Universidad de León

**Departamento de Ingeniería Eléctrica
y de Sistemas y Automática**

Creación de un Framework para el Tratamiento de Corpus Lingüísticos

Development of a Framework for Corpus Linguistic Analysis

Hugo Sanjurjo González

León, junio de 2017

El problema con el mundo es que la gente inteligente
está llena de dudas, mientras que la gente estúpida
está llena de certezas.

Heinrich Karl Bukowski

A mis padres, que me dieron todas las oportunidades disponibles a su alcance.

A mis hermanos, por su apoyo incondicional.

A mis abuelos, por hacerme sentir capaz de conseguir cualquier cosa.

Al resto de mi familia, por su preocupación e interés.

A mis amigos.

Prefacio

La presente tesis doctoral se ha realizado en el programa "Sistemas Inteligentes de la Ingeniería" (R.D. 1393/2007) con colaboración del programa "Estudios Contrastivos y Comparados Inglés/Francés/Español" y del equipo de investigación ACTRES (Análisis Contrastivo y Traducción Inglés-Español).

La investigación doctoral ha ocupado el periodo comprendido entre marzo de 2012 y junio 2017, y ha sido financiada por la Universidad de León dentro del "Programa de Ayudas para la Realización del Doctorado en el Marco de Programa Propio de Investigación de la Universidad de León – Convocatoria 2014" entre los años 2015-2017.

El candidato doctoral también ha tenido acceso a los recursos y herramientas lingüísticos desarrollados en el marco de los proyectos de investigación con referencias FFI2013-42994-R y FFI2016-75672-R, financiados por FEDER y el Ministerio de Ciencia y Tecnología.

Agradecimientos

Quisiera expresar mi agradecimiento a todas las personas que me han ayudado, animado y apoyado durante la realización de esta tesis doctoral.

En primer lugar, a mis directores de tesis, la Dra. Rosa Rabadán y el Dr. Héctor Alaiz Moretón, sin sus indicaciones, consejos, correcciones, paciencia y ayuda, esta tesis doctoral no habría llegado a buen puerto.

A los investigadores del grupo ACTRES: agradecer la implicación de la Dra. Isabel Pizarro, por su ayuda para la obtención de recursos lingüísticos, de la Dra. Marlén Izquierdo, por los consejos derivados de la experiencia en la compilación de P-ACTRES, y por supuesto de Knut Hofland, desarrollador de P-ACTRES, cuyos consejos iniciales y trabajo realizado me sirvieron de modelo para mi investigación.

No quisiera olvidarme del resto de investigadores del grupo ACTRES: gracias por vuestras muestras de apoyo.

A la dirección de la Escuela de Ingenierías y del Instituto de Automática y Fabricación, Dr. Ramón Ángel Fernández-Díaz y Dr. Ángel Alonso que, además de proporcionarme un lugar para trabajar, se preocuparon por mi bienestar. Agradecimiento que hago extensivo al Departamento de Filología Moderna de la Universidad de León por acogerme en sus instalaciones durante la última fase de desarrollo de esta tesis doctoral.

A mis compañeros del Departamento de Ingeniería Eléctrica y de Sistemas y Automática, en especial al Dr. José Manuel Alija, Dr. Francisco Jesús Rodríguez-Sedano, Dr. Isaías García, Dra. Carmen Benavides y Dr. Luis Panizo, gracias por compartir vuestras experiencias y aconsejarme.

A *Lancaster University*, en particular UCREL, por su acogida durante mi estancia de investigación. Sobre todo, agradecer la ayuda recibida por mi tutor, el Dr. Paul Rayson, gracias al cual tuve la oportunidad de conocer a un gran número de reputados investigadores, y también al Dr. Scott Piao por sus consejos y directrices. También quisiera darles las gracias por proporcionarme permiso para utilizar el *English Semantic Lexicon* en mi investigación. Agradecer también la ayuda recibida de los compañeros de Lancaster, especialmente a los que a día de hoy puedo considerar mis amigos, Vicent y Argenis.

A todos los compañeros que he tenido el placer de conocer compartiendo lugar de trabajo en la Universidad de León, con los que viví momentos divertidos y pasé calor en agosto, especialmente a José, y también a Helia, Aitana y Oliver.

A todos aquellos que aun no siendo mencionados me han ayudado en la realización de la tesis durante estos años.

Deseo que esta tesis sea digna de todo el apoyo y ayuda recibidos.

I would like to express my gratitude to all those that have helped, encouraged and supported me in the elaboration of this dissertation.

Firstly, I would like to thank my supervisors, Dr. Rosa Rabadán and Dr. Héctor Alaiz Moretón, this Doctoral thesis would not have been completed without their guidance, advice, corrections, patience and help.

To the ACTRES Research Group: Thanks should go to Dr. Isabel Pizarro, from the University of Valladolid, for her generosity concerning the location of linguistic materials, to Dr. Marlén Izquierdo, from the University of the Basque Country, sharing her experience compiling P-ACTRES, and to Mr. Knut Hofland, from the University of Bergen (Uni Computing), developer of P-ACTRES, whose initial work in P-ACTRES and generous advice served me as a model for my own research.

I should not forget the rest of the ACTRES Research Group: thank you all for your support.

To the head of School of Engineering, Dr. Ramón Ángel Fernández-Díaz, and to the director of the Institute of Automation and Manufacturing, Dr. Ángel Alonso, for providing me with an stimulating working environment. I extend my thanks to the Department of Modern Languages, for housing me during the final stages of my research for this doctoral thesis.

To my colleagues at the Department of Electric, Systems and Automatics Engineering, particularly Dr. José Manuel Alija, Dr. Francisco Jesús Rodríguez-Sedano, Dr. Isaías García, Dr. Carmen Benavides and Dr. Luis Panizo, thanks for sharing your experiences and making greatly appreciated suggestions.

To Lancaster University, especially UCREL, thank you for your warm reception during my stay as a visiting researcher. Thanks to Dr. Paul Rayson, I had the opportunity to meet a large number of well-known researchers, and thanks Dr. Scott Piao for your advice and guidance. I must also thank both of them for granting permission to employ the English Semantic Lexicon in my research. I would also like to thank all those colleagues I had the opportunity to meet in Lancaster, especially Vicent and Argenis, whom today I can call my friends.

To all the colleagues that I had the pleasure to meet at the University of León, thanks for those shared hilarious moments in the midst of the August heat, particularly José, and also to Helia, Aitana and Oliver.

To all those who have not been mentioned here and helped me during these years.

I hope that this doctoral thesis is worthy of all the support and help I have received.

Abstract

A pesar de los indudables avances en el software para el tratamiento de corpus lingüísticos en los últimos tiempos, ya sea por medio de procesamiento de corpus cada vez más grandes o inclusión de estadísticas más complejas, sigue sin tenerse en cuenta la usabilidad y el perfil no técnico del usuario final. La situación resulta más evidente cuando se trabaja con lenguas distintas del inglés y con combinaciones de lenguas, ya que la tipología y especificidad de las mismas incide en los requisitos del software, y por este motivo la disponibilidad de recursos es menor y de peor calidad.

El estado de la cuestión revela que la creación de corpus lingüísticos bi-/multilingües paralelos o comparables, así como la incorporación de etiquetados lingüísticos en los frameworks para el tratamiento de corpus lingüísticos ya existentes, obliga al usuario a disponer de ciertos conocimientos de programación, o al menos a saber ejecutar programas con usabilidad reducida y/o scripts informáticos propios, para ajustar el corpus a los requisitos establecidos por el framework utilizado. Si no se dan estas condiciones, es indispensable contar con especialistas técnicos con habilidades en programación y NLP (por sus siglas en inglés *Natural Language Processing*).

El objetivo de la tesis doctoral es, por tanto, el desarrollo de un software, denominado *ACTRES Corpus Manager*, que permita a los usuarios lingüistas construir sus propios corpus lingüísticos (monolingües, paralelos bi-/multilingües o comparables) con distintas capas de anotación (gramatical, semántica o retórica) y obtener datos lingüísticos y estadísticos sin necesidad sin necesidad de asistencia técnica en ningún punto del proceso e independientemente de las habilidades técnicas del usuario.

La estrategia seleccionada para el desarrollo de *ACTRES Corpus Manager* es la creación de un framework accesible vía web formado por distintos componentes interconectados entre sí. Cada actividad necesaria para la creación de un corpus es asignada a cada uno de estos componentes, posibilitando su fácil modificación y reutilización. *ACTRES Corpus Manager* combina la utilización de recursos

software de terceros, cuya eficiencia y validez haya sido demostrada (ej. *The IMS Corpus Workbench*, Treetagger, hunalign, etc.), junto con soluciones software propias en aquellos procesos que el estado de la cuestión ha relevado más inmaduros y/o complejos de integrar (etiquetador retórico, etiquetador semántico, etc.).

Por último, señalar que la interfaz de consulta de *ACTRES Corpus Manager* se inspira en P-ACTRES 2.0 y permite la realización de consultas complejas asistidas, basadas en expresiones regulares, así como la extracción de las estadísticas habituales, sin necesidad de que el usuario disponga de conocimientos específicos de la sintaxis del lenguaje de consulta utilizado.

Despite the unquestionable advancement in corpus linguistics software in recent times, which includes supporting larger corpora and more complex statistics, it still fails to take into account is still failing to take into account the usability and profile of the end-users. This is more evident when the working language is not English or a combination of languages, as their typology and linguistic idiosyncrasies affect software requirements, making resource availability less reliable, both in quantity and quality.

A review of the state-of-the-art shows that the creation of monolingual, bi/multilingual parallel and comparable corpora as well as the incorporation of linguistic annotation layers in the current frameworks requires programming skills from the user. These include knowing how to execute computer programs with limited usability and/or custom programming scripts as to adapt the corpus to the specifications of a particular corpus analysis software. If this is not the case, technical advice from staff with programming and NLP skills is necessary.

The objective of this study is, therefore, the development of a framework, the *ACTRES Corpus Manager*, which allows users to create their own corpora (monolingual, bi/multilingual parallel and comparable) with several annotation

layers (grammatical, semantic and rhetorical), make linguistic queries and obtain the most common statistics without technical assistance during the process and regardless of the technical skills of the user.

ACTRES Corpus Manager has been designed as a web-accessible framework composed of several interconnected components. Each corpus-creating task is assigned to one component, a strategy which enables easy reuse and modification. *ACTRES Corpus Manager* combines the use of already existing software, whose efficiency and validity is well known (e.g. The IMS Corpus Workbench, Treetagger, hunalign, etc.), together with additional custom-built software for those processes that our state-of-the-art analysis has shown to be immature and/or difficult to integrate (e.g. rhetorical tagger, semantic tagger, etc.)

Finally, it must be pointed out that the *ACTRES Corpus Manager* interface is based on P-ACTRES 2.0. It allows the user to make assisted complex queries using regular expressions and obtaining the more common corpus statistics without having expert knowledge of the corpus query language syntax.

Índice

CAPÍTULO 7

CAPÍTULO 8

CAPÍTULO 9

Table of contents

CHAPTER 7

CHAPTER 8

CHAPTER 9

Índice de figuras

Índice de tablas

Capítulo

1

Introducción

La investigación basada en el análisis de corpus lingüísticos implica la utilización de aplicaciones software de gestión de corpus que permitan al investigador realizar consultas y extraer estadísticas del mismo. Para incorporar un corpus a un sistema de gestión, la ejecución de una serie de operaciones previas resulta estrictamente necesaria. Estas operaciones dependen del tipo de corpus que se desea construir (corpus monolingüe, corpus paralelo bi-/multilingüe o corpus comparable) y de sus características (capas de anotación lingüísticas, idioma, tamaño, etc.).

La realización de cada una de estas operaciones incluye la utilización o desarrollo *ad hoc* de recursos software. El uso de estos recursos software no siempre es sencillo si no se disponen de conocimientos técnicos suficientes. Por otro lado, el desarrollo de recursos software *ad hoc* implica el conocimiento de habilidades de programación. A esta problemática hay que añadir que el número de recursos software que permiten realizar una misma operación, además de los recursos *ad hoc*, es elevado, y cada uno de ellos puede mostrar los resultados con un formato diferente.

Tras realizar todas estas operaciones, es necesario adaptar el corpus al formato requerido por el software de gestión de corpus empleado. Teniendo en cuenta el

número de formatos posibles, la necesidad de crear programas específicos que permitan una transformación directa se hace indispensable.

A esta problemática hay que añadir que cada software de gestión de corpus lingüísticos posee una funcionalidades y limitaciones concretas, no todos ellos soportan corpus bi-/multilingües o anotaciones lingüísticas particulares como la semántica. Y, además, agregar la complejidad adicional de crear corpus en lenguas distintas del inglés, ya que la preponderancia de la lengua inglesa en el desarrollo del software incide negativamente en la disponibilidad de los mismos recursos para otras lenguas.

Existen frameworks integrales para el tratamiento de corpus lingüísticos que automatizan algunos de los procesos necesarios para incorporar corpus en su sistema de gestión. Pero su funcionalidad está acotada y ciertas características requieren la ejecución de programas externos, junto a una posterior conversión del resultado obtenido a un formato definido, tarea que en la mayoría de ocasiones debe realizar un especialista técnico. Por ejemplo, el alineado de corpus paralelos en Sketch Engine (Kilgarriff, et al., 2014) debe realizarse de forma externa y disponer de un formato concreto para su reconocimiento.

Ante esta situación hay dos aspectos que se hacen evidentes:

- Es necesario desarrollar un framework integral que permita al usuario lingüista construir y analizar sus corpus lingüísticos con múltiples capas de anotación sin necesidad de disponer de conocimientos de programación, disponer de habilidades técnicas para ejecutar programas externos o requerir la asistencia de un especialista técnico.
- El desarrollo de este software ha de centrarse en la usabilidad, posibilitando al usuario lingüista la construcción de distintos tipos de corpus (monolingües, bi-/multilingües paralelos y comparables) y la inclusión a su elección de los etiquetados lingüísticos más relevantes (gramatical, semántico y retórico).

Para implementar de manera efectiva este software es necesaria la realización de las siguientes tareas:

(1) **Recopilación de información**: bibliografía y documentación sobre el software para el tratamiento de corpus lingüísticos.

(2) **Estudio de la información recopilada**:

- Determinar los procesos y operaciones implicados en el tratamiento de corpus bi-/multilingües, haciendo especial hincapié en las alternativas software existentes y en su usabilidad y relevancia.
- Análisis exhaustivo de los frameworks integrales para el análisis de corpus lingüísticos existentes en la actualidad.

(3) **Análisis, diseño e implementación del framework**.

(4) **Testeo y validación del framework**.

Señalar que la extensión de esta tesis doctoral es sustancialmente superior a la de otras tesis doctorales presentadas en el programa de doctorado de "Sistemas Inteligentes en la Ingeniería" y otros programas similares relacionados con la ingeniería informática. Esto es una consecuencia directa del carácter multidisciplinar de la tesis doctoral, que combina los campos de la ingeniería informática y de la lingüística de corpus. Por este motivo, y para beneficio de los lectores de ambas disciplinas, se ha creído conveniente proporcionar información adicional sobre la terminología empleada, e incluir un número de diagramas mayor del habitual en ingeniería para aclarar el funcionamiento del software desarrollado con mayor precisión.

Por último, describir con brevedad la estructura del resto de la tesis doctoral. En el Capítulo 2 se explican algunos conceptos básicos, como consecuencia de la temática interdisciplinar de la tesis doctoral. Además, también se describen los procesos habituales en la creación y consulta de corpus lingüísticos. El Capítulo 3 muestra el estado de la cuestión relacionado con el software especializado en el tratamiento de corpus lingüísticos, haciendo especial énfasis en los frameworks integrales para el análisis de corpus. El Capítulo 4 define la hipótesis de trabajo

junto a las necesidades, objetivos y nicho a cubrir por la tesis doctoral. El Capítulo 5 relata la metodología seguida para la creación de *ACTRES Corpus Manager*. El Capítulo 6 se encarga de mostrar toda la información sobre el proceso de diseño, implementación y análisis de *ACTRES Corpus Manager*. El Capítulo 7 describe el proceso de validación de *ACTRES Corpus Manager* por medio de pruebas de funcionalidad y de un cuestionario de usabilidad. El Capítulo 8, discusión, sirve para proveer de explicaciones adicionales a algunas funcionalidades de *ACTRES Corpus Manager*. El Capítulo 9, conclusiones, determina qué se ha conseguido gracias a la creación de *ACTRES Corpus Manager* y las líneas de trabajo futuras. Para concluir, se ofrece un glosario, donde se definen los conceptos fundamentales utilizados a lo largo de la tesis doctoral, un resumen en inglés, las referencias y los anexos.

Capítulo

2

Conceptos fundamentales

La inclusión de un capítulo de conceptos fundamentales no es una práctica habitual en el desarrollo de una tesis doctoral tradicional. Sin embargo, como consecuencia del carácter multidisciplinar de la tesis presentada, que combina los campos de la ingeniería informática y la lingüística de corpus, se ha creído conveniente introducir este capítulo con el objetivo de facilitar la lectura y comprensión de cualquiera de los lectores adscritos a una u otra disciplina.

2.1 Corpus lingüístico

2.1.1 Definición de corpus lingüístico

La palabra corpus procede del término latín *corpus*, que significa "cuerpo" y que según recoge la Real Academia Española (Real Academia Española, 2017c) se define como:

> *"conjunto lo más extenso y ordenado posible de datos o textos científicos, literarios, etc., que pueden servir de base a una investigación".*

La definición en español no es del todo precisa considerando el punto de vista de la lingüística computacional. Mucho más acertada es la proporcionada por *Oxford Dictionary of English* en lengua inglesa (Soanes & Stevenson, 2005):

> *"a collection of written or spoken material in machine-readable form, assembled for the purpose of linguistic research."*

O por *Cambridge Dictionaries Online*, también en lengua inglesa (Cambridge University Press, 2017):

> *"a collection of written or spoken material stored on a computer and used to find out how language is used."*

Una definición similar es la proporcionada en muchas fuentes relacionadas con la disciplina científica de la lingüística computacional, y más concretamente en el área de actuación de la lingüística de corpus. Por citar dos de ellas:

- (McEnery & Wilson, 2001, pág. 197):
  "it refers to a finite collection of machine-readable texts sampled to be maximally representative of a language or a variety of it."

- O (McEnery, Xiao, & Tono, 2006, pág. 345):
  "a collection of sampled texts, written or spoken, in machine readable format form which maybe annotated with various forms of linguistic information"

De estas definiciones, la principal conclusión a extraer, y lo que diferencia un corpus de una simple colección de textos digitalizados, es que la construcción el corpus satisface un propósito lingüístico (Sinclair & Ball, 1996, pág. 27) o lo que Leech denomina función particular representativa del corpus lingüístico (Leech, 1991, pág. 11). El hecho de que el corpus esté en formato *machine-readable* o digitalizado, implica que puede ser procesado por un ordenador. De este modo se

pueden realizar las tareas básicas necesarias en cualquier estudio basado en corpus: búsqueda en contexto, recuperación, ordenación y extracción o cálculo de características de forma rápida y precisa. Por último, mencionar que la incorporación de anotaciones lingüísticas maximiza las posibilidades de los estudios que pueden llevarse a cabo sobre los corpus, así como su difusión y replicabilidad.

Proporcionada la definición de qué es un corpus lingüístico es importante acotar qué no es. De acuerdo con (Sinclair, 2005, págs. 21-22), un corpus lingüístico no es:

- Una lista de palabras.
- Una colección de textos aleatorios.
- Una colección de citas.
- Un texto.
- La web.[1]

2.1.2 Tipos de corpus

Existen distintos tipos de corpus según sus características: modo, número de lenguas, límite de palabras, especificidad de su contenido o periodo temporal. A continuación se describirán los distintos tipos de corpus utilizando como fuente la clasificación proporcionada en (Villayandre-Llamazares, 2008) a la que se han añadido diversas modificaciones:

De acuerdo con el modo de la lengua existen:

- **Corpus orales**: Textos procedentes de muestras en lengua hablada que son utilizados en estudios basados en comunicaciones orales. Estos tipos de corpus pueden dividirse en dos: (1) aquellos que tiene transcripciones audio-

[1] Existen detractores (McEnery & Hardie, 2012, págs. 7-8) y defensores (Kilgariff, 2001) de este postulado.

texto y (2), aquellos cuyos elementos son audios.[2] Un ejemplo del primero es *Lancaster/IBM Spoken English Corpus* (SEC) (Knowles, Wichmann, & Alderson, 1996). En cuanto al segundo mencionar *Santa Barbara Corpus of Spoken American English* (SBCSAE) (Du Bois, Chafe, Meyer, Thompson, & Martley, 2016).

- **Corpus escritos**: Textos procedentes de muestras en lengua escrita.[3] Utilizados en estudios en comunicaciones escritas. Por ejemplo: *British National Corpus* (BNC Consortium, 2007).

Según el número de lenguas:

- **Monolingües**: Textos en una sola lengua. Permiten llevar a cabo estudios lingüísticos en un idioma concreto. Por ejemplo: Corpus de Referencia del Español Actual (CREA) (Real Academia Española, 2017b)
- **Bi-/Multilingües**: Textos en dos o varias lenguas. De acuerdo con (McEnery & Hardie, 2012, pág. 19), (McEnery, Xiao, & Tono, 2006, pág. 47), (McEnery & Xiao, 2007a, págs. 132-135), (McEnery & Xiao, 2007b, págs. 19-22), existen tres tipos de corpus que implican dos o más idiomas:
    - Tipo A: Un archivo original y diversas traducciones. Por ejemplo: P-ACTRES (Izquierdo, Hofland, & Reigem, 2008)
    - Tipo B: Pares de corpus creados empleando el mismo muestreo. Por ejemplo *Lancaster Corpus of Mandarin Chinese* (McEnery, Xiao, & Mo, 2003)
    - Tipo C: Una combinación de los anteriores. Por ejemplo: *The English-Norwegian Parallel Corpus* (ENPC) (Johansson & Hofland, 1994).

Estos tres tipos de corpus tienen distintas denominaciones, dependiendo de los autores y de acuerdo con (McEnery & Hardie, 2012, p. 19):

[2] Estos corpus quedan fuera del rango de actuación de esta tesis doctoral.
[3] Es el tipo de corpus utilizado en el desarrollo de esta tesis doctoral.

(1) para (Aijmer, Altenberg, & Johansson, 1996, págs. 11-13) y (Granger, 1996, pág. 38) el tipo A es un corpus de traducción, mientras el B es un corpus paralelo;

(2) para (Baker, 1993, pág. 248; Baker, 1995, pág. 231; Baker, 1999), (McEnery & Wilson, 2001, pág. 70) y (Hunston, 2002, pág. 38), el tipo A es un corpus paralelo y el tipo B un corpus comparable;

(3) para (Johansson & Hofland, 1994) y (Johansson, 1998, págs. 4-5) ambos son corpus paralelos;

(4) por último para Barlow (Barlow, 2003) sólo el A es un corpus paralelo.

Esta tesis utilizará el mismo criterio que en (McEnery & Hardie, 2012, pág. 19), (Baker, 1993, pág. 248; Baker, 1995, pág. 231; Baker, 1999), (McEnery & Wilson, 2001, pág. 70) y (Hunston, 2002, pág. 38), el corpus de tipo A será paralelo, y los de tipo B y C comparables. De esta forma se es consistente con el criterio del contenido del corpus: si éste posee textos originales y traducciones es paralelo, si son textos en distinto o mismo idioma con un muestreo similar, es comparable.[4]

Por lo tanto, se considera corpus paralelo aquel que contiene un texto original y sus traducciones en distintas lenguas, y un corpus comparable aquel que posee al menos dos variedades diferentes de la misma lengua, o está formado por textos que cumplen la misma función comunicativa (por ejemplo, actas de reuniones) en distintas lenguas.[5]

Dependiendo del límite de palabras:

- **Abierto o *monitor corpus***: Son corpus que se amplían con nuevos textos con el paso del tiempo. Utilizados para estudiar la evolución de una lengua

[4] Para más información sobre los corpus comparables consultar (Sharoff, Rapp, Zweigenbaum, & Fung, 2013)

[5] A partir de este punto, corpus comparables será utilizado para designar a los corpus comparables en una o varias lenguas indistintamente.

a lo largo de un intervalo temporal. Por ejemplo: *Corpus of Contemporary American English* (COCA) (Davies, 2008).

- **Cerrado**: Corpus a los que no se añaden nuevos textos. Permiten estudiar fenómenos concretos en un conjunto determinado de textos delimitados temporalmente. Por ejemplo: *British National Corpus* (BNC Consortium, 2007).

De acuerdo con la especificidad de su contenido:

- **Corpus de referencia**: Incluye textos que describen una lengua de la forma más completa posible, incorporando la mayor parte de estilos de forma equilibrada. Permiten estudiar los rasgos lingüísticos globales de un idioma. Por ejemplo, Corpus de Referencia del Español Actual (CREA) (Real Academia Española, 2017b).
- **Corpus especializado**: Se centra en una modalidad comunicativa concreta. Posibilita el estudio de rasgos lingüísticos de una modalidad comunicativa específica representada en un corpus. Por ejemplo: ZIO Corpus (UPV/EHU, 2013).

Según el periodo temporal que refleje:

- **Diacrónico o histórico**: Incluyen textos de diferentes etapas temporales sucesivas. Permite estudiar evoluciones de la lengua en un período largo de tiempo (Villayandre-Llamazares, 2008, pág. 345). Por ejemplo: CORPES XXI (Real Academia Española, 2017a).
- **Sincrónico**: Incluye textos pertenecientes a una sola etapa temporal. Posibilita el estudio de una o más variedades lingüísticas en un momento determinado del tiempo (año, período, etc.), pero sin prestar atención a su evolución (Villayandre-Llamazares, 2008, pág. 346).

2.2 Corpus y computación

La necesidad de disponer de software para llevar a cabo el tratamiento de corpus lingüísticos es consecuencia directa del volumen del material textual total de los corpus. Sin el apoyo de los ordenadores estas tareas consumirían mucho tiempo o no serían realizables (McEnery & Hardie, 2012, págs. 1-2). En el tratamiento de corpus lingüísticos intervienen diseño de algoritmos de computación, métodos de almacenamiento, codificación de caracteres, diseño de interfaces, además de los conocimientos de programación necesarios que afectan al posterior análisis de datos (Anthony, 2013a, pág. 144).

2.2.1 Lingüística de corpus, lingüística computacional y NLP

La lingüística de corpus, la lingüística computacional y el procesamiento del lenguaje natural o NLP[6] por sus siglas en inglés (*Natural Language Processing*), son tres conceptos fundamentales que han de definirse correctamente con el fin de delimitar las áreas de actuación de esta tesis.

- La lingüística computacional es una disciplina científica de carácter multidisciplinar (lingüística y ciencias de la computación) que desarrolla métodos computacionales que permiten comprobar hipótesis de investigación relacionadas con la lengua escrita u oral.
- La lingüística de corpus es un término cuya definición precisa es muy discutida entre la comunidad científica[7]: ¿es una herramienta, un método, una metodología, una disciplina, una teoría, una estrategia teórica, un paradigma práctico o metodológico o una combinación de estos? (Taylor, 2008, pág. 180). Aquí se definirá como un área de actuación de la lingüística computacional que se encarga del estudio de una lengua por medio de casos reales recopilados en un corpus lingüístico.

[6] De aquí en adelante se utilizará el término NLP en lugar de *Natural Language Processing*.
[7] Consultar (Taylor, 2008) para más información.

- Por último, el NLP es una disciplina técnica que emplea los ordenadores para manipular la lengua, ya sea oral o escrita, y crear distintos programas informáticos que son útiles para resolver diversos problemas (traductores o reconocimiento de voz entre otros).

Estos tres campos tienen en común que todos ellos emplean los corpus lingüísticos como fuente de datos para realizar sus análisis. Esta fuente de datos es exclusiva en el caso de la lingüística de corpus o de la lingüística computacional basada en corpus, y una de las principales fuentes para el NLP.

Esta tesis tiene como objetivo la creación de un software que sirva para comprobar una hipótesis de investigación relacionada con la lingüística, esto es, pertenece a la disciplina científica de la lingüística computacional. Para resolver algunos de los problemas emplea métodos de NLP, por ejemplo, el etiquetado gramatical. Por último, la utilidad del software es llevar a cabo investigaciones basadas en la lingüística de corpus.

La Figura 1 muestra la relación del software creado de acuerdo con los conceptos definidos.

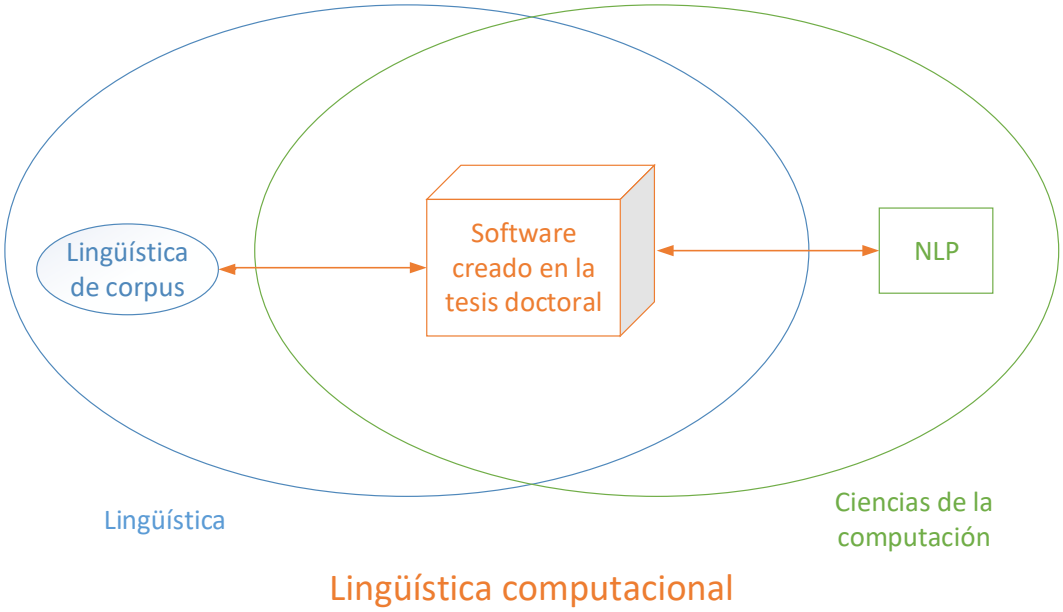


Figura 1 - Relación simplificada entre la lingüística de corpus, la lingüística computacional, el NLP y el software creado en la tesis doctoral

2.2.2 Corpus lingüístico y NLP

En la actualidad, los corpus lingüísticos y el NLP mantienen una relación de dependencia mutua. Por un lado, la necesidad de disponer de herramientas y/o aplicaciones realizadas por medio de técnicas de NLP por parte de los corpus lingüísticos y, por el otro, la dependencia de corpus lingüísticos por parte del NLP a causa del empleo de modelos probabilísticos, que requieren de grandes conjuntos de datos anotados para su entrenamiento.

2.2.2.1 Desarrollo del NLP

Las tecnologías basadas en NLP juegan un rol fundamental en la sociedad de la información globalizada actual (Bird, Klein, & Loper, 2009, pág. ix). Aparecen con frecuencia en las situaciones cotidianas que desempeña cualquier ser humano: interacción hombre máquina a través de voz, traductores automáticos o asistentes virtuales por citar algunos de ellos. Esta tendencia se mantiene en campos como la lingüística computacional donde constantemente se hace uso de etiquetadores

lingüísticos (gramaticales, semánticos), alineadores o *sentiment analysers*, entre otros ejemplos.

El NLP abarca desde el simple conteo de frecuencias de palabras, al complejo análisis de estilos de escritura (Bird, Klein, & Loper, 2009, pág. ix). Es decir, se puede decir que el NLP transforma una entrada (texto o discurso oral), en una salida concreta (por ejemplo el resultado de una consulta, traducción, respuesta o resumen) (Chanod, Hobbs, Hovy, Jelinek, & Rajman, 1999). Por "lenguaje natural" se entiende aquel lenguaje empelado por los seres humanos en sus comunicaciones diarias (Bird, Klein, & Loper, 2009, pág. ix).

El desarrollo de NLP tiene su origen en la década de los 50. Surge como una intersección de las ciencias de la computación, más concretamente de la inteligencia artificial, y de la lingüística. Inicialmente, la estrategia empleada para obtener datos que sirvieran de base para el desarrollo las aplicaciones de NLP era a través de análisis simbólicos. De acuerdo con (Chanod, Hobbs, Hovy, Jelinek, & Rajman, 1999) los análisis simbólicos se caracterizan por:

- Necesitar un análisis manual que a la postre origina una teoría.
- Implicar la creación de reglas manuales para el tratamiento del lenguaje.
- Basarse en juicios intuitivos.

El análisis simbólico es de una alta calidad, ya que se trata de un estudio meticuloso de un fenómeno concreto, y además no requiere una gran fuente de información. Pero es lento, costoso, difícil y a menudo está incompleto (Chanod, Hobbs, Hovy, Jelinek, & Rajman, 1999).

La realidad es que la lengua natural se caracteriza por ser una fuente inagotable de datos no estructurados y a menudo ambiguos. La necesidad de disponer de anotaciones gramaticales, semánticas, retóricas o de otro tipo para subsanarlo en grandes fuentes de datos provoca que los análisis simbólicos sean difícilmente realizables (Nadkarni, Ohno-Machado, & Chapman, 2011, págs. 544-545)**.** Por esta

razón, en la década de los 80 hubo una reorientación hacia la ejecución de análisis estadísticos que según (Nadkarni, Ohno-Machado, & Chapman, 2011, pág. 545) consistió en:

- El análisis en profundidad fue sustituido por aproximaciones más simples y robustas.
- La utilización de métodos probabilísticos derivados del *Machine Learning* comenzó a ser recurrente.
- Grandes conjuntos de textos anotados, los corpus lingüísticos, comenzaron a ser utilizados con el fin de entrenar a los algoritmos de *Machine Learning*.

Métodos como la creación de gramáticas libres de contexto probabilísticas (reglas con probabilidades asociadas), o la implementación de árboles de decisión empleando algoritmos como el C4.5 (Quinlan, 2014) o *Hidden Markov Models* (Baum & Petrie, 1966) comenzaron a ser habituales.

Los modelos probabilísticos son entrenados a partir de datos que son considerados correctos. Estos datos se denominan con el término inglés *gold standard*. Un *gold standard* distinto del utilizado para el entrenamiento puede ser empleado para llevar a cabo una evaluación automática del modelo probabilístico. De acuerdo con (Nadkarni, Ohno-Machado, & Chapman, 2011, pág. 545) estas dos circunstancias provocan:

- La obtención de excelentes resultados por el hecho de emplear datos exactos para el entrenamiento.
- La consecución de una evaluación automatizada empleando parte del *gold standard* que no haya sido empleado en el entrenamiento.

Según (Chanod, Hobbs, Hovy, Jelinek, & Rajman, 1999), el análisis estadístico en contraste con el análisis simbólico se caracteriza por:

- Requerir un análisis automático que origina el modelo estadístico utilizado.

- Implicar la creación de reglas automáticas de acuerdo con el modelo estadístico.
- Tener la posibilidad de evaluar el algoritmo de forma automatizada.

En resumen, los análisis estadísticos resultan más rápidos, robustos y eficientes que cualquier análisis simbólico, capaces de identificar fenómenos no apreciables a simple vista por parte de un ser humano (Chanod, Hobbs, Hovy, Jelinek, & Rajman, 1999), y por ello son la base del NLP actual.

2.2.2.2 Los corpus lingüísticos como fuente de datos del NLP

Se hace patente que los análisis estadísticos del NLP requieren de grandes fuentes de información lingüísticas anotadas con exactitud para entrenar los modelos estadísticos creados a partir de distintos algoritmos. Algunos datos empleados en el entrenamiento de modelos estadísticos basados en aplicaciones de NLP de acuerdo con (Calzolari, Choukri, Fellbaum, Hovy, & Ide, 1999) son:

- Etiquetados gramaticales.
- Etiquetados morfológicos.
- Etiquetados sintácticos.
- Colocación.
- Frecuencias de aparición.
- Etiquetados semánticos.
- Análisis del discurso.
- Comunicación interpersonal.[8]

Estos datos aparecen en documentos lingüísticos como gramáticas, diccionarios, ontologías y por encima de todos ellos, en los corpus lingüísticos. Como se ha mencionado, los corpus son una de las principales fuentes de datos sobre las que se basan los análisis (estadísticos o simbólicos) que se llevan a cabo en las aplicaciones

[8] Ver (Baker, 2010, Capítulo 5) para más información.

prácticas de NLP (Bird, Klein, & Loper, 2009, pág. 39) o (Calzolari, Choukri, Fellbaum, Hovy, & Ide, 1999), resultando especialmente útiles los corpus bi-/multilingües paralelos anotados con distintas capas, principalmente los corpus con anotación semántica, gramatical o sintáctica.

A la hora de crear un corpus para emplearlo en NLP, hay que tener en cuenta para qué tarea concreta va a ser empleado y las herramientas de análisis disponibles (Calzolari, Choukri, Fellbaum, Hovy, & Ide, 1999), así como la disponibilidad de etiquetadores lingüísticos para llevar cabo las anotaciones pertinentes.

2.2.2.3 NLP como recurso técnico del procesamiento de corpus lingüísticos

En el otro lado de la relación entre el NLP y el corpus lingüístico, la necesidad de disponer de herramientas y/o aplicaciones realizadas por medio de técnicas de NLP por parte de los corpus lingüísticos aparece desde sus inicios como consecuencia de tener que digitalizar y tratar los textos que conforman el corpus. Pero esta dependencia no queda ahí, la inmensa mayoría de las aplicaciones relacionadas con el procesamiento de corpus lingüísticos requiere de procesos basados en NLP, por ejemplo: etiquetado lingüístico o tratamiento de cadenas de texto o *strings*.

Resulta incuestionable que la investigación en NLP ha contribuido enormemente en el desarrollo de tecnología relacionada con los corpus lingüísticos (Xiao, 2010, pág. 147), pero también lo ha perjudicado debido a la tendencia a descomponer el análisis del lenguaje en varias etapas (Dale, 2010, pág. 4) (Nadkarni, Ohno-Machado, & Chapman, 2011, pág. 548), lo que origina una atomización de recursos que repercutirán en la usabilidad del software creado para el tratamiento de corpus lingüísticos, como se explicará en la sección 3.1 Problemas y limitaciones.

2.2.3 El corpus lingüístico como metodología científica

Las características que determinan la lingüística de corpus que son generalmente aceptadas por la comunidad científica/investigadora son: (1) es empírica, en la medida que los textos que conforman los corpus sobre los que se basan los experimentos son reales, (2) los corpus se almacenan en formato electrónico y son un ejemplo representativo del idioma o lengua a estudiar, (3) emplea distinto software para analizar los patrones lingüísticos que forman parte del análisis, y (4) depende de los análisis cuantitativos, basados en las frecuencias, y cualitativos, basados en la observación detallada de un fenómeno lingüístico concreto (Biber, Conrad, & Reppen, 1998, pág. 4).

Gracias a la disponibilidad de software especializado para facilitar este tipo de análisis, por ejemplo: Treetagger (Schmid, 1995) para etiquetados gramaticales, AntConc (Anthony, 2014a) para ejecutar consultas y GIZA++ (Och & Ney, 2003) para llevar a cabo alineaciones; la lingüística de corpus se ha convertido en una de las metodologías más ampliamente utilizadas en todo tipo de investigaciones lingüísticas (lingüística descriptiva, contraste, tipología, variación entre otras), ya que permite verificar o refutar de modo empírico las hipótesis y/o teorías, y de este modo hacen avanzar la investigación y el conocimiento del funcionamiento de las lenguas naturales.

Sin embargo, existen lingüistas que no consideran admisible utilizar el corpus lingüístico como método de investigación, entre ellos Noam Chomsky. Su opinión es que no es válido anteponer el análisis empírico de la información a un planteamiento meramente racionalista (Andor, 2004, pág. 97). Para Chomsky (McEnery & Wilson, 2001, págs. 5-12) (Chomsky, 1957, págs. 15-17) y en especial (Chomsky, 1962, págs. 914, 915, 921, 922, 924, 989) entre otras referencias resulta algo totalmente inaceptable.

De forma muy acertada, (McEnery & Hardie, 2012, págs. 25-27) explican el punto de vista de Chomsky. De acuerdo con su parecer, lo que Chomsky defiende es que

obtener resultados concretos a partir de un conjunto de datos recolectados para tal fin, en este caso un corpus, para afirmar o refutar una hipótesis resulta inadmisible, porque según Chomsky el corpus no representa la totalidad de la lengua, ya que es imposible, y además los textos recogidos pueden ser manipulados en alguna medida consciente o inconscientemente.

Este rechazo es, no obstante, muy minoritario y las contraargumentaciones sostienen que, por ejemplo, las ciencias naturales no sólo se basan en experimentos sino en la recogidas de ingentes cantidades de datos que se analizan, o que los mismos partidarios de Chomsky construían, de una manera u otra, pequeños corpus para apoyar o refutar sus ideas de forma inconsciente (McEnery & Hardie, 2012, pág. 25). Por ello, el corpus no ha de ser el único criterio de validación de datos. Otros métodos adicionales como los informantes o cuestionarios también han de ser considerados (McEnery & Hardie, 2012, pág. 26).

La comunidad científica, por abrumadora mayoría, avala el uso de corpus en la investigación lingüística. En palabras de (Leech, 1992) los corpus son la más potente metodología desde el punto de vista científico puesto que permite una verificación objetiva de los resultados de los estudios y no meramente subjetiva, de acuerdo con (McEnery & Wilson, 2001, págs. 7-12) las observaciones basadas en corpus son más verificables que los juicios retrospectivos. (Wasow, 2002, pág. 163) por su parte sostiene que no hay ninguna buena excusa para no probar el trabajo teórico sobre un corpus ya que en la actualidad existen multitud de fuentes de datos, así como sofisticadas herramientas de búsqueda.

Una prueba de la generalización del uso de corpus es la proliferación de publicaciones científicas especializadas, como *International Journal of Corpus Linguistics* o *Corpora*, y su reconocido prestigio, o los trabajos que utilizan esta

metodología en revistas punteras en sus respectivos campos, por ejemplo las revistas *Languages in Contrast* y *English for Specific Purposes*.[9]

Por último, aunque las investigaciones basadas en corpus posibilitan el estudio de las lenguas gracias a los usos reales representados en un corpus dado (McEnery & Wilson, 2001, pág. 1), los corpus han de cumplir una serie de requisitos para ser considerados válidos para corroborar o refutar hipótesis de investigación. Las directrices más populares son las establecidas por (Biber, 1993) y (Sinclair, 2005):

- **Equilibrio, representatividad y muestreo** (Biber, 1993, pág. 243): El muestreo se refiere a los textos individuales que conforman el corpus lingüístico: su procedencia, estilo, género por citar algunos rasgos de los mismos. El equilibrio es un concepto subjetivo que se refiere a las proporciones de las distintas clases de textos y que se corresponden con valoraciones basadas en la hipótesis de investigación que se quiere resolver o en intuiciones (Sinclair, 2005, pág. 14). El concepto de representatividad es aún más difuso. Es discutido por numerosos autores tal y como relata (Xiao, 2007, págs. 3-8). Depende en gran medida de si el corpus es de referencia o especializado, donde la saturación de un determinada característica lingüística puede causar anomalías (Xiao, 2007, pág. 7). En consecuencia, tanto equilibrio como representatividad son conceptos vagamente definidos que dependen del tipo de corpus que se quiere crear y de la hipótesis o investigación que el corpus lingüístico ayude a corroborar.
- **Tamaño**: Esta característica se refiere a las dimensiones de los textos que forman el corpus y a las dimensiones del corpus en su totalidad.
    - En cuanto al primer concepto, dimensiones de los textos que forman un corpus, de acuerdo con (Sinclair, 2005, pág. 7) los textos han de estar completos siempre que sea posible. Pero esto puede ser

[9] Los términos corpus o *corpora* aparecían en el 15% de los *highlights* de artículos aparecidos en la revista '*English for Specific Purposes*' durante 2015 y 2016.

problemático principalmente por tres motivos (Xiao, 2007, pág. 25): (1) por problemas de derechos de autor, (2) la saturación de un estilo o tema y (3) que las características lingüísticas y sus distribuciones son estables a lo largo de un texto completo, por lo que no es estrictamente necesario incluirlo en su totalidad.

  - El tamaño del corpus depende la frecuencia y distribución de la característica lingüística que se quiere estudiar en el corpus a construir (McEnery & Wilson, 2001, pág. 80). En la actualidad, las limitaciones de procesamiento son menos restrictivas, por lo que el tamaño de corpus no está tan limitado por la tecnología disponible. Además, hay otros factores que afectan a esta característica: disponibilidad de textos que satisfagan el tipo de corpus que se quiere construir o cuestiones de derechos de autor por citar algunos de ellos. Existe un debate en torno a si el tamaño realmente es un factor determinante o no, ver (Xiao, 2010, págs. 148-149) para más información.

- **Tipo de corpus**: de acuerdo con la naturaleza de la investigación que se quiera estudiar, existen diferentes tipos de corpus (detallados en el apartado 2.1.2 Tipos de corpus). Por ejemplo, para buena parte de la comunidad investigadora, para llevar a cabo estudios lingüísticos contrastivos entre dos o más lenguas lo más adecuado es utilizar un corpus comparable.

2.3 Software especializado para el tratamiento de corpus lingüísticos

2.3.1 Definición de software y conceptos relacionados

Es importante definir claramente los conceptos software, programa, aplicación, herramienta, librería, *toolkit*[10] y *framework*[11]. Sin embargo, no existe un consenso por parte de la comunidad científica para ello, ya que la definición varía de acuerdo con la disciplina y/o tecnología utilizada.

Con el fin de poder acotar y establecer categorías dentro del software a analizar se propondrán una serie de definiciones libres que sirvan para facilitar la comprensión del análisis realizado tanto por parte de los especialistas lingüísticos como de los especialistas informáticos. No se pretende por tanto establecer ninguna nueva terminología, tan sólo dejar claro el uso que aquí se hace de ella.

Los conceptos de software, programa, aplicación, herramienta, librería, toolkit y framework se pueden definir como:

- Software es cualquier programa informático utilizado por un ordenador. Software es empleado en muchas ocasiones como sinónimo de programa. Por ejemplo, tanto WordSmith Tools (Scott, 2012) como Bluealign (Sennrich & Volk, 2010) son considerados software.
- Una aplicación es un programa o conjunto de programas informáticos diseñados para ser utilizados por un usuario y que por tanto poseen un interfaz más o menos compleja. Permite realizar diferentes tareas de forma

[10] Ante su frecuente uso y para no dificultar la lectura, se utilizará el término sin ninguna diferenciación tipográfica.

[11] Ante su frecuente uso y para no dificultar la lectura, se utilizará el término sin ninguna diferenciación tipográfica.

coordinada. WordSmith Tools es un ejemplo de aplicación, por tanto, diseñado para su uso por parte de un usuario.

- Una herramienta informática es un programa que facilita la realización de una tarea específica, normalmente de bajo nivel, y que no tiene por qué diseñarse para el uso por parte de un usuario. Por ejemplo, el anteriormente mencionado Bluealign es un ejemplo de herramienta, no posee interfaz de usuario y puede ejecutarse vía comando por medio de otro programa informático.
- Una librería es un conjunto de funciones y recursos de programación empleados para desarrollar software relacionado con un tipo de implementación concreta: por ejemplo, la web, los corpus lingüísticos o el procesamiento del lenguaje natural. Un ejemplo de librería web es JQuery (The JQuery Foundation, 2017).
- Un toolkit o *suite* o similar, es un conjunto de librerías o programas que proporcionan al programador un gran número de recursos para resolver problemas o crear software en una disciplina concreta: por ejemplo, la web, los corpus lingüísticos o el procesamiento del lenguaje natural. NLTK (Bird, Klein, & Loper, 2009) es un toolkit para el procesamiento de lenguaje natural muy estrechamente ligado con la lingüística computacional.
- Un framework es un conjunto de librerías o programas que se utilizan dentro de un entorno para resolver problemas o crear software en una disciplina concreta: por ejemplo, la web, los corpus lingüísticos o el procesamiento del lenguaje natural. Este conjunto de librerías o programas pueden estar relacionados. El término framework es muy difuso y abierto, y se suele utilizar de forma muy libre. Es muy similar a la definición proporcionada en toolkit, *suite*, la diferencia reside en que el framework suele proporcionar el entorno de trabajo completo en el que se realiza la programación.

De acuerdo con estas definiciones podemos considerar que un software especializado para el tratamiento de corpus lingüísticos es:

> *Aquel programa de ordenador que ha sido creado exclusivamente para llevar a cabo procedimientos técnicos relacionados con los corpus lingüísticos.*

El tratamiento de corpus lingüísticos incluye la ejecución de diverso software como, por ejemplo:

- Etiquetadores lingüísticos.
- Alineadores.
- Programa de concordancias o concordancers.
- Programas de estadísticas (lista de palabras clave, n-gramas o colocaciones por citar algunas de ellas).
- Compiladores automáticos de corpus.
- Scripts para el tratamiento de entrada/salida.

Para llevar a cabo el tratamiento de un corpus lingüístico es necesaria la interconexión de entradas y salidas del distinto software, una tarea enormemente compleja si no se poseen conocimientos de programación, más concretamente de tratamiento de textos y flujos de datos. Esto provoca que exista una dependencia de asistencia técnica por parte de los investigadores lingüistas que intentan realizar estudios sobre corpus. La obligación de disponer de un software que englobe todas estas funcionalidades se hace por tanto patente.

Por estos motivos, se considera necesario modificar el término framework en tanto se refiera a entornos de trabajo interdisciplinares, en el que la informática se utiliza como herramienta y un área científica como base teórica. Un framework interdisciplinar se definirá como:

> *Una aplicación informática que integra distintas funcionalidades que permiten, sin salirse del mismo, llevar a cabo como mínimo, las tareas comunes de un área de actuación científica concreta.*

Tal y como se puede apreciar la definición es análoga a la proporcionada anteriormente, considerando que las librerías o programas del framework meramente informático son sustituidas por herramientas o aplicaciones plenamente funcionales para un área de actuación científica concreta.

Como consecuencia, el término framework para el tratamiento de corpus lingüísticos será empleado para designar:

> *Una aplicación informática que integra las distintas funcionalidades que permite, sin salirse del* framework*, llevar a cabo al menos las tareas comunes del tratamiento de corpus lingüísticos por parte de un usuario sin conocimientos de programación.*

Se considerarán tareas comunes mínimas e indispensables la compilación de corpus lingüísticos y la ejecución de búsquedas o concordancias sobre los mismos. La compilación de los textos se refiere a la capacidad de definir el corpus sobre el que realizar el estudio. Las búsquedas en el corpus definen la capacidad de ejecutar concordancias y mostrarlas por pantalla.

2.3.2 Aplicaciones y herramientas relacionadas con el tratamiento de corpus lingüísticos

En la Figura 2 se presenta un flujograma que muestra los procesos habituales que forman parte de un framework para el tratamiento de corpus lingüísticos completo junto con las tecnologías que lo sustentan. No sólo se incluyen las tareas básicas de compilación de corpus y ejecución de búsquedas, sino que se añade el soporte necesario para tratar distintos tipos de corpus y el cálculo de estadísticas.

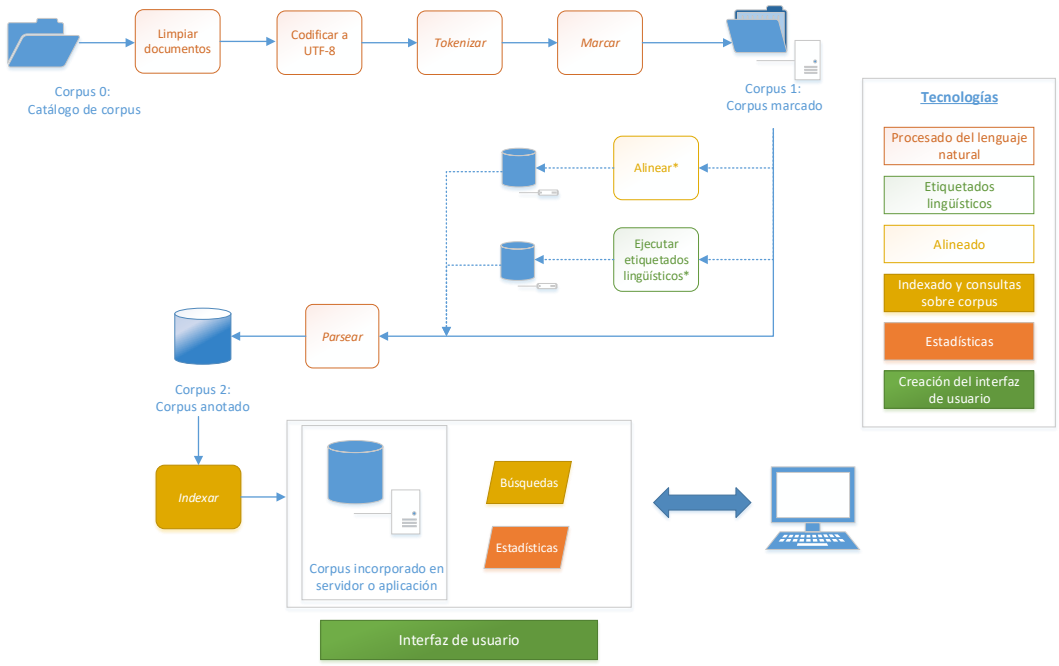


Figura 2 - Flujograma de un framework para el tratamiento de corpus lingüísticos junto a las tecnologías que lo sustentan

Aunque la mayor parte de los procesos que aparecen en la Figura 2 forman parte intrínseca de los frameworks que se estudiarán en el Capítulo 3: "Estado de la cuestión: Software especializado para el tratamiento de corpus lingüísticos", resultaría inabarcable realizar un análisis pormenorizado de todas y cada una de las aplicaciones/herramientas disponibles capaces de realizar cada uno de ellos. Este es el motivo por el que el estado de la cuestión se centrará en el análisis de frameworks disponibles y no en el análisis individual de sus componentes.

No obstante, se ha considerado conveniente proporcionar una breve descripción de las tecnologías que sustentan los procesos relacionados con el tratamiento de corpus lingüísticos en este apartado. Las tecnologías que participan en los distintos procesos y que se muestran en la Figura 2 son: (1) procesamiento del lenguaje natural (NLP), (2) etiquetados lingüísticos, (3) alineación de corpus paralelos, (4) indexación y consultas sobre corpus, (5) obtención de estadísticas cuantitativas y (6) desarrollo del interfaz de usuario.

(1) Procesamiento del lenguaje natural (NLP)

Si bien es cierto que el NLP es aplicado en la práctica totalidad de los procesos que se llevan a cabo en el tratamiento de corpus lingüísticos, existen operaciones que pueden denominarse básicas, cuyo fin es cambiar el formato de los archivos del corpus de acuerdo con los requisitos de un proceso.

Estas tareas resultan relativamente sencillas y suelen ser realizadas a través de programas *ad hoc*. El reto técnico reside en la necesidad de aplicar el proceso a cientos de documentos.

De acuerdo con la Figura 2, los procesos intrínsecamente relacionados con el procesamiento del lenguaje natural son: Limpiar documentos, Codificar a UTF-8,

Tokenizar[12], Marcar y *Parsear*[13].

- **Limpiar documentos**: Proceso de eliminación de caracteres no permitidos o inservibles, como etiquetas HTML, hipervínculos, imágenes, etc.
- **Codificar en UTF-8**: Las codificaciones permiten la representación de los caracteres en los equipos informáticos. En el pasado existían distintas codificaciones que convivían juntas provocando problemas de compatibilidad en las aplicaciones informáticas. Unicode (The Unicode Consortium, 2006), más concretamente la versión UTF-8, posibilita el procesamiento de cualquier carácter perteneciente a cualquier lengua del mundo por parte de las aplicaciones informáticas.

[12] Ante su frecuente uso y para no dificultar la lectura, se utilizará el término sin ninguna diferenciación tipográfica.

[13] Ante su frecuente uso y para no dificultar la lectura, se utilizará el término sin ninguna diferenciación tipográfica.

- **Tokenizar**: Consiste en dividir una fuente dada en *tokens* o símbolos. Estos *tokens* pueden ser palabras, frases, párrafos o cualquier otra unidad definida por el programador.
- **Marcar**: El marcado de un corpus lingüístico consiste en la inclusión de los metadatos junto a las partes en que se divide el texto (palabras, oraciones, capítulos si los hubiera, etc.) en un formato estandarizado, por ejemplo XML (*eXtensible Markup Language*) (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 1998).
- **Parsear**: El término parsear tiene múltiples acepciones dependiendo del ámbito donde se utilice. Existen parsers de lenguaje natural, parsers como parte de un compilador, o parsers como conjuntos de instrucciones que realizan una acción concreta en un lenguaje programación. En esta tesis doctoral se utilizará esta última acepción.

(2) Etiquetado lingüístico

Los etiquetados lingüísticos representan otra línea de investigación relacionada con el tratamiento de corpus lingüísticos y estrechamente ligada al subcampo del *Machine Learning*. Existen distintos tipos de etiquetado lingüístico: gramatical, semántico, retórico, sintáctico, fonético, pragmático, por citar algunos de ellos.

La disponibilidad del software para llevar a cabo los distintos tipos de etiquetado depende de tres factores principales:

i. **Idioma objetivo**: tal y como se ha manifestado con anterioridad, el desarrollo tradicional del software para el tratamiento de corpus lingüísticos gira principalmente en torno a la lengua inglesa. La práctica totalidad de los etiquetadores multilingües tienen a la versión en inglés como eje principal de desarrollo. Es por ello que la disponibilidad de etiquetadores para tratar textos en inglés es muy alta. En cuanto al resto de idiomas, depende del

número de hablantes y/o financiación de proyectos para trabajar con la lengua en cuestión (Hammarström, 2009) (Carbonell, 2010).

ii. **Tipo de etiquetado**: el framework objeto de esta tesis doctoral soporta etiquetado gramatical, semántico y retórico, algunos de los tipos de etiquetado más frecuentes y con un mayor número de usuarios lingüistas interesados.

- El etiquetado más frecuente y que está presente en gran parte de los corpus lingüísticos es el gramatical, también conocido como P*art-Of-Speech* (POS)[14] por sus siglas en inglés. Por esta razón, el número de software disponible para realizar esta tarea es muy alto, por ejemplo: Treetagger (Schmid, 1995), FreeLing POS tagger (Padró & Stanilovsky, 2012) o Stanford Log-linear Part-Of-Speech Tagger (Toutanova, Klein, Manning, & Singer, 2003). No obstante, existe gran cantidad de recursos para múltiples idiomas, no sólo para el inglés.
- El etiquetado semántico presenta dos estrategias principales: (1) basadas en ontologías, que emplean con frecuencia como fuente la base de datos léxica WordNet (Princeton University, 2010), y (2) fundamentadas en bases de datos léxicas propias como USAS (Piao, Bianchi, Dayrell, D'Egidio, & Rayson, 2015).
- En cuanto al etiquetado retórico, no existe un software comercial disponible capaz de realizar un etiquetado retórico como el que se plantea.[15]

iii. Sea cual sea el tipo de etiquetado la eficacia del mismo varía mucho en función del idioma empleado. Por ejemplo, es posible que un etiquetador tenga excelentes resultados en inglés y regulares en español. La selección

[14] De aquí en adelante se utilizarán los términos POS y gramatical indistintamente en lugar de usar el término Part-Of-Speech.

[15] El etiquetado retórico que se propone en esta tesis doctoral se detallará en el punto IX) Etiquetador retórico dentro del Capítulo 6: "Análisis, diseño e implementación ".

de uno o varios etiquetadores, dependiendo del idioma, acarrea problemas adicionales como consecuencia del uso de distintos conjuntos de etiquetas (*tagset* en inglés). Incluso empleando el mismo programa, existen diferentes *tagsets* para cada idioma, por ejemplo, en Treetagger. La ausencia de un estándar universal, tal y como se expondrá en la sección 3.1 Problemas y limitaciones resulta un problema grave.

(3) Alineado de corpus paralelos

El alineado de corpus paralelos consiste en llevar a cabo correspondencias entre un texto y su traducción, ya sea nivel de palabras, de frases, de párrafos etc.

Al intentar obtener el mayor grado de automatización posible durante el proceso de creación de corpus paralelos, se optó por estudiar sólo aquellos alineadores que fuesen no supervisados, es decir, que no requieran intervención humana y se realicen de forma automática. La contrapartida de esta estrategia es que hay que tolerar cierta tasa de error.

El alineado representa en sí mismo otro reto dentro del subcampo de las ciencias de la computación del *Machine Learning*, en el que conviven diversas aproximaciones y estrategias estadísticas que varían en base a qué y cómo se realizan las correspondencias: número de palabras, tamaño de frases, apoyo de diccionario, apoyo de traducciones, entre otros.

(4) Indexación y consultas sobre corpus

Indexar implica dotar de índices o referencias al corpus de tal modo que el sistema de búsquedas pueda acceder a la información sin tener que recorrer el corpus en su totalidad. El objetivo de utilizar un sistema de indexación es dotar de mayor rapidez a las consultas sobre corpus de gran tamaño y de este modo no limitar la funcionalidad y usabilidad del software. El hecho de posibilitar búsquedas más rápidas implica que el sistema de indexación y de consultas dependan el uno del

otro. La elección del formato de las consultas sobre el corpus se realizará en función de las opciones que permita el sistema de indexación seleccionado.

El sistema de indexación está estrechamente ligado al tratamiento de datos, y más concretamente a la mejora del rendimiento de las bases de datos. Las aproximaciones existentes se dividen en tres: (1) basadas en tecnologías estándar de bases de datos como SQL (Date & Darwen, 1993), (2) en funcionalidades de los lenguajes de programación o (3), en la utilización de recursos específicos para el tratamiento de corpus lingüísticos, como el software *The IMS Open Corpus Workbench (CWB)*[16] (Evert & Hardie, 2011).

La elección del sistema de indexación resulta crucial para el desarrollo del framework lingüístico ya que influye en el formato de las alineaciones y etiquetados lingüísticos, en caso de ser utilizadas.

(5) Obtención de estadísticas

Las estadísticas son una parte fundamental de los estudios basados en corpus lingüísticos en distintas fases de la investigación, tanto en la selección y descripción de datos, como en el análisis de resultados o la verificación de las hipótesis de investigación

Las pruebas estadísticas más habituales son:

- **Métodos de estadística descriptiva**: A través de la frecuencia de aparición de un fenómeno dado en un corpus o sección concreta de un corpus. Estas frecuencias de aparición a veces aparecen normalizadas para posibilitar comparaciones con otros estudios, por ejemplo, palabras por millón. Empleando estos métodos se pueden obtener listas de frecuencias, por ejemplo.

[16] De aquí en adelante se utilizará el término CWB en lugar de *The IMS Open Corpus Workbench*.

- **Pruebas de significación estadística**, también denominados *significance tests* en inglés. Estas pruebas permiten evaluar la probabilidad de que un determinado fenómeno sea una coincidencia como consecuencia del azar (McEnery & Hardie, 2012, pág. 51). Las pruebas de significación estadística más comunes son *chi-square*, *log-likelihood* y *t-test*. Empleando estos métodos se pueden obtener, por ejemplo, listas de palabras clave.

La extracción de estadísticas depende del tipo de sistema de indexación utilizado, puesto que es necesario acceder al corpus para obtener los datos numéricos. Por este motivo es frecuente que el sistema de indexación proporcione funciones estadísticas básicas. En la actualidad, muchas investigaciones, tal y como recogen (Hardie, 2012, pág. 405) o (Meurer, 2012, pág. 50), trabajan en ampliar el rango de funciones estadísticas mediante el empleo paquetes estadísticos basados en R (R Core Team, 2017), un lenguaje de programación orientado sobre todo a la minería de datos y la estadística.

(6) Creación del interfaz de usuario

El interfaz de usuario debe satisfacer unas condiciones mínimas de usabilidad y accesibilidad. Las principales son:

- Tener presente que el usuario final es un lingüista que no tiene porqué poseer conocimientos técnicos.
- Es recomendable que el framework imite o emule, dentro de lo posible, los puntos fuertes de los recursos existentes en otros interfaces de acceso a corpus ampliamente utilizados, como el CORPES XXI (Real Academia Española, 2017a), el *Corpus of Contemporary American English* (COCA) (Davies, 2008) u otro interfaces utilizados frecuentemente. De este modo los usuarios no tendrán que aprender nuevos patrones de uso.
- Poseer una documentación completa en la que se detallen las instrucciones de uso, aplicaciones empleadas y definición de los conceptos lingüísticos, computacionales y estadísticos además de los componentes utilizados.

- Mostrar la información pertinente de forma clara y precisa. Por ejemplo, los resultados de una concordancia.
- Permitir exportar los resultados de las consultas y las estadísticas y así facilitar la recolección de datos y evitar la repetición innecesaria del análisis.

Para concluir esta sección queda por describir como se relacionan las tecnologías descritas con el objetivo de crear y consultar un corpus en un framework como el de la Figura 2:

(1) El flujograma básico para el tratamiento de corpus lingüísticos comienza con una colección de documentos de texto plano denominada Corpus 0 o catálogo de corpus. Cada uno de los textos que forma el Corpus 0 ha de ser limpiado.

(2) A continuación, se codificará cada uno de los documentos en formato UTF-8, si es que este no fuera el formato de codificación actual de cada documento.

(3) El siguiente paso consiste en dividir cada documento en distintas unidades (palabras, frases, párrafos, capítulos o cualquier otra unidad definida por el programador) por medio de un tokenizer.

(4) El Corpus 0 resultante se anota con la información procedente del tokenizer, y de acuerdo con el tipo de corpus que se quiere construir. Se genera el Corpus 1 o corpus marcado, normalmente en formato XML o derivado.

(5) Una vez anotado el corpus, hay dos fases opcionales -pero de uso muy generalizado- de tratamiento del Corpus 1: incorporación de etiquetados lingüísticos y alineación de textos en el caso de corpus paralelos. Ambos suelen resolverse utilizando programas de terceros (ajenos al desarrollo original del framework) a causa de su complejidad.

- Incorporación de etiquetados lingüísticos: gramatical, semántico, retórico, sintáctico o fonético. Por ejemplo, citar al etiquetador gramatical Treetagger.

- En el caso de que el corpus sea paralelo se llevará a cabo una alineación de los textos, ya sea a nivel de frase, más habitual, de palabra u otra unidad. Por ejemplo, GIZA++.

(6) Con la incorporación de las anotaciones lingüísticas y alineados al corpus, se obtiene el Corpus 2 o corpus anotado (la ausencia de anotaciones o alineados también se refleja). Esta incorporación se realiza por medio de algún parser creado para esta función concreta.

(7) A continuación, el Corpus 2 se indexa e incorpora en un sistema de búsquedas con el objetivo de aumentar la velocidad de las consultas. Para llevar a cabo la indexación es necesario transformar el Corpus 2 a un formato específico. Tal y como se mencionó con anterioridad, los sistemas de consultas pueden ser propios o de terceros, por ejemplo CWB (Evert & Hardie, 2011).

(8) El resultado final es la disponibilidad del corpus, almacenado en un servidor o cargado en una aplicación. Alcanzada esta fase, se pueden hacer consultas y/o extraer estadísticas a través de un interfaz de usuario o *front-end* (capa de presentación).

2.4 Convenciones utilizadas en los términos relacionados con la creación y análisis de corpus lingüísticos

Cada escuela lingüística interpreta los conceptos relacionados con la creación y análisis de corpus de una forma distinta. Aunque el concepto base es el mismo existen diferentes connotaciones que los diferencian. A continuación, se repasa la variabilidad terminológica más habitual que pudiera llevar a confusiones conceptuales en esta tesis:

- **Colocación**: Una colocación es una secuencia de palabras que aparecen habitualmente juntas en los textos. En la actualidad es muy habitual hablar de colocaciones estadísticas, que emplean tests de significación como *Z-score*, *Mutual Information* o *Log-likelihood*, y que permiten obtener qué secuencias de palabras se dan en un texto más a menudo que lo que cabría esperar por azar. En esta tesis doctoral sólo se utilizarán colocaciones basadas en frecuencias.
- **n-grama**: El concepto de n-grama varía en cuanto a significado y nomenclatura dependiendo de si la definición es proporcionada por un lingüista o por un especialista informático. Para un informático, un n-grama es simplemente una secuencia de n palabras o n caracteres. Así un n-grama de tamaño uno es un unigrama, de tamaño dos, bigrama o diagrama, etc. Los lingüistas añaden la aclaración de que tiene que darse en un texto de forma más frecuente que otra, con lo que sólo es necesario añadir las estadísticas cuantitativas necesarias a la definición proporcionada anteriormente. El concepto de n-grama recibe denominaciones alternativas. Para (Biber, Johansson, Leech, Conrad, & Finegan, 1999) (Cortes, 2013) (Salazar, 2014) este fenómeno es un *lexical bundle*, se trataría de un *cluster* o *word cluster* para (Scott & Tribble, 2006) (Saber, 2012), un *multi-word combination* para (Gries, 2008) o *chain* para (Stubbs, 2002) entre otras denominaciones. Todas las opciones comparten la esencia de expresión multi-palabra. En esta tesis doctoral se utilizará n-grama para referirse a una secuencia de palabras de tamaño n que se da en un texto de forma más frecuente que otra.
- **Anotación y etiquetado**: La anotación puede definirse como la adición de información interpretativa y lingüística a un corpus en formato electrónico (Leech, 1997, pág. 2). La anotación también se refiere al símbolo lingüístico final del proceso, es decir, a la etiqueta lingüística añadida al corpus (Leech, 1997, pág. 2). Cuando se habla de etiquetado únicamente se refiere al

proceso de añadir etiquetas, lingüísticas en este caso, a un corpus con el objetivo de anotarlo. El etiquetado es un concepto implícito en la anotación.

- **Etiquetado retórico**: En esta tesis doctoral se habla de etiquetado retórico para referirse al proceso de añadir información retórica en un corpus lingüístico mediante el etiquetado de sus secciones. La información retórica se refiere al conjunto de estructuras internas de un texto que hacen que sea reconocible como miembro de un género textual concreto.[17]
- **Búsquedas y concordancias**: Los términos búsqueda sobre corpus y muestra de concordancias son utilizados en esta tesis doctoral de forma análoga, refiriéndose a la definición de concordancia propuesta por (Baker, Hardie, & McEnery, 2006, págs. 42-43), es decir, la lista de todas las ocurrencias de un término de búsqueda particular en un corpus, presentada en su contexto de aparición, normalmente unas pocas palabras a la izquierda y derecha del término de búsqueda, lo que es frecuentemente denominado KWIC (*Key Word In Context* – palabra clave en contexto)[18], uno de los formatos de concordancias más habituales y populares entres los usuarios de corpus.

[17] Para más información sobre el etiquetado retórico consultar las fuentes (Swales, 1990), (Swales, 2004) y (Biber, Connor, & Upton, 2007). Un ejemplo de aplicabilidad práctica del etiquetado retórico se puede encontrar en (Labrador, Ramón, Alaiz-Moretón, & Sanjurjo-González, 2014).

[18] De aquí en adelante se utilizará el término KWIC en lugar de *Key Word In Context.*

Capítulo

3

Estado de la cuestión: Software especializado para el tratamiento de corpus lingüísticos

3.1 Problemas y limitaciones

Autores como (McEnery & Hardie, 2012, pág. 36) o (Anthony, 2013a, págs. 146-147) defienden que el software facilita la labor del investigador lingüista en términos incalculables pero a la vez también limita sobremanera qué se puede hacer o no con el corpus disponible. Es decir, la investigación lingüística se ve restringida por las limitaciones del software y/o su nivel de usabilidad.

Los principales problemas y limitaciones que han surgido durante el desarrollo del software para el tratamiento de corpus desde los años 50 hasta la actualidad son los siguientes:

- Compilación del corpus.
- Usabilidad del software.

- Búsquedas sobre corpus: tamaño de corpus y tiempo de ejecución.
- Estadísticas.
- Replicabilidad de estudios sobre corpus.
- Incompatibilidad de programas: entornos operativos y software anticuado.
- Tipos de corpus soportados.
- Otros:
    - Codificación de caracteres no latinos.
    - Anotación de documentos.
    - Etiquetados lingüísticos.

3.1.1 Compilación de corpus lingüístico

Compilar los textos que forman un corpus lingüístico para su procesamiento por parte de un ordenador es una de las tareas más importantes en la creación de cualquier corpus. Para que un corpus pueda ser procesado por un ordenador ha de estar digitalizado. Antes de la aparición y difusión de internet, la obtención de textos digitalizados se hacía mediante el escaneado de documentos, con los consiguientes fallos en el reconocimiento de caracteres y sobre todo el cansancio generado de la necesidad de escanear cada documento de forma manual.

Actualmente existe una prácticamente infinita cantidad de recursos digitalizados y salvo en la realización de corpus formados por documentos muy antiguos, la limitación ha sido resuelta.

Otro aspecto relacionado con la compilación del corpus es su ubicación. Existen dos estrategias principales:

- **Corpus local**: almacenado en un equipo personal y accesible por tanto solamente para la persona que lo posee. Para su análisis ejecutará software instalado en su equipo.

- **Corpus en servidor web**: alojado en un servidor web que posibilita el acceso a través de internet. Para su análisis ejecutará un simple navegador web.

Ambas estrategias poseen sus ventajas y sus inconvenientes:

(1) El corpus local es exclusivo de la persona que maneja el equipo, ajeno a posibles robos de datos a través de la red; por otro lado, su análisis por parte de otros usuarios requiere que se facilite el corpus lingüístico; además los corpus de gran tamaño tienen problemas de procesamiento derivados de la menor capacidad computacional de los ordenadores personales en comparación con los servidores.

(2) Un corpus alojado en un servidor web posee todas las ventajas intrínsecas de internet: acceder a los corpus desde cualquier dispositivo en cualquier momento; no son necesarias instalaciones adicionales, el navegador web sirve de interfaz para llevar a cabo los distintos análisis, que se ejecutan en *background* en servidores con gran potencia computacional. Como desventaja principal está la de proporcionar corpus con datos sensibles o con protección intelectual a terceros al servidor web, cuya seguridad puede ser comprometida aún con medidas de prevención de acceso no autorizado, además de los posibles costes derivados del uso de servidores.

3.1.2 Usabilidad

La dificultad de uso del software específico para el tratamiento de corpus lingüísticos es un factor crucial. Se considerarán tres limitaciones principales relacionadas con la usabilidad de este tipo de software:

- Medios informáticos necesarios para ejecutarlo.
- Limitaciones técnicas del software.
- Necesidad de altos conocimientos técnicos para su uso.

En los primeros programas de búsqueda, denominados *concordancers*, la presencia de un técnico especialista era indispensable en casi todo el proceso. En los actuales

frameworks para el tratamiento de corpus lingüísticos coexisten aquellos que no requieren asistencia técnica pero que poseen funcionalidades limitadas, con otros más complejos que precisan de conocimientos técnicos avanzados a la hora de incorporar corpus en el framework.

La tendencia a descomponer el análisis de lenguaje varias etapas (Dale, 2010, pág. 4) (Nadkarni, Ohno-Machado, & Chapman, 2011, pág. 548), por ejemplo el etiquetado gramatical, provoca una atomización de los recursos que convierten a los programas en una especie de cajas negras secuenciales, en las que unos datos entran y salen transformados. La necesidad de unir las entradas y salidas de los distintos procesos NLP genera incompatibilidades entre programas y la casi estricta necesidad de disponer de conocimientos técnicos relacionados con el procesamiento de lenguaje.

3.1.3 Búsquedas sobre corpus

Las búsquedas sobre corpus lingüísticos sin el empleo de ordenadores tienen su origen en los estudios realizados durante la Edad Media en la creación de índices de palabras. Muy especialmente sobre la biblia Vulgata de San Jerónimo en numerosos monasterios europeos, por ejemplo, *Hugo of St. Cher* (McEnery & Hardie, 2012, pág. 37), o sobre otros escritos, como o San Isidoro de Sevilla que de acuerdo con (Carrera-de la Red, 1998) "*al escribir sus tratados, va introduciendo en el texto numerosas aclaraciones sobre las palabras que sospecha pueden encerrar dificultades para que las comprendan sus lectores*". El tiempo dedicado en realizar estos estudios evidenció que la realización de trabajos similares en textos de mayor tamaño resultaría inabarcable.

Posteriormente, (Abercrombie, 1965) describe una serie de pseudo-procedimientos que proporcionarían unos excelentes resultados en caso de ser aplicados, pero que no son realizables en términos de tiempo.

En los años 50, por primera vez y como consecuencia de la aparición de los primeros *mainframes*, también llamados ordenadores centrales, surgió la posibilidad de emplear los ordenadores para procesar los corpus lingüísticos. Estos primeros programas ejecutados en los *mainframes* permitían realizar búsquedas sobre corpus lingüísticos. Concretamente posibilitaban la ejecución y muestra de concordancias (ver definición en página 36).

Los programas que realizan concordancias se denominan con el término inglés *concordancer*. De acuerdo con (Baker, Hardie, & McEnery, 2006, pág. 44) un *concordancer* pueden definirse de la siguiente forma:

> *"A software tool that searches through a corpus for each instance of a given word, phrase or other element and the immediate context in which each instance occurs, to create a concordance."*

La forma más frecuente en la que se muestran los resultados de una concordancia es a través del formato KWIC que alinea los resultados en distintas líneas, de tal forma que la expresión buscada aparezca centrada junto a un número de palabras previas y posteriores, que se denomina contexto.

Estos primeros *concordancers* eran propietarios, pertenecientes a grupos de investigación que podían permitirse el uso y adquisición de un *mainframe*. La replicabilidad, como se detallará en el apartado 3.1.5 Replicabilidad, era muy limitada ya que los programas de concordancias sólo podían ejecutarse en el *mainframe* específico sobre el que se habían diseñado.

Tener la posibilidad de llevar a cabo búsquedas sobre corpus a través de un ordenador también acarreó la necesidad de acelerar la velocidad con la que se llevaban a cabo. De acuerdo con (Kennedy, 1998, pág. 7), en los años 70 una búsqueda básica como "*when*" en un corpus formado por un millón de palabras tardaba en torno a una hora en un primitivo *mainframe*, en los 80 esta misma concordancia sobre un IBM PC tardaba unos pocos minutos.

Una solución a este problema es la indexación el corpus lingüístico,[19] es decir, dotar de índices o referencias al corpus de tal modo que el sistema de búsquedas pueda acceder a la información sin tener que recorrer todo el corpus. De esta forma, el tiempo necesario para realizar esa misma búsqueda será mucho menor.

Indexar un corpus también tiene limitaciones relacionadas con el tamaño máximo soportado por el programa de indexación. Por ejemplo, CWB (Evert & Hardie, 2011) que usa como motor de indexación la tecnología CQP (Evert, 2009), procesa como máximo 2.1 billones de palabras en su versión de 64 bits.

3.1.4 Estadísticas

Según (McEnery & Hardie, 2012, pág. 41), dado un corpus un ordenador puede realizar dos tareas básicas: (1) contar apariciones en el texto, lo que se denomina listas de frecuencias y (2), ejecutar una concordancia y mostrarla. Estas dos tareas permiten obtener la lista de palabras clave (o *keyword list* en inglés), que no es más que una abstracción de la lista de frecuencias (o *frequency list* en inglés) en comparación con un corpus de referencia, y colocaciones o unidades fraseológicas (en inglés *collocates*) de las concordancias, secuencia de palabras que surge más a menudo. Otra estadística muy popular son los n-gramas (o *n-grams* en inglés).

La presencia o ausencia de estas estadísticas es otra de las limitaciones del distinto software.

3.1.5 Replicabilidad

La replicabilidad, es decir, la cualidad de que los resultados de un estudio se repitan bajo las mismas condiciones, es una de las características principales de cualquier

[19] De acuerdo con la aproximación que aparece en (Anthony, 2013a, pág. 152) y (Sharoff, 2006, pág. 541) , si el corpus es mayor de 100 millones de palabras, es recomendable que se encuentre indexado para facilitar la búsqueda en tiempo y forma.

experimento científico. Se trata de unos de los mayores inconvenientes derivados del uso de distinto software para el tratamiento de corpus lingüísticos.

En los *mainframes*, resultaba prácticamente imposible replicar los resultados a no ser que se empleara el mismo equipo y programa ya que, como se ha mencionado, el programa de búsqueda era creado para funcionar en un *mainframe* específico. Lo mismo sucedía en el caso de los programas para tratar los datos con posterioridad, que también eran particulares del programa de búsquedas y del *mainframe* sobre el que se ejecutaba. Todo ello resultaba tedioso, tanto para programadores como para usuarios lingüistas.

En resumen, la replicabilidad sólo era posible accediendo al mismo *mainframe* empleado originalmente. En la actualidad, la replicabilidad se puede conseguir utilizando el mismo software y corpus. Sin embargo, conseguir unos resultados estadísticos idénticos utilizando distinto software no siempre es posible, esto es debido a errores residuales (Anthony, 2013a, págs. 149-151), como por ejemplo a cómo se define el concepto de palabra.

3.1.6 Incompatibilidad de programas: entornos operativos y programas anticuados

En la Figura 2, el tratamiento de corpus lingüísticos está compuesto por distintas tareas o actividades que pueden realizarse o bien por un framework integral para el tratamiento de corpus lingüísticos o bien por medio de herramientas o aplicaciones independientes cuyas entradas y salidas se interconecten.

Por ejemplo, en muchas ocasiones el etiquetado gramatical obtenido es incompatible con el formato aceptado por un *concordancer*, que en ese caso simplemente obvia el etiquetado. Cada software tiene unos requisitos operativos distintos, es posible que una aplicación de búsqueda sea compatible únicamente con un entorno Windows, y el programa de extracción de estadísticas sólo sea operativo

en sistemas Linux. Si a todo ello le añadimos el empleo de software obsoleto cuya instalación resulta imposible en un entorno operativo moderno, la incorporación de dicho software en un flujograma de tratamiento de corpus lingüísticos resulta cuanto menos desaconsejable.

Una de las posibles soluciones es la de encapsular todo el proceso en una única aplicación, por ejemplo, por medio de un framework integral, evitando por tanto la incompatibilidad de sistemas operativos y los errores derivados de discrepancias de las entradas y salidas de datos entre los distintos programas. De este modo, se asegura el funcionamiento óptimo en cualquier parte del proceso.

3.1.7 Tipos de corpus soportados

Cada software especializado en el tratamiento de corpus lingüísticos define los tipos de corpus que soporta (descritos en el apartado 2.1.2 Tipos de corpus). Como consecuencia del predominio anglosajón, prácticamente la totalidad de los mismos están centrados en la lengua inglesa, y por ello es muy poco frecuente encontrar software que no sea capaz de procesar corpus monolingües en dicha lengua. También es habitual que puedan procesar corpus en otros idiomas, sin embargo, el problema surge de la necesidad de construir corpus bi-/multilingües, ya sean paralelos o comparables. Como se describirá en la sección 3.3 Análisis de software disponible, muy pocos gestores de corpus son capaces de procesar corpus bi-/multilingües paralelos y los que pueden hacerlo requieren asistencia técnica especializada. Por otro lado, no existe software específico para la construcción de corpus comparables,[20] por motivos más relacionados con su aplicabilidad que por su complejidad técnica.

[20] Corpógrafo V4 (Maia & Matos, 2008) supuestamente permitía el tratamiento de corpus comparables multilingües, pero no con una sola lengua. El framework no se encuentra operativo en la actualidad. Ver 3.3.1.8 Corpógrafo para más información.

3.1.8 Otros

3.1.8.1 Codificación de caracteres no latinos

Tal y como sucede en muchas disciplinas, la preponderancia del mundo anglosajón implica que el inglés sea el eje sobre el que desarrollan todas las novedades tecnológicas. Por ello, la digitalización de textos en lenguas distintas al inglés implicó la necesidad de resolver el problema de la codificación de símbolos de otros alfabetos. Intentar codificar estos símbolos supuso un auténtico desafío, por ejemplo, consultar (Baker, McEnery, Leisher, Cunningham, & Gaizauskas, 2000). La aparición del formato Unicode (The Unicode Consortium, 2006), más concretamente la versión UTF-8, resolvió esta limitación de forma definitiva. Unicode persigue la universidad, la uniformidad y la unicidad para lograr un estándar que permita representar todos los caracteres de las lenguas existentes en el mundo, así como de otras lenguas muertas, en un solo estándar de codificación de caracteres.

3.1.8.2 Anotación

Según el tipo de estudios realizados sobre los corpus lingüísticos, la anotación lingüística puede ser necesaria. La creación del estándar XML dotó de una estandarización a la anotación, de tal forma que estas anotaciones sobre corpus resultaban reutilizables y replicables. XML permite marcar, anotar o añadir metadatos a la información proporcionada en el corpus y, además, aumenta la velocidad de cualquier tarea de búsqueda en comparación con el procesamiento de texto plano. Si bien es cierto que el formato XML no siempre es estrictamente respetado en el diverso software, sí que es reconocido, y permite la creación de scripts específicos para su tratamiento sin grandes dificultades.

Otra funcionalidad relacionada con la anotación es la inclusión de etiquetas lingüísticas. Dejando de lado el formato seguido para anotar un corpus, por ejemplo, notación prefija seguida de guion o en formato *one-word-per-line* (una palabra por

línea), es más importante establecer una notación universal para cada tipo de etiquetado, de tal forma que el etiquetado pueda ser replicable y comprensible por otros investigadores. Sin embargo, en lugar de emplear un formato universal que permita la futura redistribución de los corpus etiquetados, lo más frecuente es que cada grupo de lingüistas establezca sus propios criterios.

En los años 90, hubo intentos por crear un estándar de etiquetas gramaticales y sintácticas de tal forma que los recursos creados en los distintos proyectos no se abandonaran después de que estos finalizaran (Calzolari, Choukri, Fellbaum, Hovy, & Ide, 1999). Hubo tentativas tanto por parte norteamericana, a través de *Linguistic Data Consortium* (LDC) (University of Pennsylvania, 2017), como por parte europea, a través de EAGLES (EAGLES, 1996), e incluso más recientemente algunos autores como (Petrov, Das & McDonald, 2011). Sin embargo, a día de hoy este este problema sigue sin resolverse.

Un mayor grado de coordinación entre los distintos grupos de investigación, tal y como defiende (Meyers, 2009) podría ser la solución. Pero en el mundo real cada programador sigue las pautas establecidas por el lingüista, y más frecuentemente las establecidas por el programa de etiquetado, por ejemplo Treetagger (Schmid, 1995). Pero incluso en el caso de emplear un mismo programa, estos suelen diferir en el conjunto de etiquetas empleado para cada idioma. Si se llegara a un acuerdo se posibilitaría el empleo de una gran cantidad de recursos lingüísticos (Calzolari, Choukri, Fellbaum, Hovy, & Ide, 1999) que ahora están en desuso.

Todos estos inconvenientes suponen una barrera más que dificulta la elaboración de corpus bi-/multilingües, y más concretamente la construcción de grandes corpus paralelos.

3.2 Clasificación del software para el tratamiento de corpus lingüísticos

El software especializado para el tratamiento de corpus lingüísticos[21] ha evolucionado en paralelo al desarrollo de las distintas tecnologías que lo sustentan, como no podía ser de otro modo. Este software ha progresado desde las simples herramientas y aplicaciones informáticas que únicamente permitían realizar búsquedas sobre los corpus, hasta los complejos frameworks existentes en la actualidad.

De acuerdo con la Figura 2, existen multitud de tareas o actividades relacionadas con el tratamiento de corpus lingüísticos. Estas tareas pueden llevarse a cabo por múltiples herramientas y aplicaciones. Entre estas tareas, hay dos que se consideran indispensables:

- Compilación de los textos que forman un corpus, es decir, la capacidad de definir el corpus sobre el que realizar el estudio.
- Búsquedas en el corpus, esto es, ejecutar concordancias y mostrarlas por pantalla.

