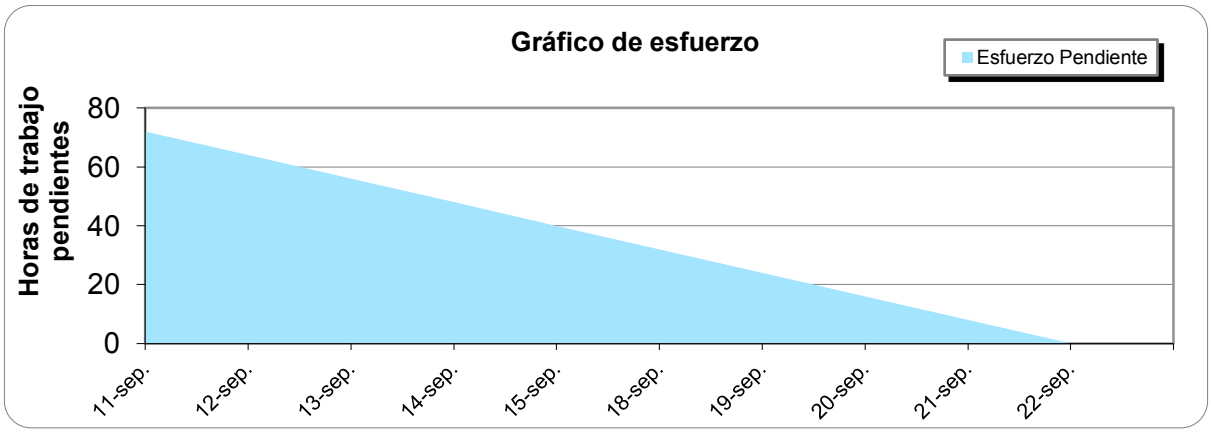


*Figura 15: Gráfico de esfuerzo vs Número de Horas – Sexto Sprint*

# 3. RESULTADOS Y DISCUSIÓN

## 3.1. Resultados

### 3.1.1. Arquitectura del Sistema

La arquitectura usada en el desarrollo del proyecto se puede apreciar en la figura 16, en la que se ilustra el usuario y la interacción que realiza Drupal con sus componentes para poder presentar la información consultada por el usuario.

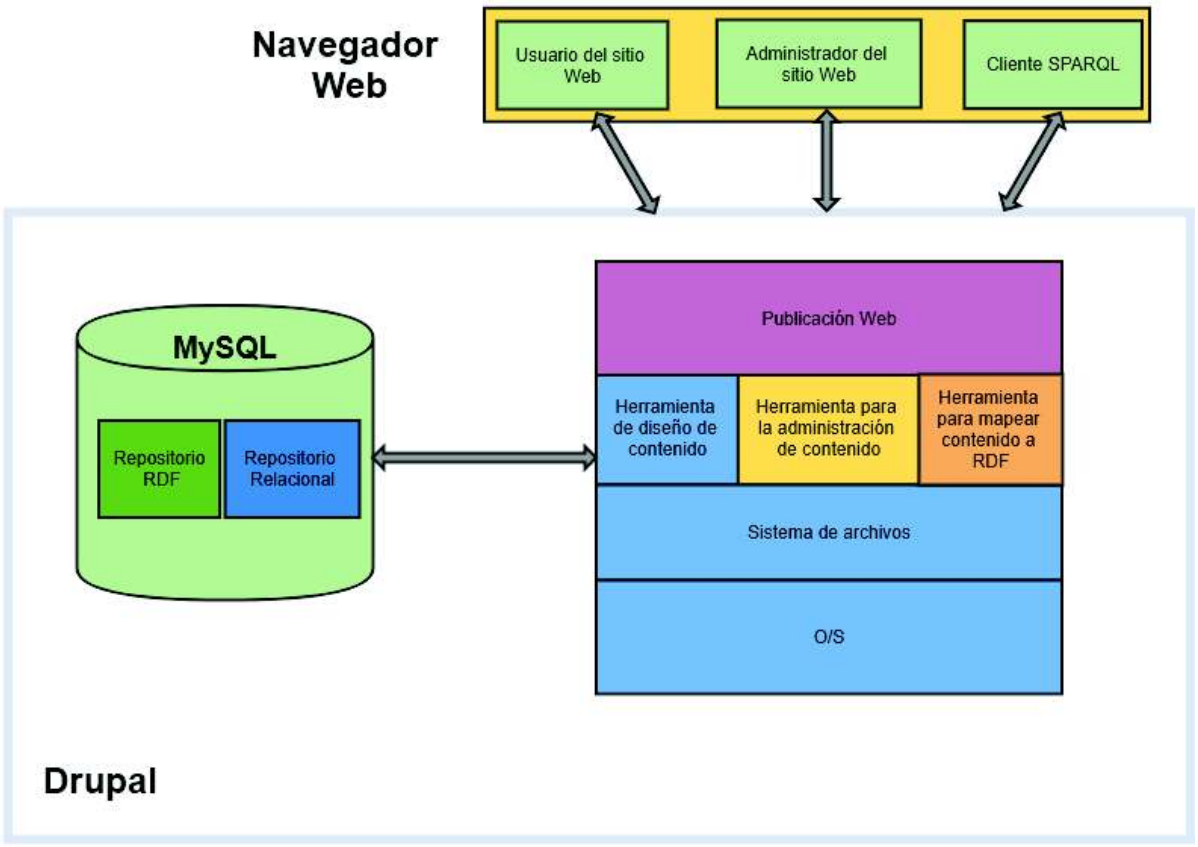


*Figura 16: Diagrama de la arquitectura del Sistema [22]*

### 3.1.2. Implementación Drupal 7

La implementación del CMS Drupal 7 se la realizó de acuerdo a los requerimientos mínimos necesarios para su funcionamiento correcto según consta en el sitio Web del CMS [2], los cuales se detallan en la tabla 19:

*Tabla 19: Especificación de Requerimientos para Drupal 7.*

| Tipo de Requerimiento | Especificación |
|---|---|
| Sistema Operativo | Ubuntu 16.04 |
| Versión del CMS | Drupal 7 |
| Servidor Web | Apache 2 |
| Base de Datos | MySql 5.7 |
| PHP | 5.6 |

Las ventajas principales que ofrece Drupal son la flexibilidad al momento de implementar sitios Web, ya que dentro de la comunidad que da soporte al proyecto, existen varios módulos, que son funcionalidades que se pueden añadir al sitio, que pueden ser implementados y usados de acuerdo a la necesidad del proyecto. La documentación de estos módulos se la puede encontrar en sitio Web oficial del proyecto, así como requisitos y tutoriales que dan la facilidad de poder aplicarlos de manera adecuada.

El *Core* de Drupal 7 incluye ciertos módulos por defecto que permiten su funcionamiento básico, dentro de los cuales también se encuentra un módulo para el manejo de RDF, lo que permitió trabajar directamente con este tipo de contenido sin la necesidad de usar un paso previo para la transformación inicial a XML, dicho módulo se denomina RDFa [2].

RDFa permite expresar datos estructurados en lenguajes de marcado, tal como XHTML y HTML5; la versión que implementa el *Core* es 1.0, además que no tiene incorporada una interfaz de usuario propia ya que su objetivo es mejorar el marcado HTML, o etiquetado de HTML, generado por el contenido del *Core* de Drupal.

Los módulos son los elementos que permiten añadir nuevas funcionalidades a Drupal para adaptarlo a las necesidades de cada sitio web, es decir que son *plugins* que amplían sus posibilidades y funcionalidades. Los módulos se clasifican en 2 grandes tipos: los *Core Modules* y los *Contributed Module*s. Los *Core modules* son los que tiene por defecto Drupal y son minoría, aunque seguramente los más necesarios. Los *Contributed modules* son los módulos desarrollados por la comunidad que da soporte a Drupal y que no vienen por defecto con Drupal. Para conseguir este tipo de módulos hay que revisarlos en la web oficial de Drupal, descargarlos e instalarlos [23].

Para la implementación adecuada del sitio Web para el Proyecto se hizo uso de varios módulos, cuya documentación de la implementación puede encontrarse en el documento

anexo “Guía de instalación y configuración Drupal 7” en la que se puede encontrar el listado de todos los módulos usados, así como el orden de instalación de los mismos y la particularidad dentro de la configuración que pudieron tener ciertos módulos. Dentro de los módulos que fueron usados se pudo determinar que para la implementación de Web semántica y el manejo de RDF al nivel necesitado por este proyecto, la información de los siguientes módulos es importante:

**RDF Extensions**

Este módulo permite agregar características adicionales al módulo RDF central propio del *Core* de Drupal, el cual incluye a su vez los siguientes módulos [2]:

- *RDFx:* amplía el soporte de RDF al proporcionar un API adicional y formatos de serialización como RDF/XML, NTriples, Turtle.
- *RDF UI:* permite al administrador del sitio especificar las etiquetas RDF a un tipo de contenido a través de un entorno gráfico.
- *Evoc:* es una interfaz de usuario que permite la importación de vocabularios que pueden ser usados por RDF UI.

**SPARQL**

SPARQL es un lenguaje de consulta de coincidencia de patrones con grafos RDF, con una sintaxis similar a SQL; además este lenguaje permite realizar consultas que abarcar múltiples fuentes de información con datos dispares, es decir de fuentes locales o remotas, que contienen datos heterogéneos o semiestructurados [2].

El módulo en sí incluye la funcionalidad central del API SPARQL que habilita el uso de consultas y el *endpoint* SPARQL.

**Arc2 store**

Este módulo tiene una base de datos RDF, que puede ser accedida a través de consultas SPARQL, donde cada *Store* de ARC2 puede ser configurado para exponerse como un *endpoint* SPARQL dentro de una URL definida. Este módulo brinda una interfaz de usuario para crear y administrar los *Store* de RDF que pueden ser exportados [2].

La característica principal de este módulo es que a diferencia de la extensión RDF propia del *Core* de Drupal que permite exportar una entidad individual como RDF y que no ofrece una función para recopilar los datos RDF en un solo lugar, este módulo si lo permite al tener todos los datos RDF disponibles en un único sitio que permite la consulta de grafos RDF; así mismo otra ventaja es que el módulo permite la configuración de tripletas y la indexación de datos RDF [2].

**RDF indexer**

Este módulo usa el API de búsqueda para el manejo y optimización del indexado de los datos dentro de la base de datos RDF; una vez que se tiene listo un *Store* de ARC2 para recibir datos RDF, es necesario la configuración de Drupal para que pueda enviar los datos a dicho *Store*. Para comenzar con la indexación del contenido se necesita que el índice y el servidor de indexados de RDF se encuentren instalados, lo cual se lo realizado con un calendarizador que permitirá que el índice RDF se encuentre actualizado cada cierto intervalo de tiempo, se recomienda que la frecuencia no sea mayor de 10 minutos [2].

Los módulos que fueron instalados en el CMS Drupal 7 y el orden en el que fueron implementados se encuentran detallados en la taba 20.

*Tabla 20: Lista de módulos instalados en CMS Drupal 7*

| Módulos Instalados |
|---|
| 1. Chaos tool suite (ctools) |
| 2. Entity Api |
| 3. Libraries Api |
| 4. Views |
| 5. Token |
| 6. Context |
| 7. CKEditor |
| 8. Conditional fields |
| 9. Date |
| 10. Devel |
| 11. Diff |
| 12. Diff different |
| 13. Entity reference |
| 14. Field as block |
| 15. Inline entity form |
| 16. Pathauto |
| 17. Printer, email and PDF versions |
| 18. Transliteration |
| 19. RDF Extensions |
| 20. SPARQL |
| 21. Arc2 store |
| 22. RDF indexer |

### 3.1.3. Implementación de tipos de contenido

Los tipos de contenido son estructuras de datos creadas para el manejo de la información en Drupal 7, cada una de ellas está compuesta por uno o más campos que permitirán el ingreso de la información por parte del usuario, es decir, se crea una plantilla con todos los campos necesarios para el ingreso de la información. Esto ayuda a que no se creen varios tipos de contenido iguales para poder almacenar varias veces la misma información, sino que definiéndola una única vez podremos ingresar el contenido que desee y las veces que desee.

Los tipos de contenido son almacenados en forma de nodos, por lo que la creación de cada uno de ellos implica la creación de un nodo nuevo. Un sitio Web sencillo puede contener varios tipos de contenido como son páginas informativas, encuestas, noticias, blogs, etc., donde a cada elemento de contenido se lo denomina nodo y cada nodo pertenece a un único tipo de contenido, el cual está definido por varias configuraciones predeterminadas para todos los nodos del mismo tipo, como por ejemplo si el nodo se publica automáticamente o si permite la inserción de comentarios.

Los tipos de contenido usados para este proyecto fueron creados en base al análisis de la estructura del Reglamento General de Elecciones de la EPN, que se trata de nuestros caso de estudio, y tomando como referencia el artículo titulado "*DATA MODEL FOR STORAGE AND RETRIEVAL OF LEGISLATIVE DOCUMENTS IN DIGITAL LIBRARIES USING LINKED DATA*" [12], en el cual se menciona la estructura para los documentos legislativos del Ecuador y su uso en bibliotecas digitales. La estructura creada para el manejo de tipos de contenido del proyecto se muestra en la figura 17.

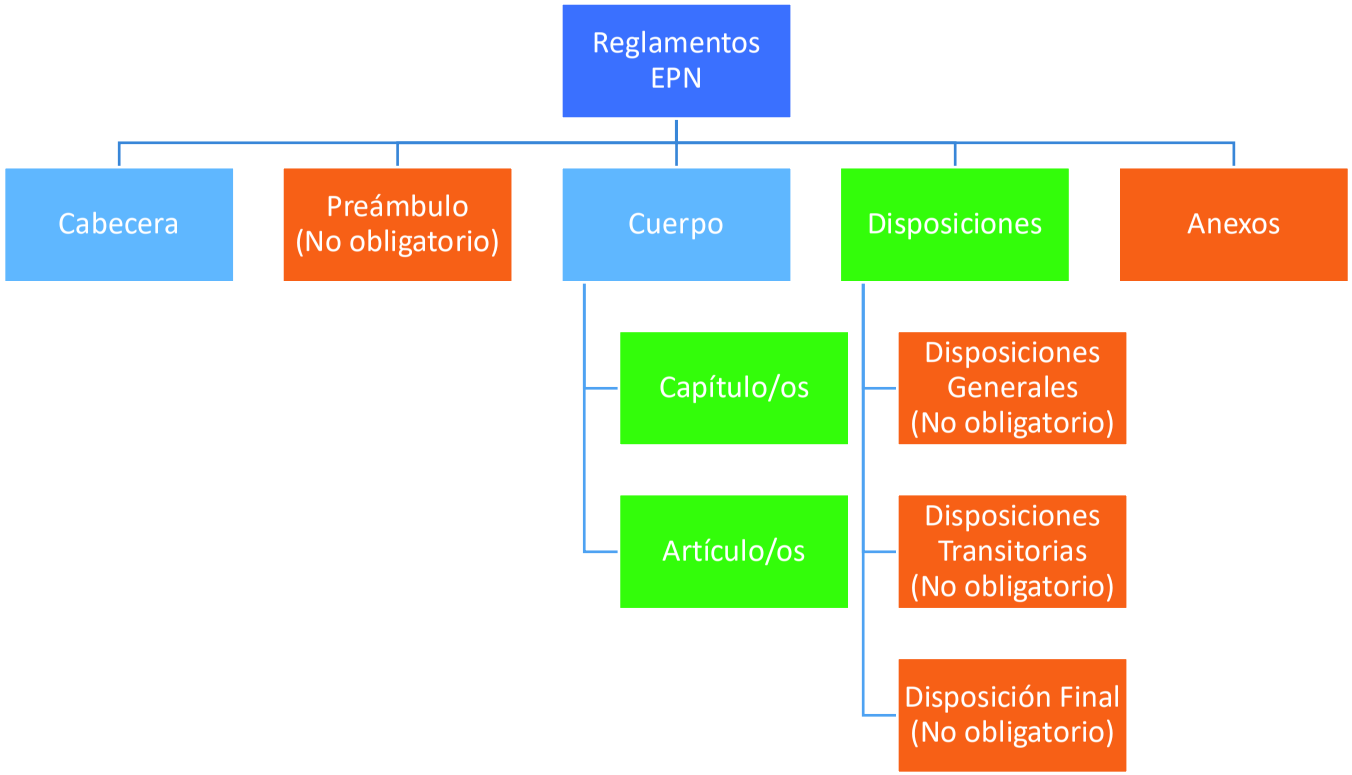


*Figura 17: Estructura creada para el manejo de tipos de contenido*

En la Figura anterior se puede observar que existen campos que no son obligatorios, esto se debe a que no todos los reglamentos contienen estos campos, por lo que en caso de que sea necesario su uso se los debe usar. El campo cuerpo puede tener o no capítulos, pero en cambios siempre tendrán artículos, por lo que existe la posibilidad de que el cuerpo esté compuesto de capítulos y artículos o también solo de artículos.

