Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Universidad
Zaragoza
1542

Trabajo de Fin de Grado

Análisis, diseño e implementación de un sistema de extracción de información orientado a la generación de infoboxes a partir de un archivo documental

Álvaro Juan Ciriaco

Director: Ángel Luis Garrido Marín
Ibercentro Media Consulting & Services S.L

Codirector: Carlos Bobed Lisbona
Universidad de Rennes 1, Francia

Ponente: Eduardo Mena Nieto
Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

Grado en Ingeniería Informática
Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza

Junio de 2018

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. ÁLVARO JUAN CIRIACO,

con nº de DNI 17765629S en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) GRADO, (Título del Trabajo) ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE EXTRACCIÓN DE INFORMACIÓN ORIENTADO A LA GENERACIÓN DE INFOBOXES A PARTIR DE UN ARCHIVO DOCUMENTAL,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 26 DE JUNIO DE 2018

Fdo: ÁLVARO JUAN CIRIACO

Análisis, diseño e implementación de un sistema de extracción de información orientado a la generación de infoboxes a partir de un archivo documental

RESUMEN

Hoy en día, el *Heraldo de Aragón* dispone de una gran cantidad de información generada diariamente a través de todas las noticias e imágenes que dispone. Toda esta información tiene que ser seleccionada, clasificada y archivada, con la finalidad de extraer información de interés para los redactores y periodistas por el departamento de documentación.

En estos momentos, el departamento de documentación no dispone de los mecanismos de síntesis y extracción de información necesarios para facilitar el trabajo. En un primer momento, disponían de un prototipo básico de un sistema que, usando plantillas, permitía al departamento de documentación generar fichas sencillas sobre las distintas entidades del sistema, con la finalidad de facilitar la búsqueda de información importante sobre estas.

Este trabajo de fin de grado es el proyecto encargado de mejorar el primer prototipo no funcional. Debido a que es un proyecto de gran tamaño y repercusión en la empresa, el trabajo se ha tenido que dividir en tres grandes bloques.

1. El primero de estos bloques consiste en el análisis del primer prototipo no funcional. Gracias a este primer análisis, se marcan las pautas y mejoras a seguir para que el sistema final sea realmente útil para el departamento de documentación, marcando los objetivos que tendrá que cumplir la aplicación final.
2. El segundo bloque corresponde a la fase de diseño del nuevo sistema, teniendo en cuenta los objetivos marcados en el primer bloque. Se lleva a cabo el estudio de los procesos necesarios y el modelo de datos necesario, junto al diseño de la interfaz y los diagramas que modelen el funcionamiento final del sistema.
3. Y por último, el tercer bloque corresponde a la implementación de un sistema de extracción de información orientado a la generación de *infoboxes* a partir de un archivo documental.

Gracias a este nuevo sistema, el departamento de documentación puede, a partir de una entidad, asociarla a una de las plantillas del sistema y obtener información de la base de datos local, DBpedia y la Web para crear una ficha simulando un *infobox*. Así pues todo el conocimiento extraído de las distintas fuentes de información puede ser accesible de una forma sencilla.

Agradecimientos

A Ángel, por ayudarme con todas las dudas y orientarme a lo largo del proyecto y a Carlos, por toda la ayuda prestada durante la fase de diseño del grafo.

A Eduardo, por sus correcciones y su ayuda durante el trabajo.

A mis padres y a toda mi familia por todo el apoyo recibido a lo largo de todos estos años.

Índice general

1. Introducción **1**
1.1. Contexto 2
1.2. Motivación 2
1.3. Objetivos 3
1.4. Tecnología utilizada 4
1.5. Organización de la memoria 4

2. Análisis del sistema **5**
2.1. Sistema inicial 5
2.2. Catálogo de requisitos del sistema 6
2.3. Identificación de los actores 7
2.4. Diagramas UML de análisis 7

3. Arquitectura del sistema **8**
3.1. Arquitectura global 8
3.1.1. Arquitectura de *EMMA* 8
3.1.2. Arquitectura de *NEREA y POIROT* 9
3.1.3. Arquitectura del sistema completo 10
3.2. Arquitectura *SOPHIE* 11

4. Diseño e implementación del sistema **15**
4.1. Procesos 15
4.2. Datos 17
4.2.1. Modelo lógico 18
4.2.2. Implementación con Virtuoso 20
4.3. Interfaces de Usuario 21
4.4. Diagramas de diseño: Diagrama de componentes 24
4.5. Desarrollo del sistema 25

5. Conclusiones **27**
5.1. Resultados obtenidos 27

5.2. Trabajo futuro 29
5.3. Estimación de tiempo empleado 29
5.4. Valoración personal 30

A. Manuales de usuario y configuración **35**
A.1. Manual de usuario de SOPHIE 35
A.1.1. Crear una plantilla 35
A.1.2. Modificar una plantilla 38
A.1.3. Eliminar una plantilla 40
A.1.4. Crear una ficha 41
A.1.5. Modificar una ficha 43
A.1.6. Eliminar una ficha 45
A.1.7. Rellenar una ficha 46
A.1.8. Visualizar una ficha 48
A.1.9. Carga conjunta 50
A.2. Manual de instalación de *SOPHIE* 51
A.2.1. Requisitos de la instalación 51
A.2.2. Instalación de la aplicación 52
A.2.3. Carga de datos iniciales 52
A.2.4. Depuración de la puesta en marcha 53
A.3. Instalación y configuración de *Virtuoso* 54
A.3.1. Requisitos previos 54
A.3.2. Manual de instalación 54
A.3.3. Manual de configuración del servicio 57

B. Análisis y diseño del sistema **59**
B.1. Diagramas de Secuencia de la aplicación 59
B.2. Caso de Uso de la aplicación 61
B.3. Diagramas de Actividad de la aplicación 62
B.4. Esquema de una ficha 63

C. Desarrollo del sistema **64**
C.1. Estructura 64
C.2. Extracción de información 66
C.2.1. Google 66
C.2.2. Wikipedia 68
C.3. Correción y verificación de errores 69
C.4. Sistemas gestores de base de datos NoSQL 71
C.4.1. Virtuoso OpenLink 73
C.4.2. Stardog 73
C.4.3. Elección de un SGBD NoSQL 74

C.5. Diseño futuro 75

D. Ampliaciones **77**
D.1. Trabajo futuro 77
D.1.1. Selección automática de la plantilla 77
D.1.2. Inferencia de datos 79

Índice de figuras

2.1. Esquema simple del sistema. 5

3.1. Arquitectura del repositorio local de datos - EMMA. 9
3.2. Arquitectura de NEREA: sistema de reconocimiento y desambiguación de entidades. 10
3.3. Esquema general de la arquitectura del sistema. 11
3.4. Arquitectura de SOPHIE: sistema encragado de la generación de las fichas. 12
3.5. Esquema que muestra el funcionamiento del sistema para seleccionar una plantilla automáticamente. 13

4.1. Diagrama de componentes del sistema. 15
4.2. Diagrama de clases que modela el modelo de datos del proyecto. . . 17
4.3. Diagrama que modela la ontología del sistema. 18
4.4. Ejemplo de tripletas para la creación de la plantilla "Político". . . . 20
4.5. Presentación en la primera versión del sistema. 21
4.6. Presentación de la información de una ficha en la última versión del sistema. 22
4.7. Cabecera de una ficha en la versión futura del sistema. 23
4.8. Diagrama de componentes de la aplicación. 24

5.1. Cronograma temporal. 30

A.1. Estado del directorio \database antes de la instalación de *Virtuoso*. 55
A.2. Creación de la variable de entorno para *Virtuoso*. 56
A.3. Creación del servicio base de *Virtuoso*. 56
A.4. Panel de configuración avanzada de *Virtuoso*. 57

B.1. Diagrama de secuencia que representa la simulación automática de una ficha a través del servicio de SOPHIE. 59
B.2. Diagrama de secuencia que representa la selección automática de plantillas para generar las fichas de una *entidad*. 60

B.3. Diagrama de secuencia que representa la verificación y corrección de una ficha. 60
B.4. Diagrama de casos de uso de la aplicación de escritorio. 61
B.5. Diagrama de actividad que representa el proceso completo de creación de una ficha. 62
B.6. Resumen de una ficha dentro del sistema. 63

C.1. Plantillas Persona y la herencia de las plantillas Político y Deportista. 64
C.2. Plantilla "Tema". 65
C.3. Ejemplo de obtención de la información de las oficinas de Google en Sydney. 66
C.4. Infobox de Wikipedia sobre Cani, jugador de fútbol del Real Zaragoza. 68
C.5. Ejemplo de expresión regular para obtener el infobox de un deportista en la Wikipedia. 69
C.6. Código de ejemplo para la extracción de la información sobre Lionel Messi en la Wikipedia. 69
C.7. Ejemplo de error en la entidad José Luis Rodríguez Zapatero. El campo nombre completo es erróneo. 70
C.8. Cuerpo de una ficha en la versión futura del sistema. 75
C.9. Cuerpo de una ficha en la versión futura del sistema. 75
C.10. Cuerpo de una ficha en la versión futura del sistema. 76
C.11. Parte final de una ficha en la versión futura del sistema. 76

D.1. Esquema simple del sistema. SOPHIE se encarga de la generación de las fichas para las entidades, las cuales se obtienen a través de NEREA y su catálogo de entidades. 78
D.2. Esquema que muestra el funcionamiento del sistema para seleccionar una plantilla automáticamente. 79
D.3. Ontología de Persona utilizada en el sistema 81
D.4. Ejemplo complejo de relación entre personas. 81

Índice de tablas

2.1. Requisitos funcionales 6
2.2. Requisitos no funcionales 6

5.1. Iteración de prueba para comprobar el porcentaje de acierto de la aplicación 28
5.2. Horas empleadas en cada tarea 30

A.1. Tareas para la depuración de la aplicación 53

C.1. Ficha de características de *Virtuoso* 73
C.2. Ficha de características de *Stardog* 74

Capítulo 1

Introducción

A lo largo de este documento se va a detallar en profundidad el análisis de *SOPHIE*. El proyecto SOPHIE está basado en la generación de *infoboxes*[16] a través de la obtención de datos de calidad de diversas fuentes, tanto locales como a través de la Web y la Web Semántica. Este trabajo se asienta sobre un primer prototipo, todavía no funcional que no cumplía los requisitos necesarios para suponer una ayuda en el trabajo del departamento de documentación. Sobre él se han diseñado e implementado nuevas funcionalidades y módulos con la finalidad de mejor y facilitar el trabajo del departamento de documentación del *Grupo HENNEO*[2]. Estos nuevos módulos consisten en una mejora sustancial de la extracción de información, ampliando el abanico de fuentes de las que se obtiene y mejorando las ya existentes, junto a un módulo de corrección y verificación encargado de asegurar que la información extraída es de calidad. Igualmente, se ha analizado en profundidad el posible trabajo futuro con la posible implementación de dos nuevos módulos capaces de convertir al proyecto en un sistema totalmente automática y con la capacidad de obtener nueva información con el estudio de sus propios datos. Con todo ello, se tiene la finalidad de obtener un sistema final factible de poner en producción y que supondrá un aumento de la calidad en la recuperación de la información.
En este documento se utilizarán el término *infobox* para referirse a hojas informativas para mostrar un resumen de la información sobre una entidad, y el término *entidad* por el cual se entiende a todo aquello que tiene identidad propia, como por ejemplo: Pablo Iglesias Turrión, Alberto Zapater, etc. También se hará referencia a los términos *plantilla* para referirse a un cuadro resumen que permite estructurar la información que se almacena en el sistema, y al término *ficha* como referencia al recurso en el que se almacena la información referente a una *entidad* en concreto.

1.1. Contexto

Este proyecto se lleva a cabo en la empresa *Ibercentro Media Consulting & Services S.L. (IMCS)* perteneciente a HENNEO, séptimo grupo de comunicación español por volumen de facturación con un crecimiento ininterrumpido y uno de los principales grupos de audiencia en su categoría. Este está destinado a la redacción y al departamento de documentación del Heraldo de Aragón, periódico de información generalista, pero centrado especialmente en Aragón.

El *Grupo HENNEO* es un conjunto de empresas que abarca una amplia área de negocio. En el área de las TIC, destacan las empresas como *IMCS* o la consultora *Hiberus Tecnología*. En el área audiovisual, está la cadena local *ZTV* y la productora de contenido *Factoria Plural*. Por último, en el área de prensa, se pueden encontrar empresas como *Heraldo de Aragón*[8], *20 Minutos* o *Heraldo de Soria*.

Tal y como se ha explicado anteriormente, este trabajo se ha llevado a cabo en *ICMS*, sin embargo el resultado final está dirigido a todos los medios de comunicación que forman el *Grupo HENNEO*, especialmente al *Heraldo de Aragón*.

El *Heraldo de Aragón* tiene a su disposición un departamento de documentación. Este se encarga de llevar un control exhaustivo de todas las noticias que se generan diariamente en el periódico para obtener de ellas información interesante. Para facilitar la tarea del departamento se creó una plataforma llamada *EMMA*[1] formada por una serie de procesos y sistemas que permiten al departamento de documentación obtener rápidamente información que haya sido archivada anteriormente. Más adelante, en la sección 3.1.1 se detallará la arquitectura de dicho sistema.

1.2. Motivación

En el contexto donde se realiza el TFG, el departamento de documentación es la sección de la empresa responsable de seleccionar, clasificar y archivar las noticias que se generan, con la finalidad de extraer nueva información de interés para los redactores y periodistas. Este departamento trabaja en una plataforma llamada *EMMA*, desarrollada por la empresa, que permite acceder a la información archivada anteriormente.
EMMA permite realizar búsquedas cuyos resultados son una lista de enlaces a página de interés, similar a cualquier buscador web. En un primer intento, se realizó el análisis y estudio de un prototipo que, usando plantillas, permite estructurar y gestionar la información de forma más sencilla, extrayendo información tanto de EMMA como de la Web Semántica para generar fichas de datos. A partir de una

[1] Entorno Multimedia de Archivo.

entidad, el sistema es capaz de detectarla, asociarla a una de las cuatro plantillas básicas y obtener información de la base de datos local y la DBpedia[7][12] para crear una ficha simulando un *infobox*. Sin embargo, la estructura de las plantillas era bastante pobre y la obtención de la información presentaba errores como la adquisición de información errónea o la no detección de campos básicos. Además, las fuentes de información desde la que se obtenían eran muy limitadas, faltando la fuente más extensa de información de la actualidad: la Web; y las plantillas no era útiles para la finalidad del sistema ya que no se adaptaban a las necesidades de los documentalistas debido a la carencia de campos útiles.
Este primer prototipo ha motivado el desarrollo e implementación de nuevas funcionalidades y módulos con la finalidad de obtener un sistema final funcional que supondrá un aumento de la calidad en la recuperación de la información, mejorando y facilitando el trabajo del departamento de documentación[14].

1.3. Objetivos

Los objetivos del TFG son los siguientes:

1. Estudiar y analizar el prototipo previo del sistema SOPHIE.
2. Mejorar la estructura de las plantillas, añadiendo nuevos campos a las ya existentes y creando nuevas plantillas para las *entidades* más utilizadas por los periodistas y los documentalistas.
3. Diseñar e implementar una nueva funcionalidad que permita al sistema obtener datos de la Web, específicamente de la Wikipedia, de Wikidata, de DBpedia y de Google, con el fin de mejorar la calidad de las fichas.
4. Diseñar e implementar un módulo para extraer datos concretos de fuentes no estructuradas (texto en lenguaje natural, ya sea de la Web, de Wikipedia o de las propias noticias del archivo).
5. Diseñar e implementar un módulo de depuración de errores capaz de detectar automáticamente los siguientes tres tipos de errores en la generación de las nuevas fichas: datos incompletos, datos inexistentes y datos erróneos.
6. Estudiar para trabajo futuro: 1) un módulo de inferencia de los datos añadiendo una mayor funcionalidad al módulo de extracción de información para que este sea capaz de conectar la información de las nuevas fichas con las que ya están recolectadas, creando así una interconexión entre las fichas del sistema, y 2) dotar al sistema de la capacidad de, dada una *entidad*, seleccionar automáticamente la plantilla más adecuada para generar la ficha para dicha *entidad*.

1.4. Tecnología utilizada

Las tecnologías y herramientas utilizadas en el TFG son las siguientes:

- El entorno de trabajo durante todo el desarrollo del proyecto ha sido *Windows 7*.
- Para la implementación del TFG, se han usado las siguientes tecnologías y herramientas: como entorno de desarrollo, se ha utilizado *Visual Studio 2010*. Como lenguaje de programación principal, se ha utilizado *Visual Basic .NET (VB.NET)*, y como gestor de bases de datos (SGBD), se ha usado *Microsoft SQL Server*.
- Para el almacenamiento de las plantillas y de toda la información de las entidades se ha utilizado *Virtuoso Open-Source*[6][2].
- Para la puesta en marcha el sistema se ha apoyado en ontologías, Linked Data, Web y otras plataformas de la empresa.

1.5. Organización de la memoria

En este primer capítulo de la memoria se ha llevado a cabo una breve introducción a modo de presentación sobre el trabajo realizado. En el capítulo 2 se muestra el trabajo de análisis realizado durante el primer tramo del proyecto, útil para definir los objetivos y las funcionalidades finales del sistema. En el capítulo 3 se muestra la arquitectura general en la que se enmarca el proyecto y la arquitectura del nuevo sistema desarrollado. El capítulo 4 detalla los pasos realizados en el segundo tramo del proyecto, con la definición de los procesos implicados, el modelo lógico de los datos y el diseño de la interfaz de usuario. En el capítulo 5, se presentan todos los resultados obtenidos, se hace una lista sobre el posible trabajo futuro y se realiza una valoración personal sobre el trabajo realizado.
Finalmente, en el anexo A, se muestran tanto el manual de usuario, como el de instalación de la aplicación y de Virtuoso. En el anexo B, se incluyen los diagramas de análisis y diseño que se han utilizado para desarrollar el sistema. El anexo C detalla todo el proceso de desarrollo del sistema, y por último el anexo D presenta el trabajo futuro a desarrollar.

[2] OpenLink Virtuoso: http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/.

Capítulo 2

Análisis del sistema

A lo largo de este apartado se describe el proceso de análisis llevado a cabo para el desarrollo del sistema. En primer lugar se establecen el punto de inicio del proyecto, es decir, el estado inicial desde el que se partía para desarrollar el sistema que se ha llevado a cabo. En segundo lugar, se describen los requisitos tanto funcionales como no funcionales que satisface el sistema, obtenidos a partir de la información obtenida por las necesidades de los usuarios. Para finalizar, se identifican los actores del proyecto y se muestran una serie de diagramas UML de diseño.

2.1. Sistema inicial

En una primera aproximación se realizó el análisis y estudio de un prototipo que, usando plantillas, permite estructurar y gestionar la información de forma más sencilla, extrayendo información tanto de EMMA como de la Web Semántica[15] para generar fichas de datos.

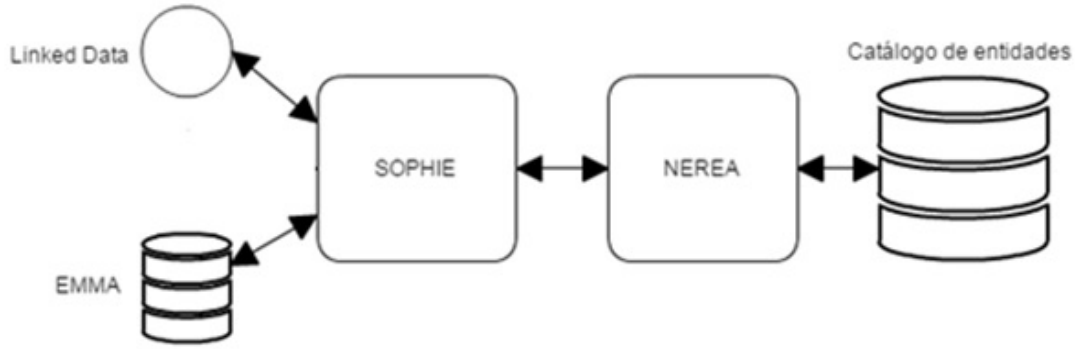


Figura 2.1: Esquema simple del sistema.

A partir de una *entidad*, el sistema era capaz de detectarla, asociarla a una de las cuatro plantillas básicas y obtener información de la base de datos local y la DBpedia[7][1] para crear una ficha básica simulando un *infobox*.

Después del análisis de esta primera aproximación se identificaron varios problemas en el primer prototipo del sistema. La estructura de las plantillas era bastante simple y la obtención de la información presentaba errores como la adquisición de información poco precisa o la no detección de campos básicos. Además, las fuentes de información desde la que se obtenían eran muy limitadas, faltando la fuente más extensa de información de la actualidad: la Web. Por otra parte, las plantillas no eran útiles para la finalidad del sistema ya que no se adaptaban a las necesidades de los documentalistas debido a la carencia de determinados campos básicos.

2.2. Catálogo de requisitos del sistema

En esta sección se van a incluir todos los requisitos que debe cumplir el sistema.