Por lo tanto, la siguiente clasificación tendrá en cuenta solamente el software que permita ejecutar consultas sobre corpus lingüísticos compilados. Es decir, el software analizado es un programa de concordancias o tiene uno integrado como uno de sus componentes

La clasificación propuesta es una variante de la establecida por (McEnery & Hardie, 2012, págs. 37-48). McEnery & Hardie categorizan al software para el tratamiento de corpus lingüístico, que denominan de forma generalista *concordancer*, en cuatro

[21] Hay que acentuar el hecho de que el software emplee corpus lingüísticos como fuente de datos. Existe software como OpenCalais (Thomson Reuters, 2017) o NVivo (QSR International Pty Ltd., 2017) que analizan grandes cantidades de información no estructurada y extraen distintas características lingüísticas automáticamente (ej. semánticas) o muestran distintas estadísticas. Esto no es un corpus lingüístico tal y como se define en el apartado 2.1.1 Definición de corpus lingüístico.

generaciones temporales. La clasificación que se propone en esta tesis doctoral se basa en ésta, y pone un mayor énfasis en los problemas y limitaciones descritos en la sección 3.1 Problemas y limitaciones.

Es decir, cada generación propuesta por McEnery & Hardie es analizada con el fin de discernir sobre las limitaciones y problemas que poseen, y si se da el caso, sobre las soluciones que incorporan para resolver los problemas y limitaciones de las generaciones anteriores.

El criterio principal para establecer la clasificación es el temporal, ya que la práctica totalidad de funcionalidades y mejoras computacionales que distinguen las diversas generaciones son consecuencia del desarrollo tecnológico asociado a cada intervalo de tiempo.

3.2.1 La aparición del corpus en formato electrónico

En 1951 Roberto Busa (Busa, 1974) creó un corpus capaz de ser procesado por un ordenador por medio de tarjetas perforadas (Winter, 1999). Este primer corpus estaba formado por textos basados en distintos trabajos de Santo Tomás de Aquino. Al disponer de un corpus capaz de ser procesado por un ordenador se demostró que la tecnología permitía llevar a cabo, de forma rápida y concisa, tareas de indexación y recuperación sobre cualquier texto disponible en formato electrónico (Chan, 2014, pág. 437).

A raíz de este hecho surgieron los primeros estudios basados en corpus electrónicos, por ejemplo (Kučera & Francis, 1967) o (Quirk, Greenbaum, Leech, & Svartvik, 1972). A partir de este momento comenzaron a desarrollarse los programas especializados en el tratamiento de corpus lingüísticos, más concretamente los primeros *concordancers*.

3.2.2 Primera generación

La primera generación de software para el para el tratamiento de corpus lingüísticos abarca las décadas de los 50, 60 y 70. Se trataban de simples *concordancers* ejecutados en *mainframes*.

Esta generación fue pionera, supuso un paso adelante en los estudios sobre corpus lingüísticos, pero estaba repleta de problemas:

- La compilación del corpus requería un informático especializado. La ubicación del corpus era local y había que emplear el equipo específico que contenía el corpus y el *concordancer*.
- El acceso a los *mainframes* era muy limitado, algo que restringía la utilización de los *concordancers*. La mayor parte de las universidades del mundo carecía de los recursos económicos para disponer de un *mainframe*, y aunque dispusieran de uno, el acceso era compartido con el resto de la comunidad universitaria. Resultaba indispensable la presencia de un técnico especializado para interactuar con el equipo. La usabilidad desde el punto de vista del usuario lingüista era por tanto nula.
- Tal y como se mencionó en el apartado 3.1.3 Búsquedas sobre corpus, la ejecución de concordancias sobre los corpus consumía mucho tiempo. Por ejemplo, el software Discon (Clark, 1966) tardaba en torno a 4 minutos en procesar 1000 líneas de poemas (Anthony, 2013a, pág. 151)
- Las estadísticas eran realizadas por programas externos que realizaban labores de conteo básicas, como la elaboración de listas de frecuencias.
- Los *concordancers* tenían que hacerse a medida del *mainframe* donde se ejecutaban, algo que repercutía en la replicabilidad de los estudios en otros equipos, que era prácticamente nula.
- La incompatibilidad con otros *mainframes* era una constante.
- El hecho de disponer de un *concordancer* significó una gran revolución tecnológica. Los problemas de codificación de caracteres no latinos, la

anotación de documentos, la adición de etiquetados lingüísticos o el procesado de corpus distintos al monolingüe en lengua inglesa no eran consideradas aún funcionalidades prioritarias.

Ejemplos de software perteneciente a esta generación son: CLOC (Reed, 1977), Concordance Generator (Smith, 1966), Discon o Drexel Concordancer Program (Dearing, 1966).

3.2.3 Segunda generación

La segunda generación comienza en el año 1981 coincidiendo con la aparición de los ordenadores personales IBM PC, y finaliza a mediados de la década de los 90. La creación de los ordenadores personales fue un hito en el desarrollo tecnológico a nivel mundial. El abaratamiento de los costes de producción facilitó el acceso a los ordenadores y el desarrollo de nuevas aplicaciones en todos los campos científicos. Esta situación contrastaba con los altos costes de derivados de la compra y mantenimiento de los *mainframes*. Por todo ello, el desarrollo de software especializado en el tratamiento de corpus se benefició de un crecimiento exponencial.

La segunda generación mejoró principalmente en términos de usabilidad. Pero como consecuencia del aumento de ésta, problemas que en la primera generación no parecían tan importantes adquirieron mayor trascendencia, por ejemplo, la codificación y la anotación de documentos.

Las características detalladas de esta generación son:

- La compilación del corpus no requería un técnico especializado de forma indispensable, lo que supuso un paso adelante con respecto a la anterior generación. La ubicación del corpus seguía siendo local. Sin embargo, el acceso al mismo se realizaba desde los equipos personales de los investigadores, no era necesario acudir a un equipo centralizado. Internet aún estaba en desarrollo, luego no era una opción de acceso.

- Cualquier investigador, sin necesidad de disponer de conocimientos informáticos avanzados, podía usar un *concordancer*. Esto supuso un éxito en el estudio de corpus lingüísticos que creció espectacularmente gracias al aumento de usabilidad del software de esta generación.
- Aunque la proliferación de ordenadores personales fue una excelente notica, su menor potencia en comparación con los *mainframes* impedía o limitaba el análisis de corpus de gran tamaño y/o las búsquedas resultaban mucho más lentas.
- Los programas de estadística seguían siendo en su mayoría independientes, como sucedía en la generación anterior. Aunque también se vieron beneficiados del crecimiento exponencial del desarrollo tecnológico.
- Los *concordancers* ahora podían utilizarse en diferentes equipos informáticos, algo que repercutía directamente en la replicabilidad.
- El hecho de que cualquier investigador del mundo pudiera realizar análisis sobre sus corpus, muchos de ellos no escritos en inglés, hizo que los problemas de codificación de caracteres no latinos adquirieran importancia a la hora de construir corpus monolingües.
- La ausencia de una estandarización en la anotación de documentos y en los etiquetados comienza a ser un problema prioritario a solucionar.
- El paulatino aumento del número de ordenadores personales y el hecho de que el software desarrollado fuese compatible con la mayoría de los equipos informáticos influyó positivamente, no sólo en la usabilidad, sino también en la eficiencia y compatibilidad de los programas relacionados con el tratamiento de corpus lingüísticos.

Ejemplos de programas de concordancias de esta generación son Kaye Concordancer (Kaye, 1990), Longman Mini-Concordancer (Chandler & Tribble, 1989), Micro-OCP (Hockey, 1993) o MicroConcord (Scott & Johns, 1993).

3.2.4 Tercera generación[22]

La tercera generación de software para el tratamiento de corpus lingüísticos surge a mediados de los 90, coincidiendo con la primera versión de WordSmith Tools en 1996 (Scott, 1996) y aún perdura. En esta generación el software es mucho más robusto y fácil de usar. Comienzan a aparecer los primeros frameworks para el tratamiento de corpus lingüísticos tal y como fueron definidos en el apartado 2.3.1 Definición de software y conceptos relacionados, como el mencionado WordSmith o AntConc (Anthony, 2014a). Estas aplicaciones permiten compilar corpus, ejecutar concordancias muchos más complejas que incluyen etiquetados lingüísticos o incluso permiten la posibilidad de extraer estadísticas empleando el mismo software.

Las mayoría de los frameworks fueron creados por grupos de trabajo reducidos, liderados por una persona y con la ayuda puntual de otras, como es el caso de Laurence Anthony con AntConc, Mick O'Donnel con UAM Corpus Tool (O'Donnel, 2008), Max Silberztein con Nooj (Silberztein, 2005) o Mike Scott con WordSmith, por citar algunos. Esta característica repercute negativamente en la idea futura de crear un entorno colaborativo, donde la comunicad científica pudiese añadir nuevas funcionalidades e integrarlas en un único framework. En estas circunstancias los entresijos del software son sólo conocidos por el programador, que ante la ausencia de colaboración incrementa el número de parches y hábitos no recomendados, como por ejemplo la ausencia de documentación.

La tercera generación se vio beneficiada de dos importantes acontecimientos:

(1) La implementación del estándar de codificación Unicode (The Unicode Consortium, 2006), más concretamente la versión UTF-8, que posibilita que

[22] El análisis de esta generación se basa en las primeras versiones del software perteneciente a la misma, ya que en la actualidad se siguen desarrollando frameworks que se adscriben a esta generación, pero que han superado algunas limitaciones, como por ejemplo el uso de *collocation networks* (Phillips, 1989). Ver 3.2.6 Situación actual para más información.

no se requieran programas específicos para el tratamiento de textos con caracteres no latinos.

(2) El desarrollo e implementación del lenguaje de marcado XML, que se convierte en el lenguaje "estándar" para llevar a cabo anotaciones sobre documentos. El lenguaje XML posibilita la creación de programas genéricos para el tratamiento de textos anotados, ya no era necesario la creación programas personalizados para tal fin.

Respecto a los problemas y limitaciones analizados, la tercera generación posee las siguientes características:

- Al igual que en la generación anterior, en su mayoría tanto la compilación del corpus como el uso del software no requiere asistencia técnica especializada generalizada. El acceso se realiza de forma local.[23]
- La usabilidad mantiene sus altas prestaciones a pesar del aumento de funcionalidades. Pero surge un problema relacionado con el incremento de número de software, ya que los programas son muy similares ente sí y suelen realizar las mismas tareas (McEnery & Hardie, 2012, pág. 43). Mencionar también que la nueva configuración "compacta" de los frameworks de tercera generación hace que algunas funcionalidades de los antiguos *mainframes* como *collocation networks* (Phillips, 1989) o el análisis multi-variable (*multi-dimensional approach* en inglés*)* (Biber, 1991) no estén disponibles ni puedan emplearse con facilidad de forma externa, tal y como se describe en (McEnery & Hardie, 2012, págs. 41-43).
- El aumento de potencia, paralelo al desarrollo de la tecnología proporciona búsquedas más rápidas en corpus de tamaño medio. Sin embargo, también comienzan a crearse corpus de mayor tamaño como consecuencia del

[23] XAIRA (Burnard & Dodd, 2005), cuya primera versión fue creada en 1994, es asignado a la tercera generación. Sin embargo, también posee características de la cuarta, ya que está desarrollado bajo el paradigma cliente-servidor. Al carecer de un sistema de indexación complejo se ha optado por incluirlo en la tercera generación.

incremento de textos digitalizados. Como aún no se han integrado sistemas de indexación complejos en las búsquedas sobre los corpus, analizar con cierta eficiencia y rapidez corpus mayores de 100 millones de palabras (Anthony, 2013a, pág. 152) (Sharoff, 2006, pág. 541) resulta complicado.

- Se proporcionan estadísticas basadas en análisis cuantitativos dentro del mismo entorno, no es necesario emplear varios programas informáticos.
- Desde el campo de la lingüística, distintos proyectos intentan elaborar estándares de etiquetas lingüísticas gramaticales y sintácticas como *Linguistic Data Consortium* (LDC) (University of Pennsylvania, 2017) o EAGLES (EAGLES, 1996).
- La replicabilidad es aún más fácil de conseguir gracias a la integración de varias funcionalidades en el mismo software como consecuencia del empleo de frameworks.
- A causa de la proliferación de equipos informáticos, comienzan a utilizarse distintos sistemas operativos, basados en Windows o en Unix. Muchos programas son únicamente ejecutables en uno de los dos entornos provocando incompatibilidades.
- Aumenta la creación de corpus en lenguas distintas del inglés. Comienzan a desarrollarse versiones modificadas de software existente para manejar corpus paralelos bilingües como por ejemplo ParaConc (Barlow, 1995) o AntPConc (Anthony, 2014b). No obstante, el alineado del corpus implica el empleo de programas específicos y conocimientos de programación para incorporar las alineaciones al software pertinente.

Ejemplos -ente otros- de frameworks pertenecientes a la tercera generación son WordSmith, AntConc, UAM Corpus Tool o Nooj.

3.2.5 Cuarta generación

La principal novedad que presenta la cuarta generación con respecto a las anteriores es que se posibilita la creación y análisis de corpus lingüísticos a través de la web.

Es posible acceder y analizar un corpus desde cualquier equipo con acceso a internet, ya que los corpus se encuentran alojados en servidores externos. Además de posibilitar el acceso a los corpus a través de internet, los servidores proporcionan una potencia muy superior a la de cualquier ordenador personal, posibilitando consultas más rápidas sobre corpus de mayor tamaño.

El software perteneciente a esta generación se ha desarrollado utilizando el paradigma cliente-servidor y proporciona un *front-end* (capa de presentación) para ejecutar las consultas y posibilitar el análisis a través de cualquier navegador web. Al igual que la generación anterior, el soporte de etiquetados lingüísticos, el uso de XML, la existencia del UTF-8, para el tratamiento de textos con caracteres no latinos y la integración de estadísticas en el propio framework para el tratamiento de corpus lingüísticos, solventan las dificultades encontradas en la primera y segunda generación. En cuanto al resto de las limitaciones y problemas, esta cuarta generación se caracteriza por:

- Existen frameworks pertenecientes a esta generación que brindan la posibilidad de que el usuario compile sus propios corpus, otros en cambio imponen que un especialista informático se encargue de compilarlos a partir de los textos proporcionados por el usuario, algo que afecta negativamente a su usabilidad. Esto es consecuencia del estricto formato necesario para que sea indexado en el sistema de búsqueda. Por otro lado, los corpus se encuentran ubicados en un servidor web, proporcionando acceso desde cualquier dispositivo con acceso a internet.
- La usabilidad se incrementa gracias al empleo del navegador web que por defecto incluya el sistema operativo, el cual se utiliza como enlace entre la framework y el usuario. No es necesario instalar ningún software adicional.
- El uso del navegador web también evita las incompatibilidades surgidas de la coexistencia de distintos sistemas operativos existentes en la tercera generación.

- Las búsquedas son más rápidas gracias a la inclusión de sistemas de indexación complejos y a la mayor capacidad computacional de los servidores. Es posible realizar búsquedas más eficientes y rápidas sobre corpus aún más grandes. Los sistemas de búsqueda e indexación más empleados son:
  (1) Sistemas basados en bases de datos SQL (Date & Darwen, 1993), como corpus.bye.edu (Davies, 2017a).
  (2) Sistemas basados en CQP (Evert, 2009), que es un sistema de indexación y búsqueda de corpus que forma parte CWB (Evert & Hardie, 2011) desarrollado en el I*nstitut für Maschinelle Sprachverabeitung* de la *University of Stuttgart* – Universidad de Stuttgart. Es uno de los más utilizados en diversos interfaces web gracias a su potencia para tratar corpus formados por millones de palabras con múltiples capas de anotación. Por ejemplo, por citar uno de ellos, Sketch Engine (Kilgarriff, et al., 2014).
  (3) Combinación de ambos métodos, como BNCweb (Hoffmann, Evert, Smith, Lee, & Berglund-Prytz, 2008) y su clon CQPweb (Hardie, 2012) que combinan CQP y SQL.
  (4) Sistemas que utilizan otros métodos de indexación y/o búsquedas, como Corpuscle (Meurer, 2012), que está programado en LISP (Steele, 1990) y emplea un sistema de indexación similar a CQP, Wmatrix (Rayson, 2009), programado en Perl (Wall, 2017), el corpus de acceso online para el idioma alemán DWDS (Geyken, Didakowski, & Siebert, 2008), que utiliza el sistema de indexación DDC (Garabik, Zimmer, Jurish, & Sokirko, 2016) con ayuda de una base de datos relacional de tipo SQL, el corpus de referencia de acceso web en alemán DeReKo (Kupietz, Belica, Keibel, & Witt, 2010), cuyo corpus está indexado empleando el modelo IDS-Text Model (IDS, 2017). Por último, citar el sistema Xaira (Burnard & Todd, 2010) (y su antecesor SARA) empleado originalmente en

desarrollo del corpus BNC y que se basa en XML, aunque no posee un sistema de indexación complejo como tal y por tanto pertenece a la generación anterior.

- El procesamiento de corpus paralelos se hace más frecuente, por ejemplo, es admitido en CQPweb o Sketch Engine, pero requiere conocimientos técnicos avanzados para la tarea de alineación de textos y/o para su inclusión en la aplicación.

El acceso web añade dos nuevos problemas: las restricciones legales de distribución de materiales con *copyright* o derechos de autor, en este caso los textos que forman los corpus; y la necesidad de disponer de recursos económicos para la adquisición de licencias o el mantenimiento del servidor donde se aloja la aplicación web.

A pesar de las ventajas proporcionadas por las aplicaciones pertenecientes a la cuarta generación, existen razones que hacen que el uso de aplicaciones pertenecientes a la tercera generación en la actualidad aún sea muy frecuente, tal y como se detallará en el apartado 3.2.6 Situación actual.

La Tabla 1 resume las características básicas del software perteneciente a cada generación.

Generación	**Características básicas**	**Ejemplos**
1ª generación 1950-1980	- Uso de *mainframes.* - Acceso local. - Uso restringido y muy exclusivo. - Escasa usabilidad. - Muy baja replicabilidad. - *Concordancers* y programas de estadística básicos.	CLOC Discon Drexel Concordancer Program

2ª generación 1981-1996	- Aparición de IBM PC y crecimiento exponencial del uso de ordenadores. - Aumento de usabilidad. - Potencia reducida. - Aparecen primeros problemas codificando caracteres no latinos.	Longman Mini-Concordancer Kaye Concordancer
3ª generación 1996-actualidad	- Empleo de frameworks. - Aparición y empleo de Unicode (ver. UTF-8) y XML. - Similitud de funcionalidad entre distintos programas. - La replicabilidad es más factible. - Comienzan a desarrollarse frameworks compatibles con el procesamiento de corpus paralelos.	AntConc Nooj UAM Corpus Tool WordSmith
4ª generación 2005-actualidad	- Basados en web. - La creación de corpus requiere asistencia técnica. - Utilización de sistemas de indexación. - Multiplataforma. - Aumenta usabilidad. - Surgen problemas de *copyright* o derechos de autor.	Sketch Engine corpus.bye.edu CQPweb Corpuscle

Tabla 1 - Características básicas asociadas al software para el tratamiento de corpus lingüísticos de cada generación

3.2.6 Situación actual

3.2.6.1 Utilización de software

En 2016 coexiste el uso de aplicaciones y frameworks pertenecientes a la tercera y cuarta generación. A pesar de las ventajas proporcionadas por el software perteneciente a la cuarta generación, su uso dista de ser universal tal y como explica Anthony (Anthony, 2013a, pág. 154) y como se ha podido comprobar a partir de comunicaciones personales con lingüistas.[24]

Algunas de las razones que explican esta circunstancia son las siguientes:

(1) Los tipos de hipótesis de investigación basadas en corpus que se pueden estudiar con los frameworks de tercera y cuarta generación son prácticamente los mismos. Además, el desconocimiento de la incorporación de nuevas funcionalidades a los frameworks de cuarta generación provoca que se continúen usando los de la tercera. Por ejemplo, el soporte lexicográfico incluido en Sketch Engine no es conocido por algunos lingüistas.

(2) La potencia de los ordenadores personales para tratar corpus de tamaño medio, no hace estrictamente necesario el empleo de aplicaciones pertenecientes a la cuarta generación, basados en web y con potentes motores de búsqueda e indexación, pero con menor usabilidad. Muy frecuentemente son considerados únicamente como interfaces web para el acceso a corpus de referencia como COCA (Davies, 2008) o BNC (BNC Consortium, 2007). Además, la incorporación de sistemas de indexación básicos en nuevas versiones de frameworks pertenecientes a la tercera generación, como por ejemplo SQLite (Hipp, 2017) en AntConc (Anthony, 2013b, pág. 15), hace que las consultas sobre corpus de tamaño medio sean mucho más rápidas que en versiones anteriores.

[24] Sondeo realizado en base a las conversaciones mantenidas con personal investigador del grupo de investigación consolidado ACTRES y con investigadores lingüistas en congresos científicos.

(3) El alto coste de suscripción para el usuario y/o de administración y mantenimiento para el gestor del corpus de cuarta generación.

(4) La necesidad de licencias de software pertenecientes a la tercera generación, junto al conocimiento de uso prolonga su utilización.

(5) La escasa usabilidad de los sistemas de acceso y registro del software de cuarta generación (Anthony, 2013a, pág. 153).

(6) En clave de usabilidad, la reticencia a cambiar de aplicación está en proporción directa a tener que aprender cómo utilizar un nuevo software y a la dificultad de uso. Cuando es necesario emplear múltiples corpus, incorporar todos ellos en un nuevo software resulta complejo. Es frecuente que los investigadores utilicen aplicaciones pertenecientes a la tercera generación, locales y cuyo uso conocen a la perfección.

(7) Según (Anthony, 2013a, pág. 153), la escasa flexibilidad para cargar nuevos recursos y/o para compartir recursos ya existentes por parte de algunas aplicaciones, por ejemplo corpus.bye.edu (Davies, 2017a).

(8) En cuanto a la protección de datos, la necesidad de discreción, la posibilidad de violación de *copyright* o derechos de autor (Anthony, 2013a, pág. 153), o la incorporación de información sensible en un servidor externo lastran la utilización de software perteneciente a la cuarta generación.

(9) La escasa estandarización de funcionalidades no ayuda en el análisis y explotación de corpus que se presentan bajo aplicaciones web muy distintas. Por ejemplo: corpus.bye.edu y CORPES XXI (Real Academia Española, 2017a).

(10) La proliferación de herramientas y aplicaciones *ad hoc* que se incorporan como soporte externo de la aplicación para el tratamiento de corpus lingüísticos de tercera generación. Por ejemplo, un etiquetador cuya salida sea compatible directamente con AntConc.

Bajo un punto de vista meramente técnico, las ventajas computacionales y de accesibilidad que proporciona la cuarta generación son inigualables en comparación con la tercera generación:

- Mayor potencia y rapidez.
- Posibilidad de procesar corpus más grandes.
- Acceso web.
- Concurrencia de usuarios sin necesidad de distribuir el corpus entre los mismos.
- Espíritu multiplataforma ingénito.
- Mayor flexibilidad con la incorporación de etiquetados lingüísticos.

Sin embargo, la utilización de aplicaciones pertenecientes a la tercera generación resulta atrayente en corpus menores de 100 millones de palabras siempre y cuando se cumplan alguna de estas circunstancias:

- El framework para el tratamiento de corpus lingüísticos de cuarta generación alternativa impide compilar corpus sin asistencia técnica, sobre todo en el caso de corpus paralelos.
- Altos conocimientos de cómo emplear el software de tercera generación en contraste con la necesidad de entrenamiento con el software de cuarta generación.
- El corpus contiene material sensible.
- Ausencia de conexión a internet.
- No es estrictamente necesario el empleo de etiquetados lingüísticos.

3.2.6.2 Problemas y limitaciones

Los problemas y limitaciones más destacados del software para el tratamiento de corpus lingüísticos en la actualidad son:

- Similitud de funcionalidades entre las distintas aplicaciones disponibles.

- Creación de herramientas y aplicaciones *ad hoc* para tratar nuevas hipótesis de investigación.
- Dificultad de obtener replicabilidad utilizando otro software similar.
- Tipos de corpus permitidos: carencia de soporte para corpus bi-/multilingües paralelos y comparables.

La mayoría de los frameworks tienen funcionalidades muy similares: todos cuentan con un programa de concordancias más o menos complejo, soporte de etiquetado gramatical y extracción de estadísticas cuantitativas. No existe apenas ninguna diferencia significativa entre emplear uno u otro salvo por cuestiones de usabilidad.

Así que para intentar resolver nuevas hipótesis de investigación frecuentemente se recurre a herramientas y aplicaciones *ad hoc*, en muchas ocasiones externas incluso a cualquier entorno de programación o framework, y cuya integración se realiza a través de entradas y salidas muy concretas para su reincorporación en el flujograma de tratamiento de corpus lingüísticos utilizado. Y aquí reside uno de los mayores inconvenientes de esta estrategia, puesto que obliga a utilizar un flujo artesanal[25] formado por distintas herramientas/aplicaciones, que de forma individualizada proporcionan más o menos la funcionalidad de un framework para el tratamiento de corpus lingüísticos.

Otro problema derivado de la creación de herramientas o aplicaciones *ad hoc* es la creación de múltiples programas similares para resolver problemas concretos por parte de distintos grupos de investigación. Estas herramientas y aplicaciones difieren en pequeños detalles, como los formatos de entrada/salida, y suelen ser poco flexibles. La utilización de una herramienta o aplicación determinada puede marcar la diferencia en lo que respecta al empleo de software compatible[26] con su salida.

[25] En el sentido de la integración manual de las entradas y salidas del software.
[26] Sin modificaciones por parte de personal técnico especializado.

Se han recogido datos a partir de la experiencia de varios lingüistas[27] que ponen de manifiesto un comportamiento recurrente: el usuario trata de construir un tipo de flujo artesanal mediante el empleo de las salidas de los distintos programas utilizados, intentado adaptar dichas salidas a través de modificaciones manuales que consumen mucho tiempo, tolerando un cierto grado de error o llevando a cabo tareas que son realizables por medio de programas informáticos sin excesiva dificultad.

Un ejemplo de flujo de tratamiento de corpus artesanal podría ser el mostrado en la Figura 3: (1) recolección manual de corpus, (2) limpieza de texto inservible en los distintos documentos del corpus (manual o no), (3) codificación a UTF-8, (4) conversión a XML, (5) empleo de programa para buscar concordancias y extracción de características y, (6) etiquetado gramatical manual o extracción NER (*Named - Entity Recognition*)[28] de las secuencias más interesantes de acuerdo con la hipótesis de investigación planteada.

[27] Grupo de investigación consolidado ACTRES (24 usuarios).

[28] De aquí en adelante se utilizará el término NER en lugar de *Named -Entity Recognition.*

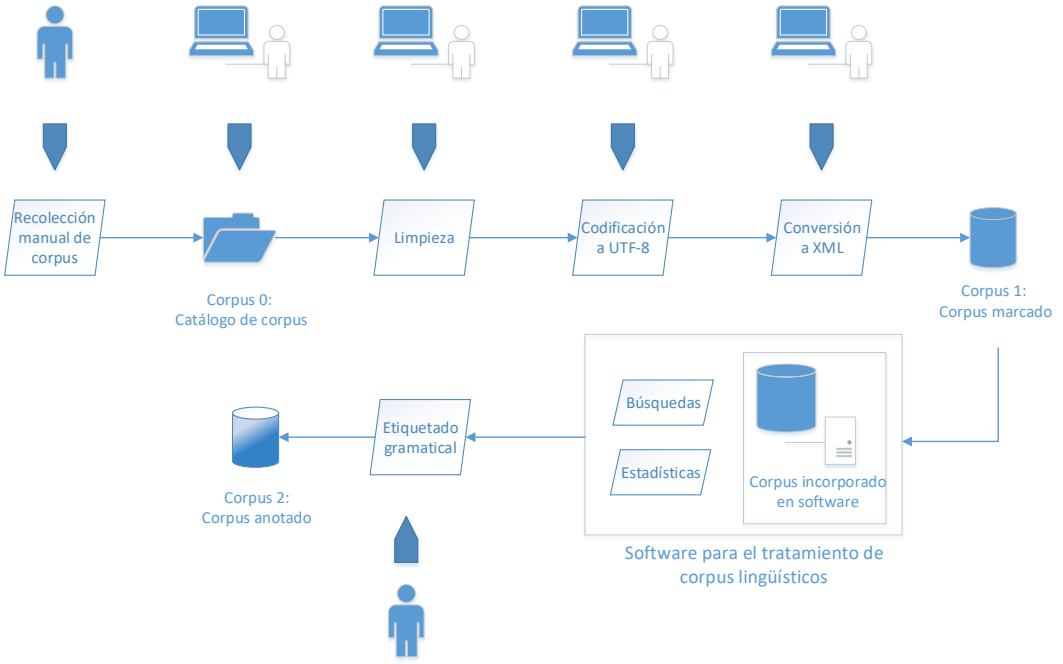


Figura 3 - Ejemplo de flujo artesanal para intentar responder a una hipótesis de investigación

Esta selección de los componentes del flujo artesanal de tratamiento de corpus adolece de graves problemas, incluida la atomización de recursos descrita por (Dale, 2010, pág. 4) y (Nadkarni, Ohno-Machado, & Chapman, 2011, pág. 548). Además, a causa del empleo de herramientas antiguas e incluso en ocasiones obsoletas, se pueden generar los siguientes problemas adicionales: los formatos de entrada/salida son muy restrictivos; falta de potencia al tratarse de herramientas no optimizadas para los nuevos sistemas operativos y procesadores y que, por tanto, no aprovechan los recursos hardware actuales al máximo; algunas de las operaciones han de ser relegadas a especialistas informáticos; necesidad de empleo de programas para ejecutar por consola, con interfaces muy poco atractivas, o que requieren un entorno operativo concreto que resulta difícil de emular para usuarios no especialistas en informática; y por último, el uso de software anticuado puede aumentar el número de tareas repetitivas, incrementando la fatiga y el número de errores cometidos por parte del investigador lingüista.

Todos estos problemas provocan que una misma hipótesis de investigación pueda resolverse empleando distinto software, framework o librería, cada una con sus especificaciones técnicas y de formato. Por lo que conseguir la replicabilidad exacta de un estudio científico basado en corpus es difícil de alcanzar a no ser que se empleen exactamente los mismos programas (ver 3.1.5 Replicabilidad para más información).

Por último y probablemente el mayor problema de la situación actual, en especial para los lingüistas especializados en contraste lingüístico, variación lingüística y traducción, es la ausencia de software especializado en tratamiento de corpus paralelos que no requiera de la intervención de un especialista informático en alguna fase del proceso de compilación del corpus, así como software específico que permita el procesamiento de corpus comparables.

Estamos ante dos situaciones diferenciadas:

- **Corpus paralelos**: si bien es cierto que existen frameworks para el tratamiento de corpus lingüísticos capaces de procesarlos, tanto en la tercera generación con AntPConc o ParaConc, como en la cuarta generación con Sketch Engine o CQPweb, la necesidad de conocimientos de programación o técnicos en la fase de alineado y posterior incorporación al gestor de corpus resulta indispensable. En el framework presentado en esta tesis doctoral se superarán esas barreras que los usuarios lingüistas tenían a la hora de construir sus corpus paralelos a través de una alineación no supervisada y su posterior incorporación de forma automática al framework.
- **Corpus comparables**: desde el punto de vista técnico el problema es mucho menor, no es necesaria ninguna alineación y el estudio lingüístico es realizable a través de sendas consultas en corpus considerados comparables entre sí. Sin embargo, no existe un framework funcional para el tratamiento de corpus lingüísticos completo con la funcionalidad específica de permitir

procesar corpus comparables en una o varias lenguas[29]. En este caso, la solución planteada en el framework presentado es el simple marcado de los corpus considerados comparables para así poder realizar consultas múltiples de forma simultánea sobre dichos corpus y presentar la información de forma conjunta.

En resumen, muchos de los problemas actuales residen en la limitación que el software empleado impone al lingüista a la hora de establecer sus hipótesis de investigación que puede resolver. A causa de esta limitación se emplean aplicaciones y herramientas *ad hoc* muy difíciles de integrar sin conocimientos informáticos, que se integran en el flujograma habitual para el tratamiento de corpus lingüísticos. Estas dificultades ocasionan problemas de replicabilidad, usabilidad e interconexión de datos entre los distintos componentes. Por último, la ausencia de software especializado totalmente exento de intervención técnica para permitir el tratamiento de corpus bi-/multilingües paralelos o de múltiples consultas y comparación manual en los corpus comparables, resulta llamativa.

3.2.6.3 Últimas tendencias en el desarrollo de frameworks para el tratamiento de corpus lingüísticos

Las últimas novedades relacionadas con el desarrollo de frameworks para el tratamiento de corpus lingüísticos se centran en tres aspectos: mayor velocidad y rapidez en las consultas, incorporación de estadísticas más complejas y tratamiento de corpus formados por datos procedentes de redes sociales.

Aumentar la potencia y rapidez de los programas de concordancias posibilita el análisis de corpus en menor tiempo. Esta característica no solo afecta a los *monitor corpora* sino que también influyen en la rapidez de las consultas de cualquier tipo

[29] Corpógrafo V4 (Maia & Matos, 2008) supuestamente permitió el tratamiento de corpus comparables multilingües, pero no empleando una sola lengua. El framework no se encuentra operativo en la actualidad. Ver 3.3.1.8 Corpógrafo para más información.

de corpus. Por ejemplo, mencionar la gran rapidez de las consultas en los frameworks para el tratamiento de corpus lingüísticos Corpuscle (Meurer, 2012) o corpus.bye.edu (Davies, 2017b), que en su documentación incluyen diversos estudios comparativos con otros sistemas de indexación como CQP (Evert, 2009).

Otro aspecto que centra el esfuerzo de los investigadores es la incorporación de estadísticas junto a visualizaciones gráficas en las distintas aplicaciones. Esta funcionalidad se realiza por medio del uso paquetes estadísticos basados en R (R Core Team, 2017), como recogen (Hardie, 2012, pág. 405) o (Meurer, 2012, pág. 50). Un ejemplo de estas estadísticas son las colocaciones en contexto (Brezina, McEnery, & Wattam, 2015).

Por último, la utilización de datos procedentes de las redes sociales se está convirtiendo en una tendencia generalizada, como por ejemplo los artículos de investigación (Bamman, Eisenstein, & Schnoebelen, 2014) o (Refaee & Rieser, 2014). Los datos son recolectados empleando librerías propias de las redes sociales, y suelen presentar una estructura tabular: una entrada por línea, encabezados en la primera línea, valores de cada entrada separados por tabulaciones. Para el tratamiento de estos datos en un corpus se suelen emplear aplicaciones *ad hoc* o incluso alternativas de propósito general como FireAnt (Anthony & Hardaker, 2016).

3.3 Análisis de software disponible[30]

A continuación, se analizará el software para el tratamiento de corpus lingüísticos disponible en la actualidad. De acuerdo con la definición proporcionada al

[30] Fuera de este análisis quedan el conjunto de herramientas y/o aplicaciones desarrolladas por grupos de investigación, que son empleadas conjuntamente de forma frecuente para el tratamiento de sus corpus lingüísticos, pero que son creadas como software independiente. Por ejemplo: mate-tools (Bohnet, 2010).

comienzo de la sección 2.3 Software especializado para el tratamiento de corpus lingüísticos:

> *"Es un programa de ordenador que ha sido creado exclusivamente para llevar a cabo procedimientos técnicos relacionados con los corpus lingüísticos."*

Este software se divide en tres categorías: (1) frameworks para el tratamiento de corpus lingüísticos, (2) toolkits, *suites* o similares y (3) aplicaciones y herramientas que participan en algún proceso relacionado con el tratamiento de corpus lingüísticos.

Tal y como también se mencionó en el apartado 2.3.2 Aplicaciones y herramientas relacionadas con el tratamiento de corpus lingüísticos, el análisis individual de todas las opciones software disponibles para cada operación necesaria en el tratamiento de corpus lingüísticos es irrealizable, y además se aleja del objetivo principal de esta tesis doctoral, que es la creación de un framework integral. Por estos motivos el presente análisis de software se centra en las dos categorías restantes: (1) frameworks y (2) toolkits, *suites* o similares, que además incorporan intrínsecamente software o soluciones software pertenecientes a la categoría descartada.

- Los frameworks para el tratamiento de corpus lingüísticos se definieron en la sección 2.3 Software especializado para el tratamiento de corpus lingüísticos, como

  > "*una aplicación informática que integra las distintas funcionalidades que permite, sin salirse del* framework*, llevar a cabo al menos las tareas comunes del tratamiento de corpus lingüísticos por parte de un usuario sin conocimientos de programación*".

  Por lo que:

  (1) Están orientados para su ejecución por parte de usuarios no informáticos.

(2) Poseen un interfaz gráfico que sirve como medio de interacción.

(3) No es estrictamente necesario disponer de conocimientos de programación para su uso básico.[31]

(4) Se trata de un entorno cerrado que no obliga al usuario a emplear software externo para hacer un uso básico de la aplicación.

(5) Con la proliferación de numerosos corpus de acceso web pertenecientes a la 4ª generación, resulta complejo discernir cuál es un nuevo framework, o cuál es un simple *front-end* (capa de presentación) de algún sistema de indexación (por ejemplo, CQP). Por este motivo, sólo se analizarán aquellos que incorporen mejoras funcionales notables y no sean simples plataformas web de consulta de corpus lingüísticos.

(6) El software es aún funcional.

- Los toolkits, *suites* o similares fueron también definidos en la sección 2.3 Software especializado para el tratamiento de corpus lingüísticos, y describen a

  "*un conjunto de librerías o programas independientes que proporcionan al programador un gran número de recursos para resolver problemas o crear* software *en una disciplina concreta*".

  Por lo que:

  (1) Están orientadas para su ejecución por parte de usuarios informáticos.

  (2) No es estrictamente necesario que posean un interfaz gráfico.

  (3) Es altamente recomendable disponer de conocimientos de programación para su utilización.

[31] Algunas tareas como la indexación, alineación o etiquetado, pueden requerir la participación de un especialista informático, en ese caso se detallará y se considerará un problema adicional de usabilidad.

Todas estas características permiten filtrar una gran cantidad de software que no resulta provechoso considerando el flujograma presentado en la Figura 2. Se descarta, por tanto, el análisis de herramientas y/o aplicaciones que realicen tareas individuales específicas como: programas de concordancias o búsqueda, etiquetadores, programas de estadística, parsers, etc. Si bien es cierto que un framework para el tratamiento de corpus lingüísticos posee la mayor parte de ellas como elementos funcionales intrínsecos de su estructura.

3.3.1 Frameworks para el tratamiento de corpus lingüísticos

El análisis de los frameworks para el tratamiento de corpus lingüísticos disponibles resulta crucial porque es el tipo de software que se va a crear en esta tesis doctoral. Cada framework se analizará teniendo en cuenta los problemas y limitaciones descritos en la sección 3.1 Problemas y limitaciones, que sirvieron de base para caracterizar a las distintas generaciones de (McEnery & Hardie, 2012, págs. 37-48), junto a los inconvenientes detallados en el apartado 3.2.6 Situación actual.

Por todo ello, las características analizadas han sido las siguientes:

- **Compilación de corpus**: Si el framework permite al usuario crear sus propios corpus lingüísticos o, por el contrario, sólo permite utilizar los que se encuentra cargados en el sistema.
- **Necesidad de asistencia técnica especializada:** Si son necesarios o no, conocimientos de programación o conocimientos técnicos avanzados. Lo que repercute en la necesidad de disponer de un especialista informático en algunas de las fases de la construcción del corpus lingüístico.
- **Tratamiento de corpus en múltiples lenguas**: Si es posible utilizar la aplicación con corpus en idiomas distintos del inglés.
- **Tratamiento de corpus paralelos**: Si es posible emplear el software para crear corpus paralelos multilingües.

- **Alineador integrado**: Si el software posee un alineador supervisado o no, para llevar a cabo la alineación.
- **Tratamiento de corpus comparables**: Si es posible emplear el software para crear corpus comparables (en un idioma o en varios de acuerdo con la definición proporcionada en la página 9).
- **Tecnología de indexación y búsqueda**: Especificación de la tecnología empleada para la indexación y la búsqueda en el corpus lingüístico.
- **Estadísticas**: Si la aplicación posibilita obtener estadísticas del corpus. Más concretamente se analizarán las características más utilizadas en los estudios sobre corpus lingüísticos, que son:
    - Lista de frecuencias o *frequency list* en inglés.
    - Colocaciones o *collocations* en inglés.
    - Lista de palabras clave o *keyword list* en inglés.
    - n-gramas o *n-grams* en inglés.
- **Interfaz gráfico**: Si el software posee un interfaz gráfico, y si éste es moderno y funcional.
- **Generación:** Generación a la que pertenece de acuerdo con (McEnery & Hardie, 2012, págs. 37-48).
- **Alojada en web**: Si la aplicación se encuentra alojada en la red o por el contrario ha de ser instalada en un equipo personal.
- **Soporte de etiquetados:** Si soporta corpus con anotaciones lingüísticas.
- **Etiquetadores integrados**: Si la aplicación posee etiquetadores integrados. Más concretamente se analizarán:
    - Gramatical.
    - Semántico.
    - Retórico.
- **Disponibilidad**: Se refiere a cómo es posible utilizar el framework, a través de acceso web o mediante instalación en servidor o equipo propio.

- Soporte externo: solo es posible utilizarlo en la infraestructura del creador del mismo.
- Descargable: para instalación en equipo o servidor propio.

Con el fin de obtener una visión global de las tareas realizadas por cada software analizado, se visualizará sus funcionalidades de acuerdo con un flujograma de tratamiento de corpus lingüísticos habitual similar al presentado en la Figura 2.

3.3.1.1 WordSmith

WordSmith (Scott, 2012) (Scott, 1996) es uno de los programas más utilizados por los lingüistas para compilar y analizar sus propios corpus de tamaño medio o reducido. Se trata de un software de pago que goza de una gran popularidad por ser uno de los primeros programas de concordancias de la tercera generación en aparecer, además de su facilidad de uso y su robustez. Estos hechos junto a las razones expuestas en el subapartado 3.2.6.1 Utilización de software, hace que su utilización continúe siendo una constante y goce de una gran reputación, a pesar de existir alternativas gratuitas igualmente válidas como AntConc (Anthony, 2014a). No existe demasiada información sobre su arquitectura, desarrollado por Mike Scott se encuentra bajo el amparo de *Lexical Analysis* Software *Ltd*.

WordSmith no requiere asistencia técnica y maneja corpus en múltiples lenguas. No procesa corpus paralelos ni comparables y las búsquedas se basan en un sistema propio no revelado en detalle. Aunque soporta etiquetados, no proporciona ninguna funcionalidad integrada para llevarlos a cabo. Posee las estadísticas básicas (lista de palabras, lista de palabras clave, lista de frecuencias, colocaciones) así como n-gramas (denominados *multi-word clusters*) e incluso funciones avanzadas como concgram. Su disponibilidad es a través de descarga e instalación. Como característica adicional, WordSmith posee una pequeña API para integrarlo en aplicaciones externas.

Uno de los problemas más importantes de WordSmith es que carece de etiquetadores lingüísticos integrados. Además, no posee funcionalidad para procesar corpus paralelos y comparables. La ausencia de acceso web, como característica intrínseca a su pertenencia a la tercera generación, y la dificultad para tratar con corpus de gran tamaño (más de 100 millones de palabras) son otros de sus principales inconvenientes. Para tratar de subsanar todos estos inconvenientes, WordSmith soporta el etiquetado realizado mediante un programa externo, aunque es necesario definir el formato empleado en la propia interfaz de la aplicación. También contiene una alineador de párrafos integrado que permite alinear textos.

Su principal fortaleza es su reputación y facilidad de uso, que le convierten en la aplicación de referencia para el tratamiento de corpus lingüísticos de muchas investigaciones, por ejemplo (Kenny, 2014), (Ivaska, 2014) o (Gilquin, 2015).

WordSmith	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	No
Alineador integrado	Sí*
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio (no definido)
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	Sí
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí*

Etiquetadores integrados	
Gramatical	No
Semántico	No
Retórico	No
Disponibilidad	Descargable

Tabla 2 - Resumen de características de WordSmith

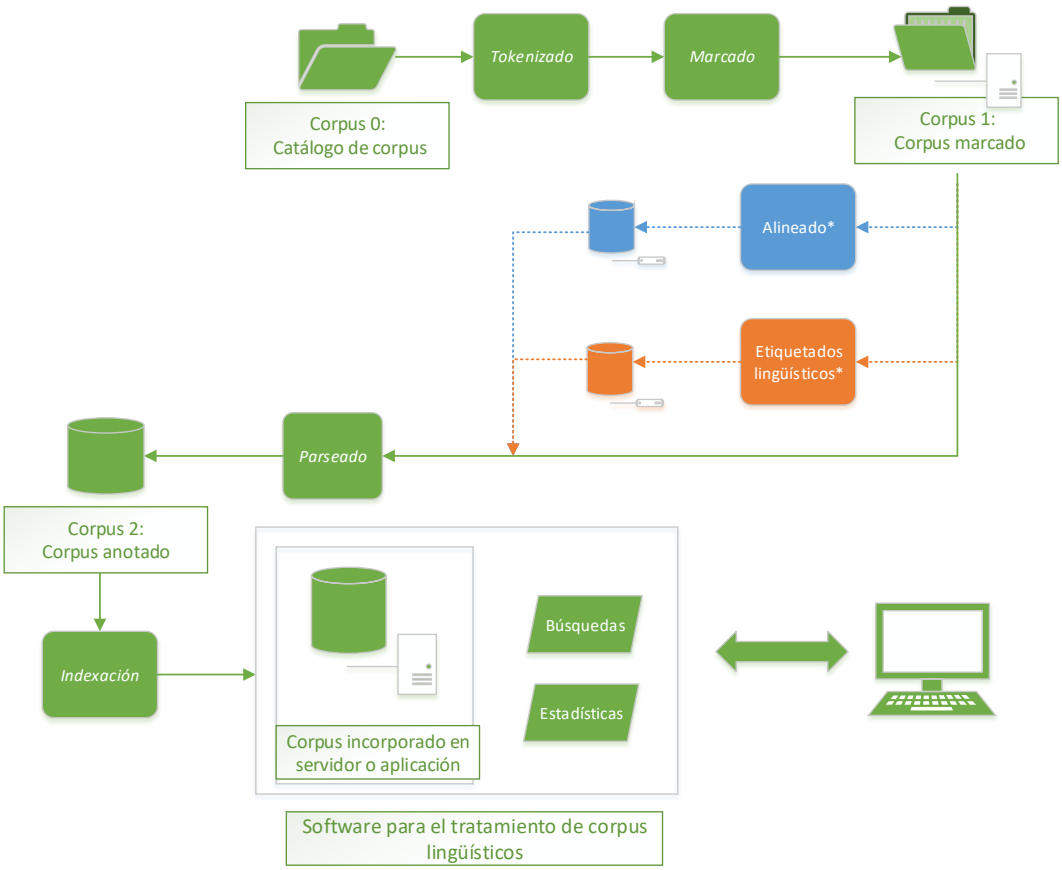


Figura 4 - Funciones realizadas por WordSmith de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado, naranja – realizado con asistencia/externamente)

3.3.1.2 AntConc y AntPConc

AntConc (Anthony, 2014a) (Anthony, 2005) es un software para el tratamiento de corpus lingüísticos desarrollado por Laurence Anthony perteneciente a W*aseda University* – Universidad de Waseda, en Tokio.

Las características del software AntConc son muy similares a las descritas en WordSmith. AntConc es un software gratuito perteneciente a la tercera generación, que permite compilar corpus en múltiples lenguas. No permite tratar corpus paralelos ni comparables. Está desarrollado en Python (Python Software Foundation, 2017b) y Qt (The Qt Company, 2017), y su sistema de búsqueda se basa en SQLite (Hipp, 2017). Al igual que WordSmith, carece de etiquetadores lingüísticos integrados, y aunque los soporta, es necesario definir su formato en el interfaz. Proporciona todas las estadísticas analizadas (lista de palabras, lista de palabras clave, lista de frecuencias, colocaciones y n-gramas). Su disponibilidad, al igual que el resto del software desarrollado por Anthony, es a través de un ejecutable.

Posee una versión para corpus paralelos, AntPConc (Anthony, 2014b), pero su funcionalidad es mucho menor y se reduce a un simple programa de concordancias. Además, el hecho de emplear dos programas distintos para procesar diferentes tipos de corpus resta usabilidad.

AntConc tiene las mismas debilidades y fortalezas que las detalladas en WordSmith, aunque cabe señalar que es gratuito y que tiene una versión simplificada que procesa corpus paralelos. No goza de tanta popularidad como WordSmith.

AntConc y AntPConc	
Necesidad de asistencia técnica	No

Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	Sí*
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio (Python y SQLite)
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	Sí
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí*
Etiquetadores integrados	
Gramatical	No
Semántico	No
Retórico	No
Disponibilidad	Descargable

Tabla 3 - Resumen de características de AntConc y AntPConc

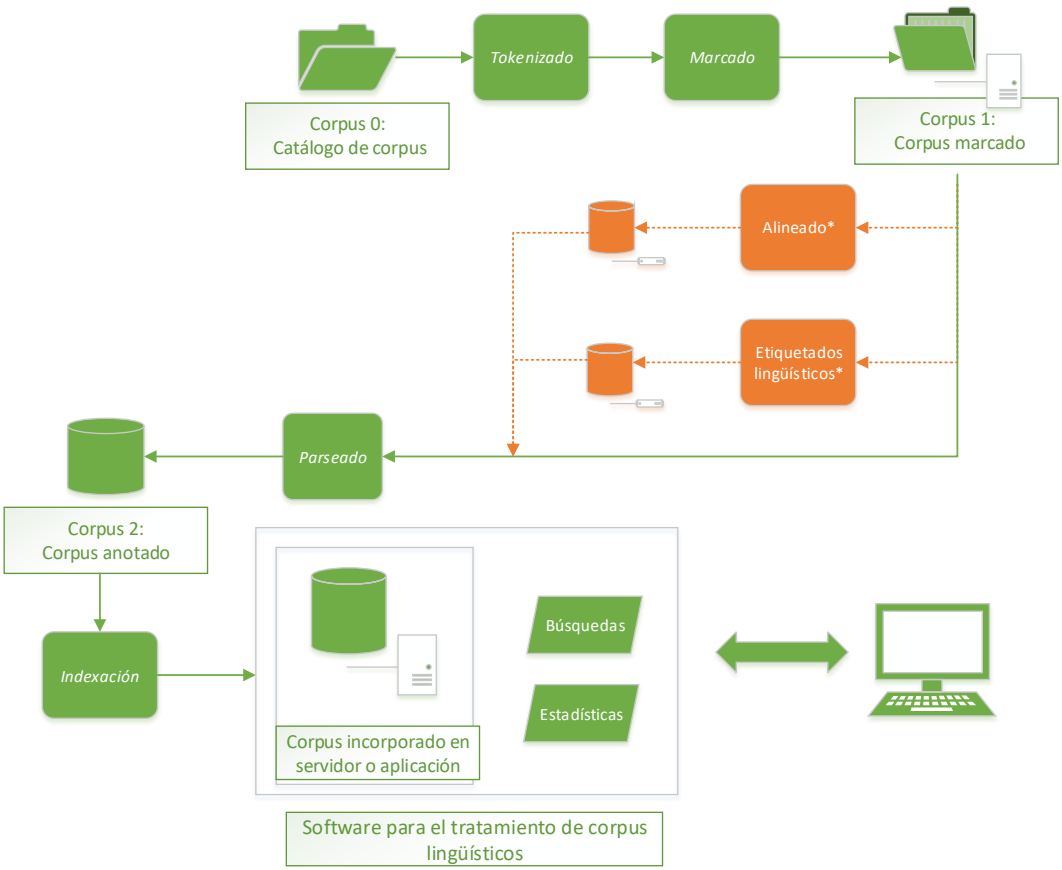


Figura 5 - Funciones realizadas por AntConc y AntPConc de acuerdo con el flujograma presentado (verde – realizado, naranja – realizado con asistencia/externamente)

3.3.1.3 MonoConc y ParaConc

MonoConc (Barlow, 2003) es un software para el tratamiento de corpus lingüísticos desarrollado por Michael Barlow. Fue uno de los primeros frameworks para el tratamiento de corpus lingüísticos de la tercera generación junto a WordSmith. Sin embargo, al contrario que este, MonoConc no tiene actualizaciones desde 2003.

Se trata de un software de pago que posee las mismas características que AntConc y WordSmith en cuanto a:

- La posibilidad de compilar corpus en múltiples lenguas.
- No procesar corpus paralelos o comparables.

- La ausencia de etiquetadores lingüísticos integrados, aunque soporte textos etiquetados con programas externos a través de la definición en su interfaz.
- La posibilidad de extracción de estadísticas como: lista de palabras, lista de palabras clave, lista de frecuencias, colocaciones y n-gramas.

No existe información sobre el tipo de indexación al ser un software comercial. Su disponibilidad es a través de descarga e instalación en el equipo informático que se desee.

Al igual que AntConc con AntPConc, posee una versión para procesar corpus paralelos denominada ParaConc (Barlow, 1995), que añade funcionalidades extra muy interesantes: como un alineador integrado bajo supervisión del usuario que permite el procesamiento de corpus paralelos sin apenas intervención técnica. Sin embargo, posee las mismas limitaciones de ausencia de etiquetado gramatical y acceso web descritas con anterioridad a lo largo de análisis del distinto software perteneciente a la tercera generación. Hay que añadir que la usabilidad se ve reducida por el hecho de emplear dos programas distintos para procesar diferentes tipos de corpus.

En resumen, MonoConc y su versión paralela ParaConc poseen las mismas debilidades y fortalezas que las detalladas en AntConc y AntPConc, añadiendo que ParaConc posee un alineador integrado.

MonoConc y ParaConc	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	Sí

Alineador integrado	Sí
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	No
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí*
Etiquetadores integrados	
Gramatical	No
Semántico	No
Retórico	No
Disponibilidad	Descargable

Tabla 4 - Resumen de características de MonoConc y ParaConc

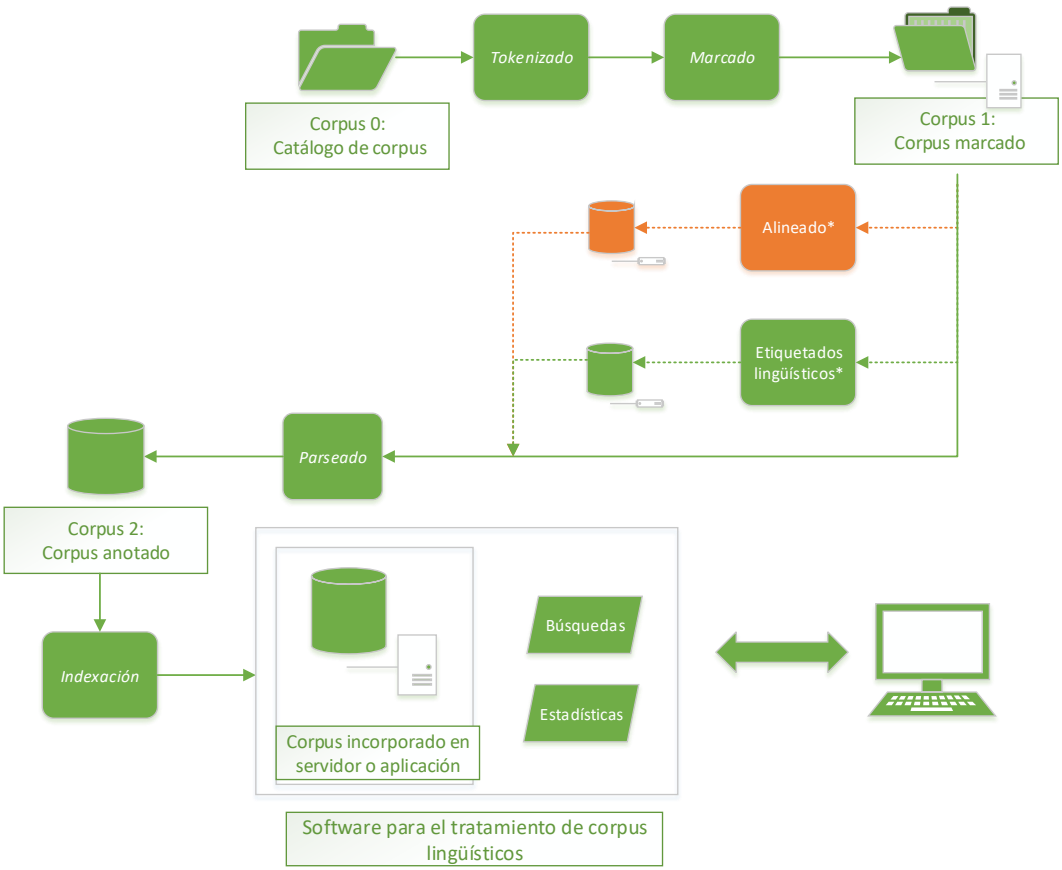


Figura 6 - Funciones realizadas por MonoConc y ParaConc de acuerdo con el flujograma presentado (verde – realizado, naranja – realizado con asistencia/externamente)

3.3.1.4 Wmatrix

Wmatrix (Rayson, 2009) es un software de gestión de corpus creado por Paul Rayson perteneciente a *Lancaster University* – Universidad de Lancaster. Basado en su tesis doctoral (Rayson, 2003), Wmatrix ha experimentado múltiples actualizaciones.

Se trata de un software de cuarta generación, que permite a los usuarios compilar sus propios corpus en inglés y etiquetarlos con dos tipos de etiquetado: gramatical basado en CLAWS (Garside, 1987) y semántico a través de USAS (Piao, Bianchi, Dayrell, D'Egidio, & Rayson, 2015). No soporta paralelismo, ni corpus comparables. Es una plataforma exclusiva para la comunidad universitaria de

Lancaster o bajo pago por suscripción. Mediante acceso web permite al usuario la extracción de las estadísticas más comunes: listas de frecuencias, concordancias, colocaciones, lista de palabras clave, lista de palabras, n-gramas, así como un programa de concordancias avanzado para ejecutar consultas y obtener estadísticas relacionadas con los distintos tipos de etiquetados.

El punto fuerte de Wmatrix es el soporte de etiquetado semántico y la ausencia de asistencia técnica en todas las fases de compilación de corpus. Sin embargo, no soporta corpus en idiomas distintos al inglés, ni corpus comparables o paralelos.

Wmatrix	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	No
Tratamiento de corpus paralelos	No
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio (Perl)
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	Sí
Interfaz gráfico	Sí

Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí (CLAWS)
Semántico	Sí (USAS)
Retórico	No
Disponibilidad	Soporte propio

Tabla 5 - Resumen de características de Wmatrix

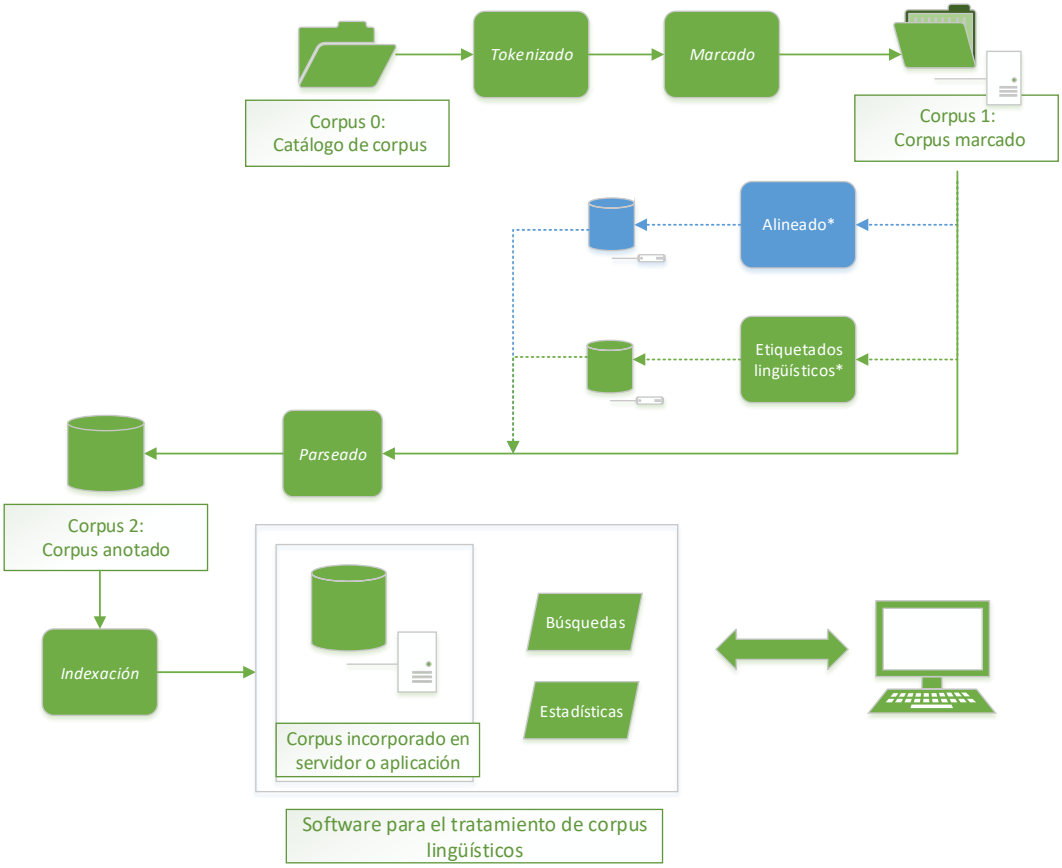


Figura 7 - Funciones realizadas por Wmatrix de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.5 Sketch Engine

Sketch Engine (Kilgarriff, et al., 2014) es uno de los frameworks para el tratamiento de corpus lingüísticos monolingües y multilingües más avanzado desde el punto de vista de los requisitos analizados. Durante el desarrollo de la tesis se fueron incorporando novedades relacionadas con lexicografía y soporte de corpus paralelos que lo hicieron aún más completo. Aun así, no logra las cotas de usabilidad, independencia de asistencia técnica en todo el proceso y soporte de múltiples capas de etiquetados lingüístico que se pretenden alcanzar en la tesis aquí presentada.

Sketch Engine está creado por la empresa *Lexical Computing Ltd.*, con Adam Kilgarriff y Pavel Rychlý como desarrolladores. Está basado parcialmente en la tesis doctoral de Pavel Rychlý (Rychlý, 2000) y el desarrollo posterior de sus distintos componentes (Rychlý, 2007). Inicialmente su desarrollo empleó los lenguajes de programación C++ (Stroustrup, 1995) y Python.

Se trata de un software de pago perteneciente a la cuarta generación. Posibilita la compilación de corpus por parte del usuario e incluso posibilita recopilarlos a través de la web por medio de WebBootCat (Baroni, Kilgarriff, Pomikalek, & Rychlý, 2006). Tiene un marcado espíritu multilingüe y permite el procesado de corpus paralelos multilingües, no sólo bilingües, aunque no integra un alineador. No soporta corpus comparables. Admite una sola capa de etiquetado, gramatical, e incluye la posibilidad de usar diferentes etiquetadores integrados como Treetagger (Schmid, 1995) y FreeLing POS tagger (Padró & Stanilovsky, 2012) entre otros. El sistema de búsqueda, posee un potente programa de concordancias basado en CQP (Evert, 2009) denominado *Manatee*, un interfaz gráfico para búsquedas sobre corpus llamada *Bonito*. Es capaz de extraer multitud de estadísticas: lista de palabras, lista de palabras clave, lista de frecuencias, colocaciones, n-gramas, además de estadísticas automáticas derivadas del comportamiento gramatical y de los distintos tipos de colocaciones de cada palabra. Por último, se encuentra alojado

en una infraestructura perteneciente a *Lexical Computing Ltd*, que resulta muy eficiente.

Sketch Engine posee una versión libre y gratuita del programa denominada NoSketch Engine (Rychlý, 2007), con un interfaz mucho más simple, estadísticas básicas: listas de frecuencias, colocaciones, listado de palabras. Carece de estadísticas relacionadas con la lexicografía, e implica que el proceso de subida de textos y etiquetados requiera de conocimientos técnicos al carecer de interfaz. Posee el mismo problema respecto al tratamiento de corpus paralelos y comparables que SketchEngine.

En resumen, Sketch Engine es una de las mejores opciones para tratar corpus monolingües con una única capa de etiquetado lingüístico en casi cualquier idioma. La necesidad de concomimientos técnicos para llevar a cabo el procesamiento de corpus paralelos, la ausencia de soporte de corpus comparables, la falta de otros tipos de etiquetados diferentes al gramatical y que sea de pago son sus mayores debilidades.

Sketch Engine	
Necesidad de asistencia técnica	Sí*
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	Sí
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	CQP

Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	Sí
Interfaz gráfico	Sí
Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí (Treetagger, FreeLing entre otros)
Semántico	No
Retórico	No
Disponibilidad	Soporte propio

Tabla 6 - Resumen de características de Sketch Engine

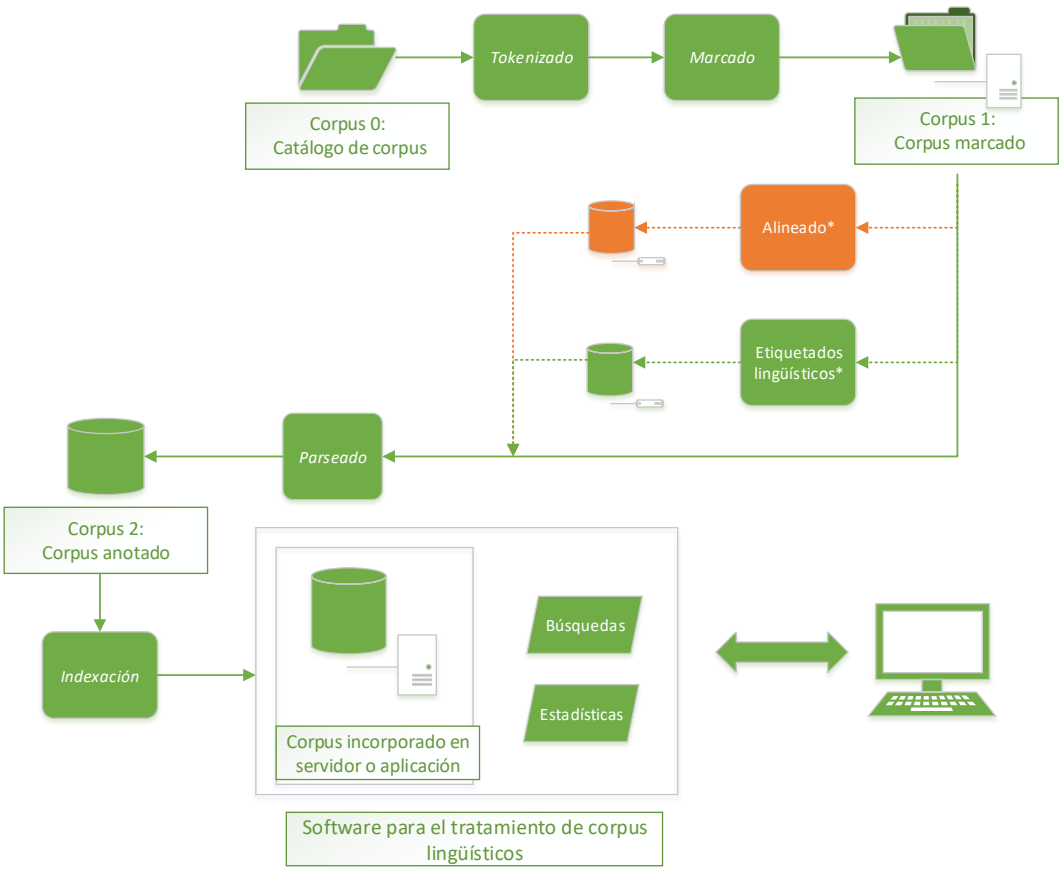


Figura 8 - Funciones realizadas por Sketch Engine de acuerdo con el flujograma presentado (verde – realizado, naranja – realizado con asistencia/externamente)

3.3.1.6 IntelliText

IntelliText (Wilson, Hartley, Sharoff, & Stephenson, 2010) fue creado con el apoyo de *University of Leeds* – Universidad de Leeds, siendo Anthony Hartley el desarrollador principal del proyecto, con el apoyo de James Wilson, Serge Sharoff y Paul Stephenson.

IntelliText es un software de cuarta generación que no requiere asistencia técnica en ningún momento del proceso. Permite al usuario compilar sus propios corpus en múltiples idiomas y etiquetados gramaticalmente por medio de la acción de Treetagger para lenguas europeas y china, así como MeCab (Kudo, 2005) para el japonés. Al igual que la mayoría de aplicaciones analizadas, el sistema de

indexación se basa en CQP. El software es compatible con el procesamiento de corpus paralelos, ya que existen varios cargados por defecto en el sistema, pero está opción no está disponible para los usuarios. Incluso en la documentación oficial (Centre for Translation Studies, 2016) no hay información de cómo realizarlo. Es de suponer, que el proceso es complejo y aunque tiene soporte, éste no está disponible para los usuarios. Por lo tanto, se intuye que los corpus paralelos existentes han sido tratados por técnicos especializados y están disponibles como una funcionalidad extra. Por otra parte, tampoco soporta el procesamiento de corpus comparables. Por último, muestra todas las estadísticas analizadas a excepción de n-gramas. La disponibilidad del software es a través del soporte proporcionado por *University of Leeds* – Universidad de Leeds, al menos en la actualidad, porque anteriormente existía una versión descargable.

El software posee otra serie de características extras que merecen ser mencionadas. Permite realizar dos tipos de exámenes relacionados con el análisis multi-variable de Biber (Biber, 1991), mostrando los resultados gráficamente gracias a la utilización del lenguaje de programación R (R Core Team, 2017). Los dos tipos de análisis son:

(1) Análisis de la dimensión 1 a la 4.
(2) Análisis de 19 características lingüísticas definidas por Biber, por ejemplo: terceras personas, verbos pasivos, entre otros.

En resumen, los puntos fuertes de IntelliText son el multilingüismo, la no exigencia de asistencia técnica, las funciones extras relacionadas con el análisis multi-variable de Biber, y su robusto interfaz gráfico, que sigue las pautas descritas en (Sharoff, 2006).

Como puntos débiles se distinguen la ausencia de paralelismo y soporte de corpus comparables y la inexistencia de etiquetados lingüísticos diferentes al gramatical.

IntelliText	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	No*
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	CQP
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	No
Interfaz gráfico	Sí
Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí (Treetagger y MeCab)
Semántico	No

Retórico	No
Disponibilidad	Soporte propio

Tabla 7 - Resumen de características de IntelliText

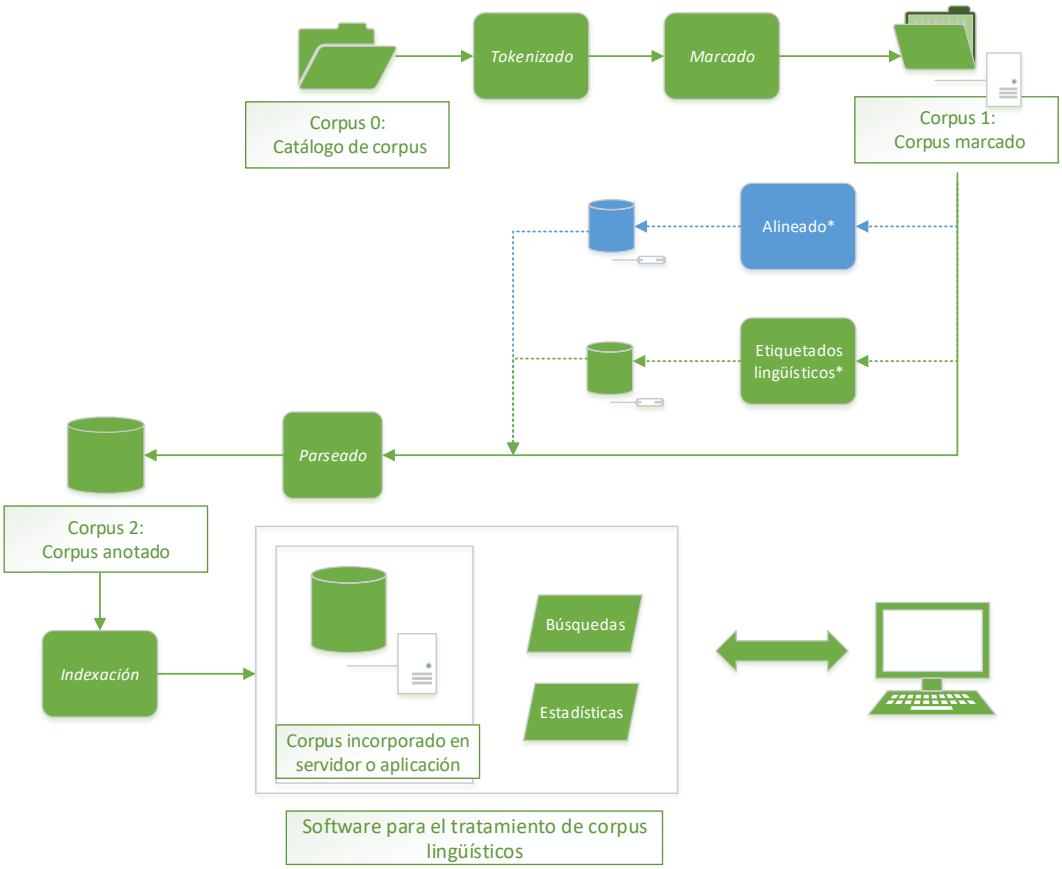


Figura 9 - Funciones realizadas por IntelliText de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.7 CQPweb

CQPweb (Hardie, 2012) es un software para el tratamiento de corpus lingüísticos perteneciente a la cuarta generación desarrollado por Andrew Hardie. Se trata de un clon de BNCweb (Hoffmann, Evert, Smith, Lee, & Berglund-Prytz, 2008), pero que

al contrario de éste, proporciona acceso a múltiples corpus y no sólo a BNC (BNC Consortium, 2007).

CQPweb posibilita la creación de corpus en múltiples idiomas. Permite la integración de varias capas de etiquetados lingüísticos gracias al uso del motor de indexación y búsqueda CQP. Sin embargo, no tiene integrado ningún etiquetador lingüístico, han de usarse programas externos. El estricto formato para la incorporación de textos en la aplicación, ya sea con etiquetado lingüístico o no, obliga a la asistencia de un técnico especializado o de conocimientos de programación. El 2 de agosto de 2016 se lanzó una nueva versión que permite la integración de corpus paralelos, pero al igual que en el caso de los etiquetados lingüísticos, requiere de la acción de un alineador externo y su posterior parseado para incorporación a la aplicación. Hasta el momento, no es posible la realización de consultas en corpus comparables, más por falta de interés que por las dificultades que conlleva. Su motor de búsqueda e indexación se basa en una solución mixta que emplea tecnología SQL y la potencia de CQP, lo que proporciona una gran velocidad a las consultas además de proporcionar funcionalidades relacionadas con el historial de ejecución de consultas y otras características. El formato de búsquedas emplea el formato CEQL (Hoffmann, Evert, Smith, Lee, & Berglund-Prytz, 2008, Capítulo. 6), una versión simplificada de la notación utilizada en CQP (Evert, 2009) que facilita la elaboración de consultas complejas por parte de los lingüistas. Las estadísticas que soporta CQPweb incluyen lista de palabras, lista de palabras clave, lista de frecuencias, colocaciones entre otras, aunque por el momento no muestra n-gramas. CQPweb posibilita su uso a través del soporte web proporcionado por *Lancaster University* – Universidad de Lancaster, permitiendo consultar y crear corpus dependiendo de los permisos de la cuenta de acceso- También permite descargar el código fuente e instalarlo en un servidor propio. Como característica adicional, destacar la posibilidad de llevar a cabo el análisis multi-variable de Biber (Biber, 1991).

Desde el punto de vista técnico, CQPweb es un software para el tratamiento de corpus lingüísticos muy completo para un administrador con conocimientos de informática: su gestión de usuarios, su historial de búsqueda, sus búsquedas caché y su activa comunidad de usuarios así lo demuestran. No obstante, desde el punto de vista del usuario lingüista sin conocimientos de programación que intenta incorporar sus textos etiquetados para crear corpus monolingües, paralelos o comparables, CQPweb adolece de usabilidad. Ciertos conocimientos de programación resultan imprescindibles para realizar el alineado o etiquetado lingüístico, así como es necesario un completo entendimiento de la estructura de datos soportada por el motor de indexación CQP y para su instalación en un servidor propio.

CQPweb	
Necesidad de asistencia técnica	Sí
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	Sí (2 de agosto de 2016)
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	CQP
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí

n-gramas	No
Interfaz gráfico	Sí
Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	No
Semántico	No
Retórico	No
Disponibilidad	Descargable

Tabla 8 - Resumen de características de CQPweb

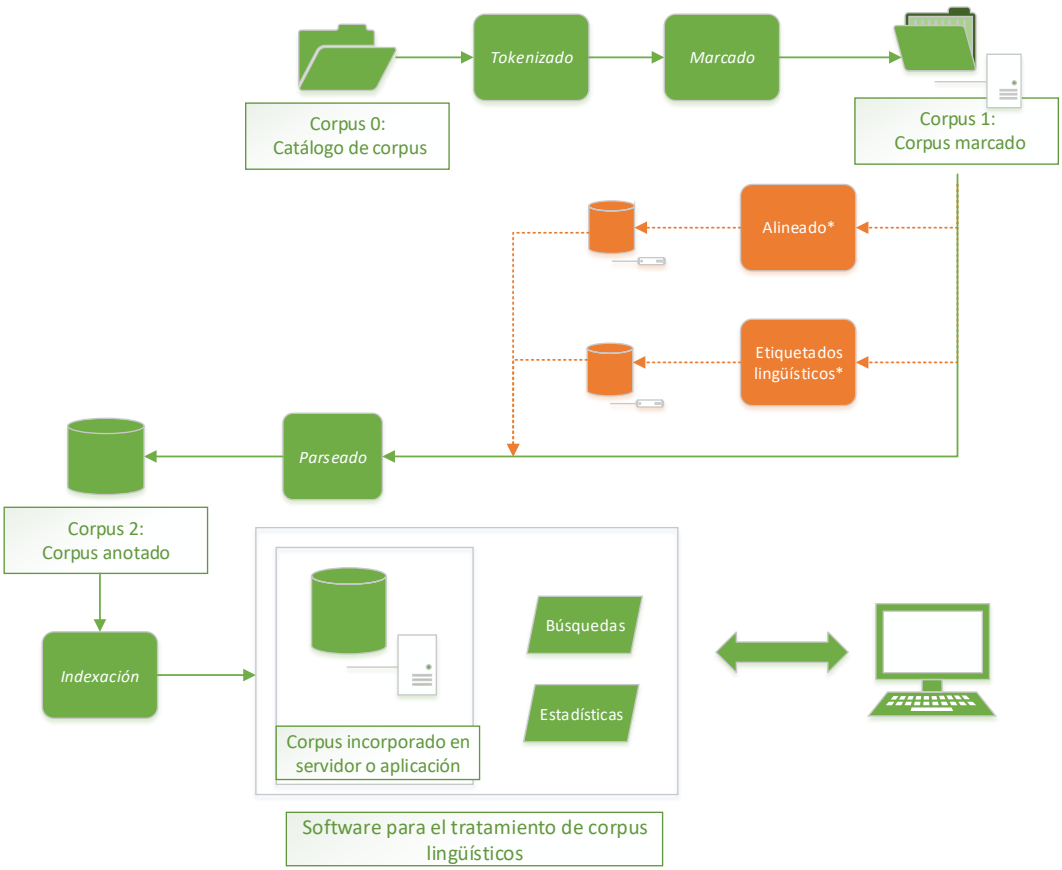


Figura 10 - Funciones realizadas por CQPweb de acuerdo con el flujograma presentado (verde – realizado, naranja – realizado con asistencia/externamente)

3.3.1.8 Corpógrafo

Corpógrafo (Sarmento, Maia, Santos, Pinto, & Cabral, 2006) (Sarmento, Maia, & Santos, 2004) (Maia & Matos, 2008) es un software para el tratamiento de corpus lingüísticos de cuarta generación desarrollado dentro del proyecto Linguateca bajo el soporte del CLUP (CLUP – *Centro Linguística da Universidade do Porto*). La versión 4 de Corpógrafo (Maia & Matos, 2008) es la más reciente y la que será analizada a continuación. Emplea la API de Nooj (Silberztein, 2017), que le añade soporte para etiquetados lingüísticos (Matos & Maia, 2008, pág. 17).

Aunque el acceso web es funcional, la percepción es que un proyecto que no está ni activo ni mantenido en la actualidad. Las razones que lo demuestran son:

- No existe control de registros, las cuentas se activan automáticamente y sin verificación. No existe notificación de que la cuenta está activa.
- Los administradores/desarrolladores no contestan a los correos electrónicos y/o formularios internos de contacto.
- Hay características que no funcionan, se detallarán en el análisis.

Se ha optado por incluirlo en el análisis porque no existen mensajes de su abandono oficial y la aplicación web funciona parcialmente.

Corpógrafo permite compilar corpus monolingües, paralelos bilingües, incluyendo un alineador integrado, sin necesidad de asistencia técnica en ningún momento. No soporta el procesamiento de corpus comparables tal y como se definen en el Capítulo 2: "Conceptos fundamentales", aunque en teoría posibilita la construcción de corpus comparables en distintos idiomas. Además, permite realizar análisis contrastivos a partir de la extracción de unidades terminológicas (Maia & Matos, 2008, pág. 80) Su tecnología de indexación y búsqueda se basa en CQP. Corpógrafo no proporciona soporte para etiquetado directo de corpus lingüísticos, aunque puede incorporar etiquetado gramatical, sintáctico y semántico en las bases terminológicas y en las concordancias a través de la API de Nooj. En cuanto a las estadísticas, permite colocaciones, listas de frecuencias, palabras clave, lista de palabras, así como n-gramas.

Posee funcionalidades relacionadas con la terminología y lexicografía, ya que permite crear bases de datos propias o incluso análisis semi-automáticos en portugués, francés, inglés, español e italiano que posibilitan la obtención de datos sobre la terminología e incluso relaciones semánticas derivadas de las funcionalidades de Nooj.

La plataforma está disponible bajo soporte de la *Universidade do Porto* – Universidad de Oporto, pero como se mencionó anteriormente, tanto el registro como la apariencia de la aplicación web no reciben mantenimiento. Existen

instrucciones para su descarga e instalación, pero algunos de los recursos indispensables para su correcta puesta en marcha no se encuentran disponibles.

Las pruebas *in situ* realizadas para comprobar la robustez y eficacia del sistema han puesto de manifiesto los siguientes problemas:[32]

- El alineador no funciona, y tampoco se puede importar una alineación con un programa externo.
- El interfaz tiene serias limitaciones de usabilidad:
    - Está obsoleta y desfasada de acuerdo con los estándares web actuales.
    - Subir los documentos uno a uno, y su asociación al corpus correspondiente consume bastante tiempo.
    - Ausencia de soporte por parte de los desarrolladores.
- La integración con Nooj, y por consiguiente todas las funcionalidades asociadas descritas, no están disponibles.

Sin que esto sirva de crítica a Corpógrafo V4, no es más que el análisis de un software para el análisis de corpus lingüísticos cuya funcionalidad de construcción de corpus sin supervisión lo convertía en una aplicación interesante, pero que ha quedado obsoleto y sus funcionalidades están comprometidas como consecuencia de no ser mantenido. Por todo ello, la realidad es, que en la actualidad Corpógrafo V4 no soporta ni etiquetados lingüísticos ni alineado sin supervisión.

Aun considerando que todas las características siguieran siendo funcionales en la actualidad, el software poseería las siguientes debilidades:

- Carece de soporte de corpus comparables en el mismo idioma.
- El etiquetado lingüístico sólo es realmente aplicable para la extracción de terminología.

[32] Pruebas realizadas en noviembre de 2016 bajo los navegadores *Firefox* (ver. 49.02) y *Chrome* (ver 54.0.2840.99).

- Posee una usabilidad muy reducida.
- El paralelismo sólo es bilingüe.
- El alineado de textos se realiza a través de un script de CWB (Evert & Hardie, 2011) cuya efectividad es reducida (ver página 218).

Corpógrafo V4	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	Sí
Alineador integrado	No disponible*
Tratamiento de corpus comparables	No*
Tecnología de indexación y/o búsqueda	CQP
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	Sí
Interfaz gráfico	Sí
Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	No disponible*

Etiquetadores integrados	
Gramatical	No disponible*
Semántico	No disponible*
Retórico	No disponible*
Disponibilidad	Soporte propio*

Tabla 9 - Resumen de características de Corpógrafo

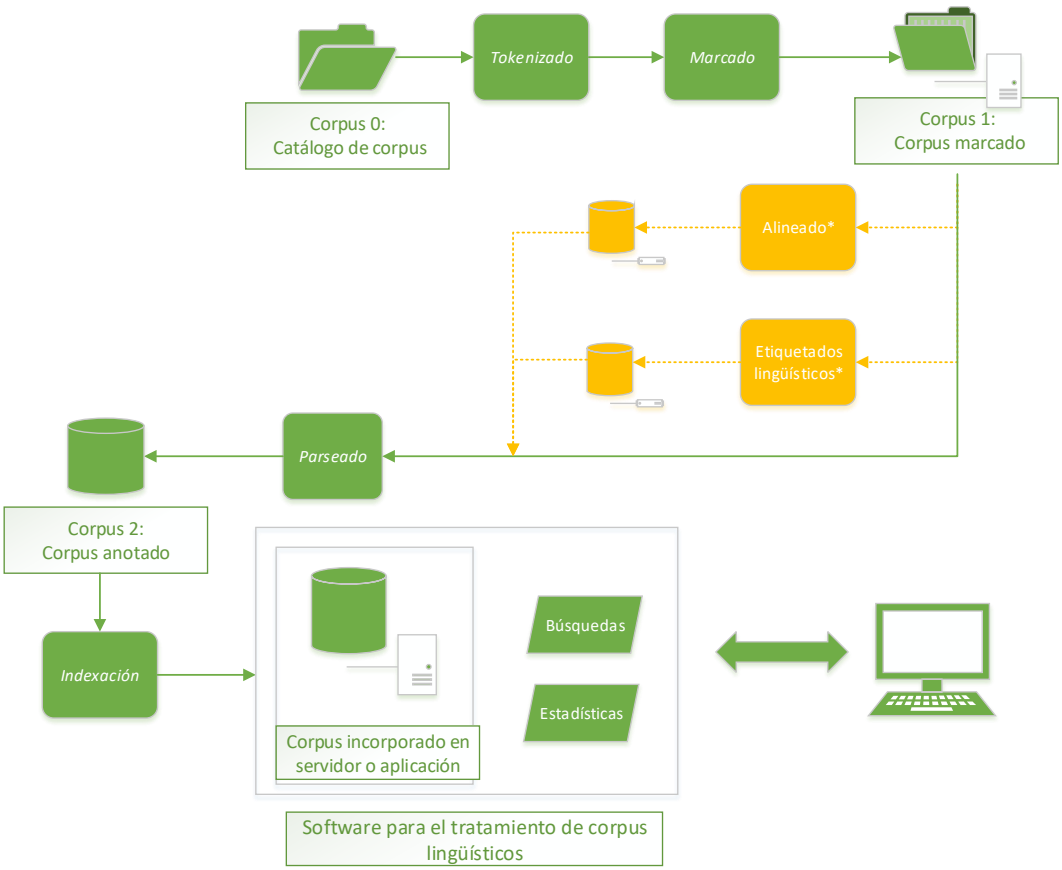


Figura 11 - Funciones realizadas por Corpógrafo de acuerdo con el flujograma presentado (verde – realizado, amarillo – no disponible)

3.3.1.9 Corpuscle

Corpuscle (Meurer, 2012) es un software para el tratamiento de corpus de cuarta generación desarrollado por Paul Merer, bajo el soporte de Uni Computing, perteneciente a *University of Bergen* – Universidad de Bergen. Paul Merer defiende que, en las actuales condiciones, donde la memoria y el espacio en disco son muy económicos, dedicar tiempo a implementar mejoras de compresión de datos no tiene sentido, y la búsqueda de algoritmos de búsqueda más eficientes ha de ser el camino en el procesamiento de corpus (Meurer, 2012, pág. 32). El motor de búsqueda es un sistema propio muy similar a CQP, no obstante, difiere en que Corpuscle emplea las posiciones del corpus para codificar los datos estructurados, mientras que en CQP están ligados al *token* o palabra en cuestión. Su creador manifiesta que con esta estructura y los algoritmos de búsquedas ideados, las consultas resultan mucho más eficientes que en otras arquitecturas (Meurer, 2012, pág. 47).