Una vez definida la estructura que tendrán los reglamentos de la EPN se deben crear los tipos de contenido que la representarán, por lo que para adaptar dicha estructura a Drupal 7 y con la finalidad de tener un mejor manejo de la información se definieron seis tipos de contenido. Cada uno de ellos tiene sus propios atributos, en donde algunos de ellos tienen relación con varios de los tipos de contenido definidos. Estos tipos de contenido son los siguientes: Artículos, Capítulos, Disposiciones Generales, Disposiciones Transitorias y Reglamento; este último va a estar conformado por la agrupación de los otros 5 tipos de contenido, y a su vez será el que se deba utilizar para el ingreso de la información. La creación y configuración de los tipos de contenido se detalla en la "Guía de Instalación y Configuración de Drupal 7", que se anexa al presente documento.

### 3.1.4. Metodologías para elaborar ontologías

Una metodología se define como un conjunto de procedimientos racionales, que se emplean para conseguir un objetivo o serie de objetivos, por lo que para la elaboración de ontologías existen varias metodologías, con sus propias características, que permiten del diseño de ontologías a partir de existentes o desde cero, es decir desde el entendimiento del dominio del problema que se quiere resolver.

El propósito de usar una ontología elaborada a través de una metodología en el proyecto es permitir realizar las traducciones entre diferentes herramientas de software al ser usada como un formato de intercambio de conocimiento. Dentro de las diversas metodologías que existen para el desarrollo de ontologías se pueden mencionar las siguientes:

***Metodología CYC***

La metodología CYC tiene como primicia el extraer manualmente el conocimiento común que está implícito en diferentes fuentes de información de primera mano, una vez obtenido suficiente conocimiento se puede adquirir nuevo conocimiento común usando herramientas de procesamiento de lenguaje natural o aprendizaje computacional. [24]

***Metodología de Uschold y King***

La metodología de Uschold y King propone algunos pasos para desarrollar ontologías que son: identificar el propósito, capturar los conceptos y relaciones entre estos conceptos y los términos utilizados para referirse a estos conceptos y relaciones y finalmente codificar la ontología. La ontología diseñada debe ser documentada y evaluada, además que permite la creación de ontologías a partir de una existente. [25]

***Metodología de Grüninger y Fox***

La metodología de Grüninger y Fox tiene como primer paso identificar intuitivamente las aplicaciones posibles en las que se usará la ontología, después se usa un conjunto de preguntas en lenguaje natural para determinar el ámbito de la ontología. Las preguntas son usadas para extraer los conceptos principales, sus propiedades, relaciones y axiomas. [26]

***Metodología KACTUS***

La metodología Kactus tiene su fundamento en adquirir una base de conocimiento en base a la abstracción, ya que en cuantas más aplicaciones se construyen, las ontologías se hacen más generales y se alejan de una base de conocimiento, por lo que se propone elaborar una base de conocimiento para una aplicación específica. Si se necesita una nueva base de conocimiento dentro de un dominio similar se generaliza la primera base de conocimiento en una ontología y se adapta para las dos aplicaciones, y así sucesivamente. [27]

***Metodología METHONTOLOGY***

La metodología Methontology tiene como objetivo construir ontologías tanto partiendo desde cero como reusando otras ontologías, o a través de un proceso de reingeniería e incluye los siguientes pasos: identificación del proceso de desarrollo de la ontología donde se incluyen las principales actividades, un ciclo de vida basado en prototipos evolucionados y la metodología propiamente dicha, que especifica los pasos a ejecutar en cada actividad, las técnicas usadas, los productos a obtener y cómo deben ser evaluados. [28]

***Metodología SENSUS***

La metodología Sensus es un enfoque *top-down* para derivar ontologías específicas del dominio a partir de grandes ontologías; de manera general proponen identificar un conjunto de términos semilla que son relevantes en un dominio particular. Tales términos se enlazan manualmente a una ontología de amplia cobertura con la cual los usuarios seleccionan automáticamente los términos relevantes para describir el dominio y acotar la ontología SENSUS. [29]

Para el desarrollo de la ontología a ser usada en el proyecto se seleccionó la metodología Methontology, ya que permite la creación de una ontología desde cero o a partir de una existente. En este proyecto nos basamos en lo propuesto en el artículo *"DATA MODEL FOR STORAGE AND RETRIEVAL OF LEGISLATIVE DOCUMENTS IN DIGITAL LIBRARIES USING LINKED DATA*" [12], por lo que esta metodología es útil al momento de empezar a adquirir experticia y apoya en la elaboración de una ontología adecuada para su implementación a lenguaje *owl* en un software apropiado.

Las actividades que se proponen en la metodología Methontology de manera general [30] son las siguientes:

- La actividad de especificación que permite determinar por qué se construye la ontología, cuál será su uso, y quiénes serán sus usuarios finales. [31]
- La actividad de conceptualización se encarga de organizar y convertir una percepción informal del dominio en una especificación semi-formal, para lo cual utiliza un conjunto de representaciones intermedias basadas en notaciones tabulares y gráficas. El resultado de esta actividad es el modelo conceptual de la ontología. [31]
- La actividad de formalización se encarga de la transformación de dicho modelo conceptual en un modelo formal o semi-computable. [31]
- La actividad de implementación construye modelos computables en un lenguaje de ontologías (Ontolingua, RDF Schema, OWL, etc.). La mayor parte de las herramientas de ontologías permiten llevar a cabo esta actividad de manera automática. [31]
- La actividad de mantenimiento se encarga de la actualización y/o corrección de la ontología, en caso de llegar a ser necesario. [31]

En la figura 18 se puede apreciar las tareas que implica la metodología de elaboración de ontologías Methontology, de las cuales se hace uso las primeras seis tareas, que permiten el diseño de la ontología, ya que las demás tareas son propias de la implementación de la misma, que se la realiza con el software Protégé, las cuales serán descritas de forma breve a continuación.

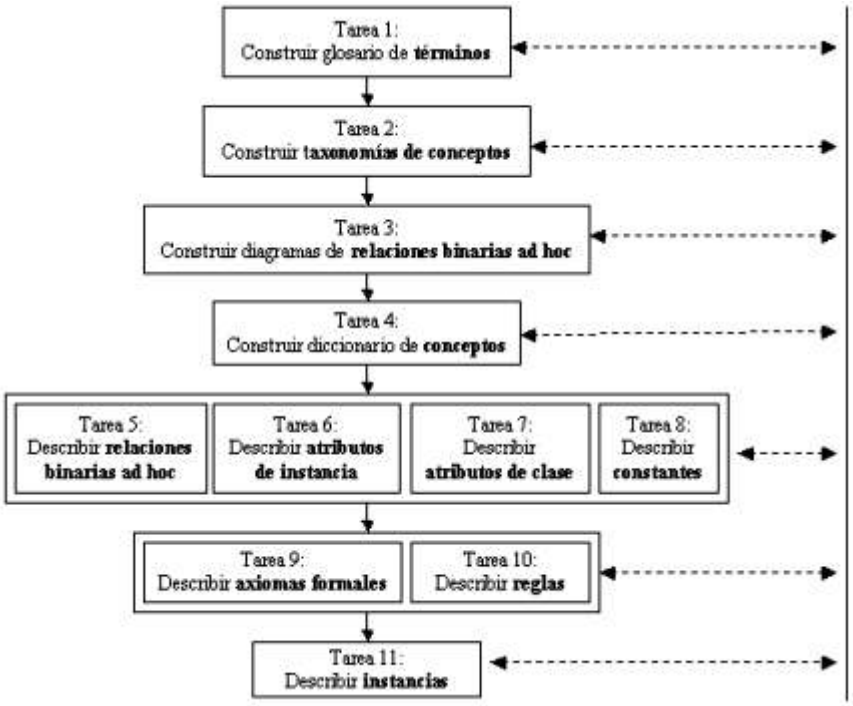


*Figura 18: Vista General de las tareas de Methontology [31]*

### 3.1.5. Diseño de ontología legislativa

#### Tarea 1: Construir glosario de términos

En la primera tarea se debe construir un glosario de términos que incluyan los términos relevantes del dominio y sus descripciones en lenguaje natural [31]. En la tabla 21 se puede ver el glosario de términos construido para el manejo de reglamentaciones.

*Tabla 21: Glosario de términos*

| Nombre | Descripción | Tipo |
|---|---|---|
| Reglamento | Nombre del reglamento de la Escuela Politécnica Nacional | Concepto |
| Fecha de Vigencia | Fecha en la cual empieza a tener validez un reglamento | Atributo de clase |
| Está compuesto por | El reglamento está compuesto por | Relación |
| Forma parte de | Una subclase forma parte de la clase principal | Relación |

#### Tarea 2: Construir taxonomías de conceptos

En la segunda tarea se construye la taxonomía de conceptos que definen su jerarquía, por los conceptos pueden estar contenidos unos en otros [31]. En la figura 19 se puede apreciar la taxonomía de conceptos construida para nuestra ontología.

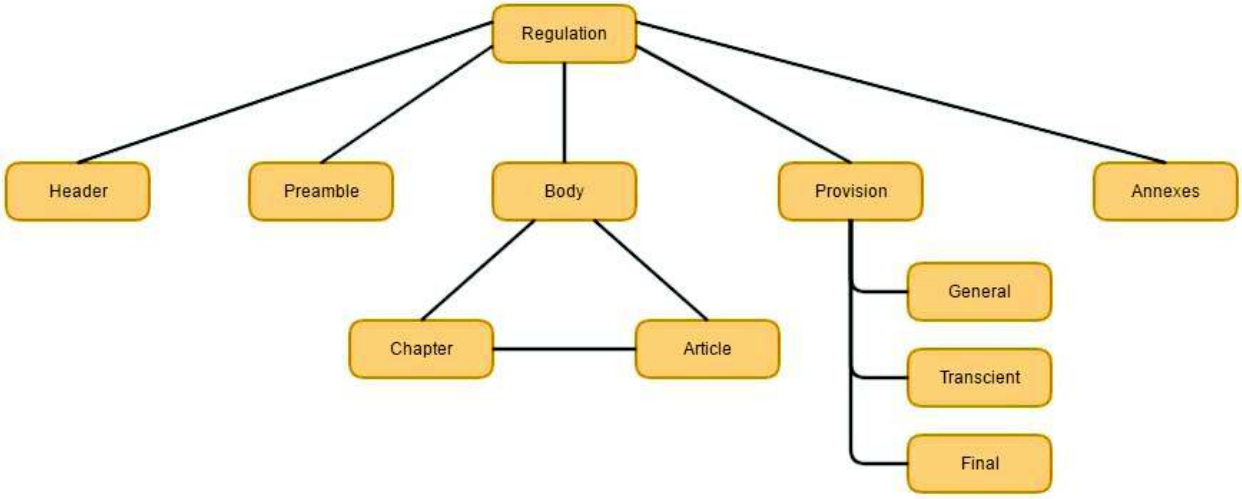


*Figura 19: Taxonomía de conceptos*

**Tarea 3: Construir diagramas de relaciones binarias ad hoc**

La tercera tarea de la metodología tiene como propósito establecer las relaciones binarias ad hoc existentes entre conceptos de la taxonomía propuesta [31]. En las figuras 20, 21, 22, 23, 24, 25, 26, 27, 28, 29 y 30 se muestran las relaciones binarias ad hoc de todos los conceptos.