Tabla 2.1: Requisitos funcionales

ID Req	**Requisitos funcionales**
RF - 1	El sistema debe ser capaz de extraer información de EMMA, Linked Data y la Web.
RF - 2	El sistema debe disponer de una plantilla para cada uno de los tipos de entidades más utilizados por los documentalistas.
RF - 3	El sistema debe ser capaz de comprobar si la información de una ficha es válida.
RF - 4	El sistema debe ser capaz de corregir la información de una ficha si esta no es válida.
RF - 5	El sistema debe permitir crear, modificar y eliminar tanto fichas como plantillas.

Tabla 2.2: Requisitos no funcionales

ID Req	**Requisitos no funcionales**
RNF - 1	El sistema se debe poder integrar con las distintas herramientas de la empresa.
RNF - 2	El sistema debe respetar las guías de estilo de la empresa.

[1] https://wiki.dbpedia.org/.

2.3. Identificación de los actores

Para descubrir los problemas del primer prototipo era necesario identificar los actores que interactúan con el sistema, y analizar la forma en que lo hacen:

- El departamento de documentación: encargado de diseñar las plantillas, elegir las entidades y validar las fichas generadas por SOPHIE.
- Departamento de informática: encargado de mantener el sistema y crear nuevas funciones que recuperen los valores deseados en cada campo de la ficha.
- La redacción: usuarios finales de los *infoboxes*, los cuales tendrán acceso a ellos en lectura cuando recuperen información relacionada con alguna de las *entidades* descrita en SOPHIE.

2.4. Diagramas UML de análisis

En esta sección se van a incluir todos los diagramas UML realizados en la fase de análisis del proyecto, para poder llevar a cabo el posterior desarrollo del sistema. Estos diagramas se encuentran en el anexo B.

Capítulo 3

Arquitectura del sistema

En este capítulo se describe la arquitectura del sistema mencionado. La empresa donde se ha llevado a cabo el proyecto tiene a su disposición una plataforma llamada EMMA que permite al servicio de documentación almacenar, obtener y gestionar las fotografías y las noticias, publicadas por los distintos medios de comunicación que pertenecen al *Grupo Henneo*. Esta plataforma es accesible a través de una web interna y que permite realizar búsquedas avanzadas sobre toda la información recopilada en su base de datos. El proyecto se enmarca dentro de esta plataforma, mejorando la calidad en la búsqueda de información dentro de EMMA y sobre todo, mejorando la presentación de la información almacenada en ella.

3.1. Arquitectura global

En esta sección se presentan los distintos módulos y sistemas de los que depende SOPHIE.

3.1.1. Arquitectura de *EMMA*

Es la plataforma formada por un conjunto de repositorios de datos y procesos. Es utilizada tanto por los documentalistas como por los periodistas para acceder y gestionar cualquier tipo de información previamente archivada en la base de datos.

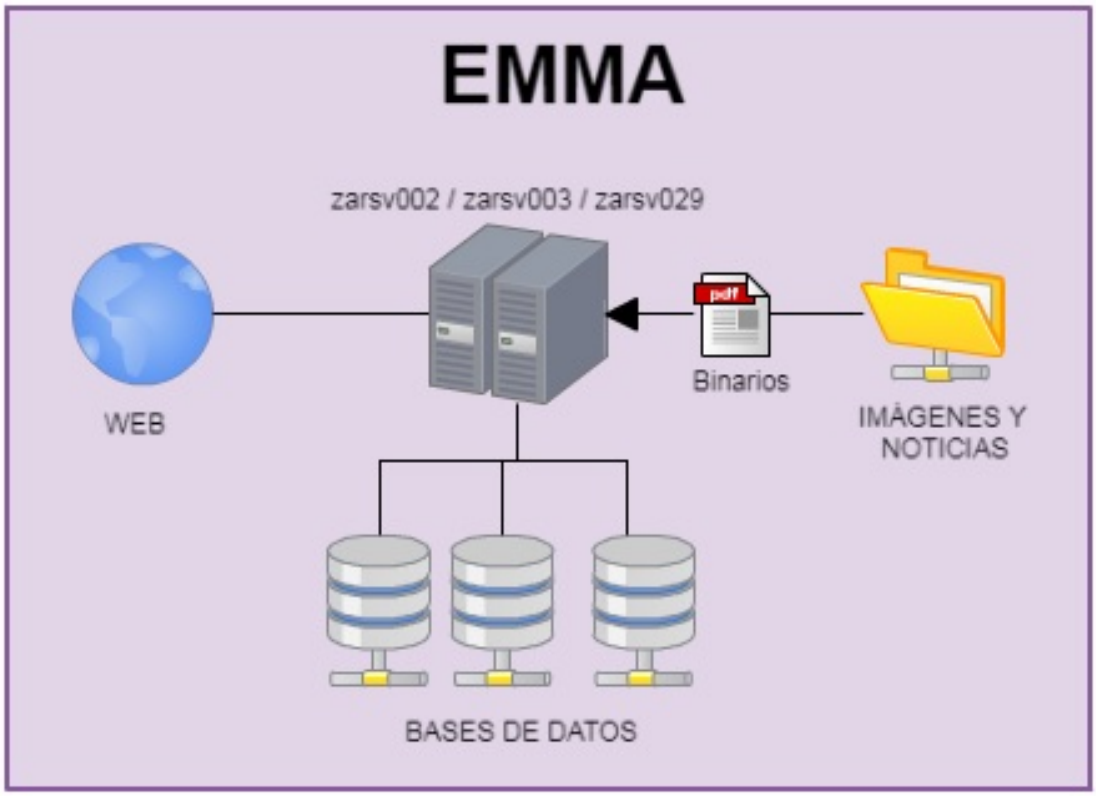


Figura 3.1: Arquitectura del repositorio local de datos - EMMA.

En este repositorio local se almacenan todas las imágenes y noticias tanto del *Heraldo de Aragón* como del resto de prensa que pertenece al *Grupo Henneo*. Esto hace que EMMA cuente con una gran cantidad de información de calidad convirtiéndola en una de las fuentes principales del proyecto, ya que se confía plenamente en la veracidad de la información almacenada en ella.

3.1.2. Arquitectura de *NEREA y POIROT*

Es un sistema automático de reconocimiento de entidades y desambiguación. El objetivo principal de *NEREA*[13] es reconocer las entidades que aparecen en un documento, enviando la información necesaria sobre estas a SOPHIE, que se encargará de obtener la información de las fuentes de datos para generar finalmente el infobox correspondiente.

Para cada documento que analiza NEREA, obtiene los *namedEntities*[1] y accede al catálogo de entidades local, generado anteriormente, para relacionar la entidad

[1]Es un objeto en el mundo real, ya sea personas, ubicaciones, organizaciones, productos, etc., que se puede expresar con un nombre propio.

del texto con la entidad que se encuentra en el repositorio de datos local. Durante la fase de identificación, si se detecta que existe una ambigüedad, el sistema delega la tarea a *POIROT* para obtener la entidad más apropiada. POIROT utiliza información extra para seleccionar la entidad final, principalmente se basa en obtener una bolsa de palabras para cada una de las posibles entidades y compararla mediante el algoritmo del coseno para obtener el porcentaje de similitud entre las entidades comparadas.

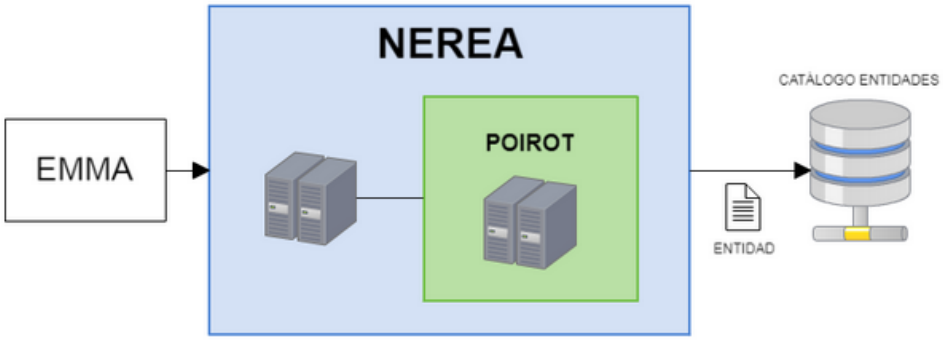


Figura 3.2: Arquitectura de NEREA: sistema de reconocimiento y desambiguación de entidades.

3.1.3. Arquitectura del sistema completo

En esta sección se presenta el sistema global, compuesto por SOPHIE, NEREA, POIROT y EMMA. Para comprender correctamente el funcionamiento del sistema es necesario nombrar varias aplicaciones desarrolladas previamente en la empresa, que hacen posible que SOPHIE funcione correctamente. En primer lugar, SOPHIE es el sistema encargado de generar las fichas. Este se integra con NEREA, cuyo funcionamiento es el siguiente: NEREA analiza un texto local y detecta la aparición de las distintas entidades. Para cada una de las entidades, comprueba si ya está en el *Catálogo de Entidades*. En caso de no existir ninguna entidad con ese nombre, NEREA recoge toda la información posible de las distintas fuentes de datos y añade esta nueva entidad al Catálogo de Entidades. Al contrario, si ya existen varias entidades en el catálogo con el mismo nombre, POIROT se encarga de detectar a cuál de estas entidades se refiere el texto. Para ello, se comparan los conjuntos de palabras almacenadas ("bag of words") para cada una de las entidades con el conjunto de palabras del texto analizado, seleccionando la que más coincidencia

tenga. Al finalizar todo este proceso, el sistema ya ha sido capaz de detectar y diferenciar qué entidad se está nombrando en el texto.

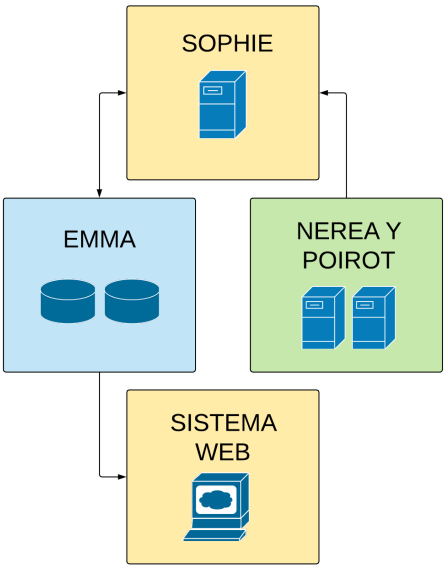


Figura 3.3: Esquema general de la arquitectura del sistema.

El trabajo propuesto se centra en analizar, diseñar e implementar el sistema SOPHIE, del que actualmente existe una primera versión no funcional. SOPHIE se encarga de perfeccionar la base de datos mediante fichas de las entidades, generadas a partir de la información extraída de las fuentes locales, la Web y la Web Semántica.

3.2. Arquitectura *SOPHIE*

Por último se presenta el nuevo sistema desarrollado llamado SOPHIE, que es el encargado de la generación de los infoboxes. Su estructura sigue el esquema de la figura 3.4.

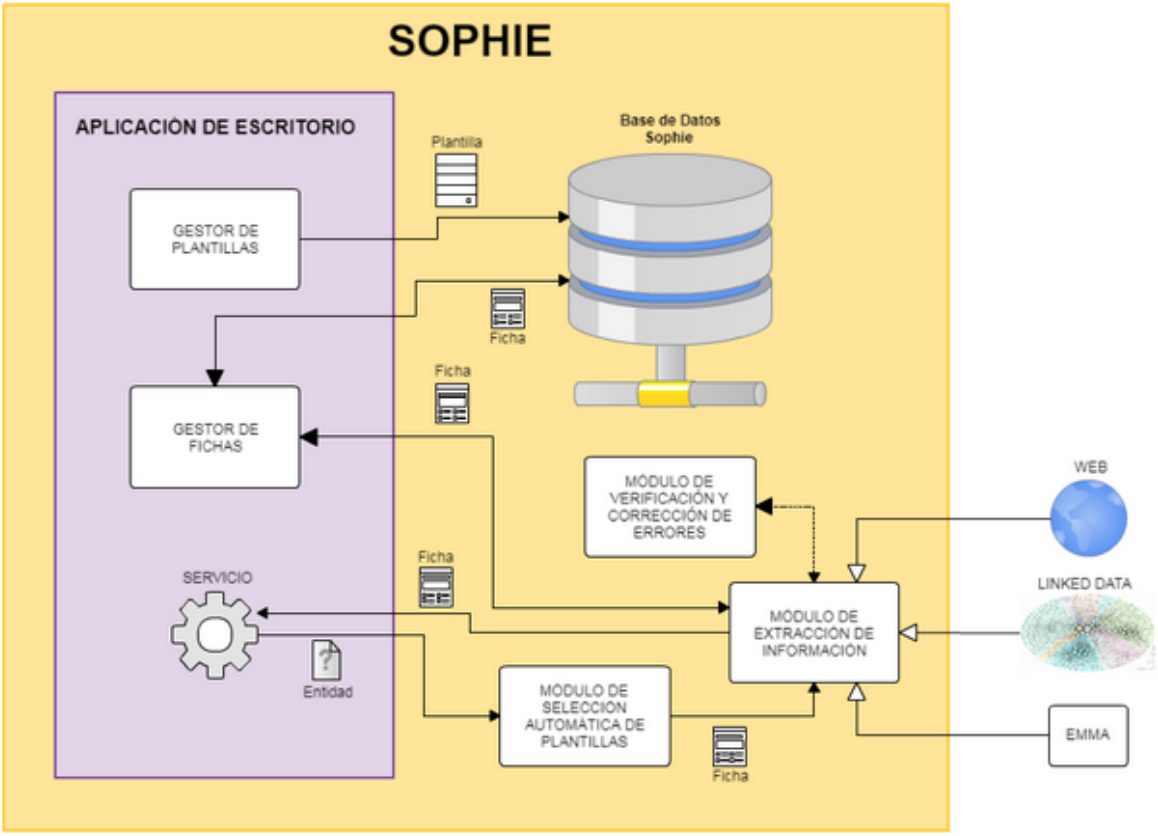


Figura 3.4: Arquitectura de SOPHIE: sistema encragado de la generación de las fichas.

En la imagen podemos distinguir varias partes importantes que forman SOPHIE. Estas son las siguientes:

- Aplicación de escritorio: es la que permite al usuario crear, editar y eliminar de forma manual tanto plantillas como fichas (asociadas a una entidad en el sistema).
  - Gestor de plantillas: crea, edita y elimina las plantillas.
  - Gestor de fichas: crea, edita y elimina las fichas.
- Base de Datos Virtuoso: en ella se almacenan tanto las plantillas como las fichas del sistema. Toda la información referente a la BD se presenta en la sección 4.2.
- Módulo de verificación y corrección de errores: es el encargado de verificar los valores de las fichas generadas. Detecta los errores en estos tipos de valores y, si es posible, los corrige.

- Módulo selección automática de plantillas: futuro módulo que permitirá automatizar por completo el sistema. La idea es que dada una entidad junto a un conjunto de documentos relevantes donde aparece la entidad, estos se comparan con los conjuntos de documentos relevantes para cada una de las plantillas y se selecciona la plantilla cuyo porcentaje de similitud entre documentos es mayor. El conjunto de documentos modelo pertenecientes a cada una de las plantillas serán los documentos que corresponden a cada una de las entidades pertenecientes a dicha plantilla. Así pues, cuantas más entidades se tengan categorizadas, mejor funcionará el sistema.
- Módulo de extracción de la información: está diseñado para obtener la información de tres fuentes de datos distintas: repositorio local (EMMA), la Web (Wikipedia) y Linked Data (DBpedia, Wikidata, ...). A partir de estas fuentes de datos, el módulo es capaz de obtener de mayor calidad de cada una de las fuentes y unificarla generando así la ficha asociada a la entidad. Una vez creada la ficha, el método de verificación y corrección de errores se encarga de comprobar que la información añadida en la ficha de la entidad es correcta, comparando el valor para cada uno de los campos de la entidad en las distintas fuentes de información y asignando un peso según la calidad de la fuente.

Así pues, todos estos módulos que forman SOPHIE permiten que el sistema funcione de la siguiente manera, siguiendo el esquema de la figura 3.5.

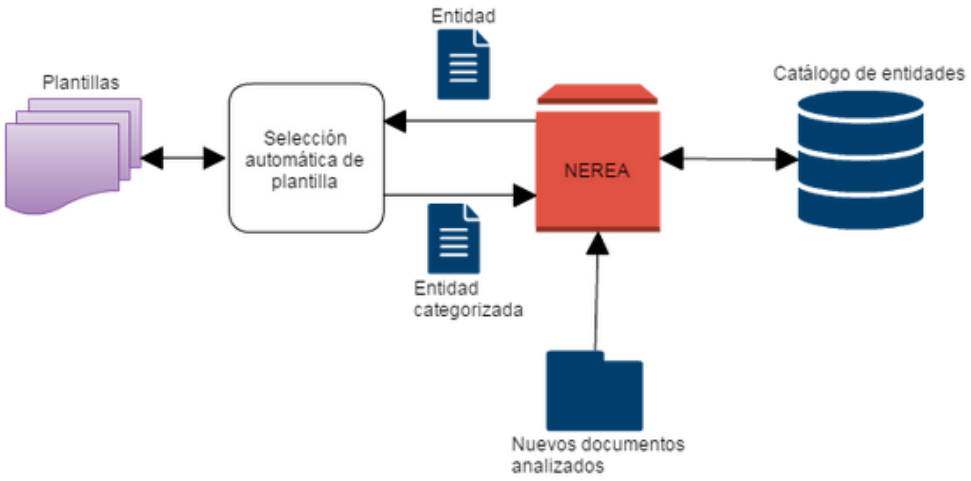


Figura 3.5: Esquema que muestra el funcionamiento del sistema para seleccionar una plantilla automáticamente.

En un primer momento NEREA reconoce las entidades que aparecen en un documento y las almacena en el *Catálogo de Entidades* del sistema. Ahora, SOPHIE

con su módulo de *Selección automática de plantillas* recorre las entidades almacenadas anteriormente y selecciona la plantilla que más se adapta a cada una de estas. Una vez seleccionadas las plantillas, el sistema ya tiene categorizada cada una de las entidades y ya esta preparado para poder rellenar de forma automática todas estas fichas con el *Módulo de extracción de información*. Cuando ya estás fichas generadas, el usuario puede indicar al sistema que verifique y corrija la información de una entidad, marcando así los errores que han ocurrido en la etapa de extracción de información.

Capítulo 4

Diseño e implementación del sistema

4.1. Procesos

El sistema diseñado durante el trabajo está formado por tres componentes relacionados entre sí. Estos son: la aplicación de escritorio, el servicio de generación automática de fichas y el módulo de extracción de información.

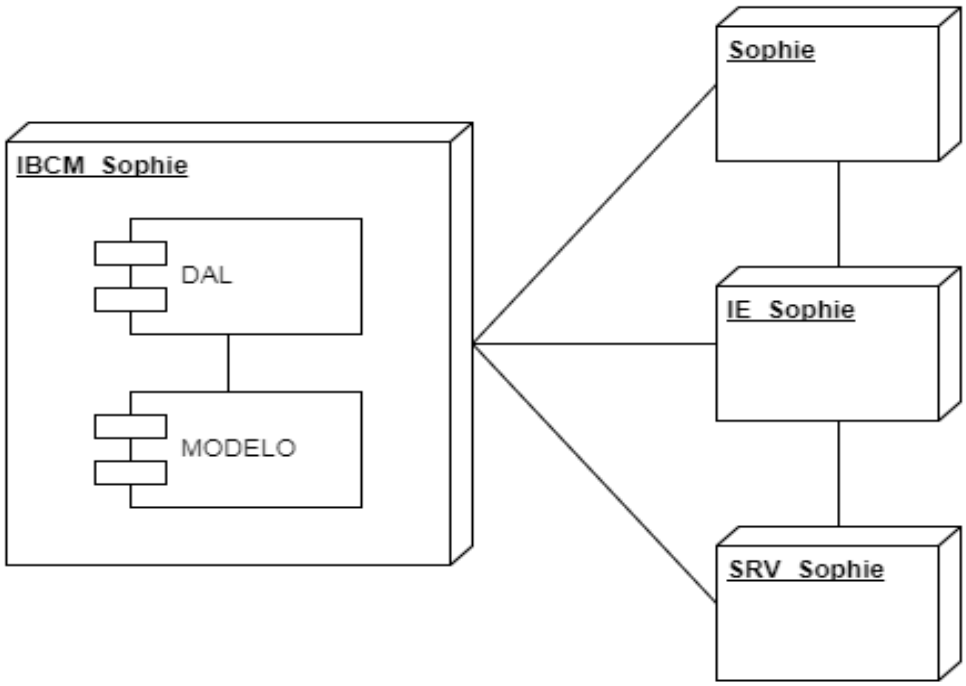


Figura 4.1: Diagrama de componentes del sistema.

Como se puede observar en el diagrama anterior, *IBCM_Sophie* es la librería utilizada por todo el proyecto para poder acceder a las fuentes de información empleadas por el sistema. Esta se ha dividido en dos módulos:

- **MODELO:** representa el modelo de los datos del sistema, es decir, ofrece una serie de clases para poder gestionar todos los datos que se utilizarán a lo largo del proyecto.
- **DAL:** ofrece todas las funciones de acceso a los datos y las funciones que permiten trabajar con estos mediante un conjunto de clases con las que se accede a cada una de las fuentes de información.