Corpuscle soporta múltiples tipos de etiquetado, pero no incorpora los etiquetadores, es una plataforma que pone especial énfasis en el acceso web, tanto es así que en el futuro permitirán acceso mediante una API tipo REST a través de JSON (ECMA International, 2013) y XML. Con una atractiva interfaz proporciona estadísticas de colocaciones, listas de frecuencias, palabras clave y lista de palabras. Soporta corpus en múltiples idiomas, pero hasta el momento no soporta paralelismo ni corpus comparables. La necesidad de asistencia técnica para construir el corpus resulta indispensable ya que el corpus ha de cumplir una serie de requisitos para poder ser incluido en el sistema. Hasta el momento, el acceso a la plataforma sólo está disponible bajo desde el servidor de *The CLARINO Bergen Centre* (Meurer, 2017).

Corpuscle	
Necesidad de asistencia técnica	Sí*

Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	No
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	No
Interfaz gráfico	Sí
Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	No
Semántico	No
Retórico	No
Disponibilidad	Soporte propio

Tabla 10 - Resumen de características de Corpuscle

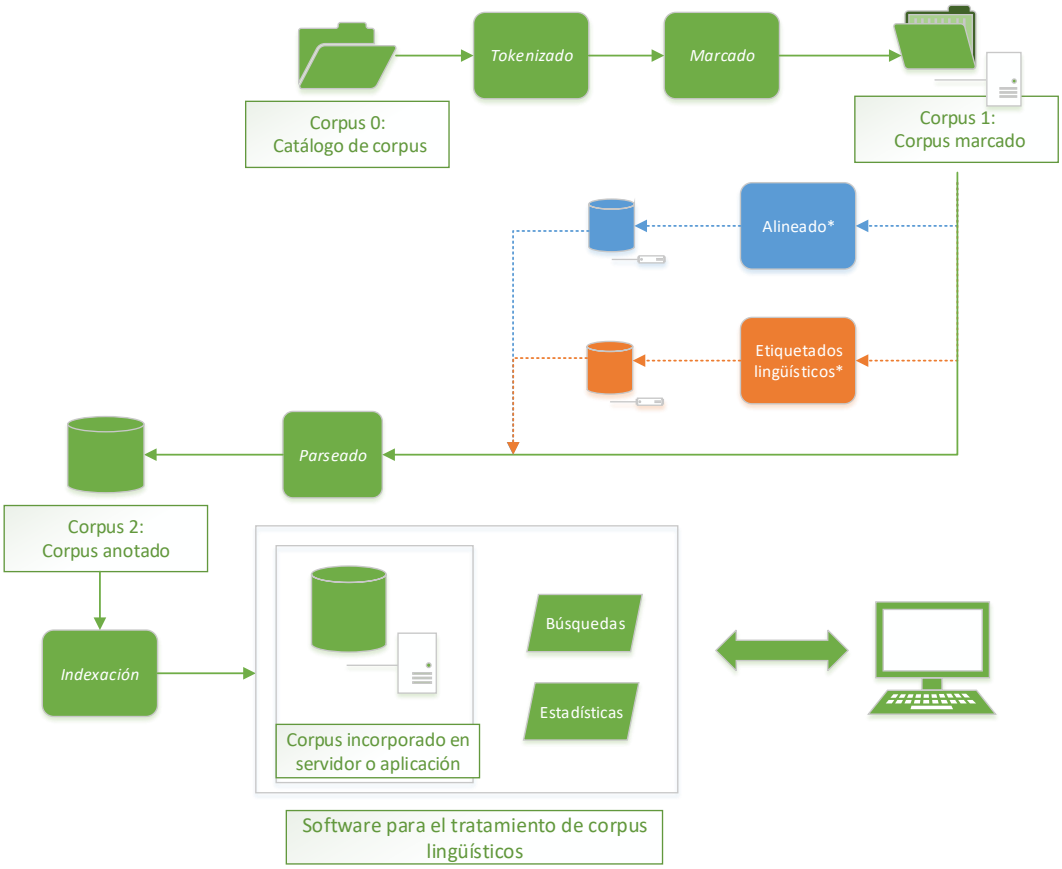


Figura 12 - Funciones realizadas por Corpuscle de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado, naranja – realizado con asistencia/externamente)

3.3.1.10 corpus.byu.edu

corpus.byu.edu (Davies, 2017a) es un software perteneciente cuarta generación desarrollado por Mark Davies, profesor de lingüística perteneciente a la *Brigham Young University* en Utah.

Se trata de un interfaz web que permite el acceso a múltiples corpus lingüísticos pre-cargados y que no permite compilar corpus propios. Los motivos por los que se analiza corpus.bye.edu a pesar de que existen otras plataformas web basadas en motores de indexación SQL, CQP o propios que no lo han sido, son los siguientes:

(1) Es la plataforma oficiosa de acceso al corpus de referencia de mayor tamaño en inglés americano, COCA (Davies, 2008).
(2) Posee un novedoso y eficiente sistema de indexación propio basado en SQL.

Aunque no permite la compilación de corpus por parte del usuario, sí que posibilita comparaciones entre fragmentos de texto propios y corpus cargados en el sistema con el fin de obtener información para construir vocabularios. No es necesaria ni asistencia técnica, ni conocimientos informáticos avanzados en ninguna circunstancia para su uso. corpus.byu.edu contiene corpus en distintos idiomas, aunque no existen corpus paralelos ni comparables. El sistema de indexación y búsqueda es un modelo propio basado en SQL, que permite realizar búsquedas extremadamente rápidas. Soporta únicamente etiquetado gramatical por medio de CLAWS, para el idioma inglés. En el resto de idiomas, aunque existe, no se han podido averiguar los etiquetadores gramaticales utilizados. Las estadísticas que permite obtener son: lista de palabras, lista de palabras clave, lista de frecuencias, colocaciones, así como n-gramas e incluso la construcción de vocabularios.

En resumen, corpus.byu.edu tiene su principal déficit en que no permite compilar corpus propios. A lo que hay que añadir, como en la mayoría de casos analizados, que no permite el tratamiento de corpus paralelos ni comparables.

Como puntos fuertes, se encuentran la rapidez de sus consultas, por ejemplo, consultas complejas sobre un corpus de 520 millones de palabras normalmente tardan de 2 a 3 segundos (Davies, 2017b), y que es un excelente recurso para consultar corpus de referencia como COCA o BNC.

corpus.byu.edu	
Necesidad de asistencia técnica	No
Compilación de corpus	No
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	Sí
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	SQL
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-grams	Sí
Interfaz gráfico	Sí
Generación	4ª
Alojada en web	Sí
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí (CLAWS y otros)
Semántico	No

Retórico	No
Disponibilidad	Soporte propio

Tabla 11 - Resumen de características de corpus.byu.edu

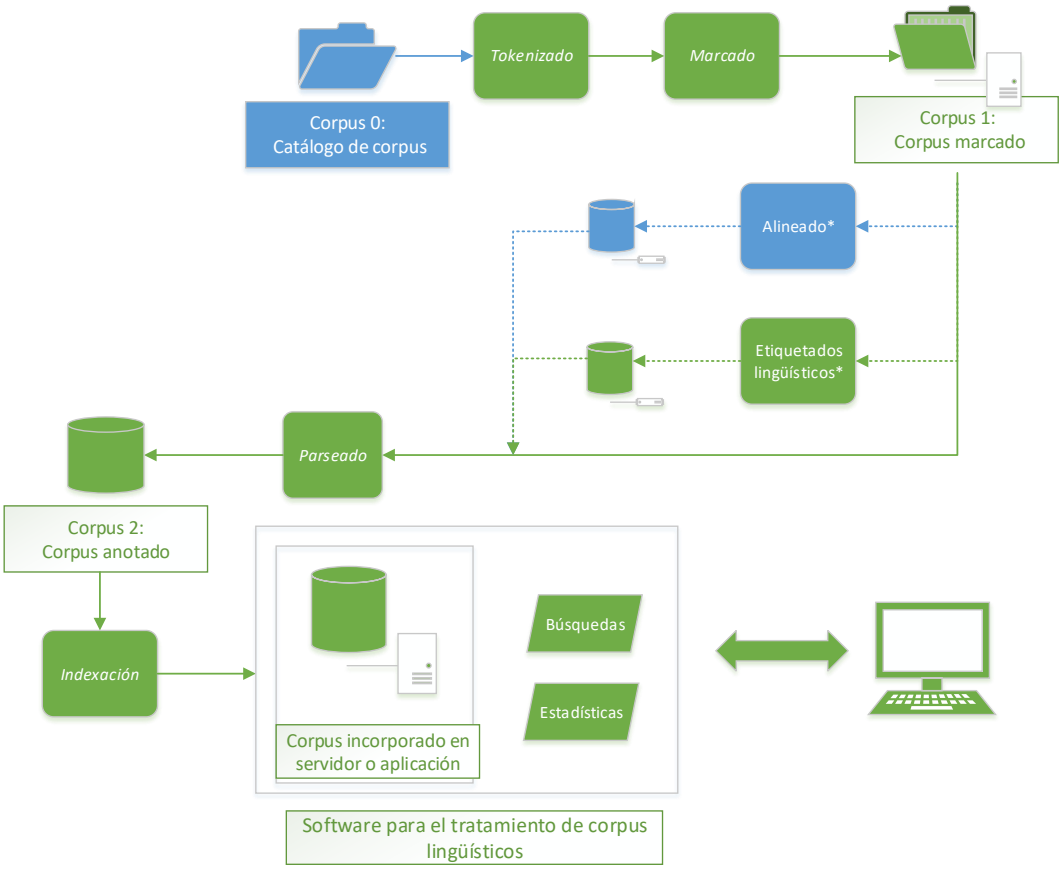


Figura 13 - Funciones realizadas por corpus.bye.edu de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.11 #LancsBox: Lancaster Desktop Corpus Toolbox

#LancsBox: Lancaster Desktop Corpus Toolbox (Brezina, McEnery, & Wattam, 2015) es un framework para el tratamiento de corpus lingüísticos de tercera generación desarrollado por *Lancaster University* – Universidad de Lancaster.

#LancBox permite la compilación de corpus lingüísticos por parte del usuario sin asistencia técnica en ningún momento del proceso. Soporta múltiples idiomas,

permite el procesamiento de corpus monolingües, pero carece de soporte para corpus comparables o bi-/multilingües paralelos. Hay que señalar que, aunque no soporta corpus comparables sí que permite la comparación de corpus mediante la división de su interfaz. No existe demasiada información sobre su sistema de indexación y búsqueda, sólo que está desarrollado en Java (Gosling, Joy, Steele, Bracha, & Buckley, 2014). Sus desarrolladores aseguran que el tamaño del corpus soportado depende de la potencia del equipo utilizado, permitiendo el procesamiento de corpus de 1 millón de palabras sin dificultad en cualquier equipo, y hasta 70 millones en los ordenadores más potentes. #LancsBox posee un etiquetador gramatical integrado, Treetagger, pero no soporta ni incluye etiquetador semántico ni retórico. En cuanto a las estadísticas, es la principal cualidad de #Lancbox, proporcionando una atractiva vista de resultados de las denominadas "colocaciones en contexto" (Brezina, McEnery, & Wattam, 2015), un tipo de estadística relacionada con las *collocation networks* de (Phillips, 1989).[33] Para mostrar esta información #LancsBox posee integrado el software, antes independiente, GraphColl (Wattam, 2015). La extracción de las colocaciones en contexto es muy flexible, posibilitando el ajuste y modificación del algoritmo estadístico utilizado. Entre el resto de estadísticas, permite la extracción de listas de frecuencias, pero no n-gramas, ni lista de palabras clave.

Como conclusión, #LancsBox es un software que permite la consulta de corpus monolingües y la extracción de colocaciones con una gran variedad de parámetros y con un formato visual muy atractivo, sin necesitar asistencia técnica en ningún momento. Sin embargo, carece de soporte para otros etiquetados distintos del gramatical, y no admite corpus paralelos bi-/multilingües ni comparables.

[33] En la página 53 se dijo que el software de tercera generación carecía de esta funcionalidad, si bien es cierto que #LancsBox pertenece a la tercera generación, su última versión data de 2016.

#LancsBox	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	No
Alineador integrado	No
Tratamiento de corpus comparables	No*
Tecnología de indexación y/o búsqueda	Propio (Java)
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	No
n-grams	No
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí (Treetagger)
Semántico	No

Retórico	No
Disponibilidad	Descargable

Tabla 12 - Resumen de características de #LancsBox

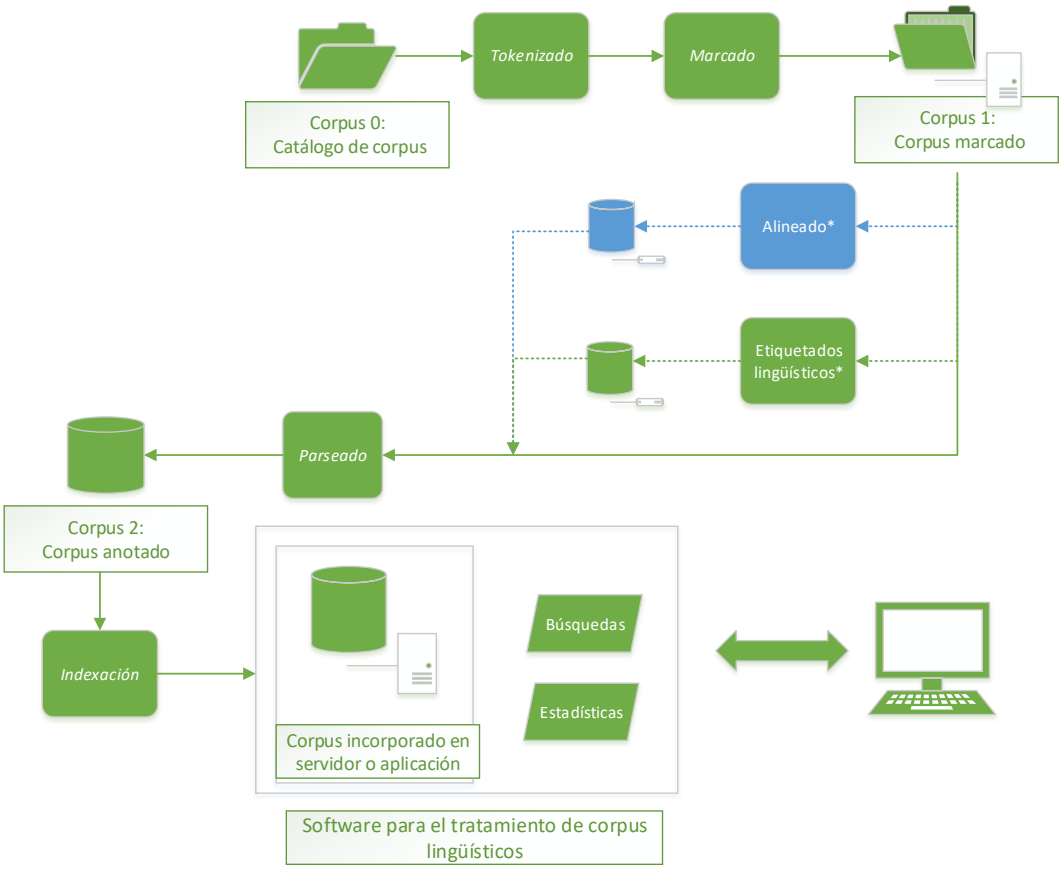


Figura 14 - Funciones realizadas por #LancsBox de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.12 Otros frameworks

En este subapartado se describirán aquellos frameworks que si bien poseen las propiedades de: compilar un corpus, proporcionar búsquedas y/o obtener algunas estadísticas; su objetivo principal es más servir de apoyo para la creación de recursos lingüísticos como gramáticas y diccionarios, o como software de

anotación. Es decir, no se han diseñado para llevar a cabo estudios sobre corpus lingüísticos a semejanza del resto de software analizado, pero éste puede realizarse.

3.3.1.12.1 CLaRK System

CLaRK System (Simov, et al., 2001) es un software de tercera generación para el tratamiento de corpus lingüísticos desarrollado por Kiril Simov, Zdravko Peev, Milen Kouylekov, Alexander Simov y Marin Dimitrov bajo el sustento de *Tübingen-Sofia International Graduate Programme in Computational Linguistics and Represented Knowledge* (CLaRK), y cuyo principal objetivo es la creación de recursos lingüísticos como etiquetadores gramaticales o sintácticos, gramáticas, diccionarios para el idioma búlgaro.

CLaRK System tiene un sistema de anotación y búsqueda basado en tecnología XML, que se utiliza para crear restricciones en las búsquedas y posibilita la creación de los distintos componentes lingüísticos Carece de soporte para corpus paralelos o comparables, extracción de estadísticas y de etiquetados lingüísticos integrados. Su disponibilidad es a través de descarga.

CLaRK System	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	No
Tratamiento de corpus paralelos	No
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio (XPath)

Estadísticas	
Lista de frecuencias	No
Colocaciones	No
Lista de palabras claves	No
n-gramas	No
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí (propio)
Semántico	No
Retórico	No
Disponibilidad	Descargable

Tabla 13 - Resumen de características de CLaRK System

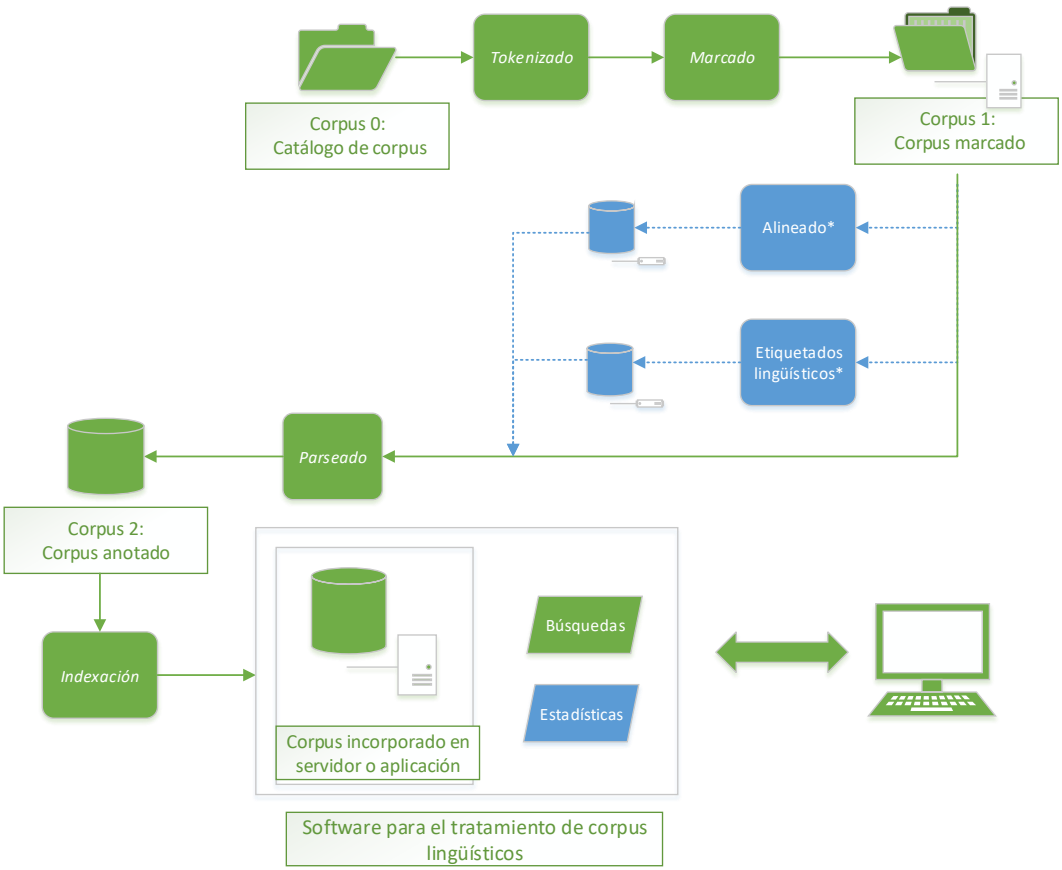


Figura 15 - Funciones realizadas por CLaRK System de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.12.2 UAM Corpus Tool

UAM Corpus Tool (O'Donnel, 2008) es un software para el tratamiento de corpus lingüísticos desarrollado por Mick O'Donnel bajo el amparo de la Universidad Autónoma de Madrid. UAM Corpus Tool permitir a los usuarios sin conocimientos específicos de programación etiquetar textos o incluso imágenes exportando el resultado en formato XML de manera sencilla y rápida. Sobre estos documentos XML, es posible llevar a cabo estudios lingüísticos o construir bases de datos para entrenar modelos estadísticos relacionados con diversas áreas del procesamiento de lenguaje natural.

UAM Corpus Tool permite la compilación de corpus, maneja corpus monolingües en distintos idiomas, pero no paralelos ni comparables. Su motor de búsquedas se basa en CQP. Proporciona estadísticas relacionadas con n-gramas, colocaciones, listas de frecuencia, palabras clave o listas de palabras. Posee un sencillo interfaz gráfico, pero no tiene acceso web, es por ello que pertenece a la tercera generación. Soporta etiquetado manual y tiene integradas funciones de etiquetado gramatical por medio de Treetagger y Stanford POS Tagger (Toutanova, Klein, Manning, & Singer, 2003), etiquetado sintáctico para francés, alemán, árabe y chino por medio de Stanford Parser (Chen & Manning, 2014). Carece de funcionalidad para realizar etiquetado retórico o semántico, aunque soporta cualquier tipo de anotación lingüística. El software ha de instalarse en el equipo donde quieran llevarse a cabo las distintas tareas.

En resumen, su principal punto fuerte es el soporte de cualquier tipo de anotación lingüística personalizada y que tiene integrados el etiquetado gramatical y sintáctico. No soporta el paralelismo ni el tratamiento de corpus comparables, tampoco integra funcionalidades para llevar a cabo etiquetados semánticos o retóricos ni tiene acceso web.

UAM Corpus Tool	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	No
Alineador integrado	No
Tratamiento de corpus comparables	No

Tecnología de indexación y/o búsqueda	CQP
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí
Lista de palabras claves	Sí
n-gramas	Sí
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí
Etiquetadores integrados	
Gramatical	Sí
Semántico	No
Retórico	No
Disponibilidad	Descargable

Tabla 14 - Resumen de características de UAM Corpus Tool

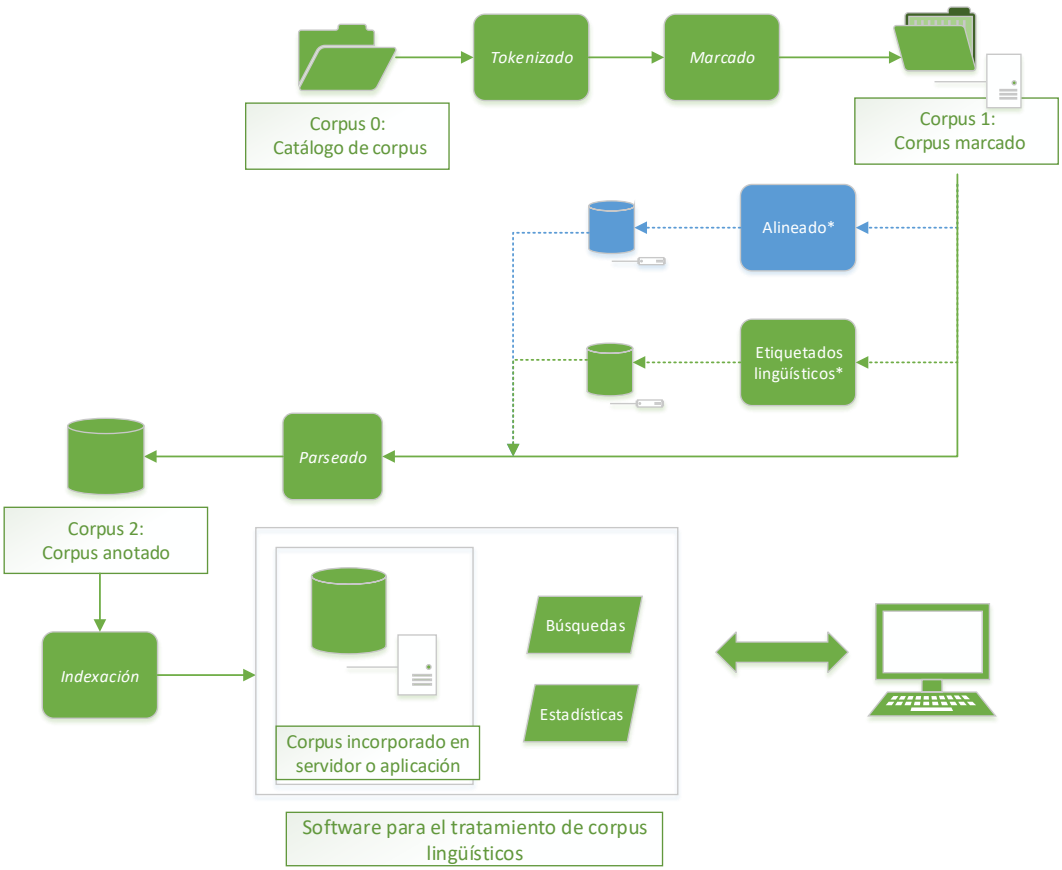


Figura 16 - Funciones realizadas por UAM Corpus Tool de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.12.3 Nooj

Nooj (Silberztein, 2017) es un software para el tratamiento de corpus lingüísticos desarrollado por Max Silberztein en 2002 y desarrollado bajo el amparo de la *Université de Franche-Comté* - Universidad del Franco Condado. Se trata de un software de tercera generación y por tanto sin acceso vía web, aunque permite la integración de sus componentes en programas externos que podrían ejecutarse vía web a partir de un desarrollo propio. Su principal funcionalidad es la posibilidad de crear recursos léxicos, terminológicos y gramaticales. Además, muchos de estos datos se extraen de forma automática.

El motor de búsquedas se basa en un sistema propio (Silberztein, 2005) desarrollado en Microsoft .NET (Platt, 2002). No requiere asistencia técnica especializada para llevar a cabo la construcción de corpus o análisis. Soporta corpus en múltiples lenguas, pero no maneja corpus paralelos o comparables. Posee estadísticas de colocaciones y listas de frecuencias, pero no lista de palabras clave, ni n-gramas. Nooj integra distintos tipos de etiquetado lingüístico: sintáctico, gramatical y permite crear etiquetados semánticos personalizados basados en sinónimos de WordNet (Princeton University, 2010) o propios. No admite etiquetado retórico. Nooj es proporcionado para su descarga e instalación en un equipo personal.

En resumen, su punto fuerte es la generación semiautomática de gramáticas y diccionarios, pero no maneja corpus paralelos, ni comparables ni el acceso vía web directo.

Nooj	
Necesidad de asistencia técnica	No
Compilación de corpus	Sí
Tratamiento de corpus en múltiples lenguas	Sí
Tratamiento de corpus paralelos	No
Alineador integrado	No
Tratamiento de corpus comparables	No
Tecnología de indexación y/o búsqueda	Propio (.NET)
Estadísticas	
Lista de frecuencias	Sí
Colocaciones	Sí

Lista de palabras claves	No
n-gramas	No
Interfaz gráfico	Sí
Generación	3ª
Alojada en web	No
Soporte de etiquetados nativo	Sí*
Etiquetadores integrados	
Gramatical	Sí
Semántico	Sí (WordNet)
Retórico	No
Disponibilidad	Descargable

Tabla 15 - Resumen de características de Nooj

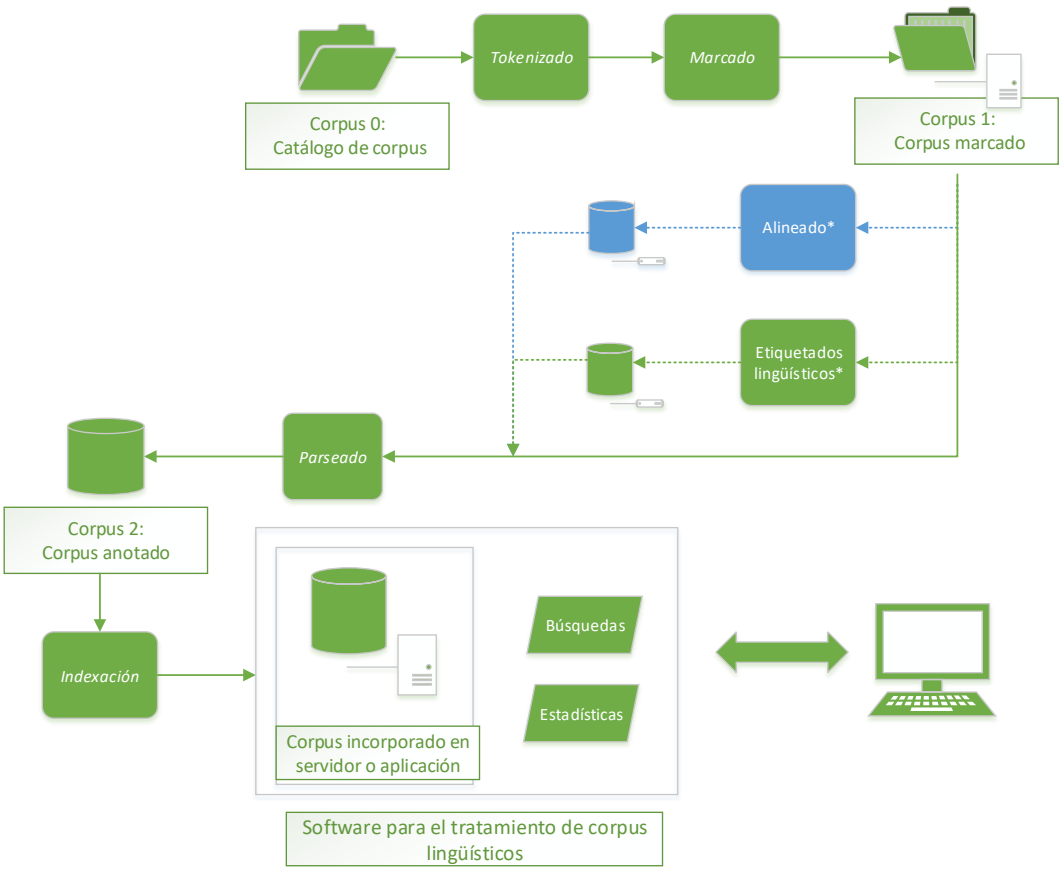


Figura 17 - Funciones realizadas por Nooj de acuerdo con el flujograma presentado (verde – realizado, azul – no realizado)

3.3.1.12 Resumen de características de los frameworks

	WordSmith	**AntConc y AntPConc**	**MonoConc y ParaConc**	**Wmatrix**	**Sketch Engine**	**IntelliText**	**CQPweb**
No requiere asistencia técnica	✗	✗	✓	✓	✗	✓	✗
Compilación de corpus	✓	✓	✓	✓	✓	✓	✓
Tratamiento de corpus multilingües	✓	✓	✓	✗	✓	✓	✓
Tratamiento de corpus paralelos	✓	✓	✓	✗	✓	*	✓
Alineador integrado	✗	✗	✗	✗	✗	✗	✗
Tratamiento de corpus comparables	✗	✗	✗	✗	✗	✗	✗
Tecnología de indexación y búsqueda	Propio	Propio (Python)	Propio	Propio (Perl)	CQP	CQP	CQP
Estadísticas							
Lista de frecuencias	✓	✓	✓	✓	✓	✓	✓
Colocaciones	✓	✓	✓	✓	✓	✓	✓
Lista de palabras claves	✓	✓	✓	✓	✓	✓	✓
n-gramas	✓	✓	✗	✓	✓	✗	✗
Interfaz gráfico	✓	✓	✓	✓	✓	✓	✓
Generación	3ª	3ª	3ª	4ª	4ª	4ª	4ª
Alojada en web	✗	✗	✗	✓	✓	✓	✓
Soporte de etiquetados	✓	✓	✓	✓	✓	✓	✓
Etiquetadores integrados							
Gramatical	✗	✗	✗	✓	✓	✓	✗
Semántico	✗	✗	✗	✓	✗	✗	✗
Retórico	✗	✗	✗	✗	✗	✗	✗
Disponibilidad	Descargable	Descargable	Descargable	Soporte propio	Soporte propio	Soporte propio*	Ambos

	Corpógrafo	**Corpuscle**	**corpus.byu.edu**	**#LancsBox**	**CLaRK System**	**UAM Corpus Tool**	**Nooj**
No requiere asistencia técnica	✓	✗	✓	✓	✓	✓	✓
Compilación de corpus	✓	✓	✓	✓	✓	✓	✓
Tratamiento de corpus en múltiples lenguas	✓	✓	✓	✓	✗	✓	✓
Tratamiento de corpus paralelos	*	✗	✗	✗	✗	✗	✗
Alineador integrado	*	✗	✗	✗	✗	✗	✗
Tratamiento de corpus comparables	✗	✗	✗	✗	✗	✗	✗
Tecnología de indexación y búsqueda	CQP	Propio	SQL	Propio (Java)	Propio (XPath)	CQP	Propio (.NET)
Estadísticas							
Lista de frecuencias	✓	✓	✓	✓	✗	✓	✓
Colocaciones	✓	✓	✓	✓	✗	✓	✓
Lista de palabras claves	✓	✓	✓	✗	✗	✓	✗
n-gramas	✓	✗	✓	✗	✗	✓	✗
Interfaz gráfico	✓	✓	✓	✓	✗	✓	✓
Generación	4ª	4ª	4ª	3ª	3ª	3ª	3ª
Alojada en web	✓	✓	✓	✗	✗	✗	✗
Soporte de etiquetados	✓	✓	✓	✓	✓	✓	✓
Etiquetadores integrados							
Gramatical	*	✗	✓	✓	✓	✓	✓
Semántico	*	✗	✗	✗	✗	✗	✓
Retórico	✗	✗	✗	✗	✗	✗	✗
Disponibilidad	Soporte propio*	Soporte propio	Soporte propio	Soporte propio	Descargable	Descargable	Descargable

Tabla 16 - Resumen de características de los framework

3.3.2 Toolkits, suites y similares

La intrínseca relación entre el corpus lingüístico y el NLP hace que muchos toolkits, *suites* y similares relacionados con el NLP e incluso frameworks, hayan sido analizados.

Los términos "toolkit, *suite* y similares' y 'framework especializado' se definieron en la sección 2.3 Software especializado para el tratamiento de corpus lingüísticos.

De acuerdo con estas definiciones, pueden considerarse dos toolkits dedicados en exclusiva al tratamiento de corpus lingüísticos, se trata de CWB (Evert & Hardie, 2011)[34] y de Uplug Corpus Tools (Tiedemann, 2013).

En el caso de NLP existen multitud de opciones que, aunque más orientados a tareas relacionadas con el procesamiento del lenguaje, también permiten realizar las tareas básicas de compilar un corpus y búsquedas de concordancias.

Sin embargo, si la hipótesis de trabajo es proporcionar un framework especializado en el tratamiento de corpus lingüísticos bi-/multilingües de gran tamaño con múltiples capas de anotación y en el que el usuario lingüista pueda llevar a cabo todo el proceso sin intervención técnica, el empleo único de este tipo de software no es la solución.

3.3.2.1 The IMS Open Corpus Workbench (CWB)

The IMS Open Corpus Workbench (CWB) (Evert & Hardie, 2011) es una colección de herramientas que permite manejar y hacer consultas sobre grandes corpus anotados lingüísticamente.

De acuerdo con (Evert & Hardie, 2011, pág. 1), CWB es una arquitectura para el análisis de corpus lingüísticos ampliamente utilizada, diseñada por el IMS (*Institut*

[34] Denominado en realidad *workbench*, aunque la idea es similar a la expresada con el término *toolkit*.

für Maschinelle Sprachverarbeitung) de *University of Stuttgart* - Universidad de Stuttgart, que consiste en una serie de herramientas para indexar, gestionar y consultar corpus muy grandes, con múltiples anotaciones a nivel de palabra. Uno de los componentes centrales de CWB es el procesador de consultas de corpus, CQP Query Language o CQP-syntax o simplemente CQP (Evert, 2009), un sistema de concordancias extremadamente potente y eficiente que implementa un lenguaje flexible de búsqueda de dos niveles, que permite especificar patrones de consulta complejos tanto al nivel de una palabra como de anotación (Evert & Hardie, 2011, pág. 8).

CWB proporciona funciones para administrar corpus lingüísticos: nombres, metadatos o ubicación de almacenamiento; además de las funciones relacionadas con la indexación, las búsquedas y las estadísticas empleando el mencionado CQP.

Está orientado a su utilización por parte de especialistas informáticos, que lo utilizan como *back-end* (capa de acceso de datos) para sus gestores de corpus. Es necesario desarrollar un *front-end* (capa de presentación) que interaccione con los datos proporcionados por el *back-end*.

La solución CWB/CQP es ampliamente utilizados como sistemas de gestión e indexación en múltiples frameworks para el tratamiento de corpus lingüísticos, como por ejemplo los previamente analizados CQPweb y Sketch Engine.

3.3.2.2 Uplug Corpus Tools

Uplug Corpus Tools (Tiedemann, 2013) es un toolkit formado por diversas herramientas que permiten la creación y tratamiento de corpus paralelos anotados. Fue desarrollado parcialmente en la tesis doctoral de Jörg Tiedemann (Tiedemann, 2003) y a través del proyecto PLUG (*Parallel Corpora in Linköping, Uppsala and Göterborg*). Uplug Corpus Tools consiste en una serie de módulos que permiten el procesamiento de corpus paralelos, y obtener datos relevantes para llevar a cabo estudios relacionados con la lexicografía y las máquinas de traducción.

Uplug Corpus Tools puede tratar corpus lingüísticos a semejanza de los frameworks para el tratamiento de corpus lingüísticos descritos en el apartado 3.3.1 Frameworks para el tratamiento de corpus lingüísticos, pero a partir de módulos interconectados entre sí por medio de conocimientos de programación. Por otra parte, carece de funcionalidad interna para etiquetar textos de forma distinta al etiquetado gramatical habitual, así como interfaz gráfico para la mayoría de los procesos. Posee una funcionalidad de alineación de textos muy potente además de concordancias sobre corpus paralelos.

3.3.2.3 Toolkits, *suites* y frameworks especializados en NLP

La construcción de un flujo de trabajo como el mostrado en la Figura 2 a través del uso exclusivo de distintos componentes creados por medio de toolkits, *suites* y frameworks especializados en NLP es realizable. Sin embargo, esta estrategia posee una serie de inconvenientes que impiden que la utilización de recursos software perteneciente a esta categoría sea una solución factible para el tipo de framework que se pretende construir en esta tesis doctoral.

Los principales inconvenientes son los siguientes:

(1) Este tipo de software se caracteriza por proporcionar componentes individuales que han de integrarse en un flujo de trabajo.

(2) Para ello, es necesario y/o recomendable disponer de conocimientos de programación, como mínimo, para:
   - Interconectar las entradas y salidas
   - Tratar datos y resultados

(3) Es indispensable conocer en profundidad el toolkit, *suite* o framework en cuestión, así como la API proporcionada y el lenguaje de programación empleado en el entorno de trabajo. La curva de aprendizaje suele ser alta incluso para los programadores cualificados.

(4) En la mayoría de los casos, es obligatorio emplear los recursos ofrecidos por el toolkit, *suite* o framework para llevar a cabo las acciones del flujo de

trabajo. Por ejemplo, el uso de un etiquetador gramatical cuyos resultados no son plenamente satisfactorios para un idioma concreto.

(5) Relacionado con el punto anterior, la no disponibilidad de algunas funcionalidades, por ejemplo, un programa de alineación, y el hecho de no poder utilizar una alternativa externa por ser incompatible o por las dificultades técnicas para su integración.

(6) Aunque el entorno proporcionado por el toolkit, *suite* o framework de NLP permita construir corpus y realizar búsquedas, el sistema de indexación no es suficientemente sofisticado como para facilitar búsquedas robustas y rápidas sobre corpus de gran tamaño con diversos etiquetados lingüísticos

(7) Su integración en aplicaciones web no es trivial y no siempre es posible.

Algunos *suites* como GATE (Cunningham, et al., 2011) poseen interfaces para el desarrollo de diversas operaciones, funcionando como cajas negras, que permiten construir un flujo sin tener conocimientos de programación. No obstante, su operatividad es reducida al tratar grandes bloques de información y su funcionalidad no es tan completa como para posibilitar la creación de un flujograma como el mostrado en la Figura 2, además de los problemas mencionados anteriormente.

Por ello, este tipo de software está más orientado a la creación de recursos lingüísticos para el entrenamiento de modelos estadísticos, gracias a los cuales se pueden construir, entre otros, recursos los siguientes:

- Etiquetadores gramaticales en distintos idiomas.
- Alineadores.
- Etiquetadores semánticos.
- Traductores automáticos.
- Herramientas de NER de (NER - Named Entity Recognition en inglés).

- Herramientas de detección de MWE (MWE - M*ultiWord Expression* en inglés).[35]
- Herramientas de IR (IR - *Information Retrieval* en inglés).

O bien permite crear aplicaciones NLP *ad hoc* para tareas concretas como, por ejemplo:

- Aplicación de WSD (WSD - *Word Sense Disambiguation* en inglés).[36]
- Aplicación NER para un ámbito determinado.
- Parsers específicos.

En resumen, todos estos recursos informáticos, pese a que algunos son autodenominados frameworks en sus especificaciones, están más orientados a programadores expertos que a usuarios lingüistas (Nadkarni, Ohno-Machado, & Chapman, 2011, pág. 549), no realizan todas las tareas descritas en el flujo y su utilización es compleja aunque se posean altos conocimientos técnicos de programación.

Algunos ejemplos de toolkits, *suites* y frameworks especializados en NLP se describen a continuación:

3.3.2.3.1 NLTK

NLTK (Bird, Klein, & Loper, 2009) es uno de los toolkits más utilizados para tratar corpus lingüísticos en investigación y aprendizaje. Desarrollado por Steven Bird y Edward Loper bajo el sustento de *University of Pennsylvania* - Universidad de Pensilvania, programado en Python (Python Software Foundation, 2017b), proporciona acceso a una gran colección de corpus y material lingüístico (diccionarios, modelos estadísticos, etiquetadores). Permite crear aplicaciones de forma muy sencilla, lo que resulta de gran facilidad para labores de enseñanza o

[35] De aquí en adelante se utilizará el término MWE en lugar de *MultiWord Expression.*
[36] De aquí en adelante se utilizará el término WSD en lugar de *Word Sense Disambiguation.*

pedagógicas. NLTK posee funcionalidades específicas de gestión de corpus y estadísticas muy potentes:

(1) Acceso a corpus.
(2) Colocaciones.
(3) Programas de concordancias.
(4) Machine Learning.
(5) Medidas de evaluación.
(6) Probabilística y estimación

De acuerdo con (Bird, Klein, & Loper, 2009, pág. xv), sus principales objetivos que persigue NLTK son la simplicidad, la consistencia, la modularización y la extensibilidad.

3.3.2.3.2 GATE

GATE (Cunningham, et al., 2011) es una *suite* de herramientas para NLP desarrollada en Java (Gosling, Joy, Steele, Bracha, & Buckley, 2014). Creada bajo el auspicio de *University of Sheffield* – Universidad de Sheffield. Se caracteriza por poseer un potente interfaz gráfico que permite a los usuarios lingüistas sin conocimientos de programación ejecutar diversas tareas, representadas mediante módulos, sobre flujos de programación creados por ellos mismos. Por ejemplo, un usuario puede llevar a cabo el proceso de tokenizar un texto, etiquetarlo gramaticalmente y extraer las entidades que aparezcan junto a su frecuencia de aparición mediante un módulo NER estándar. Permite su integración en aplicaciones desarrolladas por los programadores mediante librerías Java, o incluso en aplicaciones de Apache Tomcat (Chopra, Li, & Genender, 2007) para servirlas directamente en la web. Entre las características específicas de GATE destacan sus algoritmos de *Machine Learning* y su soporte ontológico por citar algunas de ellas.

3.3.2.3.3 FreeLing

FreeLing (Padró & Stanilovsky, 2012) es una *suite* de herramientas de análisis textuales. Desarrollado bajo el auspicio del centro de investigación TALP (*The Center for Language and Speech Technologies and Application*s) perteneciente a la *Universitat Politècnica de Catalunya* - Universidad Politécnica de Cataluña. De acuerdo con (TALP-UPC, 2016), FreeLing es una librería orientada al desarrollador que proporciona servicios para analizar el lenguaje natural en múltiples idiomas, es decir, proporciona módulos para procesar texto que sirven como base para diversas aplicaciones lingüísticas, relacionadas con *Machine translation* o etiquetadores. No hay que confundir FreeLing con una herramienta de análisis de texto independiente, hay que integrarlo en otra aplicación mayor desarrollada por un programador. De hecho, aunque está escrita en C++, existen APIs para que puedan integrarse en aplicaciones desarrolladas en Java, Perl o Python. Por todo ello, carece de interfaz de usuario.

3.3.2.3.4 Ellogon

Ellogon (Petasis, Karkaletsis, Paliouras, Androutsopoulos , & Spyropoulos, 2002) es, de acuerdo con la documentación de su web (Petasis, 2014), un entorno de ingeniería del lenguaje de propósito general multi-idioma y multiplataforma. Está programado en C++ y posee funcionalidades similares a las definidas en GATE: potente interfaz gráfico que permite a los usuarios lingüistas sin conocimientos de programación ejecutar diversas tareas. Además, permite su integración en aplicaciones desarrolladas en una gran variedad de lenguajes C++, Java, Perl o Python entre otros.

La principal diferencia respecto a GATE es que Ellogon almacena las anotaciones lingüísticas separadas de los documentos originales y que carece de soporte ontológico (Oliveira, 2004, pág. 127), un inconveniente ya que esta característica facilitaría la extracción de datos. Para más información sobre las diferencias entre ambos entornos consultar (Bank & Schierle, 2012, págs. 3480-3481).

3.3.2.2.5 Otros

Existen multitud de ejemplos adicionales, más orientados si cabe al NLP, por ejemplo: Apache OpenNLP (The Apache Software Foundation, 2017), ScalaNLP (Hall, 2014) o NLP4J (Emory NLP, 2017). Además de una gran cantidad de librerías creadas por programadores para tratar de resolver sus hipótesis de investigación y que son liberadas en distintos repositorios web como GitHub (GitHub Inc, 2017), PyPI (Python Software Foundation, 2017a) o npm (npm Inc, 2017).

3.4 Conclusiones

El análisis de software disponible ha puesto de manifiesto que no existen frameworks específicos para tratar corpus bi-/multilingües paralelos y comparables que no requieran asistencia técnica en algún momento del proceso. El software que es capaz de procesar corpus paralelos requiere asistencia especializada y/o de diversos conocimientos de programación para usar programas de alineación, crear y ejecutar parsers o utilizar entornos operativos específicos. En cuanto al procesamiento de corpus comparables en uno o varios idiomas, no existe un software operativo y funcional en la actualidad que permita su procesamiento. Por último, la incorporación autónoma de etiquetados lingüísticos gramaticales ha sido superada en algunos frameworks, sin embargo, sólo Wmatrix incorpora anotación semántica no supervisada en lengua inglesa, y no existe ningún framework que integre un etiquetador retórico.

Capítulo

4

Hipótesis de trabajo: necesidades, objetivos y nicho

La tesis que aquí se presenta consiste en la creación de un framework para el tratamiento de corpus lingüísticos monolingües, bi-/multilingües paralelos y comparables con distintas capas de anotación, en el que el usuario final, un lingüista, es el actor principal sobre el que gira el desarrollo del software. Se trata, por tanto, de una tesis doctoral realizada desde el punto de vista de la usabilidad y la aplicabilidad de los usuarios finales, que evita la asistencia por parte de especialistas técnicos en cualquiera de las operaciones habituales acaecidas en la creación o búsqueda sobre corpus lingüísticos. Este framework se denominará *ACTRES Corpus Manager*.

Nuestra hipótesis de partida es que la creación de corpus lingüísticos anotados, ya sean corpus monolingües, corpus bi-/multilingües paralelos o corpus comparables, en cualquiera de las distintas aplicaciones para el tratamiento de corpus lingüísticos actuales implica la participación indispensable de especialistas informáticos o, al menos, la disposición de conocimientos de programación, para alcanzar un desarrollo eficaz como el que se pretende alcanzar en esta tesis doctoral.

Por otro lado, las búsquedas estándar sobre corpus lingüísticos, aunque centradas en el usuario final, dependen del rango de tipos de corpus soportados por el propio software, luego, el usuario también está limitado. Además, no siempre es sencillo realizar búsquedas complejas empleando patrones de búsqueda y/o etiquetados, salvo que se disponga de conocimientos avanzados del lenguaje de consulta del software para el tratamiento de corpus lingüísticos que se haya utilizado.

Según los datos analizados en el Capítulo 3: "Estado de la cuestión: Software especializado para el tratamiento de corpus lingüísticos", no existe ningún software en la actualidad que cumpla con los criterios de independencia de asistencia técnica que se quieren alcanzar en el desarrollo de esta tesis en cuanto a creación y búsqueda sobre cualquier tipo de corpus lingüístico anotado descrito con anterioridad. Esta tesis parte del supuesto de que la mayor parte de los lingüistas no tienen por qué disponer de conocimientos de programación y, por lo tanto, carecen de la formación indispensable para paliar las carencias de un software para el tratamiento de corpus que no disponga de funcionalidades propias de integración de etiquetados lingüísticos, alineaciones o consultas específicas sobre corpus comparables, por citar algunas de ellas.

Estas deficiencias se acentúan en el caso de que se utilicen idiomas distintos del inglés, ya que la preponderancia de la lengua inglesa sobre el resto incide en el desarrollo de software monolingüe, generalmente construido sobre y para la lengua inglesa. Es frecuente que exista software para el tratamiento de lenguas naturales que tengan su versión en inglés mucho más desarrollada que en el resto de idiomas, o que directamente utilizan la versión diseñada para la lengua inglesa en el resto de lenguas, por ejemplo, el tokenizer por palabras de la librería NLTK (NLTK Project, 2017), diseñado para lengua inglesa y utilizado indistintamente en el resto de idiomas.

Así pues, la hipótesis de trabajo se puede resumir en los siguientes puntos:

(1) Falta de usabilidad y utilidad de las aplicaciones existentes para la creación y tratamiento de distintos corpus lingüísticos con distintas capas de anotación sin supervisión por parte de un especialista técnico.

(2) Falta de formación en el campo de la programación por parte de los lingüistas.

(3) Falta de unicidad de herramientas/aplicaciones de trabajo que funcionen al mismo nivel para el inglés y las demás lenguas, en este caso el español.

Por estos puntos, existe la necesidad de construir *ACTRES Corpus Manager*, es decir, un framework para el tratamiento de corpus lingüísticos que posibilite la construcción de corpus lingüísticos monolingües, bi-/multilingües paralelos y comparables con distintas capas de anotación lingüística sin necesidad de intervención de personal técnico especializado en ninguna fase del proceso.

Para que *ACTRES Corpus Manager* cumpla con la hipótesis de trabajo es indispensable:

- Comprender y delimitar los distintos procesos y tareas, tanto a bajo como a alto nivel, necesarios para la incorporación de un corpus lingüístico en un software específico: formato de datos manejados, flujos de entrada y salida y orden de ejecución entre otros.
- Identificar qué actividades son críticas en cuanto a la necesidad de disposición de conocimientos avanzados de programación o tratamiento de software específico, por ejemplo, etiquetadores lingüísticos o alineación. Para posteriormente, evaluar la autonomía que concede cada una de estas actividades al usuario lingüista.
- Conocer en detalle el tipo de software para el tratamiento de corpus lingüísticos utilizado en el pasado y en el presente. El análisis de software no consiste únicamente en frameworks completos, sino en cualquier aplicación o herramienta cuya acción sea necesaria para incorporar un corpus a un sistema de concordancias.

- Averiguar qué software es el más utilizado y apto en cada uno de los procesos en la actualidad. Haciendo especial hincapié en descubrir cuáles son sus debilidades y fortalezas para si fuera necesario, desarrollar un software propio que mejore el existente.

ACTRES Corpus Manager facilitará la creación de cualquier tipo de corpus lingüístico por medio de un flujo de ejecución controlado e independiente de las habilidades técnicas del usuario. Es decir, el software se plantea desde el punto de vista de la usabilidad, el usuario selecciona los documentos y características de sus corpus, y *ACTRES Corpus Manager*, por medio de un flujo de ejecución secuencial, va transformando los documentos hasta convertirlos en un corpus lingüístico indexado sobre el que realizar consultas y extraer características. Todos los procesos efectuados sobre el flujo de ejecución controlado serán realizados a través de componentes aislados o módulos, cuya principal característica es su autonomía funcional, es decir, que no necesitan de la supervisión de un usuario o técnico durante su ejecución.

Para cada componente, se seleccionará el software propio o de terceros más adecuado para llevar a cabo su función, incorporando como novedad un prototipo de etiquetador retórico temático limitado y un etiquetador semántico basado en los recursos léxicos de USAS (Piao, Bianchi, Dayrell, D'Egidio, & Rayson, 2015). Es necesario interconectar estos componentes adecuando el flujo de datos a cada uno de ellos.

La contrapartida de la automatización de los procesos es la tolerancia implícita de cierta tasa de error en algunos de los componentes (alineaciones y etiquetados principalmente). No obstante, estos errores son los mismos que pueden aparecer en cualquier otro framework en el que la participación de un especialista técnico sea obligatoria, salvo que el proceso se realizara de forma manual, algo que consumiría una ingente cantidad de tiempo en corpus de tamaño medio. Por ello se asume que no se puede evitar cierta tasa de errores que varía dependiendo de múltiples factores

(precisión del etiquetado gramatical para cada idioma, tipo de traducción e idioma en corpus paralelos, etc.).

Gracias a *ACTRES Corpus Manager*, el usuario tampoco necesitará emplear múltiples herramientas software para poder cubrir las carencias que presentan en otros frameworks, evitando por tanto conocer las distintas opciones disponibles, así como sus ventajas e inconvenientes, su modo de empleo, etc. Se evita la problemática derivada de esta situación, frecuentemente abordada en la literatura, como por ejemplo (Lu, 2014) o (Weisser, 2016).

Para comprobar la validez y eficacia de *ACTRES Corpus Manager* se desarrollarán casos de uso que emulen con exactitud la problemática actual existente en la creación de corpus lingüísticos monolingües o bi-/multilingües paralelos y/o comparables con distintas capas de anotación desde el punto de vista del usuario lingüista y se realizará un cuestionario de usabilidad.

La aplicabilidad práctica de *ACTRES Corpus Manager* justifica su desarrollo, ya que su plena operatividad posibilitará que:

- Los lingüistas que utilizan corpus lingüísticos en sus investigaciones puedan emplear, sin asistencia técnica y de forma independiente, *ACTRES Corpus Manager* para crear corpus lingüísticos anotados para validar o refutar hipótesis de investigación, como fuente para construir recursos léxicos (por ejemplo, diccionarios terminológicos), etc.
- *ACTRES Corpus Manager* tendrá incidencia directa en el rango de hipótesis de investigación que se podrían resolver utilizando un único software. Por ejemplo, no será necesario emplear un software para realizar concordancias y otro diferente para realizar un etiquetado semántico.
- Beneficiará a los técnicos relacionados con la disciplina de NLP, ya que *ACTRES Corpus Manager* permitirá un incremento del número de corpus lingüísticos creados, que pueden alimentar aplicaciones de NLP, como por ejemplo *Machine Translation*.

- Ahorrará costes, ya que la creación de corpus lingüísticos consumirá menos recursos humanos y técnicos.
- El grado de autonomía alcanzado favorece la creación de corpus lingüísticos anotados de tamaño medio-grande por parte de cualquier usuario lingüista sin considerar sus habilidades técnicas o experiencia previa.

Así pues, esta tesis doctoral viene a cubrir in vacío importante que es la carencia de un software integral que permita a un usuario lingüista sin conocimientos de programación crear y consultar sus propios corpus lingüísticos monolingües y/o bi-/multilingües paralelos y/o comparables con distintas capas de anotación, independientemente del idioma de los corpus lingüísticos[37] y sin asistencia técnica en ningún momento del proceso.

[37] Las pruebas serán realizadas empleando lenguas indoeuropeas de las ramas románica y germánica. Es de suponer que si el software seleccionado encargado de la gestión de corpus admite lenguas como chino, hebreo o árabe *ACTRES Corpus Manager* también lo hará.

Capítulo

5

Metodología

5.1 Introducción

En este capítulo se explican los distintos pasos que componen la metodología seguida, que no se adscribe a ningún ciclo de vida de desarrollo software al uso ya que el grupo de trabajo está formado por un solo individuo, ni tampoco se trata de un desarrollo basado en tareas. No obstante, el ciclo de desarrollo se inspira en un desarrollo rápido de aplicaciones o *RAD* (Martin, 1991) o *Extreme programming* (Maurer & Martel, 2002), salvando las distancias, ya que muchos requisitos funcionales han surgido sobre la marcha al carecer de software o funciones específicas para realizar distintas tareas.

Las fases metodológicas principales son:

- Análisis.
- Diseño.
- Implementación.
- Validación.

Conviene aclarar que la ingeniería del software para aplicaciones web, como es el caso de *ACTRES Corpus Manager*, se diferencia de la esencia clásica de desarrollo

de la ingeniería del software tradicional en tres aspectos (Sommerville, 2011, pág. 13):

(1) La reutilización de componentes como estrategia dominante.

(2) Preponderancia del diseño incremental, es decir, no es posible establecer todos los requisitos por adelantado.

(3) La dependencia del diseño de los interfaces depende de las tecnologías facilitadas por los navegadores web.

A continuación, se describirá en detalle cada una de las fases seguidas en la metodología.

5.2 Fase de análisis

El análisis del sistema consiste en establecer los requisitos que éste debe satisfacer y de los casos de uso a los que debe hacer frente. El objetivo es evitar que en la fase de desarrollo surjan problemas que disparen los tiempos de desarrollo de las distintas tareas.

(1) Toma de requisitos:

La toma de requisitos consiste en establecer una definición precisa y completa del comportamiento del software, y que además pueda servir como base de su desarrollo (Stellman & Greene, 2005, pág. 98). Es decir, los requisitos definen las funcionalidades que el software puede realizar y las que no. Los requisitos provienen tanto de las necesidades reveladas en el Capítulo 4: "Hipótesis de trabajo: necesidades, objetivos y nicho", como de las opiniones recabadas por los usuarios potenciales de la aplicación.

(2) Especificación de casos de uso:

Los casos de uso identifican a los actores involucrados en la interacción e ilustran cuál es el tipo de interacción (Sommerville, 2011, pág. 107). En otras palabras, el caso de uso describe la interacción que el usuario tiene con el software para llevar

a cabo una tarea concreta. Los casos de uso ayudan a modelar el comportamiento del software y se basan en los requisitos establecidos en el paso anterior.

5.3 Fase de diseño

Durante el proceso de diseño del framework se identifican los componentes internos que forman parte del sistema, sus interacciones y las actividades de las que se hacen cargo. El diseño se centra en los aspectos técnicos para llevar a cabo los requisitos funcionales definidos en el paso anterior.

(1) Diseño de la arquitectura

El diseño de la arquitectura implica la creación de los siguientes diagramas:

- **Diagrama de componentes**: Un componente es una unidad independiente de software que puede estar formada a su vez por otros componentes para crear un sistema software (Sommerville, 2011, pág. 455). En el diagrama de componentes del sistema se muestran los distintos elementos que forman el framework, junto a las tecnologías que los sustentan.
- **Diagrama de despliegue**: El diagrama de despliegue muestra como los componentes del software son físicamente ubicados en el hardware (Sommerville, 2011, pág. 197) . Por ejemplo, en una aplicación web la ubicación de los componentes se reparte entre el cliente y el servidor o servidores.
- **Diagrama de clases global**: permite mostrar las clases de sistema y la relación entre las mismas cuando el paradigma de programación empleado está orientado a objetos (Sommerville, 2011, pág. 129). En otras palabras, describe los elementos encargados de gestionar parte de la lógica de *ACTRES Corpus Manager*. Se realiza un diseño preliminar de las clases, que en el futuro albergarán parte de la lógica de *ACTRES Corpus Manager*, encargada de ejecutar las distintas operaciones que llevan a cabo los componentes.

(2) Creación de diagramas de actividad

Estos diagramas pretenden mostrar las actividades que conforman un proceso del sistema y el flujo de control de una actividad a otra (Sommerville, 2011, pág. 123). Permiten diseñar el flujo de trabajo de *ACTRES Corpus Manager*, describiendo las distintas operaciones internas que se llevan a cabo para ejecutar las funcionalidades para las que el framework ha sido diseñado.

(3) Diseño del modelo de datos

El diseño del modelo de datos muestra el flujo de información del sistema. Es decir, muestra los distintos flujos de datos de *ACTRES Corpus Manager*, indicando el formato y tipo de datos utilizado por cada componente u operación que se considere significativa y que, por tanto, deba ser destacada.

(4) Diseño del interfaz de usuario

En esta fase se realiza un boceto del interfaz de *ACTRES Corpus Manager*, poniendo especial énfasis en las condiciones establecidas en la toma de requisitos, diseñando las vistas necesarias para que el usuario interactúe con el sistema con el fin de llevar a cabo las funcionalidades para las que ha sido diseñado. El diseño del interfaz es distinto del concepto de *special-purpose user interface* utilizado en la ingeniería del software tradicional (Sommerville, 2011, pág. 13).

5.3 Fase de implementación

La implementación del framework consiste en el desarrollo efectivo del software de acuerdo con el diseño realizado en la fase anterior, para ello plasma cada uno de los planteamientos establecidos en el diseño del framework. Aunque en teoría la implementación se ha de corresponder fielmente con el diseño, pueden producirse cambios procedentes de incompatibilidades software, falta de documentación de herramientas, bajo rendimiento de planteamientos seleccionados, etc.

Esta fase consiste en:

(1) Elección de lenguajes de programación

El hecho de seleccionar un lenguaje de programación u otro para desarrollar *ACTRES Corpus Manager* depende de:

i. La utilización de recursos software de terceros: estos recursos se encuentran compilados, en formato ejecutable o bien son scripts en un lenguaje de programación concreto.
ii. La idoneidad de cada lenguaje de programación para cada tarea concreta. Por ejemplo, R (R Core Team, 2017) es un lenguaje de programación empleado en operaciones estadísticas.
iii. La elección del modelo de arquitectura y la consiguiente ubicación de los componentes en las distintas partes que la conforman limitan la elección de los lenguajes de programación disponibles. Por ejemplo, PHP (The PHP Group, 2017) es un lenguaje de programación del lado del servidor, mientras Javascript (Ecma International, 2011) suele ser utilizado en el cliente.

(2) Software y recursos utilizados

Es necesario seleccionar el software más apropiado de acuerdo con las operaciones a realizar por cada uno de los componentes que forman parte de *ACTRES Corpus Manager*.

Este software incluye:

- Librerías, toolkits y *suites* que proporcionan facilidades o recursos para implementar acciones concretas. Ya sea para las acciones que se llevan a cabo directamente en los distintos componentes de *ACTRES Corpus Manager* o para su interconexión.

- Recursos software de terceros encargados de realizar alguna acción concreta en el framework. Por ejemplo, un software de terceros es utilizado para realizar el etiquetado gramatical.

(3) Desarrollo efectivo del software

En esta fase se desarrollará *ACTRES Corpus Manager* utilizando los recursos software seleccionados en la fase anterior.

5.4 Fase de validación

La última fase de la metodología a seguir es la validación de *ACTRES Corpus Manager*. El software ha de ser probado con el objetivo de comprobar que funciona correctamente. Estas pruebas son una verificación empírica que demuestran que el framework cumple con los requisitos establecidos en la fase de diseño y desarrollados en la fase de implementación.

Para evaluar si *ACTRES Corpus Manager* cumple con los requisitos establecidos se llevarán a cabo dos procesos:

(1) **Prueba conceptual**: En esta tesis el término prueba conceptual es empleado para designar a la fase interna de *testeo* y *debug* de las funcionalidades del framework. Para ello se crearán y consultarán todos y cada uno de los tipos de corpus lingüísticos soportados por el framework. Para realizar esta tarea se utilizarán recursos lingüísticos procedentes del grupo de investigación ACTRES, corpus propios formados por informes emitidos por el Banco de España y el corpus EUROPARL (Koehn, 2005).

(2) **Pruebas de aceptación y usabilidad del software**: 6 personas[38] probarán el software y realizarán un cuestionario tipo SUS (del inglés *System Usability Scale* - escala de usabilidad del sistema en español) (Brooke,

[38] Investigadores pertenecientes al grupo de investigación ACTRES procedentes de la Universidad de León, Universidad de Valladolid, Universidad de Cantabria y Universidad del País Vasco.

1996), con el objetivo de evaluar la usabilidad de *ACTRES Corpus Manager*.

Capítulo

6

Análisis, diseño e implementación de *ACTRES Corpus Manager*

En este capítulo se describe el proceso de análisis, diseño e implementación del software *que cubre las necesidades, objetivos y nichos descritos en el Capítulo 4: "Hipótesis de trabajo: necesidades, objetivos y nicho". Durante el capítulo se mostrarán una serie de diagramas (diagrama de componentes, diagrama de despliegue, diagrama de clases simplificado y diagrama de actividad) y se presentará diversa información adicional relativa al diseño del software, con el objetivo de describir ACTRES Corpus Manager de la forma precisa posible.*

En una tesis doctoral no es habitual mostrar toda esta información, no obstante, se ha considerado conveniente presentarla como resultado de la esencia multidisciplinar de la investigación realizada. De este modo se puede facilitar la lectura por parte de los expertos de todas las especialidades afectadas, además de facilitar el proceso de replicación.

6.1 Análisis

El análisis del framework consiste en dos fases: (1) toma de requisitos y (2) especificación de casos de uso.

6.1.1 Toma de requisitos

De acuerdo con (Anthony, 2013a, pág. 156), la aplicación para el tratamiento de corpus lingüísticos ideal ha de ser un framework con un alto nivel de usabilidad, útil para que:

- los investigadores que utilizan una metodología basada en corpus, *corpus-based approach*, en inglés, que necesitan el acceso a funciones estadísticas y anotaciones;
- los investigadores cuyo estudios se basen en corpus, *corpus-driven approach*, en inglés, que requieren menos funcionalidades estadísticas y más acceso al corpus sin anotaciones y con concordancias en formato KWIC,
- las labores pedagógicas de los profesores y alumnos, que necesitan un sencillo acceso y empleo de las funcionalidades básicas de estos programas, como las concordancias o la posibilidad de exportar los resultados.

Analizando estas características descritas por Anthony se diferencian claramente los sectores potenciales de aplicabilidad del software que son, por un lado, cubrir las necesidades de los investigadores que emplean corpus lingüísticos en sus estudios y, por otro, servir de recurso tecnológico que proporcione soporte en las labores docentes relacionadas con el análisis de corpus lingüísticos.

Por este motivo, la toma de requisitos se ha basado principalmente en las opiniones y valoraciones de los usuarios presumibles de estos dos sectores: investigadores y docentes. El entorno académico de investigación y docencia intrínseco a la realización de la tesis doctoral en un grupo de investigación universitario resultó ideal para llevar a cabo esta toma de requisitos.

Otra fuente de información ha sido la experiencia directa recopilada de la utilización durante 9 años del corpus P-ACTRES (Izquierdo, Hofland, & Reigem, 2008) y su posterior evolución P-ACTRES 2.0 (Sanjurjo-González & Izquierdo, próximamente) por parte de los miembros del grupo de investigación. P-ACTRES es propiedad del grupo de investigación ACTRES (ACTRES, 2017c), al cual pertenece el autor de esta tesis doctoral, Hugo Sanjurjo González. P-ACTRES fue creado fruto de la colaboración entre la Dra. M. Izquierdo como lingüista, y K. Hofland y Ø. Reigem como desarrolladores informáticos. Se trata de un corpus de consulta bidireccional inglés - español que permite llevar a cabo estudios contrastivos o de traducción a través de consultas sobre corpus paralelos (textos originales en inglés y sus correspondientes traducciones en español; textos originales en español y sus correspondientes traducciones en inglés) y comparables (textos originales en inglés y textos traducidos en inglés; textos originales en español y textos traducidos en español).

P-ACTRES 2.0 ha servido como banco de pruebas sobre el que extraer debilidades técnicas y deficiencias de usabilidad consideradas críticas desde el punto de vista de cualquier usuario lingüista.

A continuación, se describen todos los requisitos recogidos para llevar a cabo la construcción de *ACTRES Corpus Manager*.

6.1.1.1 Requisitos funcionales

Identificación del requisito:	RF01
Nombre del requisito:	Acceso a través de internet.
Descripción del requisito:	El uso de *ACTRES Corpus Manager* se realizará a través de internet por medio de cualquier navegador web.

Tabla 17 - Requisito funcional nº1

Identificación del requisito:	RF02
Nombre del requisito:	Autentificación de usuario.
Características:	Los usuarios deberán identificarse para acceder a cualquier funcionalidad del sistema.

Tabla 18 - Requisito funcional nº2

Identificación del requisito:	RF03
Nombre del requisito:	Compilación de corpus sin intervención técnica.
Características:	Los usuarios han de ser capaces de crear y analizar sus corpus sin requerir asistencia técnica en ningún momento.

Tabla 19 - Requisito funcional nº3

Identificación del requisito:	RF04
Nombre del requisito:	Soporte de creación de corpus de tamaño pequeño a medio-grande.
Características:	La aplicación ha de soportar la consulta y análisis de corpus de tamaño pequeño o medio-grande.

Tabla 20 - Requisito funcional nº4

Identificación del requisito:	RF05
Nombre del requisito:	Corpus multilingües.
Características:	La aplicación ha de soportar la creación y análisis de corpus en varios idiomas.

Tabla 21 - Requisito funcional nº5

Identificación del requisito:	RF06
Nombre del requisito:	Corpus monolingües.
Características:	La aplicación ha de soportar la creación y análisis de corpus monolingües.

Tabla 22 - Requisito funcional nº6

Identificación del requisito:	RF07
Nombre del requisito:	Corpus paralelos bi-/multilingües.
Características:	La aplicación ha de soportar la creación de corpus paralelos formados por una o más traducciones.

Tabla 23 - Requisito funcional nº7

Identificación del requisito:	RF08
Nombre del requisito:	Corpus comparables.
Características:	La aplicación ha de soportar la creación de corpus comparables formado por dos o más subcorpus en distintos o en el mismo idioma.

Tabla 24 - Requisito funcional nº8

Identificación del requisito:	RF09
Nombre del requisito:	Etiquetado gramatical.
Características:	La aplicación incorporará un etiquetador gramatical no supervisado.

Tabla 25 - Requisito funcional nº9

Identificación del requisito:	RF10
Nombre del requisito:	Etiquetado semántico
Características:	La aplicación incorporará un etiquetador semántico no supervisado.

Tabla 26 - Requisito funcional nº10

Identificación del requisito:	RF11
Nombre del requisito:	Etiquetado retórico.
Características:	La aplicación incorporará un etiquetador retórico. Ante la ausencia de etiquetadores retóricos no supervisados y la complejidad computacional y lingüística para desarrollar uno nuevo, el etiquetado retórico será supervisado.

Tabla 27 - Requisito funcional nº11

Identificación del requisito:	RF12
Nombre del requisito:	Estadísticas.
Características:	La aplicación permitirá extraer funciones estadísticas cuantitativas y cualitativas de los corpus. Más concretamente: (1) Colocaciones. (2) Listas de palabras clave. (3) N-gramas. (4) Lista de frecuencias. (5) Listas de frecuencias de cada consulta.

Tabla 28 - Requisito funcional nº12

Identificación del requisito:	RF13
Nombre del requisito:	Búsquedas o concordancias
Características:	La aplicación permitirá realizar búsquedas sobre los corpus con las siguientes características: (1) Formato KWIC. (2) Opciones de búsqueda personalizadas. (3) Búsquedas basadas en expresiones regulares. (4) Búsquedas complejas asistidas sin necesidad de disponer de conocimientos sobre expresiones regulares.

Tabla 29 - Requisito funcional nº13

Identificación del requisito	RF14
Nombre del requisito:	Diseño de interfaz responsive.
Características:	El interfaz de usuario de *ACTRES Corpus Manager* ha de mostrar la información de manera clara y precisa, ser accesible, adaptarse al tamaño de las pantallas de los distintos dispositivos desde los que se pueda acceder y ser compatible con la mayoría de los navegadores web.

Tabla 30 - Requisito funcional nº14

6.1.1.2 Requisitos no funcionales

Identificación del requisito:	RNF01
Nombre del requisito:	Información.
Descripción del requisito:	*ACTRES Corpus Manager* facilitará información sobre los componentes software que utiliza, los *tagsets* de los etiquetados y los distintos valores estadísticos empleados.

Tabla 31 - Requisito no funcional nº1

Identificación del requisito:	RNF02
Nombre del requisito:	Integridad de datos.
Características:	*ACTRES Corpus Manager* asegura la integridad y no difusión de los corpus alojados en el sistema a terceros.

Tabla 32 - Requisito no funcional nº2

Identificación del requisito:	RNF03
Nombre del requisito:	Integración en web propia.
Características:	*ACTRES Corpus Manager* ha de integrarse como un recurso dentro de la web corporativa del grupo de investigación ACTRES.

Tabla 33 - Requisito no funcional nº3

6.1.1.3 Otros requisitos

- Requisitos de rendimiento:
    - El número de usuarios conectados no será elevado puesto que en principio sólo será accesible para los miembros del grupo de investigación al que pertenece el doctorando (24 personas).

  - La creación de corpus paralelos es el proceso más crítico. Por este motivo se limita a 4 el número total de subcorpus (original + traducción), de esta forma se garantizan unos rendimientos de tiempo aceptables en el servidor de pruebas utilizado.

- Requisitos de seguridad:
  - El acceso al sistema se realiza mediante autentificación con un sistema de encriptado.
  - Registro de actividad mediante archivos log.
- Requisitos de mantenibilidad:
  - Es necesario que un administrador de sistemas supervise las incidencias derivadas de los errores surgidos durante los procesos de creación y/o análisis de corpus, si es que los hubiera.
- Requisitos de portabilidad:
  - Es estrictamente necesario el empleo de un servidor web Linux.
- Requisitos de fiabilidad y disponibilidad:
  - No se han establecido requisitos específicos de fiabilidad y disponibilidad.

6.1.1.4 Implicaciones software de los requisitos establecidos

A continuación, se muestra una tabla de las implicaciones software y de los inconvenientes asociados que tiene el cumplimiento de los requisitos funcionales establecidos en el diseño.

Requisito funcional	**Implicaciones software**	**Inconvenientes**
RF01 - Acceso a través de internet	Framework para el tratamiento de corpus lingüísticos de cuarta generación desarrollado a través del paradigma cliente-servidor.	Medidas de seguridad para evitar el acceso de intrusos.
RF02 Autentificación de usuario	Base de datos para gestión de usuarios.	Encriptación de datos.

RF03 Compilación de corpus sin intervención técnica	Pre-tratamiento mínimo de los textos del corpus .	Asunción y tolerancia de errores derivados del automatismo.
RF04 Soporte de creación de corpus de tamaño medio-grande	Incorporación de sistema de indexación.	Proceso de creación de corpus computacionalmente más complejo.
RF05 Corpus multilingües	Establecimiento de recursos lingüísticos en diversos idiomas. Ej. tokenizers específicos para cada idioma.	Utilización de librerías y recursos software externos especializados en NLP.
RF06 Corpus monolingües	Tipo de corpus admitido por defecto en cualquier sistema de gestión de corpus.	Ninguno.
RF07 Corpus paralelos bi-/multilingües	Incorporación de funcionalidad para alineado no supervisado. Consultas sobre corpus y sus alineaciones.	Proceso de creación de corpus paralelos computacionalmente más complejo. Asunción y tolerancia de errores derivados del automatismo.
RF08 Corpus comparables	Concurrencia de consultas sobre múltiples corpus considerados comparables	Establecimiento de un diseño para mostrar los resultados de las múltiples consultas sobre los corpus.
RF09 Etiquetado gramatical	Incorporación de funcionalidad para etiquetado gramatical no supervisado.	Asunción y tolerancia de errores derivados del automatismo.
RF10 Etiquetado semántico	Incorporación de funcionalidad para etiquetado semántico no supervisado.	Asunción y tolerancia de errores derivados del automatismo.
RF11 Etiquetado retórico	Creación de etiquetador retórico supervisado.	Se trata de una anotación lingüística novedosa cuya implementación efectiva implicará un flujo de creación de corpus diferenciado del resto de

		corpus a pesar de que en realidad se trata únicamente de una anotación lingüística.[39]
RF12 Estadísticas	- Incorporación de funcionalidades estadísticas basadas en frecuencias. - Incorporación de funcionalidades estadísticas basadas en tests de significación.	Incorporación de librerías externas especializadas en estadística.
RF13 Búsquedas	- Búsquedas basadas en contexto. - Implementación de distintos tipos de búsqueda de acuerdo con las preferencias de búsqueda seleccionadas. - Interfaz de consulta que lleve a cabo una búsqueda por expresiones regulares de forma asistida.	Creación de interfaz intuitiva y semi-asistida para la ejecución de consultas complejas.
RF14 Diseño de interfaz responsive	Utilización de framework de desarrollo web.	Ninguno.

Tabla 34 - Requisitos funcionales y su repercusión en el diseño del software

6.1.2 Especificación de casos de uso

De acuerdo con la toma de requisitos, la especificación de los casos de uso se basa en seis escenarios fundamentales: (1) crear un corpus monolingüe, (2) crear un corpus paralelo bi-/multilingüe, (3) crear un corpus comparable, (4) crear un corpus

[39] La diferenciación de la creación de corpus retórico como un tipo adicional de corpus se explicará en detalle en la página 177, dentro del apartado 6.3.3 Limitaciones de la implementación, y se mantendrá durante la descripción del resto de elementos del diseño del framework.

retórico, (5) consulta de un corpus de cualquier tipo y extracción de estadísticas de la consulta y (6) extraer estadísticas de un corpus.

Como consecuencia de su extensión, los casos de uso se describen en el Anexo 1.

6.2 Diseño

El principal desafío para trazar el diseño ha sido el de conseguir la mayor cuota de autonomía por parte del especialista lingüístico durante el proceso de compilación y análisis de corpus de cualquier tipo: monolingües, bi-/multilingües paralelos, comparables, incorporando o no distintos niveles de etiquetados lingüísticos a nivel de palabra, así como etiquetados estructurales.

La estrategia seleccionada para diseñar *ACTRES Corpus Manager* se centra en el usuario lingüista, alejándose de estrategias orientadas a la gestión por parte de un especialista informático, como ocurre por ejemplo en CQPweb (Hardie, 2012). El usuario se convierte en el actor principal al igual que sucede en (Wilson, Hartley, Sharoff, & Stephenson, 2010). A través del interfaz se logra ocultar toda la complejidad asociada a los procedimientos técnicos empleados para lograr la nula intervención por parte de un especialista informático. De este modo se logra que la aplicación sea accesible a un mayor número de usuarios lingüistas, no sólo a aquellos con experiencia en el uso de aplicaciones para el tratamiento de corpus lingüísticos. Como contrapartida, se asumen ciertos errores en las tareas de alineado y en los etiquetados sin supervisión, además de cierta carga de trabajo por parte del usuario relacionada con la incorporación del etiquetado retórico supervisado.

6.2.1 Diseño de la arquitectura

El diseño de *ACTRES Corpus Manager* se inspiró inicialmente en un patrón de diseño modelo-vista-controlador (Reenskaug, 1979), MVC de aquí en adelante. Este diseño (Figura 18) se caracteriza porque:

- el modelo es el componente principal de la aplicación, puesto que posee la lógica de la misma;
- la vista es el interfaz de la aplicación, que el usuario emplea para interactuar con *ACTRES Corpus Manager* e introducir datos;
- por último, el controlador se encarga de gestionar las peticiones que recibe tanto del controlador, y en algunas configuraciones incluso del modelo.

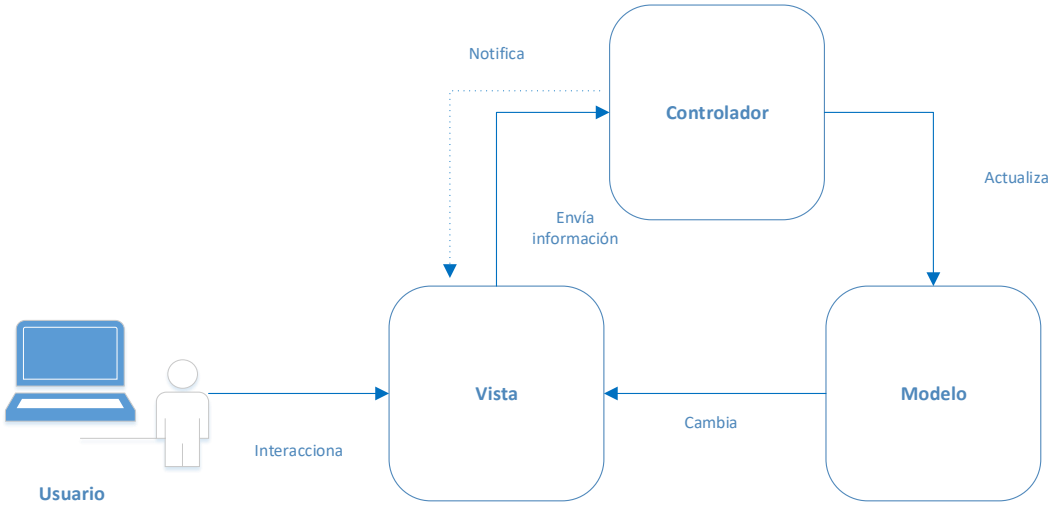


Figura 18 - Estrategia tradicional del MVC

El empleo del MVC permite una separación de componentes que facilita la reutilización y modificación de los mismos sin tener que reformar la aplicación en su conjunto. También facilita un desarrollo rápido de aplicaciones, al dividir los componentes un grupo de trabajo específico puede dedicarse a cada una de las partes. Por último, facilita la ejecución de pruebas en los distintos componentes de forma individualizada.

En la actualidad el MVC se emplea con asiduidad en aplicaciones web, sin embargo, está implementado de forma diferente a la idea original:

- En algunos frameworks de desarrollo web, por ejemplo CakePHP (Cake Software Foundation, Inc, 2017), toda comunicación entre la vista y el

modelo ha de pasar por un controlador, algo que no sucede en la idea original (Hopkins, 2013).

- Tal y como recoge (Fowler, 2006), las partes definidas en el paradigma clásico de MVC no tienen sentido en un MVC web actual ya que en realidad se encuentran diseminadas en distintos componentes. Por ejemplo, el modelo está difuminado en distintos componentes, tanto del lado del servidor, por ejemplo, PHP, como del cliente, por ejemplo, Javascript.
- La lógica de la vista es en ocasiones muy compleja para ser combinada con la lógica del modelo, algo que repercute en la separación de componentes por la que aboga el diseño.

Por estos motivos, el patrón de diseño MVC se combinó con un patrón de diseño denominado Modelo-Vista-VistaModelo (Gossman, 2005), donde la lógica de la vista es independiente del resto de la lógica del software. En este patrón de diseño la lógica de la vista incluye las interacciones entre el usuario y los distintos elementos de la vista (botones, formularios, entradas de texto, etc.) que no requieren acceso al modelo del software. Este tipo de vista se denomina vista activa, en contraposición a la vista pasiva del MVC tradicional.

Al no utilizarse el componente VistaModelo, sino un controlador al uso, similar el definido en el MVC, no se puede hablar de que se emplea un diseño de patrón Modelo-Vista-VistaModelo.

El desarrollo efectivo de la aplicación no ha empleado ningún "framework de desarrollo" que empleé de forma estricta el paradigma MVC adaptado al desarrollo web, como Spring (Pivotal Software, 2017), Struts (The Apache Software Foundation, 2016), AngularJS (Google Inc., 2017a) o Symfony (SensioLabs, 2017). En su lugar, se ha optado por combinar la acción de AJAX y formularios HTTP con PHP. De este modo la programación fue mucho más flexible y rápida.

El patrón de diseño empleado es combinado a su vez con un modelo de arquitectura de programación en tres capas típica de las aplicaciones web, que reconoce tres capas según su ubicación física en el en el cliente o el servidor: presentación, lógica y datos. Este modelo de arquitectura, además de la separación de componentes, aumenta la escalabilidad, el rendimiento y la disponibilidad de la aplicación gracias a la disposición de los distintos componentes entre el cliente y el servidor (uno o varios), aprovechando los recursos de procesamiento de cada uno de ellos.

Hay que destacar, que la estrategia tradicional del patrón de diseño MVC no se ocupa de separar la capa de datos de la lógica de la aplicación, ambas son incluidas en el modelo. Por otro lado, la capa de la presentación y su lógica asociada representan la vista activa descrita anteriormente.

En la Figura 19 se puede observar un diagrama de este diseño.

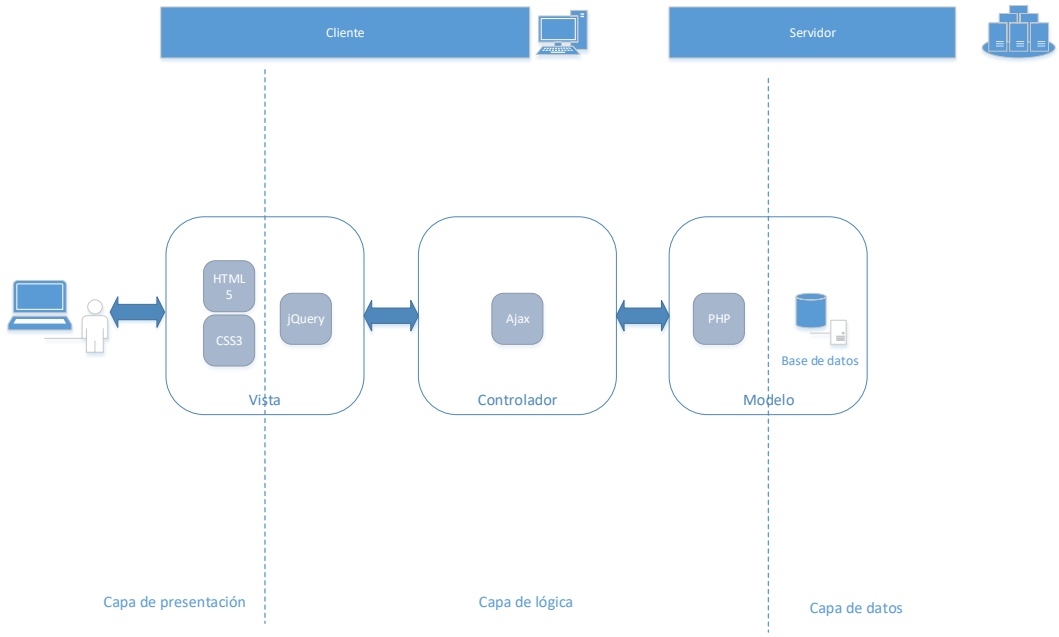


Figura 19 - MVC con vista activa y arquitectura de diseño del framework creado

La estrategia de diseño empleada es similar a la presentada en (Wang, Guo, & Song, 2009) y (Jacyntho, Schwabe, & Rossi, 2002) en cuanto a la combinación del patrón de diseño MVC y de la arquitectura multicapas.