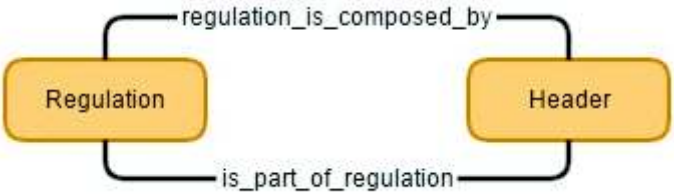


*Figura 20: Relación binaria ad hoc Regulation - Header*

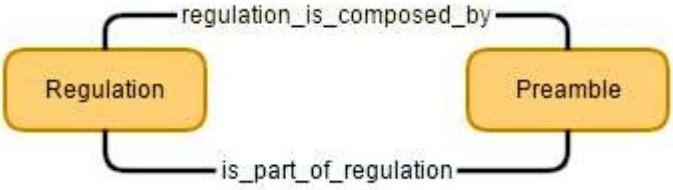


*Figura 21: Relación binaria ad hoc Regulation - Preamble*

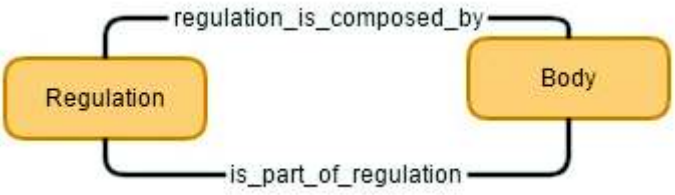


*Figura 22: Relación binaria ad hoc Regulation - Body*

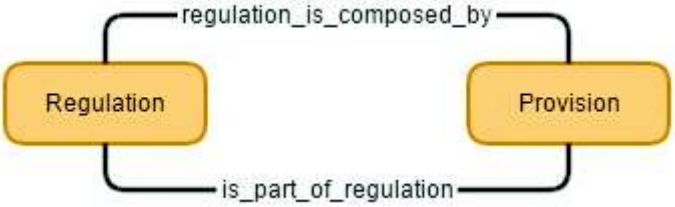


*Figura 23: Relación binaria ad hoc Regulation - Provision*

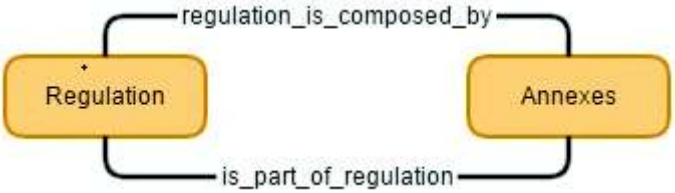


*Figura 24: Relación binaria ad hoc Regulation - Annexes*

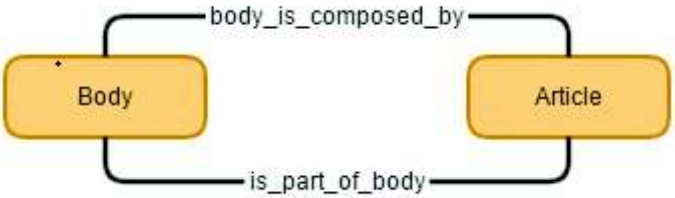


*Figura 25: Relación binaria ad hoc Body - Article*

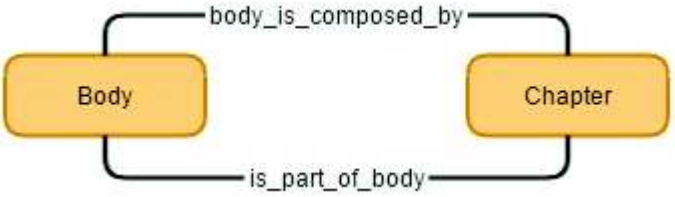


*Figura 26: Relación binaria ad hoc Body - Chapter*

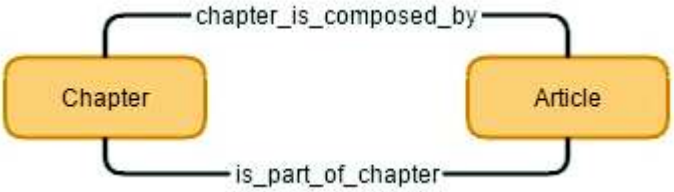


*Figura 27: Relación binaria ad hoc Chapter - Artice*

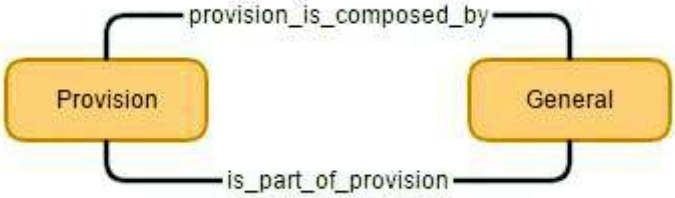


*Figura 28: Relación binaria ad hoc Provision - General*

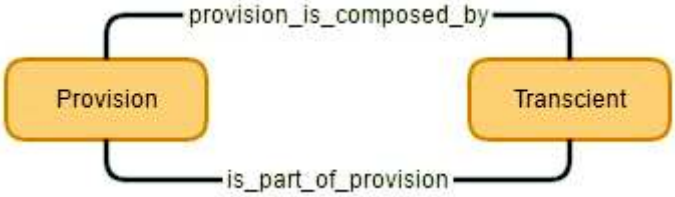


*Figura 29: Relación binaria ad hoc Provision - Transcient*

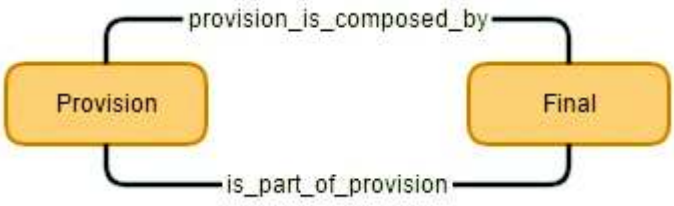


*Figura 30: Relación binaria ad hoc Provision – Final*

**Tarea 4: Construir diccionario de conceptos**

La cuarta tarea tiene como propósito contener los atributos que describen a los conceptos y sus respectivas relaciones [31]. En la tabla 22 se tiene el diccionario de conceptos elaborado para nuestra ontología.

*Tabla 22: Diccionario de conceptos*

| Nombre | Atributo | Relaciones |
|---|---|---|
| *Regulation* | | *Regulation_is_composed_by* |
| *Header* | *Regulation_Title* | *Is_part_of_regulation* |
| *Preamble* | *Description* | *Is_part_of_regulation* |
| *Body* | | *Is_part_of_regulation*<br>*Body_is_composed_by* |
| *Chapter* | *Name* | *Is_part_of_body*<br>*Chapter_is_composed_by* |
| *Article* | *Name*<br>*Description* | *Is_part_of_body*<br>*Is_part_of_chapter* |
| *Provision* | | *Is_part_of_regulation*<br>*Provision_is_composed_by* |
| *General* | *Name*<br>*Description* | *Is_part_of_provision* |
| *Transcient* | *Name*<br>*Description* | *Is_part_of_provision* |
| *Final* | *Description* | *Is_part_of_provision* |
| *Annexes* | *Document* | *Is_part_of_regulation* |

**Tarea 5: Describir relaciones binarias ad hoc**

La quita tarea se utiliza para detallar las propiedades de las relaciones ad hoc establecidas en la taxonomía de los conceptos, especificar su nombre, los nombres de sus conceptos origen y destino, su cardinalidad y su relación inversa, si existe [31]. La tabla 23 muestra las relaciones binarias ad hoc de forma sintetizada.

*Tabla 23: Relaciones binarias ad-hoc*

| Nombre de la relación | Concepto Origen | Cardinalidad | Concepto Destino | Relación Inversa |
|---|---|---|---|---|
| *Regulation_is_composed_by* | *Regulation* | 1:N | *Header Preamble Body Provision Annexes* | *Is_part_of_regulation* |
| *Body_is_composed_By* | *Body* | 1:N | *Chapter Article* | *Is_part_of_body* |
| *Chapter_is_composed_By* | *Chapter* | 1:N | *Article* | *Is_part_of_chapter* |
| *Provision_is_composed_By* | *Provision* | 1:N | *General Transcient Final* | *Is_part_of_provision* |

**Tarea 6: Describir atributos de instancia**

El objetivo de la sexta tarea es describir en detalle todos los atributos de instancia incluidos en el diccionario de conceptos. Cada fila de la tabla de atributos de instancia contiene la descripción detallada de un atributo de instancia [31]. En la tabla 24 se puede apreciar los atributos de instancia para nuestra ontología.

*Tabla 24: Atributos de instancia*

| Nombre del atributo | Concepto | Tipo Valor | Rango Valores | Cardinalidad |
|---|---|---|---|---|
| *Regulation_Title* | *Header* | Cadena de caracteres | - | (1,1) |
| *Description* | *Preamble* | Cadena de caracteres | - | (1,1) |
| *Name* | *Chapter* | Cadena de caracteres | - | (1,1) |
| *Name* | *Article* | Cadena de caracteres | - | (1,1) |
| *Description* | *Article* | Cadena de caracteres | - | (1,1) |
| *Name* | *General* | Cadena de caracteres | - | (1,1) |
| *Description* | *General* | Cadena de caracteres | - | (1,1) |
| *Name* | *Transcient* | Cadena de caracteres | - | (1,1) |
| *Description* | *Transcient* | Cadena de caracteres | - | (1,1) |
| *Description* | *Final* | Cadena de caracteres | - | (1,1) |
| *Document* | *Annexes* | Cadena de caracteres | - | (1,1) |

El resultado de la elaboración de la ontología “regulationT” se puede apreciar en la figura 31 que sigue a continuación:

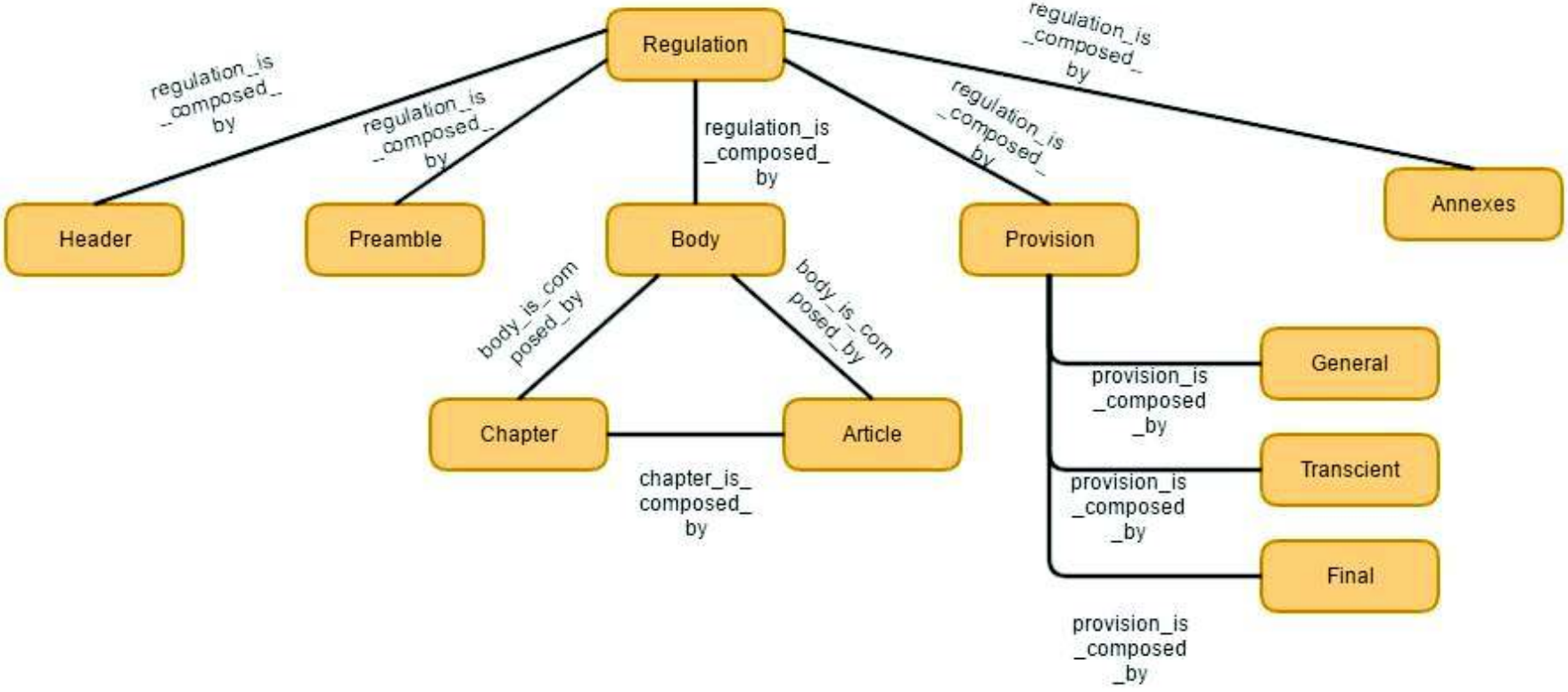


*Figura 31: Ontología “regulationT”*

### 3.1.6. Implementación de ontología legislativa

Para la implementación de la ontología legislativa se usó el software libre Protégé, en vista a que su arquitectura se puede adaptar a la creación de aplicaciones simples o complejas basadas en ontologías, además que es totalmente compatible con las últimas especificaciones OWL 2 *Web Ontology Language* y RDF *Specifications* de la *World Wide Web Consortium* (W3C) [32]. Este software puede descargarse del sitio oficial [33], cuyo ícono se muestra en la figura 32.


*Figura 32: Ícono de Software Protégé*

Una de las ventajas principales de este software es que está auspiciado por la Universidad de Stanford y es apoyado por una comunidad de software que hace aportes constantes para mejorar su funcionalidad. La interfaz de usuario es altamente configurable y permite crear el entorno perfecto su uso, además la colaboración de la comunidad incluye compartir contenidos nuevos, notas y discusiones, calendarizadores y notificaciones por correo electrónico. RDF / XML, Turtle, OWL / XML, OBO y otros formatos disponibles para la carga y descarga ontológica [32].

El uso del software no es muy complicado, ya que es intuitivo, además de existir varios tutoriales en la red que permiten aprender a usarlo de una manera rápida. Lo primero que se debe hacer dentro del software es darle un nombre a la ontología que se va a crear, para el caso actual se la denominó con el nombre “regulationT”, como se ve en la figura 33.

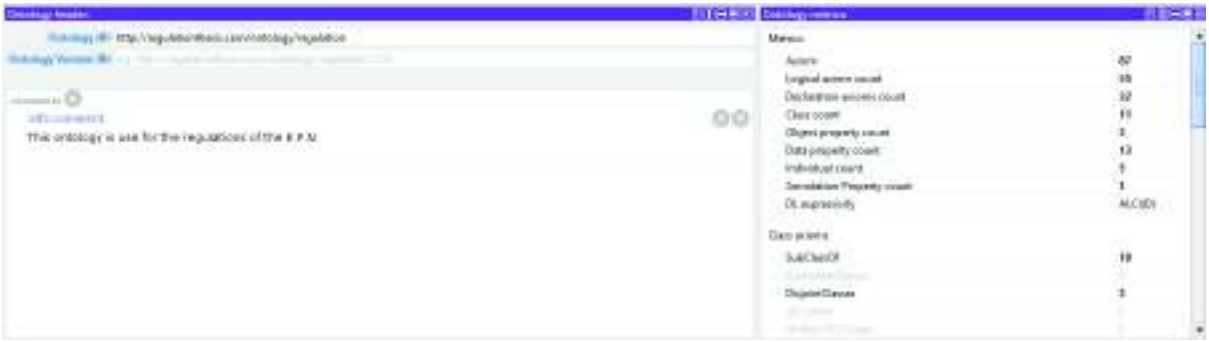

*Figura 33: Resumen ontología regulationT en software Protégé*

Para la creación de la ontología se deben crear las clases que van a componer nuestra ontología; para nuestro caso son las siguientes:

- *Regulation*
  - *Annexes*
  - *Body*
    - *Article*
    - *Chapter*
  - *Header*
  - *Preamble*
  - *Provision*
    - *Final*
    - *General*
    - *Transcient*

Las clases mencionadas anteriormente obedecen a la estructura planteada en la figura 17 que la usamos para la administración de los diferentes tipos de contenido. Dentro del software Protégé vamos a manejar la misma estructura detallada y además va a ser visualizada en forma de árbol, de tal manera que la lectura de la jerarquía de clases de la ontología "regulationT" se vuelve sencilla, tal como se ilustra en la figura 34.