SOPHIE es la aplicación de escritorio que permite a los usuarios crear, editar y eliminar tanto fichas como plantillas del sistema. Además, permite de forma manual que los usuarios generen las fichas de cualquier entidad en concreto.

IE_Sophie es el módulo de extracción utilizado por el sistema con la finalidad de obtener la información necesaria para rellenar las fichas de las entidades. Este módulo utiliza las funciones de la librería *IBCM_Sophie* para acceder a las distintas fuentes de información y obtener los datos.

Por último, *SRV_Sophie* es el servicio incluido dentro del sistema que automatizará todo el proceso de creación y selección las fichas. Una vez recibida una entidad, el servicio buscará sus posibles candidatos en el catálogo de entidades del sistema. Cuando este haya encontrado el candidato, seleccionará automáticamente la plantilla que más se asemeja a la entidad y generará automáticamente la ficha con toda la información que se pueda extraer de las distintas fuentes de información.

4.2. Datos

El Modelo de datos se compone de un total de cinco clases, tres de ellas utilizadas para modelar la estructura de las plantillas, fichas y entidades, una para gestionar todas las funciones de los datos, y por último la clase encargada de gestionar la información con la que trabajará la aplicación.

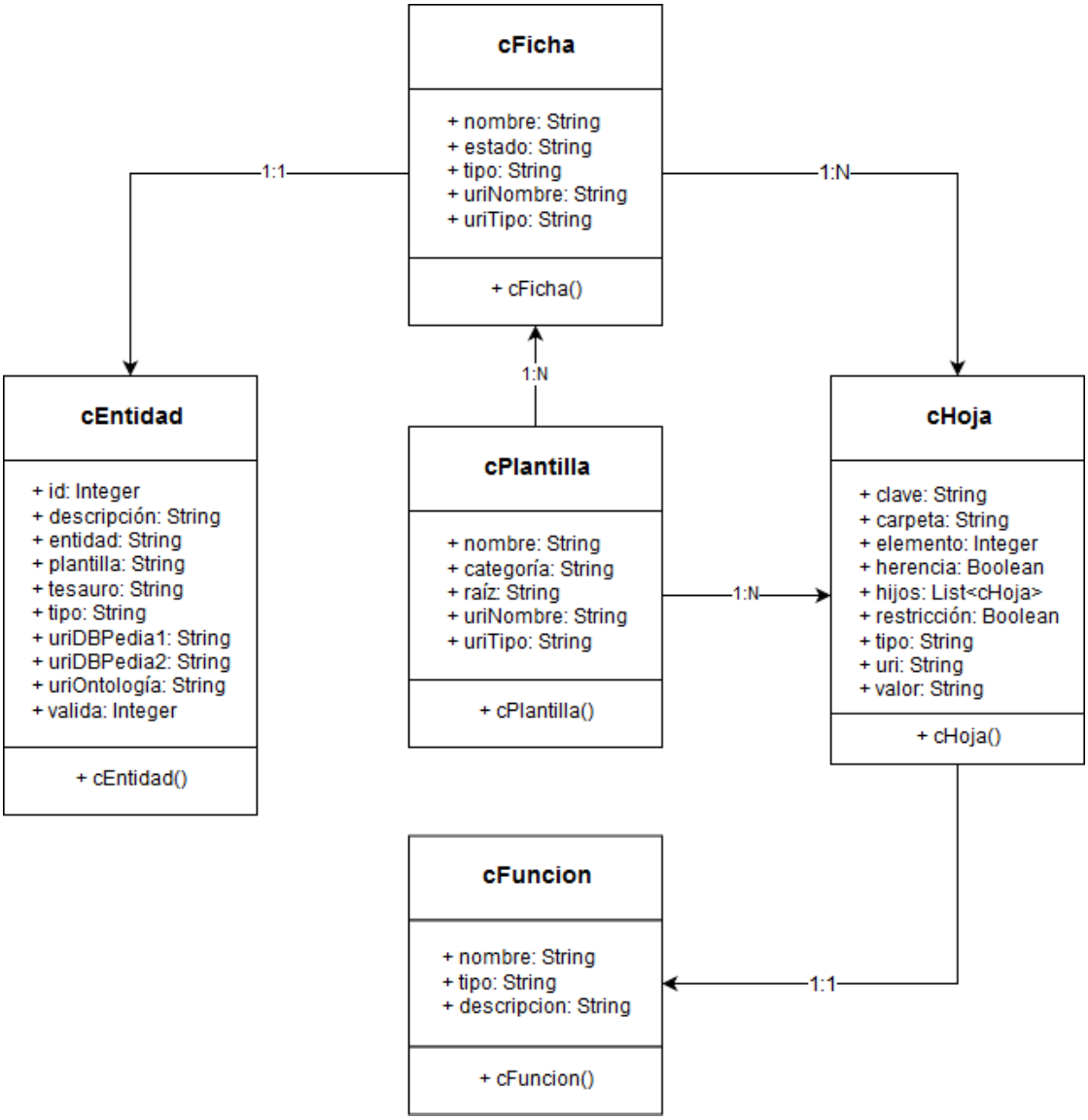


Figura 4.2: Diagrama de clases que modela el modelo de datos del proyecto.

4.2.1. Modelo lógico

Para poder almacenar toda la información que utiliza el sistema es necesario utilizar un diseño de base de datos que favorezca el funcionamiento del sistema. En este caso, se ha elegido utilizar un modelo de datos orientado a grafos en el que la información se representa en forma de nodos (vértices) y relaciones entre estos (aristas).

Esta base de datos está orientada al almacenamiento de plantillas y fichas. Estas plantillas contienen la información básica que debe tener una ficha relacionada con esta, mientras que las fichas están asociadas a una entidad y contiene toda la información posible sobre esta.

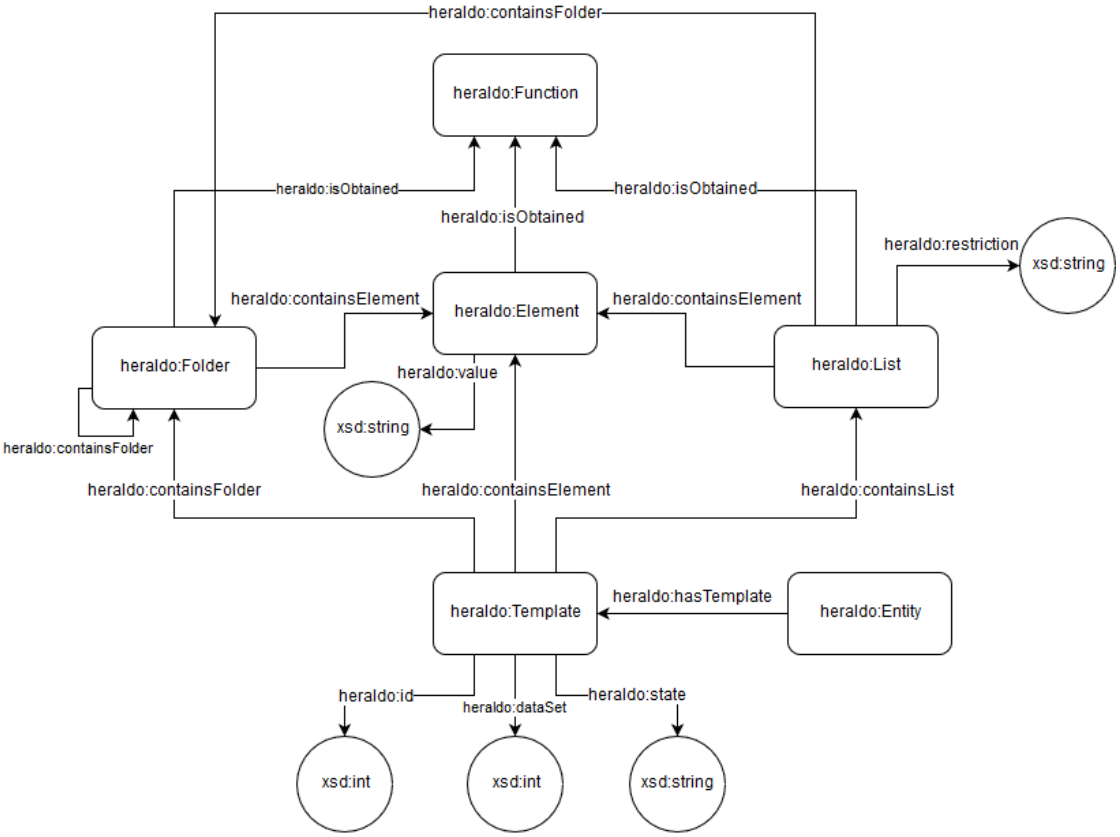


Figura 4.3: Diagrama que modela la ontología del sistema.

El modelo lógico de la base de datos consta de 4 clases y 3 tipos de datos relacionados entre sí con 14 relaciones. Estas clases son los siguientes:

- **Template:** clase que representa las plantillas.
- **DataSheet:** clase que representa las fichas.
- **Folder:** tipo de dato que representa elementos estructurados, es decir, elementos que se utilizan para poder organizar la información. Estos contienen elementos más simples en su interior, listas u otros elementos estructurados.
- **Element:** tipo de dato que representa elementos simples.
- **List:** tipo de dato que representa listas, que pueden ser tanto de elementos simples como de elementos estructurados.
- **Function:** clase que representa una función para la extracción de información para un campo de una plantilla.
- **Type:** clase que representa el tipo de dato de un campo de plantilla específico.

Además de los nodos, el modelo lógico también está formado por 14 relaciones entre estos:

- **Contains_Folder:** relación entre un ítem y un elemento estructurado.
- **Contains_Element:** relación entre un ítem y un elemento simple.
- **Contains_List:** relación entre un ítem y una lista.
- **Is_Obtained:** relación que indica la función con la que se extrae el nodo relacionado.
- **Its_Type:** relación que une el nodo con su tipo de dato.
- **Template_Parent:** relación que asocia una plantilla con su plantilla padre
- **Template:** relación entre una ficha y su plantilla.
- **ValueURI:** relación que especifica el valor de un elemento.
- **Enable:** relación que indica si la clase relacionado está habilitado.
- **State:** relación que indica el estado de la ficha relacionada.
- **Restriction:** relación que indica la restricción de la lista relacionada.
- **Value:** relación que especifica el valor de un elemento.
- **Frecuency:** relación que indica la frecuencia con la que se actualiza el servicio.
- **Time:** relación que especifica la hora a la que se ejecuta el servicio.

4.2.2. Implementación con Virtuoso

Todas las plantillas y fichas del sistema se almacenan en la base de datos Virtuoso. Las plantillas contienen la información básica que debe tener una ficha que pertenece a esta, mientras que las fichas asociadas a una entidad contienen toda la información sobre esta.

Para la creación de una plantilla en el sistema se almacenan en la base de datos las siguientes tripletas:

- Tripleta que indica que estamos insertando una nueva plantilla en el sistema.
- Tripleta que indica de qué tipo es la nueva plantilla (relación con su plantilla padre).
- Tripleta que indica la descripción de la plantilla.

Además, también se almacenan una tripleta por cada uno de los atributos de esta plantilla, indicando además de qué tipo son.

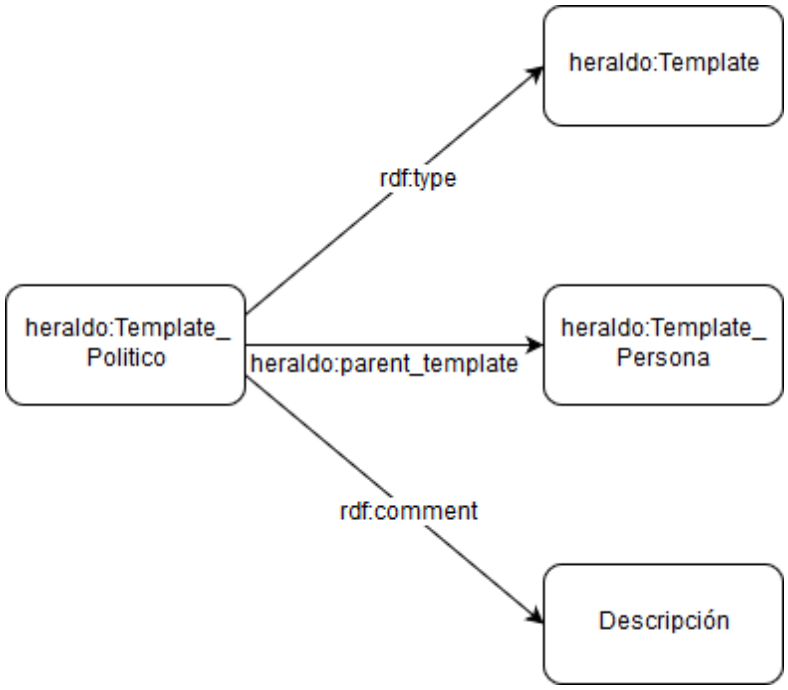


Figura 4.4: Ejemplo de tripletas para la creación de la plantilla "Político".

Para la creación de una ficha en el sistema se introducen los siguientes datos en el grafo:

- La nueva ficha o recurso.
- Información sobre el tipo al que pertenece (plantilla).
- Estado actual de la ficha: puede estar validada o no.
- La ID de la entidad asociada a la ficha en la base de datos local (EMMA).
- DataSet al que pertenece la ficha.

Además, también se almacenan todas las tripletas necesarias para cada uno de los atributos de la ficha. Un ejemplo de ficha dentro de Virtuoso se encuentra en la sección B.4.

4.3. Interfaces de Usuario

Una de las mejoras que se ha realizado sobre la primera versión no funcional del sistema es la presentación final de la información de las fichas. En la primera versión, las fichas eran visibles únicamente a través de la aplicación de escritorio.

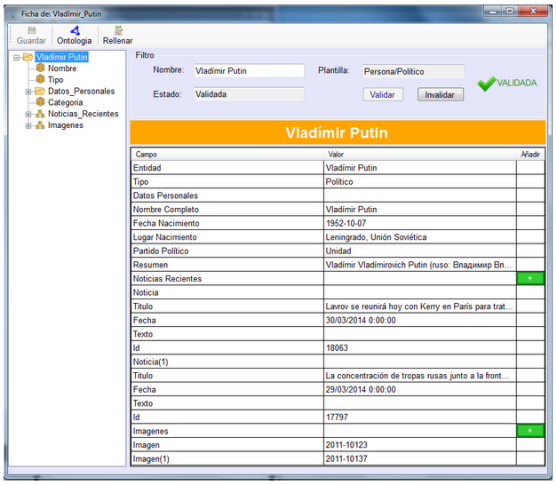


Figura 4.5: Presentación en la primera versión del sistema.

En la última versión, la visualización de las fichas se muestra a través de la aplicación pero de una forma más vistosa y entendible para el usuario, permitiendo que este pueda leer la información de forma rápida.

Figura 4.6: Presentación de la información de una ficha en la última versión del sistema.

En una futura versión del sistema, las fichas se van a presentar siguiendo una plantilla HTML profesional. El diseño de esta plantilla estará formado por una cabecera en la que se mostrará la imagen principal de la entidad, una pequeña descripción y un breve resumen de toda su información a modo de infobox, seguido del cuerpo de la página, donde aparecerán las noticias e imágenes más destacadas de la entidad.

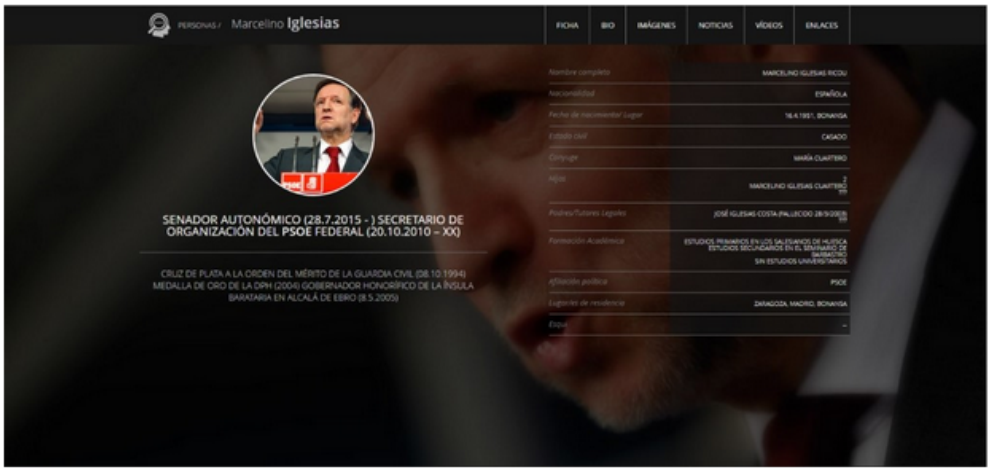


Figura 4.7: Cabecera de una ficha en la versión futura del sistema.

Por último se mostrarán las últimas imágenes y vídeos de la entidad y un apartado con los enlaces recomendados a las páginas de la entidad en Wikipedia o DBpedia, entre otras.

En general el diseño de la ficha que se utiliza es simple y claro, teniendo en cuenta las guías de desarrollo de la empresa y la información obtenida de la interacción del usuario con esta. Como resultado final, se presenta un diseño entendible por el usuario que facilita el aprendizaje y uso de la aplicación. Para más información sobre el diseño futuro de la aplicación ver la sección C.5.

4.4. Diagramas de diseño: Diagrama de componentes

En esta sección se muestra el diagrama final de componentes de la aplicación.

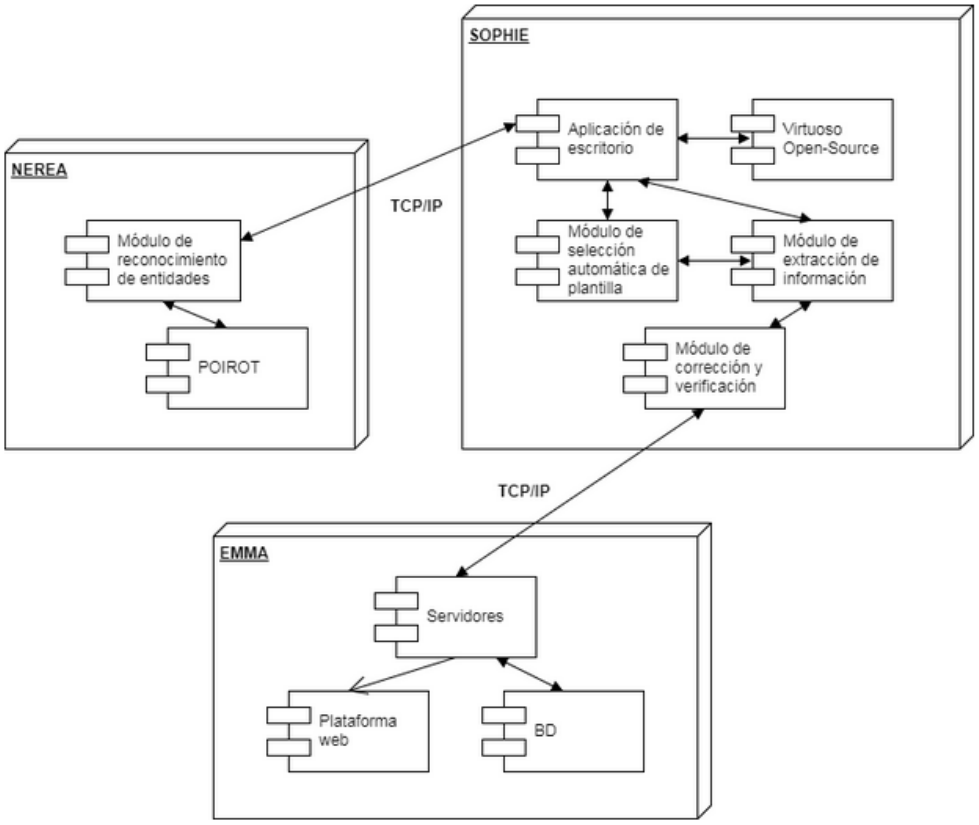


Figura 4.8: Diagrama de componentes de la aplicación.

En el esquema podemos distinguir tres componentes principales:

- ***NEREA***: aplicación de reconocimiento de entidades y desambiguación de la empresa. Se encarga de reconocer las entidades más relevantes de textos locales y de realizar la desambiguación.
  - Módulo de reconocimiento de entidades.
  - *POIROT*: se encarga de realizar la desambiguación.

- ***EMMA***: plataforma local que permite almacenar, obtener y gestionar las fotografías y noticias publicadas por los distintos medios de comunicación que pertenecen al Grupo Henneo.
  - Servidores.
  - Plataforma web.
  - Base de Datos.
- ***SOPHIE***: aplicación encargada de la boxes a través de la obtención de datos de calidad de diversas fuentes.
  - Aplicación de escritorio.
  - *Virtuoso Open-Source*.
  - Módulo de selección automática de plantilla (Futuro).
  - Módulo de extracción de información.
  - Módulo de corrección y verificación.