6.2.1.1 Diagrama de componentes

En la Figura 20 se muestra el diagrama de componentes de *ACTRES Corpus Manager*, donde se pueden observar los distintos componentes (cuadrados pequeños), junto con la tecnología que lo sustenta (color amarillo – lenguaje de programación, o verde – software externo), así como el formato de datos utilizado en la comunicación entre componentes (color naranja). La actividad Crear corpus* es especificada para cada tipo de corpus en los diagramas de las figuras 21, 22 y 23. La forma de expresar estos diagramas, no está sujeta a ningún lenguaje de modelado concreto para facilitar su lectura por parte de todos los lectores potenciales de esta tesis doctoral.

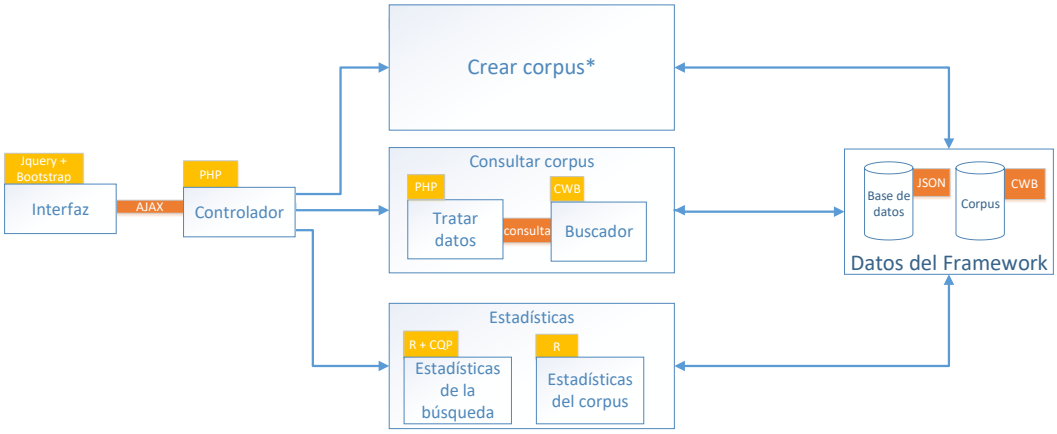


Figura 20 - Diagrama de componentes de *ACTRES Corpus Manager*

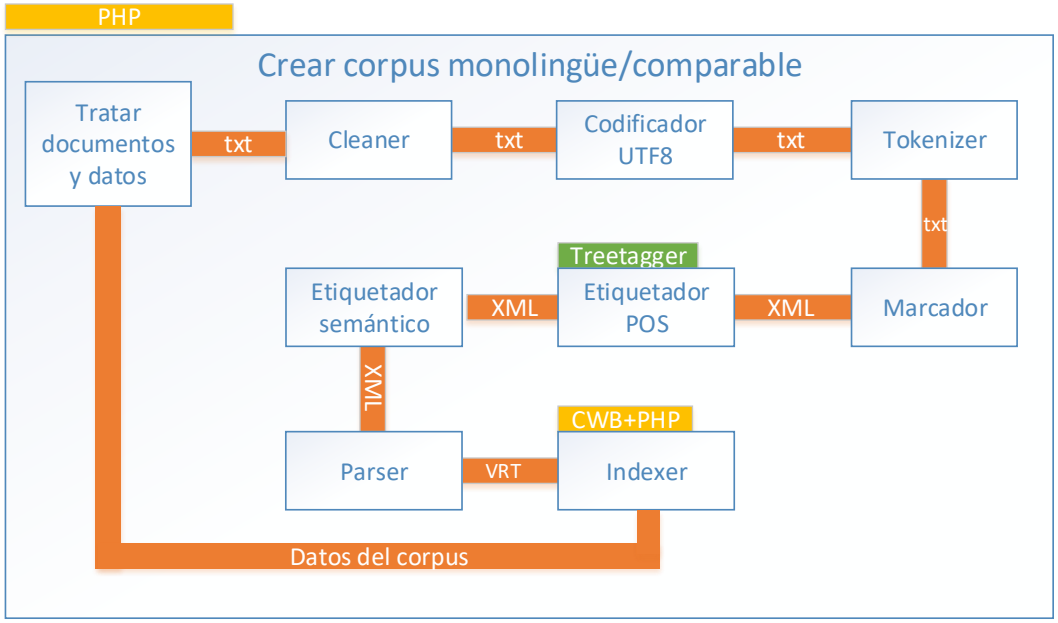


Figura 21 - Diagrama de componentes de *ACTRES Corpus Manager* para crear corpus monolingües o comparables

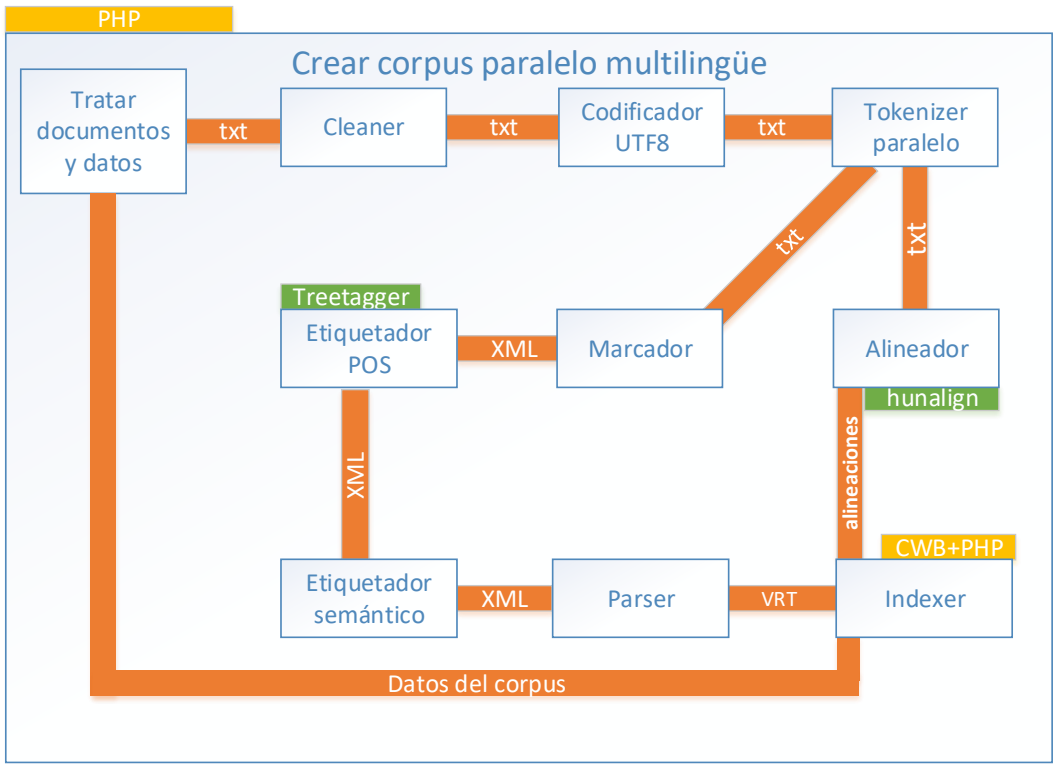


Figura 22 - Diagrama de componentes de *ACTRES Corpus Manager* para crear corpus bi-/multilingües paralelos

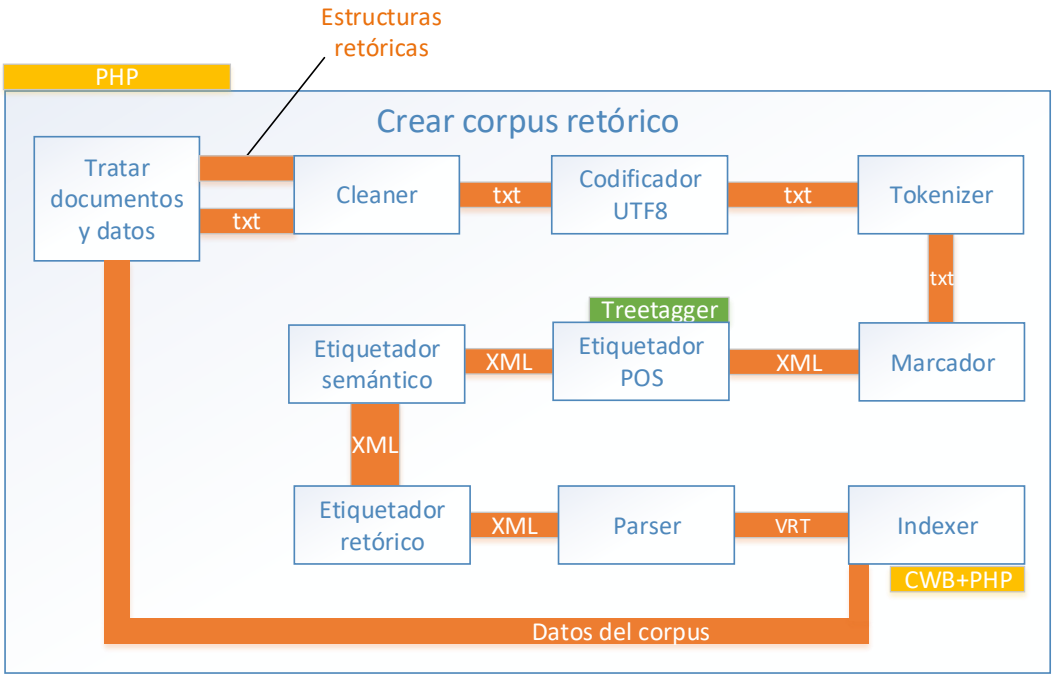


Figura 23 - Diagrama de componentes de *ACTRES Corpus Manager* para crear un corpus retórico

6.2.1.2 Diagrama de despliegue

El diagrama de despliegue se muestra en la Figura 24. Al tratarse de una aplicación web *ACTRES Corpus Manager* interactúa tanto con el cliente como con el servidor. El cliente es cualquier dispositivo con conexión a internet y un navegador web. Por otra parte, se utiliza un único servidor de tipo Linux.

Tal y como se describió anteriormente, el controlador se encuentra en el cliente y se encarga de interactuar con el modelo correspondiente, alojado íntegramente en el servidor. Por último, señalar que se emplea un único servidor para albergar tanto las bases de datos como la lógica de *ACTRES Corpus Manager* correspondiente al lado del servidor.

El diagrama de despliegue se ha diseñado empleando el lenguaje de modelado unificado UML 2.

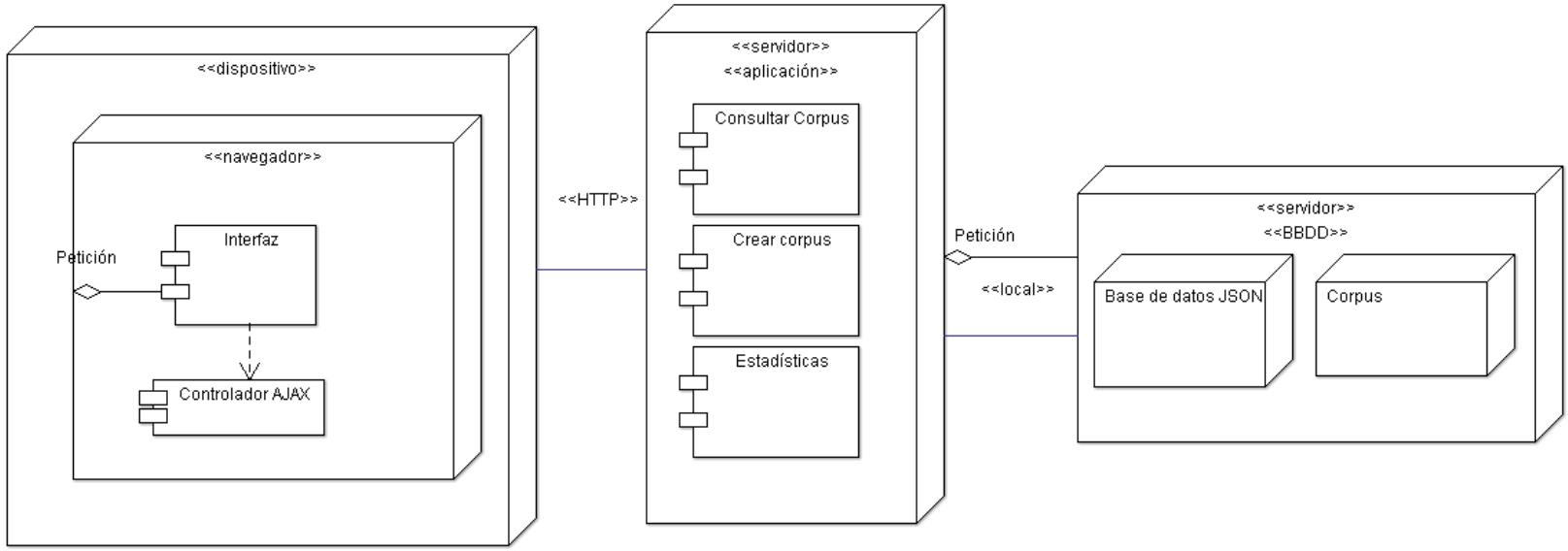


Figura 24 - Diagrama de despliegue de *ACTRES Corpus Manager*

6.2.1.3 Diagrama de clases simplificado

En las Figuras Figura 25, Figura 26, Figura 27 se muestran una serie de diagramas de clases simplificados. Para favorecer su lectura y comprensión se dividen en: (1) clases relacionadas con la creación de corpus, (2) clases relacionadas con la consulta de corpus, y (3) clases relacionadas con la extracción de estadísticas.

Dado que *ACTRES Corpus Manager* incorpora distintos recursos software cuya programación no está orientada a objetos, en los diagramas estos recursos aparecen como paquetes.

La explicación pormenorizada de las clases y los métodos se realizará en el subapartado 6.3.4.3.2 Lógica del framework, donde se mostrará un diagrama de clases completo.

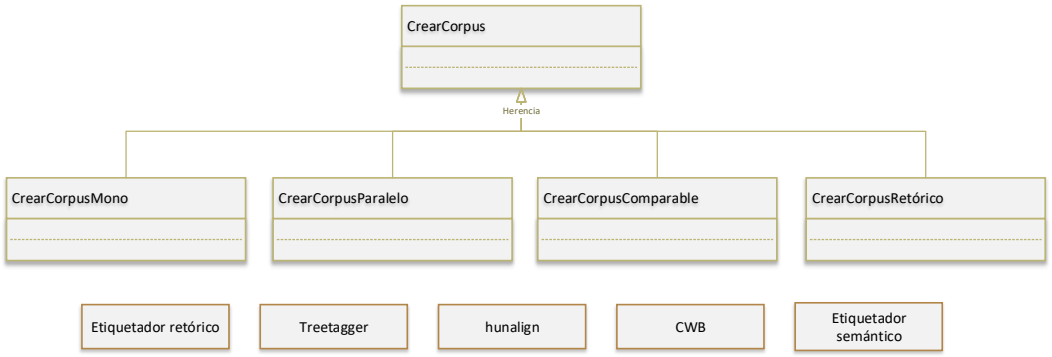


Figura 25 - Diagrama de clases simplificado relacionado con la creación de corpus

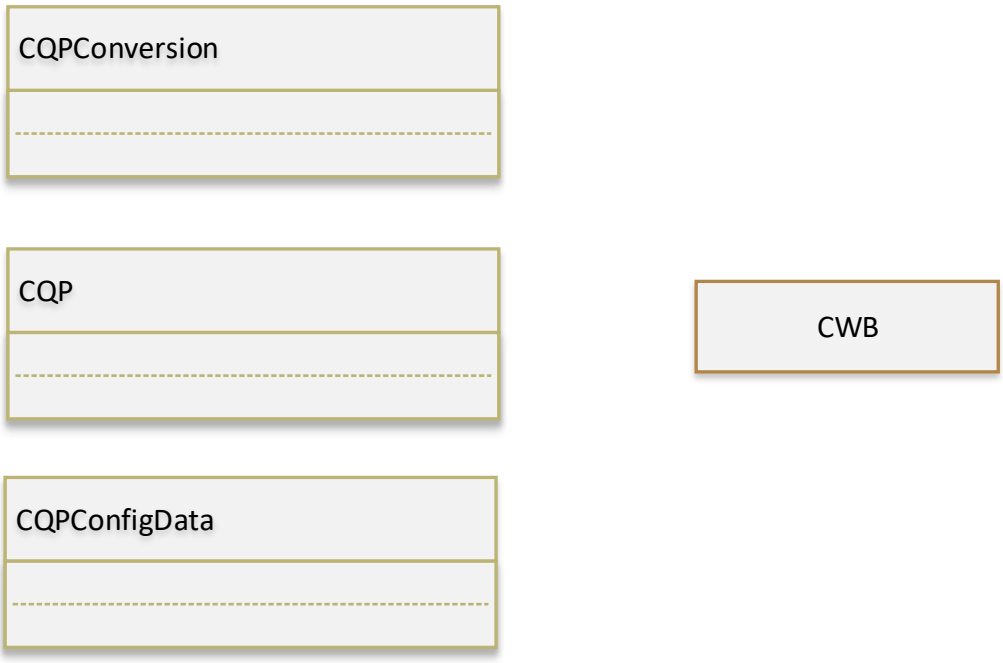


Figura 26 - Diagrama de clases simplificado relacionado con la consulta de corpus

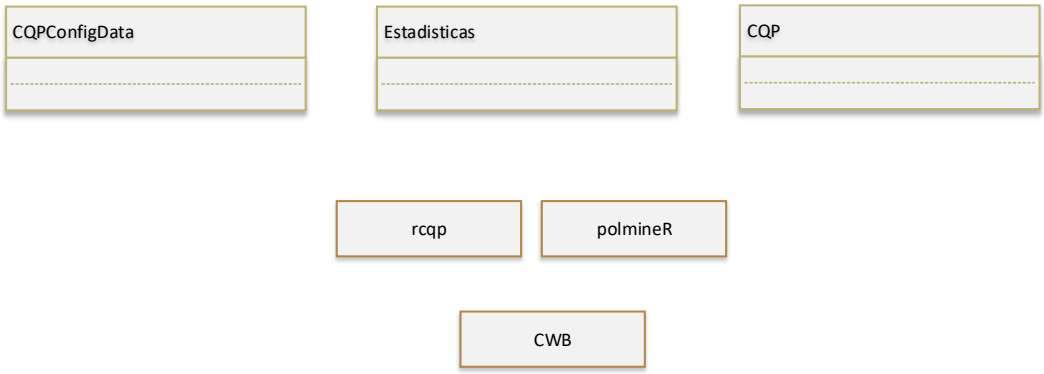


Figura 27 - Diagrama de clases simplificado relacionado con la extracción de estadísticas

Los diagramas de clases simplificados no emplean ningún lenguaje de modelado específico.

6.2.2 Diagrama de actividades

En las siguientes figuras se muestran los diagramas de actividad existentes en *ACTRES Corpus Manager*. Estos diagramas muestran el flujo de trabajo de las distintas acciones que se llevan a cabo en cada una de las actividades. Las actividades descritas son idénticas a los casos de uso especificados, es decir:

- Creación de un corpus monolingüe (Figura 28).
- Creación de un corpus paralelo bi-/multilingüe (Figura 29).
- Creación de un corpus comparable (Figura 30).
- Creación de un corpus retórico (Figura 31).
- Consulta de un corpus y extracción de estadísticas de la consulta (Figura 32).
- Obtención de estadísticas de un corpus (Figura 33).

El lenguaje empleado para la elaboración de los diagramas ha sido UML 2.

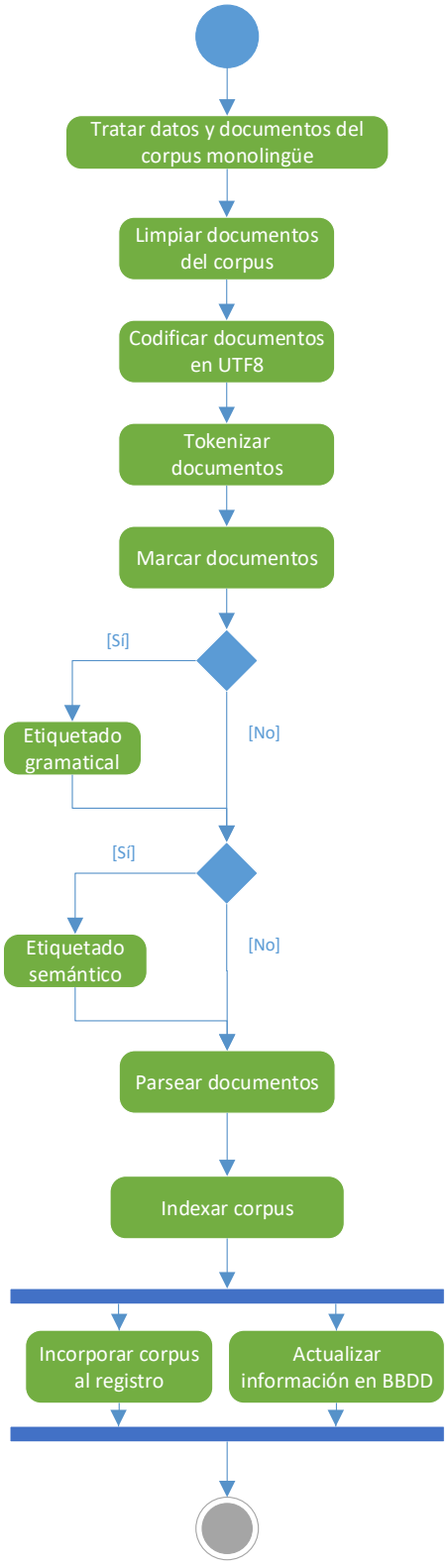


Figura 28 - Diagrama de actividad de creación de corpus monolingües

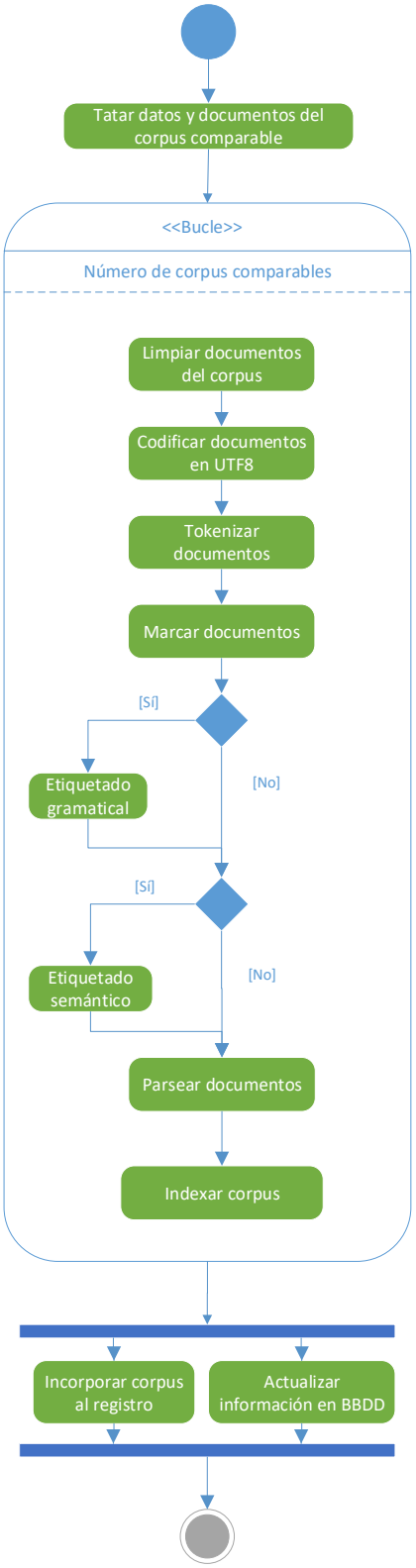


Figura 29 - Diagrama de actividad de creación de corpus comparables

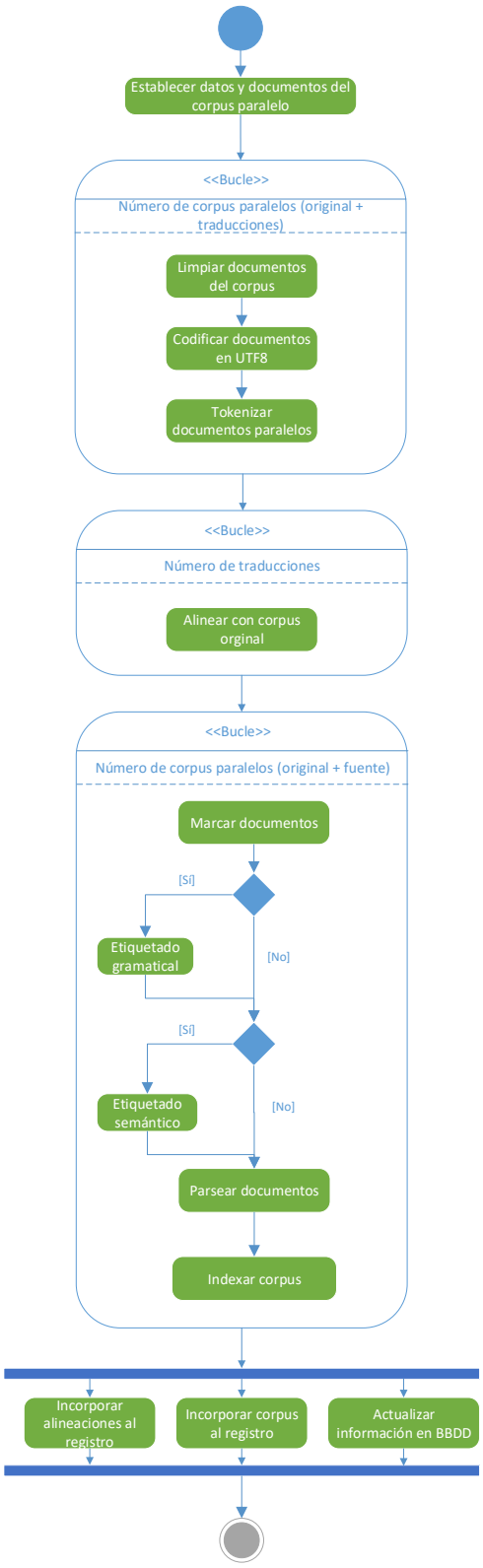


Figura 30 - Diagrama de actividad de creación de corpus paralelos

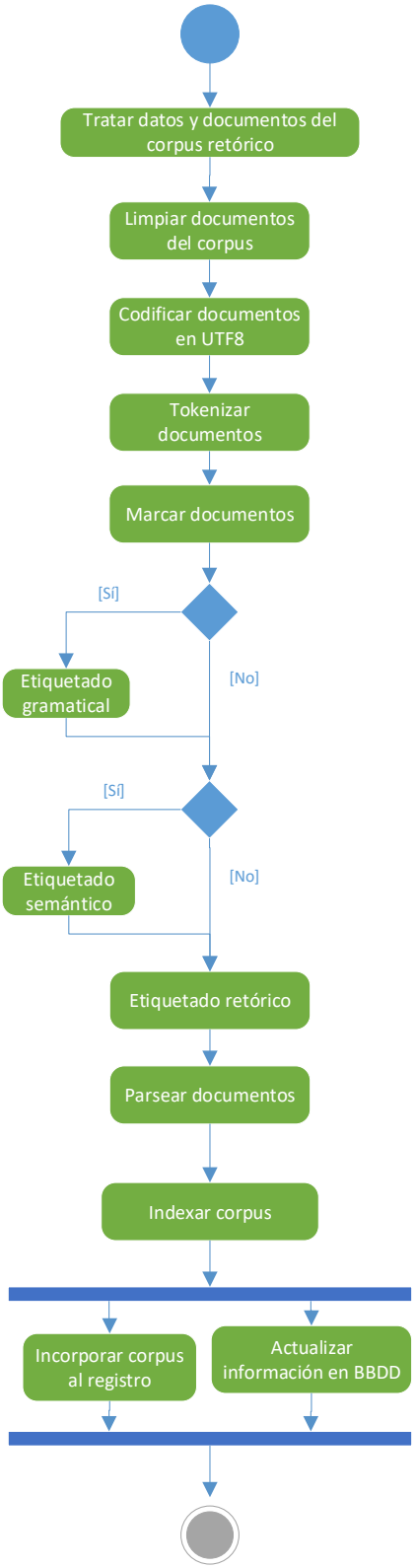


Figura 31 - Diagrama de actividad de creación de corpus retóricos

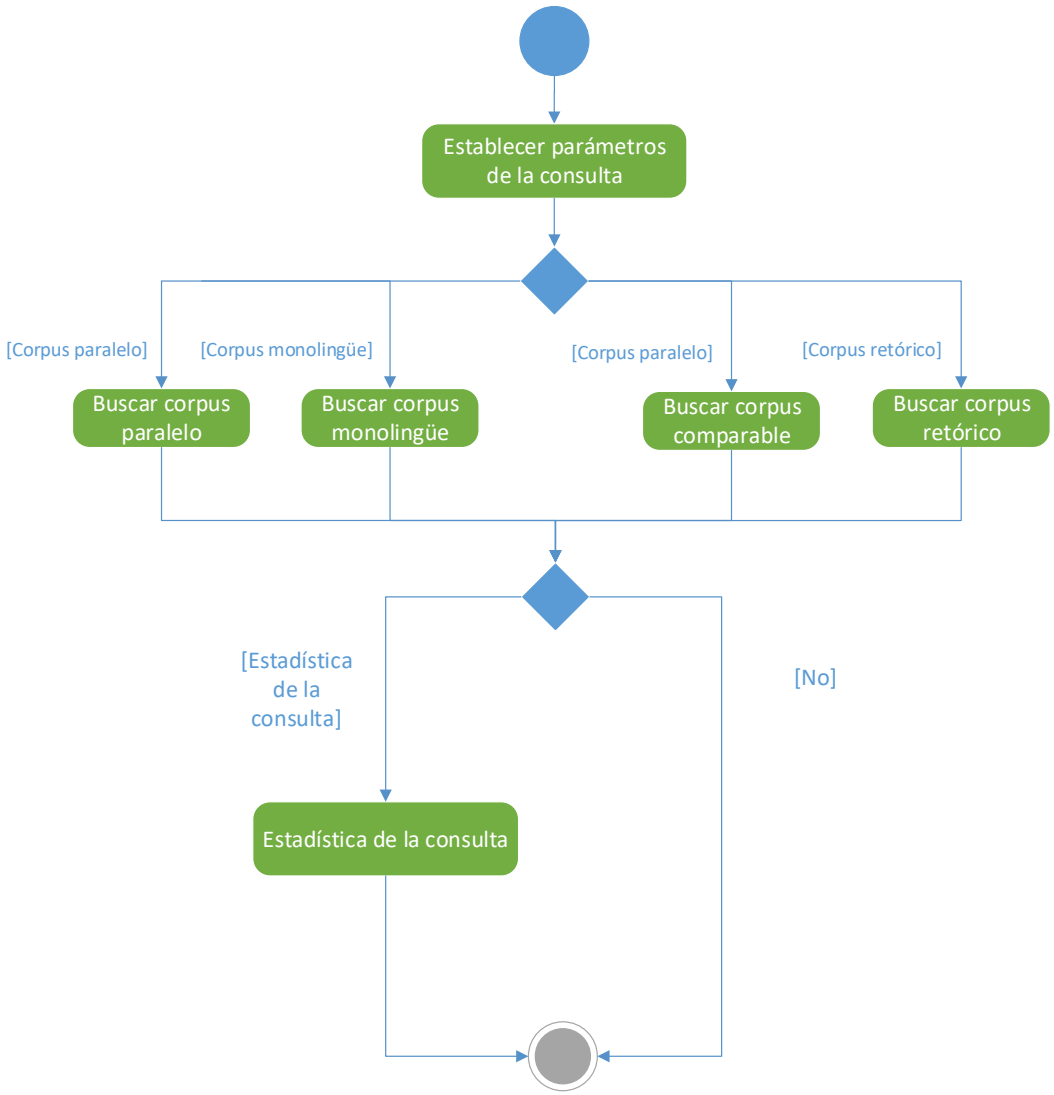


Figura 32 - Diagrama de actividad de consulta del corpus y extracción de estadísticas de la consulta

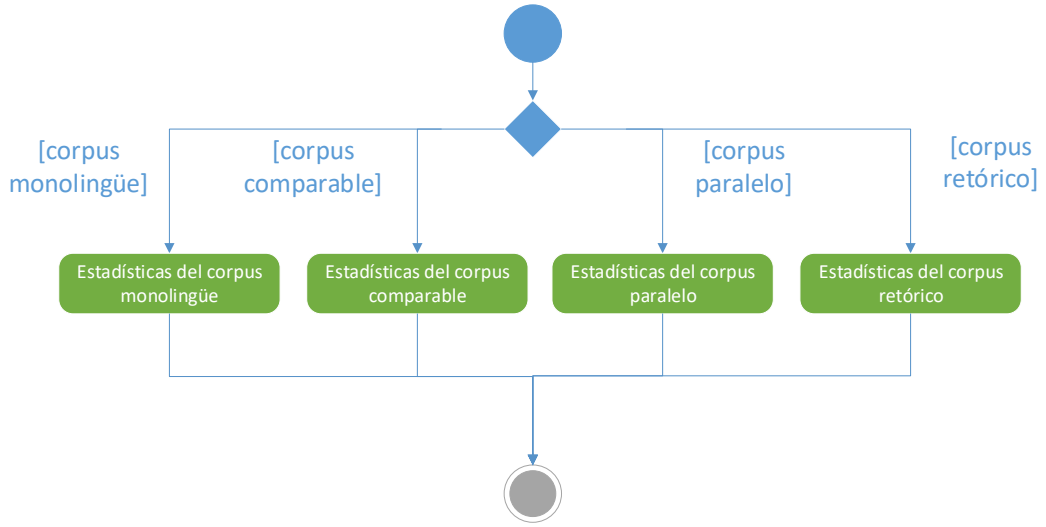


Figura 33 - Diagrama de actividad de obtención de estadísticas

6.2.3 Diseño del modelo de datos

El hecho de utilizar software externo conlleva una flexibilidad mínima en cuanto al formato de datos requerido por cada componente descrito en el diagrama de componentes. Es necesario el empleo de scripts para realizar conversiones que adecúen los flujos de datos a los requeridos para la ejecución de cada software externo.

A continuación, se describen los flujos de datos en los mismos casos que los establecidos en la especificación de los casos de uso: creación de un corpus monolingüe, creación de un corpus comparable, creación de un corpus paralelo, creación de un corpus retórico, consultar un corpus y extraer estadísticas de la consulta, y estadísticas de un corpus.

Estos diagramas han sido diseñados siguiendo la notación Gane-Sarson para DFD (*Data Flow Diagram* - Diagrama de Flujo de Datos).

El flujo de datos para la creación de corpus monolingües (Figura 34) se basa en los datos introducidos por el usuario (idioma, nombre del corpus, abreviatura) y de los

documentos que forman parte del corpus. Estos documentos van modificando su formato a lo largo del flujo gracias a la acción de los distintos componentes. Siendo en sus inicios n documentos de texto plano sin ningún formato que, tras la acción del tokenizer, se transforman en n documentos en texto plano dividido en palabras y frases, pasando posteriormente a un formato XML después de la acción del marcador, al que se le añaden anotaciones lingüísticas para, posteriormente, convertirse en un solo documento verticalizado, que tras ser indexado se incluye en el registro de los corpus de forma paralela a la introducción de los datos del corpus en la base de datos del usuario.

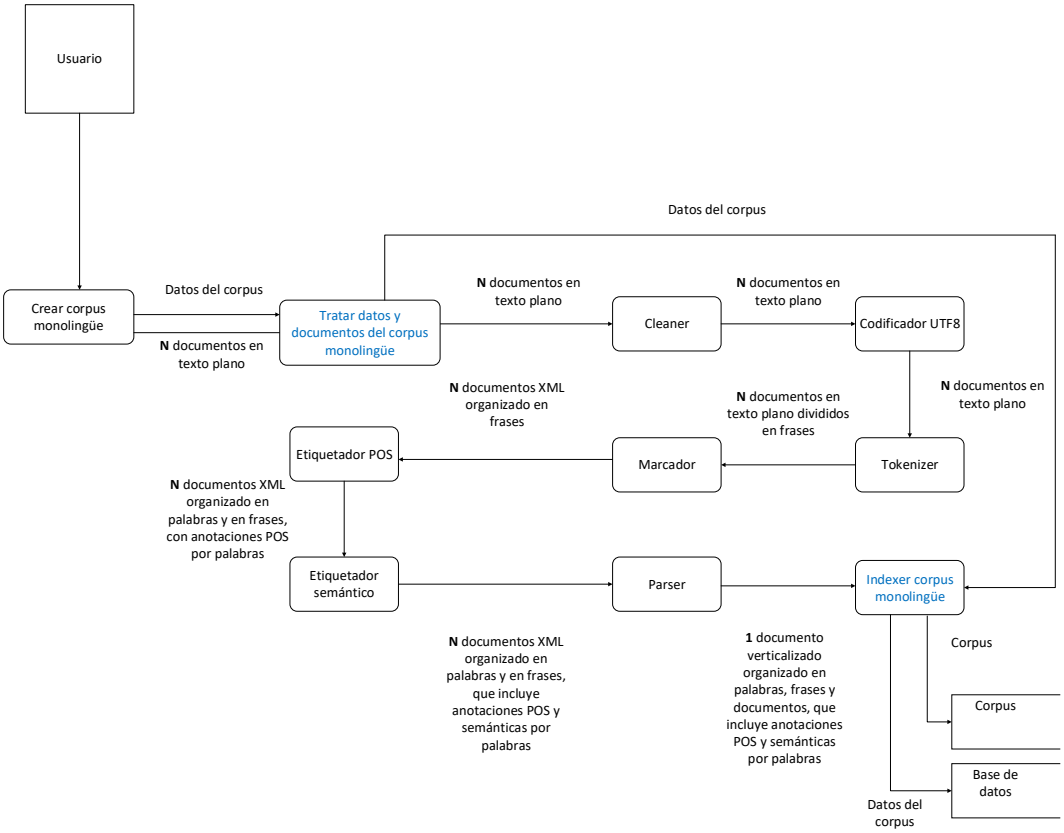


Figura 34 - Diagrama de flujo de datos para la creación de un corpus monolingüe

El flujo de datos para la creación de corpus comparables (Figura 35) es similar al del corpus monolingüe, únicamente varían los componentes de "Tratar datos y

documentos del corpus comparable" e "Indexar corpus comparable", que han de tratar con los distintos subcorpus que forman parte del corpus comparable. En cualquier caso, el flujo de datos es idéntico, como se puede apreciar a continuación.

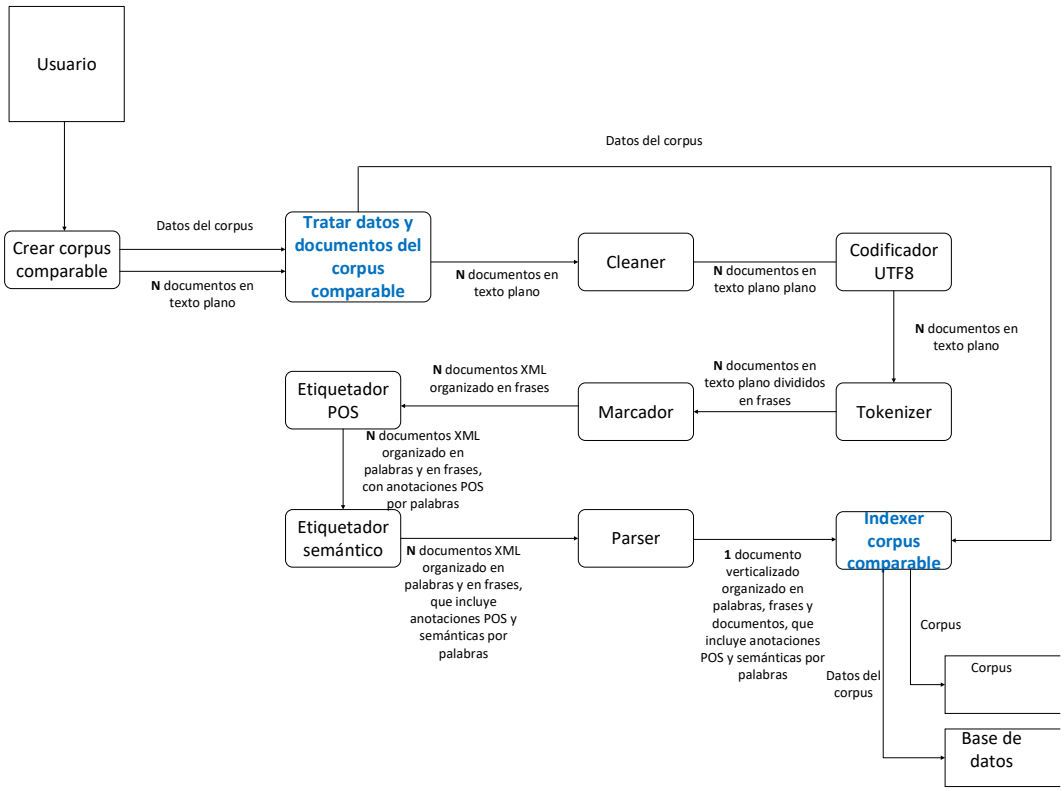


Figura 35 - Diagrama de flujo de datos para la creación de un corpus comparable

El flujo de datos para la creación de corpus paralelos (Figura 36) es similar al del corpus monolingüe, varían los componentes de "Tratar datos y documentos del corpus paralelo" e "Indexar corpus paralelo", además de la inclusión de dos nuevos componentes: "Tokenizer paralelo" y "Alineador". El componente "Alineador" incorpora un tipo de dato denominado alineaciones, que no son más que los índices de alineación del corpus paralelo. Al igual que sucede en el caso de los corpus comparables, se han de tratar con los distintos subcorpus que forman parte del corpus paralelo. El flujo de datos se puede apreciar a continuación.

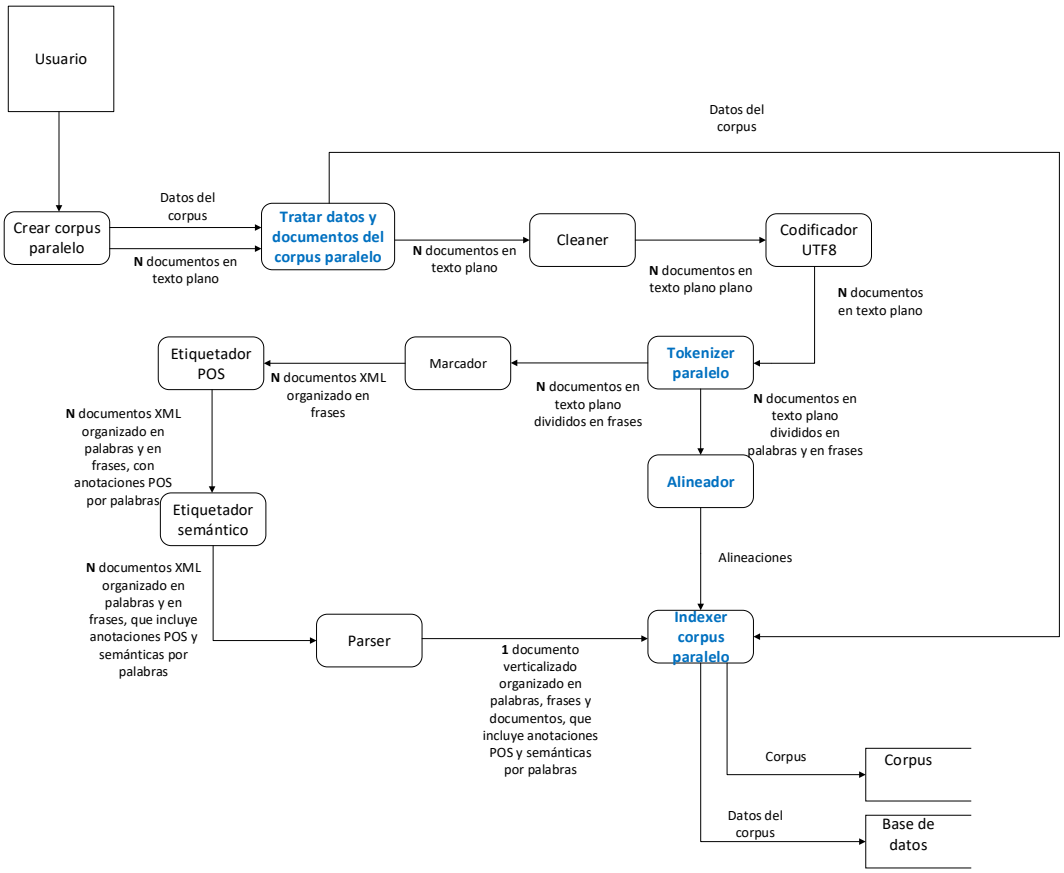


Figura 36 - Diagrama de flujo de datos para la creación de un corpus paralelo

El flujo de datos para la creación de corpus retóricos (Figura 37) también es similar al del corpus monolingüe. Varían los componentes de "Tratar datos y documentos del corpus retórico", que incluye las secciones retóricas identificadas por el usuario en cada documento de texto, la inclusión del "etiquetador retórico", que añade las categorías reconocidas por el etiquetador retórico supervisado, y por último el componente "Indexar corpus retórico".

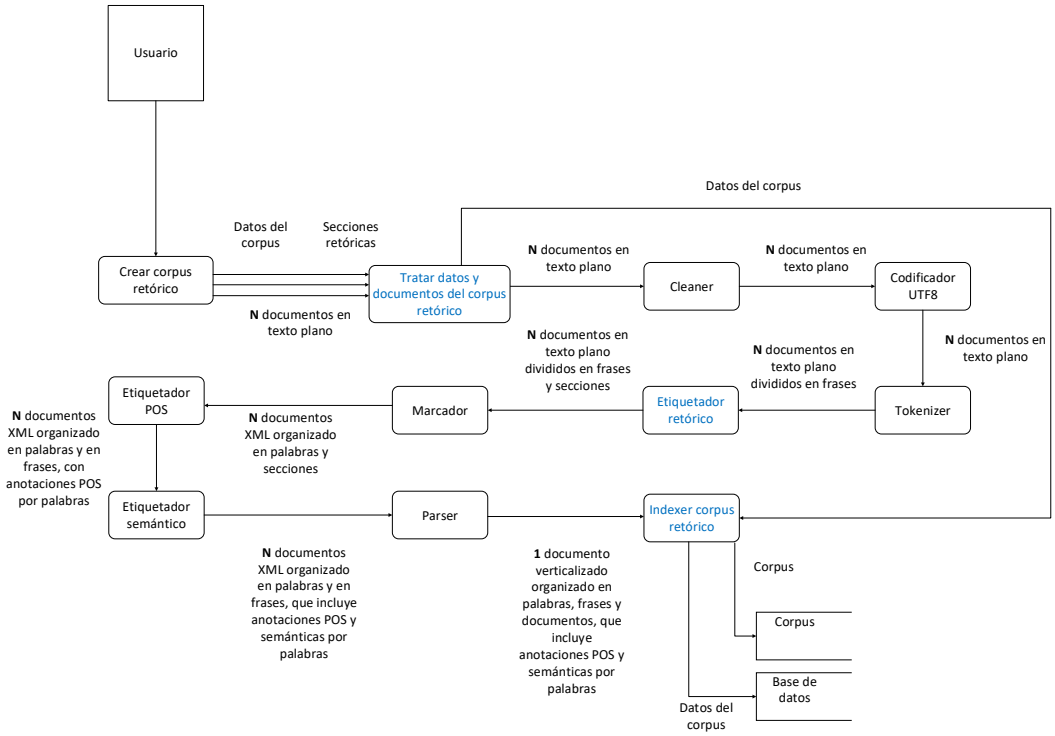


Figura 37 - Diagrama de flujo de datos para la creación de un corpus retórico

El flujo de datos para la ejecución de una consulta y de su estadística asociada (Figura 38), no importa el tipo de corpus, es mucho más simple, se recoge la consulta junto con los datos del corpus, se transforma de acuerdo con los parámetros introducidos por el usuario, y se ejecuta en el buscador. De esta forma se obtiene una serie de resultados en formato HTML para la consulta, y en formato Excel si se trata de una estadística. Estos resultados se muestran por pantalla al usuario, siendo incorporados al sistema de vistas del interfaz.

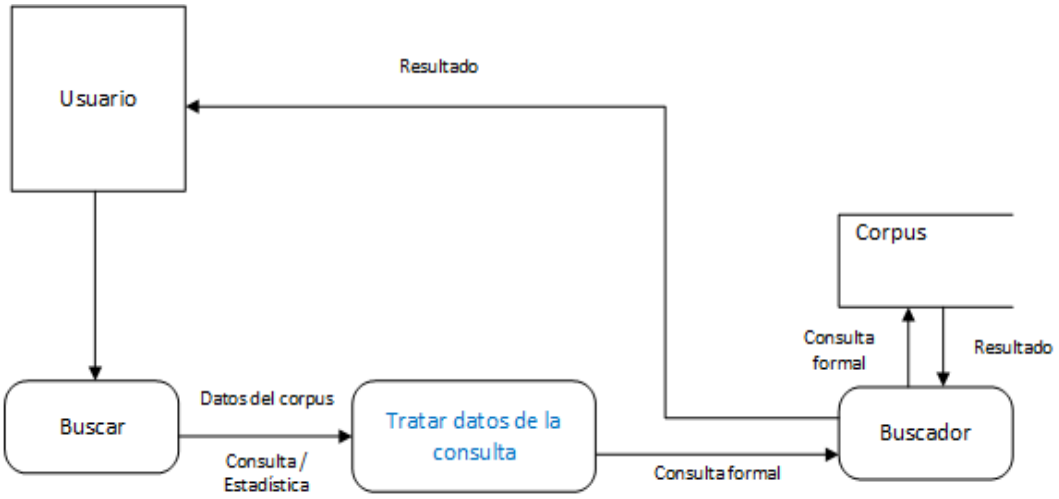


Figura 38 - Diagrama de flujo de datos para la ejecución de una consulta y de su estadística asociada

El flujo de datos para la obtención de una estadística de un corpus (Figura 39), no importa el tipo de corpus, es similar al de ejecución de una consulta. Se recoge el tipo de estadística junto con los datos del corpus, se transforma en una consulta formal y se ejecuta, obteniendo una serie de resultados en formato Excel. Estos resultados se muestran por pantalla al usuario, siendo incorporados al sistema de vistas del interfaz.

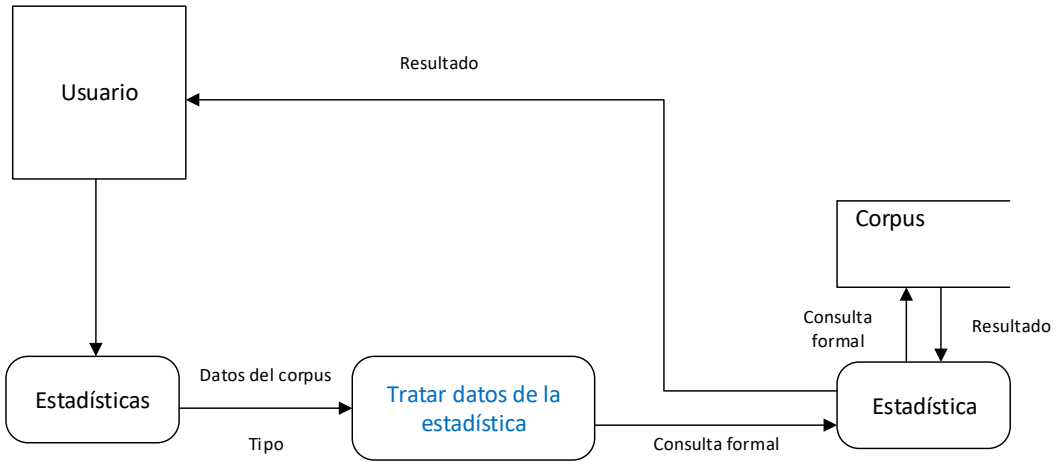


Figura 39 - Diagrama de flujo de datos para la obtención de estadísticas de un corpus

6.2.4 Diseño del interfaz de usuario

El principal reto de acuerdo con la toma de requisitos es el de crear un interfaz web que sea *responsive*, muestre instrucciones sobre cómo utilizar *ACTRES Corpus Manager* y que permita su integración en la web del grupo ACTRES. Es necesario seleccionar tecnologías que posibiliten la construcción de interfaces web válidas para todos los navegadores, sistemas operativos y tamaños de pantalla. Además, la utilización de un framework de desarrollo web facilitará la creación de nuevas versiones para la inclusión de mejoras o reparación de errores por parte de personal ajeno al desarrollo inicial del interfaz.

Mostrar los bocetos realizados para el diseño inicial de las vistas no aporta ningún valor adicional a la tesis doctoral. El desarrollo efectivo de las mismas se muestra en la sección 6.3 Implementación.

6.3 Implementación

Tal y como ha quedado demostrado a lo largo de la tesis doctoral, el tratamiento de corpus lingüísticos implica la participación y colaboración de distintas disciplinas técnicas relacionadas, sobre todo, con la lingüística computacional.

Crear un software desde cero capaz de llevar a cabo la totalidad de las tareas necesarias es algo que requeriría una gran cantidad de tiempo y la colaboración de expertos en distintas áreas: estadística, diseño de interfaces, bases de datos, NLP, etc. Si una sola persona se encargase de desarrollar todos y cada uno de los componentes software necesarios en esta tesis doctoral, se necesitaría una inmensa cantidad de tiempo para que el desarrollador se especializara en todos los procesos implicados, con el objetivo de alcanzar un grado de eficiencia y rapidez aceptable para procesar corpus de gran tamaño.

Por estos motivos, la reutilización, modificación e interconexión de recursos existentes, cuya validez ha sido demostrada y probada por la comunidad científica

es la mejor opción. Esta estrategia es combinada con la creación de nuevos recursos software en aquellos huecos que el estado de la cuestión ha revelado poco maduros o que simplemente son demasiado estrictos o incompatibles, y que por tanto no se adecuan a las necesidades del framework a desarrollar. Por último, resulta imprescindible desarrollar controladores propios, así como modelos que utilicen los recursos software encargados de realizar las tareas específicas de los distintos componentes.

6.3.1 Lenguajes de programación empleados

Los lenguajes de programación utilizados en el desarrollo de *ACTRES Corpus Manager* han sido:

- XML (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 1998) es un lenguaje empleado como formato de datos durante el proceso de creación del corpus lingüístico.
- JSON (ECMA International, 2013) es un lenguaje empleado como formato de datos en la creación de las bases de datos de usuarios y corpus asociados.
- PHP (The PHP Group, 2017) es el lenguaje principal de desarrollo de la lógica del framework como consecuencia de su idoneidad para el desarrollo de aplicaciones web. Concretamente es el lenguaje utilizado en el modelo, y sirve para la elaboración de las distintas clases y métodos que invocan la acción del software propio o externo, encargado de ejecutar la acción de cada componente. También se encarga de la adecuación del flujo de datos a los requisitos de formato establecidos por los distintos componentes. Por último, ha sido el lenguaje empleado para desarrollar el etiquetador semántico y el etiquetador retórico.
- Python (Python Software Foundation, 2017b) es empleado para el procesamiento de texto en el que el idioma del mismo sea un condicionante a la hora de ejecutar una acción. Concretamente, es empleado en el componente tokenizer.

- Perl (Wall, 2017) es utilizado únicamente en la elaboración del componente tokenizer para crear corpus paralelos.
- JavaScript (Ecma International, 2011), por medio de la librería JQuery (The JQuery Foundation, 2017), es usado para desarrollar el controlador del framework y la lógica de la vista.
- HTML5 (W3C, 2014), CSS3 (W3C, 2015) y JavaScript, a través del framework de desarrollo web Bootstrap (Otto & Thornton, 2016) es empleado para desarrollar las distintas vistas del interfaz.
- R (R Core Team, 2017) es utilizado para la extracción de estadísticas cualitativas y cuantitativas.
- GNU Bash (Free Software Foundation, 2017) es un lenguaje de comandos empleado principalmente para el tratamiento de ficheros en el servidor.

6.3.2 Software empleado

El listado de los recursos software empleados, así como su misión en *ACTRES Corpus Manager* se describe en la siguiente tabla.

Software	**Descripción**
Xubuntu 16.04.1 LTS	Sistema operativo empleado en el desarrollo del framework.
HTTPD - Servidor web Apache2	Servidor web utilizado para alojar la el framework.
NetBeans IDE 8.1	IDE para el desarrollo de aplicaciones de escritorio, móviles o web. Concretamente se ha utilizado su versión para el lenguaje de programación PHP, que como se detalló en el apartado anterior, es empleado como lenguaje de programación principal.
RStudio 0.99.903	IDE para el desarrollo de aplicaciones escritas en R. Se ha utilizado para desarrollar los scripts en R.

Mousepad 0.4.0	Editor de texto plano con soporte complementario para la creación de los scripts de Python, *bash* y Perl.
hunalign	hunalign (Varga, et al., 2005) es el software encargado de llevar a cabo el alineado no supervisado a nivel de oraciones.
Treetagger	Treetagger (Schmid, 1995), utilizado para llevar a cabo el etiquetado gramatical no supervisado.
The IMS Corpus Workbench (CWB)	El sistema de indexación y las estadísticas cuantitativas relacionadas con frecuencias de aparición hacen uso de la colección de herramientas de CWB (Evert & Hardie, 2011). Además, se hace uso de la clase *cqp.inc.php* perteneciente a CQPweb (Hardie, 2012) como *back-end* de acceso a CWB. Por último, las búsquedas se hacen por medio del módulo de CWB denominado CQP Query Language (Evert, 2009).
polmineR y rcqp	La estadística cualitativa lista de palabras clave, y las estadísticas cuantitativas n-gramas y listas de frecuencias del corpus, hacen uso del paquete estadístico escrito en R denominado polmineR (Blaette, 2017), que utiliza emplea a su vez el API de acceso a CWB rcqp (Desgraupes & Loiseau, 2016), también escrito en R.
Multilingual UCREL Semantic Analysis System (USAS)	Se ha creado una aplicación propia en PHP que utiliza los diccionarios (UCREL, 2016) pertenecientes al software Multilingual UCREL Semantic Analysis System (USAS) (Piao, Bianchi, Dayrell, D'Egidio, & Rayson, 2015).
NLTK	(Bird, Klein, & Loper, 2009) es el toolkit de Python empleado para desarrollar el componente tokenizer en Python.

Europarl tools	El tokenizer de oraciones de Europarl (Koehn, 2005) ha sido empleado para crear el componente tokenizer en Perl para la creación de corpus paralelos. Esta decisión se explicará en detalle en el siguiente apartado.

Tabla 35 - Software empleado en el desarrollo de *ACTRES Corpus Manager*

6.3.3 Limitaciones de la implementación

Durante el proceso de implementación de *ACTRES Corpus Manager* se autoimpusieron un conjunto de limitaciones relacionadas con la disposición de recursos lingüísticos, estimación de tiempos de desarrollo y utilidad práctica efectiva en el desarrollo del prototipo. A continuación, se describen estas limitaciones:

- **Idiomas soportados**: Cada idioma admitido por parte de *ACTRES Corpus Manager* impone la incorporación de unos recursos lingüísticos que permitan realizar de forma adecuada operaciones como tokenizados, anotaciones lingüísticas, alineados, etc. Carece de sentido proporcionar soporte para un número elevado de idiomas sin que haya una demanda real por parte de los usuarios potenciales de la aplicación, que en este caso se corresponden con el grupo de investigación ACTRES. Como declaración de intenciones se admitirán cuatro idiomas para la creación de corpus lingüísticos anotados: español, inglés, francés e italiano.
- **Etiquetado retórico**: Al no existir un software capaz de realizar un etiquetado retórico no supervisado y con el objetivo de que el trabajo realizado por el lingüista fuera el mínimo indispensable, se tomó la decisión de que la creación del corpus con etiquetado retórico fuera un caso diferenciado del resto de corpus. Los motivos que sustentan esta decisión son los siguientes:

  - Que el etiquetado sea supervisado implica la participación activa del usuario, que a su vez provoca que:
    - Esté restringido a textos cortos para que el etiquetado sea realizable en tiempo y forma.
    - Creación de un interfaz para el etiquetado manual que sirva para determinar las secciones retóricas.
  - La participación activa del usuario implica que la creación de un corpus paralelo o comparable con etiquetado retórico necesite demasiado tiempo y esfuerzo por parte del usuario. Este es el principal motivo por el que es restringido a corpus monolingües hasta que no se logre automatizar el proceso.
  - En resumen, se trata de un añadido de funcionalidad de *ACTRES Corpus Manager* que requiere de un estudio adicional para su integración completa como etiquetado en todos los tipos de corpus soportados por el framework.
- **Número de subcorpus admitidos**: Con el objetivo de que el rendimiento de *ACTRES Corpus Manager* sea aceptable y la visualización de resultados pueda realizarse sin dificultad, el número de subcorpus máximo que pueden formar parte de un corpus paralelo o corpus comparable será de 4.
- **Estadísticas basadas en tests de significación**: La única estadística que empleará tests de significación será la extracción de palabras clave.
- **Estadísticas de los corpus paralelos y comparables**: Las estadísticas de los corpus paralelos y comparables se basa en las estadísticas extraídas de los subcorpus que los componen. Por este motivo, la extracción palabras clave y extracción de n-gramas, complejas desde el punto de vista de necesidad de recursos computacionales, serán sólo implementadas en el caso de los corpus monolingües.
  - Todos los corpus tienen garantizado acceso a estas funcionalidades a través de la selección de cualquiera de los subcorpus que los

componen, puesto que también son considerados corpus monolingües por *ACTRES Corpus Manager*.

- **Alineación**: El alineado de corpus se realiza sobre un subcorpus original y sus respectivas traducciones, es decir, los subcorpus traducidos no están necesariamente alineados entre sí, por lo que la prioridad de *ACTRES Corpus Manager* es facilitar la búsqueda sobre el subcorpus original y sus respectivas traducciones.

6.3.4 Desarrollo: descripción de los componentes

De acuerdo con los diagramas de componentes de la Figura 20, Figura 21, Figura 22 y Figura 23, *ACTRES Corpus Manager* está formado por distintos elementos interconectados entre sí siguiendo un flujo de ejecución secuencial que varía según la actividad que el usuario seleccione.

El patrón de diseño se inspira en MVC con una vista activa, similar a la existente en el diseño MVVM (ver apartado 6.2.1 Diseño de la arquitectura para más información) pero utilizando controladores.

En el patrón de diseño seleccionada *ACTRES Corpus Manager* se divide en tres partes (Vista, Controlador y Modelo), que se describirán en detalle a continuación.

6.3.4.1 Vista

La vista está formada por un único componente que se denomina “Interfaz”. En el patrón de diseño utilizado la vista contiene los elementos visuales (botones, formularios, entradas de texto, etc.) y, además toda la lógica necesaria para tratar con todas las interacciones de los mismos que no requieran acceso al modelo del framework.

Los elementos visuales del componente interfaz se han implementado empleando el framework de desarrollo para programación web Bootstrap, que permite construir

interfaces web *responsive* basadas en HTML5, CSS3 (*Cascading Style Sheets* - hojas de estilo en cascada) y JavaScript.

Por su parte, la lógica de la vista está implementada utilizando un paradigma de programación procedural en JavaScript con ayuda de la librería JQuery, de funcionalidades Javascript propias de Bootstrap y de las librerías Javascript DropzoneJS (Meno, 2012), TableExport (Clarke, 2017) y Rangy Inputs (Down, 2017). Cada uno de estos elementos realiza diferentes funciones:

- JQuery permite el acceso a los distintos componentes del interfaz mediante una sintaxis simple, limpia y potente.
- DropzoneJS es utilizado para implementar la parte del interfaz encargada de la subida de documentos del corpus.
- TableExport es utilizado para exportar los resultados de las consultas sobre los corpus a formato csv, txt y Excel.
- Rangy Inputs es utilizado para crear el interfaz de etiquetado retórico incorporado en la lógica de la vista.

Carece de sentido presentar con exhaustivo detalle los elementos visuales utilizados o la lista de funciones que maneja la lógica de la vista, ya que no aporta ningún conocimiento novedoso y por lo tanto no añade valor alguno a la tesis doctoral. No obstante, se describirá el proceso de implementación, los recursos técnicos empleados y el listado de ficheros utilizado.

Aunque las distintas vistas del interfaz se mostrarán con detalle en el Capítulo 7: "Validación del framework", se ha creído conveniente mostrar la vista inicial del sistema junto con una breve descripción de las partes primitivas que la componen (ver Figura 40).

Figura 40 - Vista inicial de *ACTRES Corpus Manager* y sus secciones resaltadas

6.3.4.1.1 Elementos visuales

El interfaz está formado por un conjunto de vistas. Aunque la extensión de las vistas es de tipo PHP, se trata de ficheros HTML a los que se ha añadido un script en PHP para poder manejar las sesiones en el servidor (Figura 41) entre otras funcionalidades.

```php
<?php
    session_start();
    $ds = DIRECTORY_SEPARATOR;
    if(!isset($_SESSION['username'])){
        header("location:/".$_SESSION["project_name"].$ds."public_html".$ds."signin.php");
        exit;
    }else{
        require($_SESSION["project_path"].$ds.'public_html'.$ds.'php'.$ds."controller".$ds."delete_tmp.php");
        delete_tmp("mono");
    }
?>
```

Figura 41 - Ejemplo de fragmento de código PHP para manejar la sesión de usuario

El listado de vistas se muestra a continuación:

- index.php: Vista para presentar *ACTRES Corpus Manager*.
- signin.php: Vista para que el usuario se registre o identifique.

- user_info.php: Vista que muestra la información del usuario: nombre, email y corpus disponibles.
- create_index.php: Vista que presenta al usuario los tipos de corpus se pueden crear.
- create_mono.php: Vista que guía al usuario en el proceso de creación de un corpus monolingüe.
- create_comparable.php: Vista que guía al usuario en el proceso de creación de un corpus comparable.
- create_paralelo.php: Vista que guía al usuario en el proceso de creación de un corpus paralelo.
- create_retorico.php: Vista que guía al usuario en el proceso de creación de un corpus monolingüe retórico.
- query_index.php: Vista que muestra al usuario los corpus disponibles para consultar.
- query_mono.php: Vista en la que el usuario puede ejecutar una consulta o extraer estadísticas de un corpus monolingüe. Los resultados de una consulta se muestran en formato KWIC, los resultados estadísticos en formato Excel.
- query_comparable.php: Vista en la que el usuario puede ejecutar una consulta o extraer estadísticas de un corpus comparable. Los resultados de una consulta se muestran en formato KWIC, los resultados estadísticos en formato Excel.
- query_paralelo.php: Vista en la que el usuario puede ejecutar una consulta o extraer estadísticas de un corpus multilingüe paralelo. Los resultados de una consulta se muestran en formato KWIC, los resultados estadísticos en formato Excel.
- query_retorico.php: Vista en la que el usuario puede ejecutar una consulta o extraer estadísticas de un corpus monolingüe retórico. Los resultados de

una consulta se muestran en formato KWIC, los resultados estadísticos en formato Excel.

Las hojas de estilo asociadas a las vistas se han desarrollado en CSS3 y se combinan hojas de estilo propias junto a las pertenecientes a Bootstrap, con distintas modificaciones puntuales para ajustarse con exactitud al diseño deseado. El listado de ficheros CSS3 se muestra a continuación:

- Bootstrap
  - bootstrap.min.css
  - carousel.css
  - justified_nav.css
  - carousel.css
  - font-awesome.min.css
- Dropzone.js
  - upload_dropzone.css [40]
- Propios
  - query_results.css
  - create.css
  - comun.css
  - loading.css

Por último, con el fin de favorecer la legibilidad y evitar la duplicidad de código se ha empleado un sistema de plantillas web propio (Figura 42), basado en PHP para elementos redundantes aparecidos en múltiples las vistas, como son:

- La inclusión de metadatos de la página web.
- La inclusión de hojas de estilo CSS comunes.
- La inclusión de librerías Javascript comunes.
- Pie de página.
- Encabezado de página.

[40] Basado en el framework web Bootstrap.

- Diálogos de interacción: mensajes de errores, información y advertencias.

```php
<?php
    include( $_SESSION["project_path"].$ds."public_html".$ds."templates".$ds."common_scripts.html");
?>
```

Figura 42 - Ejemplo de fragmento de código PHP para la inclusión de plantillas

6.3.4.1.2 Lógica de la vista

La lógica de la vista está implementada íntegramente en Javascript, haciendo uso de la sintaxis simplificada de JQuery. La lógica de la vista se encarga del tratamiento de las interacciones ocurridas en la misma.

El listado de ficheros JavaScript y su utilidad se presenta en la Tabla 36:

Nombre del fichero	**Utilidad**
comun.js	Tratamiento de interacciones comunes a todas las vistas.
create_comun.js	Tratamiento de interacciones comunes a todas las vistas relacionadas con la creación de corpus.
create_mono.js	Tratamiento de interacciones relacionadas con la creación de corpus monolingües.
create_comparable.js	Tratamiento de interacciones relacionadas con la creación de corpus comparables.
create_paralelo.js	Tratamiento de interacciones relacionadas con la creación de corpus multilingües paralelos.
create_retorico.js	Tratamiento de interacciones relacionadas con la creación de corpus monolingües retóricos, incluyendo el etiquetado retórico manual.[41]
query_index.js	Tratamiento de interacciones relacionadas con la vista de la página inicial de consulta y extracción de estadísticas de corpus.

[41] Aunque los datos se obtendrían desde el interfaz de usuario, perteneciente por tanto a la lógica de la vista, se ha creído adecuado explicar el etiquetado retórico de forma conjunta en la lógica del framework (Ver IX) Etiquetador retórico).

query_comun.js	Tratamiento de interacciones comunes a todas las vistas relacionadas con la consulta y extracción de estadísticas de corpus.
query_mono.js	Tratamiento de interacciones relacionadas con la consulta y extracción de estadísticas de corpus monolingües.
query_comparable.js	Tratamiento de interacciones relacionadas con la consulta y extracción de estadísticas de corpus comparables.
query_paralelo.js	Tratamiento de interacciones relacionadas con la consulta y extracción de estadísticas de corpus multilingües paralelos.
query_retorico.js	Tratamiento de interacciones relacionadas con la consulta y extracción de estadísticas de corpus retóricos.

Tabla 36 - Utilidad de los ficheros Javascript utilizados en la lógica de la vista

A modo de resumen, el listado de vistas de *ACTRES Corpus Manager* junto con los ficheros Javascript, ficheros CSS y librerías externas correspondientes con las que tienen relación se muestra en la Tabla 37.

Vistas	**Fichero Javascript**	**CSS**	**Librerías empleadas**
Todas	comun.js	bootstrap.min.css justified_nav.css comun.css	JQuery Bootstrap
create_index.php	create_comun.js	bootstrap.min.css justified_nav.css comun.css	JQuery Bootstrap
create_mono.php	create_mono.js create_comun.js	bootstrap.min.css justified_nav.css comun.css jquery-ui.css carousel.css upload_dropzone.css	JQuery Bootstrap DropzoneJS
create_comparable.php	create_comparable.js create_comun.js	bootstrap.min.css justified_nav.css comun.css	JQuery Bootstrap DropzoneJS

		jquery-ui.css carousel.css upload_dropzone.css	
create_paralelo.php	create_paralelo.js create_comun.js	bootstrap.min.css justified_nav.css comun.css jquery-ui.css carousel.css upload_dropzone.css	JQuery Bootstrap DropzoneJS
create_retorico.php	create_retoricoo.js create_comun.js	bootstrap.min.css justified_nav.css comun.css jquery-ui.css carousel.css upload_dropzone.css	JQuery Bootstrap DropzoneJS RangyInputs
query_index.php	query_index.js	bootstrap.min.css justified_nav.css comun.css	JQuery Bootstrap
query_mono.php	query_mono.js	bootstrap.min.css justified_nav.css comun.css query_results.css	JQuery Bootstrap TableExport
query_comparable.php	query_comparable.js	bootstrap.min.css justified_nav.css comun.css query_results.css	JQuery Bootstrap TableExport
query_paralelo.php	query_paralelo.js	bootstrap.min.css justified_nav.css comun.css query_results.css	JQuery Bootstrap TableExport
query_retorico.php	query_retorico.js	bootstrap.min.css justified_nav.css comun.css query_results.css	JQuery Bootstrap TableExport

Tabla 37 - Vistas y ficheros Javascript, CSS y librerías asociados

6.3.4.2 Controlador

El controlador de *ACTRES Corpus Manager* actúa como intermediario entre la vista y el modelo. Tal y como se ha mencionado con anterioridad, la vista es activa y por

tanto posee su propia lógica, al contrario de lo que sucede en MVC tradicional. Únicamente aquellas interacciones del usuario que requieran acceso al modelo de datos del framework serán gestionadas por el controlador.

El controlador se comunica con la vista y modelo por medio de Javascript, vía llamadas AJAX o a través de formularios *HTTP request* tradicionales.

Cada controlador convierte los datos introducidos por el usuario en parámetros comprensibles para el modelo correspondiente y viceversa, transformando las respuestas del modelo en distintas operaciones para modificar la vista.

```
$("#create_mono_corpus").click(function(){

    var parametros = {
        "corpus_complete_name" :nombre_corpus_global,
        "language" : idioma_corpus_global,
        "POS" : POS_global,
        "semantico" : semantico_global,
        "abbr" : abbr_name_global,
        "user" : "<?php echo $_SESSION['username'] ?>",
        "tipo" : "mono"
    };

    $.ajax({
        type: "POST",
        url: "<?php echo $_SESSION['project_url'] ?>/public_html/php/model/create_mono.php",
        data:parametros,
        beforeSend: function() {
            $("#main").hide();
            $("#loading").show();
        },
        success: function(result){
            $("#loading_button").hide();
            $(".alert-success.alert_final").show();
            $("#continuar").show();
        },
        error: function(){
           $(".alert-danger.alert_final").show();
        }
    });

})
```

Figura 43 - Fragmento de código Javascript que funciona como controlador

En la Figura 43 se aprecia el comportamiento interno de un controlador:

(1) Existen una serie de parámetros (*parametros*), que pueden requerir o no una conversión previa.

(2) Una llamada al modelo correspondiente (*url*), al que se le pasan los parámetros previamente descritos a través del atributo *data*.

(3) Definición de las distintas funciones que tratan las fases por las que puede discurrir la llamada al modelo. En este ejemplo concreto: si hubiera error (*error*), antes de enviar la petición (*beforeSend*) y en caso de éxito en la respuesta procedente del controlador (*success*).

La Figura 44 describe la interacción de un controlador del framework con los elementos Vista y Modelo.

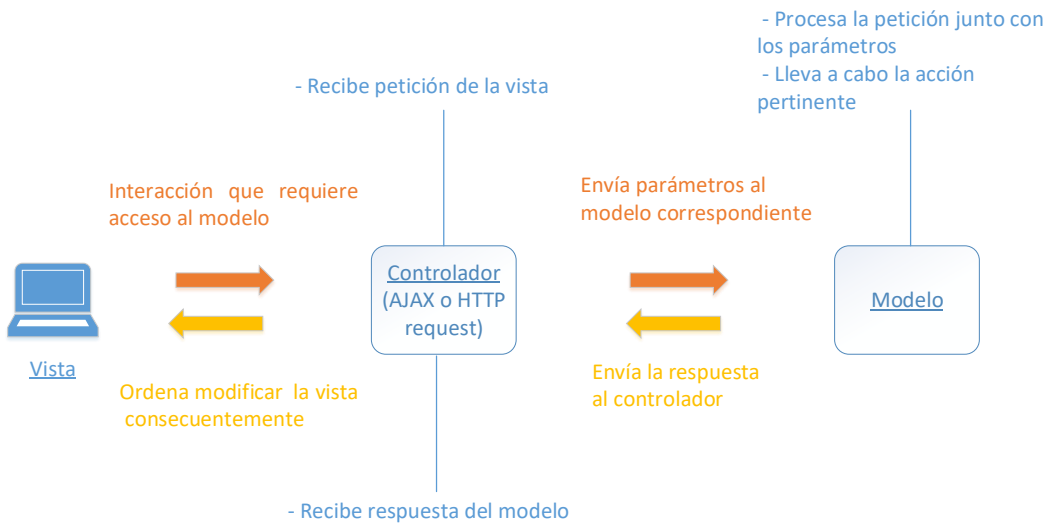


Figura 44 – Interacción del controlador diseñado con la vista y el modelo

El listado de controladores existente en *ACTRES Corpus Manager* se muestra a continuación:

- En todas las vistas:
  - Cerrar sesión de usuario.
  - Iniciar sesión de usuario.
  - Borrar archivos temporales de usuario.

- En vista para crear corpus monolingües:
    - Cancelar subida de archivos.
    - Borrar documento subido.
    - Subir documento del corpus.
    - Crear corpus monolingüe.
- En vista para crear corpus comparables:
    - Cancelar subida de archivos.
    - Borrar documento subido.
    - Subir documento del corpus.
    - Crear corpus comparable.
- En vista para crear corpus paralelos:
    - Cancelar subida de archivos.
    - Borrar documento subido.
    - Subir documento del corpus.
    - Crear corpus paralelo.
- En vista para crear corpus retóricos:
    - Cancelar subida de archivos.
    - Borrar documento del corpus.
    - Subir documento del corpus.
    - Crear corpus retórico.
- En vista para consultar corpus monolingüe:
    - Obtener lista de frecuencias.
    - Obtener colocaciones.
    - Obtener n-gramas.
    - Ejecutar consulta de corpus.
    - Obtener lista de frecuencias de la consulta.
    - Obtener lista de palabras clave.
- En vista para consultar corpus comparable:
    - Obtener lista de frecuencias.

    - Obtener colocaciones.
    - Ejecutar consulta de corpus.
    - Obtener lista de frecuencias de la consulta.
- En vista para consultar corpus paralelo multilingüe:
    - Obtener lista de frecuencias.
    - Obtener colocaciones.
    - Ejecutar consulta de corpus.
    - Obtener lista de frecuencias de la consulta.
- En vista para consultar corpus retórico:
    - Obtener lista de frecuencias.
    - Obtener colocaciones.
    - Ejecutar consulta de corpus.
    - Obtener lista de frecuencias de la consulta.

6.3.4.3 Modelo

El modelo del framework está formado por (Figura 45):

- Datos del framework.
- Lógica del framework (a excepción de la lógica de la vista previamente descrita).

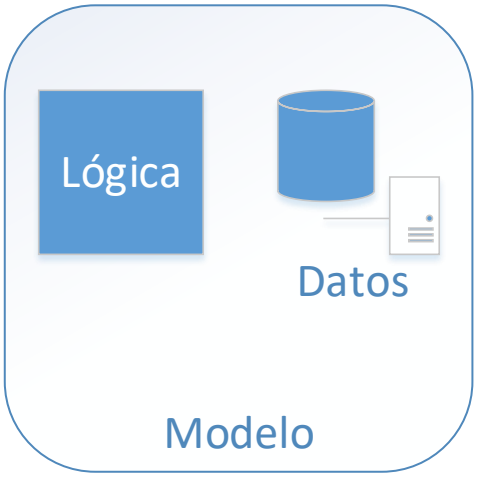


Figura 45 - Datos y lógica en el modelo del framework

Es importante señalar que la elección de CWB como software para la gestión de los corpus lingüísticos condiciona en su totalidad el diseño del modelo del framework, tanto en los datos como en la lógica.

A continuación, se describirán cada uno de los elementos del modelo creado: datos y lógica del framework.

6.3.4.3.1 Datos del framework

Los datos del framework (Figura 46) están implementados en dos configuraciones distintas:

- **Datos de usuario y corpus asociados**: Son totalmente independientes del modelo de datos del sistema de indexación. Estos datos se dividen en dos colecciones:
  - Base de datos de usuarios.
  - Base de datos de información del corpus.
- **Datos CWB**: Modelo de datos empleado por el sistema de indexación y consulta que contiene los corpus indexados e información sobre su ubicación y características de acuerdo con CWB. Se divide en dos elementos:
  - Registro.
  - Corpus indexados.

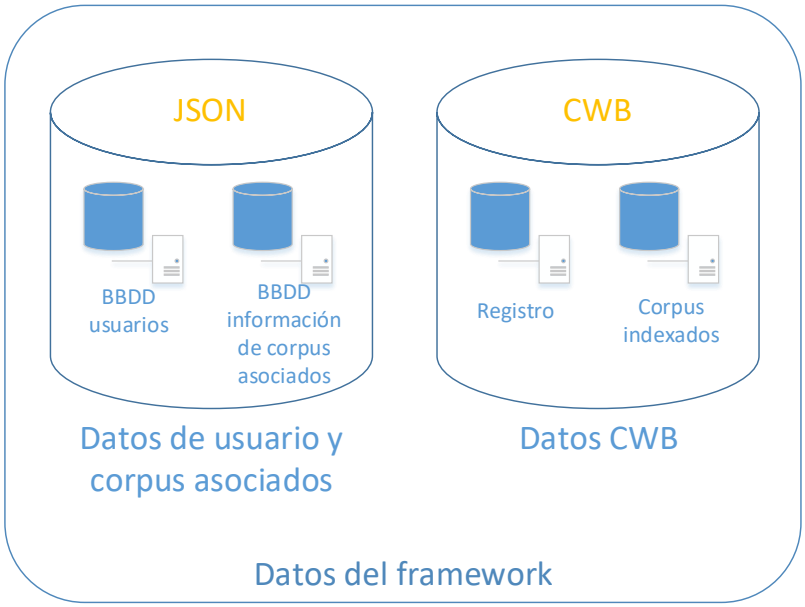


Figura 46 - Configuración de los datos del framework presentes en el componente 'Modelo'

A continuación, se detalla cada configuración de datos de forma minuciosa.

6.3.4.3.1.1 Datos de usuario y corpus asociados

Los datos de usuarios y corpus asociados están almacenados en el servidor por medio de dos bases de datos que emplean un lenguaje semiestructurado, más concretamente JSON. Las razones por la que se emplea JSON en lugar de una base de datos relacional de tipo SQL (como MySQL o SQLite) para almacenar la información son las siguientes:

- Su implementación es mucho más rápida y sencilla.
- El número de usuarios iniciales y los datos de usuario y corpus son mínimos.
- No se realizan consultas complejas sobre estos datos.
- Las ventajas inherentes a la utilización de bases de datos estructuradas (almacenamiento eficiente, consultas SQL complejas, seguridad etc.) no eran aprovechadas.

Tal y como se describió al comienzo del subapartado, los datos de usuario e información de corpus asociados están almacenados en dos colecciones distintas:

(1) **La base de datos de usuarios**: Un solo documento que almacena los datos de usuario tales como email (actúa como identificador único), nombre de usuario, contraseña encriptada y organización. Ver Figura 47.


```
"actres@unileon.es": {
    "name":"actres",
    "password":"$2y$10$cBDEzxuLeFypC7K5k3iAMudHVi6RssxGafqLsbhjY0vOk2/3biRe6",
    "organization":"university"
},
```


Figura 47 - Fragmento de base de datos de usuarios

(2) **Base de datos de información de corpus asociados**: contiene información sobre los corpus a los que tiene acceso cada usuario y sus atributos. Concretamente: tipo de corpus, nombre, abreviatura, tamaño, idioma, etiquetados lingüísticos incorporados. Existen tantos documentos de bases de datos de información de corpus asociados como usuarios registrados en el framework. A continuación, se muestra un ejemplo de fragmento de base de datos de cara tipo de corpus: monolingüe (Figura 48), paralelo multilingüe (Figura 49), comparable (Figura 50) y monolingüe retórico (Figura 51).


```
"mono":
    [
        {
            "size": "102365",
            "name": "Corpus Tolkien",
            "language": "en",
            "abbr": "tolkien",
            "POS": true,
            "SEM": true
        }
    ],
```


Figura 48 - Fragmento de base de datos de información de corpus monolingües asociados

```
"comparable":
    [
        {

            "size": [
                    "20260",
                    "11607",
                    "15643"
            ],
            "abbr": "c-avent",
            "name": "Corpus aventuras",
            "language_comp": [
                    "en",
                    "en",
                    "en"
            ],
            "num": 3,
            "POS": true,
            "SEM": false,
            "subabbrs": [
                    "tolkien",
                    "martin",
                    "sapokowski"
            ],
            "subnames": [
                    "Corpus Tolkien",
                    "Corpus Martin",
                    "Corpus Sapokowski"
            ]
    }
]
```


Figura 49 - Fragmento de base de datos de información de corpus comparables asociados

```
"paralelo":
    [
        {
            "size": "325696770",
            "size_source": "39431862",
            "size_tr": [
                "40461850",
                "43988004",
                "38966669"
            ],
            "abbr": "europarl",
            "name": "EUROPARL EN ES FR IT",
            "num": 3,
            "POS": true,
            "SEM": false,
            "language_source": "en",
            "subname_source": "EUROPARL EN",
            "subabbr_source": "europarl-en",
            "subabbrs_tr": [
                "europarl-es",
                "europarl-fr",
                "europarl-it"
            ],
            "subnames_tr": [
                "EUROPARL-ES",
                "EUROPARL-FR",
                "EUROPARL-IT"
            ],
            "language_tr": [
                "es",
                "fr",
                "it"
            ]
        }
    ]
```


Figura 50 - Fragmento de base de datos de información de corpus multilingües paralelos asociados

```
"retorico":
    [
        {
            "size": "4879",
            "name": "C-GARE 001-005 ES",
            "language": "es",
            "abbr": "c-gare-ret-es-1-5",
            "POS": true,
            "SEM": true
        }
    ]
```


Figura 51 - Fragmento de base de datos de información de corpus monolingües retóricos asociados

6.3.4.3.1.2 Datos CWB

Los datos CWB almacenan los corpus indexados y un descriptor de los mismos. El objetivo es posibilitar consultas y extracción de estadísticas. Por este motivo, está estrechamente ligado al sistema de indexación utilizado, que como se ha mencionado, es la colección de herramientas de CWB.

- Selección de software externo

Las principales razones por la que se ha empleado CWB en lugar de otras estrategias (ver el tipo de búsquedas existentes en el apartado 3.2.5 Cuarta generación para más información) es que proporciona un sistema potente y flexible para indexar y buscar en corpus de tamaño medio o grande, con múltiples capas de anotación a nivel de palabra (Evert & Hardie, 2011, pág. 2).

CWB es una tecnología frecuentemente empleada como *back-end* (capa de datos) en numerosos frameworks para el tratamiento de corpus lingüísticos (ver sección 3.3 Análisis de software disponible para más información), como por ejemplo: Sketch Engine (Kilgarriff, et al., 2014), IntelliText (Wilson, Hartley, Sharoff, & Stephenson, 2010) , CQPweb (Hardie, 2012) o Corpógrafo (Maia & Matos, 2008). La desventaja de emplear CWB es la lentitud con la que resuelve algunas consultas *multi-token* (ver (Evert & Hardie, 2011, pág. 17)), ya que transforma cada expresión

en un autómata finito, sin considerar cuál es el modo más eficiente de llevarlo a cabo. Esto se traduce en una mayor carga computacional y por tanto una disminución de la velocidad con que se lleva a cabo la consulta.

A pesar de esta debilidad, su disponibilidad, bajo licencia GNU General Public License, versión 3 (Free Software Foundation, 2007), su soporte técnico, potencia y flexibilidad hacen que sea la opción más recomendable.

- Implementación

La descripción detallada del modelo de datos de CWB puede consultarse en (Hardie, 2012, pág. 390), (Evert & Hardie, 2011, pág. 5) y (Christ, 1994).

El modelo de datos de CWB se divide en dos elementos básicos (Evert & Hardie, 2011, pág. 3): (1) Corpus indexados y (2) Registros.

(1) Corpus indexados:

Están formado por un conjunto de archivos binarios de datos que representan los textos indexados del corpus, así como los distintas anotaciones lingüísticas y marcados estructurales. Básicamente el modelo de datos de CWB es similar al de una tabla de una base de datos, donde cada fila denota un *token*, y donde cada columna es un atributo de dicho *token*.