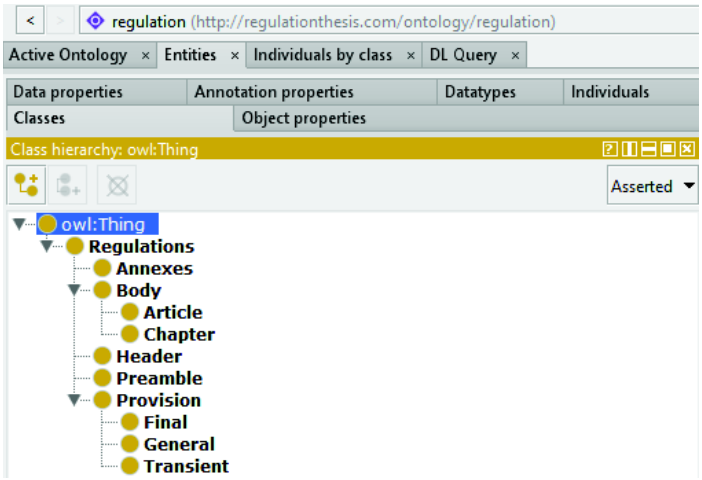


*Figura 34: Clases de ontología regulationT en software Protégé*

Cada una de las clases creadas debe tener asociada anotaciones que especifican el dominio y rangos en los cuales va a trabajar esa entidad, para el caso de la clase *Annexes* podemos apreciar la información de la descripción de la clase en la que tenemos si es subclase de otra, las instancias e incluso si no hace alguna asociación con otras clases (*disjoin*), como se aprecia en la figura 35.

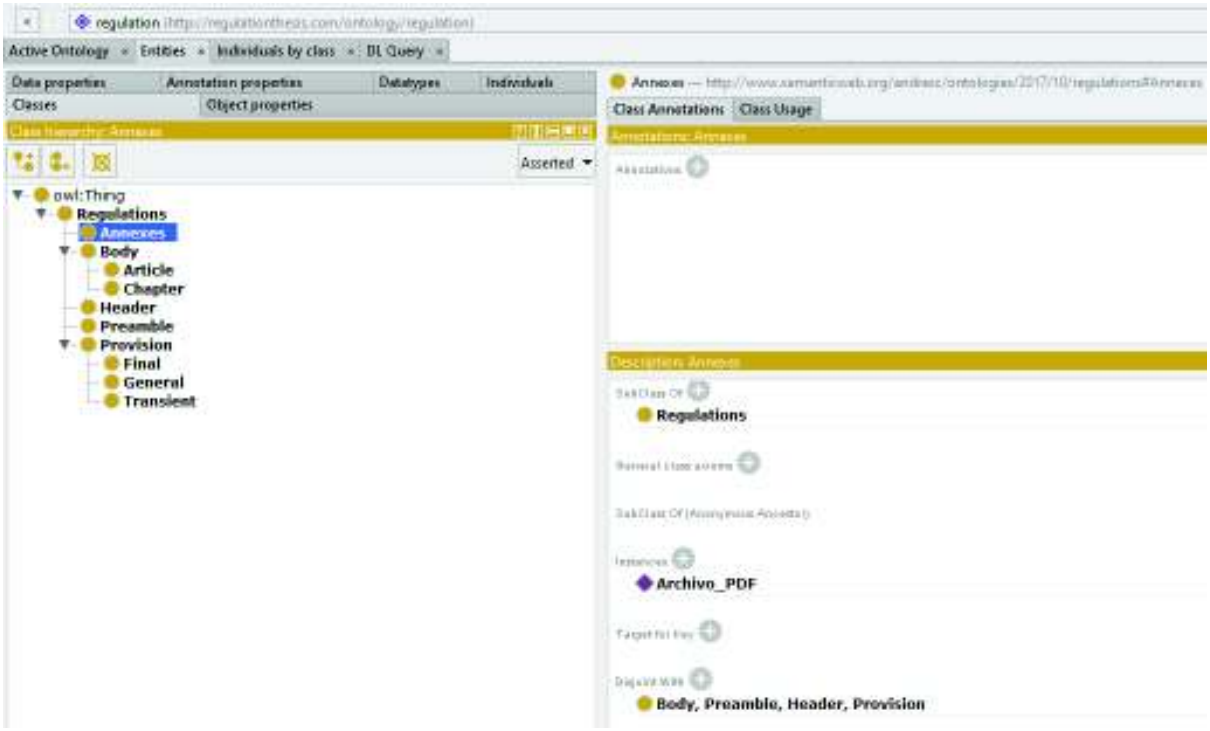


*Figura 35: Anotaciones de la clase "Annexes"*

La información correspondiente a la descripción de las clases debe ser ingresada de acuerdo a la estructura que se haya planteado para la elaboración de la ontología, por lo cual es preciso ser consistentes entre el diseño y la implementación en el software que permite la creación de ontologías.

Otro paso es la definición de propiedades de objetos con su respectiva jerarquía, que para el caso de la ontología "regulationT" nos va a servir para indicar las relaciones entre las clases y también su relación inversa, con el fin de definir los dominios y rangos de los mismos; se debe tener en cuenta que las propiedades que vamos a manejar entre los objetos debe especificar la relación directa y la relación inversa de las mismas, de tal modo que podamos decir por ejemplo que la clase *Body* está compuesta por las clases *Article* y *Chapter* lo que correspondería a la relación directa, y a su vez la clase *Article* es parte de la clase *Body* y también que la clase *Chapter* es parte de la clase *Body,* que correspondería a la relación inversa; este tipo de relación empieza ya a proporcionar de semántica a la estructura del contenido, lo cual se puede apreciar en la figura 36.

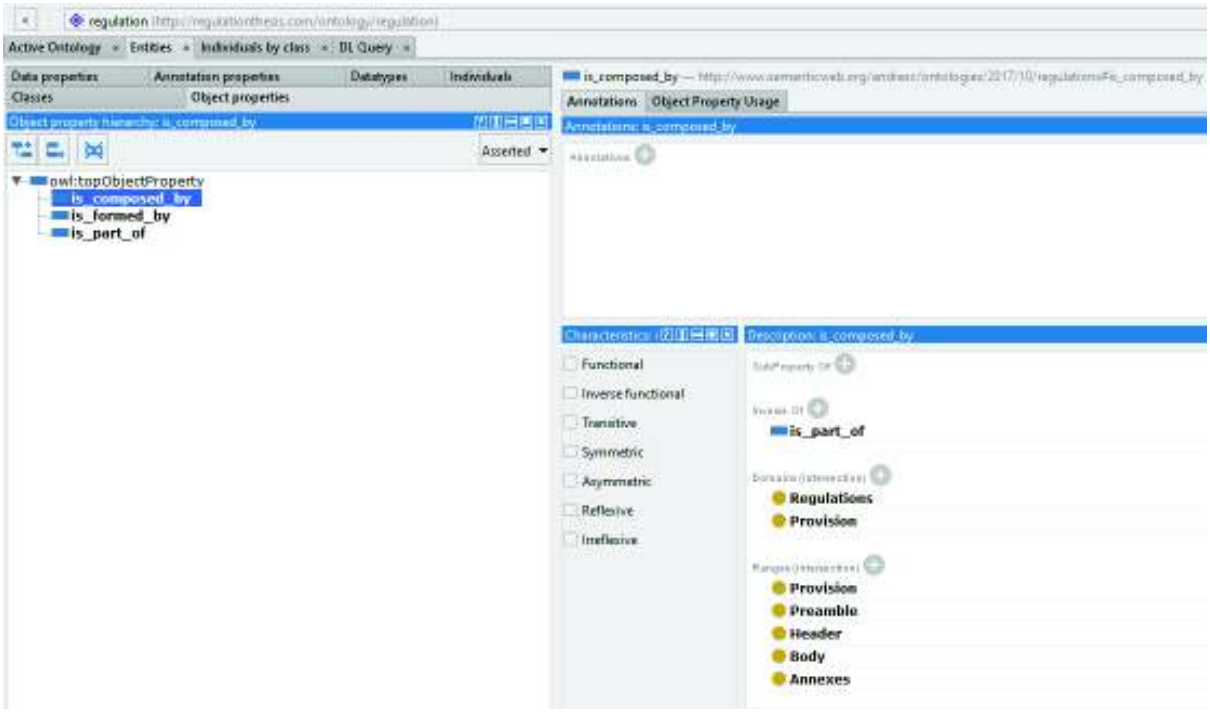


*Figura 36: Propiedades de los objetos de la ontología "regulationT"*

Como producto final de la implementación de la ontología se tiene un archivo con extensión *owl*, el mismo que debe ser importado desde el CMS Drupal para su implementación como vocabulario nuevo y posterior uso. Una vez que la ontología ha sido añadida al CMS, cuando se realice el mapeo de los diferentes tipos de contenido, vamos a poder apreciar las propiedades de la ontología dentro de las opciones de mapeo, con lo que vamos a poder dar uso a la ontología creada. En la figura 37 se aprecia una parte del código de la ontología.

regulation.owl

```
@prefix default:  <http://regulationthesis.com/ontology/regulation#> .
@prefix xsp:      <http://www.owl-ontologies.com/2005/08/07/xsp.owl#> .
@prefix owl:      <http://www.w3.org/2002/07/owl#> .
@prefix rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd:      <http://www.w3.org/2001/XMLSchema#> .
@prefix protege:  <http://protege.stanford.edu/plugins/owl/protege#> .
@prefix swrl:     <http://www.w3.org/2003/11/swrl#> .
@prefix swrlb:    <http://www.w3.org/2003/11/swrlb#> .
@prefix rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .

default:is_part_of_chapter
      a       owl:ObjectProperty ;
      rdfs:domain default:Article ;
      rdfs:range default:Chapter ;
      owl:inverseOf default:chapter_is_composed_by .

default:transient_provision_description
      a       owl:DatatypeProperty ;
      rdfs:domain default:Transient ;
      rdfs:range xsd:string .

default:Regulation
      a       owl:Class .
```

*Figura 37: Sección del archivo de ontología "regulationT"*

### 3.1.7. Pruebas sobre el sitio Web

A continuación, se presentan las pruebas funcionales que se realizaron sobre el sitio Web que corresponden a las historias de usuario y criterios de aceptación. En las tablas 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35 y 36 se detallan las pruebas de aceptación de cada una de las historias de usuario que se definieron anteriormente.

*Tabla 25: Prueba de Aceptación 1 – Definir arquitectura*

| **Prueba de Aceptación (PA1)** |
|---|
| **Número: 1**<br>**Historia de usuario: Definir arquitectura.**<br>**ID: HU01** |
| **Descripción de la prueba:**<br>Revisar la arquitectura definida para validar si es posible su posterior implementación. |
| **Prerrequisitos:**<br>Tener conocimiento de las herramientas a ser usadas. |
| **Pasos de ejecución:**<br>• Revisar uso de CMS.<br>• Revisar tecnologías de *Linked Data*.<br>• Revisar prototipo de la arquitectura planteada. |
| **Resultado esperado:**<br>Prototipo de la arquitectura planteada. |
| **Criterios de aceptación:**<br>• Debe tener componentes de software libre.<br>• Debe hacer uso de un CMS.<br>• Debe incluir tecnologías de *Linked Data*. |
| **Resultado obtenido:**<br>Prototipo de arquitectura validado y listo para implementarlo. |

*Tabla 26: Prueba de Aceptación 2 – Implementar arquitectura*

| **Prueba de Aceptación (PA2)** |
|---|
| **Número: 2**<br>**Historia de usuario: Implementar arquitectura.**<br>**ID: HU02** |
| **Descripción de la prueba:**<br>Revisar la implementación de arquitectura planteada a través de verificaciones de la funcionalidad básica del sitio Web así como de los demás componentes de la infraestructura. |
| **Prerrequisitos:**<br>Prototipo de arquitectura planteada para ser implementada. |
| **Pasos de ejecución:**<br>• Verificar implementación de CMS Drupal 7.<br>• Verificar implementación de servidor de aplicaciones.<br>• Verificar base de datos instalada.<br>• Verificar módulos de Drupal 7 instalados. |
| **Resultado esperado:**<br>Navegación dentro de la arquitectura implementada en un sitio Web. |
| **Criterios de aceptación:**<br>• Base de datos instalada debe ser MySQL.<br>• El servidor de aplicaciones debe ser Apache.<br>• El CMS instalado debe ser Drupal en su versión 7.<br>• Los módulos instalados deben ser los siguientes:<br>  ○ Arc2 store.<br>  ○ CKEditor - WYSIWYG HTML editor.<br>  ○ Conditional fields.<br>  ○ Chaos tool suite.<br>  ○ Date.<br>  ○ Devel.<br>  ○ Diff.<br>  ○ Diff different.<br>  ○ Entity reference.<br>  ○ Entity Api.<br>  ○ Field as block.<br>  ○ Inline entity form.<br>  ○ Libraries Api.<br>  ○ Pathauto.<br>  ○ Printer, email and PDF versions.<br>  ○ RDF indexer.<br>  ○ RDF Extensions.<br>  ○ SPARQL.<br>  ○ Token.<br>  ○ Transliteration.<br>  ○ Views.<br>  ○ Context. |
| **Resultado obtenido:**<br>• Navegación exitosa dentro del sitio Web sin módulos habilitados.<br>• Verificación de la instalación de componentes de arquitectura exitosa**.** |

*Tabla 27: Prueba de Aceptación 3 – Configurar módulos*

| **Prueba de Aceptación (PA3)** |
|---|
| **Número: 3**<br>**Historia de usuario: Configurar módulos.**<br>**ID: HU03** |
| **Descripción de la prueba:**<br>Revisar que la configuración de los módulos haya sido exitosa a través de una nueva navegación en la cual no se presenten errores asociados a la habilitación de módulos o dependencias de otros no instalados. |
| **Prerrequisitos:**<br>Haber instalado todos los componentes detallados en la arquitectura. |
| **Pasos de ejecución:**<br>• Navegar a través de las opciones de configuración del CMS.<br>• Navegar a través de la opción “Estructura” del CMS. |
| **Resultado esperado:**<br>Navegación dentro del CMS sin errores de configuración ni dependencias de módulos no instalados. |
| **Criterios de aceptación:**<br>• La navegación en el CMS no debe presentar errores.<br>• Los módulos habilitados deben ser los siguientes:<br>  ○ Arc2 store.<br>  ○ CKEditor - WYSIWYG HTML editor.<br>  ○ Conditional fields.<br>  ○ Chaos tool suite.<br>  ○ Date.<br>  ○ Devel.<br>  ○ Diff.<br>  ○ Diff different.<br>  ○ Entity reference.<br>  ○ Entity Api.<br>  ○ Field as block.<br>  ○ Inline entity form.<br>  ○ Libraries Api.<br>  ○ Pathauto.<br>  ○ Printer, email and PDF versions.<br>  ○ RDF indexer.<br>  ○ RDF Extensions.<br>  ○ SPARQL.<br>  ○ Token.<br>  ○ Transliteration.<br>  ○ Views.<br>  ○ Context. |
| **Resultado obtenido:**<br>• Los módulos descritos en los criterios de aceptación se encontraron habilitados.<br>• La navegación en el CMS no presentó errores de configuración ni de dependencias de módulos faltantes. |

*Tabla 28: Prueba de Aceptación 4 – Diseñar e implementar tipos de contenido*

| **Prueba de Aceptación (PA4)** |
| --- |
| **Número: 4**<br>**Historia de usuario: Diseñar e implementar tipos de contenido.**<br>**ID: HU04** |
| **Descripción de la prueba:**<br>Revisar el diseño de los tipos de contenido y su implementación dentro del CMS. |
| **Prerrequisitos:**<br>Tener habilitado y funcionando correctamente el CMS instalado. |
| **Pasos de ejecución:**<br>• Revisar el diseño de la estructura del reglamento<br>• Revisar los tipos de contenido creados y configurados dentro del CMS. |
| **Resultado esperado:**<br>Tipos de contenidos creados en el CMS de acuerdo a la estructura de reglamento diseñada, y lista para ser usada. |
| **Criterios de aceptación:**<br>• Diseño de estructura de reglamento.<br>• Tipos de contenido creados de acuerdo al diseño de estructura del reglamento. |
| **Resultado obtenido:**<br>Los tipos de contenido fueron creados correctamente de acuerdo al diseño que se planteó. |