4.5. Desarrollo del sistema

Este TFG se ha centrado tanto en la mejora de las estructura de las plantillas como en la extracción de los datos para rellenarlas, además de mejorar el modelo lógico de los datos, cuya primera versión se encontraba bastante lejos de presentar un resultado óptimo.

El pilar fundamental del TFG es la extracción de información de calidad de todas las fuentes de datos disponibles, para poder rellenar las fichas de las entidades. Uno de los puntos débiles del primer sistema, era que únicamente extraía información de la DBPedia y de la BD local. Por lo tanto, para mejorar el sistema, se le ha dotado de las características necesarias para poder obtener información de las dos fuentes principales de datos en internet:

- ***Google***:
  - Información sobre lugares y localizaciones con Google Places API.
  - Información personal almacenada en YouTube Data API.
  - Traducción al castellano de información en inglés con Google Cloud Translation API.

- ***Wikipedia***: utilizando la dll *LINQ-To-WIKI* para *Visual Basic*.
  - Se han extraido imágenes de las entidades
  - Los Infoboxes de cada una de las entidades, que junto a expresiones regulares, devuelven una gran cantidad de información de calidad.

Toda la fase de desarrollo del sistema, incluido el módulo de extracción de información, se encuentra explicado detalladamente en el anexo C.

Por último, una de las decisiones más importantes es la elección un SGBD o Repositorio RDF para el sistema. En el primer prototipo, se eligió Virtuoso OpenSource como Repositorio RDF principal sin embargo para este nuevo sistema se requería realizar un nuevo estudio para ver si la elección seguía siendo la correcta. Este análisis se puede ver en la sección C.4.

Capítulo 5

Conclusiones

En esta última parte de la memoria se van a mostrar las conclusiones obtenidas en el desarrollo del TFG junto al trabajo futuro que se puede realizar sobre el sistema. Por último, se indica el tiempo empleado en desarrollar el proyecto y se realiza una valoración personal sobre el trabajo realizado.

5.1. Resultados obtenidos

Teniendo en cuenta los objetivos seleccionados al principio del trabajo, se puede indicar que todos se han cumplido con éxito.

En un primer momento se estudió y analizó el prototipo previo del sistema SOPHIE. Un prototipo muy básico y no funcional que presentaba una primera idea sobre el sistema final que se quería conseguir. Lo primero fue realizar una serie de modificaciones básicas en la estructura de las plantillas, añadiendo nuevos campos a las ya existentes y creando nuevas plantillas para las entidades más utilizadas por los periodistas y documentalistas. Esta mejora se complementó con el diseño e implementación de una nueva funcionalidad en el sistema que permite obtener datos de la Web, específicamente de la Wikipedia, DBpedia y de Google, el cual está explicado en el anexo C.2.

Una vez realizadas las mejoras necesarias en el sistema, se pasó a diseñar e implementar el segundo un módulo de depuración de errores capaz de detectar automáticamente los siguientes tres tipos de errores en la generación de nuevas fichas: datos incompletos, datos inexistentes y datos erróneos.

Finalmente, gracias a las mejoras y la implementación de los nuevos módulos se ha conseguido diseñar e implementar un sistema totalmente renovado y funcional capaz de extraer información de distintas fuentes externas e internas para generar infoboxes de las entidades en el sistema.

Al finalizar el desarrollo del sistema se ha realizado una prueba general del

sistema con un DataSet de prueba, que contiene tanto entidades locales como internacionales. Con esta prueba se buscaba comprobar el porcentaje de acierto del sistema y conocer la opinión del departamento de documentación. Para llevar a cabo la prueba se rellenaron a mano todas las fichas del DataSet de prueba y se almacenaron en la BD. A estas fichas las llamamos *fichas modelo* ya que son las que contienen los valores correctos para cada uno de los campos. Una vez almacenadas se procede a rellenar todas las fichas del DataSet de forma automática con el sistema y se comprueba el porcentaje de similitud entre todos los campos de la ficha rellenada y la ficha modelo. Por último, se hace la media de esos porcentajes y se obtiene el porcentaje final de similitud de la ficha.

Tabla 5.1: Iteración de prueba para comprobar el porcentaje de acierto de la aplicación

Entidad	**Porcentaje de similitud**
Albert Rivera	71 %
Alberto Zapater	77 %
Alejandro Sanz	68 %
Andrés Iniesta	58 %
Valentino Rossi	49 %
José Luis Soro	75 %
Cristiano Ronaldo	63 %
Dani Carvajal	58 %
Fernando Alonso	56 %
Sergio Ramos	58 %
Javier Lambán	63 %
Luisa Fernanda Rudí	67 %
Marc Gasol	63 %
Neymar	66 %
RESULTADO FINAL	**64 %**

Con los resultados obtenidos se deja al sistema en un punto de partida más que aceptable para seguir mejorándolo y poder obtener en un futuro un sistema más preciso en cuanto a la extracción de la información. Estos resultados han sido valorados y aceptados tanto por los documentalistas como por el equipo de desarrollo.

5.2. Trabajo futuro

En esta sección se muestran las posibilidades de ampliación del sistema para hacerlo más útil y automático. Una vez desarrollado el proyecto, se procedió a analizar las posibles mejoras que sería necesarias para facilitar, todavía más, el trabajo de los documentalistas. Para ello se habló con las todas las partes necesarias y se decidió que este trabajo se va a basar en:

- Implementar un módulo de selección automática de plantillas, que sea capaz de seleccionar automáticamente la plantilla correcta para cada una de las entidades del sistema. En el sistema actual, para generar una nueva ficha es necesario que los documentalistas seleccionen la plantilla a la que pertenece. Si se quieren generar muchas fichas esta tarea se puede convertir en algo molesto, por lo tanto, una de las prioridades es que el sistema sea capaz de detectar de forma automática la plantilla a la que pertenece una nueva entidad. La mejor opción para implementar este módulo es asignar a cada una de las plantillas del sistema unos documentos modelo que se compararán con los documentos más relevantes de cada una de las entidades a categorizar para comprobar si el porcentaje de similitud es alto, en cuyo caso, indicará que la entidad pertenece a dicha plantilla.
- Implementar un módulo de inferencia de datos para que el sistema sea capaz de conectar todos los datos que tiene a su disposición, permitiendo así al sistema obtener nueva información a partir de los datos de los que ya dispone. Ahora mismo, el sistema de fichas únicamente almacena los distintos campos de estas junto a su valor. Sería muy interesante añadir una nueva funcionalidad que permitiese al sistema ser capaz de conectar la información de las nuevas fichas con las que ya están recolectadas, creando así una conexión entre todas las fichas del sistema. En un primer momento la implementación de este módulo va a tener como finalidad la inferencia de los datos en las relaciones entre las personas, permitiendo que el sistema sea capaz, a partir de relaciones básicas entre personas, detectar relaciones más complejas.

Un primer desarrollo de estos módulos está explicado en el anexo D.1.

5.3. Estimación de tiempo empleado

En esta última sección de la memoria se presenta una estimación de las horas empleadas en la realización del TFG. En la figura 5.1 se puede ver el cronograma temporal de este TFG.

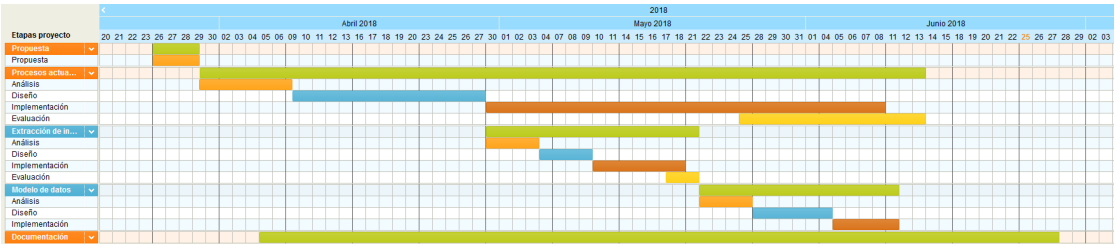


Figura 5.1: Cronograma temporal.

Tabla 5.2: Horas empleadas en cada tarea

Tarea	**Tiempo**
Análisis y estudios previos	100 horas
Diseño	60 horas
Implementación	200 horas
Evaluación	50 horas
Documentación	70 horas

5.4. Valoración personal

La elaboración de este proyecto ha sido mi primera experiencia en el desarrollo de un sistema de tamaño medio dentro de una empresa. Esto me ha permitido aplicar en un entorno de trabajo real todos los conocimientos adquiridos a lo largo de los años de estudio, más en concreto, los conocimientos de *Ingeniería del Software* (para la planificación y análisis previo del proyecto) y de *Sistemas de Información*.

El hecho de de realizar el trabajo en un entorno real, obliga a adaptarse a las herramientas y tecnologías utilizadas por la empresa. Esto me ha permitido conocer y formarme en lenguajes de programación *.NET*, que eran totalmente desconocidos para mí, además de profundizar en los conocimientos de los sistemas *Windows*. Por otro lado, también he podido trabajar con Virtuoso Open-Source, un repositorio RDF, que utiliza el lenguaje de consultas SPARQL[4][11].

El hecho de realizar el TFG en *IMCS* me ha ayudado a conocer el funcionamiento y la planificación de una empresa de gran tamaño. Todo esto me ha permitido realizar un proyecto real, cuya puesta en marcha ayuda y mejora el trabajo del grupo de documentalistas de la empresa.

Bibliografía

[1] Heraldo de Aragón, *https://www.heraldo.es/.* Última visita: 28 de Junio de 2018

[2] Grupo HENNEO, *https://www.henneo.com/.* Última visita: 28 de Junio de 2018

[3] RDF, *https://www.w3.org/RDF/.* Última visita: 28 de Junio de 2018

[4] SPARQL, *https://www.w3.org/TR/sparql11-overview/.* Última visita: 28 de Junio de 2018

[5] NoSQL, *http://nosql-database.org/.* Última visita: 28 de Junio de 2018

[6] Virtuoso Open-Source Edition, *http://vos.openlinksw.com/owiki/wiki/VOS/.* Última visita: 28 de Junio de 2018

[7] DBPedia, *https://wiki.dbpedia.org/, http://es.dbpedia.org/.* Última visita: 28 de Junio de 2018

[8] Heraldo de Aragón, *https://www.wikipedia.org/.* Última visita: 28 de Junio de 2018

[9] WMK Trochim, JP Donnelly, *Research methods knowledge base.* 2007

[10] Berners-Lee, T., *Design Issues: Linked Data.* 2006

[11] Olaf Hartig, Christian Bizer, Johann-Christoph Freytag, *Executing SPARQL Queries over the Web of Linked Data.* 2009

[12] Soren Auer,Christian Bizer,Georgi Kobilarov,Jens Lehmann, Richard Cyganiak, Zachary Ive, *DBpedia: A Nucleus for a Web of Open Data.* 2007

[13] Ángel L. Garrido, Sergio Ilarri, SUsana Sangiao, Adrian Gañán, Alejandro Bean, Óscar Cardiel, *NEREA: Named Entity Recognition and Disambiguation Exploiting Local Document Repositories.* 2015

[14] A. L. Garrido, A. Peiro, S. Ilarri, *Hypatia: An expert system proposal for documentation departments.* 2014

[15] A. L. Garrido, P. Blázquez, M. G. Buey, S. Ilarri, *Knowledge Obtention Combining Information Extraction Techniques with Linked Data.* 2015

[16] A. L. Garrido, Susana Sangiao, Óscar Cardiel, *Improving the Generation of Infoboxes from Data Silos through Machine Learning and the use of Semantic Repositories.* 2017

[17] Bizer, Christian, Heath, Toma, Berners-Lee, *Linked Data - the story so far.* 2009

[18] P.T. Nguyen, Yutaka Matsuo, Mitsuru Ishizuka, *Exploiting Syntactic and Semantic Information for Relation Extraction from Wikipedia.* 2007

[19] Silviu Cucerzan, *Large-Scale Named Entity Disambiguation Based on Wikipedia Data.* 2007

Anexos

Apéndice A

Manuales de usuario y configuración

En esta primera parte de los Anexos se incluyen el manual de usuario, para facilitar el uso de la aplicación de escritorio, y el manual de instalación del sistema.

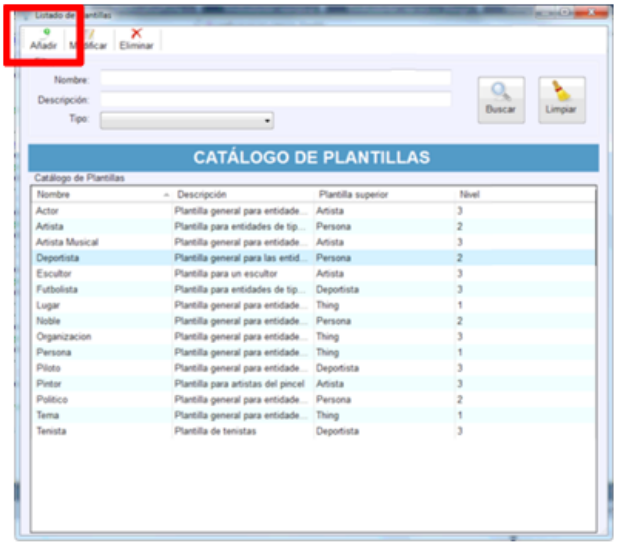
A.1. Manual de usuario de SOPHIE

En este manual se explica cómo realizar todas las operaciones básicas con plantillas y fichas en SOPHIE. Para trabajar con la aplicación, el usuario debe acceder a la aplicación de EMMA y buscar en el menú principal la aplicación de SOPHIE.

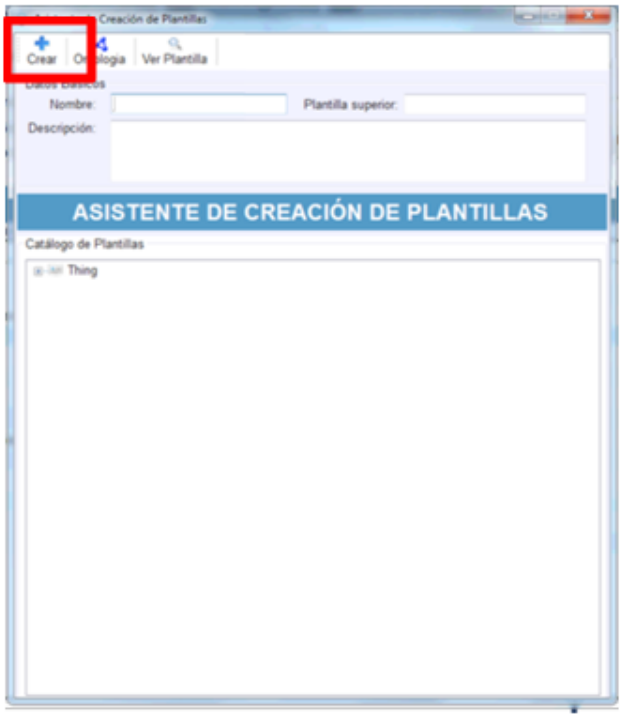
A.1.1. Crear una plantilla

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Plantillas.

3. Aparecerá la siguiente pantalla en la que se muestran todas las plantillas del sistema. Para añadir una nueva plantilla bastará con pulsar sobre el botón AÑADIR.



4. Aparecerá la siguiente pantalla.

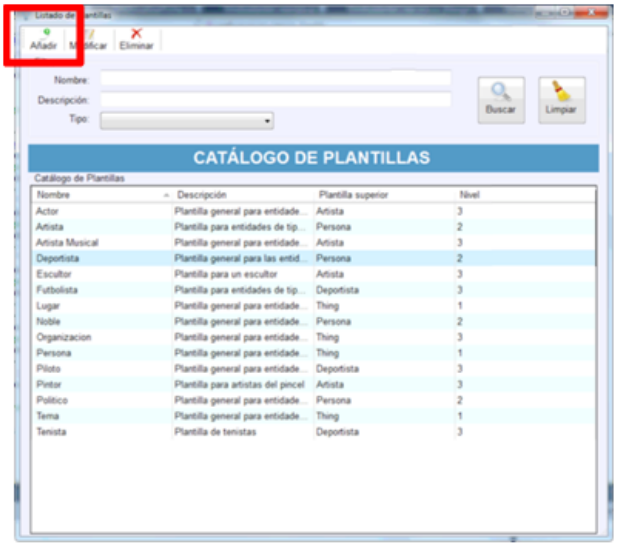


- **Nombre:** Se debe indicar el nombre de la nueva plantilla que se va a crear.
- **Plantilla superior:** Se debe indicar cuál será su plantilla padre de la que herede los campos (si no queremos que herede de nadie hay que indicar que la plantilla padre es THING).
- **Descripción:** Se debe indicar una breve descripción de la plantilla que se va a crear.

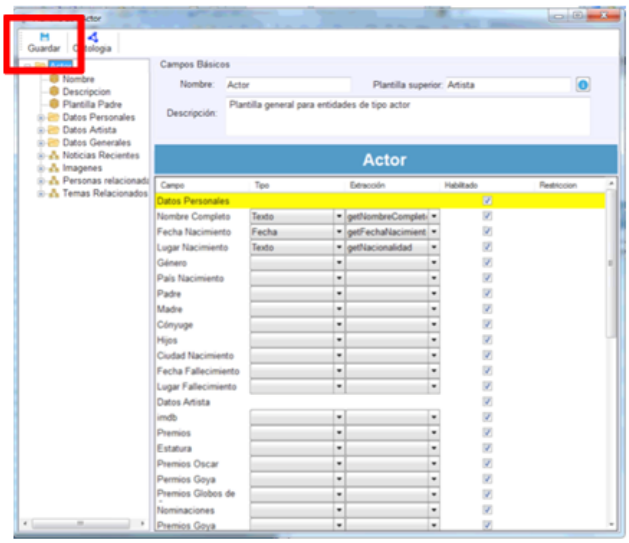
A.1.2. Modificar una plantilla

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Plantillas.

3. Aparecerá la siguiente pantalla en la que se muestran todas las plantillas del sistema. Para añadir una nueva plantilla bastará con pulsar sobre el botón MODIFICAR.



4. Aparecerá la siguiente pantalla.



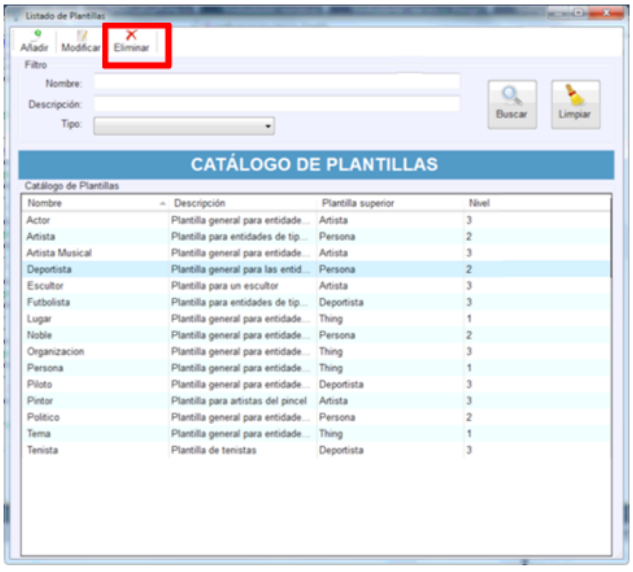
Esta es una vista general de la plantilla, con todos los campos que la forman y todas las funciones utilizadas para extraer dichos campos desde las distintas fuentes de extracción. El usuario puede añadir, modificar y eliminar cualquier campo, además de cambiar la función de extracción asociada.

1. **Campo:** Indica el nombre del campo de la plantilla.
2. **Tipo:** Indica de qué tipo es el valor extraído.
3. **Extracción:** Indica la función que se utiliza para extraer la información.
4. **Habilitado:** Indica si el campo está habilitado o no.
5. **Restricción:** Indica si la lista tiene que cumplir la restricción del sistema (únicamente se utiliza en listas).

A.1.3. Eliminar una plantilla

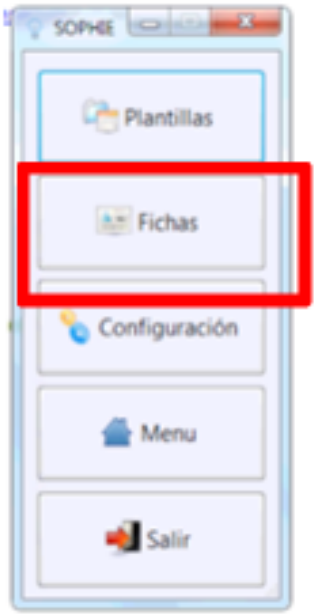
1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Plantillas.

3. Aparecerá la siguiente pantalla en la que se muestran todas las plantillas del sistema. Para añadir una nueva plantilla bastará con pulsar sobre el botón ELIMINAR.