Durante la indexación, CWB incorpora una columna inicial donde se almacena la posición que ocupa cada *token* en el corpus para proporcionar un acceso eficiente. Esta posición es secuencial respecto al orden de aparición del texto, hecho que lo diferencia de los sistemas de indexación basados en base de datos (Evert & Hardie, 2011, pág. 6).

CWB soporta tres tipos de datos (Evert & Hardie, 2011, pág. 7):

- ***p-attributes* o atributos posicionales**: Denotan anotaciones a nivel de *token* o palabra.

- ***s-attributes* o atributos estructurales**: Denotan regiones del texto del corpus. A su vez pueden incluir pares atributo clave-valor al estilo XML, que pueden ser indexados en el modelo de datos. Estos atributos clave-valor son subsidiarios del atributo estructural al que pertenecen.
- ***alignment attributes* o atributos de alineación**: Denotan las alineaciones en el caso de corpus paralelos.

Cada atributo posicional es enlazado con el índice asignado a cada palabra, de tal forma que el acceso interno, similar a la consulta de un *array* o vector, resulta muy sencillo.

Por ejemplo, en la Figura 52, el acceso a la etiqueta gramatical del *token* "*The*" sería a través de "*The[POS]*". Los atributos estructurales representan pares de posiciones dentro del corpus. Por ejemplo, el atributo estructural "*s*" con "*id=0"* estaría representado por el par (0,16).

Para explicar estos conceptos se procederá a desarrollar el ejemplo de la Figura 52, que se corresponde con uno de los modelos de datos que se pueden utilizar en el framework:

- **p-attributes**: Etiquetado gramatical, lema y etiquetado semántico
- **s-attributes**: Oraciones (denominadas s) con atributo clave-valor id, y textos (denominadas texto) con atributo clave-valor nombre.
- **alignment attributes**: Clave-valor id de cada oración, en el caso de corpus paralelos.

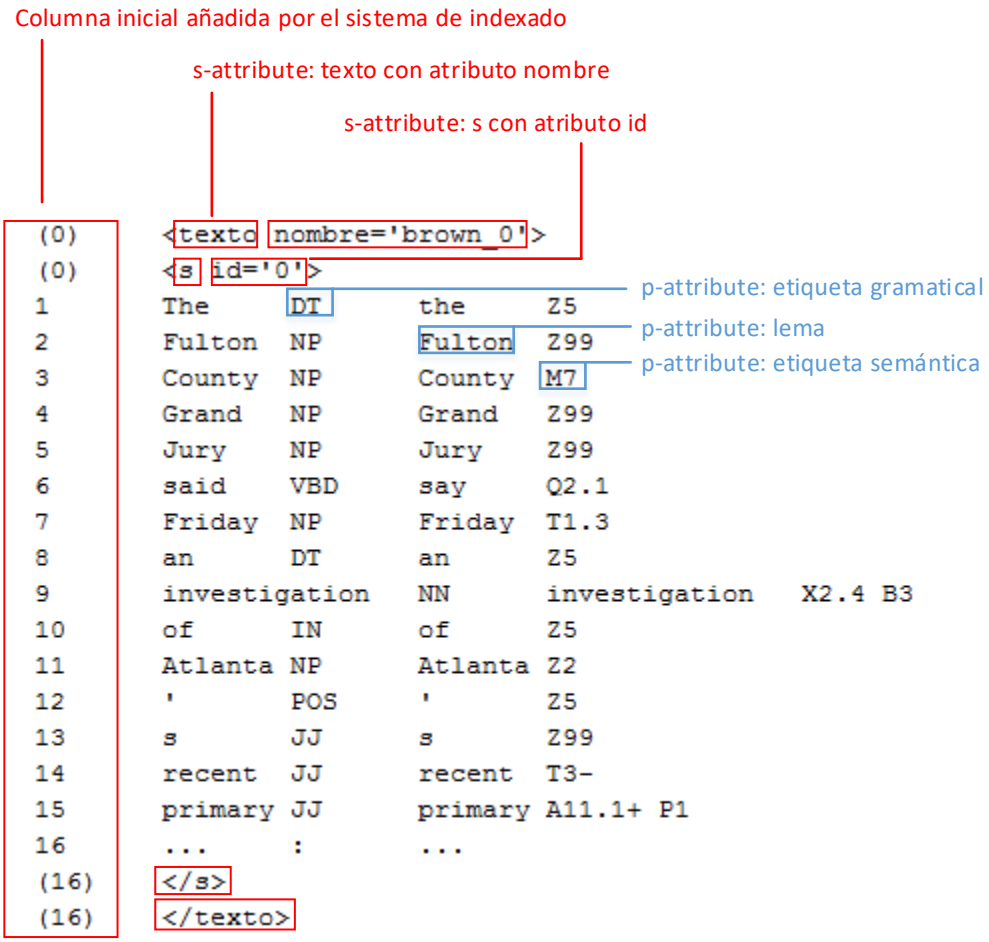


Figura 52 - Fragmento del modelo de datos empleado para almacenar los corpus en formato CWB

(2) Registro:

Contiene la información relativa a la ubicación y a la descripción completa de los datos de cada corpus indexado en el sistema. Un ejemplo de registro se presenta en la Figura 53.

```
## registry entry for corpus HELLO

NAME "Corpus Hello"
# corpus ID (must be lowercase in registry!)
ID   hello
# path to binary data files
HOME /corpora/data/actres/brown
# optional info file (displayed by "info;" command in CQP)
INFO /corpora/data/actres/brown/.info

##
## p-attributes (token annotations)
##

ATTRIBUTE word
ATTRIBUTE pos
ATTRIBUTE lemma
ATTRIBUTE sem


##
## s-attributes (structural markup)
##

# <texto nombre=".."> ... </texto>
# (no recursive embedding allowed)
STRUCTURE texto
STRUCTURE texto_nombre          # [annotations]

# <s id=".."> ... </s>
# (no recursive embedding allowed)
STRUCTURE s
STRUCTURE s_id                  # [annotations]
```

Figura 53 - Contenido del registro de un corpus indexado en formato CWB

6.3.4.3.2 Lógica del framework

La lógica del framework utiliza PHP como el lenguaje de programación principal, empleando un paradigma de programación orientada a objetos. Las razones que motivaron la selección de PHP fueron:

- Su naturaleza eminentemente web, que encaja con los requisitos de acceso a través de internet de un framework de cuarta generación. Esta cualidad hace innecesaria la utilización de otro lenguaje de programación que sirva como enlace entre el servidor y la infraestructura web.

- La extensa funcionalidad de PHP permite que pueda utilizarse para casi cualquier cometido.
- Su sintaxis, simple y legible, lo convierten en un lenguaje de programación fácil de usar, reutilizar y modificar.

Cada una de las clases utilizadas en la programación posee diversos métodos que pueden requerir o no la acción de un software externo y/o desarrollado con lenguajes de programación diferentes a PHP y que, además, pueden emplear otros paradigmas de programación como funcional o por comandos. Cada método es además responsable de adecuar el formato del flujo de datos al requerido para ejecutar su función adecuadamente. Por último, destacar que el orden de ejecución de los procesos es definido en la instanciación de cada clase.

El diagrama de clases completo de *ACTRES Corpus Manager*, así como sus métodos asociados se muestran en el Anexo 2.

Con el fin de facilitar la lectura del documento, se procederá a describir las funcionalidades que *ACTRES Corpus Manager* ofrece, que se corresponden con los casos de uso descritos en el apartado 6.1.2 Especificación de casos de uso; combinando en este caso la extracción de estadísticas de corpus y de una consulta en un mismo punto. Se describirán la clase y/o subclases específicas que se encargan de la implementación de cada caso de uso.

En la Tabla 38 se detalla qué clase es responsable de cada funcionalidad de acuerdo con los casos de uso definidos.

Casos de uso/funcionalidad	**Clase**
Creación de un corpus monolingüe	Clase: CrearCorpus Subclase: CrearCorpusMono
Creación de un corpus comparable	Clase: CrearCorpus Subclase: CrearCorpusComparable
Creación de un corpus bi-/multilingüe paralelo	Clase: CrearCorpus Subclase: CrearCorpusParalelo
Creación de un corpus retórico	Clase: CrearCorpus

	Subclase: CrearCorpusRetorico
Consulta de un corpus	Clase: ConsultarCorpus Clase: CQPConfigData Clase: CQP Clase: CQPInterchangeFile
Estadísticas	Clase: *Estadisticas* Clase: CQPConfigData Clase: CQP Clase: CQPInterchangeFile

Tabla 38 - Funcionalidad y clases encargada de su implementación

6.3.4.3.2.1 Crear corpus lingüísticos

En el sistema de gestión de corpus seleccionado, CWB, el corpus es la unidad fundamental de datos. Este concepto de corpus no tiene por qué adscribirse al concepto de corpus establecido por los lingüistas (Evert & Hardie, 2011, pág. 4). Bajo el punto de vista de la tesis presentada, el corpus que representa la unidad básica de CWB es un corpus monolingüe.

CWB no diferencia distintos tipos de corpus a bajo nivel, es decir, todos los corpus (monolingües, paralelos o comparables) son tratados como corpus monolingües. Nativamente, CWB posee una funcionalidad de incorporación de atributos de alineación (ver página 197) para la creación de corpus paralelos, que permite que las consultas sobre corpus considerados paralelos estén vinculadas entre sí. Por otro lado, no existe ninguna funcionalidad para distinguir corpus comparables.

Para solucionar este déficit, la información relativa a los corpus comparables y retóricos,[42] y para que el proceso resulte semejante, también la información de los corpus monolingües y paralelos multilingües, se incorpora directamente en la "Base de datos de usuario y corpus asociados" descrita en el subapartado 6.3.4.3.1 Datos del framework.

[42] Los corpus retóricos son considerados como corpus monolingües con una anotación estructural más compleja.

Los procesos seguidos para incorporar los distintos tipos de corpus en CWB son idénticos, a excepción de la incorporación de los atributos de alineación. Esto es consecuencia del hecho de que a bajo nivel todos los corpus son tratados como corpus monolingües. No obstante, se ha optado por utilizar subclases específicas para cada tipo de corpus que se puede crear. Los motivos que justifican esta decisión han sido los siguientes:

- Favorecer la modularización del framework. Es decir, si en el futuro CWB u otro sistema de gestión de corpus disponible distingue los tipos de corpus lingüísticos mencionados, su implementación en la lógica de corpus no requerirá de cambios en el diseño del framework en su totalidad, únicamente se modificarán las subclases afectadas. Por ejemplo, si CWB incorpora una funcionalidad para tratar corpus comparables, únicamente la subclase *CrearCorpusComparables* y sus métodos serían modificados.
- Aunque los procesos llevados a cabo para crear los distintos tipos de corpus son muy similares, existe una lógica asociada que no lo es. Por ejemplo:
    - Crear un corpus comparable requiere la incorporación de un número de subcorpus definido por el usuario, que son almacenados en el sistema de almacenamiento temporal del servidor en una configuración concreta. Tratar esta diferencia de organización a la hora de ejecutar los distintos métodos implica la necesidad de implementación de una lógica adicional.
    - El nombre descriptivo del corpus, los subcorpus y los documentos de los mismos varía de acuerdo con el tipo de corpus creado. Esto también requiere de una programación adicional.
    - La creación de un corpus paralelo implica la creación de métodos adicionales como *alinear* o *crearIndicesDeAlineacionParalelo*.

Estas condiciones en su conjunto justifican la decisión de emplear distintas clases con sus métodos asociados en lugar de reutilizar el código en una sola clase.

La clase principal encargada de crear corpus lingüísticos es CrearCorpus, clase padre a su vez de CrearCorpusMono (crear corpus monolingüe), CrearCorpusComparable (crear corpus comparable), CrearCorpusParalelo (crear corpus bi-/multilingüe paralelo) y CrearCorpusRetorico (crear corpus monolingüe retórico).

En la Figura 54 se muestra la relación entre la clase CrearCorpus y sus clases hijas con los datos del framework ("Datos de usuario y corpus asociados" y "Datos CWB").

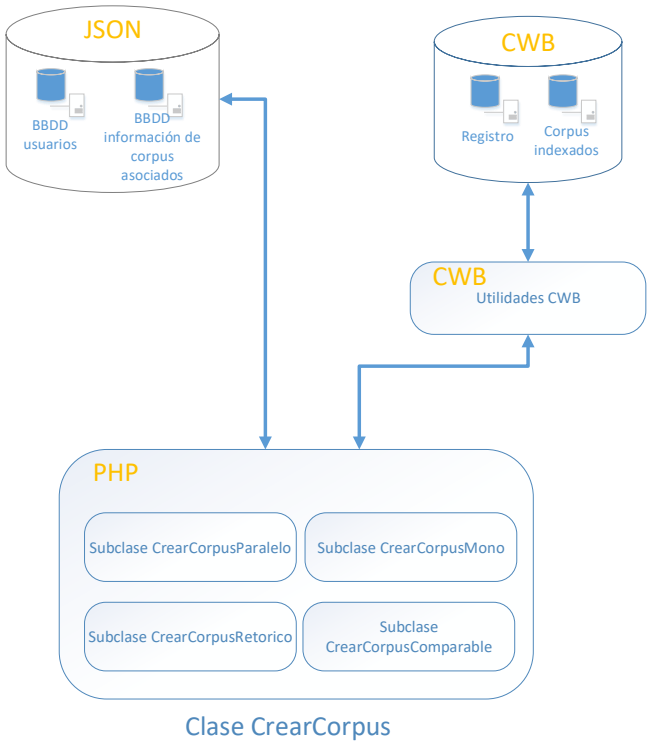


Figura 54 - Clases implicadas en la funcionalidad "Crear Corpus" y su relación con los datos del framework (Datos de usuario y corpus asociados y Datos CWB)

Con el objetivo de facilitar la comprensión de la funcionalidad de la clase CrearCorpus y de sus clases hijas, se describirá de forma genérica cada una de las actividades realizadas por los componentes que forman el flujo de ejecución, para

posteriormente asociarlas directamente con los métodos de cada clase, explicando además las singularidades de acuerdo con el tipo de corpus creado.

I) Tratar documentos y datos del corpus

La primera actividad realizada para crear un corpus es el establecimiento como variables de la lógica del framework de:

- Los datos de los corpus introducidos por el usuario, que incluyen: tipo de corpus, nombre, abreviatura, idiomas, etiquetados y documentos del corpus en formato txt.
- Las rutas a recursos software del servidor: variables de entorno, ubicación de binarios y programas externos.

Esta actividad ha sido realizada íntegramente en PHP y su acción está dividida en dos métodos:

(1) Método genérico para establecer las rutas a recursos software del servidor perteneciente a la clase padre CrearCorpus.

(2) Método específico para cada tipo de corpus, perteneciente a cada una de las subclases específicas. Esto es debido a que cada tipo de corpus posee unos datos concretos:

- Corpus monolingüe:
  - nombre del corpus
  - abreviatura
  - idioma
  - tipos de etiquetados
  - documentos del corpus en formato txt
- Corpus comparable:
  - nombre del corpus
  - nombres de los subcorpus
  - abreviaturas de los subcorpus
  - idiomas de los subcorpus

- tipos de etiquetado
- documentos de los subcorpus en formato txt

- Corpus paralelo:
  - nombre del corpus
  - nombre de los subcorpus
  - abreviaturas de los subcorpus
  - idiomas de los subcorpus
  - tipos de etiquetado
  - documentos de los subcorpus en formato txt:
    - Con el fin de saber qué textos de cada subcorpus original se corresponden con los textos de cada uno de los subcorpus traducidos, resulta indispensable que los textos originales y traducidos comiencen por la misma secuencia textual o número seguido de una barra baja (_). Por ejemplo:

Idioma	**Texto 1**	**Texto 2**
Español (es)	1_Hola.txt	2_Adios.txt
Inglés (en)	1_Hello.txt	2_Bye.txt
Francés (fr)	1_Bonjour.txt	2_Au revoir.txt

Tabla 39 - Configuración necesaria en los nombres de los documentos de los corpus paralelos

- Corpus retórico:
  - nombre del corpus
  - abreviatura
  - idioma
  - etiquetado retórico manual[43]

[43] Aunque los datos se obtendrían desde el interfaz de usuario, perteneciente por tanto a la lógica de la vista, se ha creído adecuado explicar el etiquetado retórico de forma conjunta en la lógica del framework.

- tipos de etiquetado adicionales
- documentos del corpus en formato txt

En las siguientes tablas (Tabla 40 y Tabla 41) se puede apreciar toda la información de forma resumida:

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus monolingüe	CrearCorpus	establecerDatosIniciales()	PHP
Corpus comparable	CrearCorpus	establecerDatosIniciales()	PHP
Corpus paralelo multilingüe	CrearCorpus	establecerDatosIniciales()	PHP
Corpus retórico monolingüe	CrearCorpus	establecerDatosIniciales()	PHP

Tabla 40 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Tratar documentos y datos" (I)

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus monolingüe	CrearCorpusMono	establecerDatosMono()	PHP
Corpus comparable	CrearCorpusComparable	establecerDatosComparable()	PHP
Corpus paralelo multilingüe	CrearCorpusParalelo	establecerDatosParalelo()	PHP
Corpus retórico monolingüe	CrearCorpusRetorico	establecerDatosRetorico()	PHP

Tabla 41 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Tratar documentos y datos" (II)

Por último, mostrar el flujo de datos detallado de la actividad (Figura 55):

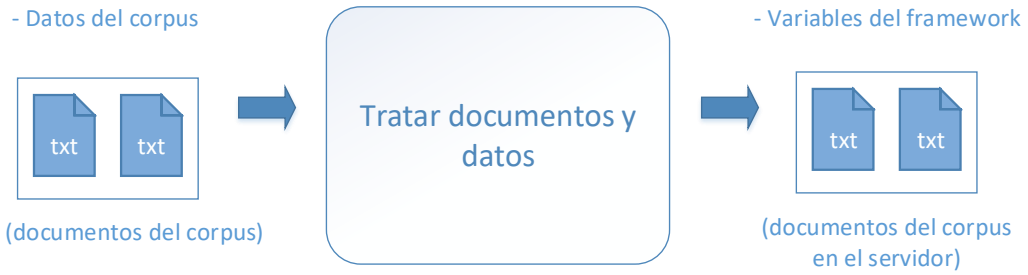


Figura 55 - Flujo de datos de la actividad "Tratar documentos y datos"

En el caso de los corpus retóricos, las estructuras retóricas son incluidas como elementos internos de los textos que forman el corpus, la lógica del framework es consciente de ello y las mantiene inalterables hasta el proceso de marcado. Se explicará en detalle en el IX) Etiquetador retórico.

II) Limpiar de documentos del corpus

La siguiente actividad a realizar es la limpieza de cada uno de los documentos del corpus o subcorpus con el fin de eliminar:

- Etiquetas HTML.
- Hipervínculos.
- Caracteres no válidos en XML.

Esta actividad también está íntegramente desarrollada en PHP. Los tipos de corpus, clases, métodos asociados y lenguajes de programación de esta actividad se muestran en la Tabla 42.

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus monolingüe	CrearCorpusMono	cleanMono()	PHP
Corpus comparable	CrearCorpusComparable	cleanComparable()	PHP
Corpus paralelo multilingüe	CrearCorpusParalelo	cleanParalelo()	PHP

Corpus retórico monolingüe	CrearCorpusRetorico	cleanRetorico()	PHP

Tabla 42 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Limpiar de documentos del corpus"

El flujo de datos detallado de esta actividad es el siguiente (Figura 56):

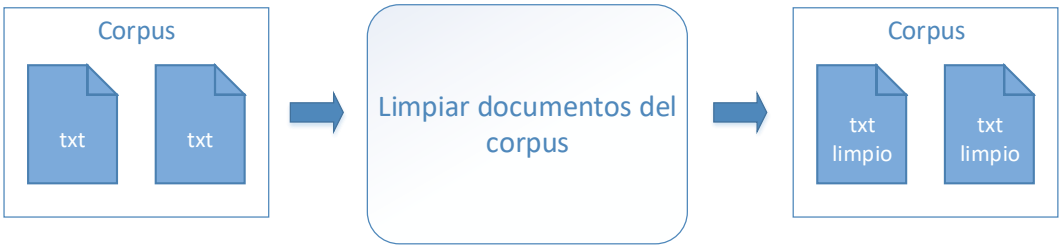


Figura 56 - Flujo de datos de la actividad "Limpiar documentos del corpus"

III) Codificar documentos en UTF-8

El siguiente proceso consiste en convertir los textos al formato de codificación de caracteres Unicode, más concretamente al estándar UTF-8, si es que este no es ya el formato del documento (ver sección 3.1 Problemas y limitaciones para más información).

El codificado en UTF-8 es un proceso más complejo de lo esperado puesto que es necesario conocer con exactitud la codificación actual del documento para realizar la conversión de forma adecuada. En muchas ocasiones el usuario desconoce el formato de sus documentos, por lo que se optó por establecer un detector automático para tal fin. Sin embargo, los detectores de codificación tampoco son 100% precisos porque al reconocer un carácter perteneciente a una codificación específica, dan por hecho que ese es el estándar empleado en el resto del documento (por ejemplo, el comando *iconv* del *shell* de Linux). En muchas ocasiones esto no es así, dando lugar a falsos positivos, sobre todo con el estándar ISO-8859-1.

La implementación del codificador está realizada en *bash* y en Python (Figura 57).

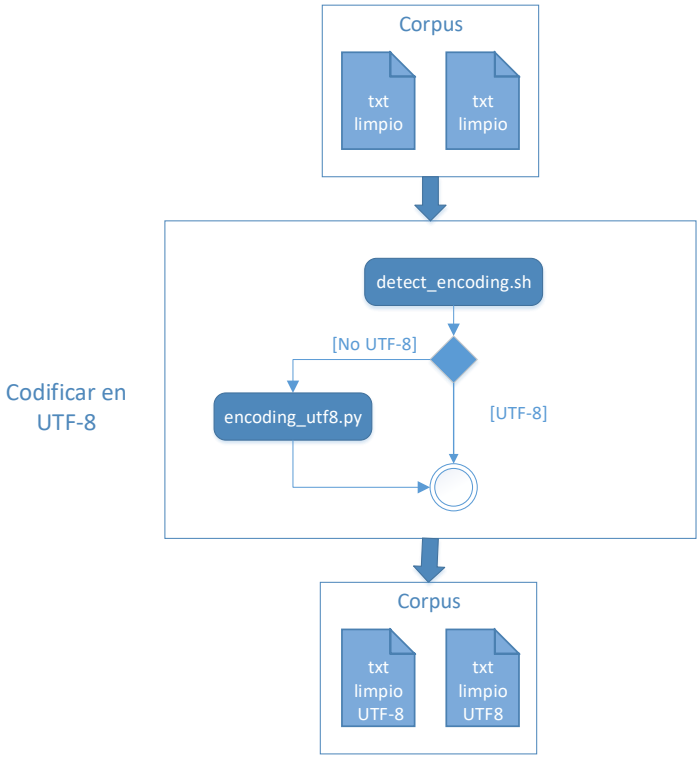


Figura 57 - Flujo de datos de la actividad "Codificar documentos en UTF-8"

Siguiendo el flujo de la Figura 57, a través de un script *bash*, denominado *detect_encoding.sh*, se detecta la codificación del documento, para posteriormente convertirlo en UTF-8 a través de un script en Python denominado *encoding_utf8.py* que utiliza funcionalidades del módulo *getopt*.

Los métodos que gestionan la acción de la codificación en UTF-8 son los siguientes (Tabla 43):

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus monolingüe	CrearCorpusMono	UTF8Mono()	PHP+bash+ Python
Corpus comparable	CrearCorpusComparable	UTF8Comparable()	PHP+bash+ Python

Corpus paralelo multilingüe	CrearCorpusParalelo	UTF8Paralelo()	PHP+bash+ Python
Corpus retórico monolingüe	CrearCorpusRetorico	UTF8Retorico()	PHP+bash+ Python

Tabla 43 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Codificar documentos en UTF-8"

IV) Tokenizar documentos y tokenizar documentos paralelos

Tokenización es un término procedente de la lengua inglesa que se refiere al proceso de división de una fuente dada en *tokens* o símbolos.

En esta actividad existen dos casos diferenciados:

(1) Tokenizar documentos

Esta tokenización es aplicable sólo para corpus monolingües, comparables y retóricos. En este caso, el proceso de tokenización comprende la división de los documentos del corpus en oraciones. Para mejorar la precisión de la tokenización es necesario emplear un tokenizer que se apoye en recursos lingüísticos específicos para cada idioma.

Durante el proceso de *tokenización* es añadido un identificador único numerado para cada oración, así como etiquetas para delimitar el texto, con otro identificador que se corresponde con el nombre del documento.

Por ejemplo, el siguiente fragmento de texto denominado "*ejemplo"*:

Haría falta a Wenger para ganar la FA Cup y tener un increíble final de Premier League para que él pudiera transformar el estado de ánimo en el club y seguir adelante. Fuentes habían creído que era "50-50" si él permanecería en los Emiratos, pero los acontecimientos del miércoles por la noche han cambiado dramáticamente eso.

Se convierte en el fragmento siguiente:

<texto nombre="ejemplo">
1_ejemplo Haría falta a Wenger para ganar la FA Cup y tener un increíble final de Premier League para que él pudiera transformar el estado de ánimo en el club y seguir adelante.
2_ejemplo Fuentes habían creído que era "50-50" si él permanecería en los Emiratos, pero los acontecimientos del miércoles por la noche han cambiado dramáticamente eso.
</texto>

La solución software seleccionada para llevar a cabo este proceso es la utilización del toolkit NLTK (Bird, Klein, & Loper, 2009). NLTK (ver apartado 3.3.2 Toolkits, *suites* y similares para más información) está desarrollado en Python y permite un fácil acceso a recursos lingüísticos predefinidos para cada idioma. El empleo de NLTK se impuso sobre la creación de soluciones propias de PHP como consecuencia de la gran cantidad de recursos lingüísticos que NLTK ofrece por defecto. También se descartaron lenguajes de programación que suelen ser empleados en tareas relacionadas con el procesamiento de lenguaje natural como Perl, principalmente porque es un lenguaje extremadamente difícil de mantener y reutilizar (Bird, Klein, & Loper, 2008, Prefacio).

En el caso de los corpus retóricos, las estructuras reconocidas por el usuario son obviadas y no interfieren en el proceso de tokenización.

Los métodos que gestionan la acción del tokenizer en cada clase se muestran a continuación (Tabla 44):

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus monolingüe	CrearCorpusMono	tokenizarMono()	PHP+Python
Corpus comparable	CrearCorpusComparable	tokenizarComparable()	PHP+Python
Corpus retórico monolingüe	CrearCorpusRetorico	tokenizarRetorico()	PHP+Python

Tabla 44 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Tokenizar documentos"

Por último, mostrar el flujo de datos de entrada y salida de esta actividad (Figura 58):

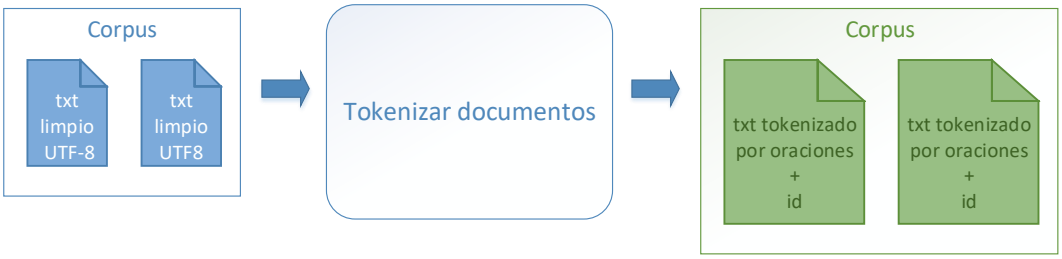


Figura 58 - Flujo de datos de la actividad "Tokenizar documentos"

(2) Tokenizer de corpus paralelos:

El tokenizer de corpus multilingües paralelos, requiere de la acción del tokenizer de oraciones previamente descrito y además de un tokenizer de palabras. La solución software adoptada para llevar a cabo la tokenización por palabras fue el empleo de un script Perl procedente de Europarl (Koehn, 2005) modificado. La utilización de un script Perl en lugar de usar soluciones existentes en NLTK se debe, en exclusiva, a que el tokenizer por palabras de NLTK basa su funcionamiento en expresiones regulares procedentes del *Penn Treebank Corpus* (NLTK Project,

2017), que puede originar pequeños problemas en palabras de idiomas distintos del inglés. Además, el tokenizer por palabras de NLTK utiliza convecciones poco frecuentes en otros tokenizers para idiomas distintos del inglés, como la distinción y transformación de comillas de apertura y cierre.

Por ejemplo, el resultado final del texto utilizado en el ejemplo anterior después de la acción del tokenizer por palabras y por oraciones de corpus paralelos es el siguiente:

Haría falta a Wenger para ganar la FA Cup y tener un increíble final de Premier League para que él pudiera transformar el estado de ánimo en el club y seguir adelante .
Fuentes habían creído que era " 50-50 " si él permanecería en los Emiratos , pero los acontecimientos del miércoles por la noche han cambiado dramáticamente eso .

Se puede que apreciar que cada palabra/*token* está separada por un espacio. Ver concretamente los signos de puntuación o la eliminación de dobles espacios. Además, este tokenizer se diferencia del tokenizer anterior en que no añade el identificador único a cada oración, ya que no es necesario para la siguiente actividad (*alinear*) desarrollada en la creación de corpus paralelos.

El método que gestiona la acción del tokenizer de corpus paralelos en la subclase CrearCorpusParalelo se muestra a continuación (Tabla 45):

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus paralelo multilingüe	CrearCorpusParalelo	tokenizarParalelo()	PHP+Python+perl

Tabla 45 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad “Tokenizar documentos paralelos”

Por último, mostrar el flujo de datos de entrada y salida de la actividad tokenizar documentos para corpus multilingües paralelos (Figura 59):

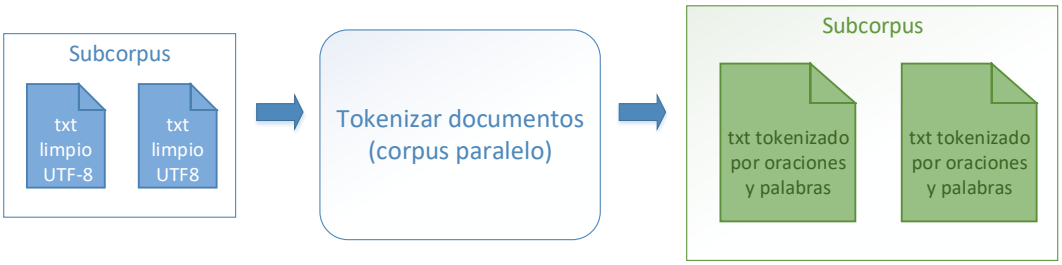


Figura 59 - Flujo de datos de la actividad "Tokenizar documentos paralelos"

V) Alinear

Esta actividad sólo está disponible en la creación de corpus paralelos. El alineado es un proceso nuclear, definitorio e indispensable en la construcción de corpus paralelos. Tal y como se ha descrito con anterioridad, consiste en emparejar unidades de un texto original, ya sean palabras, oraciones o párrafos, con su respectiva traducción.

Empleando el ejemplo utilizado en la actividad I) Tratar documentos y datos del corpus (ver página 205) para corpus paralelos, en la siguiente figura se muestra un ejemplo gráfico de alineación por oraciones. Nótese que una oración en un idioma puede corresponderse con 0, 1 o más oraciones en la traducción correspondiente en otro idioma. En el caso de la Figura 60, una oración en español se alinea con dos oraciones en inglés y sólo con una en francés.

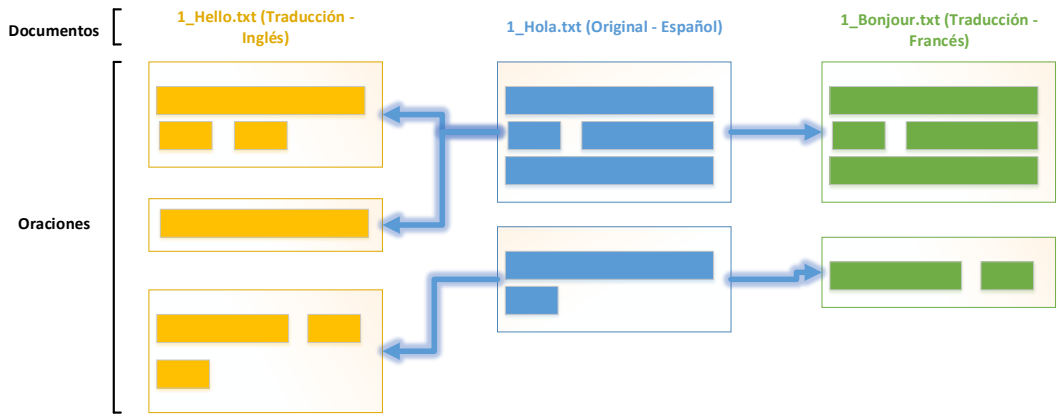


Figura 60 - Ejemplo gráfico de alineación de corpus

- Selección de software externo

Existen dos tipos de estrategias fundamentales de alineación: alineadores no supervisados o alineadores supervisados. Los alineadores no supervisados se caracterizan por no requerir la asistencia del usuario durante su ejecución, en contraposición a los alineadores supervisados, que requieren de la intervención del usuario para validar los distintos emparejamientos.

Se intenta dotar a *ACTRES Corpus Manager* de la mayor autosuficiencia posible, es decir, que el usuario lingüista no necesite asistencia técnica en ningún momento durante la creación de un corpus lingüístico y que el proceso sea automático dentro de lo posible. Por esta razón se utilizarán alineadores no supervisados

Por otra parte, existen dos tipos principales de alineadores: (1) alineadores por palabras y (2) alineadores por oraciones. El alineador por oraciones es el que menos tasa de error posee y es por tanto el tipo seleccionado en esta tesis doctoral.

La alineación por oraciones es relativamente sencilla y ofrece a su vez dos estrategias principales que pueden ser empleadas de forma aislada o combinada, (1) estrategias de alineación basadas en longitud de las oraciones y (2) estrategias de alineación basadas en correspondencias léxicas.

Estas estrategias por separado obtienen resultados similares (Tiedemann, 2011, pág. 56), que mejoran combinándolos (Tiedemann, 2011, pág. 53).

Los errores que los alineadores por oraciones no pueden evitar son los derivados del "ruido" por discrepancias entre ambos subcorpus, como por ejemplo secciones no traducidas. Estas secciones precisan un tratamiento manual previo o durante el alineado (Tiedemann, 2011, pág. 56), no implementado en *ACTRES Corpus Manager*.

Las pruebas realizadas no han empleado las últimas tendencias en el desarrollo de alineadores por los siguientes motivos:

(1) El desarrollo de aplicaciones relacionadas con la alineación se centra en la actualidad en la creación de alineadores por palabras. El objetivo principal es emplearlos en el subcampo de las ciencias de la computación del *Machine Learning*, y más concretamente en el desarrollo de aplicaciones relacionadas con la traducción automática o creación de diversos recursos bilingües.

(2) La alineación por oraciones tolera errores de mejor forma que las alineaciones por palabras, donde un simple fallo resulta muy evidente.

(3) La alineación por oraciones se puede realizar de forma supervisada o semi supervisada a través de interfaces y programas realizados para tal fin. Este proceso, aunque costoso, no es una tarea irrealizable. Un ejemplo del primero es bitext2tmx (Sánchez-Martínez, et al., 2014) y del segundo TCA (Hofland & Johansson, 1998).

Además de estas dos condiciones, alineado no supervisado y alineado por oraciones, el software seleccionado tenía que cumplir los siguientes requisitos adicionales:

- Tiempo de ejecución tolerable sobre corpus de gran tamaño.

- Facilidad de uso, en cuanto a que no sean necesarios altos conocimientos en *Machine Learning* para su empleo, ni configuraciones complejas para su instalación en un servidor web estándar.
- Funcionamiento óptimo con recursos lingüísticos limitados, es decir, que no requiera amplios recursos lingüísticos para un funcionamiento ideal.

Bajo estas premisas el software seleccionado fue hunalign (Varga, et al., 2005). hunalign ofrece unos tiempos de ejecución tolerables y unos resultados satisfactorios. Además, posibilita que en el futuro se pueda incorporar la revisión de los resultados de una alineación sin necesidad de construir una nueva interfaz para ello. Se realizaría a través de la aplicación basada en hunalign denominada LF Aligner (Farkas, 2010).

A continuación, se detallan las diversas alternativas que se estudiaron junto a hunalign:

- Algoritmo Gale-Church (Gale & Church, 1993): el motivo por el que no se utilizó el algoritmo de Gale-Church fue que el software seleccionado, hunalign (Varga, et al., 2005), combinado con información derivada de correspondencias léxicas.
- Bluealign (Sennrich & Volk, 2010): Bluealign necesita una traducción automática de algunos de los textos que se pretenden alinear, es decir, o bien del texto original o de su traducción. Esta traducción automática puede obtenerse, por ejemplo, del Traductor de Google (Google Inc, 2017b). No obstante, estas traducciones automáticas precisan de acceso a una API para poder automatizar el proceso, y en ocasiones este acceso es de pago. Además, no siempre es posible obtener traducciones automáticas sobre textos formados por cientos de miles de palabras, ya sea por cuestiones de tiempo o limitaciones técnicas del software empleado.

- Gargantua (Braune & Fraser, 2010): Las pruebas realizadas demostraron que no funcionaba correctamente en las condiciones planteadas en los experimentos.[44]
- cwb-align, es el alineador integrado por defecto en CWB. De acuerdo con su creador:

  > *"cwb-align isn't a particularly sophisticated sentence aligner, so it's likely to get some cases wrong. You may be seeing particularly bad performance if you're using the default parameter settings, which are intended for related languages and are based on sentence length (in characters), character n-gram counts and identical words."* (Evert, 2014)

  Esta declaración asevera que cwb-align no es sofisticado y que requiere de configuraciones distintas para cada par de idiomas si se quieren obtener unos resultados óptimos. Por estos motivos se desechó su utilización.

Para más información acerca del alineado consultar el libro "*Bitext alignment*" de Jörg Tiedemann (Tiedemann, 2011).

- Implementación

Para llevar a cabo el alineado con hunalign son necesarios:

  - Los documentos del corpus tokenizados por oraciones y por palabras procedentes de la actividad Tokenizer de corpus paralelos.
  - Diccionarios bilingües que, aunque no obligatorios, mejoran el rendimiento. Para este cometido se utilizaron diccionarios bilingües en ambas direcciones a partir de los recursos proporcionados por LF Aligner

[44] Corpus paralelo formado por dos subcorpus, con siete documentos cada uno, con un tamaño de 57.078 palabras y 59.934 palabras respectivamente. Las pruebas se realizaron sobre un servidor Ubuntu 16.04.1 alojado en una máquina virtual con 1Gb de RAM y un procesador Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz.

(Farkas, 2010)[45] y de material léxico del grupo ACTRES. El diccionario ha de poseer el formato "término en idioma 1 @ término en idioma 2" (Ver Figura 61).

```
light grey @ grisáceo
amber @ ámbar
amber @ ambarino
blue @ azul
bluish @ azulado
indigo @ añil
orange @ naranja
```

Figura 61 - Formato del diccionario de hunalign

Hay que tener presente que un corpus paralelo está formado por dos o más subcorpus, siendo uno de ellos un subcorpus original y el resto, uno o más, traducciones del mismo en otros idiomas. hunalign ha de ejecutarse sobre cada par "subcorpus original – subcorpus traducido" existente, y más concretamente sobre cada par "documento del subcorpus original – documento correspondiente del subcorpus traducido".

La Figura 62 muestra el proceso de alineación de un corpus paralelo multilingüe con hunalign. En el ejemplo, el corpus paralelo está formado por tres subcorpus: un subcorpus original en español formado por dos documentos y un subcorpus en francés y otro en inglés con sus respectivas traducciones.

[45] LF Aligner proporciona diccionarios multilingües basados en información procedente de los glosarios de Wiktionary (Wikimedia Foundation, 2017) y Eurovoc (Unión Europea, 2017).

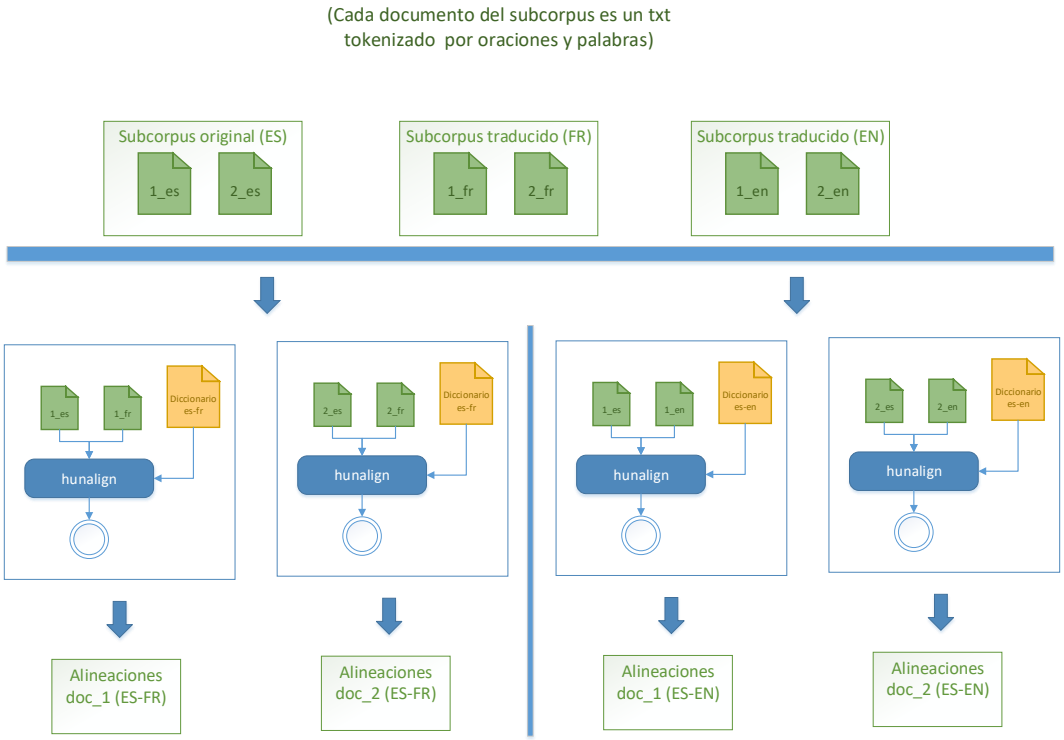


Figura 62 - Ejemplo de ejecución de hunalign

Durante este proceso se afrontaron diversos problemas:

- hunalign posee una desventaja fundamental, consume mucha memoria al tratar textos de más de 10.000 oraciones (Toral, Poch, Pecina, & Thurmair, 2012, pág. 58) (Varga, 2015). Para superar este problema se utilizó la funcionalidad de alineado parcial que permite dividir la tarea principal en tareas más pequeñas. Este proceso es comúnmente denominado "procesamiento por lotes".
- hunalign permite obtener la salida del alineado mediante (1) pares de índices correspondientes a las distintas oraciones, o mediante (2) pares de oraciones originales y sus correspondientes oraciones traducidas.
  - La primera opción (1) es obviamente la más adecuada, ya que era mucho más eficiente y se podía integrar fácilmente en el flujo de creación del corpus.

- Sin embargo, al utilizar la funcionalidad de procesamiento por lotes de hunalign, éste en ocasiones añade índices de oraciones inexistentes al final del procesamiento de cada uno de los lotes. Estos índices representan a oraciones que no existen en el corpus, luego la integración de los índices en el sistema de indexación provoca errores.
- Durante las pruebas realizadas, había ocasiones en lo que los índices finales se correspondían con oraciones existentes, luego el error no sucedía.
- Para explicar el problema, se dispone de un corpus paralelo formado por dos documentos y con cinco oraciones cada uno. La salida del alineado sin procesamiento por lotes se muestra en la Figura 63 y la de con procesamiento por lotes en la Figura 64. Comparando ambas imágenes se aprecia la adición de un índice en el procesamiento por lotes en la Figura 64.

Figura 63 - Comparación de salida de hunalign sin procesamiento por lotes

Figura 64 - Comparación de salida de hunalign con procesamiento por lotes

- Para prevenir estos errores, la salida del alineado seleccionada fue la de los pares de oraciones original - traducción. Esta decisión implicó la creación de un método adicional (*crearIndicesDeAlineacionParalelo*) que se encarga de crear los índices a partir de las alineaciones detectadas por el alineador, formadas por las oraciones que conforman cada uno de los subcorpus.
- En la Figura 65 se muestra el funcionamiento del método creado:

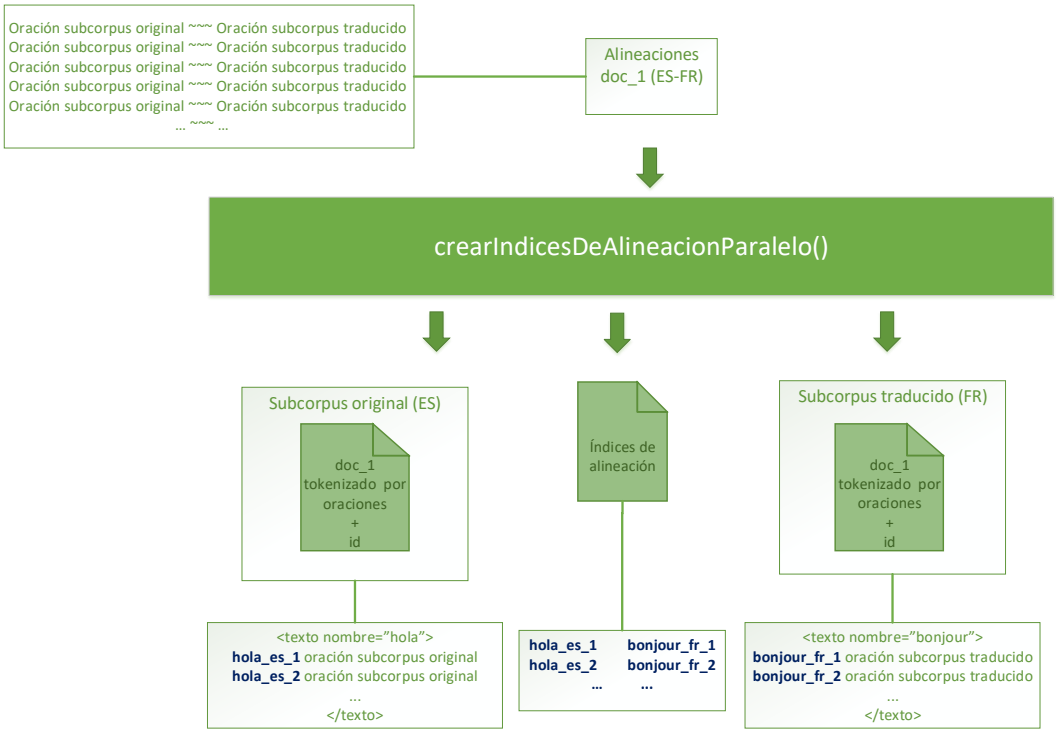


Figura 65 - Funcionamiento del método *crearIndicesDeAlineacionParalelo*

Los métodos que gestionan que gestionan la acción del alineador son los siguientes (Tabla 46):

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus paralelo multilingüe	CrearCorpusParalelo	alinear()	PHP+ Python+ hunalign
Corpus paralelo multilingüe	CrearCorpusParalelo	crearIndicesDeAlineacionParalelo()	PHP

Tabla 46 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Alinear"

Por último, mostrar el flujo de datos de entrada y salida detallado de esta actividad exclusiva de la creación de corpus bi-/multilingüe paralelos (Figura 66):

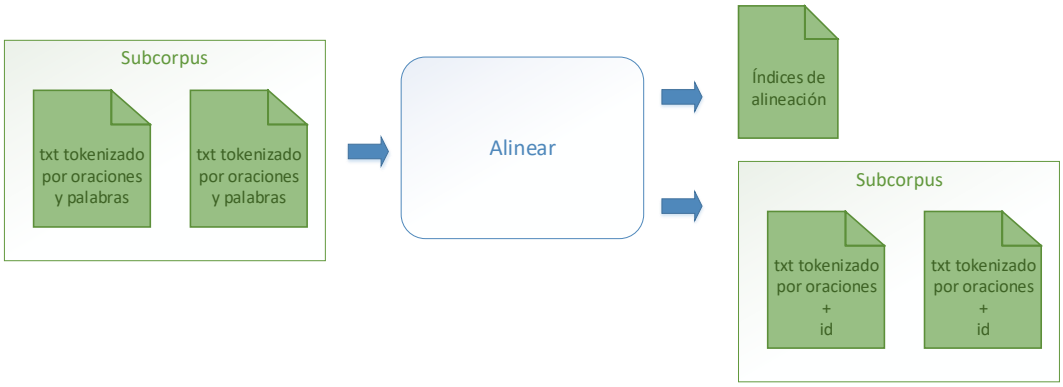


Figura 66 - Flujo de datos de la actividad "Alinear"

VI) Marcar

El marcado del corpus lingüístico consiste en incluir los metadatos junto a las partes en que se divide el texto (palabras, oraciones, capítulos si los hubiera, etc.) en un formato estandarizado, en este caso, en XML. Se trata de un proceso muy importante porque permite organizar el corpus en un formato limpio, reconocible y compatible con otros sistemas de corpus o cualquier otro software.

En *ACTRES Corpus Manager*, la inclusión de metadatos a nivel de documentos de corpus se ha eliminado, ya que implicaba una demora sustancial en la preparación

de los documentos del corpus. En versiones futuras se implementará esta funcionalidad a través de su inclusión en el interfaz.

La actividad de marcado se realiza sobre cada uno de los textos que forman el corpus a través del lenguaje de programación PHP. Técnicamente, el marcado en *ACTRES Corpus Manager* consiste únicamente en la inclusión de los identificadores de oraciones y de las secciones retóricas en el caso del corpus monolingüe retórico, ya que los marcadores que delimitan cada uno de los textos ya fueron incluidos en la actividad V) Alinear en el caso de corpus paralelo, y en la actividad Tokenizar documentos en el resto de casos.

En la Figura 67 se muestra un ejemplo de marcado sobre un texto que forma parte de un corpus monolingüe:

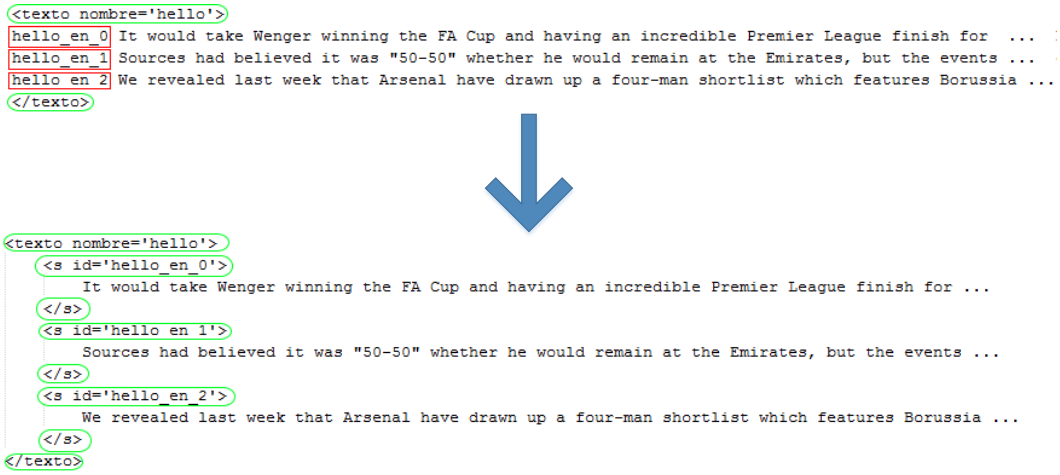


Figura 67 - Ejemplo de marcado de documento en *ACTRES Corpus Manager*

Los métodos que gestionan la acción del marcador en cada clase se muestran a continuación (Tabla 47):

Tipo de corpus	**Clase**	**Método**	**Recursos técnicos**
Corpus monolingüe	CrearCorpusMono	marcarMono()	PHP

Corpus comparable	CrearCorpusComparable	marcarComparable()	PHP
Corpus paralelo multilingüe	CrearCorpusParalelo	marcarParalelo()	PHP
Corpus retórico monolingüe	CrearCorpusRetórico	marcarRetórico()	PHP

Tabla 47 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Marcar"

Por último, el flujo de datos de entrada y salida de esta actividad para los corpus monolingües, bi-/multilingües paralelos y comparables se muestra en la Figura 68, y para corpus monolingües retóricos en la Figura 69.

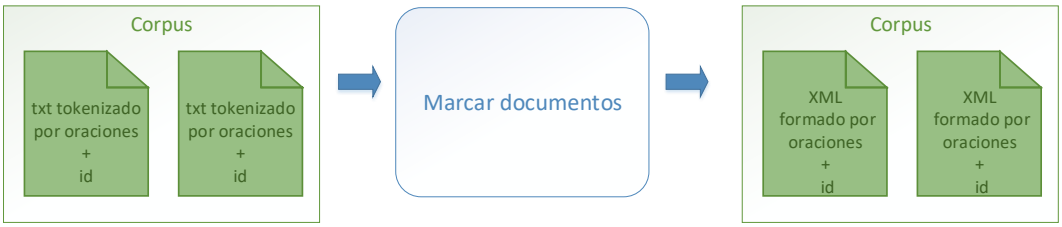


Figura 68 - Flujo de datos de la actividad "Marcar"

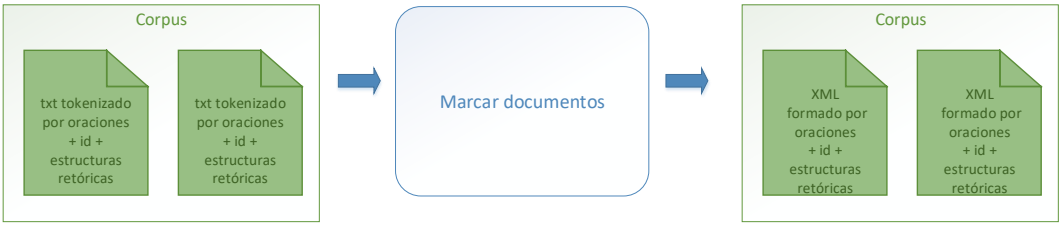


Figura 69 - Flujo de datos de la actividad "Marcar" con estructuras retóricas

VII) Etiquetador gramatical

- Selección de software externo

Este tipo de etiquetado consiste en asignar a cada una de las palabras una categoría gramatical. El desarrollo de aplicaciones para realizar etiquetado gramatical no supervisado ha sido una de las tareas que el NLP trató de resolver desde sus comienzos y ha evolucionado en paralelo al desarrollo de NLP, es decir, ha ido de las reglas manuales a la utilización de modelos probabilísticos (ver apartado 2.2.2 Corpus lingüístico y NLP para más información).

La precisión que la documentación otorga a la mayor parte de los etiquetadores gramaticales oscila entre el 90% y el 98% a nivel de palabra (Manning, 2011, pág. 171), (Paroubek, 2007, pág. 113) (Giesbrecht & Evert, 2009, pág. 27) que, considerando la precisión otorgada al ser humano, que es del 97% (Manning, 2011, pág. 172), es un valor muy alto.

Sin embargo, esta precisión es un poco engañosa. Extrapolando la precisión a nivel de oraciones de un etiquetador con 96% de precisión a nivel de palabra, se reduce al 54,2%. Es decir, casi 1 de cada 2 oraciones contiene un error de etiquetado (Paroubek, 2007, pág. 113).

Además, la precisión reclamada por los distintos etiquetadores gramaticales depende de factores tales como el tamaño de las muestras sobre la que se realizan las pruebas y sobre todo el género al que pertenecen los textos (Paroubek, 2007, pág. 111) tal y como demuestran la proliferación de estudios específicos que tratan de medir la efectividad de un etiquetador gramatical en un tipo concreto de textos como (Tian & Lo, 2015), (Giesbrecht & Evert, 2009) o (Valverde, 2011).

Siguiendo los criterios generales establecidos al comienzo de este punto (multilingüe, usable, preciso y flexible) se procedió a seleccionar el software encargado del etiquetado gramatical. Al final del proceso se obtuvieron tres candidatos principales:

(1) Treetagger (Schmid, 1995): es el etiquetador más utilizado y uno de los que goza de mayor reputación entre los lingüistas consultados. Fue uno de los primeros etiquetadores multi idioma, y su formato de etiquetado (tabulado, una entrada por línea) está ampliamente extendido.

(2) Stanford POS Tagger (Toutanova, Klein, Manning, & Singer, 2003): tiene una elevada efectividad en lengua inglesa (97.32% (Manning, 2011, pág. 173)) y altos rendimientos en cualquier otra lengua con la que haya sido entrenado. Además, se actualiza frecuentemente. Como desventaja, mencionar que la lentitud es sin duda uno de sus mayores hándicaps (Cer, De Marneffe, Jurafsky, & Manning, 2010) y que requiere de grandes datos de entrenamiento para un funcionamiento óptimo.

(3) Módulo Part-of-Speech Tagger de FreeLing (Padró & Stanilovsky, 2012): en base a las pruebas realizadas es el mejor etiquetador gramatical en español y posee unos excelentes resultados en lengua inglesa. Como contrapartida, forma parte de la *suite* de herramientas FreeLing (Padró & Stanilovsky, 2012), lo que le resta usabilidad.

Finalmente, Treetagger fue el software seleccionado. Las razones que sostienen esta decisión son:

- Las opiniones de los lingüistas consultados aconsejan su uso en caso de tratar con múltiples idiomas.[46]
- Existencia de numeroso material disponible etiquetado con Treetagger.[47]
- Alto equilibrio entre eficiencia y precisión, junto con el soporte de multilingüismo.

[46] Encuesta realizada en base a conversaciones con personal investigador del grupo de Investigación consolidado ACTRES, estancia en centro de investigación UCREL perteneciente a *Lancaster University* – Universidad de Lancaster, entre otras.

[47] Por ejemplo, el corpus P-ACTRES 2.0 está etiquetado íntegramente con Treetagger.

- Su selección propiciaba el empleo de un único software para el etiquetado gramatical en cualquier lengua.

Entre las limitaciones de Treetagger destacan su escaso reconocimiento de MWE, aunque es ampliable por medio de la ampliación del módulo léxico correspondiente.

- Implementación

Treetagger únicamente necesita los documentos que forman los corpus, las etiquetas XML añadidas en el proceso anterior o durante el etiquetado retórico manual son omitidas automáticamente por el programa. Durante su ejecución, Treetagger realiza una tokenización por palabras de cada documento, lo transforma en formato "*one word per line*" y a continuación añade el lema y la etiqueta gramatical correspondiente (Figura 70).

```
<texto nombre='hello'>
    <s id='hello_en_0'>
        It would take Wenger winning the FA Cup and having an incredible Premier League finish for ...
    </s>
</texto>
```

```
<texto nombre='hello'>
<s id='hello_en_0'>
Palabra   POS     Lema
It        PP      it
would     MD      would
take      VB      take
Wenger    NP      Wenger
winning   VBG     win
the       DT      the
FA        NP      FA
Cup       NP      Cup
and       CC      and
having    VBG     have
an        DT      an
incredible        JJ      incredible
Premier   NP      Premier
League    NP      League
finish    VB      finish
for       IN      for
...       :       ...
</s>
</texto>
```

Figura 70 - Ejemplo de salida de Treetagger

Los métodos que gestionan la acción del etiquetador gramatical en cada clase se muestran a continuación (Tabla 48):

Tipo de corpus	**Clase**	**Método**	**Lenguaje**
Corpus monolingüe	CrearCorpusMono	treetaggerMono()	PHP+Treetagger
Corpus comparable	CrearCorpusComparable	treetaggerComparable()	PHP+Treetagger
Corpus paralelo multilingüe	CrearCorpusParalelo	treetaggerParalelo()	PHP+Treetagger
Corpus retórico monolingüe	CrearCorpusRetórico	treetaggerRetórico()	PHP+Treetagger

Tabla 48 - Tipos de corpus, clases, métodos asociados y lenguaje de programación empleados en la implementación de la actividad "Etiquetado gramatical"

El flujo de datos de entrada y salida de esta actividad se muestran a continuación (Figura 71):

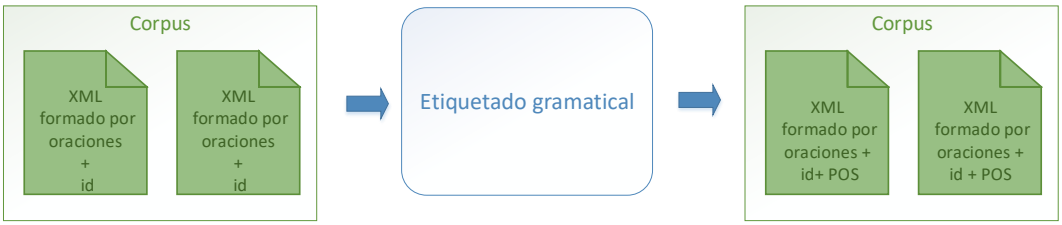


Figura 71 - Flujo de datos de la actividad "Etiquetado gramatical"

VIII) Etiquetador semántico

- Selección de recursos externos

ACTRES Corpus Manager incorpora un etiquetador semántico cuya función consiste en asignar a cada palabra una categoría de acuerdo con su significado.

El etiquetado semántico es un proceso muy complejo a causa de la ambigüedad del lenguaje. Esto se debe a la polisemia de las palabras, es decir, los distintos significados de las palabras dependiendo de su contexto. Por tanto, se trata de un proceso mucho más exigente que el etiquetado gramatical, porque se basa en el

conocimiento, ya sea a partir de ontologías, lexicones[48] (McEnery, Xiao, & Tono, 2006, pág. 38), datos de la Wikipedia (por ejemplo (Zhang & Rettinger, 2014)) o incluso por medio de la alineación de palabras en un corpus paralelo (Dyvik, 1998).

También existen estrategias para asignar etiquetas semánticas empleando técnicas de *Machine Learning*, utilizando datos de entrenamiento etiquetados de forma manual. No obstante, estas estrategias están mucho más orientadas a su aplicación directa en procesos NER (Liu, Zhang, Wei, & Zhou, 2011) , WSD (Navigli, Faralli, Soroa, de Lacalle, & Agirre, 2011), *sentiment analysers* (Pak & Paroubek, 2010) entre otros.

ACTRES Corpus Manager incorpora un etiquetador semántico en PHP, basado en los diccionarios (UCREL, 2016)[49], utilizados por el software *Multilingual UCREL Semantic Analysis System* (USAS) (Piao, Bianchi, Dayrell, D'Egidio, & Rayson, 2015)[50] [51].

Los recursos lingüísticos ofrecidos por USAS fueron seleccionados por delante de otras alternativas como WordNet (Princeton University, 2010) por los siguientes motivos:

- La taxonomía empleada en USAS se basa en el diccionario *Longman Lexicon of Contemporary English* (McArthur, 1981), lo que asegura en

[48] A lo largo de la tesis doctoral el término lexicón es empleado para designar a cualquier base de datos léxica con independencia de la mayor especificidad otorgada por los expertos en lingüística. Por ejemplo: tesauro, glosario o diccionario.

[49] El material léxico en lengua inglesa que se utiliza en esta tesis como fuente de información semántica es propiedad de Lancaster University – Universidad de Lancaster. Ha sido cedido por el Dr. Paul Rayson y el Dr. Scott Piao únicamente con fines demostrativos.

[50] Durante la realización de la tesis doctoral, el autor de la misma realizó una estancia de corta duración en el centro de investigación internacional UCREL (U*niversity Centre for Computer Corpus Research on Language*), perteneciente a *Lancaster University* – Universidad de Lancaster. En esta estancia se adquirieron conocimientos del funcionamiento del etiquetador USAS, así como de la metodología empleada para su construcción.

[51] El autor de esta tesis ha participado en la ampliación y corrección del módulo léxico para lengua española de USAS. Ver http://ucrel.lancs.ac.uk/usas/ - Spanish Semantic Tagger para más información y (Jiménez-Yáñez, Sanjurjo-González, Rayson, & Piao, Próximamente)

cierta medida su validez y motivación lingüística. Se trata de 21 campos de discursos y 232 categorías semánticas.

- Las etiquetas semánticas de USAS muestran campos semánticos que agrupan los sentidos de la palabra, que están relacionados en virtud de su conexión en algún nivel de generalidad dentro del mismo concepto. Los grupos no sólo incluyen sinónimos y antónimos sino también hipónimos e hiperónimos (Archer, Wilson, & Rayson, 2002).
- WordNet no tiene una taxonomía limpia, aunque organiza los sustantivos y verbos de forma jerárquica. Se trata más bien de grupos de sinónimos interconectados.

- Implementación

El etiquetador semántico se ha desarrollado en PHP y emplea los lexicones de USAS como fuente de datos. Además, para su funcionamiento requiere de la acción previa del etiquetador gramatical Treetagger.

Se afrontaron diversas dificultades durante la implementación del etiquetador semántico:

- Cada lexicón de USAS para idiomas distintos del inglés posee el formato "lema - etiqueta gramatical - etiquetas semánticas" (Figura 72):

```
absoluto        adj     A5.1+++ A1.7-
abstenerse      verb    A1.3/A1.8- A9 S9
abstinencia     noun    A1.3/A1.8- A9 S9
abstracción     noun    A1.6 P1 X2 A9+
abstracto       adj     A1.6 C1 X2 Q4.1 A9+
absurdo         adj     S1.2.6-
abuelo          noun    S4fm
```

Figura 72 - Formato del lexicón de USAS

- La inclusión del lema y de la etiqueta gramatical en el lexicón implica que es necesario realizar o haber realizado un etiquetado gramatical. Luego hay

que emplear Treetagger, aunque no se realice un etiquetado gramatical previo.

- USAS emplea un *tagset* gramatical simplificado (Tabla 49) que no se identifica con el utilizado por Treetagger, luego es necesario realizar una tarea de mapeado o conversión entre etiquetas.
  - Además, Treetagger emplea un *tagset* diferente para cada idioma, luego es necesario hacer una conversión de *tagset* por cada lengua soportada.

USAS simplificado	**Significado**
abbr	Abreviatura
adj	Adjetivo
adv	Adverbio
art	Artículo
conj	Conjunción
fw	Palabra extranjera o sin categoría conocida
intj	Interjección
noun	Nombre
num	Número
part	Se como partícula
port	Contracción
prep	Preposición
pnoun	Nombre propio
pron	Pronombre
punc	Signo de puntuación
sym	Símbolo
verb	Verbo

Tabla 49 - *Tagset* gramatical simplificado empleado por USAS a excepción de la lengua inglesa

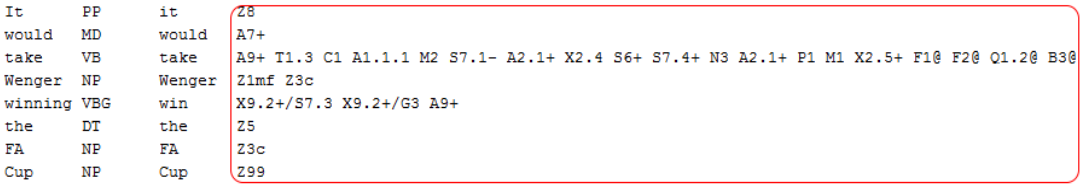
- En el caso del idioma inglés:
  - El lexicón de USAS emplea el formato "palabra - etiqueta gramatical - etiquetas semánticas" en lugar de utilizar el formato "lema - etiqueta gramatical - etiquetas semánticas".
  - Las etiqueta gramaticales empleada en la construcción del lexicón en lengua inglesa no es la etiqueta simplificada descrita en la Tabla 49. En este caso, proceden del *tagset* C7 de CLAWS (Garside, 1987), un etiquetador gramatical para lengua inglesa de pago. Para solucionar el problema y armonizar el empleo de Treetagger en todos los idiomas se llevó a cabo un proceso de transformación de las etiquetas C7 de CLAWS al formato Treetagger en el lexicón en lengua inglesa utilizado.
  - El nivel de detalle empleado por el *tagset* C7 es mucho mayor que el existente en Treetagger, concretamente 137 categorías frente a 58, por lo que en la conversión realizada se perdió contenido semántico. No obstante, el hecho de que el lexicón sea creado a partir de la palabra en lugar del lema, minimiza este hecho.
- Al utilizar Treetagger como base para el reconocimiento de las distintas palabras, el reconocimiento de expresiones MWE se adscribe a las reconocidas por éste.

Bajo estas premisas, el proceso de etiquetado semántico se divide en dos: (1) si el corpus está en lengua inglesa o (2) si no lo está:

(1) Corpus en lengua inglesa:

- Si el corpus no está ya etiquetado gramaticalmente, se crea un corpus temporal etiquetado.
- Se extrae la etiqueta POS correspondiente a cada palabra.
- Se busca en el lexicón semántico en inglés el par etiqueta POS-palabra.
- Se devuelve la etiqueta semántica o en su defecto Z99 para indicar que no hay resultados.

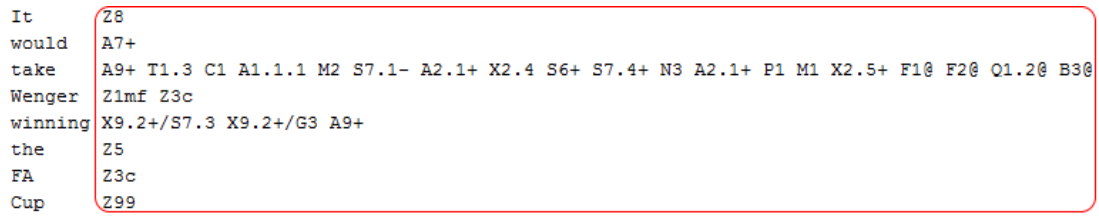
- Se añade la etiqueta semántica al corpus existente.
  - Si el corpus ha sido etiquetado gramaticalmente, se añade la etiqueta a continuación (Figura 73).



```
It      PP      it      Z8
would   MD      would   A7+
take    VB      take    A9+ T1.3 C1 A1.1.1 M2 S7.1- A2.1+ X2.4 S6+ S7.4+ N3 A2.1+ P1 M1 X2.5+ F1@ F2@ Q1.2@ B3@
Wenger  NP      Wenger  Z1mf Z3c
winning VBG     win     X9.2+/S7.3 X9.2+/G3 A9+
the     DT      the     Z5
FA      NP      FA      Z3c
Cup     NP      Cup     Z99
```

Figura 73 - Etiqueta semántica añadida a un corpus con etiquetado gramatical

  - En caso contrario, se añade como primer elemento (Figura 74).



```
It      Z8
would   A7+
take    A9+ T1.3 C1 A1.1.1 M2 S7.1- A2.1+ X2.4 S6+ S7.4+ N3 A2.1+ P1 M1 X2.5+ F1@ F2@ Q1.2@ B3@
Wenger  Z1mf Z3c
winning X9.2+/S7.3 X9.2+/G3 A9+
the     Z5
FA      Z3c
Cup     Z99
```

Figura 74 - Etiqueta semántica añadida a un corpus sin etiquetado gramatical

(2) Corpus en idioma distinto del inglés:

- Si el corpus no está ya etiquetado gramaticalmente, se crea un corpus temporal etiquetado.
- Se extrae el lema y la etiqueta POS correspondiente a cada palabra.
- Se convierte la etiqueta POS al formato simplificado utilizado en el lexicón correspondiente de USAS (ver Tabla 49).
- Se busca en el lexicón semántico del idioma correspondiente el par etiqueta POS simplificada-lema.
- Se devuelve la etiqueta semántica o en su defecto Z99 para indicar que no hay resultados.
- Se añade la etiqueta semántica al corpus existente:
  - Si el corpus ha sido etiquetado gramaticalmente, se añade la etiqueta a continuación (Figura 73).

- En caso contrario, se añade como primer elemento tras la propia palabra (Figura 74).

A modo de resumen se presentan los diagramas que muestra la ejecución y el flujo de datos del etiquetador semántico creado, a partir de un corpus que ha sido previamente etiquetado gramaticalmente (Figura 75) y otro que no lo ha sido (Figura 76):

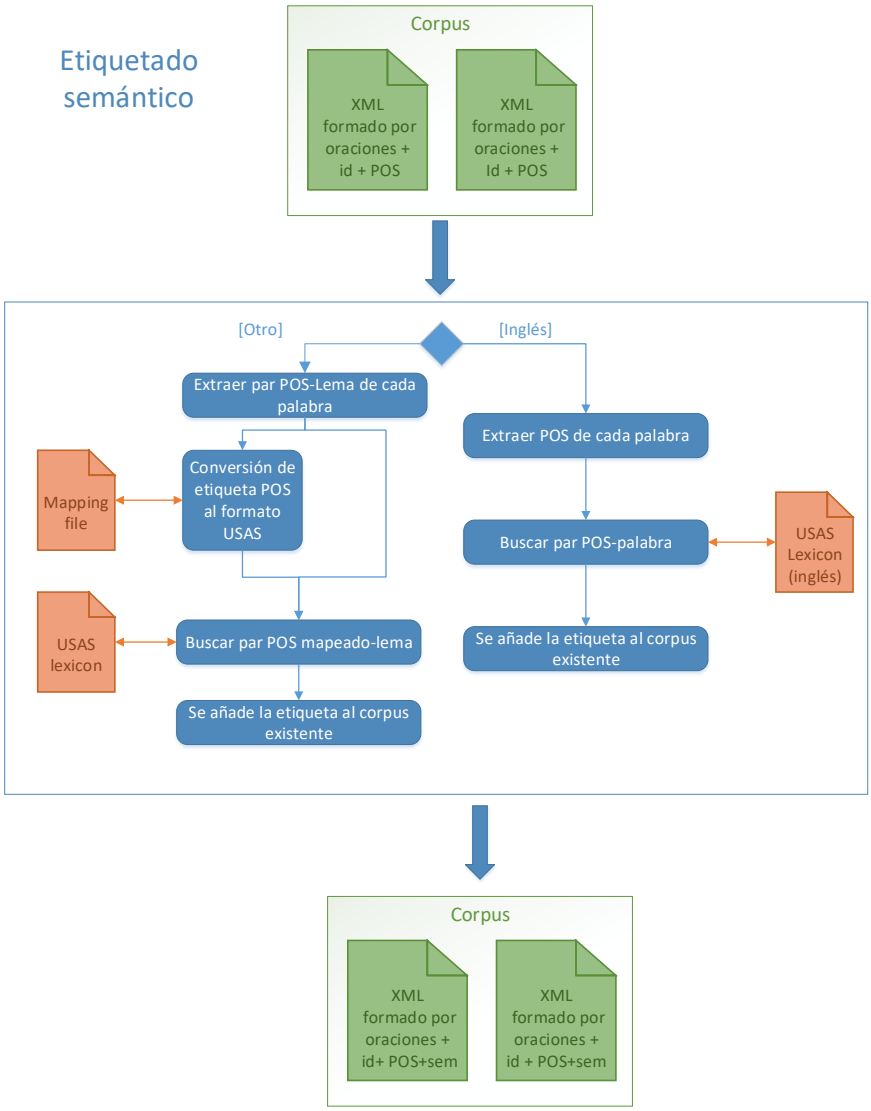


Figura 75 - Flujo de datos y ejecución de la actividad "Etiquetado semántico" de un corpus previamente etiquetado gramaticalmente

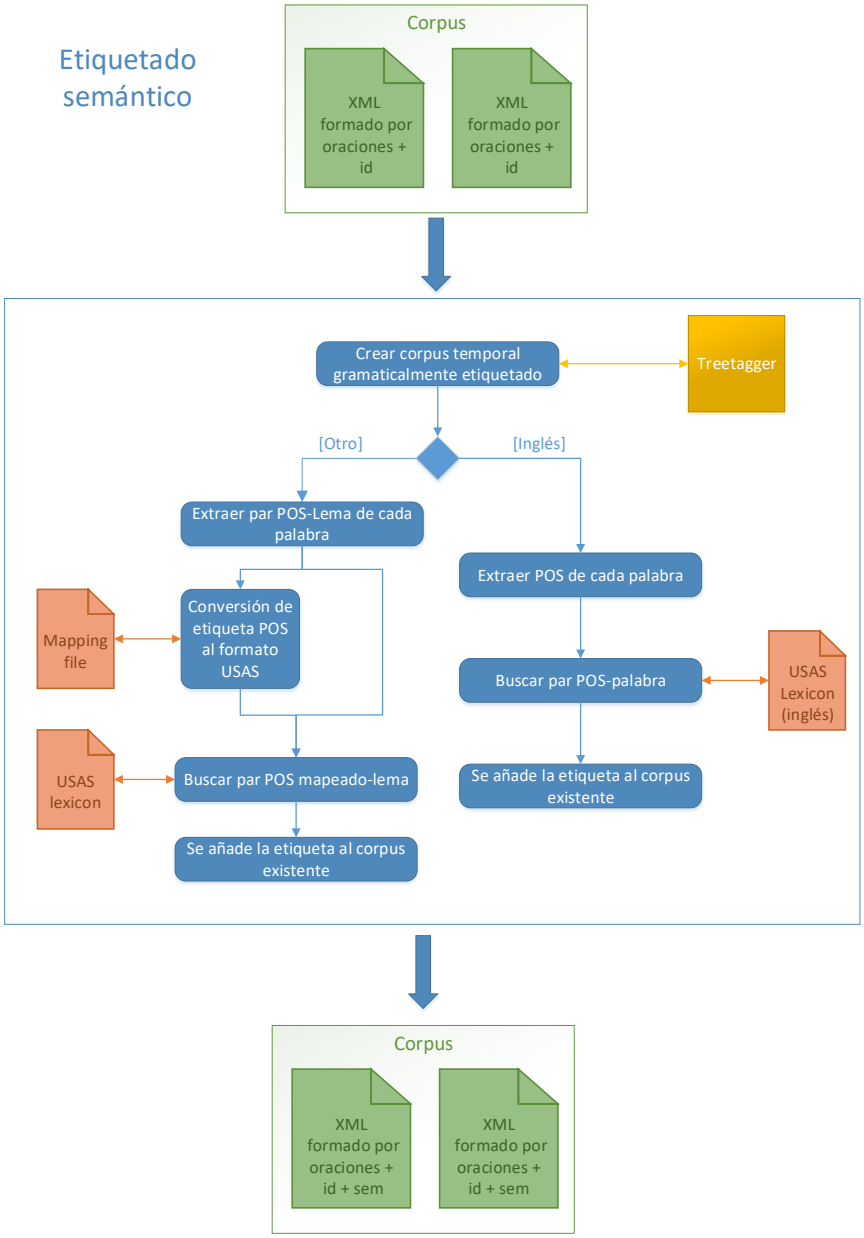


Figura 76 - Flujo de datos y ejecución de la actividad "Etiquetado semántico" de un corpus que no ha sido previamente etiquetado gramaticalmente

Los métodos que gestionan la acción del etiquetador semántico en cada clase se muestran a continuación (Tabla 50):

Tipo de corpus	**Clase**	**Método**	**Recursos técnicos**
Corpus monolingüe	CrearCorpusMono	semtaggerMono()	PHP+Treetagger+ USAS
Corpus comparable	CrearCorpusComparable	semtaggerComparable()	PHP+Treetagger+ USAS
Corpus paralelo multilingüe	CrearCorpusParalelo	semtaggerParalelo()	PHP+Treetagger+ USAS
Corpus retórico monolingüe	CrearCorpusRetórico	semtaggerRetórico()	PHP+Treetagger+ USAS

Tabla 50 - Tipos de corpus, clases, métodos asociados y recursos técnicos empleado en la implementación de la actividad "Etiquetado semántico"

IX) Etiquetador retórico

- Descripción del etiquetador

En el Capítulo 2: "Conceptos fundamentales", se definió el etiquetado retórico como el proceso de incorporar anotaciones retóricas a un corpus lingüístico, considerando las anotaciones como el conjunto de estructuras internas de un texto que hacen que sea reconocible como miembro un género textual concreto.

El etiquetado retórico implementado en *ACTRES Corpus Manager* es una declaración de intenciones que pretende demostrar la capacidad del framework para tratar etiquetados lingüísticos distintos de los más habituales (gramatical y semántico). Para más información acerca de las limitaciones del etiquetador retórico consultar el apartado 6.3.3 Limitaciones de la implementación.

El etiquetado retórico implementado posee una serie de peculiaridades:

- Tal y como sucede en el caso del etiquetado gramatical y semántico, no hay unanimidad a la hora de establecer etiquetas retóricas estándar, y más en este caso, donde cada tipo de texto puede poseer unas estructuras internas propias, con mayor o menor detalle.

- Con el objetivo de estandarizar el proceso de creación y consulta, se utilizan las etiquetas empleadas en el etiquetado retórico de los corpus comparables utilizados en los generadores de escritura del grupo de investigación ACTRES (ACTRES, 2017b).
- La creación de estas etiquetas requiere de un estudio y análisis lingüístico, realizado en géneros textuales e idiomas concretos, que en este caso sólo están disponibles en inglés y en español. Como consecuencia de los recursos lingüísticos disponibles, está limitado únicamente para dos idiomas, inglés y español, en lugar de para los 4 admitidos en el resto de tipos de corpus.

Ante la imposibilidad de crear un etiquetador retórico genérico, es decir, válido para cada tipo de texto con un conjunto de etiquetas estándar, se tomó la decisión de crear uno específico basado en un género textual concreto. Dentro de los recursos lingüísticos disponibles en el grupo de investigación ACTRES, se tomó la decisión de utilizar las estructuras jerárquicas para la redacción de actas de reuniones en lengua inglesa (Universidad de León, 2013) y el corpus C-GARE (ACTRES, 2017a), ya que los textos no eran excesivamente largos y su composición es relativamente sencilla.

- Implementación

Desde el punto de vista computacional, el etiquetado retórico diseñado consiste en dos acciones que se realizan de forma independiente:

(1) Por un lado, el reconocimiento de las estructuras prototípicas de las actas de reuniones, ya sea en inglés o en español. Esta acción es realizada por el usuario durante la subida de los documentos.[52] Para ello, se utilizarán un conjunto de etiquetas derivadas de los corpus C-GARE, etiquetados retóricamente de forma manual por el grupo de investigación ACTRES. El

[52] La fase 1 del etiquetado retórico pertenece a la lógica de la vista, pero como se ha mencionado con anterioridad, se ha creído adecuado explicar el conjunto del etiquetado retórico en un solo punto.

conjunto de etiquetas empleado y sus significados se muestran en la Tabla 51:

Etiquetas	**Significado**
SETTER - ORG - DID - MEET o DATE o PLACE o MID	Datos del acta y de la reunión - Nombre de la empresa/institución - Código del documento - Localización de la reunión o Fecha o Lugar o Identificación de la reunión
PARTICIPANTS (Participants) - PRESENT o ACTING o GUEST - ABSENT	Participantes - Presentes o Con voto o Con voz pero sin voto - Ausentes
WARM (Warming Up) - PRE o WELCOME o ANNOUNCE - PREVIOUSM - AGENDA - MATTER o OLD o NEW	Apertura/Inicio - Preliminares o Bienvenida o Avisos - Aprobación de actas anteriores - Puntos del orden del día - Asuntos a incorporar o Pendientes o Nuevos
PROCITEMS (Procedural Items) - ITEM - INFO o EXPO o PROPOSAL - DISCUSSION - ACTION - REACT o VOTE o AGREE	Asuntos tratados - Punto 1, Punto 2 … - Información o Exposición o Propuesta - Debate - Acción - Resultado o Voto o Acuerdo
ANY (Any other business) - STTMNT - REQST - DEBATE - RESPONSE	Ruegos y preguntas - Información de la dirección - Ruego - Opiniones - Respuesta
WRAP (Wrap-Up)	Cierre de la session

- NEXT - ADJOURN - SUBMSN - SGNTR	- Próxima reunion - Fórmula de cierre - Entrega de acta - Firma
UNCLASSIFIED	Sin clasificar

Tabla 51 - Etiquetas empleadas en el etiquetado retórico y su significado

La lógica de la vista se encarga de manejar la asignación de estas etiquetas por medio de un sencillo interfaz (ver Figura 77), en la que el usuario ha de:

i. Seleccionar un texto.

ii. Presionar el botón con la etiqueta deseada. *ACTRES Corpus Manager* engloba el texto seleccionado con la etiqueta correspondiente de forma automática.

Cabe señalar que CWB no permite incluir etiquetas anidadas recursivamente (Evert, 2009, pág. 6), es decir una etiqueta <*NEXT*> dentro de otra etiqueta <*NEXT*>. Estos elementos serán ignorados.

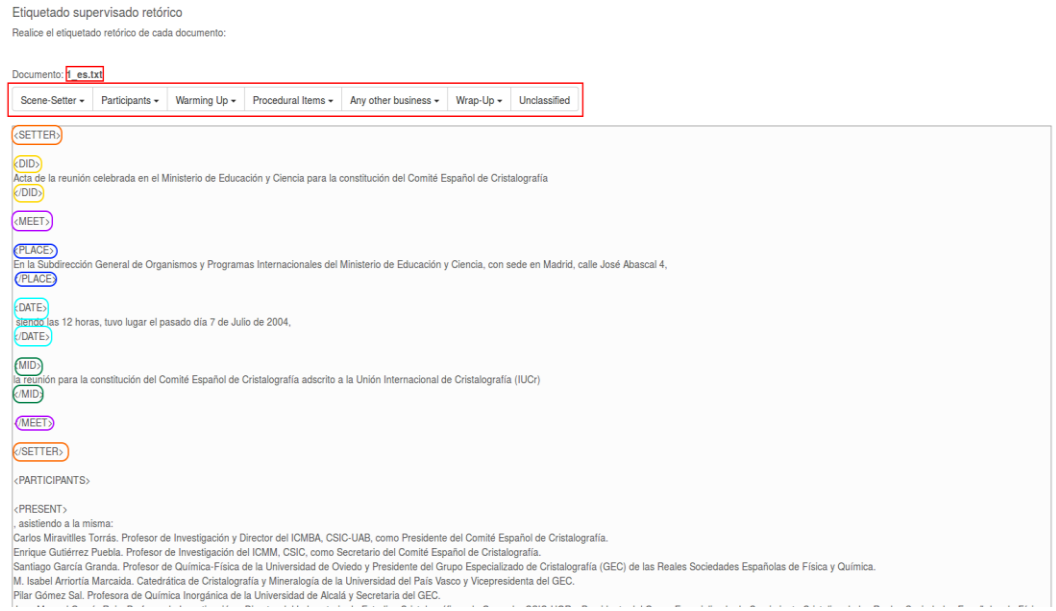


Figura 77 - Interfaz de etiquetado retórico

Tal y como se describió anteriormente,[43] esta información es recogida en la actividad I) Tratar documentos y datos del corpus, que la incluye en los documentos subidos por el usuario.

Un ejemplo del proceso de texto etiquetado retóricamente en la fase (1) se muestra en la Figura 78.