*Tabla 29: Prueba de Aceptación 5 – Diseñar e implementar ontología legislativa*

| **Prueba de Aceptación (PA5)** |
|---|
| **Número: 5**<br>**Historia de usuario: Diseñar e implementar ontología legislativa**<br>**ID: HU05** |
| **Descripción de la prueba:**<br>Revisar la ontología diseñada e implementada “regulationT”. |
| **Prerrequisitos:**<br>Conocer el funcionamiento del software Protégé para la elaboración de ontologías. |
| **Pasos de ejecución:**<br>• Revisar el diseño de la ontología “regulationT”.<br>• Revisar la ontología elaborada en el software Protégé.<br>• Revisar la importación de la ontología en el CMS. |
| **Resultado esperado:**<br>Contar con una ontología adaptada al dominio de reglamentaciones que pueda ser usada en el CMS. |
| **Criterios de aceptación:**<br>• Archivo en extensión *owl* de la ontología “regulationT”.<br>• Ontología agregada dentro del CMS.<br>• Validación de que se pueda hacer uso de la ontología importada dentro del CMS. |
| **Resultado obtenido:**<br>• Primer intento: En el primer intento no se pudo verificar el uso de la ontología importada debido a que no se presentó el vocabulario para el uso de sus propiedades.<br>• Segundo intento: Una vez corregida la importación se pudo verificar el correcto uso de la ontología, al poder ver que se presentaban las propiedades del vocabulario importado en la opción de mapeo de los diferentes tipos de contenido. |

*Tabla 30: Prueba de Aceptación 6 – Crear base de datos RDF*

| **Prueba de Aceptación (PA6)** |
|---|
| **Número: 6**<br>**Historia de usuario: Crear base de datos RDF**<br>**ID: HU06** |
| **Descripción de la prueba:**<br>Revisar la creación de la base de datos para RDF. |
| **Prerrequisitos:**<br>Tener el sistema de gestión de base de datos MySQL. |
| **Pasos de ejecución:**<br>• Revisar la base de datos RDF creada.<br>• Revisar la estructura de la misma que cumpla formato RDF. |
| **Resultado esperado:**<br>Base de datos RDF creada y lista para almacenar información. |
| **Criterios de aceptación:**<br>• Base de datos RDF creada en MySQL. |
| **Resultado obtenido:**<br>Se verificó la estructura RDF de la base de datos creada en MySQL. |

*Tabla 31: Prueba de Aceptación 7 – Mapear tipos de contenido*

| **Prueba de Aceptación (PA7)** |
|---|
| **Número: 7**<br>**Historia de usuario: Mapear tipos de contenido.**<br>**ID: HU07** |
| **Descripción de la prueba:**<br>Realizar el mapeo de los tipos de contenido creados. |
| **Prerrequisitos:**<br>• Ontología agregada al CMS<br>• Base de datos RDF creada. |
| **Pasos de ejecución:**<br>• Realizar el mapeo de tipos de contenido creados. |
| **Resultado esperado:**<br>Tipo de contenidos mapeados. |
| **Criterios de aceptación:**<br>• Verificación de la estructura del CMS, dentro de sus opciones de tipos de contendido, que se encuentren mapeados de acuerdo a la estructura de tipos de contenido de reglamentos y a la ontología “regulationT” creada. |
| **Resultado obtenido:**<br>Los tipos de contenido fueron mapeados de acuerdo a la estructura de reglamentos propuesta con propiedades de la ontología “regulationT”. |

*Tabla 32: Prueba de Aceptación 8 – Ingresar contenido*

| **Prueba de Aceptación (PA8)** |
|---|
| **Número: 8**<br>**Historia de usuario: Ingresar contenido.**<br>**ID: HU08** |
| **Descripción de la prueba:**<br>Realizar el ingreso de un reglamento, usando la interfaz gráfica del CMS con su opción de ingreso de nuevo contenido. |
| **Prerrequisitos:**<br>• Tener configurado el CMS Drupal 7.<br>• Tener creados los tipos de contenido de Reglamentos.<br>• Tener mapeados los tipos de contenidos con la ontología "regulationT" creada.<br>• Tener creada la base de datos RDF. |
| **Pasos de ejecución:**<br>• Crear un nuevo contenido.<br>• Ingresar información del Reglamento General de Elecciones de la EPN en el CMS.<br>• EL ingreso de la información debe hacerse por cada uno de los tipos de contenido que se presentarán en la interfaz gráfica, de tal modo que se guarden los nodos correspondientes a dichos tipos de contenido. |
| **Resultado esperado:**<br>Nuevo contenido ingresado, Reglamento General de Elecciones de la EPN, en el CMS. |
| **Criterios de aceptación:**<br>• Funcionalidad que permita el ingreso de un nuevo contenido.<br>• Verificación de la estructura de Reglamentaciones en los tipos de contenido que se pide ingresar en el CMS.<br>• Visualización en el sitio Web del Reglamento General de Elecciones de la EPN. |
| **Resultado obtenido:**<br>El ingreso de contenido se realizó a través de una interfaz gráfica del CMS, posterior a lo cual se verificó en el sitio Web que el Reglamento General de Elecciones de la EPN se encontraba ingresado. |

*Tabla 33: Prueba de Aceptación 9 – Verificar endpoint del sitio Web*

| **Prueba de Aceptación (PA9)** |
|---|
| **Número: 9**<br>**Historia de usuario: Verificar *endpoint* del sitio Web.**<br>**ID: HU09** |
| **Descripción de la prueba:**<br>Verificar que se pueda hacer uso del *endpoint* del Sitio Web para la realización de consultas SPARQL. |
| **Prerrequisitos:**<br>Tener correctamente configurado el módulo SPARQL de Drupal. |
| **Pasos de ejecución:**<br>• Ingresar a la opción "*Sparql Endpoint*" de Drupal.<br>• Realizar la consulta por defecto del *endpoint*.<br>• Verificar el resultado de la consulta como información RDF. |
| **Resultado esperado:**<br>Endpoint listo para recibir consultas |
| **Criterios de aceptación:**<br>• Consulta SPARQL debe devolver información RDF. |
| **Resultado obtenido:**<br>El *endpoint* devolvió información de tripletas RDF correctamente. |

*Tabla 34: Prueba de Aceptación 10 – Diseñar e implementar pruebas*

| **Prueba de Aceptación (PA10)** |
|---|
| **Número: 10**<br>**Historia de usuario: Diseñar e implementar pruebas.**<br>**ID: HU010** |
| **Descripción de la prueba:**<br>Revisar las pruebas de aceptación propuestas para las historias de usuario. |
| **Prerrequisitos:**<br>N/A |
| **Pasos de ejecución:**<br>• Revisar las pruebas de aceptación propuestas para cada historia de usuario. |
| **Resultado esperado:**<br>Tablas con pruebas de aceptación validadas. |
| **Criterios de aceptación:**<br>• Deben existir Pruebas de Aceptación por cada historia de usuario. |
| **Resultado obtenido:**<br>Se tienen revisadas y aprobadas las pruebas de aceptación. |

*Tabla 35: Prueba de Aceptación 11 – Diseñar e implementar consultas SPARQL*

| **Prueba de Aceptación (PA11)** |
|---|
| **Número: 11**<br>**Historia de usuario: Diseñar e implementar consultas SPARQL.**<br>**ID: HU011** |
| **Descripción de la prueba:**<br>Revisar las consultas SPARQL diseñadas y verificar sus resultados a través del e*ndpoint* del sitio Web. |
| **Prerrequisitos:**<br>• Tener ingresado el contenido “Reglamento General de Elecciones” de la EPN.<br>• Contar con el *endpoint* del sitio Web habilitado. |
| **Pasos de ejecución:**<br>• Revisar las consultas SPARQL diseñadas.<br>• Ejecutar las consultas diseñadas a través del *endpoint* del sitio Web. |
| **Resultado esperado:**<br>Resultados en formato RDF. |
| **Criterios de aceptación:**<br>• Revisar que el resultado de las consultas ejecutadas presente información en formato RDF.<br>• Las consultas SPARQL deben ser realizadas a través del *endpoint* del sitio Web. |
| **Resultado Obtenido:**<br>Cuando se ejecutaron las consultas SPARQL a través del *endpoint* del sitio Web se obtuvieron los resultados en formato RDF y presentados en la pantalla, además que se permitió la descarga de dicha información en un archivo, el mismo que puede ser descargado en varios formatos como JSON, RDF, XML, etc. |

*Tabla 36: Prueba de Aceptación 12 – Documentar instalación, configuración y uso del sitio Web*

| **Prueba de Aceptación (PA12)** |
|---|
| **Número: 12**<br>**Historia de usuario: Documentar instalación, configuración y uso del sitio Web.**<br>**ID: HU012** |
| **Descripción de la prueba:**<br>Revisar la documentación de la instalación, configuración y uso del sitio Web validando que contenga todos los puntos requeridos para su correcta implementación y manejo. |
| **Prerrequisitos:**<br>• Tener conocimiento de las herramientas a ser usadas. |
| **Pasos de ejecución:**<br>• Revisar la instalación de Drupal 7.<br>• Revisar la instalación de los módulos externos para Drupal 7.<br>• Revisar la configuración de los módulos para Drupal 7.<br>• Saber manejar el sitio web para la administración del contenido. |
| **Resultado esperado:**<br>Documentos para la instalación, configuración y uso del sitio Web. |
| **Criterios de aceptación:**<br>• Se debe tener una guía de instalación y configuración debe explicar cómo se debe instalar Drupal 7 en Linux.<br>• La guía de instalación y configuración debe explicar cómo se deben instalar y configurar todos los módulos necesarios para el funcionamiento del sitio Web.<br>• Se debe tener un manual de usuario, el cual debe explicar cómo se debe manejar el sitio Web para la administración de la información en el mismo. |
| **Resultado Obtenido:**<br>Se obtuvieron dos documentos:<br>• Guía de instalación y configuración Drupal 7.<br>• Manual de usuario. |

### 3.1.8. Ingreso de contenido

Una vez que el sitio fue configurado se procedió a ingresar un nuevo contenido para hacer uso de las opciones parametrizadas en el CMS, para lo cual nos dirigimos a la página principal del sitio Web, que para el caso del servidor local es:

*http://localhost/ManagementOfRegulations/*

Después del desarrollo local, el sitio se encuentra actualmente publicado en la siguiente URL:

*http://regulationthesis.com/ManagementOfRegulations/*

En cualquiera de los dos casos se procede de la siguiente manera:

En primer lugar, se ingresa al sitio Web de manejo de reglamentaciones, en donde se encuentra una página de bienvenida al sitio, con las opciones disponibles para cualquier usuario y también con la opción de inicio de sesión para un usuario registrado, tal como se muestra en la figura 38.


*Figura 38: Página de bienvenida del sitio Web de Manejo de Reglamentaciones*

Una vez dentro del sitio, hay que dirigirse a la pestaña de “Agregar Contenido” (*Add Content*) y seleccionamos la opción Reglamentos EPN, como se muestra en la figura 39.

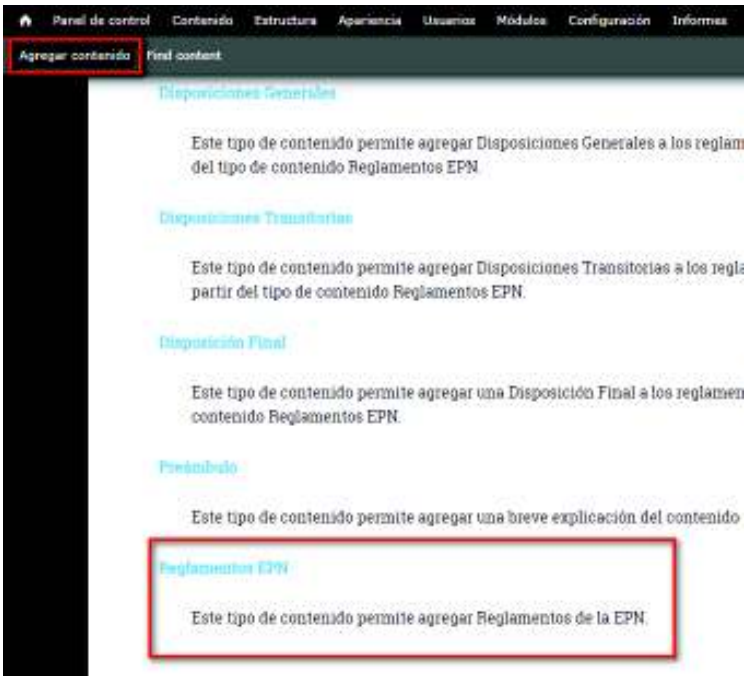


*Figura 39: Página de Contenido disponible en el CMS*

Se abrirá un formulario donde se procede al ingreso de la información solicitada, en la estructura de reglamentos creada, pidiendo los campos necesarios para este contenido, empezando por el nombre del reglamento y el preámbulo, lo cual puede ser apreciado en la figura 40.

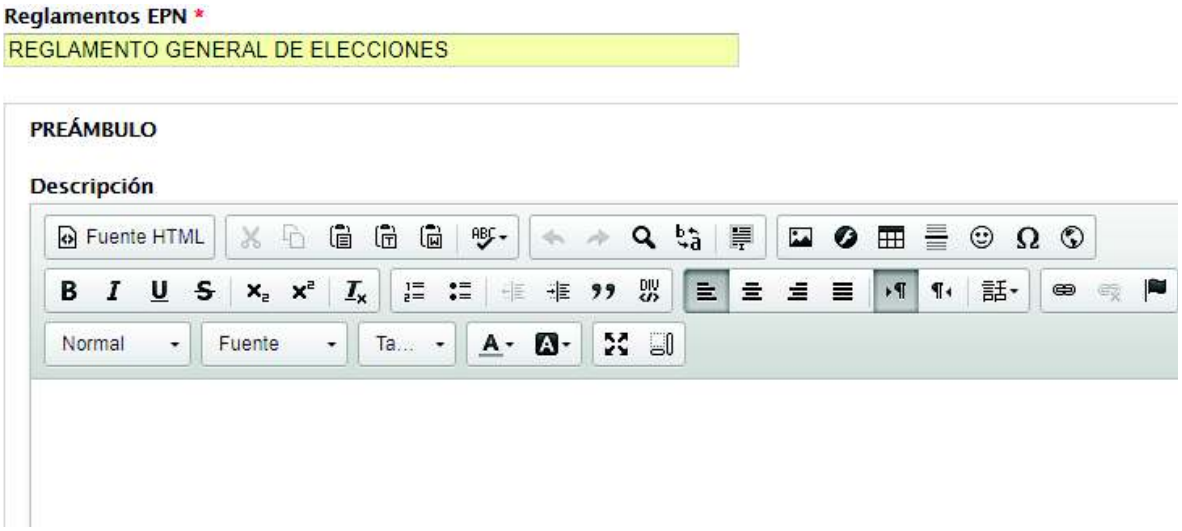


*Figura 40: Ingreso de contenido – Nombre del Reglamento*

Como siguiente punto se pide elegir el tipo de cuerpo, que puede ser únicamente artículos o de tipo capítulos y artículos, para el caso del Reglamento General de Elecciones de la EPN es la segunda opción mencionada, dichas opciones se pueden ver en la figura 41.