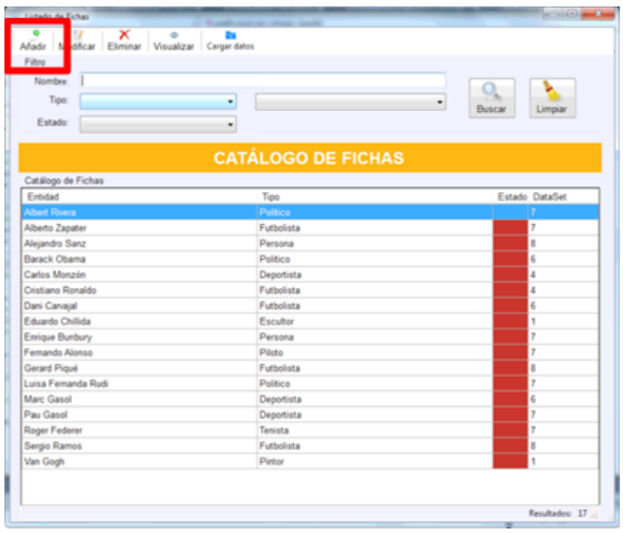


A.1.4. Crear una ficha

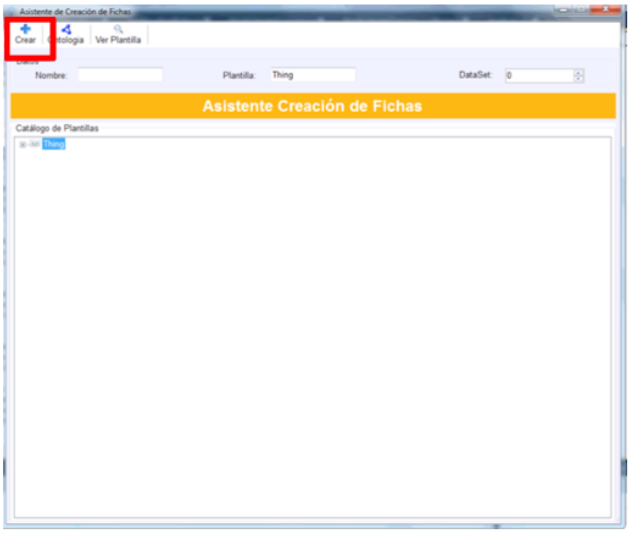
1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Fichas.



3. Aparecerá la siguiente pantalla en la que se muestran todas las fichas del sistema. Para añadir una nueva ficha bastará con pulsar sobre el botón AÑADIR.



4. Aparecerá la siguiente pantalla.

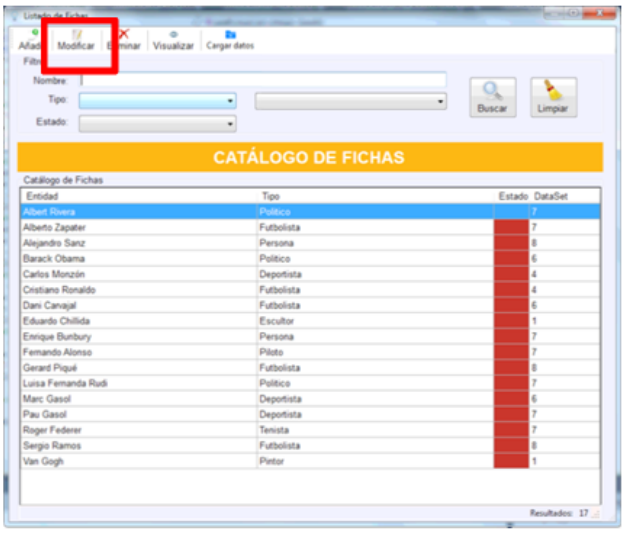


- **Nombre:** Se debe indicar el nombre de la nueva ficha. Este nombre tiene que coincidir con el nombre de una de las entidades del catálogo de NEREA.
- **Plantilla:** Se debe indicar a qué plantilla pertenece la ficha.
- **DataSet:** Se debe indicar a que grupo de pruebas pertenece la ficha.

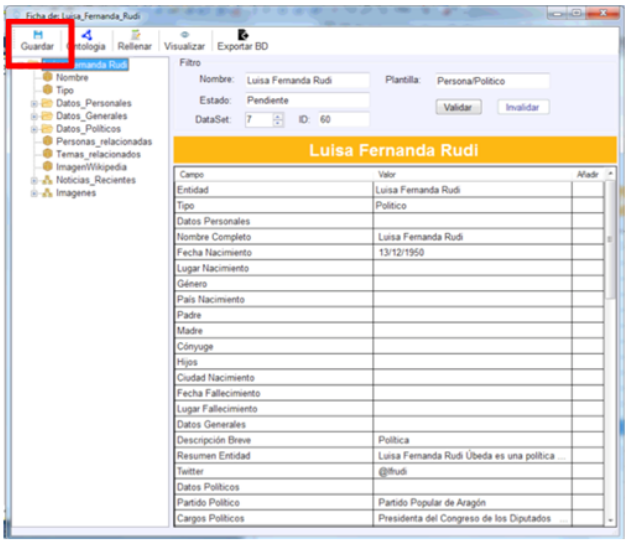
A.1.5. Modificar una ficha

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Fichas.

3. Aparecerá la siguiente pantalla en la que se muestran todas las fichas del sistema. Para modificar una ficha bastará con pulsar sobre el botón MODIFICAR.



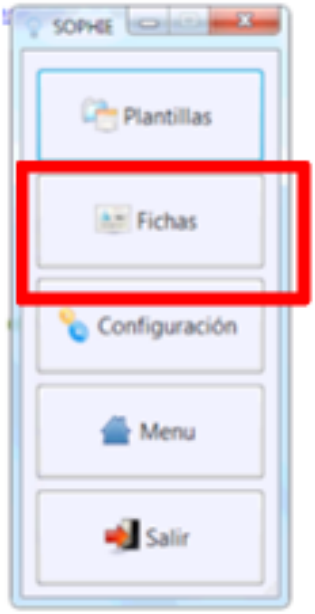
4. Aparecerá la siguiente pantalla.



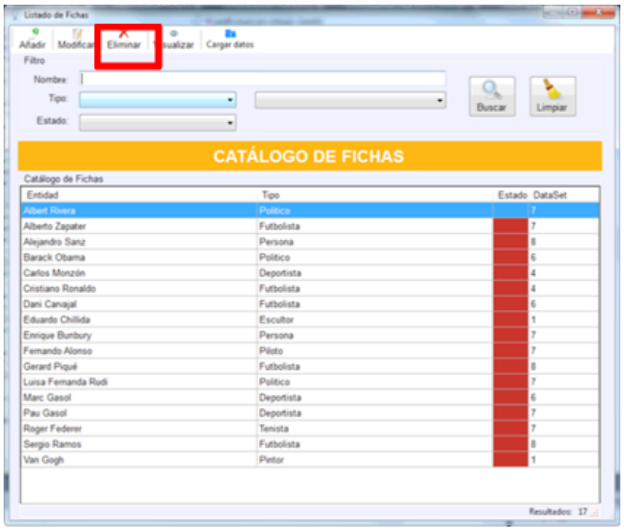
Esta es una vista general de la ficha, con todos los campos que la forman y el valor actual de estos campos. El usuario tiene la posibilidad de modificar a mano cada uno de los campos haciendo doble click sobre estos. Una vez se han modificado los campos, el usuario deberá pulsar el botón GUARDAR.

A.1.6. Eliminar una ficha

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Fichas.



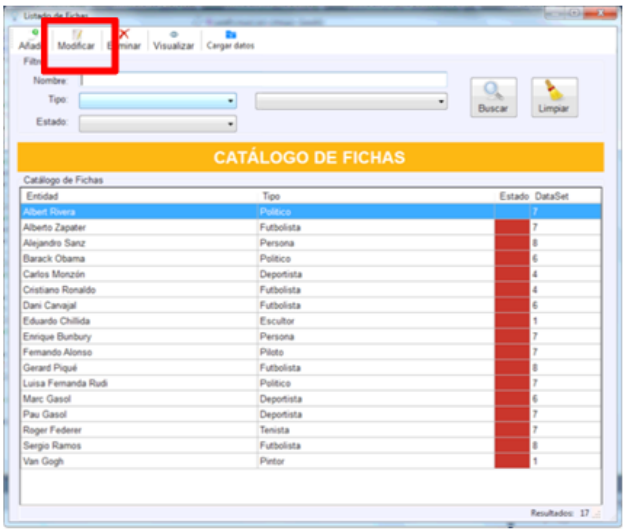
3. Aparecerá la siguiente pantalla en la que se muestran todas las fichas del sistema. Para borrar una ficha bastará con pulsar sobre el botón ELIMINAR.



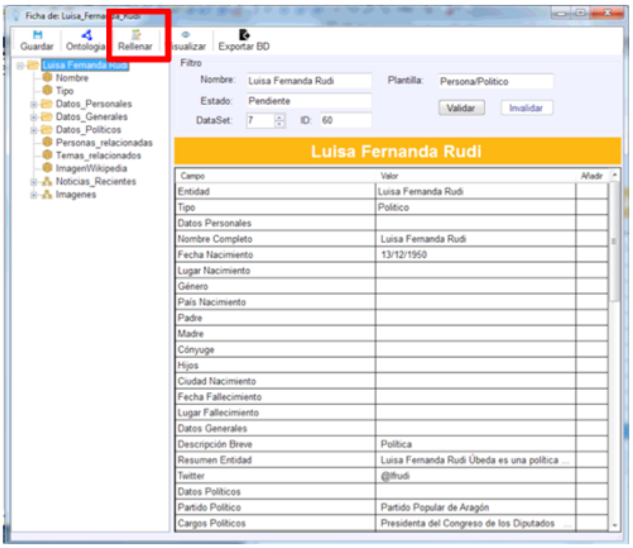
A.1.7. Rellenar una ficha

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Fichas.

3. Aparecerá la siguiente pantalla en la que se muestran todas las fichas del sistema. Para cambiar una ficha bastará con pulsar sobre el botón MODIFICAR.



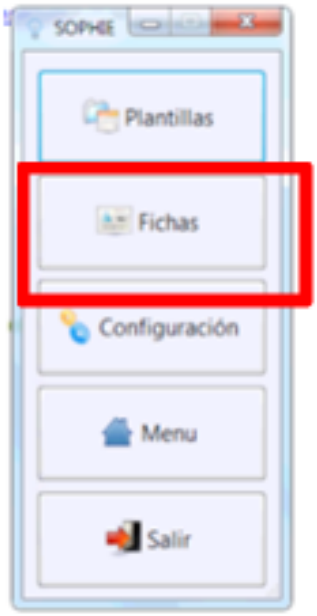
4. Aparecerá la siguiente pantalla.



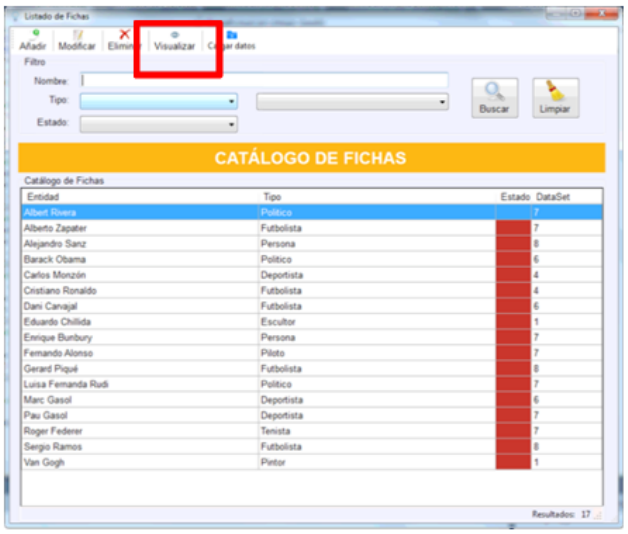
Esta es una vista general de la ficha, con todos los campos que la forman y el valor actual de estos campos. Para rellenar la ficha de forma automática con la información extraída de las distintas fuentes de información el usuario deberá pulsar el botón RELLENAR.

A.1.8. Visualizar una ficha

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Fichas.



3. Aparecerá la siguiente pantalla en la que se muestran todas las fichas del sistema. Para ver una ficha bastará con pulsar sobre el botón VISUALIZAR.

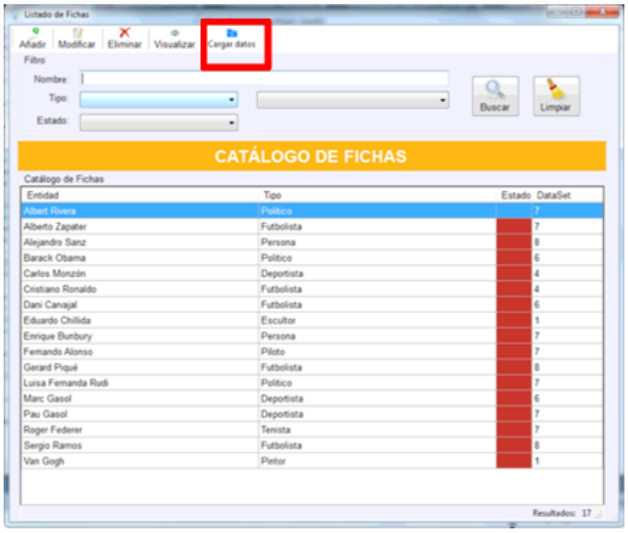


4. Aparecerá la siguiente pantalla.

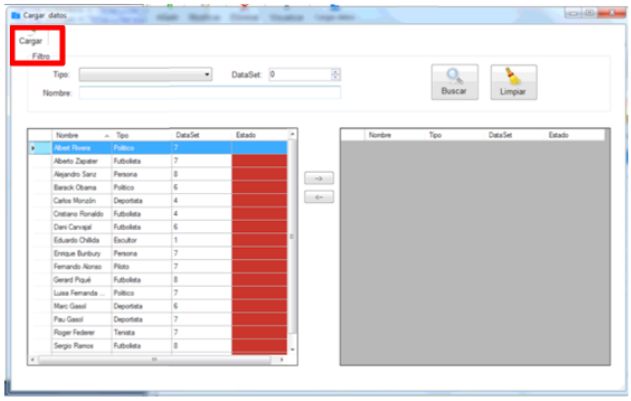
A.1.9. Carga conjunta

1. Acceder a la aplicación a través de EMMA.
2. En la aplicación entrar en la opción Fichas.

3. Aparecerá la siguiente pantalla en la que se muestran todas las fichas del sistema. Para cargar los datos de las fichas bastará con pulsar sobre el botón CARGAR DATOS.



4. Aparecerá la siguiente pantalla.



En la parte izquierda de la pantalla aparecen todas las fichas del sistema. Para seleccionarlas, el usuario deberá pulsar el botón ->"para desplazar las fichas que quiere rellenar a la parte derecha de la pantalla. Una vez seleccionadas las fichas, el usuario deberá pulsar el botón CARGAR para comenzar con la tarea (se le notificará al usuario que el proceso puede ser costoso, y en todo momento se le indicará el avance del proceso con una barra de progreso).

A.2. Manual de instalación de *SOPHIE*

A continuación, se van a enumerar los pasos necesarios para poner en marcha la aplicación, además de las pautas para añadir nuevos elementos a esta.

A.2.1. Requisitos de la instalación

Los requisitos previos de hardware y software necesarios para proceder a la instalación son los siguientes:

1. **Hardware:**
   a) Servidor de alojamiento.
   b) Servidor de EMMA.
   c) Servidor de Virtuoso.
2. **Software:**
   a) Sistema Operativo compatible con la aplicación (*Windows 7*)
3. **Redes y comunicaciones:**
   a) La máquina donde se aloja la aplicación debe disponer de conexión a Internet

A.2.2. Instalación de la aplicación

Las pautas y procedimientos para instalar la aplicación son los siguientes:

1. Comprobar que el resto de sistemas locales con los que se comunica funcionan correctamente.
2. Instalar y configurar la base de datos *Virtuoso*.
3. Compilar el código fuente como DLL.
4. Desplegar el código compilado en EMMA, como una opción más dentro del menú principal de aplicaciones.
5. Puesta en producción del sistema.
6. Instalación del sistema en los ordenadores de los documentalistas.

Una vez se han realizado todos los pasos el sistema estará listo para ser usado por los documentalistas. Estos deberán acceder al sistema desde el menú principal de EMMA y para poder empezar a usarlo introducirán nuevas entidades en el catálogo de NEREA, generarán manualmente la ficha y la rellenarán automáticamente, una vez rellenada comprobarán si los datos son correctos y validarán la ficha. Cuando la ficha está validad indica que ya está lista para usarse y se puede visualizar correctamente.

En un futuro, una vez validada la ficha se podrá exportar a *SQL* y con esto se generará una plantilla en formato HTML. La intención es que en un futuro todo el sistema funcione sobre *SQL Server*, ya que es la tecnología con la que está más familiarizada la empresa.

A.2.3. Carga de datos iniciales

Una vez la aplicación se ha puesto en marcha, al usuario le pueden surgir varios casos de uso:

1. Se genera una nueva entidad en el catálogo de entidades de NEREA: Cuando esto ocurre, la aplicación ya puede generar una nueva ficha para esta entidad. Para ello, el usuario debe buscar manualmente a qué plantilla pertenece esta ficha y generarla. Una vez la ficha vacía de la entidad ya se ha creado, es el momento de rellenarla. Para ello, el usuario puede decidir si rellenar la ficha a mano o buscar la información de esta en las distintas fuentes de información.
2. Dar de alta una nueva entidad: El usuario tiene que introducir manualmente la entidad en el catálogo de entidades de NEREA. En este se almacena tanto el nombre de la entidad como su enlace a la Dbpedia.

3. Crear una nueva plantilla: En algún momento puede surgir la necesidad de generar una nueva plantilla para un tipo de entidades en concreto. Para ello, el usuario tiene que acceder a la pestaña de "Plantillas" y pulsar sobre el botón "Añadir".

   Para crear una nueva plantilla es necesario añadir un nombre, una descripción y seleccionar la plantilla padre de la que heredará todos los campos. Si se da el caso de que no tiene plantilla padre, se debe seleccionar la plantilla "Thing" como padre.

   Una vez se han rellenado todos los datos necesarios y se genera la plantilla se pueden añadir nuevos campos a esta (además de los campos de la plantilla padre) o eliminar campos sobrantes. Cuando la plantilla ya se tiene configurada correctamente, se almacena en la Base de Datos de Virtuoso.

A.2.4. Depuración de la puesta en marcha

Tabla A.1: Tareas para la depuración de la aplicación

Prueba a realizar	**Resultado esperado**
Listar las plantillas	Listado de todas las plantillas almacenadas en Virtuoso.
Listar las fichas.	Listado de todas las fichas almacenadas en Virtuoso.
Crear una nueva plantilla.	Almacenado de la nueva plantilla.
Crear una nueva ficha.	Almacenado de la nueva ficha.
Modificar una plantilla.	Almacenado de la plantilla con los nuevos campos.
Modificar una ficha.	Almacenado de la ficha con los nuevos campos.
Eliminar una plantilla.	Eliminación de la plantilla en *Virtuoso*.
Eliminar una ficha.	Eliminación de la ficha en *Virtuoso*.
Rellenar una ficha de manera automática.	Obtención de información de las distintas fuentes.

A.3. Instalación y configuración de *Virtuoso*

En esta sección se muestra el manual de instalación y configuración de Virtuoso en un entorno de trabajo Windows. Para realizar estos manuales se ha obtenido la información de la página web oficial de *OpenLink Virtuoso*[1].

A.3.1. Requisitos previos

Los requisitos necesarios para poder ejecutar el Repositorio RDF *Virtuoso* son:

1. Instalación previa de *Microsoft Visual C++ 2010 Redistributable (x64)* descargable desde *Microsoft*[2].
2. Se requiere tener instalado *Microsoft Visual C++ 2012 Redistributable (x64)* descargable desde *Microsoft*[3] correspondiente.
3. Disponer de un sistema de 64 bits.

A.3.2. Manual de instalación

Para instalar el repositorio se tienen que seguir los siguientes pasos, en el orden indicado:

1. Descargar el programa de la página web oficial de Virtuoso OpenLink[4].
2. Descomprimir el fichero descargado en la ruta en la que se vaya a instalar el *software*.
3. Descargar el fichero **php.ini** de la siguiente página web[5] en el directorio **\database** que se encuentra en la carpeta descomprimida en el anterior punto.

[1] OpenLink Virtuoso: http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/.
[2] *MS Visual C++ 2010*: https://www.microsoft.com/es-es/download/details.aspx?id=14632.
[3] *MS Visual C++ 2012*: https://www.microsoft.com/es-es/download/details.aspx?id=30679.
[4] Descarga principal de *Virtuoso*:
http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSDownload.
[5] Fichero php.ini: ftp://download.openlinksw.com/support/vos/php.ini.