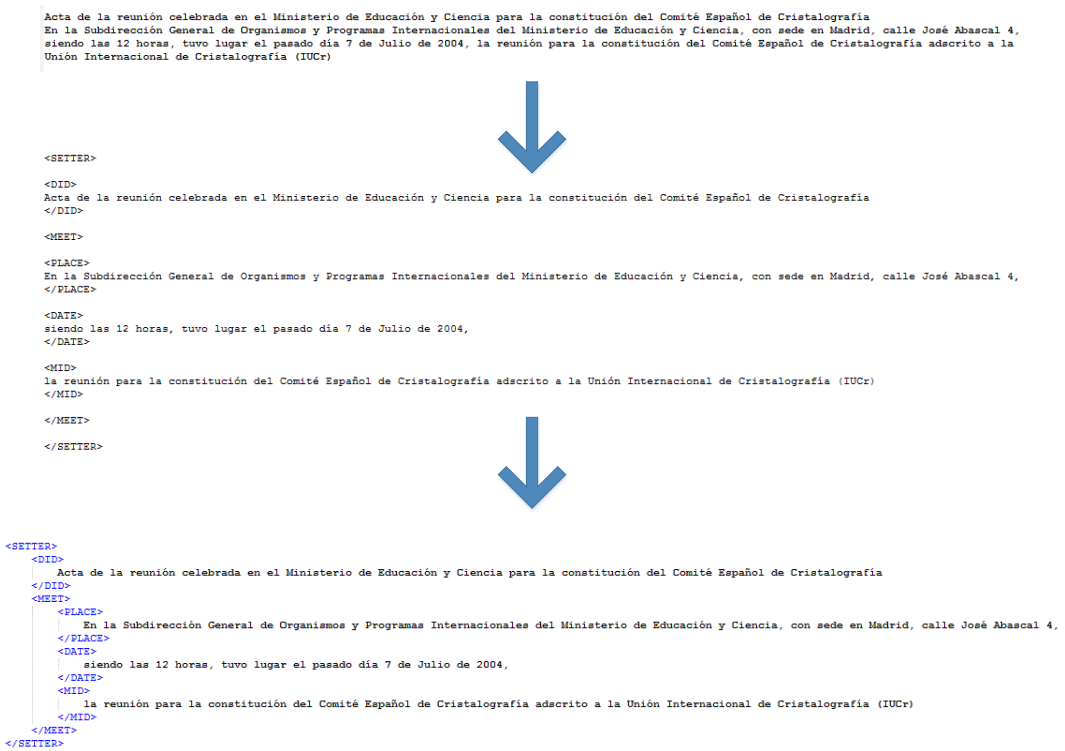


Figura 78 - Texto etiquetado retóricamente – fase 1

(2) La siguiente fase del etiquetado, consiste categorizar las palabras procedentes de estas estructuras prototípicas. Este proceso se realiza de forma automática de acuerdo con el contenido de un lexicón creado para tal fin basado en las estructuras jerárquicas para la redacción de actas de reuniones en lengua inglesa (Universidad de León, 2013).

Las categorías definidas en el lexicón creado son:

- Día de la semana (DÍA)

- Lugar (LUGAR)
- Mes (MES)
- Nombre e identificaciones (NP_I_IP)
- Objeto (OBJETO)
- Ninguna (NONE)

El formato del lexicón empleado se muestra en la Figura 79. Es simplemente la palabra seguida de la etiqueta retórica.

```
asamblea        NP_I_IP
autoridades     NP_I_IP
comisión        NP_I_IP
comité          NP_I_IP
delegación      NP_I_IP
```

Figura 79 - Formato del lexicón retórico

La inclusión de etiquetado retórico en el corpus se hace añadiendo la etiqueta retórica en la última posición (Figura 80) de los atributos posicionales (si es que el corpus ha sido etiquetado gramatical o semánticamente).

```
la              ART     el              Z5              NONE
constitución    NC      constitución    G1.1 B2 O1/A1.8+ NONE
del             PDEL    del             Z5              NONE
Comité          NP      Comité          S7.1+/S5+c      NP_I_IP
Español         NP      Español         Z2/Q3 Z2/S2     NONE
de              PREP    de              Z5              NONE
Cristalografía  NP      Cristalografía  Z99             NONE
```

Figura 80 - Etiqueta retórica añadida a un corpus con etiquetado gramatical y semántico

El método que gestiona la acción de la segunda parte del etiquetador retórico, ya incluido en la lógica del framework se muestra en la Tabla 52.

Tipo de corpus	**Clase**	**Método**	**Recursos técnicos**
Corpus retórico monolingüe	CrearCorpusRetórico	rettaggerRetórico()	PHP

Tabla 52 - Tipo de corpus, clase, método asociado y lenguaje de programación empleado en la implementación de la actividad "Etiquetado retórico – fase 2"

El flujo de datos de entrada y salida de esta actividad depende de si se han realizado el etiquetado gramatical o semántico con anterioridad, luego existen cuatro escenarios posibles mostrados en las figuras Figura 81, Figura 82, Figura 83 y Figura 84.

Figura 81 - Flujo de datos del componente "Etiquetado retórico" sin ningún etiquetado previo

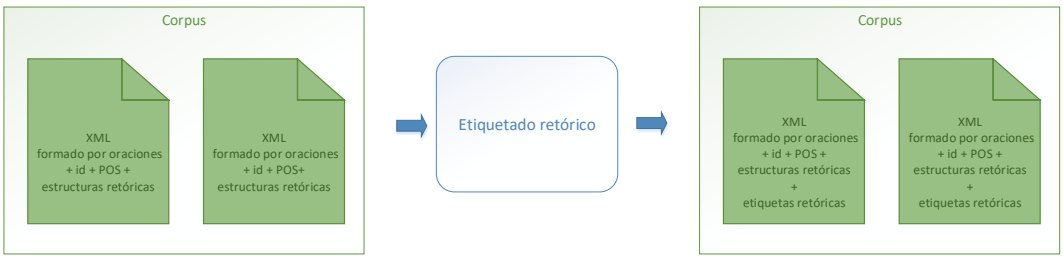


Figura 82 - Flujo de datos del componente "Etiquetado retórico" con etiquetado gramatical previo

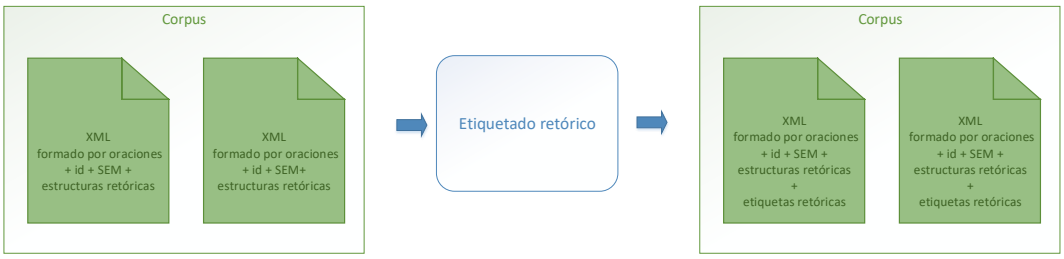


Figura 83 - Flujo de datos del componente "Etiquetado retórico" con etiquetado semántico previo

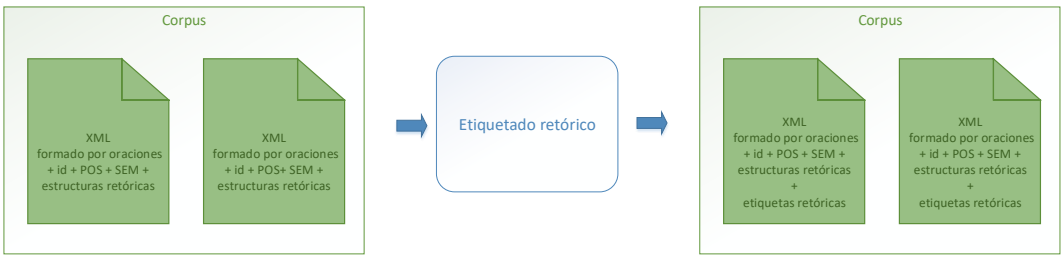


Figura 84 - Flujo de datos del componente "Etiquetado retórico" con etiquetado gramatical y semántico previo

X) Parser

El término parser tiene múltiples acepciones dependiendo del ámbito donde se utilice. Existen parsers de lenguaje natural, parsers como parte de compiladores informáticos, o parsers como conjuntos de instrucciones que realizan una acción concreta en un lenguaje programación. En esta tesis doctoral se utilizará esta última acepción.

La acción que realiza el parser en *ACTRES Corpus Manager* es la de adecuar el formato de los distintos documentos del corpus al requerido por el sistema de indexación. Los requisitos establecidos son:

- Formato de una palabra por línea (*one word per line*).
- Un solo archivo por corpus.
- Utilizar el modelo de datos de CWB en cuanto a la utilización de atributos posicionales (separados por tabulaciones), estructurales (utilizando

etiquetas XML y pares atributo valor) y de alineación (índices separados por tabulaciones) (Ver subapartado 6.3.4.3.1.2 Datos CWB).

El parsing es realizado a lo largo de todo el proceso, no adscribiéndose a ningún método en concreto. La mayoría de los métodos realizan alguna tarea relacionada con la función del parser (Tabla 53):

Tipo de corpus	**Clase**	**Métodos relacionados**
Corpus monolingüe	CrearCorpusMono	cleanMono() UTF8Mono() tokenizarMono() alinear() crearIndicesDeAlineacionParalelo() marcarMono() treetaggerMono() semtaggerMono() parserFinalMono()
Corpus comparable	CrearCorpusComparable	cleanComparable() UTF8Comparable() tokenizarComparable() alinear() crearIndicesDeAlineacionParalelo() marcarComparable() treetaggerComparable() semtaggerComparable() parserFinalComparable()
Corpus paralelo multilingüe	CrearCorpusParalelo	cleanParalelo() UTF8Paralelo() tokenizarParalelo() alinear() crearIndicesDeAlineacionParalelo() marcarParalelo() treetaggerParalelo() semtaggerParalelo() parserFinalParalelo()
Corpus retórico monolingüe	CrearCorpusRetórico	cleanRetorico() UTF8Retorico() tokenizarRetorico() alinear() crearIndicesDeAlineacionRetorico()

		marcarRetorico() treetaggerRetorico() semtaggerRetorico() parserFinalRetórico()

Tabla 53 - Tipos de corpus, clases y métodos relacionados con la actividad "Parsear documentos"

Los únicos métodos implementados relacionados en exclusiva a la labor de parseado y que no han sido descritos son *parserFinalMono*(), *parserFinalComparable*(), *parserFinalParalelo*() y *parserFinalRetórico*().Su función es la de agrupar el corpus en un solo documento de tipo vrt (verticalizado), y dar el formato de *one-word-per-line* al corpus si es que este no ha sido etiquetado gramatical, semánticamente o retóricamente con anterioridad, ya que si lo ha sido ya tendrá el formato adecuado.

El flujo de datos se presenta en la Figura 85.

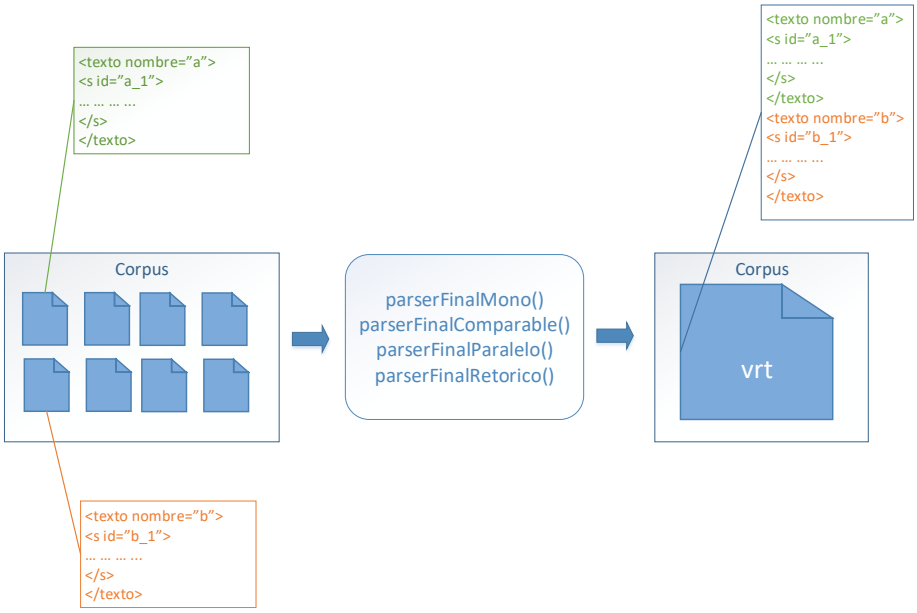


Figura 85 - Flujo de datos de los métodos *parserFinalMono*(), *parserFinalComparable*(), *parserFinalParalelo*() y *parserFinalRetórico*() relacionados con la actividad "Parsear documentos"

La actividad del parser en su conjunto se puede resumir en Figura 86:

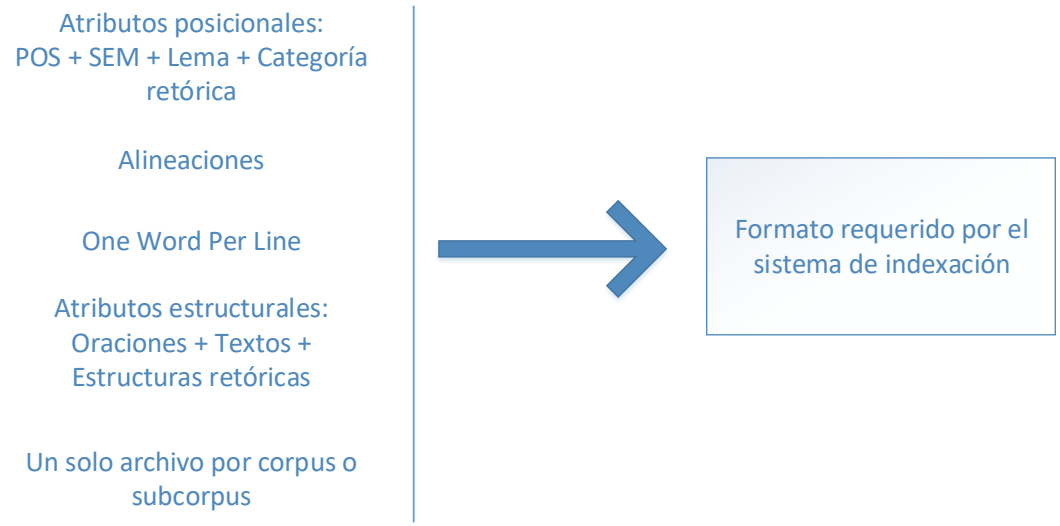


Figura 86 - Resumen de la actividad del realizada en "Parsear documentos"

XI) Indexar

El sistema de indexación y consulta seleccionado, CWB, incluye todos los componentes y *scripts* necesarios para incorporar corpus monolingües y las alineaciones de los corpus paralelos en su infraestructura.[53] No obstante, es necesario construir programas *ad hoc* para automatizar el proceso. Es conveniente usar *shell scripts* para interactuar directamente con las utilidades de CWB.

La indexación ha de respetar el modelo de datos presentado el subapartado 6.3.4.3.1 Datos del framework, es decir:

- Es necesario definir los atributos estructurales utilizados. Todos los corpus creados en *ACTRES Corpus Manager* poseen los atributos estructurales de texto y oración, que incluyen a su vez un identificador clave-valor de tipo XML.
    - Si se crea el corpus retórico se añadirán los atributos estructurales relativos al etiquetado de actas de reuniones definidos en la actividad IX) Etiquetador retórico.

[53] Para más información sobre el sistema de indexación consultar (Evert, 2016b).

- Es necesario definir los atributos posicionales, derivados de los etiquetados gramatical y semántico.
    - Si se crea un corpus retórico, se añadirán los atributos posicionales relativos a la categoría retórica.
- Por último, es necesario definir los atributos de alineación si el corpus es de tipo paralelo.

Cada combinación posible implica una serie de comandos de indexación diferentes. Por ejemplo, se quiere indexar un corpus con etiquetado gramatical y semántico, inclueyendo los atributos estructurales que tienen por defecto todos los corpus (oración y texto), los comandos ejecutados serían:

cwb-encode -c utf8 -d **$1** -f **$2** -R **$3** -xsB -P pos -P lemma -P sem -S texto:0+nombre -S s:0+id

cwb-make -r **$3** -V **$5**

Donde en el primer comando codifica el corpus:

- -c utf8: indica el tipo de codificación empleada
- -d **$1**: la ubicación donde se almacenará el corpus indexado en el sistema de archivos del servidor
- -f **$2**: el documento vrt que contiene el corpus
- -R **$3**: define el registro a utilizar, donde se almacenará el descriptor del corpus indexado
- -xSB: indica que el corpus está en formato XML
- -P pos: declara un atributo posicional denominado pos, es decir, el etiquetado gramatical
- -P lemma: declara un atributo posicional denominado lemma, es decir, el lema de la palabra procedente del etiquetado gramatical

- -P sem: declara un atributo posicional denominado sem, es decir, el etiquetado semántico
- -S texto:0+nombre : declara un atributo estructural denominado texto con un atributo clave-valor denominado nombre. Presente en todos los corpus.
- -S s:0+id : declara un atributo estructural denominado s (de *sentence* en inglés, oración) con un atributo clave-valor denominado id. Presente en todos los corpus.

El segundo comando sirve para validar el corpus creado, donde:

- -r **$3**: define el registro a utilizar
- -V **$5**: indica el corpus a validar

La Figura 87 muestra un ejemplo visual de los atributos a tener en cuenta en la indexación del corpus.

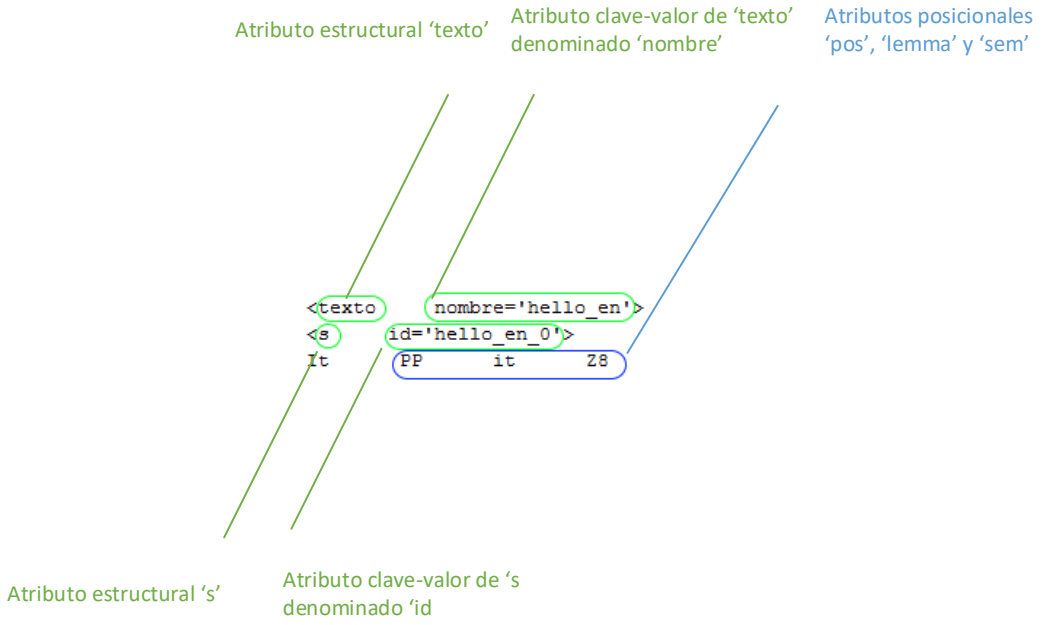


Figura 87 – Ejemplo gráfico de las partes a tener en cuenta en la indexación del corpus

Si el corpus es retórico, es necesario añadir adicionalmente los atributos estructurales del etiquetado retórico, que son:

-S SETTER -S ORG -S DID -S MEET -S DATE -S PLACE -S MID -S PARTICIPANTS -S PRESENT -S ACTING -S GUEST -S ABSENT -S WARM -S PRE -S WELCOME -S ANNOUNCE -S PREVIOUSM -S AGENDA -S MATTER -S OLD -S NEW -ES PROCITEMS -S ITEM -S EXPO -S PROPOSAL -S DISCUSSION -S AFR -S AAGNS -S OTHER -S ACTION -S WHAT -S WHO -S REACT -S VOTE -S AGREE -S ANY -S STTMNT -S REQST -S DEBATE -S FOR -S AGAINST -S RESPONSE -S WRAP -S NEXT -S ADJOURN -S SUBMSN -S SGNTR -S UNCLASIFFIED

Y la categoría retórica del etiquetado retórico basado en el lexicón creado:

-P reth

En el caso de crear corpus paralelos, se han de incluir las alineaciones posteriormente a la indexación de los subcorpus paralelos. Se trata de un proceso que ha de hacerse dos veces por cada par subcorpus original - subcorpus traducción. Para ello:

cwb-align-import -r **$1** -nh -e -l1 **$2** -l2 **$3** -s s -k {id} **$4**
cwb-align-import -r **$1** -nh -i -e -l1 **$2** -l2 **$3** -s s -k {id} **$4**

donde el primer comando importa la primera alineación subcorpus original – subcorpus traducido, y el segundo la alineación inversa (por medio de la opción -i):

- -r **$1**: define el registro a utilizar
- **-nh**: no se utiliza encabezado en el archivo de alineación
- **-e**: se aceptan las alineaciones vacías o sin correspondencia
- -l1 **$2**: denota el corpus original
- -l2 **$3**: denota el corpus traducido

- -s s: indica el atributo estructural usado en la alineación, en este caso oraciones especificadas mediante s
- {id}: el atributo clave-valor empleado como índice, en este caso id perteneciente al atributo estructural s
- **$4**: archivo donde se encuentran las alineaciones

La Figura 88 muestra un ejemplo gráfico de incorporación de alineaciones en el sistema de indexación

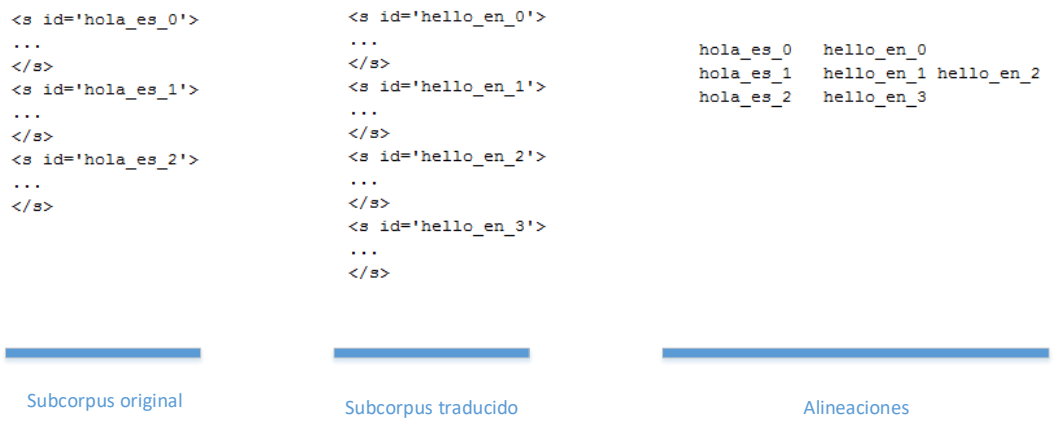


Figura 88 - Ejemplo gráfico de incorporación de alineaciones en el sistema de indexación

Los métodos que gestionan la acción del indexar en cada clase se muestran en la Tabla 54.

Tipo de corpus	**Clase**	**Método**	**Recursos técnicos**
Corpus monolingüe	CrearCorpusMono	indexarMono()	PHP + CWB
Corpus comparable	CrearCorpusComparable	indexarComparable()	PHP + CWB
Corpus paralelo multilingüe	CrearCorpusParalelo	indexarParalelo()	PHP + CWB

Corpus retórico monolingüe	CrearCorpusRetórico	indexarRetorico()	PHP + CWB

Tabla 54 - Tipos de corpus, clases y métodos relacionados con la actividad "Indexar"

El flujo de datos de esta actividad se puede apreciar en la Figura 89.

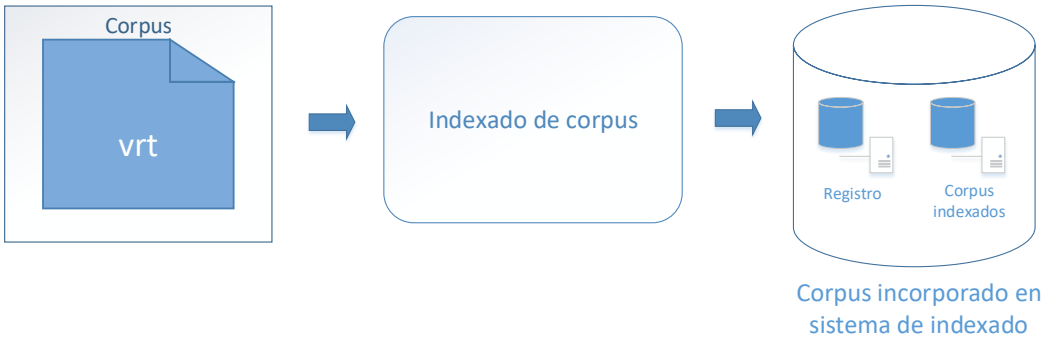


Figura 89 - Flujo de datos de la actividad "Indexar"

La diferenciación de los distintos tipos de corpus, en especial los retóricos y comparables, se realiza en la siguiente actividad.

XII) Actualizar información en base de datos

Tal y como se recogió al comienzo de este subapartado, todos los corpus son tratados como corpus monolingües por parte del sistema de gestión de corpus CWB, que únicamente reconoce los corpus paralelos por medio de la inclusión de los atributos de alineación correspondientes.

La diferenciación de los distintos tipos de corpus es realizada a través de su inclusión en la base de datos "Datos de usuario y corpus asociados", más concretamente en la colección "Base de datos de información de corpus asociados". El formato de los datos introducidos de esta colección se ha mostrado previamente para el corpus monolingüe (Figura 48), paralelo bi-/multilingüe (Figura 49), comparable (Figura 50) y monolingüe retórico (Figura 51).

Por último, además de incluir cada corpus en esta base de datos, cada subcorpus que forma parte de un corpus paralelo, comparable o retórico, también es incluido en la lista de corpus monolingües.

Los métodos que gestionan la acción del indexar en cada clase se muestran en la Tabla 55.

Tipo de corpus	**Clase**	**Método**	**Recursos técnicos**
Corpus monolingüe	CrearCorpus	actualizarInfoCorpus()	PHP
Corpus comparable	CrearCorpus	actualizarInfoCorpus()	PHP
Corpus paralelo multilingüe	CrearCorpus	actualizarInfoCorpus()	PHP
Corpus retórico monolingüe	CrearCorpus	actualizarInfoCorpus()	PHP

Tabla 55 - Tipos de corpus, clases y métodos relacionados con la actividad "Parsear documentos"

El flujo de datos de esta actividad se puede apreciar en la Figura 90.

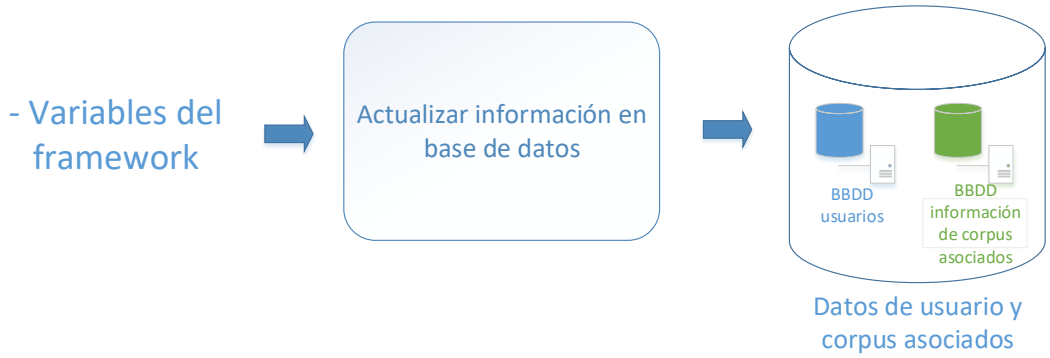


Figura 90 - Flujo de datos de la actividad "Actualizar información en base de datos"

6.3.4.3.2.2 Consultar corpus lingüísticos

El siguiente caso de uso es el de consultar corpus lingüísticos. Las consultas se hacen por medio del módulo de CWB denominado CQP[54] (Evert, 2009) que proporciona la posibilidad de realizar búsquedas sobre los corpus indexados a través del lenguaje de consultas denominado CQP Query Language. La elección de este lenguaje de consultas es una consecuencia directa de la selección de CWB como sistema de indexación. El lenguaje de consultas CQP Query Language permite la utilización de expresiones regulares a nivel de palabra y de anotación (Evert & Hardie, 2011, pág. 8).

Para interactuar con el módulo de consultas CWB/CQP es necesario utilizar una API que actúe como intermediaria. CWB proporciona esta funcionalidad mediante su API oficial CWB/Perl. Para un uso correcto es aconsejable disponer de conocimientos de programación en Perl. Si no se disponen de estos conocimientos, es posible utilizar alternativas no oficiales que emplean Python o R como lenguajes de programación.

En esta tesis doctoral se ha optado por emplear la clase CQP (Hardie, Baroni, Evert, & Sharoff, 2016), perteneciente a CQPweb (Hardie, 2012) aunque con un funcionamiento independiente. Esta clase se combina con las clases de elaboración propia CQPConfigData y CQPConversion (Ver Figura 91).

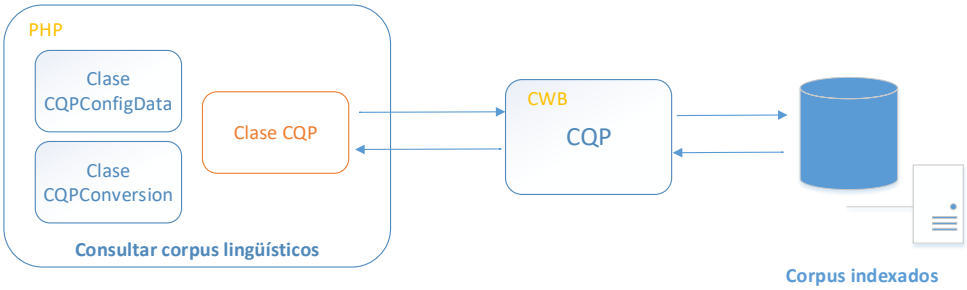


Figura 91 - Clases implicadas en la funcionalidad "Consultar corpus lingüísticos" y su relación con el componente CWB/CQP

[54] Para no confundirlo con la clase CQP en PHP, se denominará CWB/CQP

La selección la clase CQP es realizada exclusivamente por utilizar PHP, el mismo lenguaje de programación que el empleado en el desarrollo del modelo de *ACTRES Corpus Manager*.

Para realizar una consulta sobre un corpus se requiere de:

- Opciones de búsqueda.
- Consulta.

El resultado devuelto es un documento HTML que el controlador facilita a la vista correspondiente. El flujo de datos de la funcionalidad se muestra la Figura 92:

Figura 92 - Flujo de datos de la funcionalidad "Consultar corpus"

A continuación, se describirán las clases y métodos empleados en la consulta de corpus. Al contrario de lo que sucedió en la funcionalidad 6.3.4.3.2.1 Crear corpus lingüísticos, no es necesario describir actividades para facilitar la comprensión, ya que el flujo es mucho más sencillo.

I) Descripción de las clases

(1) Clase CQPConfigData

CQPConfigData es una clase elaborada en exclusiva para la implementación de *ACTRES Corpus Manager* cuya utilidad reside en que permite obtener las distintas rutas necesarias para la ejecución de programas externos o almacenamiento de archivos temporales (Tabla 56).

Método	Utilidad
obtenerProgramPath()	Devuelve la ruta del programa CWB en el servidor.
obtenerRegPath()	Devuelve la ruta del registro de los 'Datos CWB' de un usuario del framework.
obtenerQueryTmpPath()	Devuelve la ruta de la carpeta donde se ubicarán los archivos temporales relacionados con la consulta.
establecerDatosCQP()	Establece variables de la clase relacionadas con las rutas del programa CWB, de la ubicación de los corpus, etc.

Tabla 56 - Métodos de la clase CQPConfigData y su utilidad

(2) Clase CQPConversion

CQPConversion es una clase elaborada en exclusiva para la implementación de *ACTRES Corpus Manager* cuya utilidad reside en que permite transformar las consultas devueltas por los métodos de la clase CQP en distintos formatos (Tabla 57).

Método	Utilidad
convertToXML()	Transforma el resultado de una consulta en XML
convertToHTMLTable()	Transforma un documento XML en una tabla HTML

Tabla 57 - Métodos de la clase CQPConversion y su utilidad

(3) Clase CQP

La clase CQP (Hardie, Baroni, Evert, & Sharoff, 2016) pertenece al framework para el tratamiento de corpus lingüísticos CQPweb (Hardie, 2012), y sirve para comunicarse con el módulo CWB/CQP utilizando el lenguaje de programación PHP. De forma directa, *ACTRES Corpus Manager* únicamente emplean cuatro métodos de esta clase externa (Tabla 58).

Método	**Utilidad**
execute()	Permite ejecutar comandos CQP Query Language. Más concretamente se refiere al establecimiento de parámetros, mostrar/ocultar anotaciones, conteos, formato del resultado devuelto, etc. Para una información detallada consultar (Evert, 2016a).
query()	Permite ejecutar consultas sobre corpus de forma segura, evitando las inyecciones de código. Para más información (Evert, 2016a, pág. 8).
set_corpus()	Permite establecer el corpus sobre el que se va a realizar la consulta.
get_corpus_tokens()	Permite obtener el tamaño de un corpus.

Tabla 58 - Métodos empleados de la clase CQP y su utilidad

II) Implementación

Es necesario proporcionar una explicación adicional de cómo se realizan las consultas a través de la utilización de los métodos de estas clases. Hay que señalar que tal y como sucede en la creación de corpus lingüísticos, al reconocer CWB únicamente corpus monolingües, CQP Query Language sólo proporciona consultas sobre estos. Luego, para llevar a cabo las consultas sobre el resto de tipos de corpus hay que emplear distintas estrategias.

A continuación, se muestra las estrategias empleadas para ejecutar una consulta sobre los distintos tipos de corpus soportados por *ACTRES Corpus Manager*.

(1) Corpus monolingüe

A modo de ejemplo, en la Figura 93 se muestra la imagen de uno de los interfaces de búsqueda de *ACTRES Corpus Manager*.

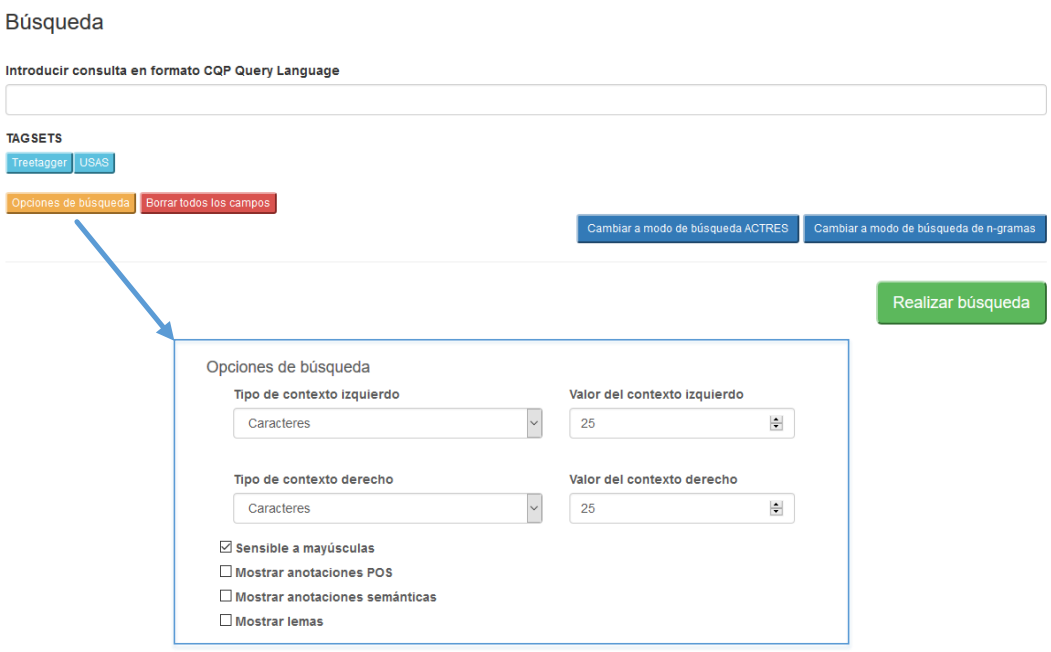


Figura 93 - Imagen del interfaz de búsqueda de *ACTRES Corpus Manager*

1. Se obtienen las distintas rutas del sistema de archivos a través de los métodos de la clase CQPConfigData. (*obtenerProgramPath*, *obtenerRegPath* y *obtenerQueryTmpPath*)
2. Se selecciona el corpus a utilizar por medio del método **set_corpus** de la clase CQP.
3. A través del método **execute** de la clase CQP se establecen las opciones de la consulta. Para ellos es necesario utilizar distintos comandos CQP Query Language.
    - Por ejemplo, para establecer el contexto de la búsqueda y ajustes internos relacionados con el formato de los resultados se emplea el comando:

      set PrintOptions opción
    - Para mostrar u ocultar anotaciones en los resultados de la consulta se utiliza el comando:

      show +opción

4. Posteriormente, se ejecuta la consulta por medio del método **query** de la clase CQP.

   ACTRES Corpus Manager proporciona dos tipos de búsquedas.

   a. **Búsquedas avanzada** (Figura 94): Permite introducir una consulta en sintaxis CQP Query Language directamente. Se facilita información sobre los *tagsets* empleados para que puedan incluirse en la consulta.

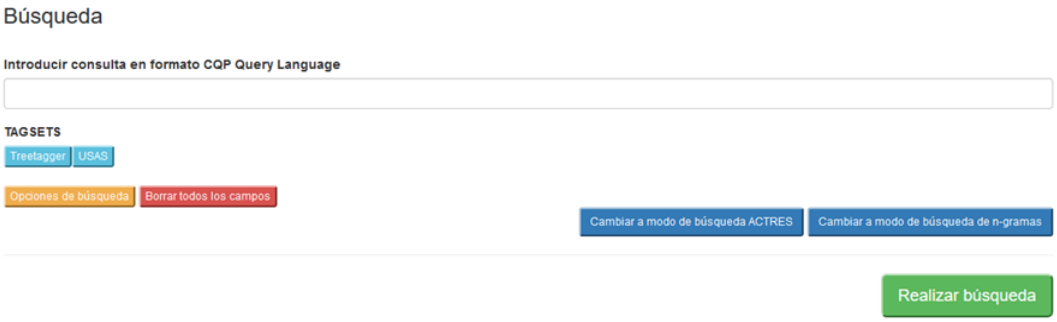


Figura 94 - Imagen del interfaz avanzado de búsqueda

   b. **Búsqueda ACTRES** (Figura 95): Basada en el interfaz de búsqueda de P-ACTRES 2.0 (Sanjurjo-González & Izquierdo, próximamente) e ideada originalmente en P-ACTRES (Izquierdo, Hofland, & Reigem, 2008). Se trata un tipo de consulta que permite la utilización de búsqueda por patrones (también llamados expresiones regulares) de forma asistida. Para ello, el interfaz creado proporciona tres secuencias de texto que brindan la posibilidad de establecer tres parámetros de búsqueda.

Figura 95 - Imagen del interfaz ACTRES de búsqueda y de las tres secuencias disponibles

Cada una de las secuencias posee una opción para cambiar su tipo (ver Figura 96). Este tipo es el parámetro que simula el comportamiento de una búsqueda por patrones, aprovechando para ello los recursos ofrecidos por CQP Query Language:

- palabra completa (Ejemplo: *[word=”andando”]*)
- principio de palabra (Ejemplo: *[word=”and.*”]*)
- final de palabra (Ejemplo: *[word=” *.ando”]*)
- cualquier parte de la palabra (Ejemplo: *[word=” *.an.*”]*)
- Y el mismo comportamiento en el caso de los lemas: lema, principio de lema, parte de lema y final de lema

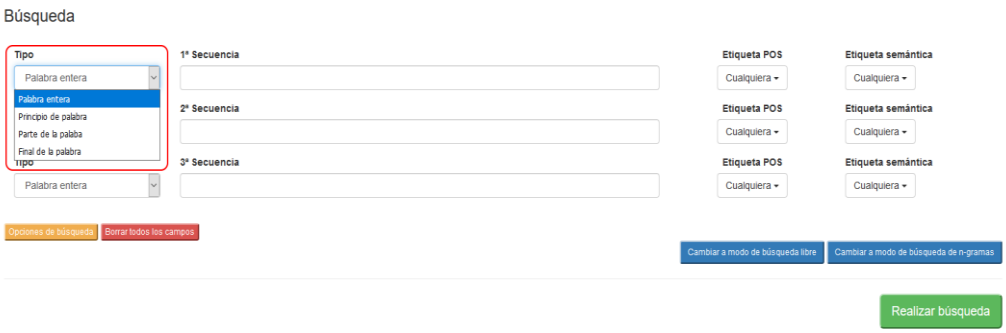


Figura 96 - Imagen del interfaz ACTRES de búsqueda y de los tipos de secuencias 1/2

Con el objetivo de proporcionar búsquedas que impliquen más de tres palabras, la segunda y tercera secuencia dan la posibilidad adicional de ampliar el rango de búsqueda, es decir, que la secuencia a buscar no sea adyacente, sino en un rango desde 0 a 9 palabras o lemas a elección del usuario (Figura 97).

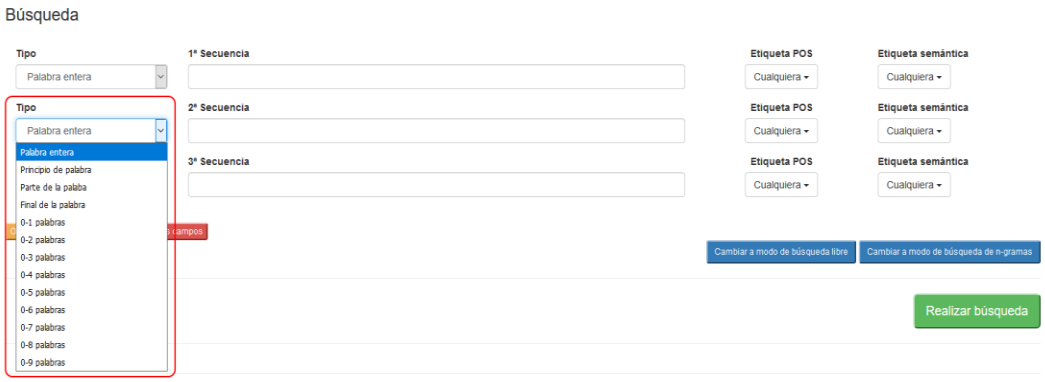


Figura 97 - Imagen del interfaz ACTRES de búsqueda y de los tipos de secuencias 2/2

También se pueden especificar las etiquetas POS y/o semántica que se deseen en cada secuencia (ver Figura 98).

Figura 98 - Imagen del interfaz ACTRES de búsqueda y de la especificación de etiqueta POS y semántica

De acuerdo con lo descrito en la actividad VII) Etiquetador gramatical, el etiquetado gramatical incorpora una etiqueta para cada palabra. Por ejemplo, “PP” es la etiqueta empleada por Treetagger para denotar un pronombre personal en inglés. Con el

fin de facilitar la consulta ACTRES estas etiquetas son transformadas en su respectiva categoría (ver Figura 99).

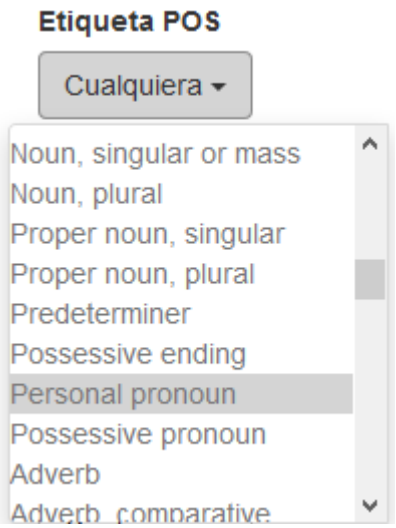


Figura 99 - Imagen de la transformación en lenguaje natural de cada etiqueta POS en el interfaz

Sin embargo, como consecuencia de la utilización de Treetagger, es necesaria una transformación diferente para cada idioma ya que como se ha mencionado con anterioridad, Treetagger emplea distintas etiquetas gramaticales para cada idioma. Por ejemplo, la etiqueta gramatical en idioma español para denotar pronombres personales es "PPX", en lugar de "PP" como en inglés.

En el caso de las etiquetas semánticas, el proceso fue más sencillo porque las etiquetas son uniformes para cada idioma (ver Figura 100).

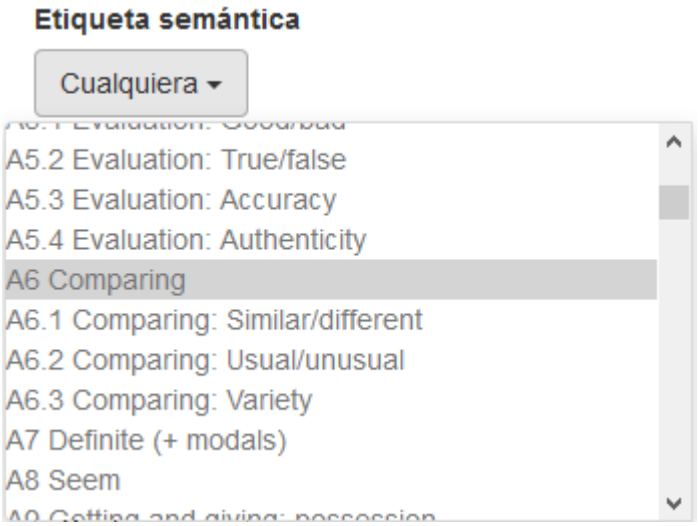


Figura 100 - Imagen de la transformación en lenguaje natural de cada etiqueta semántica en el interfaz

Por último, en las opciones de búsqueda también se puede establecer si la búsqueda es sensible o no a las mayúsculas, algo que en la sintaxis CQP Query Language va implícito en el comando de búsqueda a través de la adición del string "*%c*".

A modo de ejemplo y con el fin de facilitar sobre todo la comprensión de los pasos 3) y 4), se procederá a escribir una consulta en CQP Language Query basada en una consulta realizada en el interfaz ACTRES:

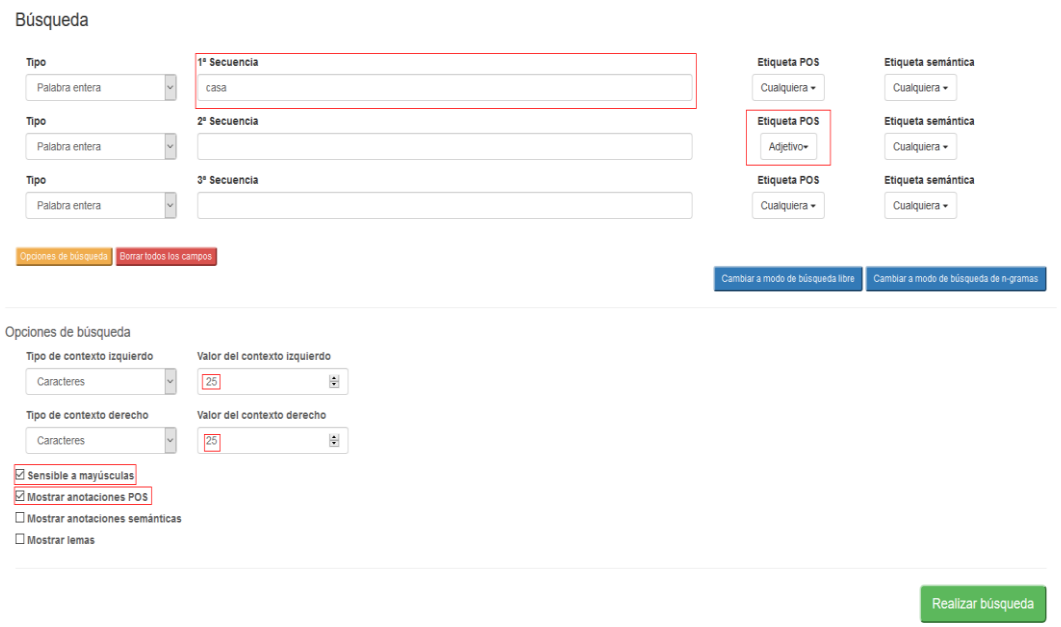


Figura 101 - Ejemplo de búsqueda en el interfaz ACTRES

La consulta representada en la Figura 101 implicará los siguientes comandos CQP Query Language mostrados en la Figura 102:

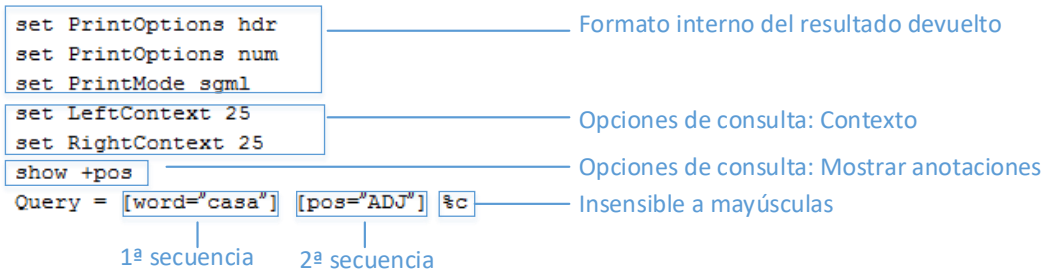


Figura 102 - Ejemplo de comandos CQP Query Language

5. En cualquiera de las dos interfaces de búsqueda, una vez realizada la consulta a través de los métodos de la clase CQP, se ejecutan los métodos de la clase CQPConversion: *convertToXML* y *convertToHTMLTable*, que permiten transformar los resultados devueltos por la consulta, primero en XML y posteriormente en HTML. Aunque idealmente este último método debería de formar parte del controlador, al añadirse a la consulta parámetros pertenecientes a

utilidades de CWB se ha optado por implementarlo en el modelo del framework.

6. El resultado se envía de vuelta al controlador que actualiza la vista consecuentemente a través de una tabla en formato KWIC. Por ejemplo, ver Figura 103.

Número de resultados: 73

Corpus BROWN POS SEM
Tamaño: 102365 palabras

Doc. ID	Contexto izq.	Resultado	Contexto der.
2842	Texas School for the Deaf	here	. Operating budget for t
7800	There is little optimism	here	that the Communists will
7919	MODIFIED The inclination	here	is to accept a de facto
10320	-management expert , said	here	yesterday . Mr. Reama to
12994	gubernatorial nomination	here	last night with a pledge
13034	Inn : " When I come back	here	after the November elect
15115	urchmen pointed out , and	here	is an example : Last mon
15913	a round of consultations	here	on future tactics in the
19017	ncy rates in major hotels	here	ranged from 48 to 74 per
20150	ght . The fire department	here	has been torn for months
20212	fire fighters association	here	offered a $5,000 reward
23745	ork Yankees , who come in	here	tomorrow for a night gam
23856	okie-of-the year " , flew	here	late this afternoon from
24228	the entire New York squad	here	from St. Petersburg , in
24604	e , featured seventh race	here	today , and paid $7.20 s
25098	rrow at Franklin Hospital	here	. Waters injured his lef
25128	d in the knee and he came	here	to consult the team phys

Figura 103 - Ejemplo de vista con el resultado de una consulta

(2) Corpus comparable

La implementación es idéntica al corpus monolingüe, pero en lugar de realizar una sola consulta, se realiza una por cada subcorpus comparable. En el caso de corpus comparables que estén en el mismo idioma se proporcionan consultas sobre el texto, si están en distinto idioma únicamente se permiten consultas sobre sus anotaciones lingüísticas. Los resultados finales se muestran en tantas tablas como subcorpus lo compongan.

(3) Corpus paralelo

Para realizar una consulta paralela basta con mostrar los atributos de alineación que CWB soporta nativamente. Para ello, es necesario emplear el comando de CQP Query Language:

show +corpusparaleloen

a través del método **execute** de la clase CQP.

Si la búsqueda realizada emplea correspondencias directas entre los distintos subcorpus, se utiliza el operador ":" en el momento de especificar la búsqueda a través del método **query** de la clase CQP. Por ejemplo, buscar la palabra "Hola" y su correspondencia en el corpus paralelo "*CorpusParaleloEn*" con la palabra "*Hello*".

*Query = [word="Hola"] :**CORPUSPARALELOEN** [word="Hello"];*

Por último, el interfaz de búsqueda varía para posibilitar la introducción de parámetros en todos los subcorpus del corpus paralelo (ver Figura 104).

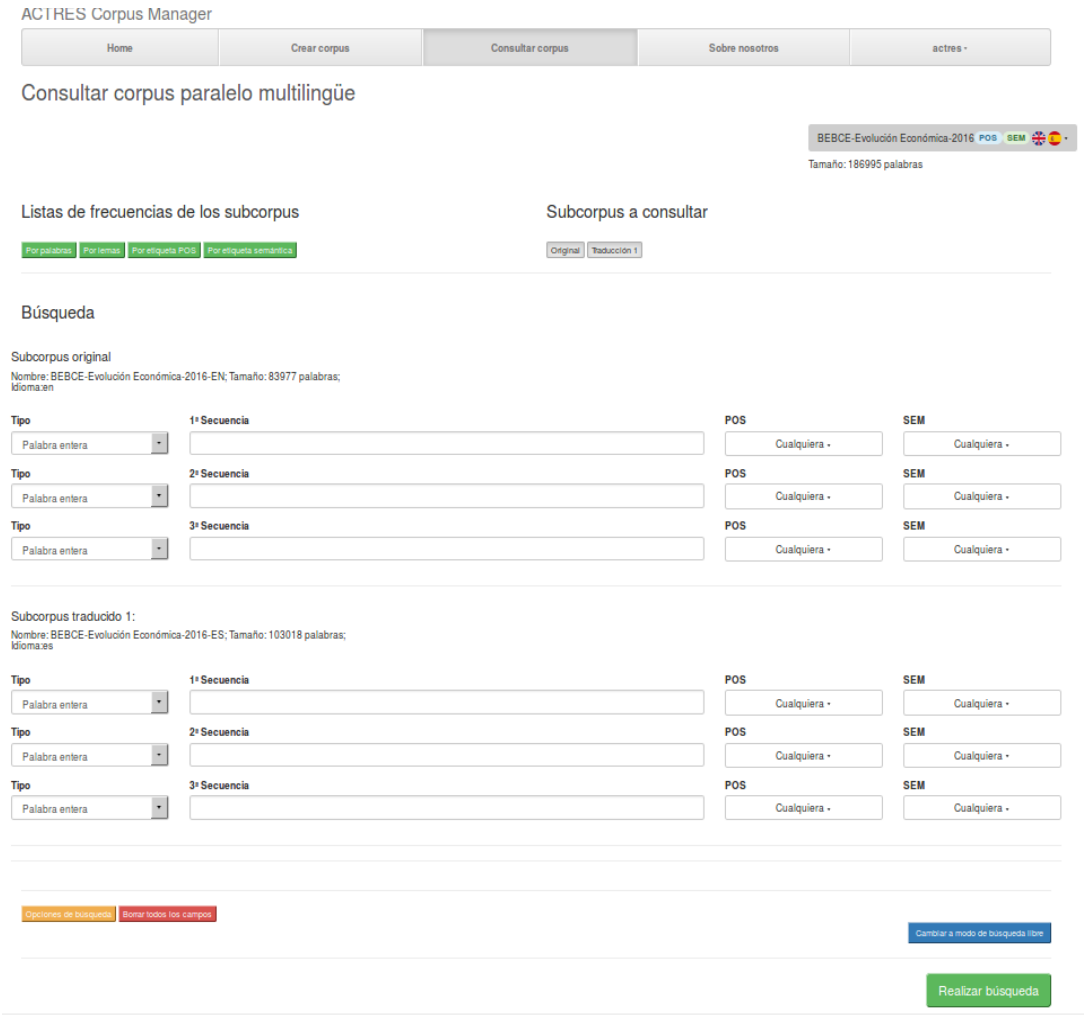


Figura 104 - Vista para realizar consultas sobre corpus multilingües paralelos

(4) Corpus retórico

La implementación es idéntica al corpus monolingüe, pero en las consultas se podrá especificar la región donde realizar la consulta y la categoría de acuerdo con la etiqueta retórica de cada palabra (ver Figura 105).

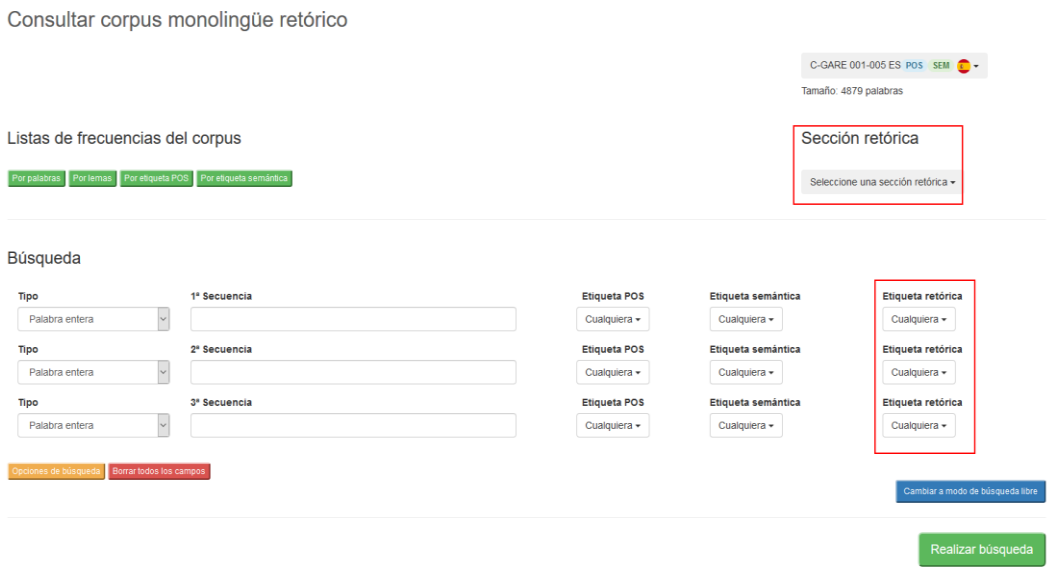


Figura 105 - Selección de región en consulta retórica

Para especificar la región donde realizar la consulta es necesario utilizar el parámetro **within** en la ejecución del método **query** de la clase CQP. Por ejemplo, para encontrar todas las apariciones de la palabra "*welcome*" en el atributo estructural "*warm*" basta con ejecutar esta consulta:

Query = [word="welcome"] ***within*** *warm;*

6.3.4.3.2.3 Estadísticas de consultas y de corpus lingüísticos

El último caso de uso planteado en *ACTRES Corpus Manager* es de la extracción de estadísticas de las consultas y de las estadísticas de los corpus lingüísticos. La descripción conjunta de estos dos procesos diferenciados en los casos de uso se hace como consecuencia exclusiva de ser implementados a partir de las mismas clases.

Para la obtención de estadísticas, *ACTRES Corpus Manager* emplea utilidades predefinidas de CQP Query Language y del paquete estadístico polmineR (Blaette, 2017) con apoyo de la librería rcqp (Desgraupes & Loiseau, 2016) , ambos basados en el lenguaje de programación R (R Core Team, 2017). La razón por la que se

utiliza polmineR es que no requiere conocimientos avanzados de estadística para ejecutar los tests de significación estadística indispensables para la extracción de palabras clave. Además, facilita la realización de operaciones estadísticas cuantitativas que resultarían mucho más complejas o no realizables utilizando únicamente comandos CQP Query Language.

Por otro lado, rcqp es utilizado por polmineR como API en R para comunicarse con el módulo CWB/CQP. rcqp también podría utilizarse para realizar consultas sobre corpus lingüísticos, al igual que la clase CQP en PHP previamente descrita, pero *ACTRES Corpus Manager* no utiliza esta funcionalidad.

La clase de elaboración propia '*Estadisticas*'[55] coordina la utilización de las distintas funcionalidades disponibles bien por medio de la clase CQP (lista de frecuencias de una consulta, colocaciones) o bien de las librerías polmineR y rcqp (listas de frecuencias de un corpus, lista de palabra clave, extracción de n-gramas).

La clase CQPConfigData, al igual que sucede en el caso de la funcionalidad "Crear Corpus lingüísticos", es utilizada para el establecimiento de variables y configuraciones de almacenamiento.

La Figura 106 muestra la relación entre las clases, programas externos y el módulo CWB/CQP.

[55] Se denomina 'Estadisticas' sin tilde porque las tildes no pueden ser utilizadas en el lenguaje de programación empleado.

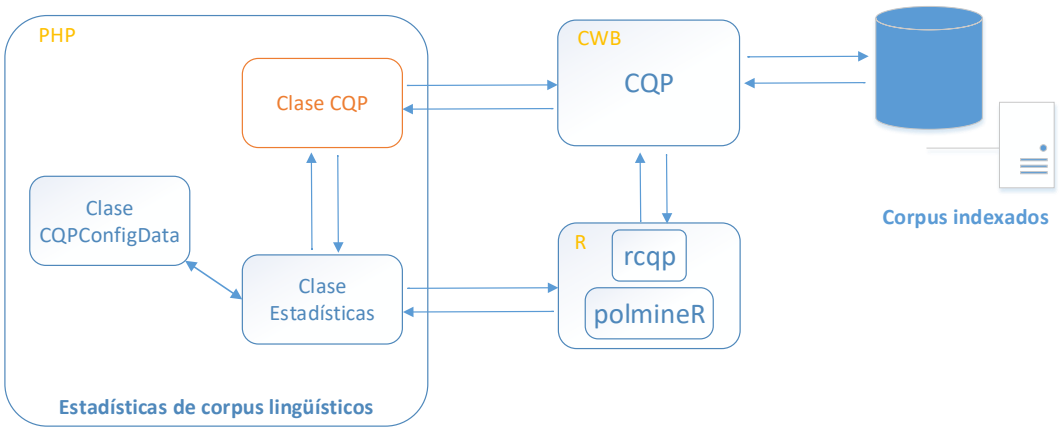


Figura 106 - Clases, y librerías externas implicadas en la funcionalidad "Estadísticas de corpus lingüísticos" y su relación con el componente CWB/CQP

La extracción de estadísticas de un corpus lingüístico únicamente requiere que el usuario seleccione una funcionalidad estadística: lista de frecuencias de una consulta, colocaciones de una consulta, lista de frecuencias del corpus, lista de palabras clave o búsqueda de n-gramas, así como las opciones estadísticas adicionales que dependan del tipo de funcionalidad estadística seleccionada.

El resultado devuelto es un documento de tipo XLSX (Excel). Una referencia al documento creado es enviada al controlador del framework para que se la facilite a la vista correspondiente.

El flujo de datos de la funcionalidad se muestra en la Figura 107.

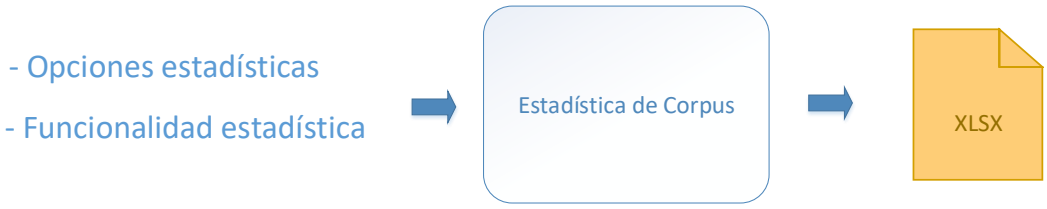


Figura 107 - Flujo de datos de la funcionalidad "Estadística de corpus"

No se tratan diferenciaciones estadísticas respecto a los tipos de corpus, todos ellos son tratados como corpus monolingües. Es decir, la extracción de estadísticas en cada corpus se rige por el siguiente proceso:

- **Corpus monolingüe**: unidad básica de corpus para el sistema de indexación.
- **Corpus paralelo multilingüe**: las estadísticas se extraerán sobre cada subcorpus, tratándose cada cual como un corpus monolingüe.
- **Corpus comparable**: las estadísticas se extraerán sobre cada subcorpus, tratándose cada cual como un corpus monolingüe.
- **Corpus retórico**: las estadísticas se extraerán de la misma forma que las de un corpus monolingüe.

Por otra parte, como consecuencia de la complejidad computacional y de la mayor relevancia y utilidad en el caso de los corpus monolingües, la lista de palabras clave y la extracción de n-gramas sólo se implementan en caso de estos. De todas formas, estas funcionalidades son accesibles por cualquier subcorpus seleccionado como corpus monolingüe.[56]

A continuación, se describirán las clases y métodos empleados en la estadística de corpus.

I) Descripción de las clases

(1) Clase CQPConfigData

Previamente descrita.

(2) Clase CQP

Previamente descrita.

[56] Para más información sobre esta decisión consultar la página 177 perteneciente al apartado 6.3.3 Limitaciones de la implementación.

(3) Clase '*Estadisticas'*

'Estadisticas' es una clase elaborada en exclusiva para la implementación de *ACTRES Corpus Manager* cuya utilidad reside en que permite enlazar con los recursos externos (rcqp, polmineR y clase CQP), además de transformar las resultados devueltos por los métodos de la clase CQP en distintos formatos (Tabla 59).

Método	**Utilidad**
establecerDatosIniciales()	Establece variables relacionadas con las rutas del programa CWB, de la ubicación de los corpus, etc.
frequencyListQueryConversion()	Transforma la lista de frecuencias de la consulta en formato XLSX (Excel).
freqCollocateListConversion()	Identifica los emparejamientos de la lista de colocaciones y transforma el documento en un nuevo formato.
freqCollocateTableConversion()	Realiza el conteo de las colocaciones y convierte la lista procedente de *freqCollocateListConversion* en una tabla de colocaciones.
freqCollocateAllConversion()	Transforma la tabla de colocaciones en formato XLSX (Excel).
frequencyListCorpus()	Obtiene la lista de frecuencias de un corpus.
establecerDatosNgram()	Establece como parámetros las opciones estadísticas seleccionadas en la búsqueda de n-gramas.
ngrams ()	Obtiene la lista de n-gramas de un corpus.
getKeywords()	Obtiene la lista de palabras clave.

Tabla 59 - Métodos empleados de la clase 'Estadisticas' y su utilidad

II) Implementación

A continuación, se detallará la implementación de cada funcionalidad estadística ofrecida. Se obvia la utilización del método *establecerDatosIniciales* de la clase '*Estadisticas'*, que se emplea al comienzo de todas ellas.

(1) Lista de frecuencias de la consulta

La lista de frecuencia de una consulta muestra el número de apariciones en el texto de la misma de acuerdo con unos parámetros. Aunque se trate de una función estadística, la lista de frecuencias de una consulta está estrechamente ligada las consultas sobre corpus lingüísticos. Por este motivo emplea métodos de la clase CQP como **execute**, **query** y **set_corpus** (descritos con anterioridad) que permiten ejecutar la consulta nuevamente.

Para obtener la lista de frecuencias de una consulta, se utiliza la funcionalidad **count** del lenguaje de consultas CQP Query Language a través del método **execute** de la clase CQP. Adicionalmente, el usuario debe seleccionar cómo desea organizar la lista de frecuencias de la consulta: por palabras, por lemas, por etiquetas gramaticales o por etiquetas semánticas. Simplemente se añade la funcionalidad de CQP Query Language **by**, seguido del tipo de organización que el usuario haya seleccionado.

Por ejemplo, el usuario desea saber la lista de frecuencias de acuerdo con su etiqueta gramatical, el comando introducido en el método **execute** de la clase CQP sería:

count Query by POS

Tras obtener el resultado de la consulta, ésta se transforma a través del método *frequencyListQueryConversion* de la clase '*Estadisticas*' en un archivo XLSX (Excel) por medio de un script propio en R. Una referencia al archivo XLSX se envía de vuelta al controlador (ver Figura 108).

	A	B
1	Count	Sequence
2	7	NN
3	7	NP
4	4	JJ

Figura 108 - Ejemplo de lista de frecuencias de una consulta

rcqp también permite extraer listas de frecuencias de las consultas, pero se declinó usarlas por los siguientes motivos:

- Con el fin de armonizar la ejecución de las consultas, se optó por hacerlas siempre a través de la clase CQP.
- rcqp sólo permite la extracción de la lista de frecuencias de un parámetro por consulta por defecto. Si la lista de frecuencias tiene más de un parámetro (por ejemplo, dos palabras) no se valoran.

(2) Colocaciones

Una colocación es una secuencia de palabras que aparecen juntas con frecuencia en un texto. Tal y como sucede en la lista de frecuencias de la consulta, las colocaciones emplean los métodos **execute**, **query** y **set_corpus** de la clase CQP para realizar la consulta nuevamente.

Emulando el método empleado en P-ACTRES 2.0, las colocaciones de la consulta se basan en el contexto de los resultados obtenidos, variando de -5 a +5 las posiciones adyacentes y realizando un conteo de cada emparejamiento.

Para obtener las distintas frecuencias del contexto desde -5 a +5 del emparejamiento de la consulta, se realiza una lista de frecuencias de cada rango. Es decir, considerando *match[0]* el emparejamiento de la consulta realizada se realizan conteos desde -5 hasta +5 empleando para ello la funcionalidad **count** y **on** del lenguaje de consultas CQP Query Language a través del método **execute** de la clase CQP.

Adicionalmente, el usuario debe seleccionar cómo desea organizar la lista de frecuencias de la consulta: por palabras, por lemas, por etiquetas gramaticales o por etiquetas semánticas. Para ello se utiliza la funcionalidad **by** previamente descrita.

Por ejemplo, el usuario desea saber las colocaciones de una consulta de acuerdo con su etiqueta gramatical, los comandos introducidos en el método execute de la clase CQP serían:

count by POS on match[0]
count by POS on match[-5]..match[0]
count by POS on match[-4]..match[0]
...
count by POS on march[0].. match[4]
count by POS on match[0]..match[5]

Todas estas frecuencias se agrupan por medio del método *freqCollocateListConversion* de la clase '*Estadisticas'*, para posteriormente realizar el conteo de cada una de las palabras (o etiquetas gramaticales, o lemas o etiquetas semánticas) del entorno de la consulta buscada. Por último, se transforma en formato XLSX (Excel) a través del método *freqCollocateAllConversion* de la clase '*Estadisticas'*, por medio de un script propio en R.

Una referencia al archivo XLSX (Excel) se envía de vuelta al controlador. Una ejemplo de archivo XLSX (Excel) es mostrada en la Figura 109.

	A	B	C	D	E	F	G	H	I	J	K	L
1	-5	-4	-3	-2	-1	target	1	2	3	4	5	Total
2	1	3	0	3	5	the	0	1	0	2	0	15
3	0	0	0	1	0	and	0	0	0	3	1	5
4	0	0	0	0	1	a	0	1	1	0	1	4
5	0	1	1	1	0	of	0	0	0	0	0	3
6	0	0	0	1	0	on	1	0	0	0	1	3
7	1	0	2	0	0	to	0	0	0	0	0	3
8	1	0	1	0	0	bunker	0	0	0	1	0	3
9	1	0	0	0	0	.	2	0	0	0	0	3
10	0	0	0	0	0	,	2	1	0	0	0	3
11	0	0	0	0	2	18th	0	0	0	0	0	2
12	0	0	0	0	2	9th	0	0	0	0	0	2
13	0	1	0	0	0	in	0	0	1	0	0	2
14	1	0	0	0	0	into	0	1	0	0	0	2
15	0	0	0	0	0	down	0	0	1	0	1	2
16	0	0	0	0	0	that	0	0	1	1	0	2
17	0	0	0	0	0	had	0	0	0	0	2	2
18	0	0	0	0	1	bright	0	0	0	0	0	1
19	0	0	0	1	0	approach	0	0	0	0	0	1
20	0	0	0	1	0	bunkered	0	0	0	0	0	1
21	0	0	0	1	0	guarding	0	0	0	0	0	1
22	0	0	0	1	0	over	0	0	0	0	0	1
23	0	0	0	1	0	preserve	0	0	0	0	0	1
24	0	0	1	0	0	at	0	0	0	0	0	1
25	0	0	1	0	0	ball	0	0	0	0	0	1

Figura 109 - Ejemplo de lista de colocaciones de una consulta

Aunque en la actualidad frameworks para el tratamiento de corpus lingüísticos como SketchEngine o CQPweb emplean colocaciones estadísticas basadas en tests de significación como *Z-score, Mutual Information* y *Log-likelihood*, en *ACTRES Corpus Manager* se ha optado por facilitar únicamente colocaciones basadas en frecuencias como consecuencia de los criterios establecidos en los requisitos de *ACTRES Corpus Manager*. Las colocaciones estadísticas se diferencian de las colocaciones por frecuencias en que implican que una secuencia de palabras se da en un texto más a menudo que lo que cabría esperar por azar

(3) Lista de frecuencias de un corpus

La lista de frecuencia de un corpus muestra la frecuencia de aparición de cada palabra existente en un corpus lingüístico. La lista de frecuencias de un corpus emplea la funcionalidad proporcionada por rcqp a través de la función **cqp_flist**. En este caso, al no existir consulta, sólo la clase '*Estadisticas*' es empleada. Ni la clase CQP ni la clase CQPConfigData son utilizadas.

El usuario debe seleccionar cómo desea organizar la lista de frecuencias del corpus: por palabras, por lemas, por etiquetas gramaticales o por etiquetas semánticas.

La función **cqp_flist** es utilizada en un script R propio, que además transforma el resultado en archivo XLSX (Excel) (ver Figura 110).

	A	B
1	the	6372
2	,	5187
3	.	4171
4	of	2859
5	be	2850
6	and	2181
7	to	2132
8	a	2122
9	in	2020
10	"	1434
11	for	969
12	'	930
13	have	863
14	that	842
15	on	689
16	he	657
17	at	632
18	with	564
19	as	517
20	it	504

Figura 110 - Ejemplo de lista de frecuencias de un corpus

(4) Lista de palabras clave

La lista de palabras clave es una abstracción de la lista de frecuencias en comparación con otro corpus que se denomina internamente 'corpus de referencia'. La lista de palabras clave de un corpus emplea el paquete estadístico polmineR, más concretamente su funcionalidad **compare**, que permite comparar dos corpus y extraer el listado de palabras clave por medio de un tests de significación estadístico (*chi-square* o *log-likelihood*).

Tal y como sucede en la extracción de la lista de frecuencias de un corpus, al no existir consulta, sólo la clase '*Estadisticas*' es empleada. El usuario debe seleccionar el tipo de tests de significación a utilizar y el corpus de referencia que desea utilizar para la comparación.

La función **compare,** junto a otras funciones requeridas de polmineR, son utilizadas en un script R propio que además transforma el resultado en un archivo XLSX (Excel) (ver Figura 111).

	A	B	C	D	E	F	G
1		word	count_coi	count_ref	exp_coi	chisquare	rank_chisquare
2	1	Cup	2	-1	0,001777951	2249,803979	1
3	2	revealed	2	-1	0,001777951	2249,803979	2
4	3	incredible	2	0	0,003555903	1122,912049	3
5	4	boss	2	1	0,005333854	747,2825675	4
6	5	finish	2	1	0,005333854	747,2825675	5
7	6	believed	2	2	0,007111806	559,4686978	6
8	7	drawn	2	2	0,007111806	559,4686978	7
9	8	events	2	2	0,007111806	559,4686978	8
10	9	features	2	2	0,007111806	559,4686978	9
11	10	remain	2	2	0,007111806	559,4686978	10
12	11	changed	2	3	0,008889757	446,7810729	11
13	12	mood	2	3	0,008889757	446,7810729	12
14	13	Roger	2	7	0,016001563	246,4513898	13
15	14	stay	2	7	0,016001563	246,4513898	14
16	15	having	2	11	0,023113369	169,4058013	15
17	16	winning	2	11	0,023113369	169,4058013	16
18	17	coach	2	12	0,02489132	157,024097	17
19	18	whether	2	16	0,032003126	121,2566656	18
20	19	Premier	2	17	0,033781078	114,6683869	19
21	20	Thomas	2	19	0,03733698	103,374693	20
22	21	Wednesday	2	21	0,040892883	94,04572612	21
23	22	able	2	22	0,042670835	89,96452102	22
24	23	League	2	24	0,046226738	82,74432971	23
25	24	club	2	30	0,056894446	66,50151465	24

Figura 111 - Ejemplo de lista de palabras clave de acuerdo con el método estadístico *chi-square*

(5) n-gramas

Un n-grama es una secuencia palabras que se da en un texto de forma más frecuente que otra. Así un n-grama de tamaño uno es un unigrama, de tamaño dos, bigrama o diagrama, etc.

CWB soporta nativamente la extracción de n-gramas a través de la utilidad CWB denominada *cwb-scan-corpus*. Sin embargo, es un proceso que requiere mucha memoria y tiene escasa flexibilidad a la hora de tratar los resultados y de establecer parámetros de consulta de n-gramas.

Por este motivo se emplea funcionalidad **ngramObject** del paquete estadístico polmineR. Tal y como sucede en la extracción de la lista de frecuencias de un corpus, al no existir consulta sólo la clase '*Estadisticas*' es empleada.

El usuario sólo ha de seleccionar obligatoriamente el valor de n (Figura 112), siendo el resto de parámetros optativos (introducción de secuencias de texto y sus palabras, lemas, etiquetas gramaticales y semánticas asociadas) .

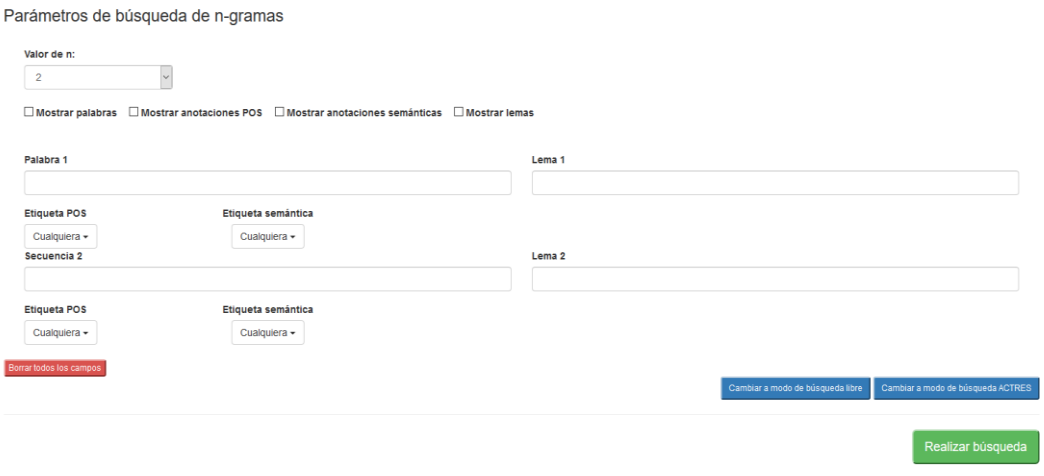


Figura 112 - Interfaz de búsqueda de n-gramas

La función **ngramObject** junto al establecimiento de los distintos parámetros de las opciones estadísticas de la a búsqueda de n-gramas se realiza en un script R

propio que además transforma el resultado en un archivo XLSX (Excel) (ver Figura 113).

	A	B	C	D
1	1	,	del	4
2	2	a	Wenger	2
3	3	Wenger	para	2
4	4	para	ganar	2
5	5	ganar	la	2
6	6	la	FA	2
7	7	FA	Cup	2
8	8	Cup	y	2
9	9	y	tener	2
10	10	tener	un	2
11	11	un	increble	2
12	12	increble	final	2
13	13	final	de	2
14	14	de	Premier	2
15	15	Premier	League	2
16	16	League	para	2
17	17	para	que	2
18	18	que	l	2
19	19	l	pudiera	2
20	20	pudiera	transformar	2

Figura 113 - Ejemplo de búsqueda de n-gramas de tamaño 2

6.3.4.3.3 Diagrama general de la implementación del modelo del framework

La Figura 114 muestra un diagrama general del modelo creado, haciendo especial hincapié la interacción de los distintos componentes que forman parte de CWB, los datos del framework y la lógica programada en PHP. Se excluyen las relaciones con el controlador y la vista.

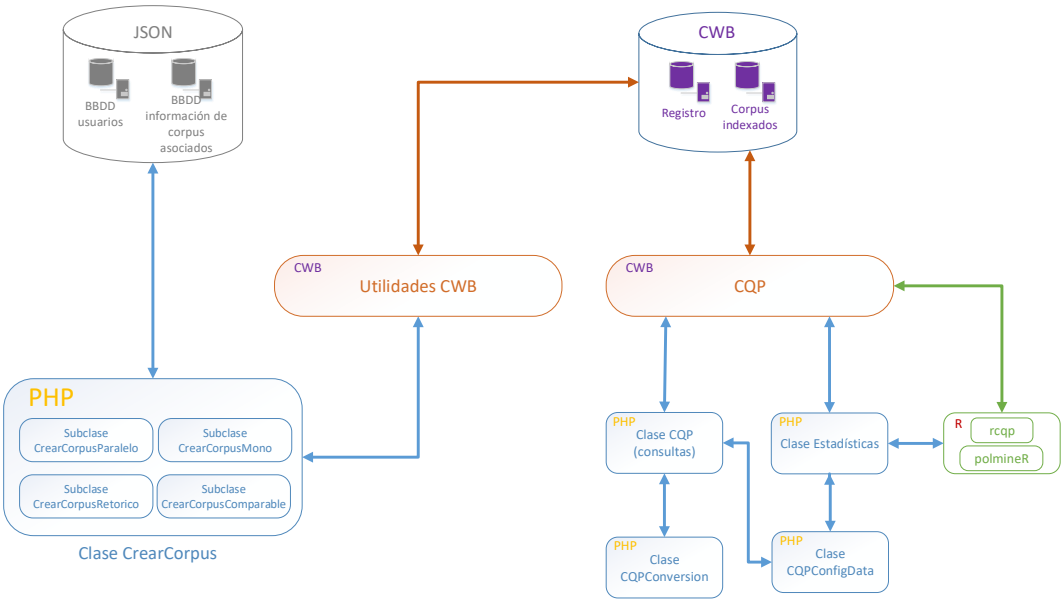


Figura 114 - Diagrama general de la implementación del modelo de *ACTRES Corpus Manager*

Capítulo

7

Validación del framework

El objetivo de la validación es comprobar que ACTRES Corpus Manager realiza las funciones descritas en el Capítulo 6: "Análisis, diseño e implementación". Para ello se han llevado a cabo dos tipos de pruebas: una prueba conceptual por parte del autor de la tesis doctoral, con el objetivo de testear y depurar errores internos de funcionabilidad relacionados con la lógica de la aplicación; y una prueba usabilidad del software, por parte de usuarios externos que tienen como finalidad medir el grado de usabilidad de ACTRES Corpus Manager por parte de usuarios potenciales de la aplicación.

7.1 Prueba conceptual

7.1.1 Introducción

Tal y como se definió en el Capítulo 5: "Metodología", el término prueba conceptual utilizado en esta tesis se corresponde con una fase de testeo y depuración de errores ejecutada por el desarrollador del software.

El objetivo de la prueba conceptual es comprobar que no exista ningún fallo en las distintas funcionalidades que el software puede ejecutar. En el caso de que exista

un error, éste ha de ser corregido, realizándose la tarea de comprobación de nuevo. Esta prueba conceptual es realizada por el autor de la tesis, utilizando recursos lingüísticos compilados para tal fin.

Para probar las funcionalidades de *ACTRES Corpus Manager* se ejecutarán los casos de uso definidos en el subapartado 6.1.2 Especificación de casos de uso, individualizando los distintos tipos de corpus en las consultas y estadísticas:

(1) crear un corpus monolingüe.
(2) crear un corpus comparable.
(3) crear un corpus paralelo multilingüe.
(4) crear un corpus monolingüe retórico.
(5) consultar y extraer estadísticas de un corpus monolingüe.
(6) consultar y extraer estadísticas de un corpus comparable.
(7) consultar y extraer estadísticas de un corpus paralelo multilingüe.
(8) consulta y extraer estadísticas de un corpus monolingüe retórico.

7.1.2 Recursos lingüísticos utilizados

Para la ejecución de las pruebas conceptuales se han utilizado dos tipos de recursos lingüísticos de acuerdo con su modo de incorporación a *ACTRES Corpus Manager*:

(1) Corpus integrados directamente al sistema de gestión, es decir, sin utilizar el framework. Al encontrarse indexados en formato reconocido por CWB, se introducen en el sistema de gestión de corpus directamente y se actualiza la base de datos. Estos corpus son utilizados como corpus de comparación para la extracción de palabras clave o como corpus sobre los que realizar consultas masivas y de esta forma comprobar la robustez del software.
(2) Corpus recopilados con el único objetivo de ser usados como corpus de prueba y que fueron incorporarlos al sistema de gestión de corpus mediante el framework. *ACTRES Corpus Manager* no pretende ser un sustituto de plataformas de acceso a grandes corpus como corpus.bye.edu o el portal CQPweb de *Lancaster University* – Universidad de Lancaster (Hardie,

2016), sino que es un software que permite la creación de distintos tipos de corpus anotados propios, de cualquier tamaño por parte de investigadores individuales sin conocimientos de programación. Por este motivo se ha creído conveniente que las pruebas de creación, consulta y extracción de estadísticas se realicen sobre corpus lingüísticos recopilados de este modo.

Los recursos lingüísticos específicos empleados y su uso en las pruebas conceptuales se muestran en la siguiente tabla:

Corpus	**Utilización**
Corpus monolingüe BE-Informe Anual (2008-2015) (español)	• Corpus monolingüe en español recopilado en exclusiva para la realización de las pruebas conceptuales y que está formado por los informes anuales emitidos por el Banco de España (Banco de España, 2015) desde el año 2008 hasta el 2015. • Se ha utilizado con el objetivo de crear, consultar y extraer estadísticas de un corpus monolingüe.
Corpus paralelo BEBCE-Evolución Económica-2016 (inglés-español)	• Corpus paralelo en inglés-español recopilado en exclusiva para la realización de las pruebas conceptuales y que está formado por la sección "Evolución económica y monetaria" de los informes procedentes del Boletín Económico del Banco Central Europeo del año 2016, publicados por el Banco de España en inglés (Banco de España, 2016b) y español (Banco de España, 2016a) .

	• Se ha utilizado con el objetivo de crear, consultar y extraer estadísticas de un corpus paralelo multilingüe.
Corpus comparable C-GARE	• Corpus comparable compuesto por actas de reunión en inglés y en español perteneciente al grupo ACTRES (ACTRES, 2017a). • Se ha utilizado con el objetivo de crear, consultar y extraer estadísticas de un corpus comparable multilingüe.
Corpus retórico C-GARE 001-005 ES	• Corpus formado por 5 documentos del subcorpus español del C-GARE que se han etiquetado retóricamente. Concretamente los documentos: o 001MTwsXX100630BsnsEs o 002MTwsXX041111BsnsEs o 003MTwsXX070110BsnsEs o 004MTwsXX100310BsnsEs o 005MTwsXX090219BsnsEs • Se ha utilizado con el objetivo de crear, consultar y extraer estadísticas de un corpus monolingüe con etiquetado retórico.
Corpus bidireccional P-ACTRES 2.0 (español<->inglés)	• Corpus incorporado directamente en sistema de gestión de corpus. Está formado por: o 2 corpus paralelos: • Español original - Inglés traducido • Inglés original - Español traducido

	○ Que a su vez son comparables entre sí del modo: • Español original - Español traducido • Inglés original - Inglés traducido • Español original - Inglés original • Español traducido - Inglés traducido • Se ha utilizado con el objetivo de servir como corpus comparable y paralelo sobre el que realizar consultas masivas y evaluar la robustez del sistema. • Además también se ha empleado para crear, consultar y extraer estadísticas de un corpus comparable en un solo idioma.
Corpus paralelo EUROPARL (inglés-español-francés-italiano)	• Corpus incorporado directamente en sistema de gestión de corpus. Se ha utilizado para tres objetivos: ○ Corpus de comparación para la extracción de palabras clave. ○ Corpus paralelo multilingüe formado por 4 subcorpus sobre el que realizar consultas masivas y evaluar la robustez del sistema. ○ Como cada subcorpus es considerado también un corpus monolingüe, cada uno de ellos se ha empleado como un corpus

	monolingüe sobre el que realizar consultas masivas en los distintos idiomas y de este modo evaluar la robustez del sistema.

Tabla 60 - Recursos lingüísticos empleados y su utilización en las pruebas conceptuales

7.1.3 Resultados

Mostrar el proceso completo de creación, consulta y extracción de todos los tipos de corpus soportados por el sistema en el presente documento aumentaría la extensión del mismo y no resultaría provechoso debido a que, por un lado, los procesos fueron descritos en detalle en el capítulo anterior (consultar Capítulo 6: "Análisis, diseño e implementación de *ACTRES Corpus Manager*"), y por otro, muchos de ellos comparten lógica y vistas.

