UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto de Fin de Grado en Ingeniería en Tecnologías de la Información

DESARROLLO DE UN SISTEMA DE ETIQUETADO Y CONSULTA SEMÁNTICOS PARA USUARIOS DE LA UNED

MIGUEL EXPÓSITO MARTÍN

Dirigido por: JOSÉ LUIS FERNÁNDEZ VINDEL

Co-dirigido por: RAFAEL MARTÍNEZ TOMÁS

Curso: 2017-2018: 2ª Convocatoria

DESARROLLO DE UN SISTEMA DE ETIQUETADO Y CONSULTA SEMÁNTICOS PARA USUARIOS DE LA UNED

Proyecto de Fin de Grado de modalidad oferta específica

Realizado por: Miguel Expósito Martín

Dirigido por: José Luis Fernández Vindel

Co-dirigido por: Rafael Martínez Tomás

Tribunal calificador

Presidente: D/Dª.

Secretario: D/Dª.

Vocal: D/Dª.

Fecha de lectura y defensa:

Calificación:

Agradecimientos

A mi familia, por soportarme.

Resumen

El presente proyecto tiene como objeto el desarrollo de un sistema de *playground* para SPARQL y RDF que habrá de servir como herramienta educativa de introducción a las tecnologías de la Web Semántica. Su propósito es facilitar una interfaz de uso sencilla para que usuarios profanos puedan introducirse en este tipo de tecnologías a un nivel básico, permitiendo la realización de actividades académicas sobre manejo sencillo de grafos así como sobre consultas SPARQL al conjunto de datos de trabajo o a *endpoints* externos.

Si bien es cierto que la Web Semántica alcanza hoy en día un estado de madurez razonablemente avanzado, el ritmo de cambio de las tecnologías de *Frontend* en JS es, cuanto menos, vertiginoso. Unido a ello, se tiene que la mayor parte de las implementaciones de componentes de la Web Semántica han sido desarrolladas en tecnologías de *backend* como *Java* (*Apache Jena*) o *Python* (*rdflib*), no existiendo aún estándares o implementaciones maduras puramente en JS.

Se abordan, por tanto, tres retos: la necesidad de conseguir un producto sencillo y fácilmente utilizable por usuarios no expertos, la dificultad para encontrar componentes maduros que aúnen Web Semántica con *frontend* y la conveniencia de apostar por un *framework* de desarrollo en JS estable y con una curva de aprendizaje suave.

Para dar solución a estos problemas, se ha optado por desarrollar una *Single Page Application* (SPA) con Vue.js (un framework JS que se jacta de aglutinar las mejores características de Angular y React, sus principales competidores) e integrarlo con las implementaciones recomendadas por el grupo de trabajo de bibliotecas JS rdfjs y con una plataforma de consultas para la web flexible y modular. El sistema está planteado para ejecutarse en un sencillo navegador contemporáneo, descargándose a través de un simple servidor web (siendo este la única infraestructura necesaria para su distribución).

Dado el alto nivel de incertidumbre existente en la implementación de las necesidades del proyecto, se ha optado por utilizar un enfoque metodológico incremental e iterativo basado en Extreme Programming y apoyado sobre tableros Kanban, lo que ha permitido una mejor organización y evolución del proyecto.

El resultado final ofrece un producto básico de introducción a las tecnologías de la Web Semántica y permite pensar en un desarrollo del mismo tan ambicioso como las propias necesidades de los equipos docentes, dado el estado de madurez actual del frontend web.

Abstract

<This project ...>

Índice general

Índice de figuras

?ndice de tablas

Capítulo 1

Introducción general y objetivos

<Intro ... >

1.1. Motivación y objetivos

<Ejemplo biblio Eckel (2003); Gamma et al. (2005)>

<Ejemplo referencia a sección 1.2 y a subsección de otro capítulo, que en este caso es un anexo A.1.1. **Para que funcionen correctamente las referencias a otros capítulos al exportar a pdf, debe estar abierto el archivo maestro, abrir/editar el archivo a exportar desde él y exportar desde el maestro**.>