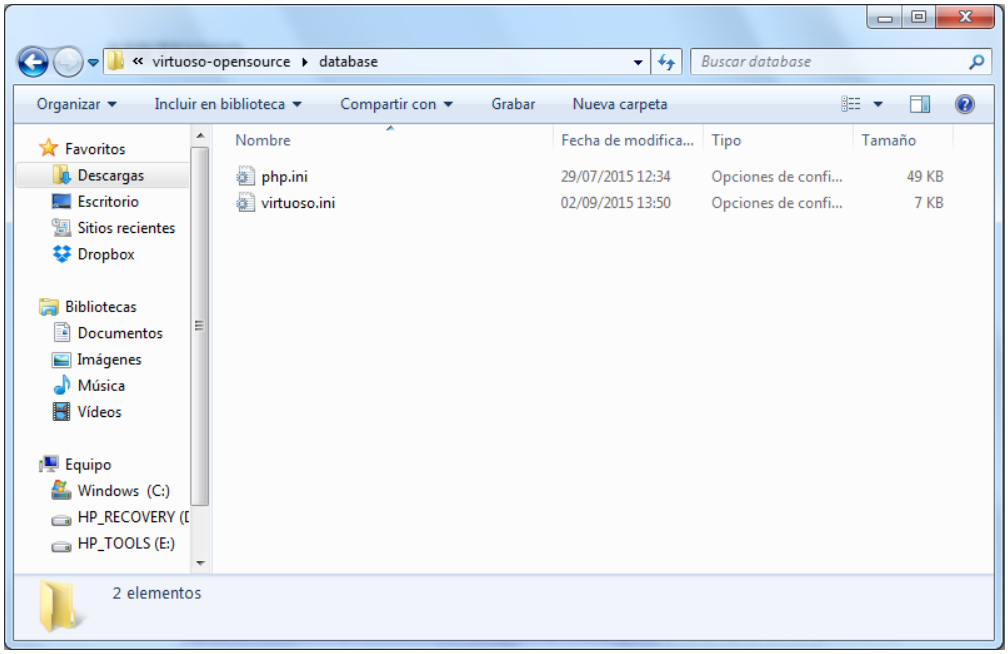


Figura A.1: Estado del directorio \database antes de la instalación de *Virtuoso*.

4. Crear una variable de entorno del sistema para *Virtuoso*.
5. Asignarle a la variable de entorno anterior la ruta en la que se encuentra el directorio de raíz de la instalación del *software*.
6. Modificar la variable de entorno de sistema **PATH**, añadiendo al final los directorios **\bin** y **\lib** de la instalación de *Virtuoso*: **(PATH_ACTUAL;%VIRTUOSO_HOME%\bin;%VIRTUOSO_HOME%\lib)**
7. Reiniciar el sistema.

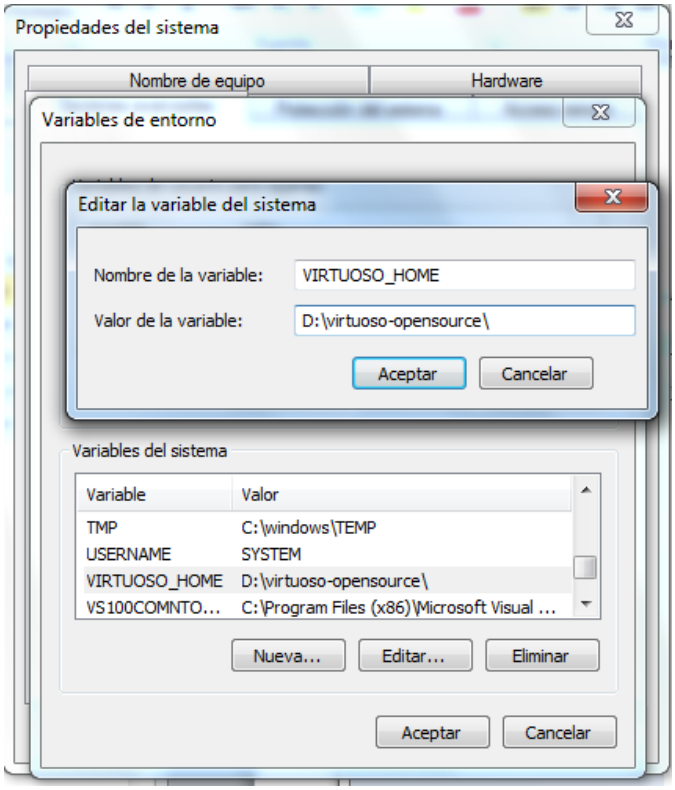


Figura A.2: Creación de la variable de entorno para *Virtuoso*.

8. Por último se necesita crear un nuevo servicio que es el que pondrá en funcionamiento el nuevo Repositorio RDF. Para ello, desde el directorio **database** de la instalación se debe ejecutar el siguiente comando mediante la consola del sistema:

```
virtuoso-t +service create +instance "Virtuoso-1" +configfile virtuoso.ini .
```

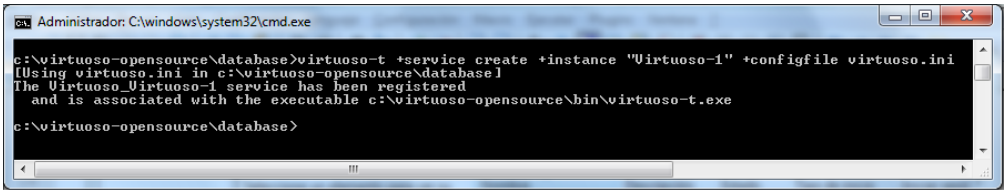


Figura A.3: Creación del servicio base de *Virtuoso*.

A.3.3. Manual de configuración del servicio

En esta sección se muestran los conceptos básicos y más avanzados para configurar el servicio de Virtuoso OpenLink.
La configuración más básica, es decir, ficheros utilizados para log, el puerto de acceso, número de hilos de ejecución, etc. se ajustan en el fichero de configuración **virtuoso.ini** que se encuentra en el directorio **\database** de la instalación de *Virtuoso*.

Si queremos configurar parámetros más avanzados, el usuario tiene que acceder a **Conductor**, el panel de control de *Virtuoso* a través de la URL[6] de acceso al panel. En este panel de control, el usuario puede modificar la configuración básica nombrada anteriormente de una forma más sencilla, apoyándose en la interfaz que presta al usuario. Sin embargo, también puede acceder a la configuración avanzada, ya que el panel permite crear usuarios y roles, establecer mecanismos de seguridad, monitorizar el estado del sistema, instalar o desinstalar módulos, etc.

En la figura A.4 se puede ver el aspecto que tiene el panel de configuración de *Virtuoso*.

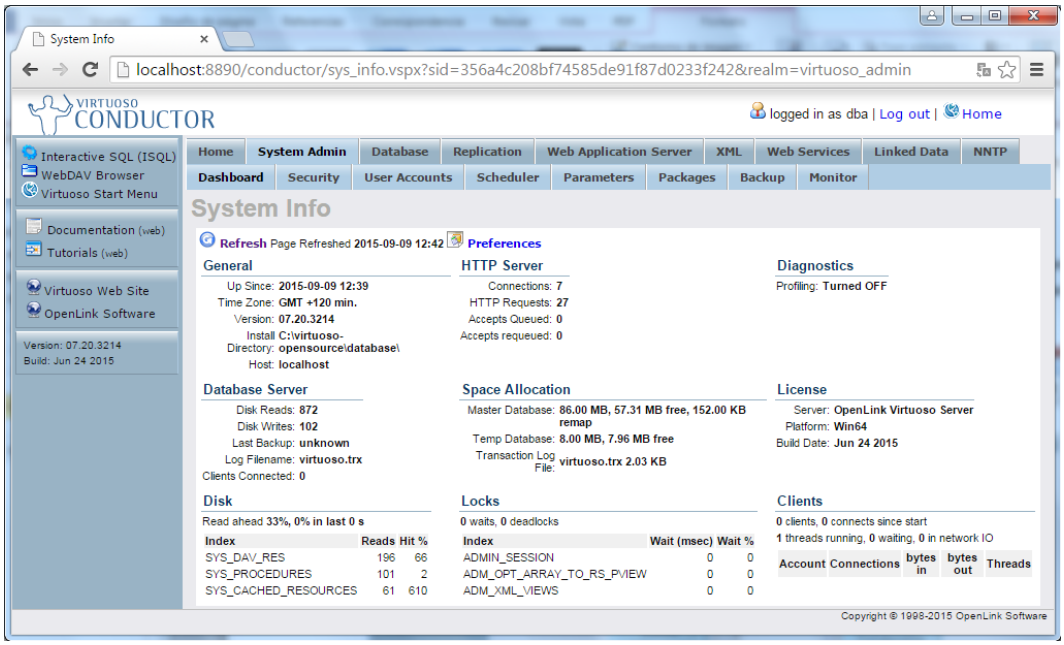


Figura A.4: Panel de configuración avanzada de *Virtuoso*.

[6]URL de acceso a Conductor: http://<nombre_del_servidor>:8890/conductor.

Configuración por defecto

A continuación, se presentan los valores por defecto de algunos parámetros imprescindibles para el acceso al panel de configuración avanzada de *Virtuoso*:

- URL: Si se está trabajando sobre la misma máquina, se puede utilizar **http://localhost/**. En caso de que el servidor esté en una máquina con nombre DNS, se puede acceder a través de dicho nombre.
- Puerto: 8890
- URL de Conductor: http://<nombre_servidor>:8890/conductor
- URL del punto de acceso SPARQL: http://<nombre_servidor>:8890/sparql
- Usuario administrador: dba
- Contraseña del usuario administrador: dba

Apéndice B

Análisis y diseño del sistema

En esta parte de los Anexos se incluye la información referente al análisis y diseño del sistema desarrollado en el TFG.

B.1. Diagramas de Secuencia de la aplicación

En esta sección se presentan algunos de los diagramas de secuencia de la aplicación, que modelan las relaciones entre el usuario y el sistema para poder realizar algunas de las acciones del sistema.

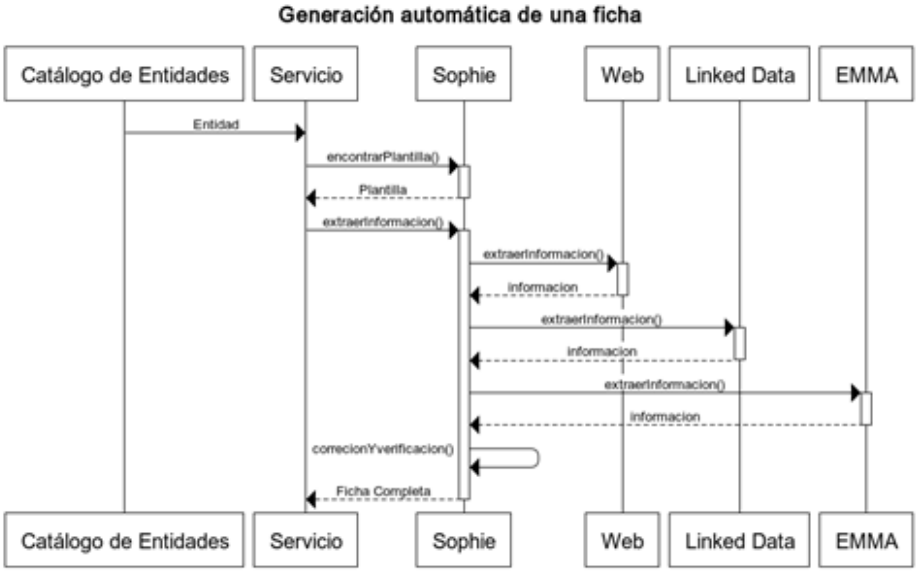


Figura B.1: Diagrama de secuencia que representa la simulación automática de una ficha a través del servicio de SOPHIE.

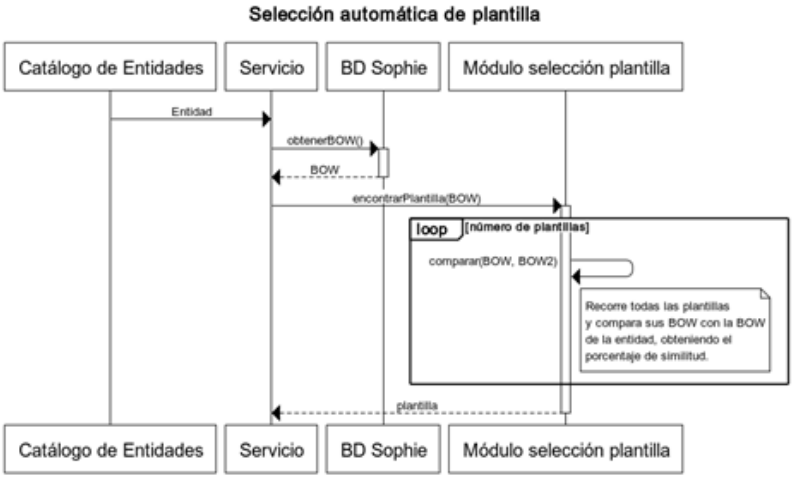


Figura B.2: Diagrama de secuencia que representa la selección automática de plantillas para generar las fichas de una *entidad*.

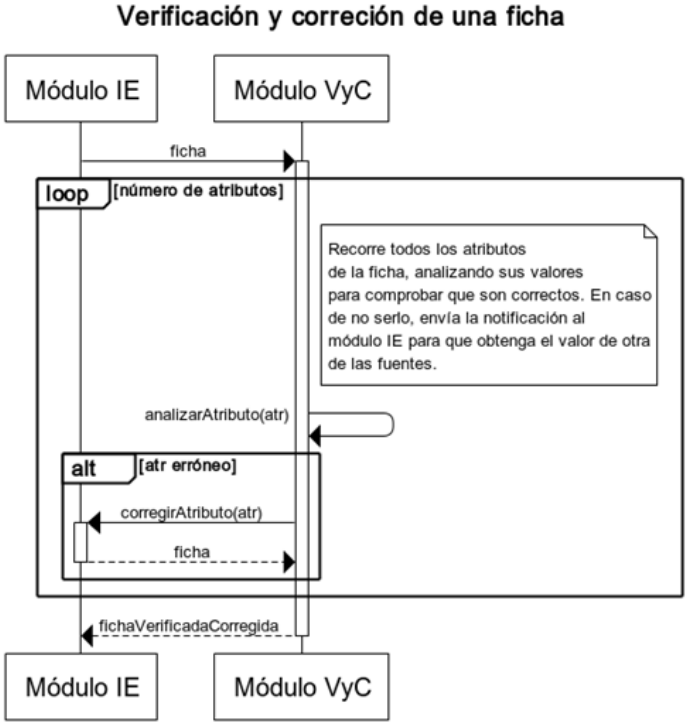


Figura B.3: Diagrama de secuencia que representa la verificación y corrección de una ficha.

B.2. Caso de Uso de la aplicación

En la figura B.4 se muestra el diagrama de casos de uso que muestra todos los casos de uso de la aplicación, con las relaciones entre ellos, los actores y los distintos sistemas.

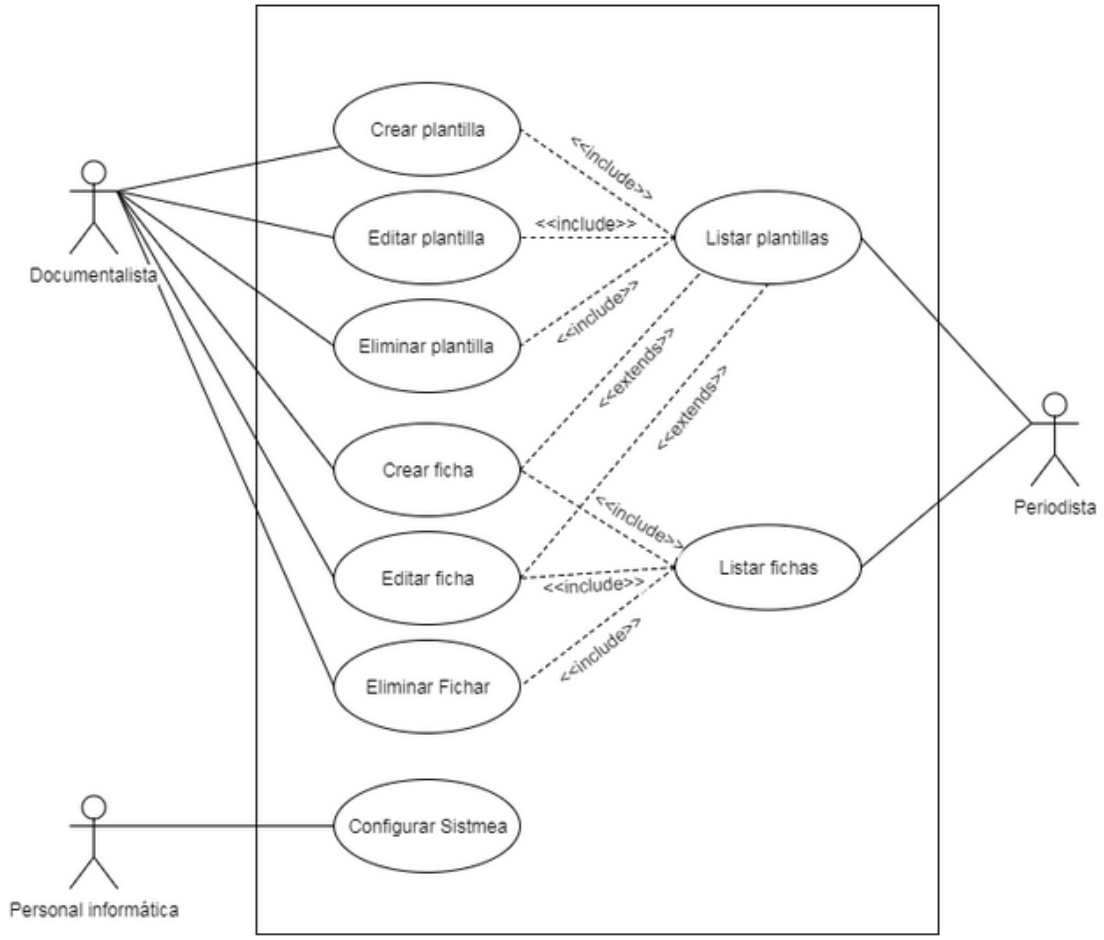


Figura B.4: Diagrama de casos de uso de la aplicación de escritorio.

B.3. Diagramas de Actividad de la aplicación

Para describir el proceso más complejo del sistema, que es el de creación de una nueva ficha, se ha realizado un esquema de actividad que se puede ver en la figura B.5.

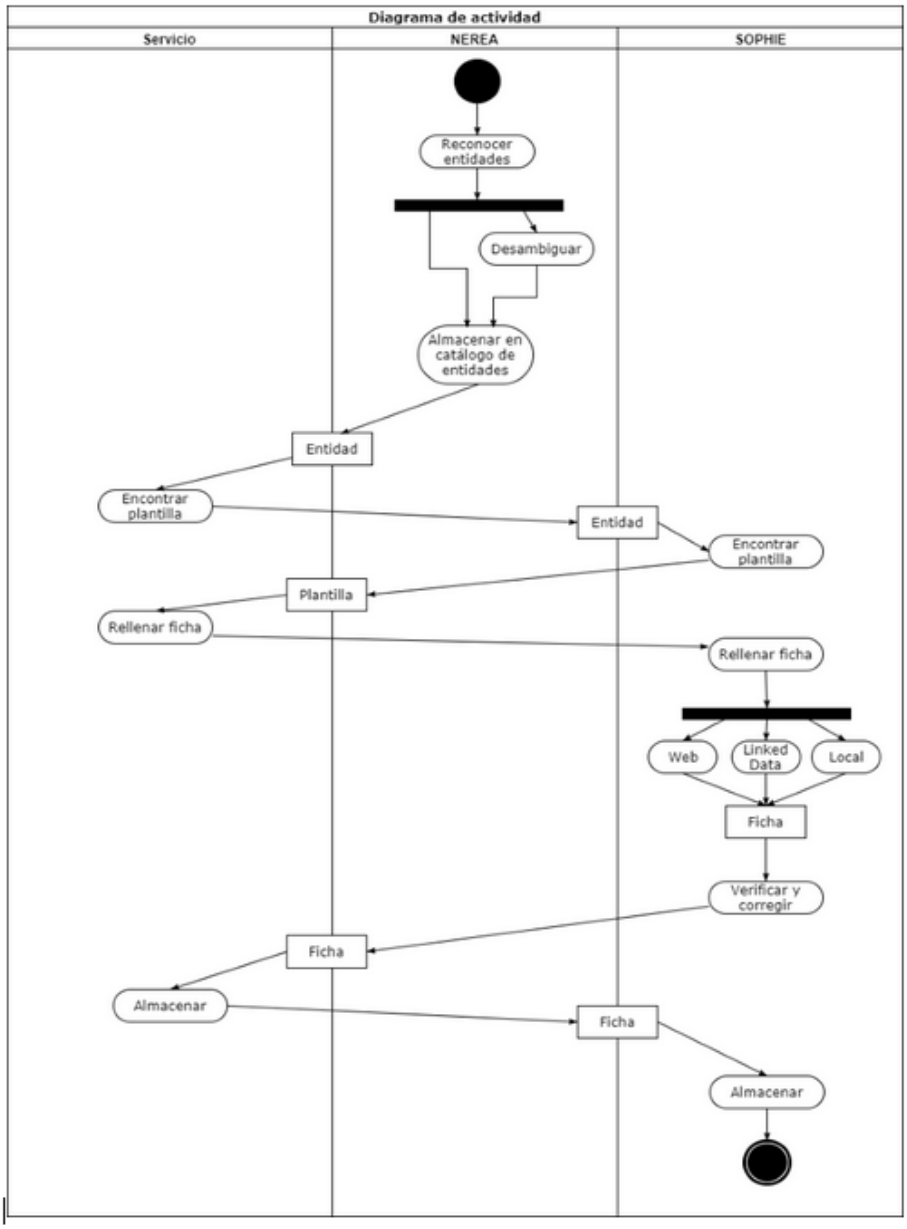


Figura B.5: Diagrama de actividad que representa el proceso completo de creación de una ficha.