Por este motivo, no se mostrará todo el proceso de creación, consulta y extracción de estadísticas completo de cada tipo de corpus lingüístico soportado por *ACTRES Corpus Manager*, sino que se proporcionarán los ejemplos más relevantes en cada uno de los casos.

7.1.3.1 Primeros pasos

La pantalla inicial (Figura 115) muestra las funcionalidades disponibles de *ACTRES Corpus Manager*, crear y consultar corpus, así como información relativa a los programas software de terceros empleados. En el menú principal hay un acceso directo a cada una de estas funcionalidades, así como acceso a la página de inicio, zona de usuario e información adicional sobre *ACTRES Corpus Manager*.

Figura 115 - Pantalla inicial de *ACTRES Corpus Manager*

Para emplear *ACTRES Corpus Manager* es necesario estar registrado y disponer de una contraseña (ver Figura 116).

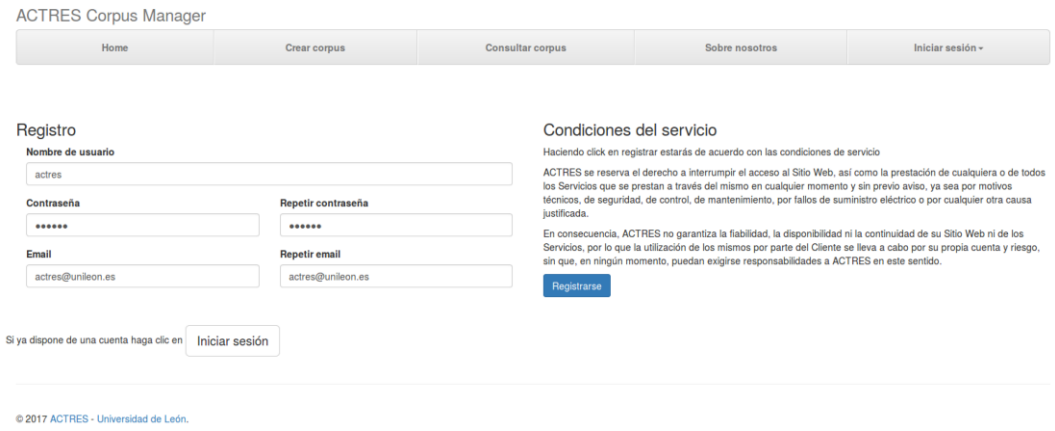


Figura 116 - Pantalla de registro de *ACTRES Corpus Manager*

Una vez introducidos los datos de acceso, se podrá crear un corpus o acceder a los que estén vinculados a la cuenta de usuario.

7.1.3.2 Creación de un corpus lingüístico

Los pasos seguidos para la creación de corpus lingüísticos son muy similares sea cual sea el tipo de corpus seleccionado. Esta similitud se debe a CWB sólo reconoce corpus monolingües en su sistema de indexación (ver página 201), por lo que es evidente que el proceso de creación es muy similar y las diferenciaciones son manejadas por la lógica del corpus, explicadas detalladamente en la sección 6.3 Implementación.

Las diferencias más reseñables entre los distintos tipos de corpus y el corpus monolingüe son:

- Corpus bi-/multilingüe paralelo:
  - Es necesario especificar el número de subcorpus que forman parte del corpus.
  - Es necesario subir los documentos del corpus con un nombre determinado.
- Corpus comparable:
  - El proceso es análogo a la creación de corpus paralelos a excepción de que la utilización de nombres específicos no es necesaria.
- Corpus monolingüe retórico:
  - El etiquetado retórico es incluido únicamente en el caso específico de construcción de corpus monolingüe etiquetado retóricamente, luego la única diferencia con respecto a la creación de corpus monolingües es que es necesario etiquetar manualmente los documentos de los corpus por medio del interfaz de usuario (Ver figuras Figura 117 y Figura 118). Por este motivo se recomienda que el tamaño del corpus no sea demasiado grande.[57]

[57] Para más información consultar la descripción del etiquetado retórico en la página 177.

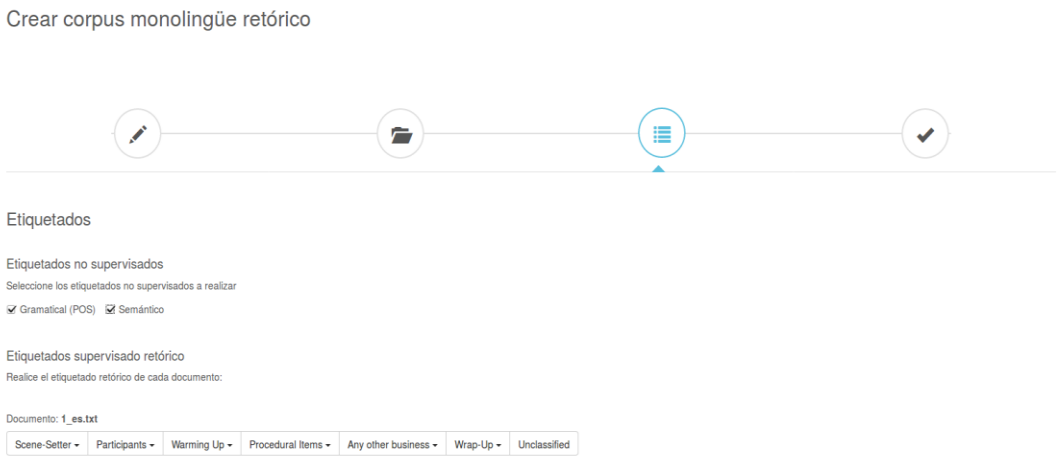


Figura 117 - Pantalla de etiquetados para la construcción de corpus monolingües etiquetados retóricamente

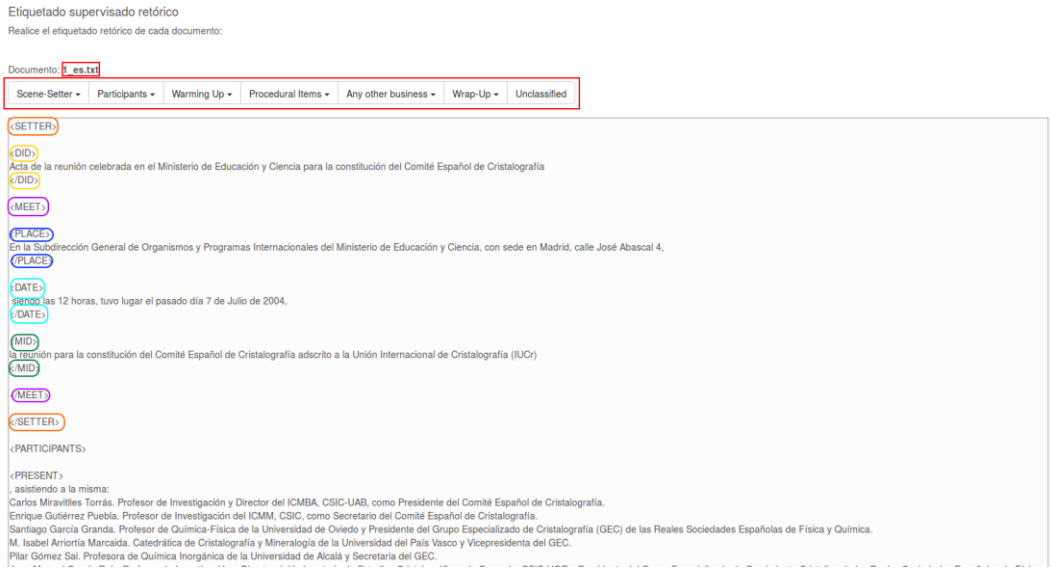


Figura 118 - Fragmento de etiquetado retórico en español

A continuación, se muestra el proceso de creación de un corpus lingüístico, apoyándose en un ejemplo de creación de corpus paralelo que utiliza el corpus BEBCE-Evolución Económica-2016 (inglés-español).

(1) Se presiona sobre "*Crear corpus*" o "*Crea tu primer corpus*" (Figura 119).

Figura 119 - Paso 1: Creación de corpus paralelo

(2) Posteriormente, se muestran los distintos tipos de corpus disponibles, así como información relativa a su implementación. Se presionará sobre el botón "*Crear*" del tipo de corpus bi-/multilingüe paralelo (Figura 120).

Figura 120 - Paso 2: Creación de corpus paralelo

(3) Aparecerá una nueva vista, denominada "*Número de subcorpus*" (Figura 121). Se debe especificar el número de subcorpus que formará el corpus paralelo. Este paso es exclusivo de la creación de corpus paralelos y comparables.

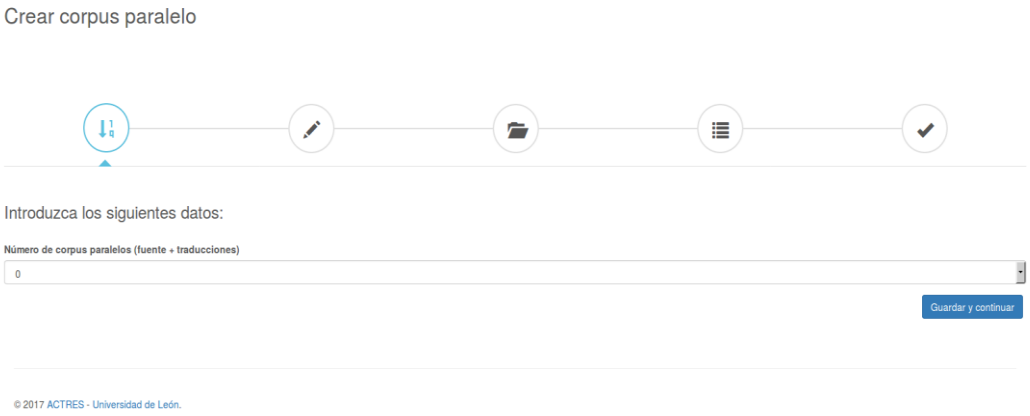


Figura 121 - Paso 3: Creación de corpus paralelo

El corpus paralelo que se desea crear está formado por dos subcorpus así que se seleccionará "*2*" y se presionará el botón "*Guardar y continuar*".

(4) La siguiente vista se denomina "*Datos*" y es común para todos los tipos de corpus soportados por el sistema. En esta vista se introducen los siguientes datos:

- **Nombre del corpus**.
  - En el caso de los corpus paralelos y comparables, también se ha de especificar el nombre de los subcorpus.
- **Nombre raíz del corpus**: Se utilizará para identificar a los ficheros y como abreviatura interna del corpus.
- **Idioma del corpus**.
  - En el caso de los corpus paralelos y comparables, se ha de especificar el idioma de cada uno de los subcorpus.

En el ejemplo mostrado se introducirá la siguiente información (Figura 122):

- **Nombre del corpus**: "*BEBCE-Evolución Económica-2016*"
- **Nombre raíz**: "*bebceee16*".

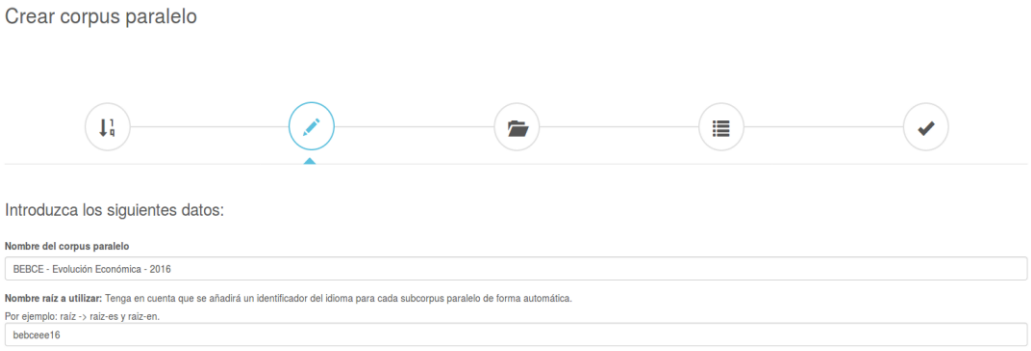


Figura 122 - Paso 4: Creación de corpus paralelo 1/2

Y la correspondiente información relativa a los subcorpus (Figura 123):

- **Nombre del subcorpus original**: "*BEBCE-Evolución Económica-2016-EN*".
- **Idioma del subcorpus original**: "*Inglés*".
- **Nombre del subcorpus traducido**: "*BEBCE-Evolución Económica-2016-ES*"
- **Idioma del subcorpus traducido**: "*Español*".

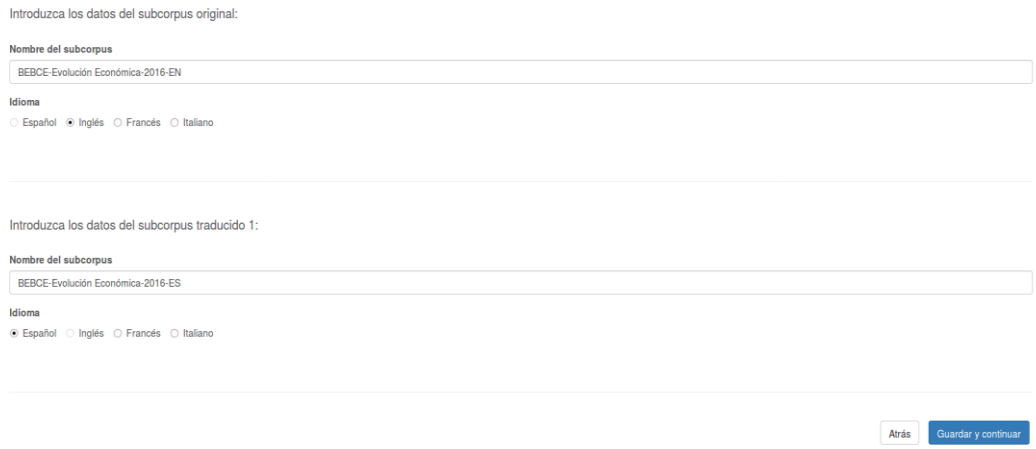


Figura 123 - Paso 4: Creación de corpus paralelo 2/2

(5) Una vez completados los datos, se han de subir los documentos que formarán el corpus. En el caso de los corpus paralelos y comparables, la subida de documentos se hará sobre cada subcorpus (Figura 124).

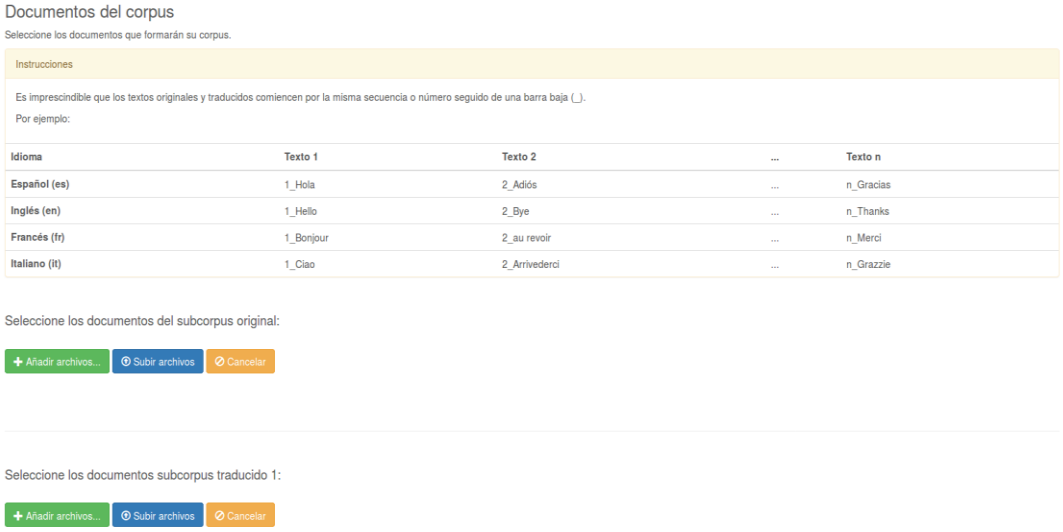


Figura 124 - Paso 5: Creación de corpus paralelo 1/3

Además, los documentos que forman los corpus paralelos tienen la peculiaridad de que han de tener un nombre de documento concreto. Es necesario que cada par documento original – documento traducción

comience por la misma secuencia seguido de una barra baja (_). El motivo es que de esta forma el framework sabe qué documentos han de alinearse entre sí.[58]

El corpus paralelo utilizado en el ejemplo está formado por 8 pares de documentos inglés-español (Figura 125 y Figura 126).

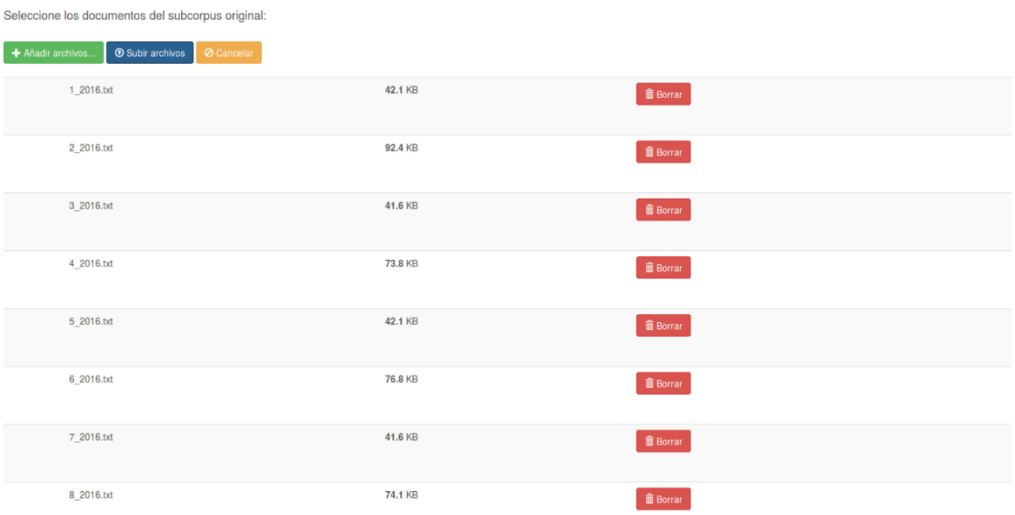


Figura 125 - Paso 5: Creación de corpus paralelo 2/3

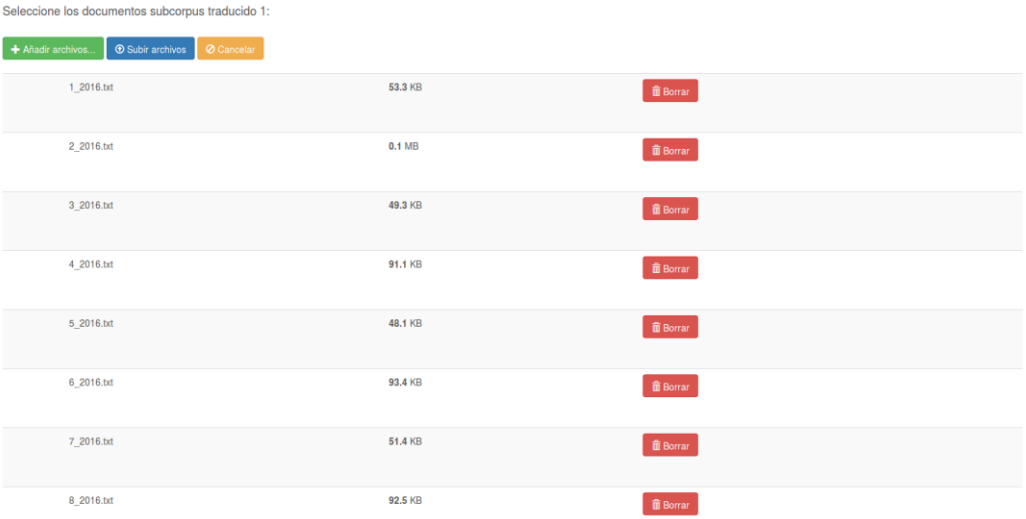


Figura 126 - Paso 5: Creación de corpus paralelo 3/3

[58] Para más información consultar la descripción de la actividad de alineado V) Alinear.

(6) El siguiente paso es seleccionar los etiquetados que se desean incluir en el corpus lingüístico: gramatical o semántico. En el ejemplo utilizado, se seleccionarán ambos etiquetados (Figura 127).

Figura 127 - Paso 6: Creación de corpus paralelo

(7) La siguiente vista sirve para verificar los datos del corpus a crear (Figura 128).

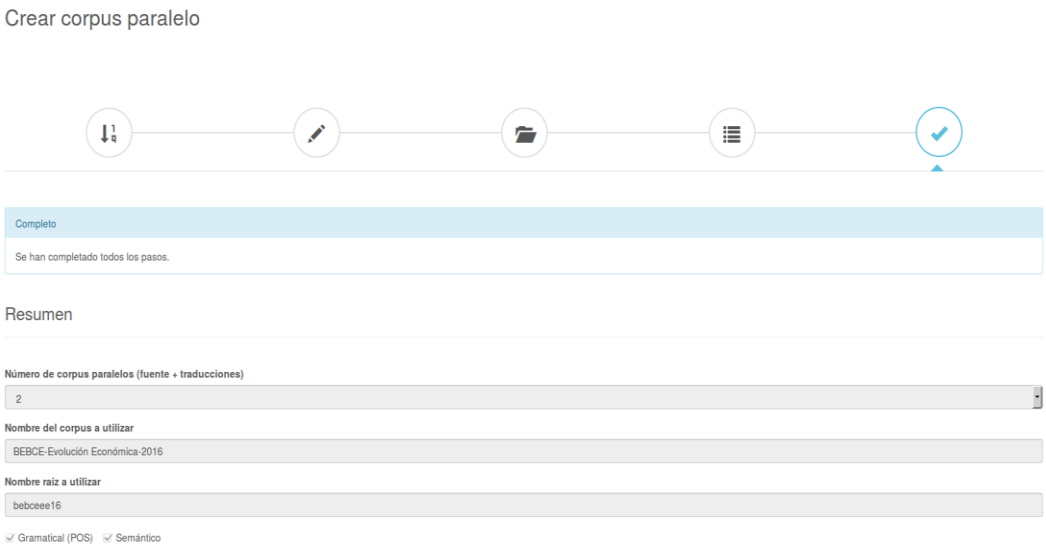


Figura 128 - Paso 7: Creación de corpus paralelo 1/2

En caso de que exista algún error, basta con acceder a la vista oportuna utilizando los iconos de la parte superior (Figura 129).

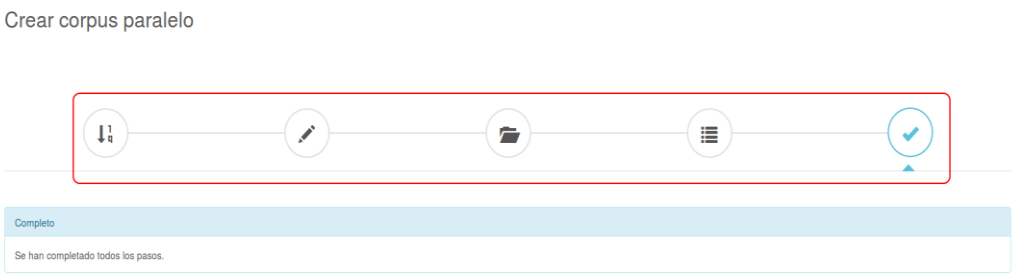


Figura 129 - Paso 7: Creación de corpus paralelo 2/2

(8) Después de presionar el botón ”*Construir corpus*”, la información recopilada y los documentos del corpus son enviados al servidor que creará el corpus lingüístico y lo registrará en el sistema. Dependiendo del tamaño, tipo y etiquetados del corpus, el proceso puede llevar más o menos tiempo (Figura 130).

Figura 130 - Paso 8: Creación de corpus paralelo 1/2

Al finalizar la creación del corpus, se informará al usuario de si el proceso se ha llevado correctamente o si ha habido algún error (Figura 131).

Figura 131 - Paso 8: Creación de corpus paralelo 2/2

7.1.3.3 Consulta de un corpus lingüístico

Tal y como sucede en la creación de corpus lingüísticos, es necesario estar registrado e iniciar sesión. Una vez realizada esta acción aparecerán todos los corpus a los que tiene acceso el usuario, clasificados en categorías: corpus monolingües, corpus multilingües paralelos, corpus comparables y corpus monolingües con etiquetado retórico (Figura 132).

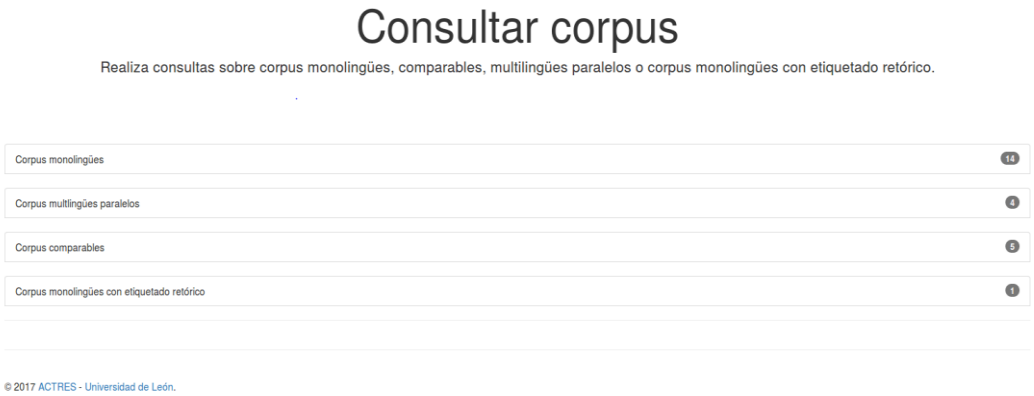


Figura 132 - Pantalla inicial de consulta de corpus en *ACTRES Corpus Manager*

Al presionar sobre cualquiera de las categorías aparecerá el nombre de los corpus pertenecientes a cada una de ellas junto a su idioma o idiomas y etiquetados incorporados (Figura 133).

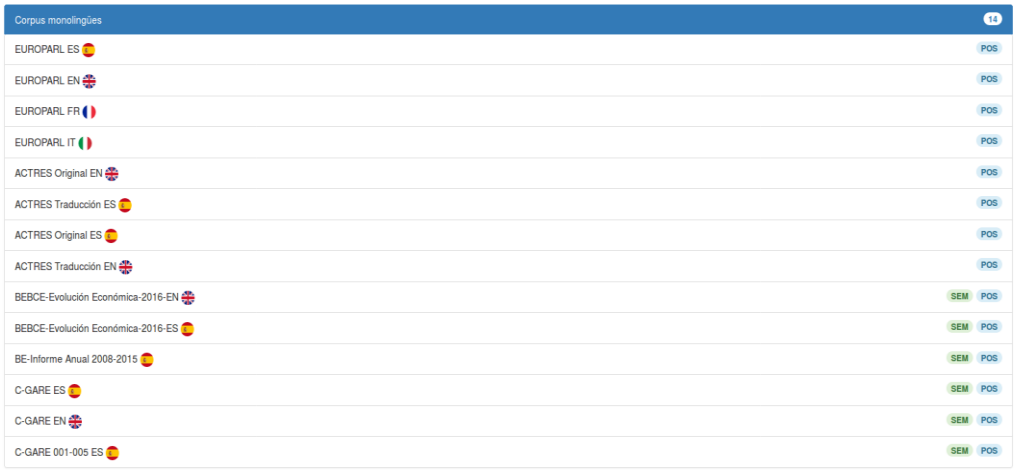


Figura 133 - Corpus monolingües disponibles, junto a su idioma y etiquetados

Existen dos tipos de interfaces de búsqueda, búsqueda libre y búsqueda ACTRES.

(1) La búsqueda libre (Figura 134) permite realizar búsquedas directamente empleando el lenguaje CQP Query Language, para lo que se proporciona información sobre los nombres internos del corpus, sobre los *tagsets* empleados y sobre el lenguaje de consultas. El interfaz de búsqueda es idéntico para cualquiera de los tipos de corpus soportados por *ACTRES Corpus Manager*.

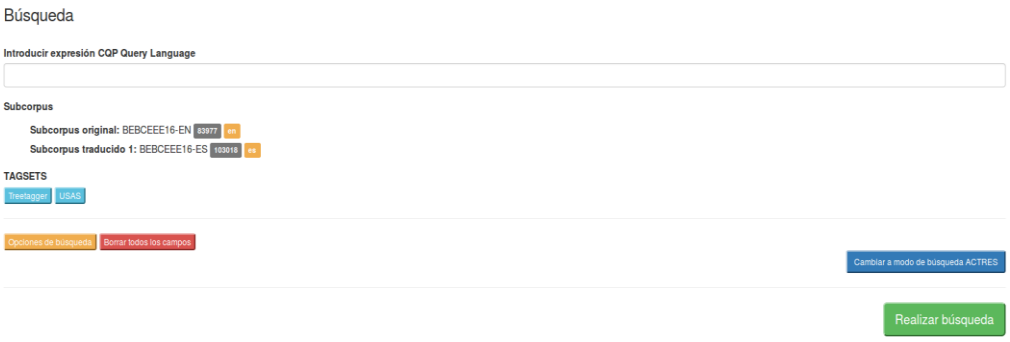


Figura 134 - Búsqueda libre en *ACTRES Corpus Manager*

(2) La búsqueda ACTRES (Figura 135), ideada originalmente por K. Hofland y Ø. Reigem en P-ACTRES (Izquierdo, Hofland, & Reigem, 2008), es una

búsqueda semi-asistida diseñada para todos los usuarios lingüistas, independiente del conocimiento del lenguaje de consultas CQP Query Language del que dispongan. Este interfaz de búsqueda posee tres secuencias de texto junto con distintos selectores para especificar los valores relativos al etiquetado lingüístico y al uso de expresiones regulares que se quieran especificar en la consulta (ver página 259 para más información).

Figura 135 - Búsqueda ACTRES o asistida en *ACTRES Corpus Manager*

El proceso de búsqueda y el interfaz es prácticamente idéntico en todos los tipos de corpus soportados, aunque poseen pequeñas diferencias. Con el objetivo de no repetir información, se realizará una búsqueda completa sobre el corpus paralelo previamente creado. Posteriormente, se realizarán búsquedas sobre el resto de corpus soportados de forma menos minuciosa. En los distintos ejemplos se utilizará el interfaz de búsqueda de tipo ACTRES, por resultar la opción más atractiva desde el punto de vista de la usabilidad del usuario lingüista.

7.1.3.3.1 Corpus paralelo bi-/multilingüe

(1) El primer paso es seleccionar el corpus. En cada entrada se muestra el nombre del corpus junto con su idioma y los etiquetados lingüísticos que incluye (Figura 136), en este caso se selecciona "*Boletín Económico BCE 2016 EN-ES*" dentro de la categoría "*Corpus multilingüe paralelo*".

Figura 136 - Selección de corpus multilingüe paralelos

(2) A continuación, se muestra la vista que incluye el interfaz de búsqueda, que por defecto se corresponde directamente con el tipo ACTRES (Figura 137).

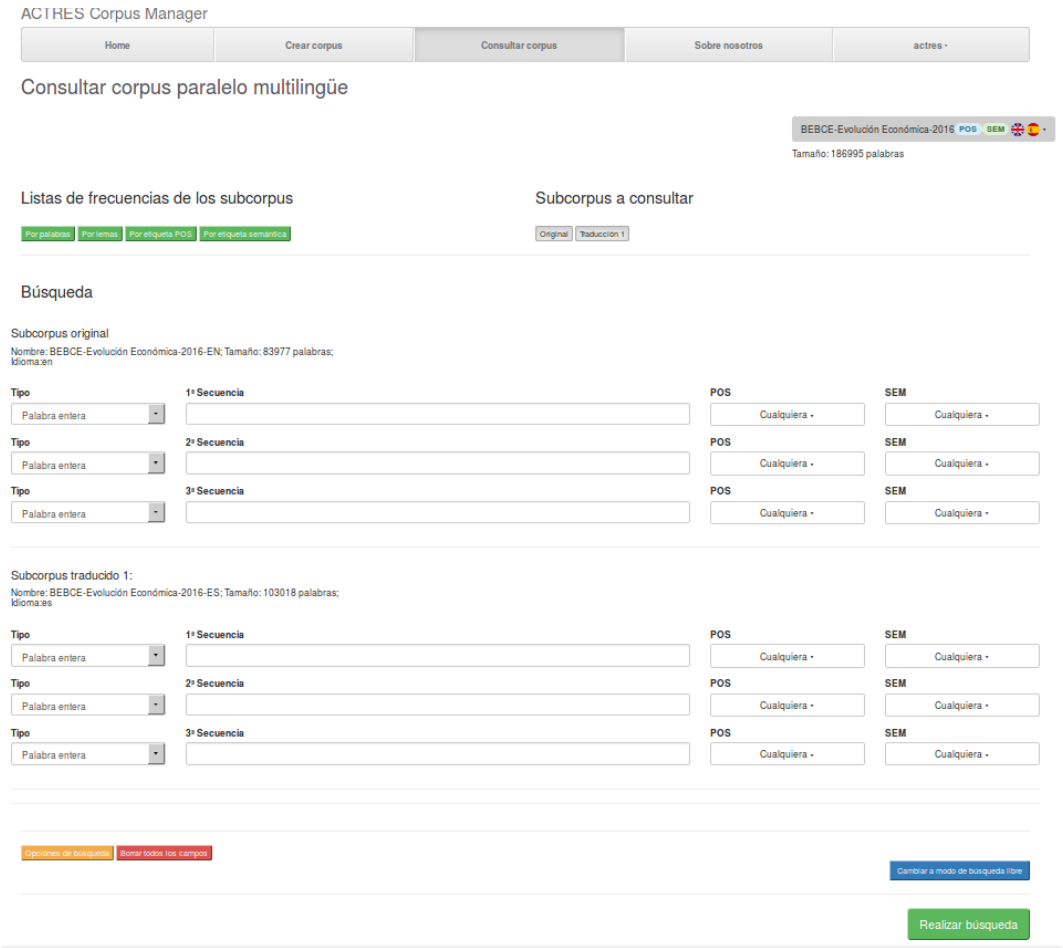


Figura 137 - Interfaz inicial de la consulta de corpus paralelos

(3) En la parte superior derecha del interfaz (Figura 138) aparece el nombre del corpus, junto con la información relativa a sus idiomas, etiquetado y número de palabras.

Figura 138 - Información del corpus paralelo multilingüe seleccionado

(4) La siguiente zona a destacar relacionada con la consulta de corpus lingüísticos es la zona "*Subcorpus a consultar*" (Figura 139). En el caso de disponer de corpus paralelos formados por dos o más traducciones, por medio de esta sección pueden ocultarse o mostrarse aquellos subcorpus que no sean relevantes para el análisis llevado a cabo por el lingüista. En el ejemplo ejecutado, sólo se dispone de un subcorpus traducido, luego esta opción no es práctica.

Figura 139 - Subcorpus a mostrar en la consulta de un corpus paralelo

(5) A continuación, se muestra el interfaz de búsqueda (Figura 140), con tantos formularios como subcorpus componga el corpus paralelo. De este modo se pueden realizar búsquedas por correspondencias entre el subcorpus original y las traducciones. En este ejemplo se disponen de dos formularios. Adicionalmente, en cada formulario, se muestra información relativa a la tamaño e idioma de cada subcorpus.

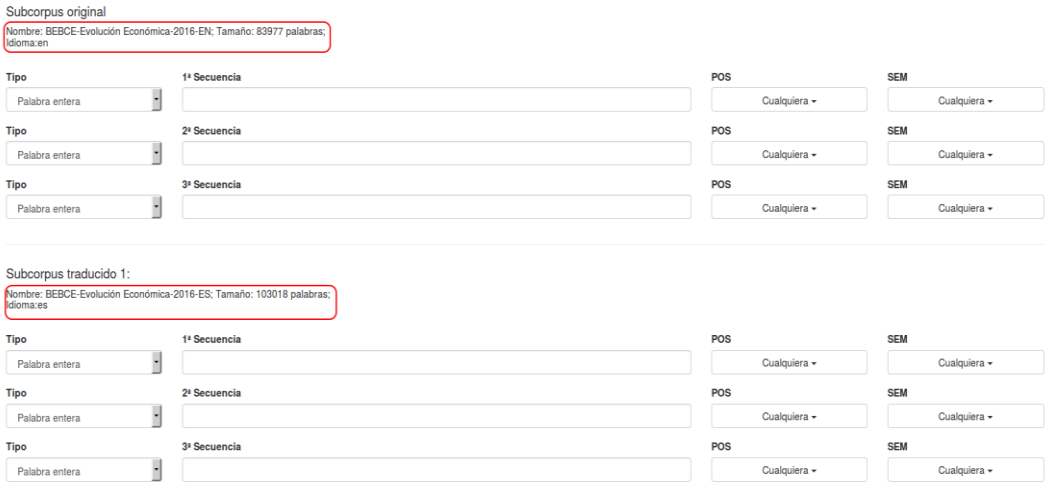


Figura 140 - Formularios de búsqueda en corpus paralelos multilingües

(6) A continuación, se mostrarán tres ejemplos de consulta:

i. **Consulta 1**: Un ejemplo de consulta puede ser la palabra "*HICP*", de la cual se desconoce su traducción al español (Figura 141).

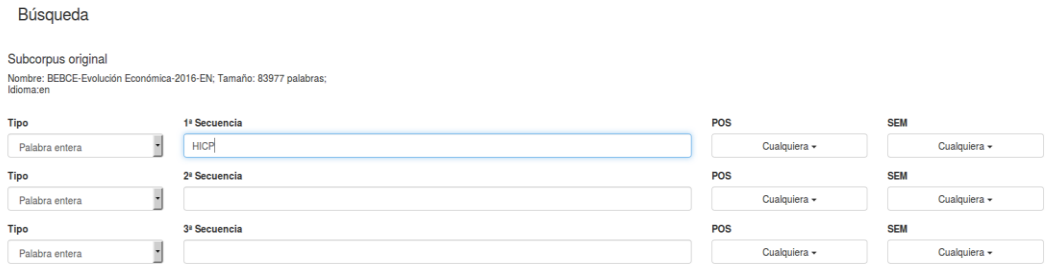
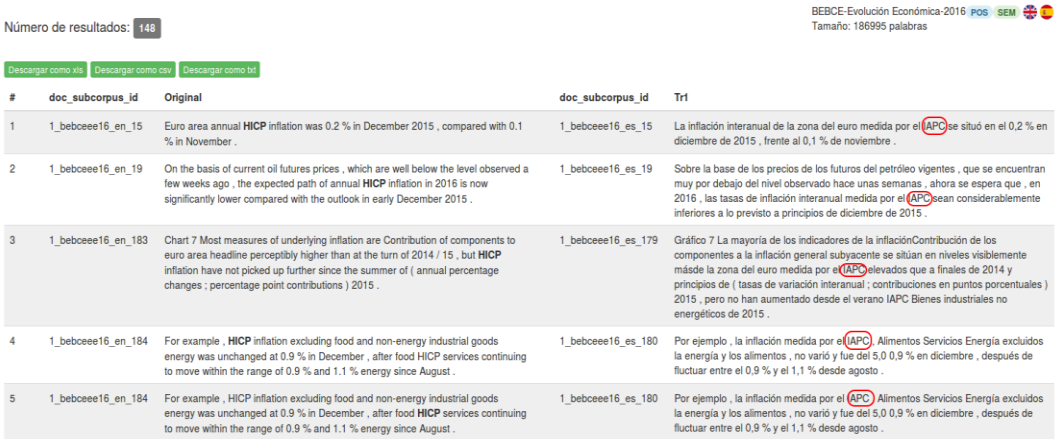


Figura 141 - Formulario de búsqueda para la consulta 1 en un corpus multilingüe paralelo

Aunque el corpus esté alineado por oraciones se puede deducir que la traducción es "*IAPC*" (Figura 142).



Número de resultados: 148

BEBCE-Evolución Económica-2016 POS SEM
Tamaño: 186995 palabras

Descargar como xls | Descargar como csv | Descargar como txt

#	doc_subcorpus_id	Original	doc_subcorpus_id	Tr1
1	1_bebceee16_en_15	Euro area annual **HICP** inflation was 0.2 % in December 2015 , compared with 0.1 % in November .	1_bebceee16_es_15	La inflación interanual de la zona del euro medida por el IAPC se situó en el 0,2 % en diciembre de 2015 , frente al 0,1 % de noviembre .
2	1_bebceee16_en_19	On the basis of current oil futures prices , which are well below the level observed a few weeks ago , the expected path of annual **HICP** inflation in 2016 is now significantly lower compared with the outlook in early December 2015 .	1_bebceee16_es_19	Sobre la base de los precios de los futuros del petróleo vigentes , que se encuentran muy por debajo del nivel observado hace unas semanas , ahora se espera que , en 2016 , las tasas de inflación interanual medida por el IAPC sean considerablemente inferiores a lo previsto a principios de diciembre de 2015 .
3	1_bebceee16_en_183	Chart 7 Most measures of underlying inflation are Contribution of components to euro area headline perceptibly higher than at the turn of 2014 / 15 , but **HICP** inflation have not picked up further since the summer of (annual percentage changes ; percentage point contributions) 2015 .	1_bebceee16_es_179	Gráfico 7 La mayoría de los indicadores de la inflaciónContribución de los componentes a la inflación general subyacente se sitúan en niveles visiblemente másde la zona del euro medida por el IAPC elevados que a finales de 2014 y principios de (tasas de variación interanual ; contribuciones en puntos porcentuales) 2015 , pero no han aumentado desde el verano IAPC Bienes industriales no energéticos de 2015 .
4	1_bebceee16_en_184	For example , **HICP** inflation excluding food and non-energy industrial goods energy was unchanged at 0.9 % in December , after food HICP services continuing to move within the range of 0.9 % and 1.1 % energy since August .	1_bebceee16_es_180	Por ejemplo , la inflación medida por el IAPC , Alimentos Servicios Energía excluidos la energía y los alimentos , no varió y fue del 5,0 0,9 % en diciembre , después de fluctuar entre el 0,9 % y el 1,1 % desde agosto .
5	1_bebceee16_en_184	For example , HICP inflation excluding food and non-energy industrial goods energy was unchanged at 0.9 % in December , after food **HICP** services continuing to move within the range of 0.9 % and 1.1 % energy since August .	1_bebceee16_es_180	Por ejemplo , la inflación medida por el IAPC , Alimentos Servicios Energía excluidos la energía y los alimentos , no varió y fue del 5,0 0,9 % en diciembre , después de fluctuar entre el 0,9 % y el 1,1 % desde agosto .

Figura 142 - Resultado de la consulta 1

ii. **Consulta 2**: otro ejemplo de consulta es conocer qué preposición acompaña a los verbos que terminan con el sufijo "*ease*" en el subcorpus en lengua inglesa (Figura 143), y como es este comportamiento traducido en el subcorpus español.

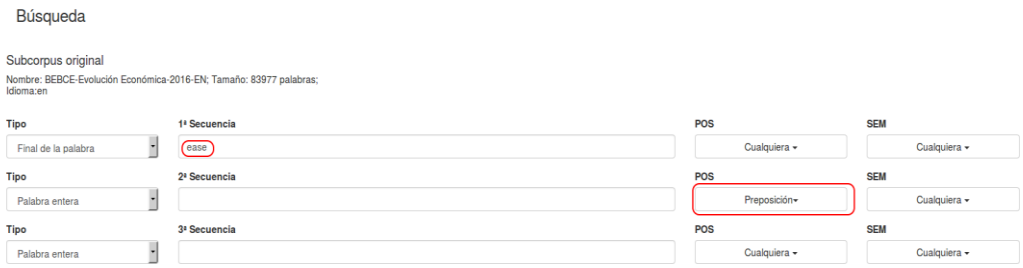
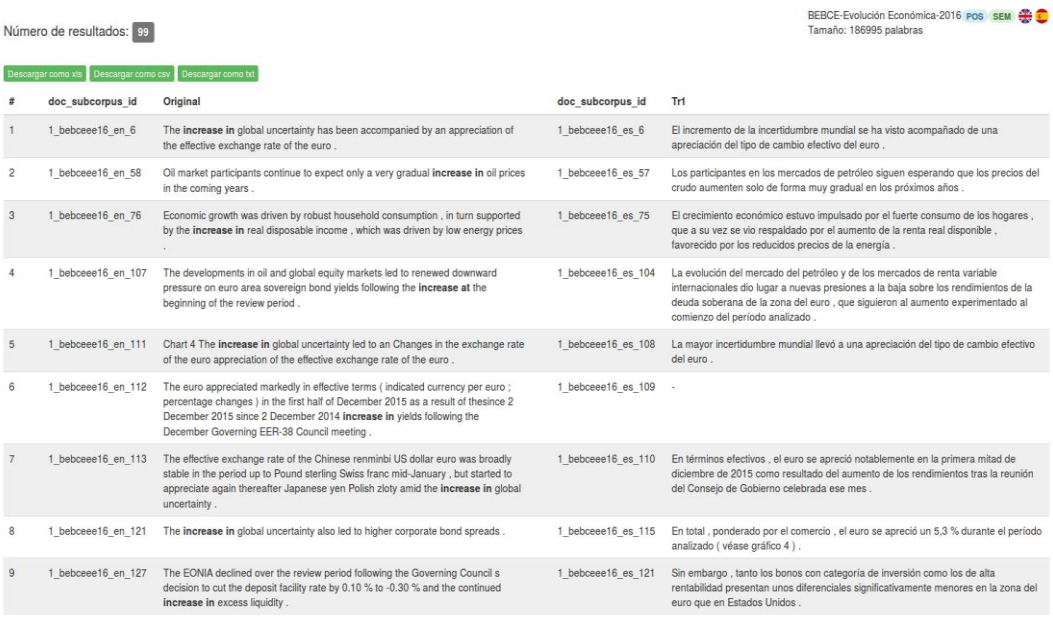


Figura 143 - Formulario de búsqueda para la consulta 2 en un corpus multilingüe paralelo

Los resultados de la consulta se muestran en la Figura 144.



Número de resultados: 99

BEBCE-Evolución Económica-2016 POS SEM
Tamaño: 186995 palabras

Descargar como xls | Descargar como csv | Descargar como txt

#	doc_subcorpus_id	Original	doc_subcorpus_id	Tr1
1	1_bebceee16_en_6	The **increase in** global uncertainty has been accompanied by an appreciation of the effective exchange rate of the euro .	1_bebceee16_es_6	El incremento de la incertidumbre mundial se ha visto acompañado de una apreciación del tipo de cambio efectivo del euro .
2	1_bebceee16_en_58	Oil market participants continue to expect only a very gradual **increase in** oil prices in the coming years .	1_bebceee16_es_57	Los participantes en los mercados de petróleo siguen esperando que los precios del crudo aumenten solo de forma muy gradual en los próximos años .
3	1_bebceee16_en_76	Economic growth was driven by robust household consumption , in turn supported by the **increase in** real disposable income , which was driven by low energy prices .	1_bebceee16_es_75	El crecimiento económico estuvo impulsado por el fuerte consumo de los hogares , que a su vez se vio respaldado por el aumento de la renta real disponible , favorecido por los reducidos precios de la energía .
4	1_bebceee16_en_107	The developments in oil and global equity markets led to renewed downward pressure on euro area sovereign bond yields following the **increase at** the beginning of the review period .	1_bebceee16_es_104	La evolución del mercado del petróleo y de los mercados de renta variable internacionales dio lugar a nuevas presiones a la baja sobre los rendimientos de la deuda soberana de la zona del euro , que siguieron al aumento experimentado al comienzo del período analizado .
5	1_bebceee16_en_111	Chart 4 The **increase in** global uncertainty led to an Changes in the exchange rate of the euro appreciation of the effective exchange rate of the euro .	1_bebceee16_es_108	La mayor incertidumbre mundial llevó a una apreciación del tipo de cambio efectivo del euro .
6	1_bebceee16_en_112	The euro appreciated markedly in effective terms (indicated currency per euro ; percentage changes) in the first half of December 2015 as a result of thesince 2 December 2015 since 2 December 2014 **increase in** yields following the December Governing EER-38 Council meeting .	1_bebceee16_es_109	-
7	1_bebceee16_en_113	The effective exchange rate of the Chinese renminbi US dollar euro was broadly stable in the period up to Pound sterling Swiss franc mid-January , but started to appreciate again thereafter Japanese yen Polish zloty amid the **increase in** global uncertainty .	1_bebceee16_es_110	En términos efectivos , el euro se apreció notablemente en la primera mitad de diciembre de 2015 como resultado del aumento de los rendimientos tras la reunión del Consejo de Gobierno celebrada ese mes .
8	1_bebceee16_en_121	The **increase in** global uncertainty also led to higher corporate bond spreads .	1_bebceee16_es_115	En total , ponderado por el comercio , el euro se apreció un 5,3 % durante el período analizado (véase gráfico 4) .
9	1_bebceee16_en_127	The EONIA declined over the review period following the Governing Council s decision to cut the deposit facility rate by 0.10 % to -0.30 % and the continued **increase in** excess liquidity .	1_bebceee16_es_121	Sin embargo , tanto los bonos con categoría de inversión como los de alta rentabilidad presentan unos diferenciales significativamente menores en la zona del euro que en Estados Unidos .

Figura 144 - Resultado de la consulta 2

iii. **Consulta 3**: Relacionada con la consulta anterior, se dispone a buscar la palabra en inglés anotada semánticamente con la categoría "*A2.1"* (de acuerdo con el tagset USAS se define como "*Affect: Modify, change* - Afectar: modificar, cambiar"), y que además se corresponde con la palabra en español "*incremento"* (Figura 145).

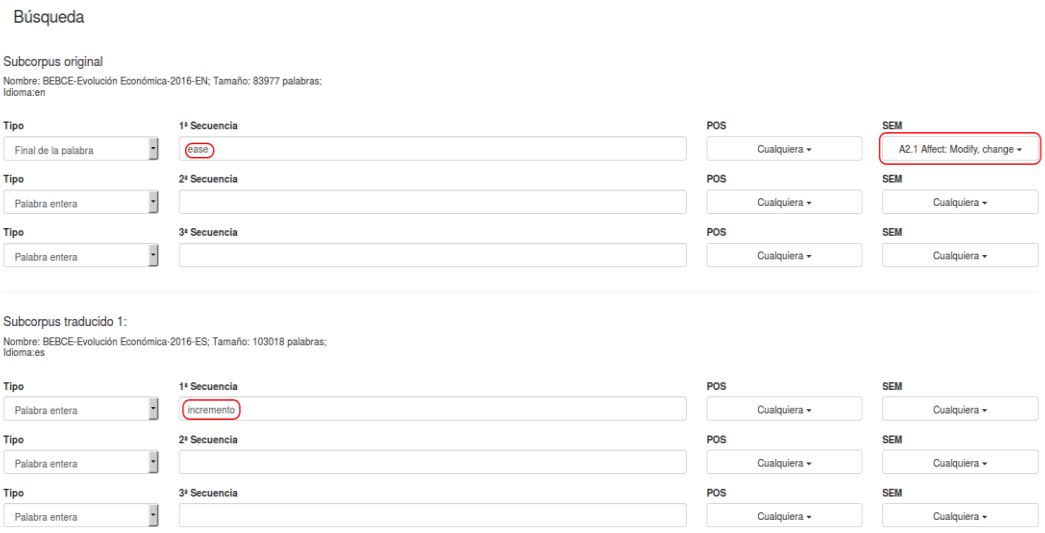
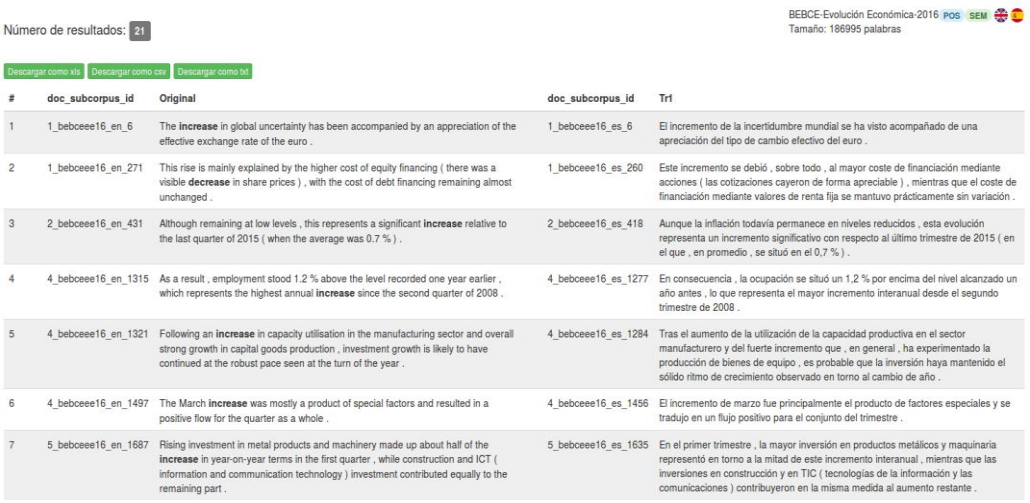


Figura 145 - Formulario de búsqueda para la consulta 3 en un corpus multilingüe paralelo

Los resultados de la consulta se muestran en la Figura 146.



#	doc_subcorpus_id	Original	doc_subcorpus_id	Tr1
1	1_bebceee16_en_6	The **increase** in global uncertainty has been accompanied by an appreciation of the effective exchange rate of the euro .	1_bebceee16_es_6	El incremento de la incertidumbre mundial se ha visto acompañado de una apreciación del tipo de cambio efectivo del euro .
2	1_bebceee16_en_271	This rise is mainly explained by the higher cost of equity financing (there was a visible **decrease** in share prices) , with the cost of debt financing remaining almost unchanged .	1_bebceee16_es_260	Este incremento se debió , sobre todo , al mayor coste de financiación mediante acciones (las cotizaciones cayeron de forma apreciable) , mientras que el coste de financiación mediante valores de renta fija se mantuvo prácticamente sin variación .
3	2_bebceee16_en_431	Although remaining at low levels , this represents a significant **increase** relative to the last quarter of 2015 (when the average was 0.7 %) .	2_bebceee16_es_418	Aunque la inflación todavía permanece en niveles reducidos , esta evolución representa un incremento significativo con respecto al último trimestre de 2015 (en el que , en promedio , se situó en el 0,7 %) .
4	4_bebceee16_en_1315	As a result , employment stood 1.2 % above the level recorded one year earlier , which represents the highest annual **increase** since the second quarter of 2008 .	4_bebceee16_es_1277	En consecuencia , la ocupación se situó un 1,2 % por encima del nivel alcanzado un año antes , lo que representa el mayor incremento interanual desde el segundo trimestre de 2008 .
5	4_bebceee16_en_1321	Following an **increase** in capacity utilisation in the manufacturing sector and overall strong growth in capital goods production , investment growth is likely to have continued at the robust pace seen at the turn of the year .	4_bebceee16_es_1284	Tras el aumento de la utilización de la capacidad productiva en el sector manufacturero y del fuerte incremento que , en general , ha experimentado la producción de bienes de equipo , es probable que la inversión haya mantenido el sólido ritmo de crecimiento observado en torno al cambio de año .
6	4_bebceee16_en_1497	The March **increase** was mostly a product of special factors and resulted in a positive flow for the quarter as a whole .	4_bebceee16_es_1456	El incremento de marzo fue principalmente el producto de factores especiales y se tradujo en un flujo positivo para el conjunto del trimestre .
7	5_bebceee16_en_1687	Rising investment in metal products and machinery made up about half of the **increase** in year-on-year terms in the first quarter , while construction and ICT (information and communication technology) investment contributed equally to the remaining part .	5_bebceee16_es_1635	En el primer trimestre , la mayor inversión en productos metálicos y maquinaria representó en torno a la mitad de este incremento interanual , mientras que las inversiones en construcción y en TIC (tecnologías de la información y las comunicaciones) contribuyeron en la misma medida al aumento restante .

Figura 146 - Resultado de la consulta 3

(7) En todas las consultas (Figura 142, Figura 144 y Figura 146), el resultado final es una tabla que muestra los resultados de la misma, con el elemento buscado en negrita, junto a su contexto (KWIC), que por defecto en el caso de corpus paralelos es de una oración. El contexto mostrado en el subcorpus

sobre el que se realiza la búsqueda puede modificarse (Figura 147), no así el de los corpus paralelos, que por cuestiones relacionadas con CWB es siempre una oración.[59]

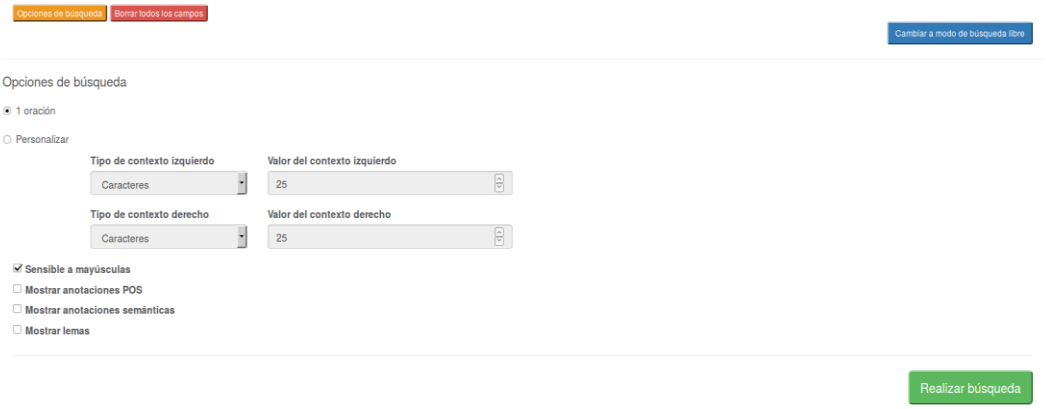
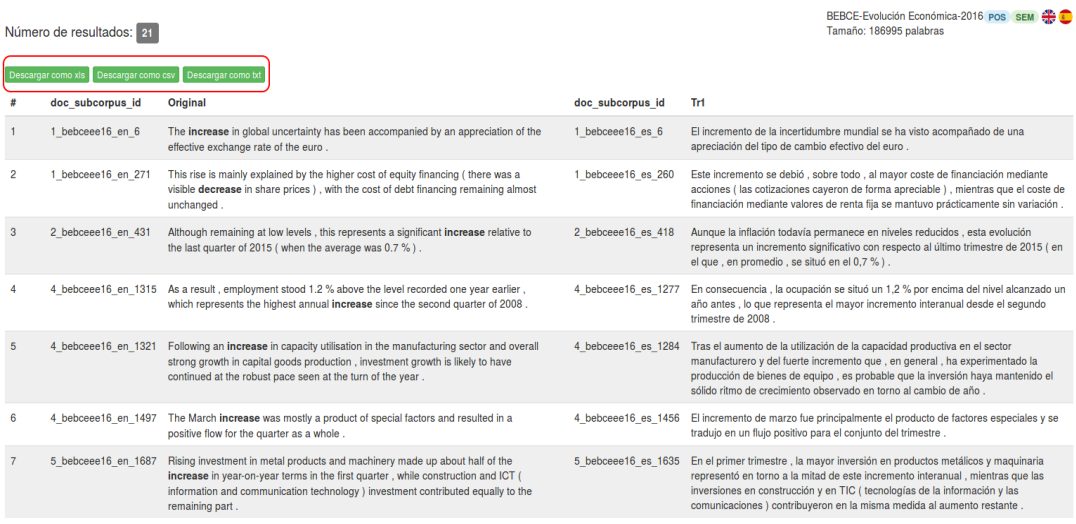


Figura 147 - Opciones de búsqueda en *ACTRES Corpus Manager*

(8) Los resultados de la consulta pueden descargarse en formato Excel, txt o csv (Figura 148).



#	doc_subcorpus_id	Original	doc_subcorpus_id	Tr1
1	1_bebceee16_en_6	The **increase** in global uncertainty has been accompanied by an appreciation of the effective exchange rate of the euro .	1_bebceee16_es_6	El incremento de la incertidumbre mundial se ha visto acompañado de una apreciación del tipo de cambio efectivo del euro .
2	1_bebceee16_en_271	This rise is mainly explained by the higher cost of equity financing (there was a visible **decrease** in share prices) , with the cost of debt financing remaining almost unchanged .	1_bebceee16_es_260	Este incremento se debió , sobre todo , al mayor coste de financiación mediante acciones (las cotizaciones cayeron de forma apreciable) , mientras que el coste de financiación mediante valores de renta fija se mantuvo prácticamente sin variación .
3	2_bebceee16_en_431	Although remaining at low levels , this represents a significant **increase** relative to the last quarter of 2015 (when the average was 0.7 %) .	2_bebceee16_es_418	Aunque la inflación todavía permanece en niveles reducidos , esta evolución representa un incremento significativo con respecto al último trimestre de 2015 (en el que , en promedio , se situó en el 0,7 %) .
4	4_bebceee16_en_1315	As a result , employment stood 1.2 % above the level recorded one year earlier , which represents the highest annual **increase** since the second quarter of 2008 .	4_bebceee16_es_1277	En consecuencia , la ocupación se situó un 1,2 % por encima del nivel alcanzado un año antes , lo que representa el mayor incremento interanual desde el segundo trimestre de 2008 .
5	4_bebceee16_en_1321	Following an **increase** in capacity utilisation in the manufacturing sector and overall strong growth in capital goods production , investment growth is likely to have continued at the robust pace seen at the turn of the year .	4_bebceee16_es_1284	Tras el aumento de la utilización de la capacidad productiva en el sector manufacturero y del fuerte incremento que , en general , ha experimentado la producción de bienes de equipo , es probable que la inversión haya mantenido el sólido ritmo de crecimiento observado en torno al cambio de año .
6	4_bebceee16_en_1497	The March **increase** was mostly a product of special factors and resulted in a positive flow for the quarter as a whole .	4_bebceee16_es_1456	El incremento de marzo fue principalmente el producto de factores especiales y se tradujo en un flujo positivo para el conjunto del trimestre .
7	5_bebceee16_en_1687	Rising investment in metal products and machinery made up about half of the **increase** in year-on-year terms in the first quarter , while construction and ICT (information and communication technology) investment contributed equally to the remaining part .	5_bebceee16_es_1635	En el primer trimestre , la mayor inversión en productos metálicos y maquinaria representó en torno a la mitad de este incremento interanual , mientras que las inversiones en construcción y en TIC (tecnologías de la información y las comunicaciones) contribuyeron en la misma medida al aumento restante .

Figura 148 - Formatos de exportación de resultados

[59] Como consecuencia de que los corpus están alineados por oraciones, no se sabe con total seguridad la parte concreta de la oración traducida que contiene el elemento a buscar. Este es el motivo por el que se muestra toda la oración del corpus traducido.

7.1.3.3.2 Corpus monolingüe

El proceso es análogo a la búsqueda sobre corpus multilingües paralelos descrito con anterioridad con la salvedad de que:

(1) Sólo existe un formulario de búsqueda.

(2) La tabla de resultados destaca con mayor empeño la búsqueda realizada por el usuario al no tener que mostrar información relativa a los subcorpus alineados como en el caso anterior.

Por ejemplo, se selecciona el corpus monolingüe "*BE-Informe Anual (2008-2015)*" y se busca la palabra "*situación*" seguida de un adjetivo (Figura 149).

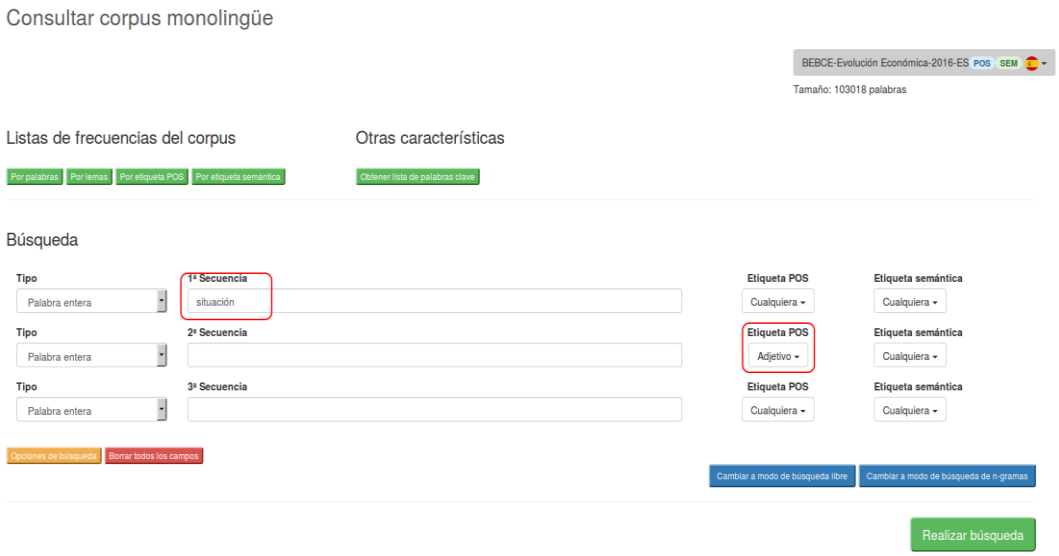


Figura 149 - Ejemplo de consulta sobre corpus monolingüe

Los resultados se muestran en la Figura 150.

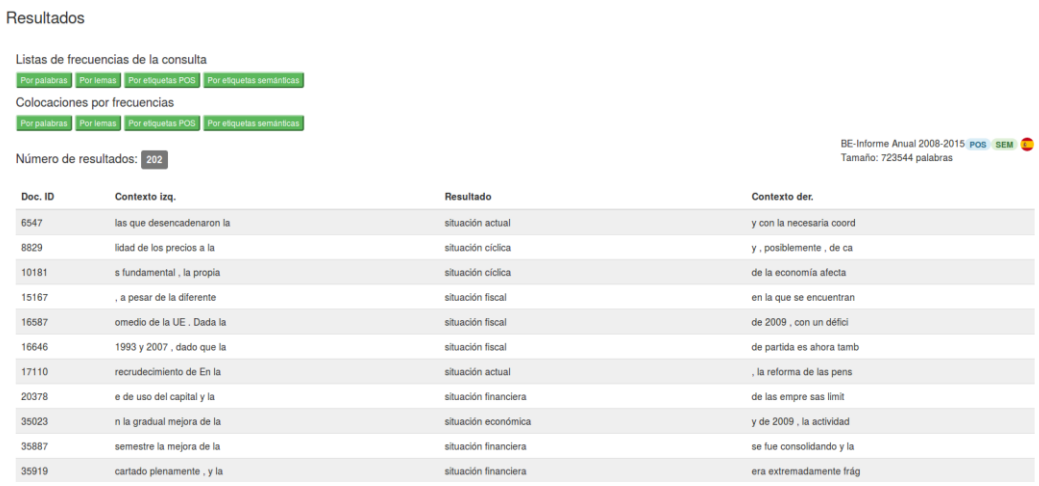


Figura 150 - Resultado de consulta sobre corpus monolingüe

7.1.3.3.3 Corpus comparable

En el caso de los corpus comparables, la opción de búsqueda por secuencias de texto sólo está disponible si los subcorpus tienen el mismo idioma (Figura 151).

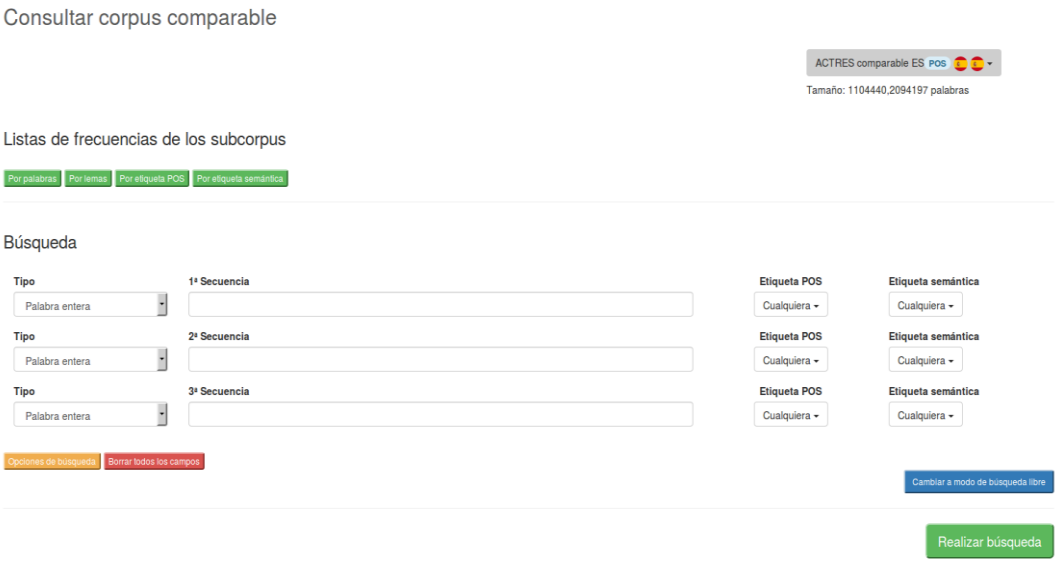


Figura 151 - Interfaz de búsqueda para corpus comparables en el mismo idioma

En caso contrario el interfaz es el presentado en la Figura 152.

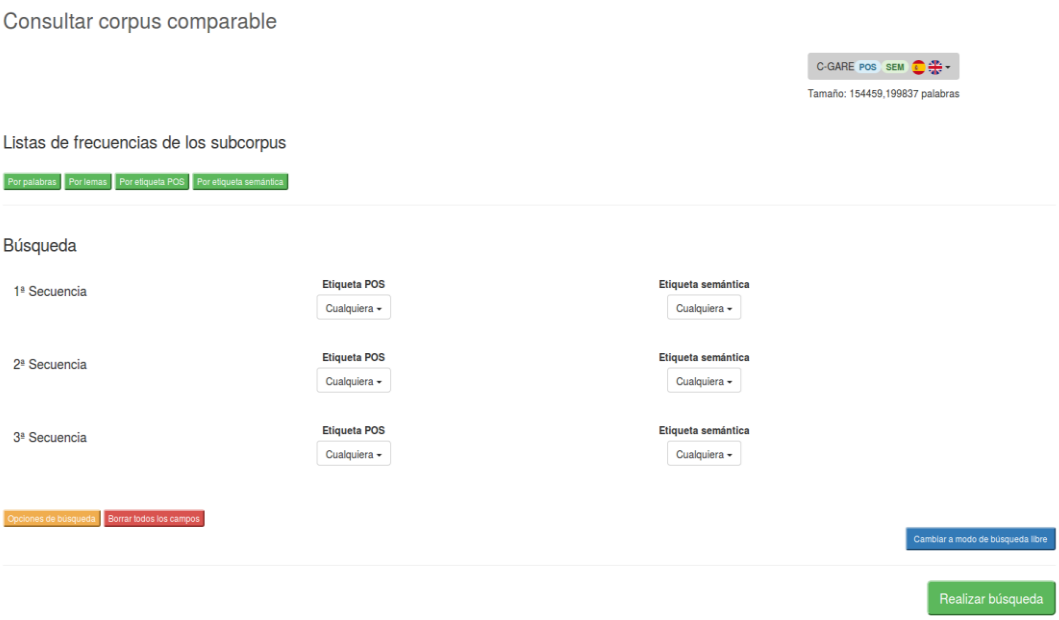


Figura 152 - Interfaz de búsqueda para corpus comparables en distinto idioma

Además, los resultados se muestran en diversas tablas, tantas como subcorpus compongan el corpus comparable.

Un ejemplo de búsqueda sobre corpus comparables en el mismo idioma, utilizando para ello el corpus comparable *"P-ACTRES Inglés Original - Inglés Traducido*" es la búsqueda de la palabra "*to*" (Figura 153).

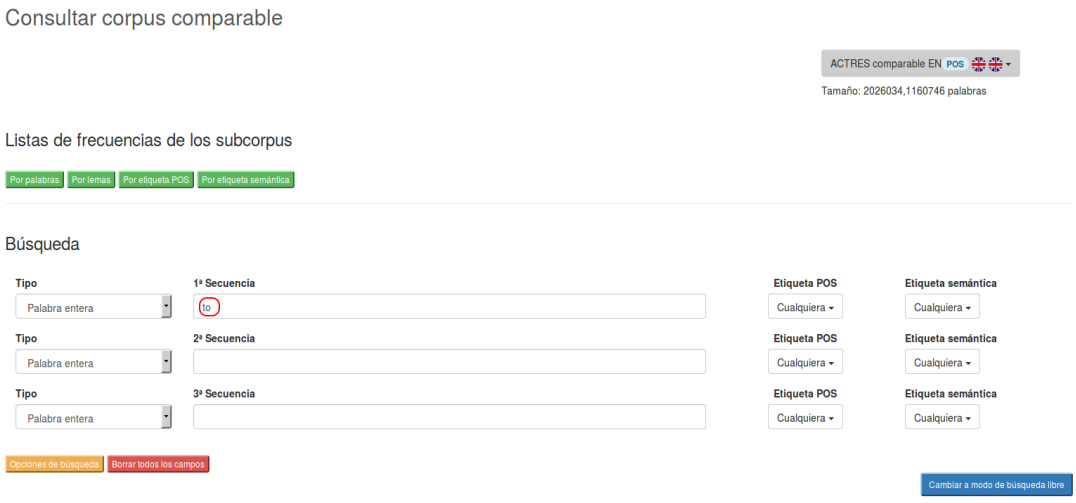


Figura 153 - Ejemplo de consulta sobre un corpus comparable en el mismo idioma

Cuyos resultados se muestran en general en la Figura 154, y en detalle para cada subcorpus en las figuras Figura 155 y Figura 156.

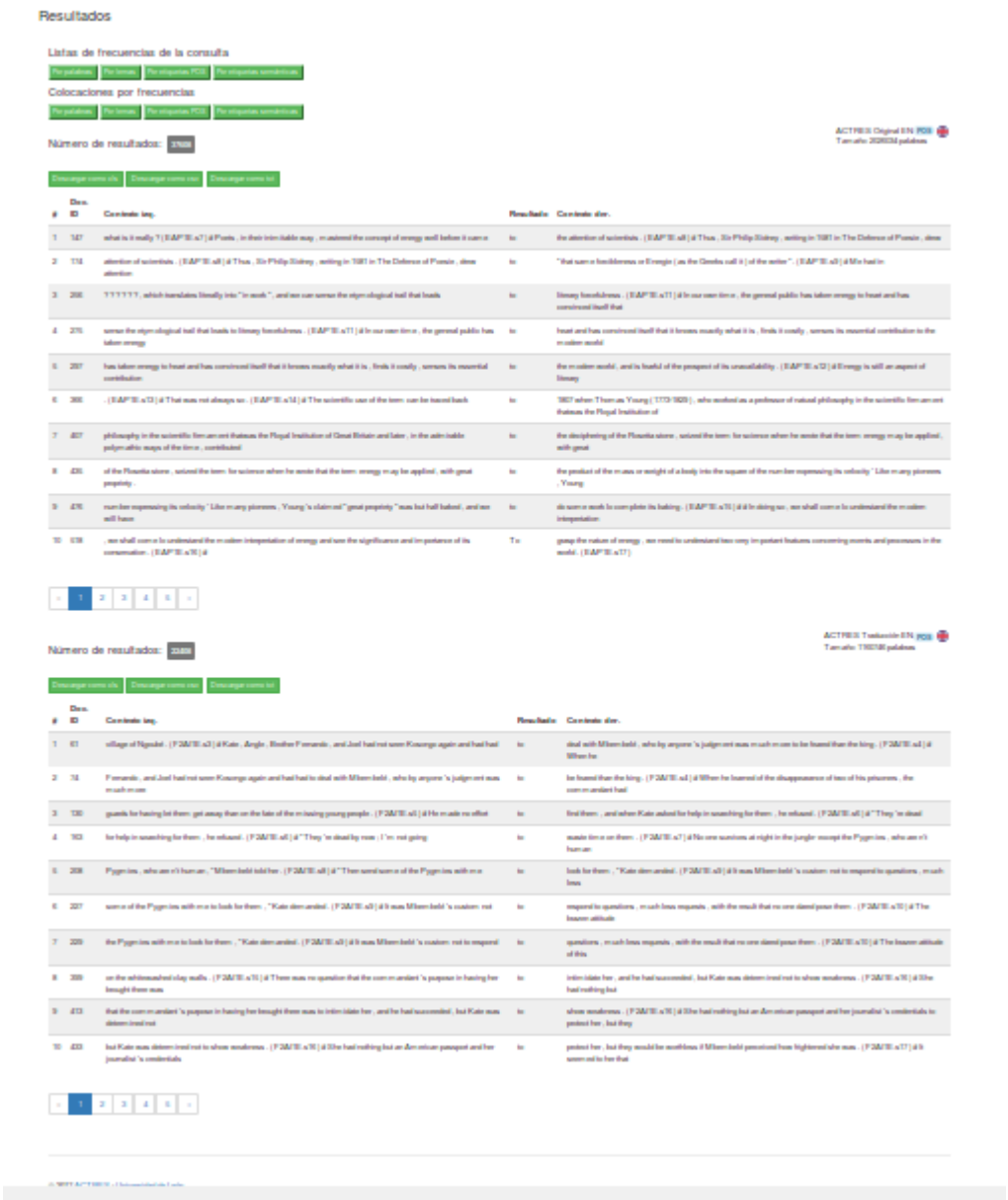
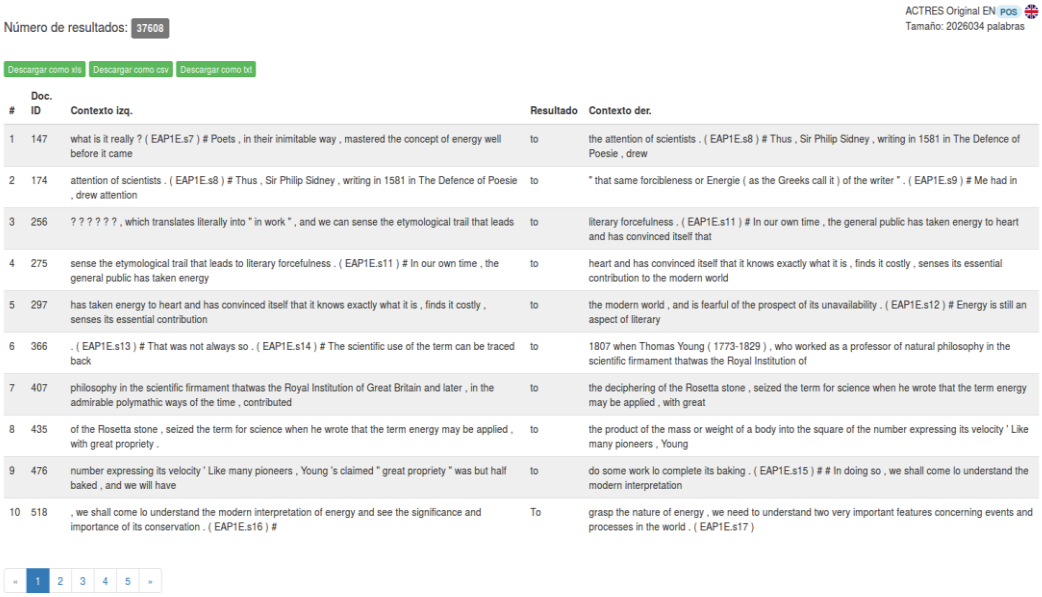


Figura 154 - Resultados de la consulta sobre el corpus comparable



Número de resultados: 37608

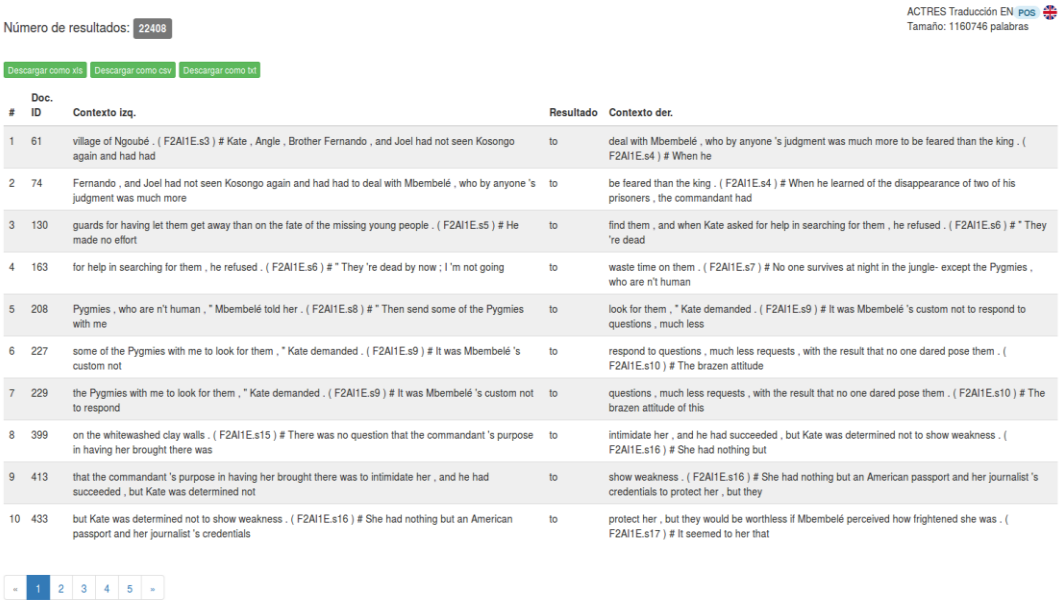
ACTRES Original EN POS
Tamaño: 2026034 palabras

Descargar como xls | Descargar como csv | Descargar como txt

#	Doc. ID	Contexto izq.	Resultado	Contexto der.
1	147	what is it really ? (EAP1E.s7) # Poets , in their inimitable way , mastered the concept of energy well before it came	to	the attention of scientists . (EAP1E.s8) # Thus , Sir Philip Sidney , writing in 1581 in The Defence of Poesie , drew
2	174	attention of scientists . (EAP1E.s8) # Thus , Sir Philip Sidney , writing in 1581 in The Defence of Poesie , drew attention	to	" that same forcibleness or Energie (as the Greeks call it) of the writer " . (EAP1E.s9) # Me had in
3	256	? ? ? ? ? ? , which translates literally into " in work " , and we can sense the etymological trail that leads	to	literary forcefulness . (EAP1E.s11) # In our own time , the general public has taken energy to heart and has convinced itself that
4	275	sense the etymological trail that leads to literary forcefulness . (EAP1E.s11) # In our own time , the general public has taken energy	to	heart and has convinced itself that it knows exactly what it is , finds it costly , senses its essential contribution to the modern world
5	297	has taken energy to heart and has convinced itself that it knows exactly what it is , finds it costly , senses its essential contribution	to	the modern world , and is fearful of the prospect of its unavailability . (EAP1E.s12) # Energy is still an aspect of literary
6	366	. (EAP1E.s13) # That was not always so . (EAP1E.s14) # The scientific use of the term can be traced back	to	1807 when Thomas Young (1773-1829) , who worked as a professor of natural philosophy in the scientific firmament thatwas the Royal Institution of
7	407	philosophy in the scientific firmament thatwas the Royal Institution of Great Britain and later , in the admirable polymathic ways of the time , contributed	to	the deciphering of the Rosetta stone , seized the term for science when he wrote that the term energy may be applied , with great
8	435	of the Rosetta stone , seized the term for science when he wrote that the term energy may be applied , with great propriety .	to	the product of the mass or weight of a body into the square of the number expressing its velocity ' Like many pioneers , Young
9	476	number expressing its velocity ' Like many pioneers , Young 's claimed " great propriety " was but half baked , and we will have	to	do some work lo complete its baking . (EAP1E.s15) # # In doing so , we shall come lo understand the modern interpretation
10	518	, we shall come lo understand the modern interpretation of energy and see the significance and importance of its conservation . (EAP1E.s16) #	To	grasp the nature of energy , we need to understand two very important features concerning events and processes in the world . (EAP1E.s17)

« 1 2 3 4 5 »

Figura 155 - Resultados de la consulta sobre el subcorpus comparable 1



Número de resultados: 22408

ACTRES Traducción EN POS
Tamaño: 1160746 palabras

Descargar como xls | Descargar como csv | Descargar como txt

#	Doc. ID	Contexto izq.	Resultado	Contexto der.
1	61	village of Ngoubé . (F2AI1E.s3) # Kate , Angle , Brother Fernando , and Joel had not seen Kosongo again and had had	to	deal with Mbembelé , who by anyone 's judgment was much more to be feared than the king . (F2AI1E.s4) # When he
2	74	Fernando , and Joel had not seen Kosongo again and had had to deal with Mbembelé , who by anyone 's judgment was much more	to	be feared than the king . (F2AI1E.s4) # When he learned of the disappearance of two of his prisoners , the commandant had
3	130	guards for having let them get away than on the fate of the missing young people . (F2AI1E.s5) # He made no effort	to	find them , and when Kate asked for help in searching for them , he refused . (F2AI1E.s6) # " They 're dead
4	163	for help in searching for them , he refused . (F2AI1E.s6) # " They 're dead by now ; I 'm not going	to	waste time on them . (F2AI1E.s7) # No one survives at night in the jungle- except the Pygmies , who are n't human
5	208	Pygmies , who are n't human , " Mbembelé told her . (F2AI1E.s8) # " Then send some of the Pygmies with me	to	look for them , " Kate demanded . (F2AI1E.s9) # It was Mbembelé 's custom not to respond to questions , much less
6	227	some of the Pygmies with me to look for them , " Kate demanded . (F2AI1E.s9) # It was Mbembelé 's custom not	to	respond to questions , much less requests , with the result that no one dared pose them . (F2AI1E.s10) # The brazen attitude
7	229	the Pygmies with me to look for them , " Kate demanded . (F2AI1E.s9) # It was Mbembelé 's custom not to respond	to	questions , much less requests , with the result that no one dared pose them . (F2AI1E.s10) # The brazen attitude of this
8	399	on the whitewashed clay walls . (F2AI1E.s15) # There was no question that the commandant 's purpose in having her brought there was	to	intimidate her , and he had succeeded , but Kate was determined not to show weakness . (F2AI1E.s16) # She had nothing but
9	413	that the commandant 's purpose in having her brought there was to intimidate her , and he had succeeded , but Kate was determined not	to	show weakness . (F2AI1E.s16) # She had nothing but an American passport and her journalist 's credentials to protect her , but they
10	433	but Kate was determined not to show weakness . (F2AI1E.s16) # She had nothing but an American passport and her journalist 's credentials	to	protect her , but they would be worthless if Mbembelé perceived how frightened she was . (F2AI1E.s17) # It seemed to her that

« 1 2 3 4 5 »

Figura 156 - Resultados de la consulta sobre el subcorpus comparable 2

7.1.3.3.4 Corpus monolingüe etiquetado retóricamente

Por parte de los corpus monolingües etiquetados retóricamente, se añade un selector para permitir realizar búsquedas sobre secciones concretas del corpus, y otro para determinar la categoría asignada a cada palabra durante el etiquetado retórico.

Por ejemplo, en las siguientes figuras se muestra la consulta para buscar las palabras con la etiqueta retórica "*Objeto*" que aparezcan en la sección retórica etiquetada como "*Acciones*" (Figura 157).