<Ejemplo referencia a figura 1.1>

<Ejemplo referencia a tabla 1.1>

<Ejemplo de nomenclatura (POO)>

1	2	3	4
aaaaaa	aaaaaa	aaaaaa	aaaaaa
aaaaaa	aaaaaa	aaaaaa	aaaaaa

Cuadro 1.1: Ejemplo de tabla

Figura 1.1: Ejemplo de figura con un pie largo para mostrar el uso del ítem del menú de Lyx Insertar->Título breve. La utilidad del título breve se aprecia en la lista de figuras, es ahí donde aparece, si no, aparecería todo el pie en la lista, también es conveniente para títulos de capítulos secciones y demás largo que ocupen más de una línea en la lista de índice correspondiente.(Para insertar salto de línea en pie o en enumeración Ctrl-Enter)
La figura o tabla hay que insertarla dentro de un flotante.
Siempre hay que Insertar->Etiqueta de la figura (fig:Ejemplo-de-figura), tabla etc, para referenciarla desde el texto.

<Ejemplo de listado de código, ver listado 1.1, lo mejor para utilizarlo es copiar el recuadro y cambiar el código contenido, así conservaremos las opciones que se han establecido, como: mostrar números de línea, quebrar líneas largas. etc. Para ver las opciones y poder modificarlas click derecho sobre el listado y elegir Configuración. Para cambiar el listado contenido copiar del IDE el trozo de código que se desee y pegarlo dentro del listado con Ctrl+May+v, en el menú de Lyx Editar->Pegado especial->Texto simple.>

1.2. Estructura de la memoria

La memoria de esta proyecto se estructura en los siguientes capítulos:

1. Introducción general y objetivos
2. <añadir los demás capítulos>
3. Conclusiones y trabajos futuros

<Comentar los capítulos>

Listado de código 1.1: Código de edit

```
1     public final boolean edit() {
2
3         ArrayList<String> campos = cargaCampos();
4
5         Scanner in = new Scanner( System.in );
6
7     /* La variable resp sera true si el usuario acepta la edicion
8      */
9
10    boolean resp = false;
11        int numero_total_opciones = campos.size() +
              opciones_no_campos;
12        System.out.println("\n" + toString()); // muestra el
              registro
13
14        editLoop: while (true) {
15            editMensaje();
16            System.out.println("Seleccione un numero (1-" +
                  numero_total_opciones + "):  ");
17            int opcion;
18            if ( in.hasNextInt() ) {
19                opcion = in.nextInt();
20                in.nextLine();
21            } else {
22                System.out.println("\nOPCION NO VALIDA. POR FAVOR
                      , INTRODUZCA UN NUMERO.");
23                in.nextLine();
24                continue;
25            }
26            campos = procesaOpcion(campos, opcion, in);
27            if(salir) {
28                if(aceptar)
29                    resp = true;
30                salir = false;
31                aceptar = false;
32                break editLoop;
33            }
34
35        }
36        return resp;
37    }
```

Capítulo 2

Fundamentos teóricos y estado del arte

2.1. Fundamentos teóricos

<....>

2.1.1. Computación evolutiva

<....>

2.1.1.1. Breve historia

<....>

2.1.1.2. Tipos de computación evolutiva

<....>

2.1.2. Algorítmos genéticos

<....>

2.1.2.1. Características

<....>

2.1.2.2. Elementos de un algoritmo genético

<....>

2.1.2.3. Parámetros

<....>

2.2. Estado del arte

<....>

2.2.1. Visión general

<....>

2.2.2. Tendencias

<....>

2.2.2.1. Mecanismos autoevolutivos

<....>

2.2.2.2. Algorítmos meméticos

<....>

2.2.2.3. Computación evolutiva y robótica

<....>

2.2.2.4. Enjambre de partículas

<....>

2.2.3. Aplicaciones

<....>

2.2.4. El futuro

<....>

Capítulo 3

Conclusiones y trabajos futuros

3.1. Conclusiones

<....>

3.2. Trabajos futuros

<....>

Bibliografía

Eckel, B. (2003). *Thinking in Patterns: Problem-Solving Techniques using Java*. http://mindview.net/Books/TIPatterns/.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (2005). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Boston, MA.

Anexo A

<Título Anexo A>

<...>

A.1. <Primera sección anexo>

A.1.1. <Primera subsección anexo>