B.4. Esquema de una ficha

En esta sección se describe con la ayuda de la figura B.6 la forma con la que se almacena en Virtuoso una nueva ficha. En este caso se quiere añadir la ficha de la entidad *Iker Casillas* al sistema.

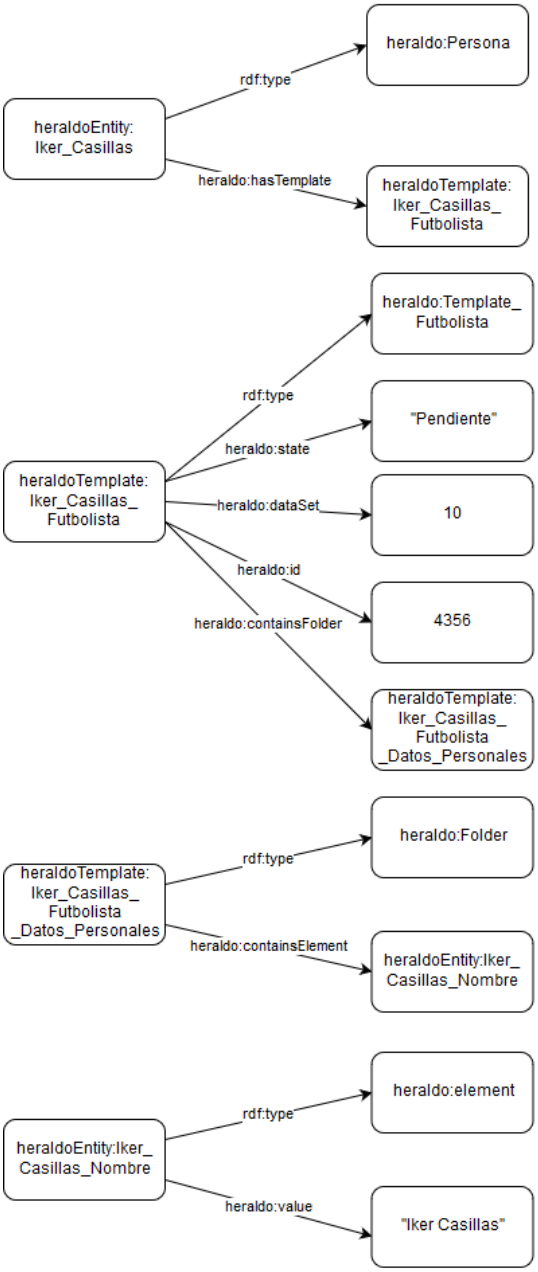


Figura B.6: Resumen de una ficha dentro del sistema.

Apéndice C

Desarrollo del sistema

C.1. Estructura

Una de las partes más importantes de este TFG es la mejora tanto en la estructura de las plantillas como en la extracción de los datos para rellenar estas.

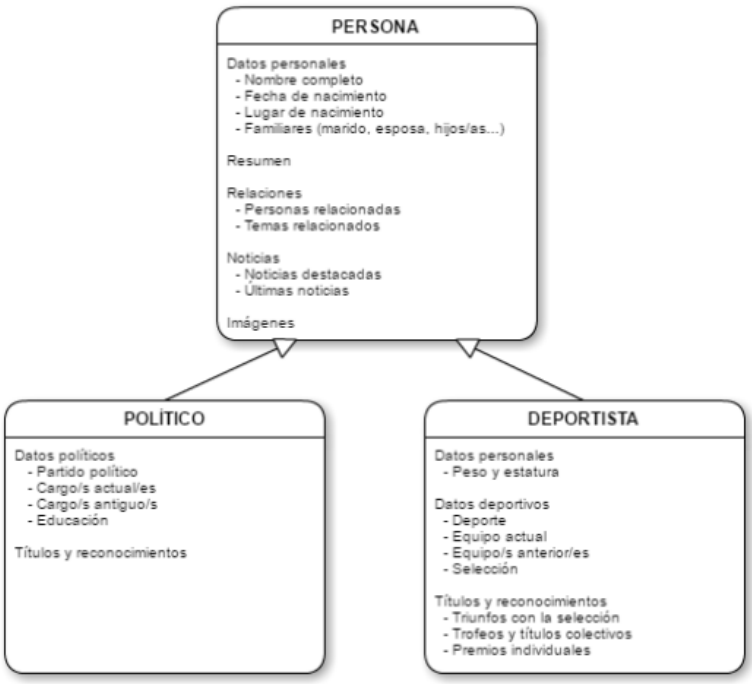


Figura C.1: Plantillas Persona y la herencia de las plantillas Político y Deportista.

Teniendo en cuenta las necesidades de la empresa y analizando las noticias del periódico se llega a conclusión de que la mayoría de artículos diarios tratan sobre entidades políticas y deportistas, por lo tanto, a lo largo de este TFG se pondrá especial énfasis en la mejora del funcionamiento del sistema sobre estas entidades.

Otra de las plantillas que se encuentran incompletas en el sistema es la de "Tema". Los atributos son correctos sin embargo falta relacionar los temas con las noticias y personas para que sea más útil la información de la entidad. Por lo tanto, se van a añadir los atributos Personas relacionadas con el tema y Noticias + personas relacionadas. Para comprender mejor estos atributos se usa el siguiente ejemplo: se tiene el tema "Caso Gürtel". Las personas relacionadas podrían ser Luis Bárcenas, Francisco Correa, Mariano Rajoy, etc... Para cada una de estas aparecen las últimas noticias que las relacionan con el tema, por ejemplo, en el caso de Rajoy, las últimas noticias sobre su futura testificación en el caso.

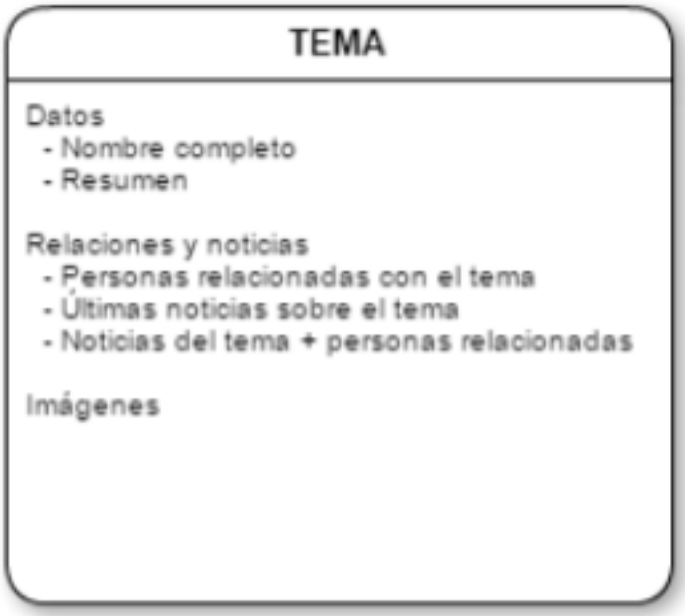


Figura C.2: Plantilla "Tema".

C.2. Extracción de información

Uno de los principales pilares del TFG es el conseguir extraer información de calidad de todas las fuentes de datos disponibles para poder rellenar las fichas de las entidades. Uno de los puntos débiles del sistema es que la información que se obtiene pertenece únicamente a los datos locales de la empresa y a la DBpedia. A pesar de que estas fuentes tienen una gran cantidad de datos, esta no es suficiente para muchas de las entidades. Para mejorar la calidad de la información de las fichas es fundamental mejorar el sistema para que este sea capaz de obtener información de la Web. Las principales fuentes de información en internet son Google y Wikipedia[18][?].

C.2.1. Google

Tiene a disposición de los usuarios una gran cantidad de APIs con las que se puede obtener información de la Web. En este caso se van a utilizar tres de ellas: *"Google Places API"*, *"YouTube Data API"* y *"Google Cloud Translation API"*.

```
String api = "AIzaSyBLJAkq5WisEIqS5hOmMuDVmrYFfC8Zgyk";
String url = "https://maps.googleapis.com/maps/api/place/details" +
    "/json?reference=CmRYAAAAciqGsTRX1mXRvuXSH2ErwW-jCINE1aLiwP64M" +
    "CWDN5vkXvXoQGPKldMfmdGyqWSpm7BEYCgDm-iv7Kc2PF7QA7brMAwBbAcqMr" +
    "5i1f4PwTpaovIZjysCEZTry8Ez30wpEhCNCXpynextCld2EBsDkRKsGhSLayuRyFsex6JA6NPh9dyupoTH3g&key=";

// ------------------------------RECIBIR JSON FUNCIONA
Stream webStream;
String webResponse = "";
HttpWebRequest req;
HttpWebResponse res;
req = (HttpWebRequest)WebRequest.Create(url + api);
req.Method = "GET";
res = (HttpWebResponse)req.GetResponse();
webStream = res.GetResponseStream();
StreamReader sr = new StreamReader(webStream);
while (sr.Peek() >= 0)
{
    webResponse = sr.ReadToEnd();
}

tbRespuesta.Text = webResponse.Length + " ---------------------- " +
    webResponse;
```

Figura C.3: Ejemplo de obtención de la información de las oficinas de Google en Sydney.

La primera de ellas, *"Google Places API"* es una de las más útiles para sacar información y fotografías sobre un lugar en concreto (ciudad, restaurante, monumento, etc...). Esta permite al usuario hacer una búsqueda (en un radio indicado) de lugares indexados en google con el nombre o texto que se pasa como parámetro. Una vez se obtienen las IDs de estos lugares, se puede conseguir información como un breve resumen, horario de apertura, teléfono, fotografías, comentarios de la gente, etc...

La segunda de las APIs llamada *"YouTube Data API"* permite consultar de forma similar a la API anterior datos e información sobre todos los canales de YouTube. El usuario puede realizar una búsqueda de canales que devuelve una lista con todos los canales que cumplen las condiciones. Una vez se tiene la ID del canal se pueden consultar todos los datos de este, como una pequeña descripción, fecha de creación, seguidores, comentarios, etc...

Por último se utiliza *"Google Cloud Translation API"* para poder traducir información de la mayoría de los idiomas. Se utiliza principalmente para traducir datos obtenidos de la web en inglés al español. La mayoría de la información que se encuentra por internet está en inglés, por lo tanto para poder rellenar las fichas de algunas entidades la mejor opción es buscar la información tanto en la web como en la DBpedia inglesa y después traducir esta al castellano. Si se quisiese obtener directamente la información en español, la mayoría de veces sería imposible (ya que no se encuentra) o la información sería muy escasa. También se puede utilizar para buscar información de una entidad en su idioma natal, en el caso de las personas, donde se encontrará por norma general más información que en otros idiomas. Esta API únicamente se va a utilizar para traducir datos como los infoboxes de la Wikipedia o grupos de palabras pequeños, ya que la traducción que realiza no suele ser correcta para grandes cantidades de información.

C.2.2. Wikipedia

La Wikipedia es una enciclopedia libre y editada colaborativamente. Actualmente contiene más de 37 millones de artículos en 287 idiomas que han sido redactados conjuntamente por voluntarios de todo el mundo, lo que le ha convertido en la mayor fuente de información de la Web. Todo esto hace de Wikipedia una pieza fundamental en el TFG para la obtención de información de las distintas entidades. Una de las características más útiles para obtener la información es el estudio de los infoboxes, una manera de presentar la información de una entidad al usuario de forma resumida y estructurada. Un ejemplo de estos se puede observar en la siguiente figura:

Figura C.4: Infobox de Wikipedia sobre Cani, jugador de fútbol del Real Zaragoza.

Para la creación de estos infoboxes, Wikipedia tiene una lista de plantillas predeterminadas que pueden ser editadas por los usuarios para que esta se adecue mejor a la entidad para la que se quiere crear el infobox. Entre estas plantillas predeterminadas se encuentran las de Ficha de autoridad y Ficha de deportista, entre muchas otras, que permiten a los usuarios crear fácilmente infoboxes para políticos y deportistas.

Para obtener toda la información sobre una página de la Wikipedia se usa la dll *LINQ-To-WIKI* para *Visual Basic* que permite obtener todos las datos en cualquiera de los idiomas. Una vez se obtiene la página entera, lo que se quiere

es extraer toda la información de los infoboxes. Para ello, se utilizan distintas expresiones regulares dependiendo del tipo de entidad de la que queramos extraer el infobox.

```
{{Ficha de deportista([^}}]*\n|(.*{{.*}}.*\n))*}}\n
```

Figura C.5: Ejemplo de expresión regular para obtener el infobox de un deportista en la Wikipedia.

```
String patron = "{{Ficha de deportista([^}}]*\n|(.*{{.*}}.*\n))*}}\n";
var pages2 = wikiEspañol.Query.allpages().Where(p => p.prefix == "Lionel_Messi").Pages
    .Select(
            p =>
            new
            {
                title = p.info.title,
                text = p.revisions().FirstOrDefault()
            })
        .ToEnumerable();
var arraypages2 = pages2.ToArray();
Regex r = new Regex(patron, RegexOptions.IgnoreCase);
String texto = arraypages2[0].text.ToString();
String fichaMessiIngles = r.Match(texto).ToString();
tbPagina.Text = fichaMessiIngles;
```

Figura C.6: Código de ejemplo para la extracción de la información sobre Lionel Messi en la Wikipedia.

C.3. Correción y verificación de errores

La generación automática de fichas del sistema genera varios errores que se dividen en tres tipos:

- **Errores por datos incompletos:** el sistema encuentra el dato que corresponde al campo pero el dato que devuelve está incompleto.
- **Datos inexistentes:** el sistema no encuentra el dato que corresponde al campo indicado.
- **Datos erróneos:** el sistema encuentra un dato para el campo pero este dato es erróneo.

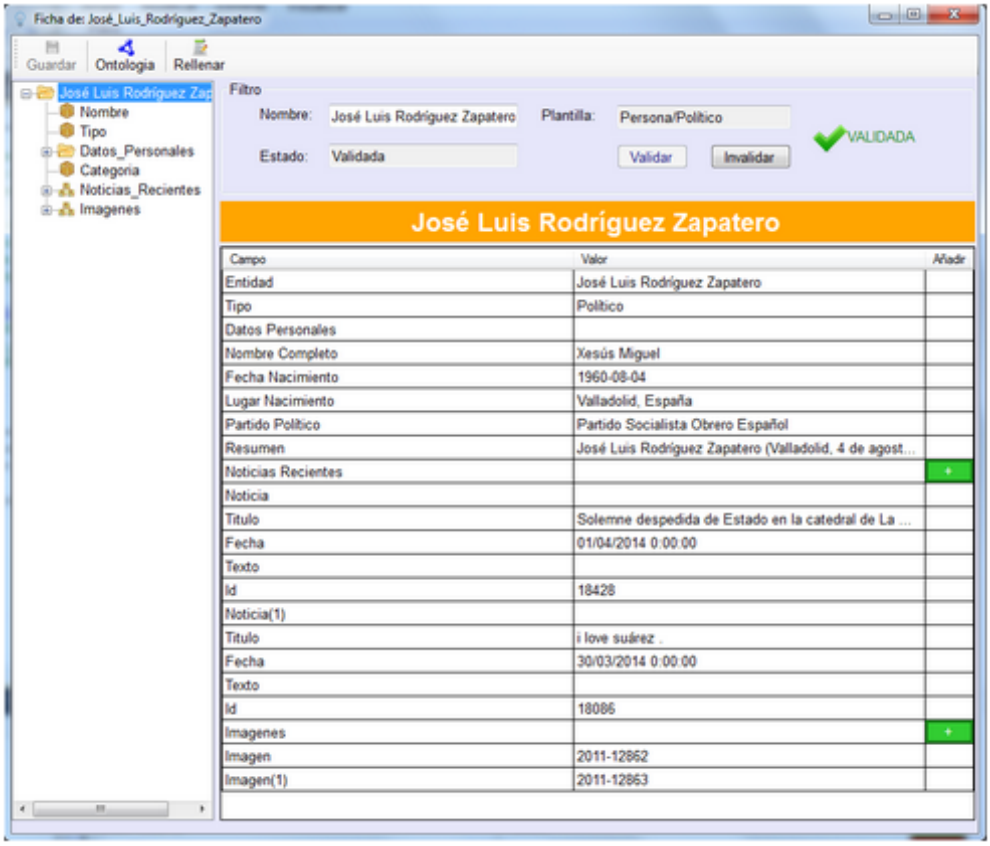


Figura C.7: Ejemplo de error en la entidad José Luis Rodríguez Zapatero. El campo nombre completo es erróneo.

Uno de los objetivos es el implementar un sistema de corrección y verificación de estos errores que, automáticamente los detecte y encuentre una solución para estos. Una parte muy importante de la implementación de este sistema es el avance que se ha nombrado en los puntos anteriores sobre la obtención de la información a partir de fuentes en la Web, ya que ahora el sistema obtiene los datos de tres fuentes distintas: EMMA (base de datos local), DBpedia y la Web. Cuantas más fuentes de información tenga el sistema, muchos más datos hay para verificar y corregir los errores.

Esta nueva función del sistema entrará en funcionamiento una vez se haya rellenado la ficha por completo. Cuando este proceso haya terminado, el sistema de corrección y verificación analizará la calidad de los datos en busca de errores. Para ello, se intentará obtener y comparar la información obtenida para un mismo campo de la ficha entre las distintas fuentes de información.

Si se comprueba que la mayoría de estas fuentes coinciden, el dato será el correcto. En caso contrario, se identificará que tipo de error es y se intentará buscar una solución con toda la información obtenida.

Para mejorar el sistema de corrección y verificación de errores se pueden asociar a cada una de las fuentes de información distintos grados de fiabilidad. Por ejemplo, si la ficha que se está generando corresponde a un político aragonés, por norma general, la información más completa y fiable será la proveniente de la base de datos local del *Heraldo de Aragón*. Sin embargo, si se está generando una ficha sobre un futbolista extranjero, la Wikipedia presenta la información más completa y fiable.

El sistema deberá de ser capaz de encontrar para cada entidad cuales son las fuentes más fiables a la hora de corregir errores, y para cada una de estas entidades asignará un peso distinto a cada una de las fuentes dependiendo de la fiabilidad de sus datos.

C.4. Sistemas gestores de base de datos NoSQL

Para poner en marcha el sistema es necesario disponer de un gestor de base de datos que se adapte a este. En esta sección se explica el proceso de elección de un SGBD para el sistema, partiendo de un primer análisis realizado anteriormente por la empresa para desarrollar el prototipo previo no funcional del sistema.

Debido a las necesidades del nuevo sistema, se valoran varias opciones de gestores de bases da datos NoSQL[5] de tipo grafo o de tripo RDF[3]. Algunas de las características de las que debe disponer son las siguientes:

- El SGBD NoSQL debe ser ejecutable sobre sistemas *Windows* ya que toda la arquitectura de la empresa se basa en este entorno.
- Los requisitos físicos no deben ser excesivos, ya que, como se ejecutará sobre un servidor virtual, existen restricciones respecto a ciertas características.
- El SGBD se debe poder manejar desde la plataforma .NET.

Para llevar a cabo la elección de un SGBD se ha tomado como referencia las clasificaciones proporcionadas por DB-Engine[1] centrándose en los principales productos de cada una de las dos categorias estudiadas, y el análisis previo realizado por la empresa. Así pues mediante un proceso de descarte se han obtenido las dos opciones que mejor se adaptan al proyecto.

[1] Clasificaciones DB-Engine: http://db-engines.com/en/ranking.

1. En una primera iteración se planteaba la utilización de dos tipos de SGBD: orientados a grafo y de tipo RDF. Debido a las necesidades del sistema, se ha considerado que los SGBD orientados a grafo son demasiado complejos, por lo tanto se pueden descartar.
2. Una de las principales características que debe cumplir es que debe ser ejecutable sin problemas en un entorno de trabajo *Windows*. Así pues, se descartan por completo todos los SGBD que son únicamente ejecutables en *Linux* o *MAC*.
3. Se descartan todos los SGBD que están orientados a mercados en concreto ya que es posible que no se adapten a las necesidades del proyecto.
4. Teniendo en cuenta todos los puntos anteriores, quedan disponibles tres SGBD: **MarkLogic**, **Virtuoso** y **Stardog**.
5. **MarkLogic** es uno de los SGBD más caros, ya que una licencia para producción de este software cuenta entre 18.000 y 32.000 euros, por lo tanto queda completamente descartado.
6. Por lo tanto, sólo nos quedan dos SGBD: uno de ellos, **Virtuoso**, conocido por ser utilizado por organizaciones como DBPedia[2], y por otro lado, **Stardog**, es un SGBD bastante reciente.