*Figura 41: Ingreso de contenido – Cuerpo del reglamento*

Una vez seleccionado el tipo de cuerpo del reglamento, se debe ingresar el nombre del capítulo, posterior a lo cual se crea un nuevo nodo para ingresar la información correspondiente a los artículos contenidos en el capítulo, dichos campos se aprecian en la figura 42.


*Figura 42: Ingreso de contenido – Capítulo*

La clase Artículo pide ingresar la información del nombre y descripción del artículo, dichos campos se poder ver en la figura 43.


*Figura 43: Ingreso de contenido – Artículo*

Para guardar con un formato específico la información de la descripción de los campos de información es necesario seleccionar la opción “Full HTML”, de otro modo se almacenará como texto plano. La opción descrita anteriormente se puede ver en la figura 44.

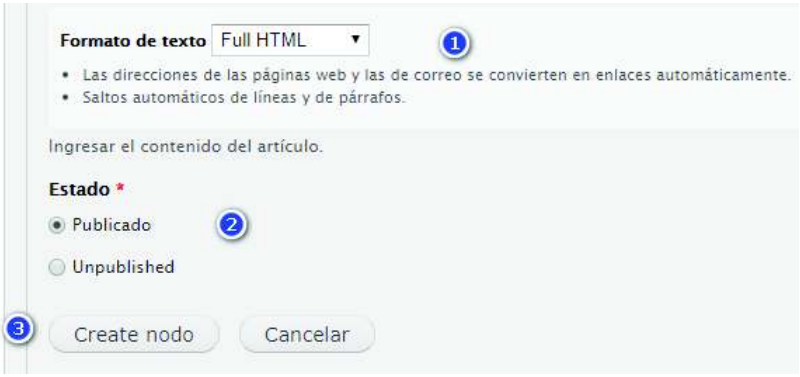


*Figura 44: Ingreso de contenido – Formato Full HTML*

Una vez ingresado el artículo se procede a crear el nodo “Capítulo”, a través de un botón de “Crear nodo”, que puede evidenciarse en la figura 45.


*Figura 45: Ingreso de contenido – Creación de Capítulo con Artículo como subclase*

Cuando se ha creado el nuevo nodo, puede evidenciar los capítulos ingresados mostrando la opción de mantenimiento del nodo al permitir las operaciones de editar y eliminar, dichas opciones se ilustran en la figura 46.

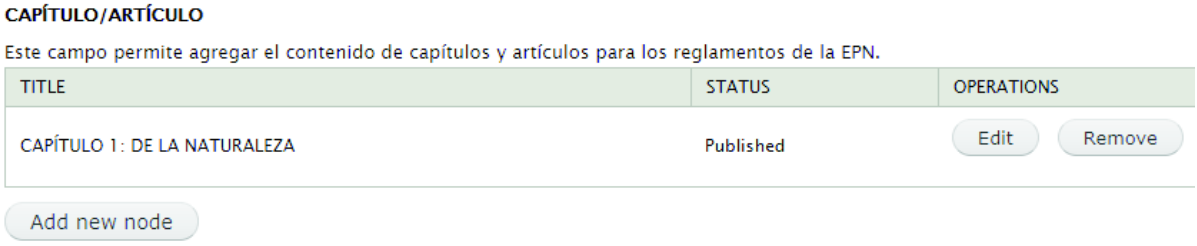


*Figura 46: Capítulos ingresados*

En la figura 47 se puede apreciar que en el formulario de ingreso del reglamento también se presenta la opción para ingresar la información correspondiente a la clase Disposiciones, empezando por las disposiciones generales, para la que pide nombre y descripción.


*Figura 47: Ingreso de contenido – Disposiciones Generales*

Una vez ingresada la información, se puede visualizar las disposiciones almacenadas que se presentan en el sitio Web, tal como lo ilustra la figura 48.


*Figura 48: Disposiciones Generales ingresadas*

El siguiente paso es el ingreso de la información de disposiciones transitorias, para lo cual también se pide ingresar el nombre y su descripción, los campos que piden ser ingresados se aprecian en la figura 49.

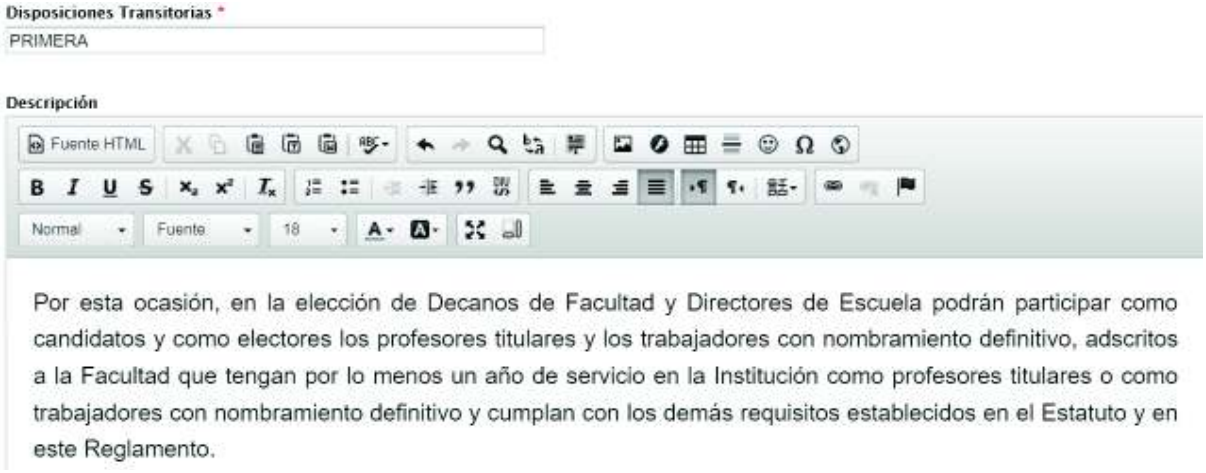


*Figura 49: Ingreso de contenido – Disposiciones Transitorias*

Para el caso de la disposición final no se pide ingresar un nombre, ya que se trata de una única disposición, por lo que únicamente pide el ingreso de la descripción de la disposición, esta característica se aprecia en la figura 50, donde solo existe el campo Descripción de la Disposición Final.

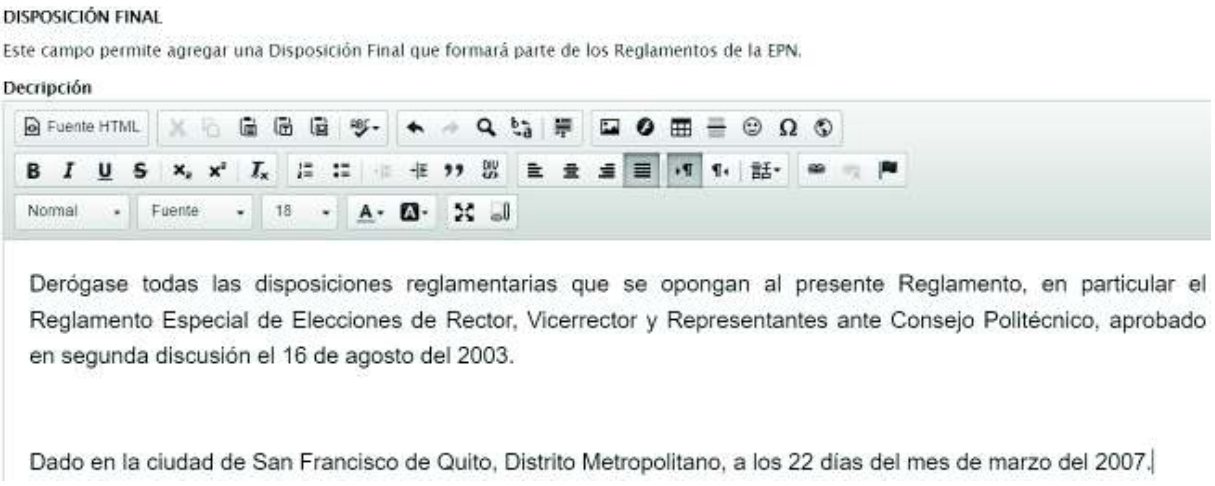


*Figura 50: Ingreso de contenido – Disposición Final*

Existe una opción que permite seleccionar la fecha de inicio de vigencia del reglamento, la cual presentará la fecha desde cuando entra en vigencia un reglamento; la figura 51 muestra el calendario que tiene el CMS para seleccionar la fecha mencionada.

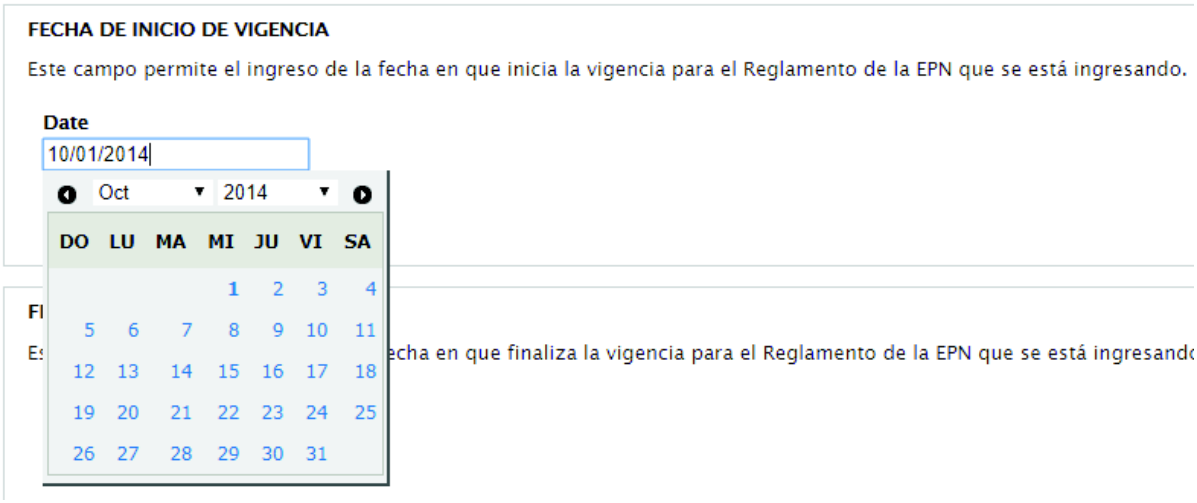


*Figura 51: Fecha de inicio de vigencia de Reglamento*

Existe una opción para seleccionar la fecha de fin de vigencia del reglamento, que nos va a indicar cuando dejó de estar vigente un reglamento. En la figura 52, se aprecia que el CMS hace uso de un calendario para la selección de la fecha mencionada,

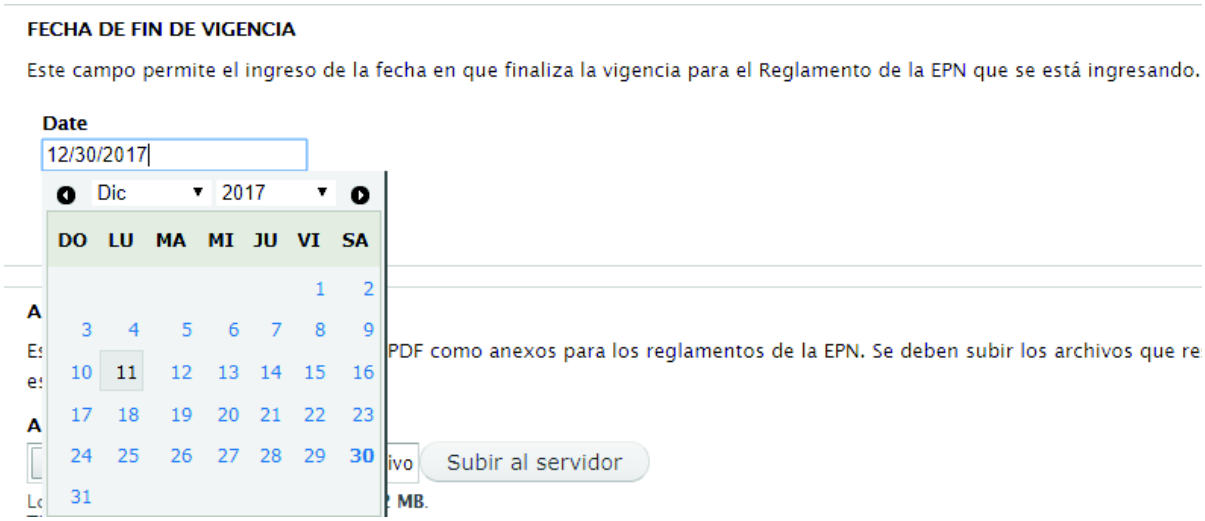


*Figura 52: Fecha de fin de vigencia de Reglamento*

Como parte del tipo de contenido anexo, se permite la carga de un archivo en formato PDF, para que pueda ser descargado por el usuario final; esta opción se aprecia en la figura 53.


*Figura 53: Ingreso de contenido – Carga de anexo*

El contenido ingresado se puede verificar como un nodo creado con la opción de mantenimiento Eliminar, esto se puede evidenciar en la figura 54, donde se visualiza el nombre del archivo PDF que ha sido almacenado en el CMS.

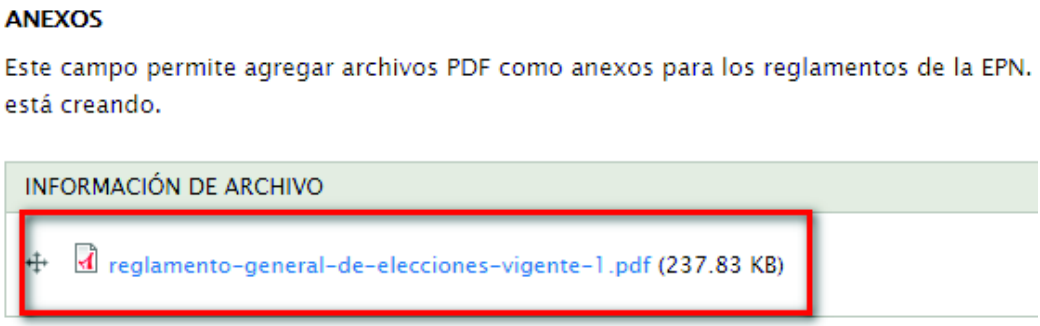


*Figura 54: Anexo ingresado*

La característica que maneja Drupal, al ser un administrador de contenido, es que presenta disponible la opción para agregar comentarios a las revisiones que se realicen sobre el tipo de contenido, lo cual nos permite manejar un historial de los cambios que pueden darse sobre un reglamento específico, como se aprecia en la figura 55.

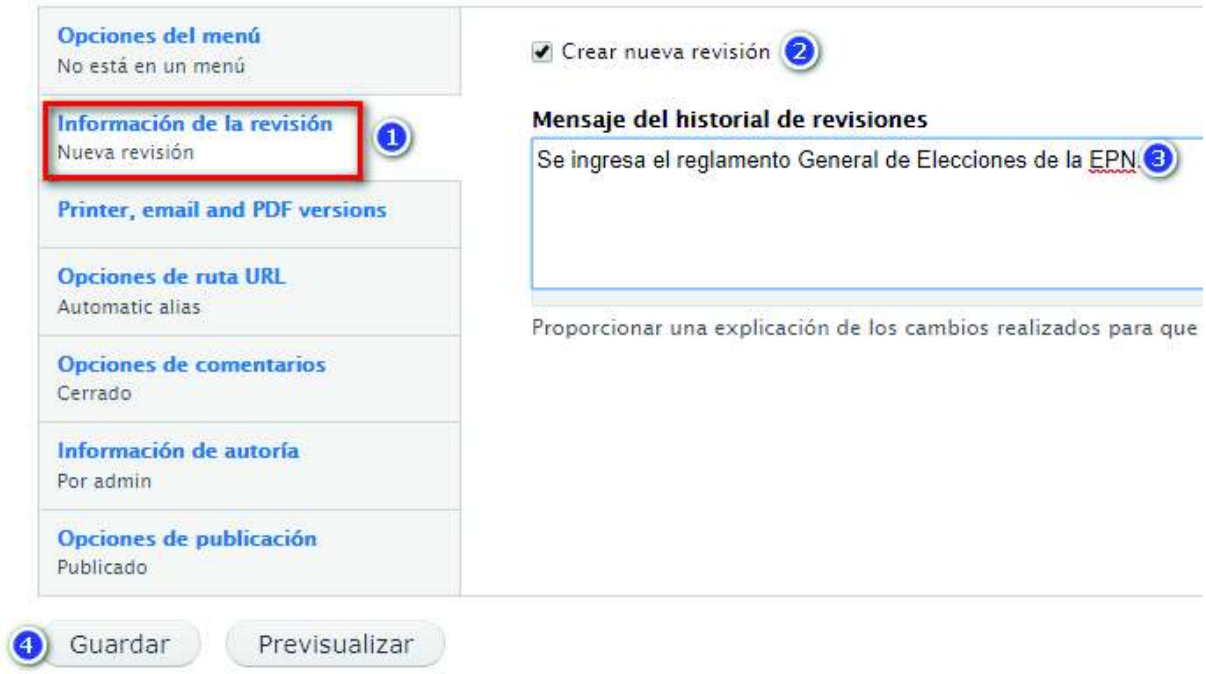


*Figura 55: Ingreso de contenido – Razón de modificación*

Cuando se concluye con los pasos señalados anteriormente vamos a contar con un reglamento almacenado en nuestro CMS; adicional al ingreso de contenido es necesaria la indexación del contenido, para que se formen las tripletas en la base RDF y pueda ser consultada toda la información a través de consultas SPARQL.

### 3.1.9. Implementación de consultas SPARQL

Para poder ejecutar las consultas SPARQL que han sido propuestas para poder verificar que el contenido tiene datos enlazados a través de RDF, se lo debe realizar a través del *endpoint* del sitio Web, en donde se puede determinar la forma que la que se va a presentar el resultado de la consulta, sea en un archivo que se lo descarga en diversos formatos, o presentándolos como una tabla HTML

Las consultas planteadas se ejecutaron de manera correcta pudiendo visualizar su resultado en la pantalla en a través de una tabla HTML, en donde se puede apreciar una tripleta RDF como resultado, obteniendo el sujeto, predicado y objeto.

En la figura 56 se muestra en el *endpoint* para consultas SPARQL que cuando se abre presenta una consulta por defecto, la cual presenta el resultado mostrado en la figura 57.


*Figura 56: Endpoint para consultas SPARQL con consulta predefinida*

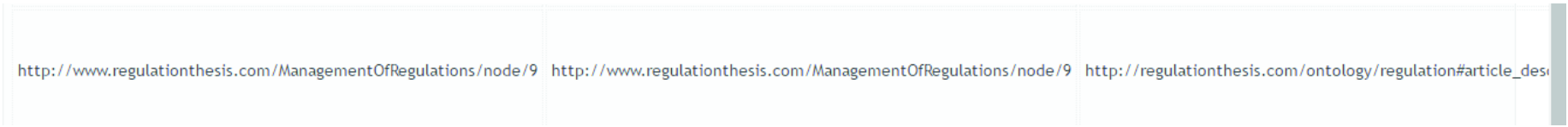


*Figura 57: Resultados de la consulta SPARQL predefinida*

***Consultar reglamentos ingresados***

Para realizar la consulta de los reglamentos que se encuentran almacenados en nuestro sitio Web, se han definido los siguientes prefijos:

- **rdf**: Hace referencia al espacio de nombres (*namespaces*) para RDF.
- **node**: Hace referencia a la URL en la que se encuentran almacenados todos los reglamentos de la EPN; es importante conocer que con la ayuda de este prefijo podremos especificar el nombre de cualquiera de ellos que exista en el sitio Web, por lo que se debe ingresar el nombre tal y como se muestra en la URL del sitio.
- **regt**: Nos permite hacer consultas en base a la ontología desarrollada, en este caso especificaremos la URI usada para la importación de la ontología.

La consulta formulada es la siguiente:

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix node: <http://regulationthesis.com/ManagementOfRegulations/>
prefix regt: <http://regulationthesis.com/ontology/regulation#>

SELECT ?reglamentos
WHERE {
  ?reglamentos rdf:type regt:Regulation .
}
```

El resultado de la consulta formulada para consultar los reglamentos ingresados se presenta en la figura 58.


*Figura 58: Resultado de consulta Obtener Reglamentos*

***Consultar información de un reglamento específico***

La siguiente consulta nos permitirá traer la información del Reglamento General de Elecciones de la Escuela Politécnica Nacional

```
prefix node: <http://regulationthesis.com/ManagementOfRegulations/content/>
prefix regt: <http://regulationthesis.com/ontology/regulation#>

SELECT ?predicados ?objetos
WHERE {
  node:reglamento-general-de-elecciones ?predicados ?objetos .
}
```

El resultado de la consulta formulada para consultar información de un reglamento específico se presenta en la figura 59.

| predicados | objetos |
|---|---|
| http://www.w3.org/1999/02/22-rdf-syntax-ns#type | http://regulationthesis.com/ontology/regulation#Regulation |
| http://purl.org/dc/terms/date | 2017-12-10T19:32:15-05:00 |
| http://purl.org/dc/terms/created | 2017-12-10T19:32:15-05:00 |
| http://purl.org/dc/terms/modified | 2017-12-10T19:32:15-05:00 |
| http://rdfs.org/sioc/ns#num_replies | 0 |
| http://regulationthesis.com/ontology/regulation#regulation_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/node/5 |
| http://regulationthesis.com/ontology/regulation#regulation_is_composed_by | capitulosArticulos |
| http://regulationthesis.com/ontology/regulation#body_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/content/cap%C3%ADtulo-1-de-la-naturaleza |
| http://regulationthesis.com/ontology/regulation#body_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/content/cap%C3%ADtulo-2-de-la-convocatoria |
| http://regulationthesis.com/ontology/regulation#provision_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/content/primera |
| http://regulationthesis.com/ontology/regulation#provision_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/content/segunda |
| http://regulationthesis.com/ontology/regulation#provision_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/content/primera-0 |
| http://regulationthesis.com/ontology/regulation#provision_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/content/segunda-0 |
| http://regulationthesis.com/ontology/regulation#provision_is_composed_by | http://regulationthesis.com/ManagementOfRegulations/node/12 |
| http://regulationthesis.com/ontology/regulation#validity_init_date | 1412139600 |
| http://regulationthesis.com/ontology/regulation#validity_finish_date | 1512882000 |
| http://regulationthesis.com/ontology/regulation#regulation_title | REGLAMENTO GENERAL DE ELECCIONES |

*Figura 59: Resultado de consulta Obtener Información de un Reglamento*

***Consultar los capítulos de un reglamento***

Esta consulta nos permite traer todos los capítulos asociados al reglamento.

```
prefix node: <http://regulationthesis.com/ManagementOfRegulations/content/>
prefix regt: <http://regulationthesis.com/ontology/regulation#>

SELECT ?capitulos
WHERE {
  node:reglamento-general-de-elecciones regt:body_is_composed_by ?capitulos .
}
```

El resultado de la consulta formulada para consultar los capítulos de un reglamento se presenta en la figura 60.

| capitulos |
| --- |
| http://regulationthesis.com/ManagementOfRegulations/content/cap%C3%ADtulo-1-de-la-naturaleza |
| http://regulationthesis.com/ManagementOfRegulations/content/cap%C3%ADtulo-2-de-la-convocatoria |

*Figura 60: Resultado de consulta Preámbulo de Reglamento*

***Consultar las disposiciones generales de un reglamento***

Esta consulta nos permite traer todas las disposiciones generales especificando la clase de la ontología a la cual pertenece.

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
prefix node: <http://regulationthesis.com/ManagementOfRegulations/content/>
prefix regt: <http://regulationthesis.com/ontology/regulation#>

SELECT ?disposicionesGenerales
WHERE {
  node:reglamento-general-de-elecciones regt:provision_is_composed_by
?disposicionesGenerales .
  ?disposicionesGenerales rdf:type regt:General .
}
```

El resultado de la consulta formulada para consultar las disposiciones generales de un reglamento con especificación de clase se presenta en la figura 61.

| disposicionesGenerales |
| --- |
| http://regulationthesis.com/ManagementOfRegulations/content/primera |
| http://regulationthesis.com/ManagementOfRegulations/content/segunda |

*Figura 61: Resultado de consulta Preámbulo especificando la clase*

## 3.2. Discusión

La solución desarrollada permitió la implementación de un sitio Web para el manejo de reglamentaciones usando tecnologías de Linked Data, en donde además toda la información publicada a través del sitio podrá ser consultada mediante el uso de SPARQL. Esto se logra debido a la implementación de una ontología propia que se adapta a la estructura actual de los reglamentos de la Escuela Politécnica Nacional, siendo nuestro caso de estudio el Reglamento General de Elecciones de la institución mencionada.

Existen muchas ontologías en la actualidad, las cuales son usadas para diferentes propósitos, pero a medida que el tiempo transcurre las necesidades aumentan y dichos vocabularios van creciendo o a su vez se crean nuevos. Es por eso que en base a los estudios realizados y tomando en cuenta los trabajos previos expuestos anteriormente nuestro trabajo aporta con la creación de una ontología denominada "regulationT" que está diseñada para adaptarse a la estructura que presentan los reglamentos de la EPN, la cual nos permite tener una definición formal de tipos, propiedades y relaciones entre entidades, además de delimitar el dominio de la información que va a manejar el sitio Web, siendo este únicamente un vocabulario legislativo, propio de Reglamentaciones. Esta ontología podrá ser usada y mejorada en base a las necesidades que puedan existir en un futuro.

Algo que se debe tomar en cuenta es que el diseño de ontologías no debe hacerse de manera experimental sin seguir algún procedimiento formal, pues puede tener como resultado una ontología que no incluya todos los conceptos, relaciones o instancias que se necesitan para resolver un problema en particular, razón por la cual hay que escoger una metodología de elaboración de ontologías. Para el desarrollo de este proyecto se escogió Methontology, ya que seguir cada una de sus tareas fue sencillo y abarcó todos los requerimientos para limitar el dominio de reglamentos de la Escuela Politécnica Nacional.

La integración de la ontología con la información se la realizó a través de RDFa que nos provee Drupal 7 y el que a su vez nos permite inyectar la semántica de los datos directamente sobre el código HTML [34]. Para el usuario final del sistema lo más importante es poder visualizar el contenido, mas no llegar a ver la estructura de la página Web presentada, pero esto se puede verificar por un usuario con conocimiento en HTML y con una herramienta que le permita hacerlo. Una alternativa es usar las herramientas de desarrollo que provee el navegador Google Chrome, con el fin de validar que la información ingresada en el sitio cuente con las etiquetas RDF respectivas para cada tipo de contenido que tiene la estructura del dominio de la información a través de la opción Inspeccionar de dicha herramienta, como

se puede apreciar en la figura 62, donde se inspecciona una sección de la página web y se aprecia su código HTML.

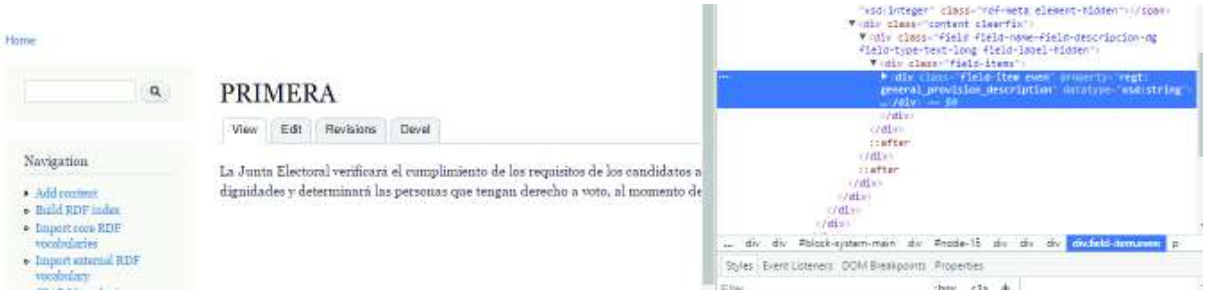


*Figura 62: Herramientas de desarrollador de Google Chrome*

Si nos fijamos bien el código HTML de la página presentada podemos observar que dentro del *div* que se está inspeccionando se tiene una propiedad, en la que se evidencia el prefijo "regt" de la ontología "regulationT" que provee de semántica al sitio Web, así como se muestra en la figura 63.

```
▼<div class="field-items">
  ▼<div class="field-item even" property="regt:general_provision_descripcion"
  datatype="xsd:string">
    ▶<p>…</p>
    </div>
  </div>
```

*Figura 63: Propiedad RDF de la ontología "regulationT" en el código HTML de la página*

De la misma manera se puede hacer una inspección a las demás secciones del código HTML de la página, donde se podrá evidenciar las propiedades del vocabulario que maneja el sitio Web, de acuerdo a la estructura que se planteó para todos los tipos de contenido de reglamentaciones.

Esta información acerca de las propiedades del vocabulario de los tipos de contenido por sí sola ya tiene un significado, porque si vemos la figura 64 la propiedad es "*general_provision_descripcion*", para nuestro lenguaje natural significa la descripción de una disposición general, cuyo contenido es "La Junta Electoral verificará el cumplimiento de los requisitos de los candidatos a las diferentes dignidades y determinará las personas que tengan derecho a voto, al momento de la convocatoria" [21].

### 3.2.1. Análisis Costo – Beneficio del uso de Software Libre

El costo que se incurrió en el desarrollo del proyecto se puede apreciar en las tablas 37, 38, 39, 40, 41 y 42 que están a continuación, donde se detalla los costos incurridos en el desarrollo.