Para cada uno de estos dos SGBD se ha creado una tabla con las características principales, para así poder comprarlos y elegir el que más se adapta al proyecto. Estas tablas contienen las siguientes características:

- Tipo: Hace referencia al tipo de base de datos. Éstas pueden ser de distintos tipo: orientadas a documentos, RDF, XML, Grafos, etc.
- Plataforma: Sistemas en los que se puede ejecutar la aplicación: Windows, Linux, Mac, Cloud, etc.
- Lenguajes de consulta: Hace referencia al lenguaje utilizado para consultar la base de datos.
- Lenguajes de programación: Hace referencia a los lenguajes para los cuales hay disponibles librerías de acceso. Si dispone de librería para .NET únicamente se nombrará esta.
- Licencia y precio.
- Requisitos físicos: Se valorará positivamente los SGBD que no requieran una gran cantidad de recursos físicos.
- Observaciones: Cualquier dato que sea necesario tener en cuenta.

[2]DBPedia: http://dbpedia.org

C.4.1. Virtuoso OpenLink

Es uno de los SGBD NoSQL de los más conocidos a nivel global. Ha sido desarrollado por la compañía **OpenLink Software**. Es lo se conoce como un servidor universal, ya que que además de ofrecer el almacenamiento de datos, también dispone de características como servidor de aplicaciones o servidor de ficheros e implementa diversos protocolos. Además de la versión comercial, está disponible un proyecto abierto llamado **OpenLink Virtuoso** lo que ha permitido que este SGBD sea ampliamente utilizado por varias universidades. En la tabla C.1, se puede ver la ficha comparativa correspondiente a este SGBD.

Tabla C.1: Ficha de características de *Virtuoso*

Característica	**Descripción**
Tipo	Relacional, RDF, XML
Plataforma	Multiplataforma
Lenguajes de consulta	SPARQL
Lenguajes de programación admitidos	.NET, C#
Licencia	Versión OpenSource: Libre Versión Completa: Comercial
Precio	Libre: Gratuito Comercial: entre 100 y 5.000 euros
Requisitos físicos	Dependen de la licencia
Observaciones	

C.4.2. Stardog

Este SGBD fue desarrollado por **Complexible Inc** en 2012. Al ser tan reciente, tiene algunas características que otros no poseen como puede ser el soporte para la versión 1.1 de SPARQL. En la tabla C.2, se puede ver la ficha comparativa de este SGBD.

Tabla C.2: Ficha de características de *Stardog*

Característica	**Descripción**
Tipo	RDF
Plataforma	Multiplataforma
Lenguajes de consulta	SPARQL
Lenguajes de programación admitidos	.NET, Java
Licencia	Desarrollo: Libre Producción: Comercial
Precio	Libre: Gratuito Comercial: Es necesario ponerse en contacto con los desarrolladores
Requisitos físicos	
Observaciones	

C.4.3. Elección de un SGBD NoSQL

Una vez analizados los dos SGBD nombrados, y teniendo el cuenta el análisis llevado a cabo en el desarrollo del primer prototipo del sistema, se llega a la conclusión de que la mejor opción es mantener la elección de Virtuoso como SGBD. Esto se debe a varios motivos:

- Coste de implementación: al disponer de una primera versión los costes son bastante bajos.
- Coste de aprendizaje: la empresa ya disponía de conocimiento previo sobre el SGBD.
- Se adapta al sistema: su modelo de grafos se adapta perfectamente a las necesidades del nuevo sistema.
- Coste de físico: su uso requiere de poco coste físico y por lo tanto se puede utilizar en un servidor virtual.
- Funciona sin problemas en sistemas *Windows*.
- Se puede manejar desde la plataforma *.NET*.

C.5. Diseño futuro

En un futuro la visualización de las fichas se realizará utilizando una plantilla HTML diseñada por profesionales. Esta plantilla cumple los estándares de diseño de la empresa y ha sido aceptada por todos los actores identificados en el proyecto. La plantilla contiene una cabecera donde se muestran los datos más importantes de la entidad, como puede ser el nombre, ciudad de nacimiento, etc.

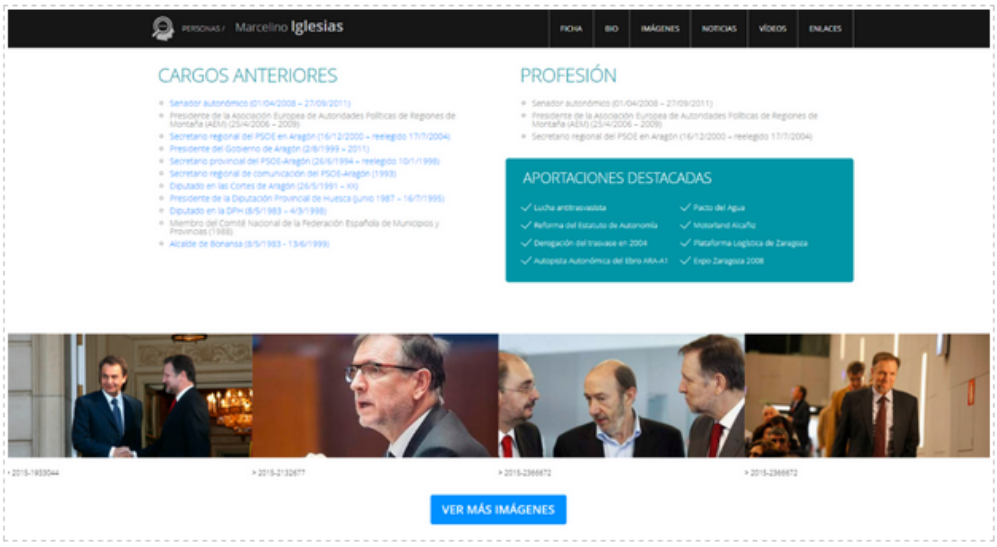


Figura C.8: Cuerpo de una ficha en la versión futura del sistema.

Seguido, en el cuerpo de la plantilla, aparecen los datos específicos de la entidad, junto a un breve resumen de esta.

Figura C.9: Cuerpo de una ficha en la versión futura del sistema.

Figura C.10: Cuerpo de una ficha en la versión futura del sistema.

Por último, aparecen las noticias destacadas y las últimas imágenes de esta entidad en el repositorio local de datos.

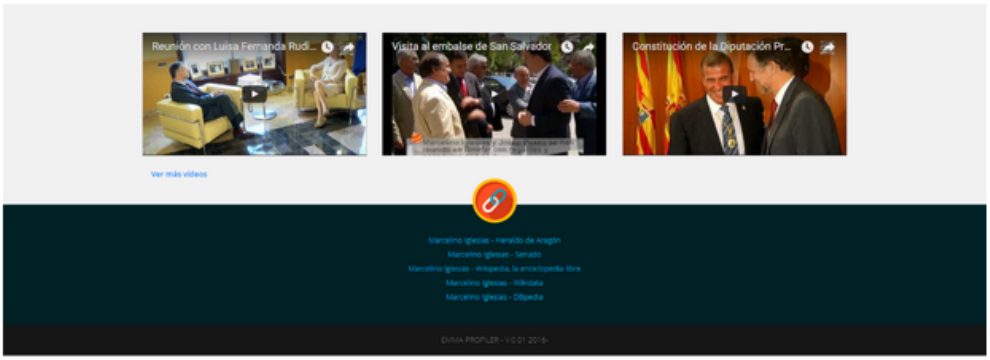


Figura C.11: Parte final de una ficha en la versión futura del sistema.

Apéndice D

Ampliaciones

D.1. Trabajo futuro

En esta sección se detalla por completo el trabajo futuro planeado para mejorar el sistema desarrollado.

D.1.1. Selección automática de la plantilla

El sistema encargado de generar las fichas, llamado SOPHIE, se integra con NEREA, la aplicación de reconocimiento de entidades y desambiguación de la empresa. NEREA se encarga de reconocer las entidades más relevantes de textos locales almacenados en la base de datos de la empresa y de realizar la desambiguación con la ayuda del subsistema POIROT. El funcionamiento es el siguiente: NEREA analiza un texto local y detecta la aparición de las distintas entidades. Para cada una de las entidades, comprueba si ya está en el *Catálogo de Entidades*. En caso de no existir ninguna entidad con ese nombre, NEREA recoge toda la información posible de las distintas fuentes de datos y añade esta nueva entidad al *Catálogo de Entidades*. Al contrario, si ya existen varias entidades en el catálogo con el mismo nombre, POIROT se encarga de detectar a cuál de estas entidades se refiere el texto. Para ello, se comparan los conjuntos de palabras almacenadas ("bag of words") para cada una de las entidades con el conjunto de palabras del texto analizado, seleccionando la que más coincidencia tenga. Al finalizar todo este proceso, el sistema ya ha sido capaz de detectar y diferenciar qué entidad se está nombrando en el texto.

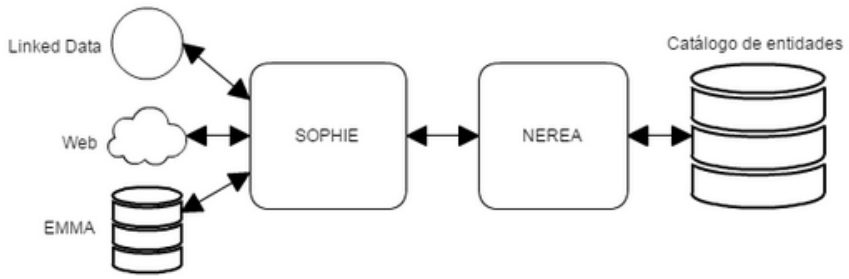


Figura D.1: Esquema simple del sistema. SOPHIE se encarga de la generación de las fichas para las entidades, las cuales se obtienen a través de NEREA y su catálogo de entidades.

A la hora de generar una nueva ficha para una entidad en el sistema actual, es necesario que los documentalistas seleccionen la plantilla a la que pertenece o con la que más se representa esta entidad, si esta no pertenece a ninguno de los cuatro tipos básico: Persona, Lugar, Organización y Tema. Si se quieren generar muchas fichas que no pertenecen a las plantillas básicas, esta tarea puede convertirse en algo molesto, por lo tanto, es necesario que el sistema sea capaz de detectar automáticamente la plantilla de una nueva entidad.

Actualmente, si se desea crear una nueva ficha, por ejemplo, para Rubén Gracia Calmache (jugador del Real Zaragoza), si se genera automáticamente el sistema detectará que es una entidad persona por lo tanto utilizará la plantilla Persona. La selección es correcta, sin embargo, en la plantilla Persona no están algunos de los atributos más utilizados para un futbolista como pueden ser: equipo actual, títulos, partidos jugados, etc... Lo correcto sería que el sistema fuese capaz de detectar que, además de ser una persona, este es un jugador de fútbol, por lo tanto, la mejor plantilla para generar su ficha sería la de Deportista.

Para conseguir que el sistema identifique de forma automática la plantilla para generar la ficha se utilizará la siguiente metodología. Una vez NEREA ha detectado una nueva entidad y ha realizado la desambiguación, se debe identificar la plantilla con la que más se relaciona. Para ello, se comparan los documentos más relevantes donde aparece la entidad con un conjunto de documentos modelo pertenecientes a cada una de las plantillas. La plantilla con la que mayor similitud tenga, será la que se utilizará para crear la ficha para esta entidad. El conjunto de documentos modelo pertenecientes a cada una de las plantillas serán los documentos que corresponden a las entidades pertenecientes a la plantilla específica. Es decir, si el sistema sabe que Mariano Rajoy y Pablo Iglesias son políticos, a la hora de comparar una nueva entidad para saber si esta pertenece a la plantilla políticos, comparará sus documentos con los de Mariano Rajoy y Pablo Iglesias.

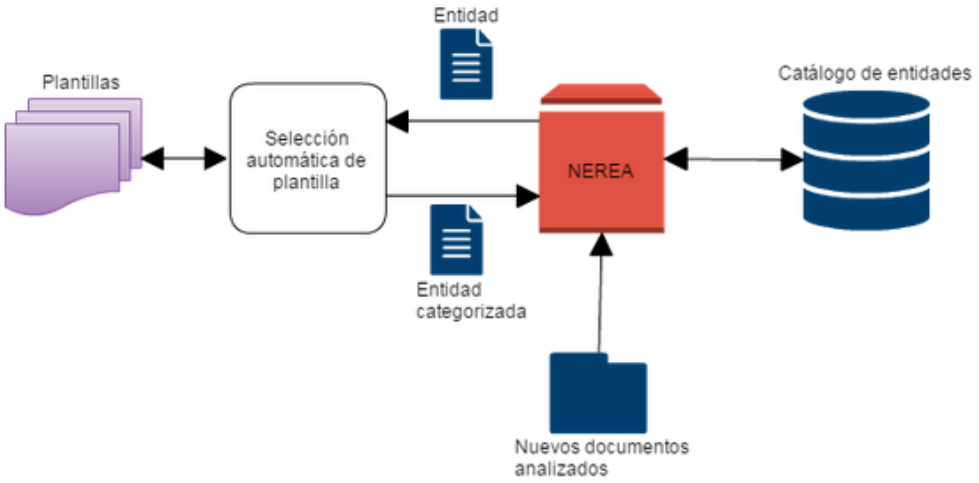


Figura D.2: Esquema que muestra el funcionamiento del sistema para seleccionar una plantilla automáticamente.

Con el objetivo de que el sistema funcione correctamente desde un primer momento, se seleccionaran varias entidades de ejemplo para cada una de las plantillas del sistema, junto a su conjunto de documentos modelo. Cuantas más entidades de ejemplo se tomen y mayor sea su número de documentos, mejor funcionará el sistema.

Para comprobar que el sistema está funcionando correctamente se pueden utilizar los infoboxes de Wikipedia[19]. Esta tiene distintas plantillas para generar los infoboxes, y al obtener la información de estos te indica el nombre de la plantilla sobre la que se basa. Por lo tanto, bastaría con comprobar el nombre de la plantilla que utiliza una entidad en Wikipedia para comprobar que el sistema automático ha categorizado correctamente la entidad.

D.1.2. Inferencia de datos

Actualmente el sistema consta de una base de conocimiento[9][10][17] implementada para almacenar fichas de datos, sin embargo el sistema de fichas únicamente almacena los distintos campos de las fichas junto a su valor. Uno de las tareas más interesantes es añadir una mayor funcionalidad al módulo de extracción de información para que este sea capaz de conectar la información de las nuevas fichas con las que ya están recolectadas, creando así una interconexión entre todas las fichas del sistema.

Para implementar esta mejora en el sistema es necesario desarrollar un nuevo módulo de inferencia que trabaje basándose en una serie de reglas creadas ya anteriormente que le ayuden a deducir nuevos datos en la base de conocimiento. Este

módulo de inferencia está compuesto por dos elementos: intérprete de reglas que es el mecanismo de razonamiento que determina qué reglas de la base de conocimiento se pueden aplicar, y la estrategia de control encargada de la resolución de conflictos.

Para obtener las conclusiones y conexiones se van a utilizar diferentes tipos de reglas:

1. **Modus Ponens:** Es la regla de inferencia más común y sirve para obtener conclusiones simples. Se estudia la premisa de la regla y, si esta es cierta, la conclusión pasa a formar parte del conocimiento. "Si A es cierto, entonces B es cierto" – "A entonces B".
2. **Modus Tollens:** Al igual que la regla anterior, esta sirve para obtener conclusiones simples. En este caso, primero se estudia la conclusión y si es falsa, se concluye que la premisa también lo es. "Si A es cierto, entonces B es cierto" – "B es falso, entonces A es falso".

Unido a distintos tipos de estrategias de inferencia:

1. **Encadenamiento de Reglas:** Es una de las estrategias de inferencia más utilizadas. Esta estrategia se utiliza cuando las premisas de algunas reglas coinciden con las conclusiones de otras. Si estas reglas se encadenan, los hechos pueden utilizarse para obtener nuevos hechos. Esto se repite hasta que no se pueden obtener más conclusiones.
2. **Encadenamiento de Reglas orientado a un objetivo:** La principal característica de este algoritmo es que necesita que el usuario seleccione el nodo objetivo, una vez elegido el algoritmo navega a través de las reglas del sistema para encontrar una solución para el nodo objetivo. Si el algoritmo no encuentra ninguna solución con la información dada por el usuario, este vuelve a preguntarle en busca de más información.

Este trabajo se centrará en estudiar la implementación de este módulo para la inferencia de los datos en las relaciones entre las personas. Es decir, el sistema será capaz de detectar, entre otras relaciones, que: si A es hijo de B, y C es hijo de B, entonces A y C son hermanos.

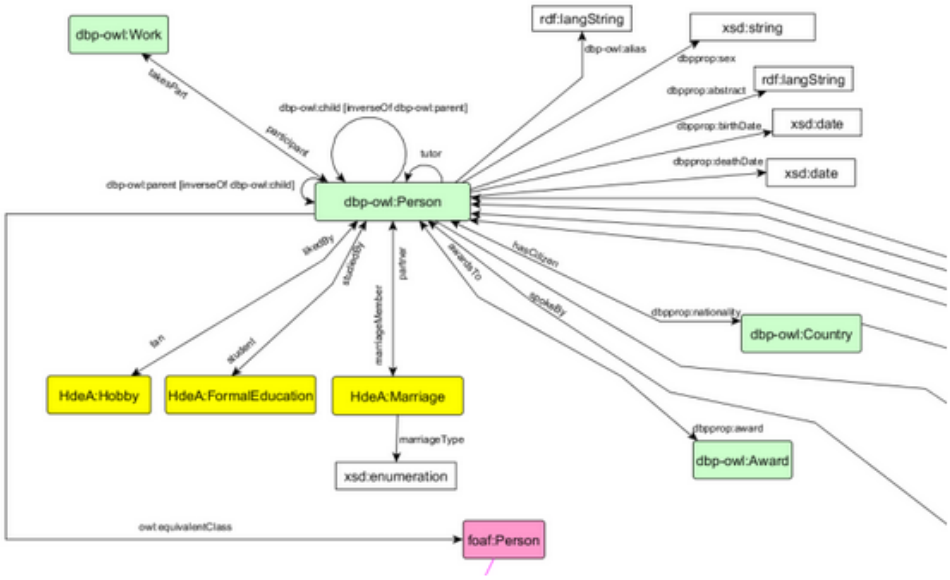


Figura D.3: Ontología de Persona utilizada en el sistema

Como se puede observar en la figura anterior, la ontología utilizada tiene la relación marriageMember que relaciona dos personas como marido y mujer, además permite indicar qué número de matrimonio es, por si una persona se ha casado varias veces. A partir de esta relación, el sistema será capaz de detectar que: si A es la mujer de B, entonces B es el marido de A. También se puede observar la relación child/parentque relaciona al padre con su hijo y viceversa.

A partir de estas relaciones, y utilizando reglas y estrategias mencionadas en este mismo punto, el sistema debería de ser capaz de detectar relaciones entre personas más complejas, como la de la siguiente figura.

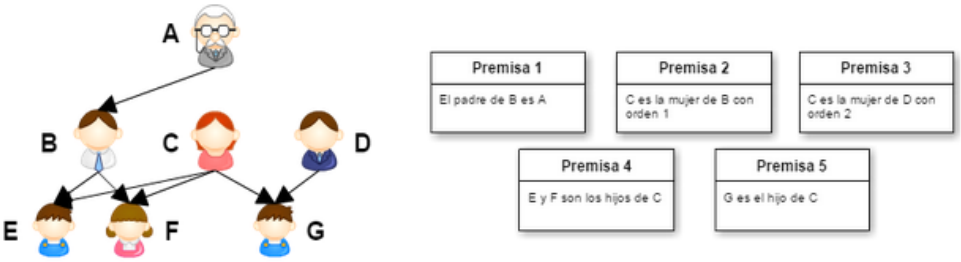


Figura D.4: Ejemplo complejo de relación entre personas.

A partir de las premisas que aparecen en la figura XIII el sistema deberá ser capaz de obtener las siguientes conclusiones:

1. La ex-mujer de B era C.
2. La mujer de D es C.
3. El abuelo de E y F es A.
4. E y F son hermanastros de G.

Para mejorar el funcionamiento del módulo de inferencia será necesario añadir nuevas relaciones en la ontología del sistema. Este trabajo se va a centrar, sobre todo, en relacionar a las personas según sus relaciones familiares. Para que el sistema sea capaz de detectar y analizar el ejemplo anterior, será necesario que la entidad Person de la ontología tenga nuevas relaciones:

1. **Sister/Brother:** relaciona a una persona con su hermano o hermana.
2. **Cousin:** relaciona a una persona con su primo o prima.
3. **Grandfather/Grandmother:** relaciona a una persona con su abuelo o abuela.
4. **Grandson/Granddaughter:** relaciona a una persona con su nieto o nieta.
5. **Uncle/Aunt:** relaciona a una persona con su tío o tía.
6. **Nephew/Niece:** relaciona a una persona con su sobrino o sobrina.
7. **Father-in-law/Mother-in-law:** relaciona a una persona con su suegro o suegra.
8. **Son-in-law/Daughter-in-law:** relaciona a una persona con su yerno o nuera.
9. **Brother-in-law/Sister-in-law:** relaciona a una persona con su cuñado o cuñada.
10. **Otherfamilyrelationships:** relaciona a dos personas indicando que estas tienen algún tipo de relación lejana entre ellos, como por ejemplo, primos hermanos, primos segundos, bisabuelos, etc.