La primera variable corresponde al tiempo incurrido por el equipo de desarrollo en el entendimiento de *Linked Data* y la investigación de las herramientas que se usaron, lo cual puede evidenciarse en la tabla 37 a continuación.

*Tabla 37: Costo de investigación previa*

| Miembro | Rol | Costo hora-hombre | Horas totales | Costo investigación |
|---|---|---|---|---|
| Felipe Borja | Desarrollador | 5.00 US$ | 120 h | 600.00 US$ |
| Andrés Cusme | Desarrollador | 5.00 US$ | 120 h | 600.00 US$ |

La segunda variable corresponde al desarrollo de la metodología usada, en la cual se encuentran los seis *sprints* que fueron realizados, que puede visualizarse en la tabla 38 a continuación.

*Tabla 38: Costo de desarrollo de sprints*

| Miembro | Rol | Costo hora-hombre | Horas totales | Costo desarrollo *Sprints* |
|---|---|---|---|---|
| Felipe Borja | Desarrollador | 5.00 US$ | 240 h | 1,200.00 US$ |
| Andrés Cusme | Desarrollador | 5.00 US$ | 240 h | 1,200.00 US$ |

La tercera variable a ser analizada es el costo de adquisición de licencias del software base posibles para el desarrollo del proyecto, variable que puede apreciarse en la tabla 39 donde se aprecia el costo del software de licencia libre usado y de su par con licencia pagada.

*Tabla 39: Costo herramientas para el desarrollo*

| Software | Tipo | Valor |
|---|---|---|
| OpenLink Virtuoso | Software Propietario | 99.99 US$ |
| Drupal 7 | Software Libre | 0.00 US$ |

OpenLink Virtuoso mantiene un canal directo para el soporte al cliente del producto, a través de la apertura de casos en el sitio oficial, mientras que para Drupal la comunidad se encarga de resolver el tema de soporte a través de preguntas abiertas en diferentes blogs. Hay que

considerar el tiempo que se invierte en obtener respuestas a dudas a través del soporte de la herramienta, lo cual traducido a un costo monetario puede verse en la tabla 40.

*Tabla 40: Costo hombre en obtención de Soporte*

| Software | Tarea | Costo hora-hombre | Tiempo por caso | Costo en 20 casos |
|---|---|---|---|---|
| OpenLink Virtuoso | Generar el caso | 5.00 US$ | 2 h | 200.00 US$ |
| Drupal | Navegar en blogs | 5.00 US$ | 2 h | 200.00 US$ |

Al realizar un resumen de los costos incurridos en el desarrollo, podemos obtener el valor total invertido con el uso de herramientas de software libre, reflejado en la tabla 41.

*Tabla 41: Valor total con herramientas de software libre*

| Variable | Valor |
|---|---|
| Investigación | 1,200.00 US$ |
| Desarrollo *sprints* | 2,400.00 US$ |
| Costo herramientas de desarrollo | 0.00 US$ |
| Costo soporte | 200.00 US$ |
| **Total** | 4,000.00 US$ |

En el caso de haber usado software propietario el costo hubiese sido el mostrado en la tabla 42 a continuación.

*Tabla 42: Valor total con herramientas de software propietario*

| Variable | Valor |
|---|---|
| Investigación | 1,200.00 US$ |
| Desarrollo *sprints* | 2,400.00 US$ |
| Costo herramientas de desarrollo | 99.99 US$ |
| Costo soporte | 200.00 US$ |
| **Total** | 4,099.99 US$ |

La diferencia monetaria en el uso de herramientas para el desarrollo de software libre no es muy grande comparada con herramientas de software propietario, pero al haber hecho uso de herramientas de software libre se obtuvo un poco de conocimiento en la estructura del software Drupal, ya que para resolver ciertas dudas en su implementación se pudo revisar su código fuente y entender cómo es su funcionamiento.

# CONCLUSIONES

## 3.3. Conclusiones

- A través de *Linked Data* se pudo llegar a tener los datos publicados a través de un sitio con información accesible desde cualquier otro sitio Web que maneje consultas SPARQL sin la necesidad de la exposición de servicios, sino únicamente teniendo en línea nuestro portal.
- El uso de la metodología *Scrum* hizo posible la conclusión exitosa del proyecto, debido a que permitió una gestión adecuada del desarrollo del proyecto, al tener varias iteraciones con entregables que aportaron valor y funcionalidad al sitio Web implementado.
- Las tecnologías de *Linked Data* permiten tener datos enlazados que al ser consultados no solo traen la información visible para un usuario final, sino que también llevan consigo la estructura semántica proporcionada a través de RDF.
- *Linked Data* al ser una serie de principios y tecnologías que permiten enlazar datos existentes en la Web y usar RDF, basado en la notación de tripletas de conocimiento, permite que el significado de la información sea explícito, con lo que es posible programar agentes inteligentes que consuman esta información.
- Una vez que la *Word Wide Web* llegue a tener todos sus datos enlazados se tendrá una base de conocimiento universal en línea, con lo que se podría llegar a consultar información de cualquier dominio de datos, a través de *SPARQL* sin importar la forma de implementación de los sitios Web.
- Una ontología propia para el manejo de un dominio de información sirve para delimitar de una manera correcta la estructura o vocabulario que se va a manejar, pues para cada tipo de contenido del sitio Web va a existir una propiedad que permita entender de mejor manera el significado de la información que almacena.
- El uso de administradores de contenido (CMS) como Drupal facilita el desarrollo de un sitio Web, ya que permite la creación de páginas sencillas que son estáticas, o de mayor complejidad como en este caso que se incluyó semántica al contenido ingresado.
- Los módulos que se acoplan al *Core* del CMS brindan la facilidad de agregar funcionalidad al sitio Web, ya que debido a la variedad de módulos se puede escoger los necesarios para el sitio Web deseado.
- Un CMS brinda facilidades para la creación de sitios Web, pero se identificó una desventaja en cuanto a la flexibilidad en la funcionalidad que entregan los módulos, debido a que es necesario trabajar de acuerdo a las prestaciones del módulo, mas no es posible estructurar el sitio de una manera diferente a la preestablecida.

- EL software libre presenta varias ventajas al momento de ser personalizado por el usuario, pero también presenta inconvenientes cuando se busca solución a problemas que se presentan en su uso, ya que el soporte no es tan eficiente y obtener una respuesta o solución a un problema puede tomar más tiempo que el que pudiera tomar si fuera una herramienta de software propietario, que en su mayoría presentan sitios oficiales de soporte que brindan soluciones y respuestas en tiempos más cortos sin la necesidad de indagar en blogs para obtener dicha solución.

## 3.4. Recomendaciones

- Para trabajos futuros con el CMS Drupal es importante recalcar que cada versión tiene sus propios módulos publicados por lo que hay que analizar la compatibilidad de los mismos con la versión del CMS ya que cada versión nueva de Drupal mejora características del CMS pero no siempre es compatible con versiones anteriores de los módulos.
- Se recomienda usar módulos de Drupal que se encuentren vigentes, es decir que tengan mantenimiento por parte de la comunidad de Drupal, pues existen varios módulos que aparentemente dotan de funcionalidad RDF al CMS pero que al no haber recibido continuidad han sido abandonados y presentan varias fallas que no suceden con módulos actualizados.
- El diseño de la ontología debe ser realizado tomando en cuenta una metodología, ya que se deben establecer las clases, sus relaciones, propiedades y demás elementos que permitan estructurar consultas SPARQL que accedan a cada parte de la información que existe en el sitio.
- Para crear una nueva ontología se recomienda adquirir un mayor conocimiento en el manejo del software Protégé, pues esta posee una función que permite evaluar previamente si una consulta SPARQL está bien estructurada.
- Las consultas SPARQL debe ser ejecutadas especificando el prefijo de la ontología a la cual se desea hacer referencia, debido a que un sitio Web puede manejar simultáneamente varias ontologías, de tal modo que el resultado de dicha consulta presente únicamente la información deseada.
- Para un trabajo futuro en el consumo de la información publicada como *Linked Data* se recomienda el uso de complementos en el lenguaje de programación Java, como Apache Jena, que posee API's para el manejo de SPARQL y RDF de forma nativa.
- Para el despliegue de este proyecto en un nuevo ambiente es importante seguir el anexo "Guía de instalación y configuración Drupal 7", ya que se describen paso a paso las configuraciones que se realizaron para tener integrado el CMS al manejo de información con *Linked Data*.

- Nuestro trabajo fue implementado en un servidor Linux de prueba, por lo que se recomienda que este proyecto sea implementado en los servidores de la Escuela Politécnica Nacional y que de esta manera pueda ser usado por toda la comunidad Politécnica cuando se la requiera.

# 4. REFERENCIAS BIBLIOGRÁFICAS


[1] OpenLink Software, «OpenLink Virtuoso,» 2015. [En línea]. Available: https://virtuoso.openlinksw.com/. [Último acceso: 01 08 2017].

[2] Drupal Community, «Drupal,» 2017. [En línea]. Available: https://www.drupal.org/. [Último acceso: 01 08 2017].

[3] W3C, «Guía Breve de Linked Data," Linked Data,» 2017. [En línea]. Available: http://www.w3c.es/Divulgacion/GuiasBreves/LinkedData. [Último acceso: 31 08 2017].

[4] W3C, «Guía Breve de Web Semántica,» 2017. [En línea]. Available: https://www.w3c.es/Divulgacion/GuiasBreves/WebSemantica. [Último acceso: 20 08 2017].

[5] L. Codina, Web semántica y sistemas de información documental, España: Ediciones Trea, S.L., 2009.

[6] W3C, «Resource Description Framework (RDF),» 25 02 2014. [En línea]. Available: https://es.wikipedia.org/wiki/Resource_Description_Framework. [Último acceso: 08 08 2017].

[7] W3C, «Resource Description Framework (RDF): Concepts and Abstract Syntax,» 10 02 2004. [En línea]. Available: https://www.w3.org/TR/rdf-concepts/#section-triples. [Último acceso: 08 08 2017].

[8] J. Tauberer, «What is RDF and what is it good for?,» 01 2008. [En línea]. Available: https://github.com/JoshData/rdfabout/blob/gh-pages/intro-to-rdf.md. [Último acceso: 08 08 2017].

[9] W3C, «Lenguaje de Ontologías Web (OWL),» 2004. [En línea]. Available: https://www.w3.org/2007/09/OWL-Overview-es.html. [Último acceso: 08 08 2017].

[10] W3C, «OWL Web Ontology Language Use Cases and Requirements,» 10 02 2004. [En línea]. Available: https://www.w3.org/TR/webont-req/#onto-def. [Último acceso: 08 08 2017].

[11] M. Hallo, S. Luján y J. Trujillo, «Transforming Library Catalogs into Linked Data,» de *ICERI2014 Proceedings*, Seville-Spain, 2014.

[12] M. Hallo, S. Luján y A. Maté, «Data model for storage and retrieval of legislative documents in Digital Libraries using Linked Data,» de *EDULEARN15 Proceedings*, Barcelona-Spain, 2015.

[13] A. A. Peri, «SIDRA: XML en la gestión y explotación de la documentación jurídica,» *SCIRE,* vol. 15, nº 1, pp. 111-1244, 2009.

[14] IBM, «Tamino v10.1,» Global Solution, [En línea]. Available: http://www-304.ibm.com/partnerworld/gsd/solutiondetails.do?solution=4009&expand=true&lc=en. [Último acceso: 12 11 2017].

[15] U. d. Alcalá, «Máster Dirección Proyectos,» [En línea]. Available: http://www.uv-mdap.com/programa-desarrollado/bloque-iv-metodologias-agiles/metodologias-agiles-vs-tradicionales/. [Último acceso: 16 09 2017].

[16] O. A. Pérez, «Cuatro enfoques metodológicos para el desarrollo de Software: RUP - MSF - XP - SCRUM,» *Inventum,* vol. 1, nº 10, pp. 64-78, 2011.

[17] K. Schwaber, Agile Project Management with Scrum, United States: Microsoft Press, 2004.

[18] R. Herranz, Despegar con Scrum, España: Lulu, 2016.

[19] SCRUMstudy, Una guía para el cuerpo de conocimiento de SCRUM (Guía SBOK), Phoenix-United States: VMEdu, Inc, 2016.

[20] PMOinformatica, «Plantillas Scrum: historias de usuario y criterios de aceptación,» 1 Octubre 2012. [En línea]. Available: http://www.pmoinformatica.com/2012/10/plantillas-scrum-historias-de-usuario.html. [Último acceso: 10 08 2017].

[21] Escuela Politécnica Nacional, Reglamento General de Elecciones, Quito: Offset-EPN, 2014.

[22] A. De Winne, «Decoupling Drupal,» de *Drupal Camp Toronto*, Toronto, 2014.

[23] Comunic Art, «Tutorial Drupal,» 2017. [En línea]. Available: http://www.cursosdrupal.com/content/modulos. [Último acceso: 02 08 2017].

[24] D. Lenat y R. Guha, Building Large Knowledge-Based Systems: Representation and Interference in the Cyc project, Massachusetts-United States: Addison-Wesley, 1990.

[25] M. Uschold y K. Martin, «Towards a Methodology for Building Ontologies,» de *Workshop on Basic Ontological Issues in Knowledge Sharing*, Cambridge-United Kingdom, 1995.

[26] M. Gruninger, «The logic of enterprise modeling,» de *Reenginering the Enterprise*, London-United Kingdom, Chapman & Hall, 1995, pp. 83-98.

[27] A. Bernaras, I. Laresgoiti y J. Corera, «Building and Reusing Ontologies for Electrical,» de *In Proceedings of the European Conference on Artificial Intelligence*, Budapest-Hungary, 1996.

[28] M. Fernández, A. Gómez-Pérez y N. Juristo, «METHONTOLOGY: From Ontological Art to Ontological Engineering,» de *Workshop on Ontological Engineering*, California-United States, 1997.

[29] B. Swartout, R. Patil, K. Knight y T. Russ, «Toward distributed use of large-scale,» de *In AAAI-97 Spring Symposium Series on Ontological Engineering*, California-United States, 1997.

[30] J. A. Guzman, M. López y I. Durley, «Metodologías y métodos para la construcción de ontologías,» *Scientia et Technica,* vol. 2, nº 50, pp. 133-140, 2012.

[31] O. Corcho, M. Fernández-López, A. Gómez-Pérez y A. López-Cima, «Construcción de ontologías legales con la metodología,» de *Law and the Semantic Web. Legal Ontologies, Methodologies, Legal Information Retrieval, and Applications*, Berlin-Germany, Springer-Verlag, 2005, pp. 142-157.

[32] Stanford Center for Biomedical Informatics Research, «Protégé,» 2016. [En línea]. Available: https://protege.stanford.edu/. [Último acceso: 01 07 2017].

[33] Stanford Center for Biomedical Informatics Research, «Protégé,» 2016. [En línea]. Available: https://protege.stanford.edu/. [Último acceso: 02 08 2017].

[34] Drupal, «RDF/RDFa (D7),» 05 08 2013. [En línea]. Available: https://www.drupal.org/node/574624. [Último acceso: 08 08 2017].

# 5. ANEXOS

ANEXO 1: Guía de instalación y configuración Drupal 7.

ANEXO 2: Manual de usuario.

ANEXO 3: Glosario de Términos.