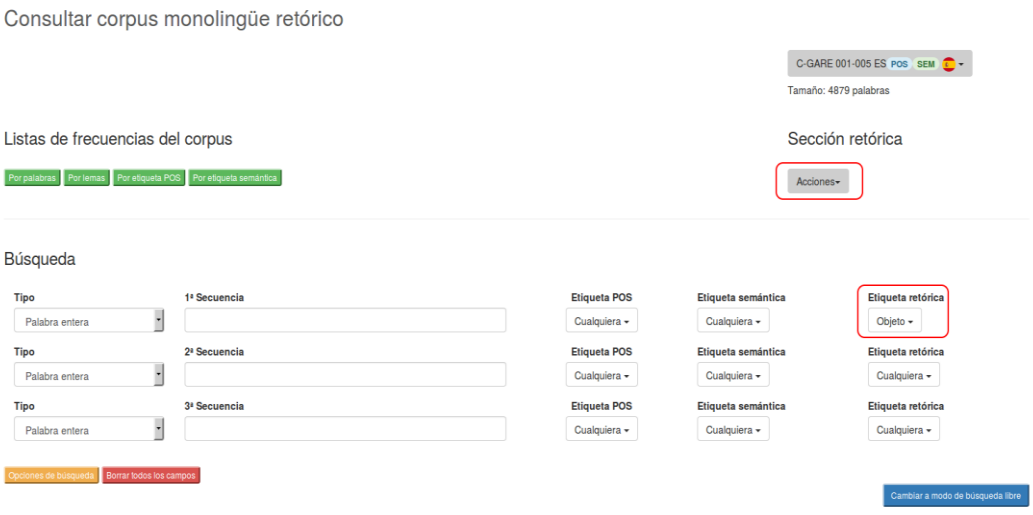


Figura 157 - Ejemplo de consulta sobre corpus monolingüe etiquetado retóricamente

El resultado de esta consulta se muestra a continuación (Figura 158):

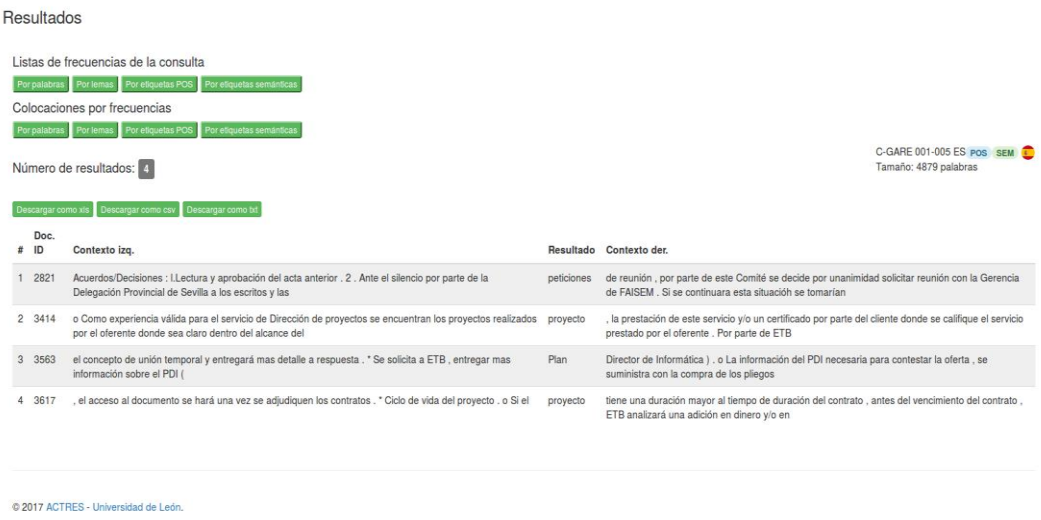


Figura 158 - Resultados de la consulta sobre el corpus monolingüe etiquetado retóricamente

7.1.3.4 Extracción de estadísticas de un corpus lingüístico

Las estadísticas están orientadas al corpus monolingüe, es decir, en el caso de corpus paralelos o comparables, las estadísticas del corpus son consecuencia de la extracción individual de las estadísticas de los subcorpus que los componen.[60]

La Figura 159 muestra un diálogo para seleccionar el subcorpus sobre el que se desea extraer la lista de frecuencias en un corpus paralelo.

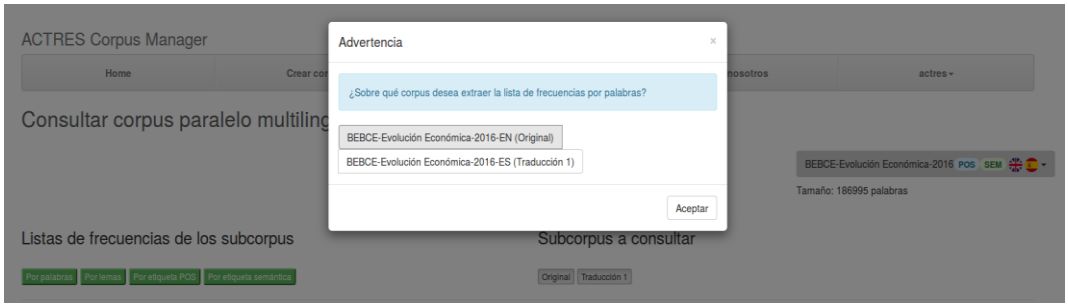


Figura 159 - Selección de subcorpus sobre el que extraer estadísticas en corpus paralelos

[60] Para más información sobre esta decisión consultar la página 177 perteneciente al apartado 6.3.3 Limitaciones de la implementación.

Dos de las estadísticas soportadas, búsqueda de n-gramas y extracción de palabras clave, han sido exclusivamente implementadas para los corpus monolingües.[61]

A continuación, se muestra un ejemplo de extracción de estadísticas de un corpus lingüístico, se utilizará para ello el corpus monolingüe "*BE-Informe Anual (2008-2015)*".

(1) La primera estadística que se puede extraer a través de *ACTRES Corpus Manager* es la lista de frecuencias del corpus, ya sea por palabras, y si se encuentra etiquetado, también por lemas, etiquetas gramaticales y etiquetas semánticas (Figura 160). En este ejemplo, se seleccionará la lista de frecuencias del corpus por palabras.

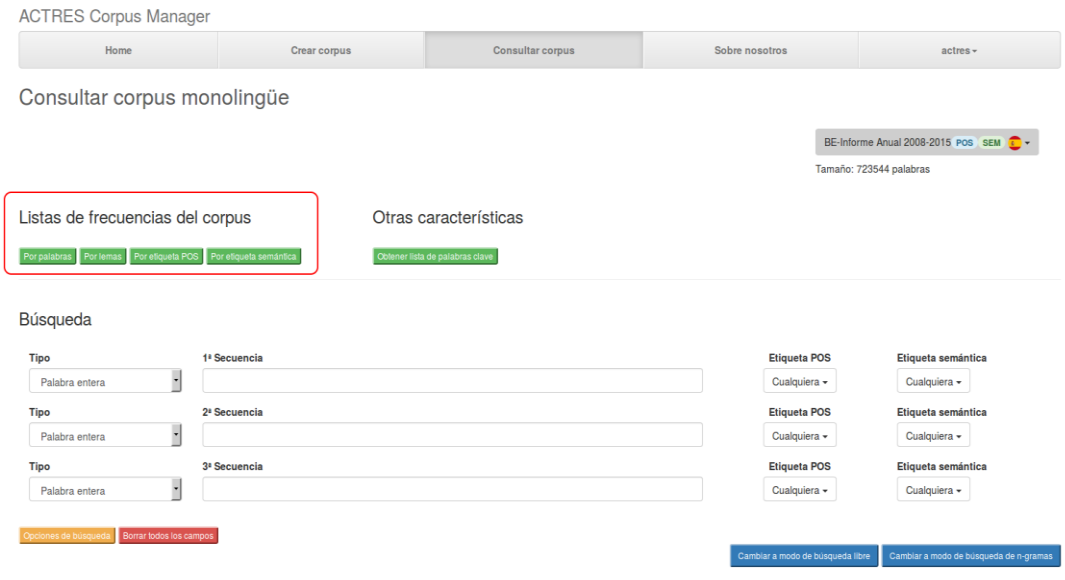


Figura 160 - Ubicación de la funcionalidad relacionada con la extracción de las listas de frecuencias del corpus

ACTRES Corpus Manager mostrará un mensaje para informar al usuario de la estadística se está realizando (Figura 161).

[61] Para más información sobre esta decisión consultar la página 177 perteneciente al apartado 6.3.3 Limitaciones de la implementación.

Figura 161 - Pantalla de carga de la extracción de las listas de frecuencias del corpus

Cuando la estadística se haya completado se mostrará un aviso (Figura 162) y se descargará automáticamente un documento Excel (Figura 163).

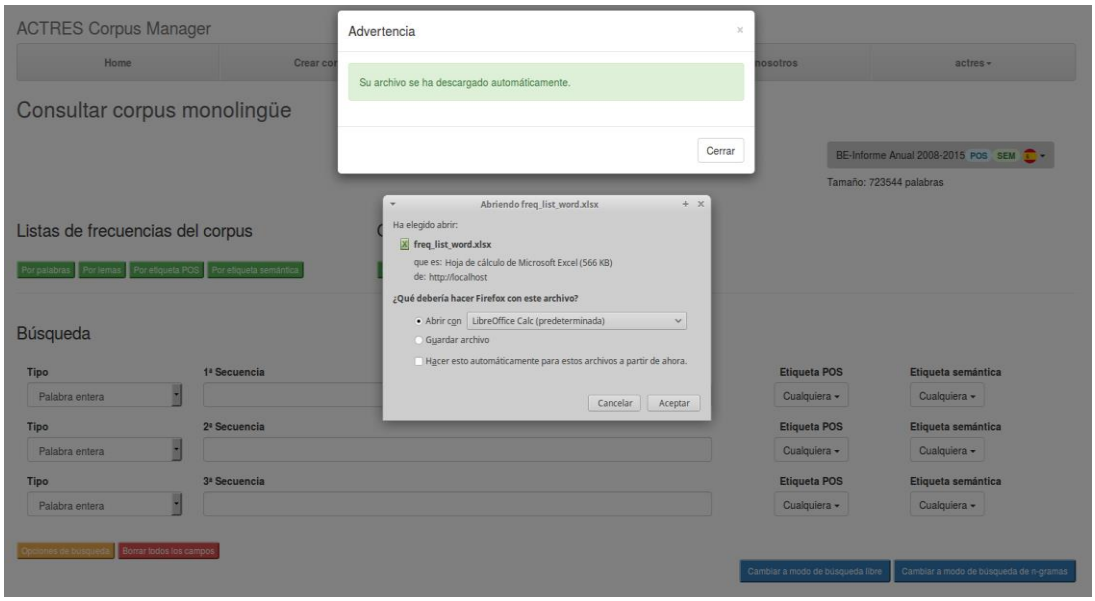
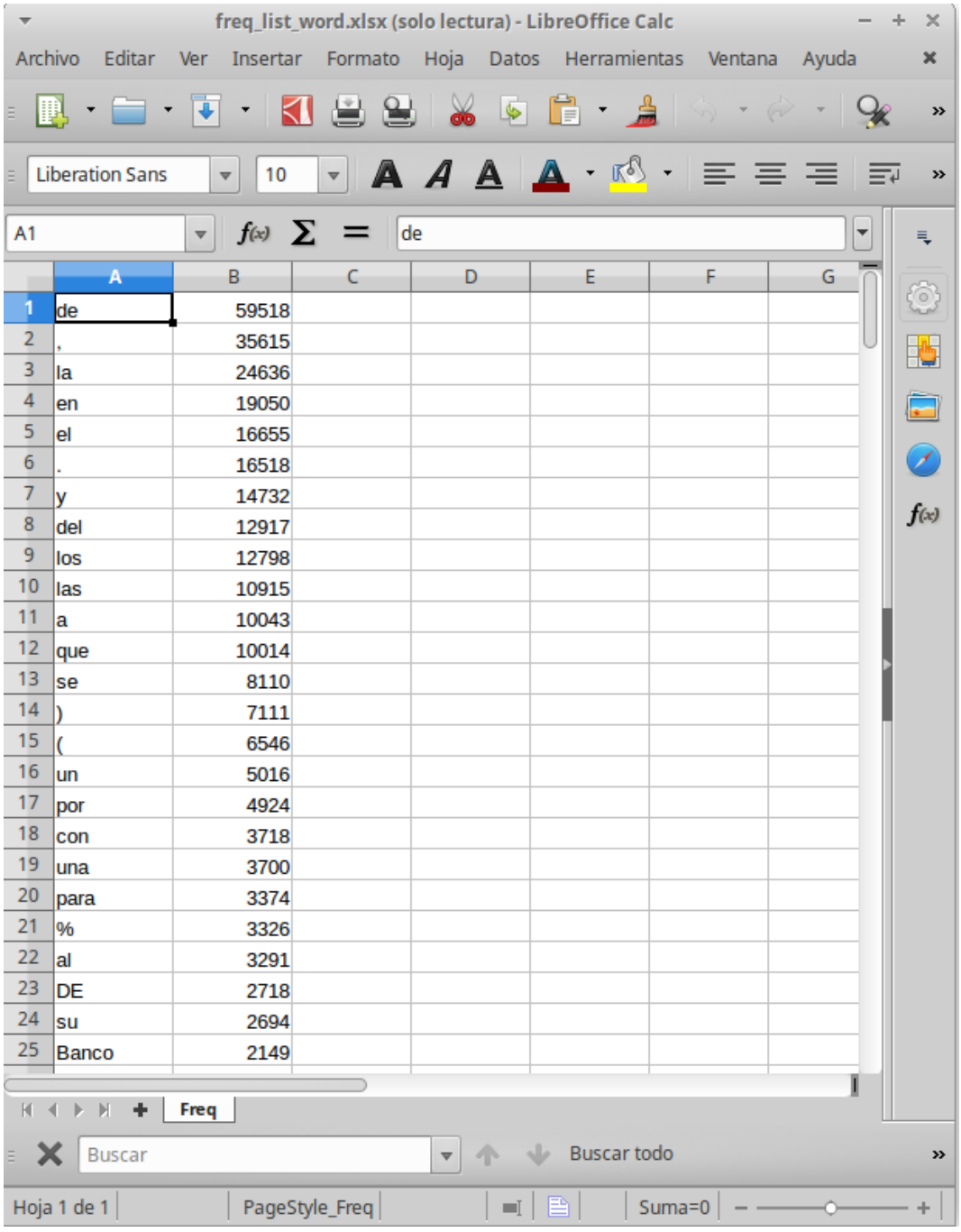


Figura 162 - Lista de frecuencias del corpus lista para descargar



	A	B
1	de	59518
2	,	35615
3	la	24636
4	en	19050
5	el	16655
6	.	16518
7	y	14732
8	del	12917
9	los	12798
10	las	10915
11	a	10043
12	que	10014
13	se	8110
14	)	7111
15	(	6546
16	un	5016
17	por	4924
18	con	3718
19	una	3700
20	para	3374
21	%	3326
22	al	3291
23	DE	2718
24	su	2694
25	Banco	2149

Figura 163 - Fragmento de la lista de frecuencias de palabras del corpus BE-Informe Anual (2008-2015)

(2) Para obtener la lista de palabras clave del corpus basta con presionar el botón correspondiente y seleccionar el corpus con el que se desea comparar

(Figura 164). Por ejemplo, en este caso seleccionaremos el corpus "EUROPARL ES", en español.

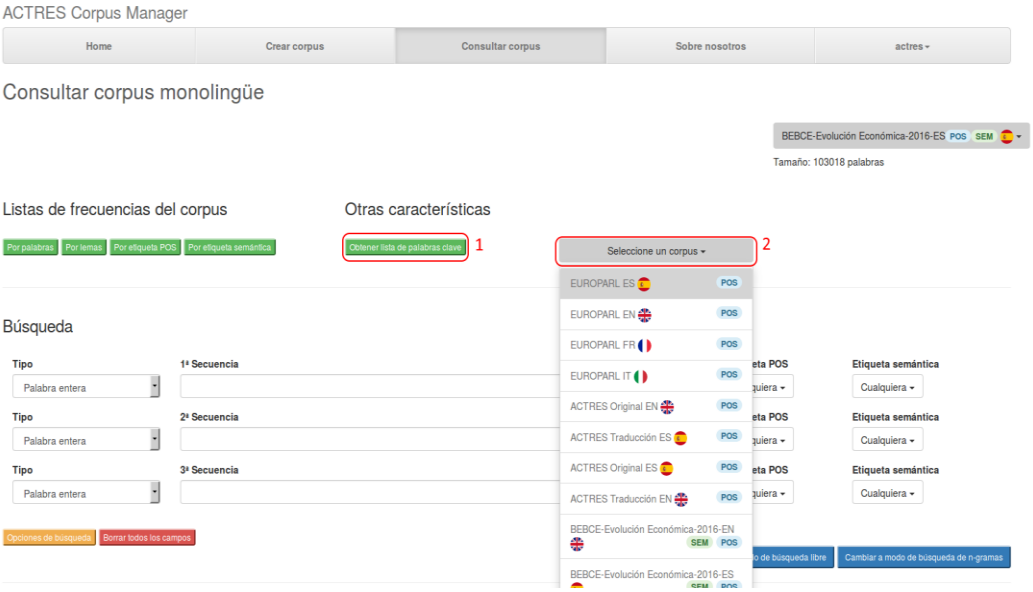


Figura 164 - Selección de corpus sobre el que realizar la comparación para extraer la lista de palabras clave

Una vez que la estadística se haya completado, se descargará un archivo Excel (Figura 165).

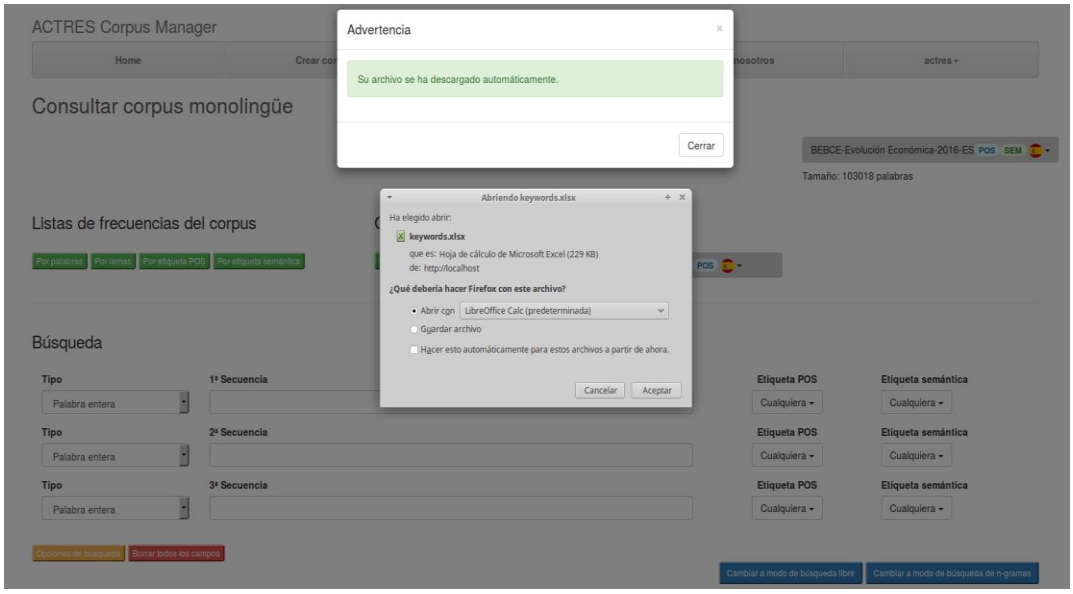


Figura 165 - Lista de palabras clave lista para descargar

El resultado final, de acuerdo con la funcionalidad de extracción de palabras clave de polmineR, utilizando el test de significación *chi-square*, se muestra en la Figura 166.

keywords.xlsx - LibreOffice Calc

Archivo Editar Ver Insertar Formato Hoja Datos Herramientas Ventana Ayuda

Liberation Sans 10

A1

	A	B	C	D	E	F	G	H
1		word	count_coi	count_ref	exp_coi	chisquare	rank_chisquare	
2	1	pp	516	-515	0,017882128	15159556,21	1	
3	2	LAS	270	-269	0,017882128	4150373,541	2	
4	3	LA	845	-832	0,23246767	3125707,9	3	
5	4	2014	854	-833	0,375524698	1975749,24	4	
6	5	LOS	277	-273	0,071528514	1091672,686	5	
7	6	DEL	736	-703	0,59011024	933173,7669	6	
8	7	gráfico	549	-520	0,518581726	590667,8413	7	
9	8	2009	1375	-1183	3,433368667	557894,1774	8	
10	9	EMPLEO	99	-98	0,017882128	557866,9974	9	
11	10	IIC	97	-96	0,017882128	535550,6045	10	
12	11	TOTAL	230	-224	0,107292771	501552,4757	11	
13	12	2012	1072	-939	2,378323087	489809,6683	12	
14	13	0,0	131	-129	0,035764257	488306,6367	13	
15	14	2011	547	-512	0,625874497	485657,1147	14	
16	15	véase	703	-644	1,05504558	475523,419	15	
17	16	-4	126	-124	0,035764257	451732,8912	16	
18	17	DE	2718	-1794	16,52308671	449735,9895	17	
19	18	CON	88	-87	0,017882128	440763,8926	18	
20	19	DEG	88	-87	0,017882128	440763,8926	19	
21	20	pr	87	-86	0,017882128	430801,4354	20	
22	21	2008	1474	-1177	5,310992157	413545,7889	21	
23	22	Activos	120	-118	0,035764257	409723,4154	22	
24	23	PARA	73	-72	0,017882128	303284,4265	23	
25	24	interanual	158	-153	0,089410642	283968,2163	24	
26	25	Tipo	70	-69	0,017882128	278863,2913	25	
27	26	AL	94	-92	0,035764257	251369,3793	26	
28	27	4.2	112	-109	0,053646385	237856,9561	27	
29	28	interanuales	60	-59	0,017882128	204861,7026	28	
30	29	1.1	59	-58	0,017882128	198087,8834	29	
31	30	2.1	97	-94	0,053646385	178385,2373	30	
32	31	Encuesta	54	-53	0,017882128	165926,986	31	
33	32	6.1	53	-52	0,017882128	159836,4463	32	

Keywords

Figura 166 - Fragmento de la lista de palabras clave del corpus BE-Informe Anual (2008-2015) en comparación con Europarl Es

(3) El resto de estadísticas de *ACTRES Corpus Manager* están relacionadas con la realización de una búsqueda. Por ejemplo, la consulta de la palabra "*otra*" en el mismo corpus (Figura 167).

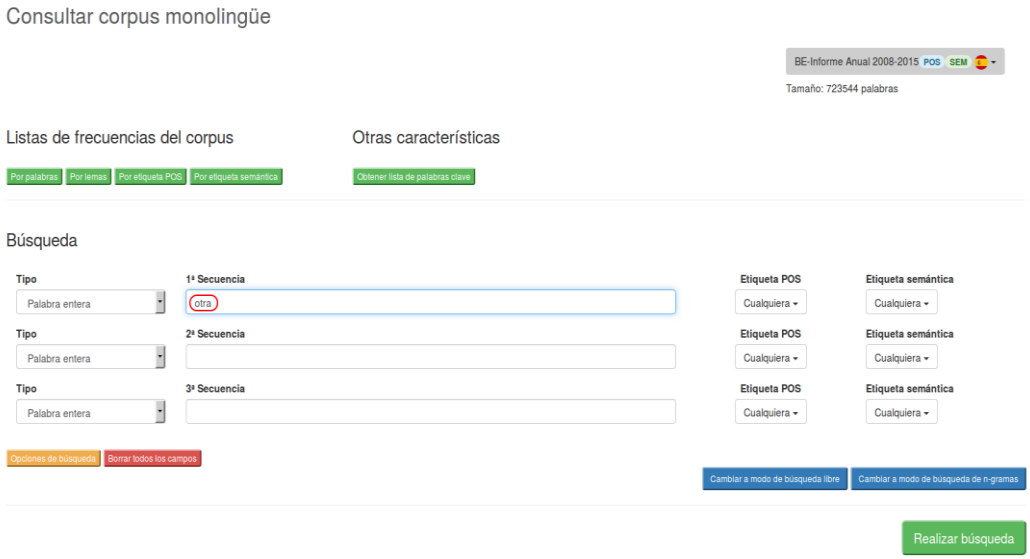


Figura 167 - Ejemplo de consulta para la extracción de estadísticas

(4) Una vez realizada la consulta (ya sea a través del modo de búsqueda ACTRES o del libre), y junto a los resultados de la misma, se pueden obtener las distintas estadísticas disponibles (Figura 168):

Figura 168 - Resultado de la consulta y ubicación de las funcionalidades estadísticas relacionadas

- **Lista de frecuencias de las consultas**: por palabras, o dependiendo del tipo de etiquetado disponible en el corpus, por lema, etiqueta gramatical, etiqueta semántica o etiqueta retórica.

  Al igual que sucede con el resto de estadísticas, se avisará de que la estadística ha sido realizada y el documento se descargará automáticamente (Figura 169).

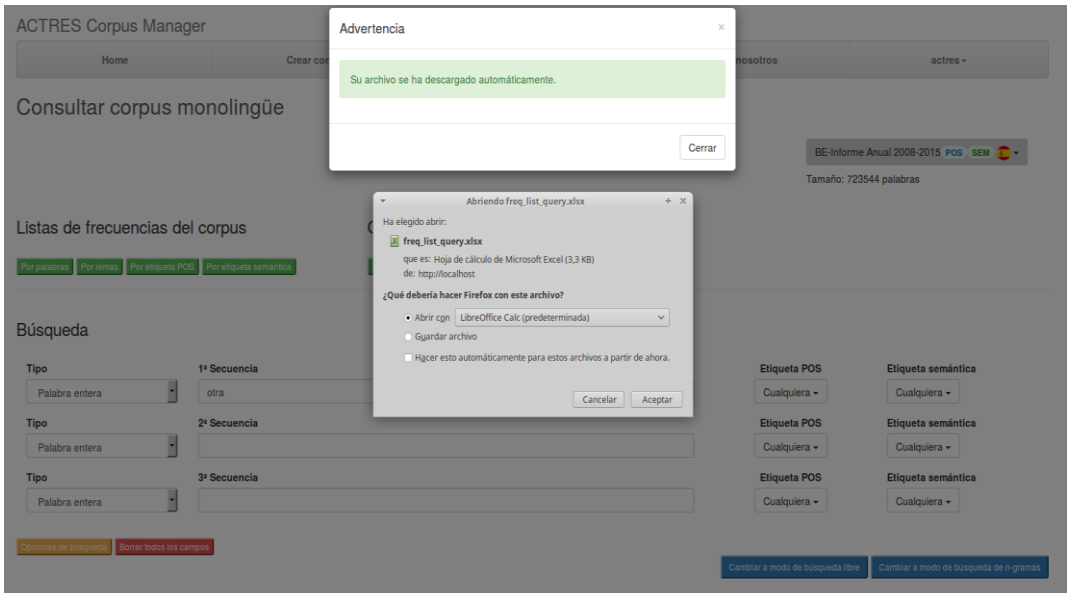


Figura 169 - Lista de frecuencias de la consulta lista para descargar

El resultado se mostrará en formato Excel. Por ejemplo, en la Figura 170 se muestra la lista de frecuencias de la consulta anterior de acuerdo con su etiqueta gramatical.

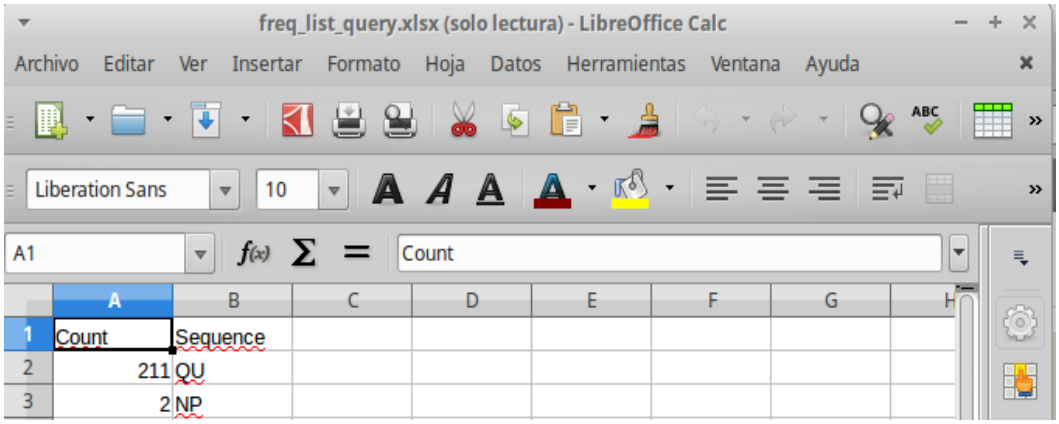
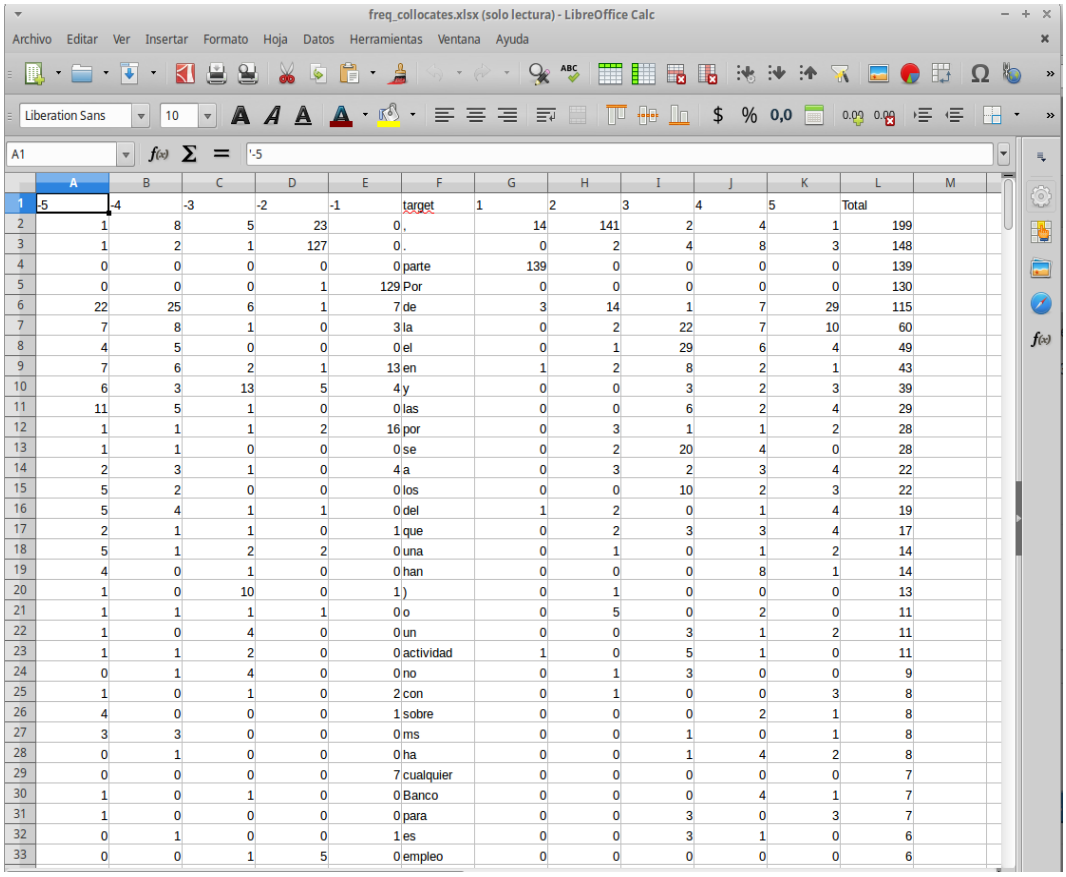


Figura 170 - Lista de frecuencias por etiqueta gramatical de la consulta realizada

- **Lista de colocaciones por frecuencia**: por palabras, o dependiendo del tipo de etiquetado realizado en la creación del corpus, por lema, etiqueta gramatical y etiqueta semántica. El resultado también se mostrará en formato Excel. La lista de colocaciones por frecuencias de la consulta anterior se muestra en la Figura 171.



freq_collocates.xlsx (solo lectura) - LibreOffice Calc

Archivo Editar Ver Insertar Formato Hoja Datos Herramientas Ventana Ayuda

Liberation Sans 10

A1 '-5

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	-5	-4	-3	-2	-1	target	1	2	3	4	5	Total	
2	1	8	5	23	0	,	14	141	2	4	1	199	
3	1	2	1	127	0	.	0	2	4	8	3	148	
4	0	0	0	0	0	parte	139	0	0	0	0	139	
5	0	0	0	1	129	Por	0	0	0	0	0	130	
6	22	25	6	1	7	de	3	14	1	7	29	115	
7	7	8	1	0	3	la	0	2	22	7	10	60	
8	4	5	0	0	0	el	0	1	29	6	4	49	
9	7	6	2	1	13	en	1	2	8	2	1	43	
10	6	3	13	5	4	y	0	0	3	2	3	39	
11	11	5	1	0	0	las	0	0	6	2	4	29	
12	1	1	1	2	16	por	0	3	1	1	2	28	
13	1	1	0	0	0	se	0	2	20	4	0	28	
14	2	3	1	0	4	a	0	3	2	3	4	22	
15	5	2	0	0	0	los	0	0	10	2	3	22	
16	5	4	1	1	0	del	1	2	0	1	4	19	
17	2	1	1	0	1	que	0	2	3	3	4	17	
18	5	1	2	2	0	una	0	1	0	1	2	14	
19	4	0	1	0	0	han	0	0	0	8	1	14	
20	1	0	10	0	1	)	0	1	0	0	0	13	
21	1	1	1	1	0	o	0	5	0	2	0	11	
22	1	0	4	0	0	un	0	0	3	1	2	11	
23	1	1	2	0	0	actividad	1	0	5	1	0	11	
24	0	1	4	0	0	no	0	1	3	0	0	9	
25	1	0	1	0	2	con	0	1	0	0	3	8	
26	4	0	0	0	1	sobre	0	0	0	2	1	8	
27	3	3	0	0	0	ms	0	0	1	0	1	8	
28	0	1	0	0	0	ha	0	0	1	4	2	8	
29	0	0	0	0	7	cualquier	0	0	0	0	0	7	
30	1	0	1	0	0	Banco	0	0	0	4	1	7	
31	1	0	0	0	0	para	0	0	3	0	3	7	
32	0	1	0	0	1	es	0	0	3	1	0	6	
33	0	0	1	5	0	empleo	0	0	0	0	0	6	

Figura 171 - Lista de colocaciones por frecuencias basadas en palabras de la consulta realizada

(5) Por último, describir el proceso de extracción de **n-gramas**. Para ello, es necesario presionar el botón "*Cambiar a modo de búsqueda de n-gramas*", el interfaz se modificará (Figura 172).

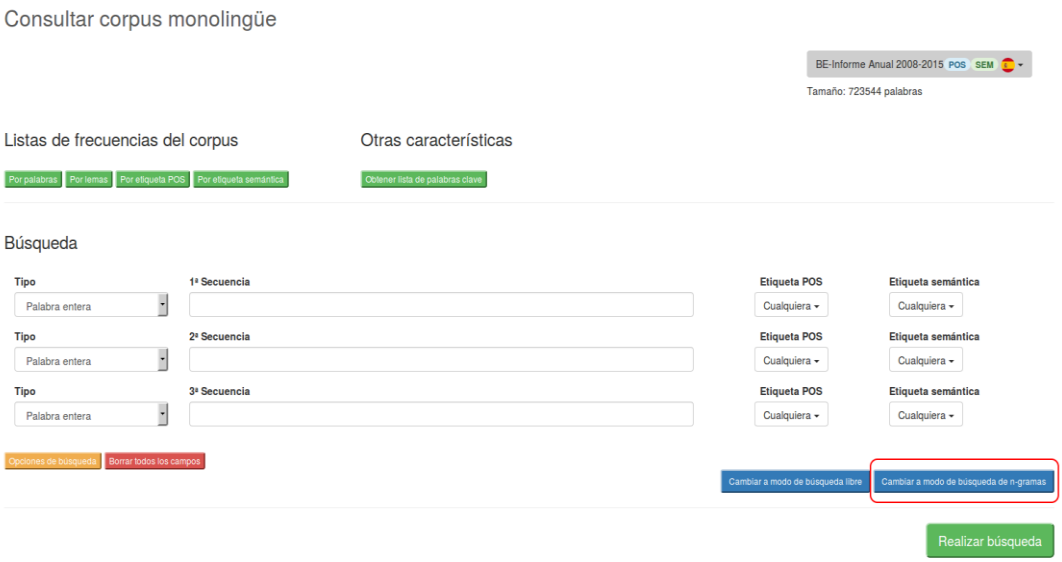


Figura 172 - Ubicación del botón "*Cambiar a modo de búsqueda de n-gramas*"

Es necesario establecer distintos parámetros del n-grama (Figura 173):

- **Valor de n**.
- **Elementos a mostrar**: palabras, o dependiendo del tipo de etiquetado realizado en la creación del corpus, por lema, etiqueta gramatical y etiqueta semántica.
- Y opcionalmente, se puede especificar el **contenido del n-grama** del mismo modo que se lleva a cabo en una consulta tradicional.

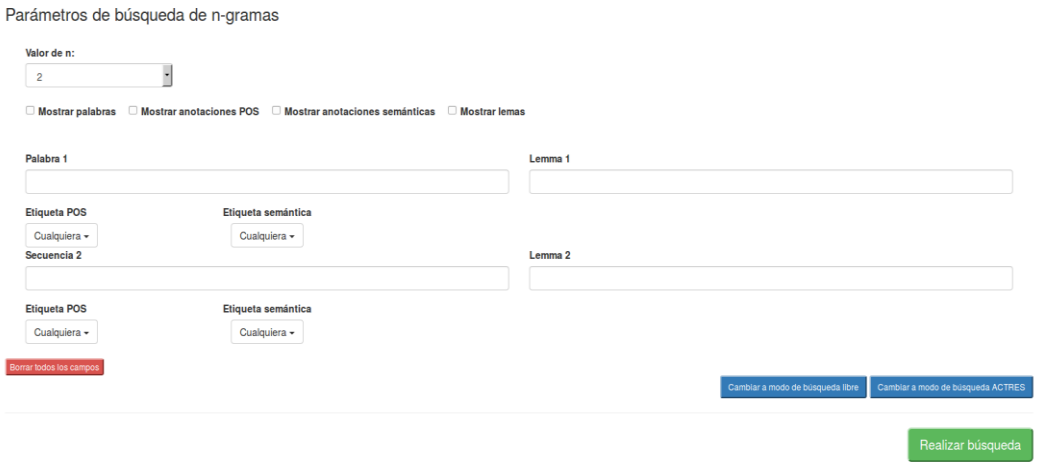


Figura 173 - Pantalla de búsqueda de n-gramas

Por ejemplo, se realizará una búsqueda de n-gramas con los siguientes parámetros (Figura 174):

- Valor de n: 2
- Mostrar anotación gramatical.
- Ningún elemento en la consulta.

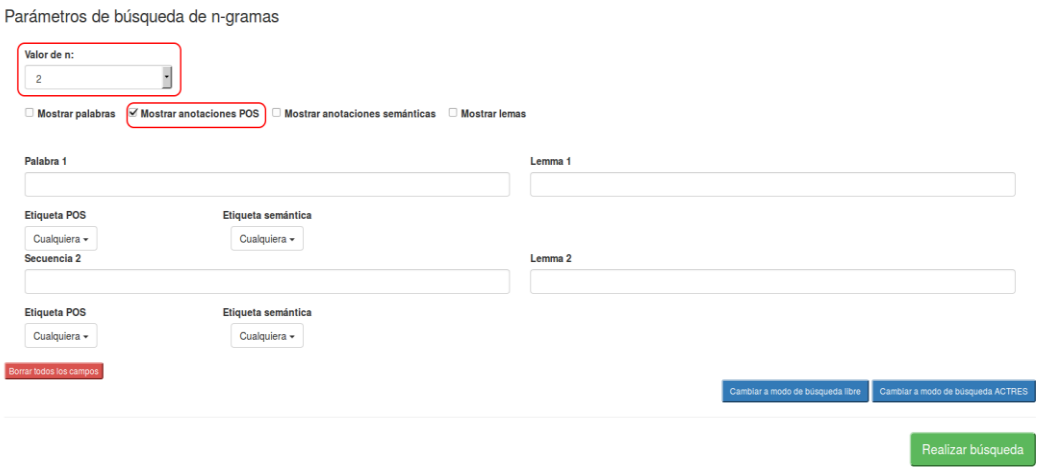


Figura 174 - Ejemplo de búsqueda de n-gramas

Una vez que se han extraído los n-gramas (Figura 175), aparecerá el mensaje para descargar el Excel.

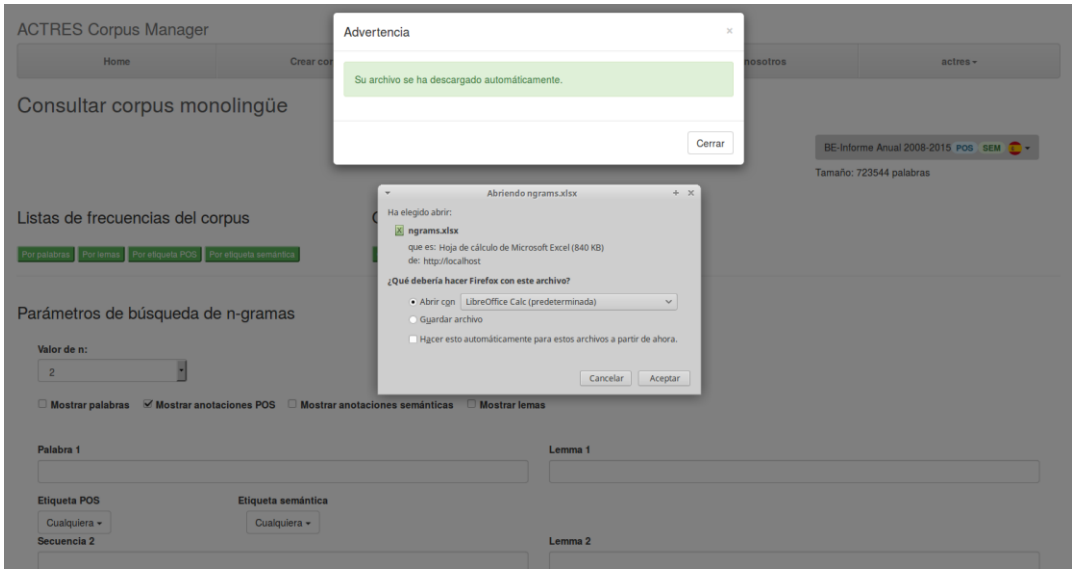
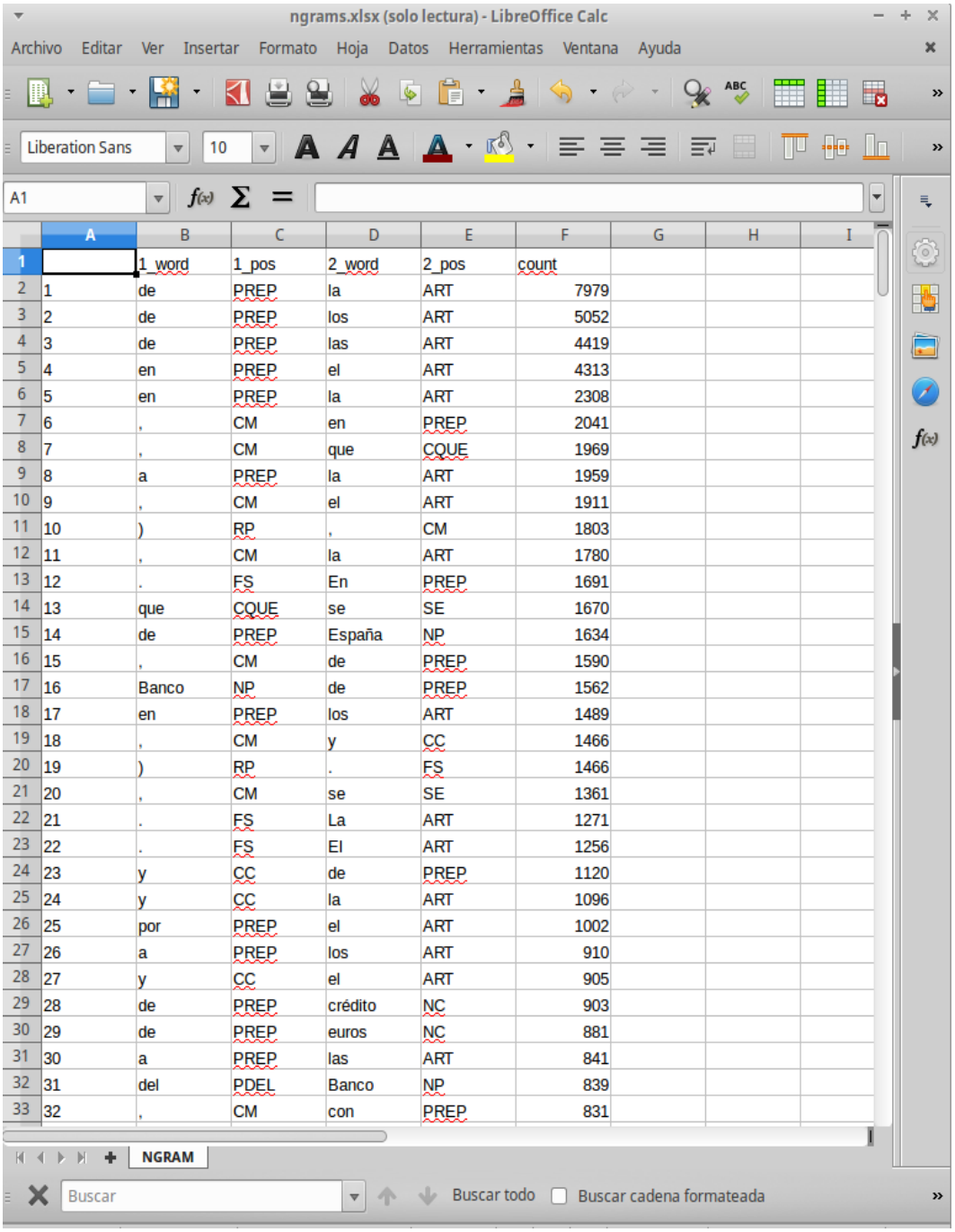


Figura 175 - n-gramas listos para su descarga

Que devuelve los siguientes resultados (Figura 176):



	A	B	C	D	E	F
1		1_word	1_pos	2_word	2_pos	count
2	1	de	PREP	la	ART	7979
3	2	de	PREP	los	ART	5052
4	3	de	PREP	las	ART	4419
5	4	en	PREP	el	ART	4313
6	5	en	PREP	la	ART	2308
7	6	,	CM	en	PREP	2041
8	7	,	CM	que	CQUE	1969
9	8	a	PREP	la	ART	1959
10	9	,	CM	el	ART	1911
11	10	)	RP	,	CM	1803
12	11	,	CM	la	ART	1780
13	12	.	FS	En	PREP	1691
14	13	que	CQUE	se	SE	1670
15	14	de	PREP	España	NP	1634
16	15	,	CM	de	PREP	1590
17	16	Banco	NP	de	PREP	1562
18	17	en	PREP	los	ART	1489
19	18	,	CM	y	CC	1466
20	19	)	RP	.	FS	1466
21	20	,	CM	se	SE	1361
22	21	.	FS	La	ART	1271
23	22	.	FS	El	ART	1256
24	23	y	CC	de	PREP	1120
25	24	y	CC	la	ART	1096
26	25	por	PREP	el	ART	1002
27	26	a	PREP	los	ART	910
28	27	y	CC	el	ART	905
29	28	de	PREP	crédito	NC	903
30	29	de	PREP	euros	NC	881
31	30	a	PREP	las	ART	841
32	31	del	PDEL	Banco	NP	839
33	32	,	CM	con	PREP	831

Figura 176 - Fragmento de los n-gramas obtenidos

7.2 Pruebas de usabilidad del framework

Siguiendo lo dispuesto en el Capítulo 5: "Metodología", la siguiente fase de la validación fue la realización de las pruebas de usabilidad por parte de usuarios potenciales de *ACTRES Corpus Manager*.

Aunque la validación del software está implícita en la ejecución correcta de las funcionalidades para las que ha sido diseñado, se ha creído conveniente complementar la validación mediante la medición del grado de usabilidad del software a través de un cuestionario SUS (Brooke, 1996).

La elección de este tipo de cuestionario de usabilidad se debe a que las pruebas se realizaron durante la fase final de desarrollo de la tesis doctoral, y como consecuencia del tiempo dispuesto y de los participantes (6), era necesario un cuestionario válido para muestras pequeñas. SUS cumplía este requisito de acuerdo con (Orfanou, Tselios, & Katsanos, 2015), (Bangor, Miller, & Kortum, 2008) o (U.S. Department of Health & Human Services, 2017). Por ejemplo, (Orfanou, Tselios, & Katsanos, 2015) defiende que SUS proporciona unos resultados fiables en muestras entre 6 y 14 participantes.[62]

Otro aspecto a señalar es que el hecho de haber realizado una toma de requisitos previa al diseño del software incrementa la posibilidad de obtener mejores resultados en este tipo de cuestionarios, ya que *ACTRES Corpus Manager* se ha implementado en parte utilizando las sugerencias de los usuarios potenciales del software.

Los seis participantes proceden de 4 universidades distintas:

- 3 de ellos son de la Universidad de León (ULe).
- 1 de la Universidad de Valladolid (UVa).
- 1 de la Universidad de Cantabria (UNICAN).

[62] Otras fuentes como (Tullis & Stetson, 2004) defienden un número mínimo de 12.

- 1 de la Universidad del País Vasco (UPV/EHU).

Los resultados reflejados en el cuestionario SUS dan una calificación media de 84.58, que de acuerdo con (Bangor, Kortum, & Miller, 2009) es considerado Excelente. Un breve análisis de los resultados individuales refleja que los investigadores con mayor experiencia en el empleo de software para el análisis de corpus han proporcionado a *ACTRES Corpus Manager* una mejor valoración. Como se ha mencionado anteriormente, este hecho puede explicarse en base a que los requisitos del framework se establecieron teniendo en cuenta, entre otras opiniones, los comentarios recabados entre usuarios habituales de este tipo de aplicaciones.

El cuestionario SUS utilizado se encuentra en el Anexo 3

Capítulo

8

Discusión

La principal innovación de *ACTRES Corpus Manager* reside en situar al usuario lingüista como la única referencia sobre la que desarrollar el software. Es decir, *ACTRES Corpus Manager* tiene las mismas funcionalidades independientemente de las habilidades técnicas del usuario que lo utilice. Se intenta alcanzar las mayores cotas de usabilidad posible, considerando al usuario lingüista como el único actor que va a emplear el software y por ello, automatizando todos los procesos críticos que hasta la fecha requerían de ciertas habilidades técnicas o el apoyo de personal técnico especializado.

La mayoría de frameworks para el tratamiento de corpus lingüísticos analizados centran sus esfuerzos en aumentar la usabilidad del software para la realización de consultas y extracción de estadísticas, olvidando el proceso de creación de los corpus. Por esta circunstancia, se limitan los tipos de corpus disponibles o se obliga a que el usuario emplee recursos externos, teniendo que adecuar el formato del corpus al requerido por el framework utilizado.

ACTRES Corpus Manager permite la creación y análisis de distintos tipos de corpus lingüísticos (monolingües, bi-/multilingües, comparables), con anotaciones lingüísticas (gramaticales, semánticas y retóricas), y sin que la disposición de conocimientos técnicos o la asistencia de un especialista técnico sea necesaria.

ACTRES Corpus Manager cumple su cometido de forma adecuada y efectiva, tal y como se demuestra en el Capítulo 7: ”Validación del framework”. En este capítulo se lleva a cabo una verificación empírica de las distintas funcionalidades del software a través de la ejecución de los casos de uso para los que fue diseñado. Además, se realiza un cuestionario de usabilidad tipo SUS por parte de usuarios potenciales de la aplicación, que con una calificación de 84.58, dejan constancia que la interacción entre el software y el usuario es eficaz, eficiente y satisfactoria.

Hay que señalar que *ACTRES Corpus Manager* es un prototipo, cuyo desarrollo ha estado sujeto a las limitaciones derivadas, principalmente, del carácter interdisciplinar de la tesis. También hay que mencionar las características de los recursos lingüísticos y computacionales existentes en el entorno de investigación. Como todo prototipo es, por ello, susceptible de mejora en el futuro en algunas de sus funcionalidades.

En primer lugar, la eficiencia computacional de *ACTRES Corpus Manager* no ha sido la prioridad absoluta. Si bien es cierto que crea y analiza corpus en intervalos de tiempo aceptables, estos dependen en gran medida del rendimiento de los recursos hardware y software más críticos: alineador y etiquetadores, más concretamente de cómo tratan grandes cantidades de información. Las pruebas software llevadas a cabo, realizadas sobre una infraestructura de testeo, únicamente han arrojado datos regulares respecto al tiempo de ejecución del etiquetador semántico y la extracción de n-gramas, que deben mejorarse.

ACTRES Corpus Manager permite la creación y consulta de corpus monolingües, multilingües paralelos y comparables en español, inglés, francés e italiano etiquetados en los niveles gramatical y semántico, y la creación de corpus monolingües etiquetados en los niveles gramatical, semántico y retórico en español e inglés. Estas decisiones son consecuencia de:

- **La verificación del multilingüismo**: Para constatar que *ACTRES Corpus Manager* era capaz de tratar con idiomas distintos al inglés se seleccionó un

número limitado de idiomas. La inclusión de los mismos requiere la adición de elementos software específicos, como por ejemplo tokenizers o alineadores, y no tenía sentido dedicar un esfuerzo superlativo en conseguir el mayor soporte multilingüe para la ejecución de un prototipo.

- **Recursos lingüísticos disponibles**: Los componentes software implementados son alimentados por recursos lingüísticos que no se encuentran en todos los idiomas. Por ejemplo, el recurso lingüístico utilizado para el etiquetado semántico (ver VIII) Etiquetador semántico), USAS (Piao, Bianchi, Dayrell, D'Egidio, & Rayson, 2015), carece de lexicón para el alemán (UCREL, 2016) .

ACTRES Corpus Manager permite la alineación de corpus paralelos entre el subcorpus original y las respectivas traducciones. Es decir, los subcorpus traducidos no están alineados entre sí y nada asegura que en una consulta paralela entre dos o más subcorpus traducidos, las alineaciones mostradas sean correctas.

Uno de los aspectos computacionales más claramente susceptibles de mejora de *ACTRES Corpus Manager* es el débil reconocimiento de MWE, que depende en su totalidad de la capacidad de identificación de estas expresiones por parte del tokenizer interno de Treetagger. La identificación de MWE por parte de Treetagger es escasa, sobre todo en lenguas distintas del inglés. La alternativa de expandir el lexicón de MWE de Treetagger para cada idioma requería tiempo y conocimiento de cada uno de los idiomas específicos, por lo que fue desechada.

ACTRES Corpus Manager soporta etiquetado retórico supervisado limitado a corpus monolingües y a la temática de actas de reuniones en inglés y español. Sin embargo, la idea inicial era incorporar un etiquetador retórico no supervisado, basado en técnicas de *Machine Learning*, por medio de un modelo entrenado a partir de corpus etiquetados retóricamente de forma manual. De esta forma se permitiría su incorporación en el flujo de creación de cualquier corpus. No obstante, se consideró que el desarrollo de esta tarea excedía los límites de esta tesis doctoral y

que habría de abordarse en futuras investigaciones. Por este motivo se creó un etiquetador retórico supervisado, que sirviera de declaración de intenciones, y demostrara que *ACTRES Corpus Manager* es capaz de tratar este tipo de anotación.

Con el objetivo principal de que el flujo de creación de corpus lingüísticos etiquetados retóricamente no requiriera una carga de tiempo excesiva, se optó por implementar la solución descrita en el punto "IX) Etiquetador retórico" correspondiente al Capítulo 6. Esta solución consiste en limitar la incorporación del etiquetado retórico al tipo de corpus más simple (monolingüe), y a una temática más o menos sencilla donde los textos no fueran excesivamente largos (actas de reuniones) utilizando un estándar de etiquetas del grupo de investigación ACTRES.

ACTRES Corpus Manager permite la extracción de estadísticas como: listas de frecuencias de los corpus, listas de frecuencias de la consulta, colocaciones basadas en frecuencias o lista de palabras clave. Cada una de estas estadísticas tiene como referencia el corpus o subcorpus sobre los que se realiza la consulta, es decir, en el caso de los corpus paralelos, estas estadísticas se extraen de acuerdo con el subcorpus donde se realiza la consulta o que el usuario selecciona. Estos subcorpus son internamente tratados como corpus monolingües. La implementación de estadísticas relacionadas con la comparación de corpus (Kilgarriff, 2001), ya sean paralelos o comparables, decidió no incluirse en *ACTRES Corpus Manager*.

La extracción de palabras clave es la única estadística cualitativa empleada, se basa la ejecución de tests de significación estadística utilizando para ello la funcionalidad proporcionada por el paquete estadístico de polmineR. Respecto a las estadísticas cuantitativas, *ACTRES Corpus Manager* soporta la extracción de listas de frecuencia del corpus, listas de frecuencias de la consulta, colocaciones y extracción de n-gramas.

El número máximo de subcorpus admitidos por *ACTRES Corpus Manager* para la creación de corpus paralelos y comparables se ha limitado a cuatro. Esto se debe a

que, de acuerdo con las pruebas realizadas, cuatro es un número adecuado para que el resultado de las consultas paralelas sea mostrado por pantalla de forma adecuada y legible. Además, es un número asumible para la capacidad computacional del servidor utilizado, sobre todo en lo que respecta al proceso crítico de la alineación.

Se pretendía disminuir cuanto fuera posible el pre-tratamiento de los textos y la inclusión de datos en los formularios de creación de corpus lingüísticos. Por este motivo se declinó la posibilidad de incluir metadatos tales como autor, fuente, género textual para cada texto o corpus. Siempre queda la posibilidad de implementarlo, pero la decisión en este framework ha sido la de no incluirlo.

Por último, destacar la ausencia de funcionalidades relacionadas con la incorporación de corpus pre-tratados en *ACTRES Corpus Manager*. El motivo es que cada tipo de corpus pre-tratado puede requerir una implementación distinta, dependiendo del formato de corpus empleado, formato de los etiquetados lingüísticos realizados, formato de la alineación en el caso de corpus paralelos, etc. Esta multiplicidad desaconsejó su incorporación en el framework actual.

Capítulo

9

Conclusiones y líneas futuras

9.1 A Conclusiones

Una vez completada la fase de validación del *ACTRES Corpus Manager*, estamos en situación de afirmar que esta tesis ha dado respuesta a las preguntas que se plantearon en los capítulos introductorios.

- Cómo abordar la creación y gestión autónoma de corpus bi-/multilingües desde el punto de vista del usuario no técnico, cuyos intereses son meramente lingüísticos y no computacionales.
- Cómo analizar la usabilidad y utilidad de las distintas aplicaciones que intervienen en las subtareas técnicas necesarias para la construcción de un corpus bi-/multilingüe.
- Cómo diseñar el flujo de ejecución de un framework que permita la creación y análisis de corpus lingüísticos con múltiples capas de anotación sin requerir ningún procesamiento externo de los documentos del corpus, ya sea por parte de personal técnico, de scripts de programación ad-hoc o de otros programas software.

En cuanto a las conclusiones a las que se ha llegado tras la realización de esta tesis doctoral, se presentan a continuación:

- La incorporación de anotaciones al corpus por parte de usuarios lingüistas requiere la ejecución de múltiples procesos previos y la posterior adaptación del corpus al formato requerido por cada software específico.
- Si el corpus lingüístico que se desea crear es bi-/multilingüe paralelo o comparable, el proceso es aún más complejo, dándose la circunstancia de que además de tener que ejecutar procesos adicionales, por ejemplo, alineado, se ha de encontrar un software que permita su procesamiento.
- Dado que el usuario lingüista no tiene por qué poseer los conocimientos técnicos necesarios que le permitan realizar estos procesos, en muchas ocasiones, la participación de personal técnico especializado para realizar el proceso es indispensable.
- La preponderancia de la lengua inglesa como eje principal de desarrollo de aplicaciones relacionadas con el procesamiento de corpus lingüísticos incide negativamente en la ausencia de versiones o de software equivalente para lenguas distintas del inglés, lo que repercute en la dificultad de realizar los procesos en el corpus no inglés.
- El análisis de software presentado en la sección 3.3 Análisis de software disponible evidencia la inexistencia de frameworks específicos que posibiliten a los usuarios lingüistas crear y analizar sus propios corpus lingüísticos (monolingües, bi-/multilingües paralelos y comparables) sin la asistencia de personal técnico en el caso muy frecuente de no disponer de conocimientos de programación.
- *ACTRES Corpus Manager* crea flujos de ejecución cerrados que evitan la necesidad de intervención técnica. Cada uno de los procesos necesarios para incorporar un corpus en el sistema de gestión está implementado por métodos específicos de cada clase de programación (por ejemplo, la clase *CrearCorpusMono*). Este hecho favorece la reutilización y modificación futura de la lógica del framework.

- La combinación e interconexión de software existente (CWB, Treetagger y hunalign entre otros) con software propio basado la utilización de librerías y toolkits tales como NLTK, polmineR, Bootstrap y en recursos lingüísticos del grupo ACTRES (etiquetas retóricas o diccionarios) o externos (lexicones semánticos de USAS y recursos liberados en la red) aseguran la eficiencia y validez del nuevo software.

9.1 B Conclusions

Once completed the validation stage, we are in a position to claim that this dissertation has provided responses to the questions posed in the introductory chapters above:

- How a non-technical user may deal autonomously with the creation and management of bi/multilingual corpora when his/her main interests are merely linguistic and not computational.
- How to assess the usability and usefulness of the software applications responsible for the technical subtasks involved in building a linguistic corpus.
- How to design an execution flow that allows creating and analysing corpora, as well as annotating at different layers without having to resort to the external processing of the corpus documents by technical specialists, or having to use *ad hoc* programming scripts and/or external software.

The conclusions reached after completing all the stages of this doctoral research are as follows:

- The addition of linguistic annotations to the corpus by the linguistic user demands the execution of a number of previous processes as well as the subsequent adaptation to the particular format requirements of the corpus linguistics software.
- If the user wants to create a bi/multilingual parallel or comparable corpus, the process becomes increasingly more complex because of the additional

processes that have to be carried out. For instance, alignment requires the execution of external alignment software.

- As the linguistic user does not necessarily have the technical skills required for executing these processing tasks, very often, the participation of technical personnel to carry them out is indispensable.
- The prevalence of English as a working language in corpus linguistics software development presents a serious downside as it creates the illusion that it "travels well" into any other language. This has affected negatively the development of similar non-English-centred software. To date, this situation makes even more difficult to carry out many of the processes in a non-English corpus.
- The software analysis presented in 3.3 Analysis of currently available software, clearly shows an absence of specific frameworks that allow linguistic users to build and analyse their own corpora (monolingual, bi/multilingual parallel and comparable) independently, without technical assistance in those very frequent cases where the linguist does not have programming skills.
- *ACTRES Corpus Manager* creates closed execution flows that spare the need for technical intervention. Each one of the required processes needed to load a corpus in the management system is implemented by methods specific to each particular programming class (for instance, *CrearCorpusMono* class). This favours future reuse and modification of the logic of the framework.
- The combination and interconnection of already existing software (for instance, CWB, Treetagger and hunalign among others) with customised software based on programming libraries and toolkits such as NLTK, polmineR, Bootstrap, together with the linguistic resources of the ACTRES research group (rhetorical tags and lexicons) as well as external resources

(for instance, USAS semantic lexicons and other lexicons released on the internet) ensure the efficiency and validity of the *ACTRES Corpus Manager*.

9.2 A Líneas futuras

Las líneas de trabajo futuro que derivan o están relacionadas directamente con el desarrollo de *ACTRES Corpus Manager* son múltiples. Entre ellas cabe destacar las siguientes:

- Superar las barreras descritas en el Capítulo 8: "Discusión", en cuanto a las limitaciones técnicas y lingüísticas afrontadas durante el desarrollo del framework.
- Incluir un etiquetado retórico no supervisado que permita etiquetar automáticamente textos prototípicos procedentes de diversos géneros textuales.
- Implementar funciones estadísticas cualitativas y cuantitativas más complejas que permitan un análisis del corpus más profundo, así como formatos de exportación de resultados más visuales, por ejemplo, utilizando gráficos.
- Incorporar funcionalidades adicionales relacionadas con extracción de automática de tesauros, análisis multi-variable de Biber (Biber, 1991), etc.
- Ampliar los recursos lingüísticos utilizados en la alineación, etiquetado gramatical, etiquetado semántico y etiquetado retórico para mejorar los resultados de los programas.

Como epílogo a esta tesis, manifestar la importancia que ha demostrado tener la colaboración entre grupos de investigadores de distintas áreas de conocimiento. El futuro de la investigación reside en potenciar la creación de grupos de trabajo interdisciplinares que posibiliten el desarrollo de proyectos que aúnen conocimientos de múltiples disciplinas científicas. De esta forma se obtendrán resultados susceptibles de ser aplicados en los entornos más dispares, y no sólo en casos aislados y/o controlados por un grupo de trabajo específico.

En el caso de esta tesis doctoral, la cooperación con los investigadores lingüistas del grupo de investigación ACTRES resultó decisiva e indispensable para que el desarrollo de *ACTRES Corpus Manager* fuese realmente útil para los usuarios finales y no un nuevo software más para la gestión de corpus lingüísticos.

9.2 B Further work

There is a number of aspects in the development of *ACTRES Corpus Manager* that call for further work. These include the following:

- Overcoming barriers described in Chapter 8, "Discussion" concerning the technical limitations faced during the development of the framework.
- Including unsupervised rhetorical tagging, which will allow automatic tagging of prototypical texts in a variety of genres.
- Implementing more complex qualitative and quantitative statistical functions that allow for more precise analyses and the exportation of results by means of more visual formats, for instance, graphic charts.
- Adding extra functionalities to the automatic extraction of lexicons (such as thesaurus), multivariate analysis (Biber, 1991), etc.
- Enlarging the corpora used to test the alignment and the grammatical, semantic and rhetorical tagging processes so as to improve their effectiveness of these programs.

As an epilogue to this dissertation, it must be acknowledged that collaboration between research groups from different fields is where the future of research lies. Interdisciplinary groups provide a communication-rich environment for the development of innovative projects that capitalize on multiple scientific disciplines.

In this doctoral research, cooperation with ACTRES group linguistic researchers was crucial and essential in order to make *ACTRES Corpus Manager* truly useful for end users, and not just another custom corpus analysis software.

Glosario

Aquellos términos que ya han sido a lo largo de la tesis, incluirán una referencia al cuerpo de la tesis doctoral.

***Ad hoc* (programa)**
Programa informático diseñado para resolver un problema muy específico y por naturaleza inútil para otros cometidos.

Alinear
Página 30

Análisis del discurso
El análisis del discurso es la disciplina dedicada a la investigación entre la forma y la función en la comunicación verbal (Renkema, 2004, pág. 1).

Anotación
Página 35

API
Del inglés *Application Programming Interface*, traducido al español como "Interfaz de Programación de Aplicaciones", de acuerdo con (Sharpened Productions, 2017) se refiere al conjunto de comandos, funciones, protocolos y objetos que los programadores pueden usar para crear software o interactuar con un sistema externo. Proporciona a los desarrolladores un estándar de comandos para llevar a cabo operaciones comunes de tal forma que no sea necesario escribir el código desde cero.

Búsquedas caché
Una búsqueda caché es aquella cuyo resultado está almacenado de antemano en la memoria y que por tanto no requiere de una nueva ejecución de la consulta para obtenerlo.

Colocación
Página 35

Compilador (informático)
Software que transforma un programa escrito en código fuente (alto nivel) en código objeto (bajo nivel) comprensible para una máquina.

Concgram
Concgram son todas las permutaciones posicionales y de constitución surgidas por la asociación de dos o más palabras (Cheng, Greaves, & Warren, 2006, pág. 411).

Concordancer
Página 41

Concordancia
Página 36

Corpus bi-/multilingüe paralelo
Página 9

Corpus cerrado
Página 10

Corpus comparable
Página 9

Corpus de referencia
Página 10

Corpus diacrónico o histórico
Página 10

Corpus escrito
Página 8

Corpus especializado
Página 10

Corpus lingüístico
Página 5

Corpus monolingüe
Página 8

Corpus oral
Página 7

Corpus sincrónico
Página 10

Corpus-based approach
Corpus-based approach es una metodología que emplea un corpus como base sobre la que analizar una hipótesis de investigación, de tal forma que a partir de un corpus representativo la pregunta se refuta, acepta o modifica a través de análisis estadísticos y otros métodos.

Corpus-driven approach
Corpus-driven approach es una metodología que emplea el corpus como punto de inicio de sus estudios lingüísticos, observan un patrón lingüístico y lo estudian. Su ausencia o presencia en un corpus se valora con distintas interpretaciones basadas en distintos aspectos, por ejemplo, la situación histórica, política, etc.

Debug
Debug significa depurar en español, y se refiere al proceso de identificación y consecuente corrección de errores existentes en un programa informático.

Etiquetado
Página 35

Expresión regular
De acuerdo con (Baker, Hardie, & McEnery, 2006, pág. 138) es un tipo de *string* o cadena de texto que puede incluir caracteres especiales (denominados *wild cards* o comodines), que posibilitan que la expresión regular coincida con más de una cadena. Por ejemplo, el carácter punto "*.*" es una expresión regular que puede representar una simple letra. Así la expresión regular "*est.*" puede significar *esto*, *esta*, *este*, etc.

Framework
Página 23

Framework interdisciplinar
Página 24

Framework para el tratamiento de corpus lingüísticos
Página 25

Herramienta (informática)
Página 23

Indexación
Página 30

Inyecciones de código
Es el término general para designar a los tipos de ataque informáticos que consisten en inyectar código que luego es interpretado/ejecutado por la aplicación. Estos tipos de ataques suelen ser posibles debido a la falta de validación de datos de entrada/salida (OWASP, 2013).

IR
Del inglés *Information Retrieval*, traducido al español como "Búsqueda y recuperación de información", de acuerdo con (Baker, Hardie, & McEnery, 2006, pág. 90) se refiere al

estudio y uso de ordenadores como medio para separar información concreta de una gran cantidad de datos, como un corpus, base de datos o red de textos como la web. Una búsqueda web es un ejemplo de IR.

Librería (informática)
Página 23

Lingüística computacional
Página 11

Lingüística de corpus
Página 11

Log (archivo)
Es un tipo archivo utilizado para registrar los distintos eventos ocurridos en un sistema informático.

Marcar
Página 28

Metadatos
De acuerdo con (Baker, Hardie, & McEnery, 2006, pág. 115) los textos en un corpus son datos, luego la información sobre los textos de un corpus son los metadatos (datos sobre datos). Esta información puede incluir el título, autor, editorial, etc.

Monitor corpus o corpus abierto
Página 9

MWE
Del inglés *MultiWord Expression*, traducido al español como “Expresiones multipalabra”, según (Sprenger, 2003, pág. 4) son combinaciones específicas de dos o más palabras que se suelen utilizar para expresar un concepto determinado.

NER
Del inglés *Named-Entity Recognition*, traducido al español como “Reconocimiento de Entidades Nombradas”, de acuerdo con (Baker, Hardie, & McEnery, 2006, pág. 120) se refiere a la identificación y etiquetado de nombres en un texto: nombres de personas, lugares, empresas y otras organizaciones. Su realización implica el análisis de la gramática y semántica del texto, por ejemplo, la identificación de nombres propios requiere de un etiquetado gramatical.

NLP
Página 12

n-grama
Página 35

Parsear
Página 28

Responsive (diseño)
Diseño responsive o adaptable se refiere a la adaptación del contenido de una web al tamaño de la pantalla donde se muestra.

REST
Del inglés *REpresentational State Transfer* traducido al español como "Transferencia de Estado Representacional", de acuerdo con su creador es un estilo de arquitectura para sistemas hipermedia que proporciona un conjunto de restricciones que, aplicadas en su conjunto, enfatizan la escalabilidad de las interacciones de componentes, la generalidad de interfaces, el despliegue independiente de componentes y de componentes intermedios para reducir la latencia de interacción, reforzar la seguridad y encapsular sistemas heredados (Fielding, 2000, Cápitulo 5).

Script
Es un programa formado por una secuencia de instrucciones o comandos de fácil acceso y modificación por parte de los programadores, ya que no requieren ser compilados.

Software
Página 22

Software especializado para el tratamiento de corpus lingüísticos
Página 23

Tokenizar
Página 28

Toolkit, *suites* y similares
Página 23

Verticalizado (documento)
Un documento verticalizado es aquel que tiene un formato de una palabra por línea (*one-word-per-line*) o más concretamente un *token* o símbolo por línea, con la forma superficial o visible en la primera columna y el resto de anotaciones a nivel de *token* especificadas en columnas adicionales separadas por tabulaciones. Las etiquetas XML han de aparecer en líneas separadas (Evert, 2016b, pág. 2).

WSD

Del inglés *Word-Sense Disambiguation*, traducido al español como "Desambiguación lingüística", de acuerdo con (Navigli, 2009, pág. 1) es la capacidad de identificar el significado de las palabras en un contexto de un modo computacional.

Summary

This dissertation sets out to develop a corpus manager software, the *ACTRES Corpus Manager*, conceived and designed from the standpoint of language researchers who routinely work with at least two languages simultaneously. The *ACTRES Corpus Manager* will allow linguists to build, annotate at different annotation layers, align and process, bi/multilingual and comparable corpora, as well as their own monolingual corpora without technical assistance during the process. In this thesis, the working languages are English, Spanish, French and Italian (except for rhetorical tagging, that only applies to Spanish and English).

Chapter 1 is the introduction to this Doctoral thesis. It focuses on the reasons that have triggered the project and reviews the state of the art in corpus analysis tools where Spanish is concerned.

A review of the state-of-the-art shows that available software forces the linguistic user to use external, independent programs (i.e. not included in the same framework) that are not necessarily compatible with each other to carry out actions such as annotation or alignment of parallel corpora. A further issue is the prevalence of English over any other language in corpus technology developments. This causes the wrong thought that English language-based technology can also directly cater to corpora that include languages other than English, which is misleading, as the programming has been done exclusively on the conceptualization of English and there are significant gaps and mismatches when it comes to processing an additional language, e.g. Spanish.

It should be emphasised that *ACTRES Corpus Manager* is not designed as an alternative to replace well-known web interfaces for accessing large or reference corpora such as corpus.bye.edu or Lancaster University CQPweb platform. Its main purpose is to allow linguists without - or with very limited programming skills to build their own corpus (monolingual, bi/multilingual parallel and comparable) and

carry out different language processing analyses avoiding the execution of external software or/and processing scripts and without additional technical assistance.

Chapter 2, "Key concepts", serves to introduce Engineering readers to the corpus linguistics and computational linguistics research fields. This thesis draws on two major fields, Computing and Linguistics. It has been carried out as part of the "Intelligent Systems in Engineering" doctoral programme, with a significant contribution from the "Contrastive Studies Spanish/English/French" program. This chapter is intended to give an overview to those readers who may be unfamiliar with concepts in corpus and computational linguistics.

Section 2.1 defines what a corpus is and the different types of linguistic corpora.

Section 2.2 reviews the research on linguistic corpus and computing, paying special attention to the fields of corpus linguistics, computational linguistics and natural language processing (NLP) as well as their mutual relationships. Then, corpus linguistics, as an empirical scientific methodology, is discussed and validated.

Section 2.3 provides some technological background to corpus processing software. First, in order to help (linguistic) readers, who may be unaware of technical argot, some definitions about technological concepts (such as software, framework or toolkit), are provided. Then, a complete overview of the main types of software involved in processes or actions related to corpus building and corpus analysis are present. Depending on its use, software is classified into six categories:

(1) Natural Language Processing
(2) Linguistic taggers
(3) Aligners
(4) Corpus indexing and querying
(5) Statistics
(6) User interface

How this software is employed in corpus linguistic processing is described in detail and some real examples of applications are offered. A basic workflow for creating and analysing a corpus using the technologies outlined above is presented here.

Section 2.4 defines some linguistic terms that could be misunderstood depending on linguistic school affinity. For instance, collocation, n-gram, tagging, annotation and concordances are defined in the way they are actually going to be used in this dissertation.

Chapter 3 describes the state-of-the-art of corpus linguistics software.

In section 3.1, the main problems and limitations related to corpus linguistics software since the 50s until the present time are discussed. Some of them include the following:

- Problems related to compilation, availability and use of early mainframes are considered.
- Limitations of concordancing methods, from early days manual concordancing until today's concordancers.
- Range of available statistics: Different types of statistics applied to corpora are analysed in detail. From simple quantitative methods, such as list of frequencies, until current qualitative methods based on statistical significance tests such as keywords extraction.
- Replicability: Until the first IBM PC appeared (1981), it was very difficult to replicate the results obtained by another researcher, because searches had to use the same corpus in the same mainframe and the same software. Installing the software or the corpus in other machine was a daunting task. The popularization of PCs and the sharing of software programs made it easier to reproduce and replicate searches and queries on a given corpus.
- Software incompatibility: In the absence of a common framework that allows users to carry out all the tasks employing the same software,

incompatibility problems often appear. As the output of different programs may not necessarily correspond to the input of others

- Encoding problems: Before the generalization of Unicode's UTF-8, which permits the representation of any symbol in any language, displaying language characters different from those of English was a complicated, time-consuming process.
- Linguistic annotation inconsistence: each linguistic tagger employs a different tagset for categorizing words. This tagset is based on the opinion of linguistic experts as there is no universal understanding among linguistic researchers in order to create a standard tagset. As a result, each tagger employs a different tagset for each language. In order to overcome this problem, mapping programs between different tagsets must be used.

Section 3.2 presents a classification of corpus linguistics software based on McEnery and Hardie's historical overview of corpus analysis tools (McEnery & Hardie, 2012, pp. 37-48). The proposed classification also takes into account the limitations detailed above. The software analysed here is what McEnery and Hardie called concordancers, meaning software that allows user to make concordances over a corpus.

It follows an explanation of what is the situation of the corpus linguistic software at present is showed. In spite of the advantages of the 4th generation of concordancers over the 3rd one (web access, fast and more potent queries, support large corpus, concurrence of users over the same corpus without replicate corpus itself, multiplatform, etc.), 3rd generation concordancers are still widely used. Some reasons, among others, that can help to explain this are:

(1) The research questions that can be answered with software of 3rd generation and 4th are quite similar.

(2) The power of current personal computers is enough for analyse corpora of medium size, which means that researchers do not have the need to invest in 4th generation hardware.

(3) The high cost of 4th generation concordances, either a in the form of subscription or maintenance fees.

(4) Investment in 3rd generation software licences as well as the know-how to use it.

(5) Poor usability of the access and registration forms for 4th generation software.

(6) Proliferation of external tools and *ad hoc* applications that are necessarily incorporated on demand to 3rd generation workflow.

Section 3.3 is devoted to a detailed analysis of corpus building software. Software is divided into two categories: (1) integral frameworks and (2) toolkit, suites and others.

- Frameworks for corpus processing are defined as any computer application that integrates different functionalities that allows users to create and make concordances over their corpora, without leaving the framework and without any special programming skill. Frameworks are analysed according to the following guidelines:
    - Compilation of corpus: Does it allow users to create their own corpora? Or does it only allow querying existing pre-loaded corpora?
    - Technical assistance: Does it require some technical assistance or having some programming skills?
    - Multilingual support: Does it allow corpora in languages other than English?
    - Parallel corpora support: Does it support bi-/multilingual parallel corpora?
    - Comparable corpora support: Does it support comparable corpora (Page 9)?
    - Does it use any indexation technology?
    - Does it include statistics functionalities? Which ones?

  - Does it include a functional user interface?
  - Which generation according to McEnery & Hardie does it belong?
  - Does it includes any built-in linguistic taggers?
  - Is it web based?
  - Can the software be downloaded to be used in user computer or in an own server?
- Regarding the second type of software, toolkits, suites and other tools, they are related to software development. In terms of usability, these do not cater to linguistic users independence as most of them allow them to build and analyse their own corpus only if these users actually have programming skills.

The conclusion of Chapter 3 is that there are no there are no specific frameworks that allow a non-technical user to create and analyse bi-/multilingual parallel and comparable corpora without technical assistance during the process. Software able to process parallel corpora requires technical assistance and/or some programming skills to use alignment programs, create and run parsers, or use specific operating environments. Concerning the creation and analysis of comparable corpora, there is no specific operational software for this task. The automatic incorporation of grammatical (POS) annotation has been overcome in some frameworks like SketchEngine, but there are still gaps and limitations in the other tagging layers. For instance, Wmatrix only supports semantic tagging for English. Rhetorical tagging has not been tracked in any of the frameworks analysed.

Chapter 4 is devoted to the "Working hypothesis: needs, aims and niche". As stated earlier, PhD dissertation involves the development of a framework for building and analysing bi/multilingual parallel and comparable corpora, as well as monolingual corpora with several annotation layers. This framework is specifically addressed to professional linguists. It is, therefore, a doctoral thesis designed from the point of view of the usability and the applicability by these end users, avoiding specialized

intervention by technical personnel in any of the processes involved in corpus building or analysis.

The initial hypothesis is that the creation of annotated corpus requires technical support or at least some programming knowledge. This is even more obvious in the case of parallel and comparable corpora. In section 3.2, we have shown that there is no software that allows the user to create and analyse annotated corpora as those previously described independently, without any technical assistance. For instance, aligning or inserting semantic tagging do require the participation of technical staff, more so if languages other than English are involved. In addition, comparable corpora, as defined in Chapter 1, are not supported by any of the software analyzed here. In the case of rhetorical tagging it is neither implemented as such in any framework. Everything is more complex in case of any of the languages will not be English, because most of the software is developed around it. In addition, no rhetorical tagging or comparable corpora, as defined in Chapter 1, are supported by any of the analysed software.

In order to develop this framework it is essential to:

- Understand and define the different activities and tasks, at both low and high level, necessary to process the corpus: data format, input and output data flows, order of execution among others.
- Identify which activities are critical concerning programming skills or knowing how to use specific software, such as linguistic taggers or aligners.
- Evaluate the autonomy granted to the user by each activity.
- Be familiar with corpus processing software used in the past and at present. This software includes not only integral frameworks, but any application or tool necessary to load a corpus into a concordance software.
- Find out which software is the most suitable and the most widely used for each of the processes. Put special emphasis in discovering strengths and

weaknesses, and be ready to develop your own software and/or upgrade already existing programs.

Thus, the framework intends to fill an existing gap in corpus linguistics regarding to the lack of independence of non-technical users in Linguistics to build and analyse any type of annotated corpus without technical assistance.

Chapter 5 is devoted to methodology. Development of proposed framework does not belong to any systems development life cycle because development team is made up by only one researcher, this doctoral candidate, and software development is not based on specific tasks. Anyway, software development is inspired in Rapid application development (RAD) (Martin, 1991) and the Extreme programming (Maurer & Martel, 2002) software development model.

Four methodology phases can be distinguished:

(1) **Analysis**: The purpose of this phase is to define the requirements and use cases of the software.

(2) **Design**: The internal components of the framework as well as their corresponding data flow and interactions are identified through the creation of several diagrams.

(3) **Implementation** of the software in accordance with the design in the previous phase.

(4) **Validation**: The software has to be tested in order to verify that it works correctly. These tests are an empirical verification that proves that the framework meets the requirements established in the design and implementation phases. The validation phase consists of two processes:

- Tests, and,
- evaluation of system usability through a SUS questionnaire (Brooke, 1996).

Analysis, design, implementation and testing are outlined in **Chapter 6.** As mentioned, because of the interdisciplinary nature of this PhD dissertation, some

concepts related to design and analysis required further explaining. This type of information is not commonly included in a common PhD dissertation belonging to any Computer Science doctoral programme.

Section 6.1 describes the analysis process. It mainly consists of establishing software requirements and design use cases. The requirements can be summarized as follows:

- Web access.
- User authentication.
- Corpus building without technical assistance.
- Support:
    - medium-sized and large corpora processing
    - corpora in different languages
    - monolingual corpora
    - bi-/multilingual parallel corpora
    - comparable corpora
- Grammatical (POS), semantic and rhetorical tagging.
- Support frequently used corpus statistics:
    - Collocations.
    - Keywords.
    - Frequency lists.
    - Frequency of queries.
    - N-grams.
- Support concordances:
    - Key-Word-In-Context format.
    - Personalised search options.
    - Use of assisted regex patterns.
- Responsive user interface

On the other hand, five use cases are distinguished:

(1) Building a monolingual corpus
(2) Build parallel corpus
(3) Build comparable corpus
(4) Building a monolingual corpus with rhetorical annotations.[63]
(5) Make a concordance over any type of corpus
(6) Calculate statistics over any type of corpus

Section 6.1 describes the design phase. The software design is a combination between traditional MVC (Model-Viev-Controller) (Reenskaug, 1979) and MVVM (Model–view–viewmodel) (Gossman, 2005). More precisely, the design makes use of the intelligent view of MVVM design pattern, which allows for separating view logic from business logic. This is combined with the controllers from classical MVC. The design pattern uses the typical three-tier architecture employed in web applications.

Section 6.2 outlines the implementation phase.

Firstly, an explanation about our approach is needed. Corpus building and analysis involves the participation and collaboration of different technical disciplines related, above all, to computational linguistics. Creating a tailor-made software solution from scratch requires a lot of time, close collaboration with experts in linguistics and professional competence in several technical areas such as statistics, database performance, NLP or user interface design.

If only one researcher were in charge of developing each one of the internal components needed in the framework, it would take such a long time that it would be incredibly time-consuming , as it would imply becoming an expert in each and all of the processes involved so as to produce an acceptable corpus processing environment.

[63] Because of the need to carry out manual rhetorical tagging, this kind of corpus is built independently.

For these reasons, reuse, modification and interconnection of existing software resources, whose validity has been sanctioned by the scientific community is the best option. This strategy is combined with the creation of new software resources to fill in the gaps. These may be caused by immature resources, as revealed by our analysis 3.3 Analysis of currently available software, or by extreme restrictiveness of the software requirements concerning data format, power processing, etc.

It is also essential to create our own custom controllers, as well as models that combine already existing and purpose-built custom software in order to produce a controlled execution flow that allows end users to build and analyse any type of corpus.

Programming languages, both proprietary and our own software, and limitations at the implementation phase are described in detail.

Framework components have been organized in three blocks: Model, View and Controller. An extensive description of each element is offered, including problems encountered, alternative approaches, data flow and other aspects related to the logic of the framework.

Additionally, a brief summary of each framework component /block is outlined.

- View

It contains the visual elements (buttons, forms, text entries, etc.) of the framework and the associated logic to deal with them. All the user interactions that do not require access to the framework model are part of the view logic.

Interface visual elements have been developed employing the Bootstrap front-end framework, which allows the user to build responsive webs based on HTML5, CSS3 and JavaScript. On the other hand, the logic of the view employed JQuery library as the main programming resource, with the support of several Javascript libraries.

- Controllers

Controllers act as intermediaries between the view and the model. Controllers will manage those user interactions that require access to the data model of the framework.

Controllers communicate view and model through Javascript, more specifically via AJAX calls or traditional HTTP requests. Each controller converts the data entered by the user into parameters intelligible to the corresponding model and vice versa, transforming the responses of the model into different operations that modify the view.

- Model

The Model component of the framework is composed by data and logic, with the exception of the logic of the view, which has been defined above. The IMS Corpus Workbench (CWB) is the selected software for corpus indexing and managing. This selection has a strong influence on the implementation of the model because it has to follow all its requirements.

- Data model

Two types of configurations form it:

(1) User data and associated corpus information: It comprises two databases, one containing user data and the other corpus information. It is implemented using JSON format.

(2) CWB data: data model employed for indexing, managing and querying corpora, as required by the CWB workbench.

- Logic of the framework

PHP was the selected programming language because:

- Its web nature, which fits the requirements of internet access of a 4th generation framework. This feature makes unnecessary the use of an

additional programming language to serve as a link between the server and the client.

- The extensive functionality of PHP, which allows PHP to be used for almost any purpose.
- Its simple and readable syntax, which makes it easy-to-use, reuse and modify.

The Logic of the framework is programmed employing the OOP[64] (Object-Oriented Programming) paradigm, which comprises classes and methods. Class methods make use of both external and custom-built software as well as scripts in order to carry out the functionality described in use cases (page 356).

Chapter 7 is devoted to the validation of the framework and it comprised two sections:

- Section 7.1 shows real examples of corpus building and corpus processing using the framework. For instance, see Figure 177.

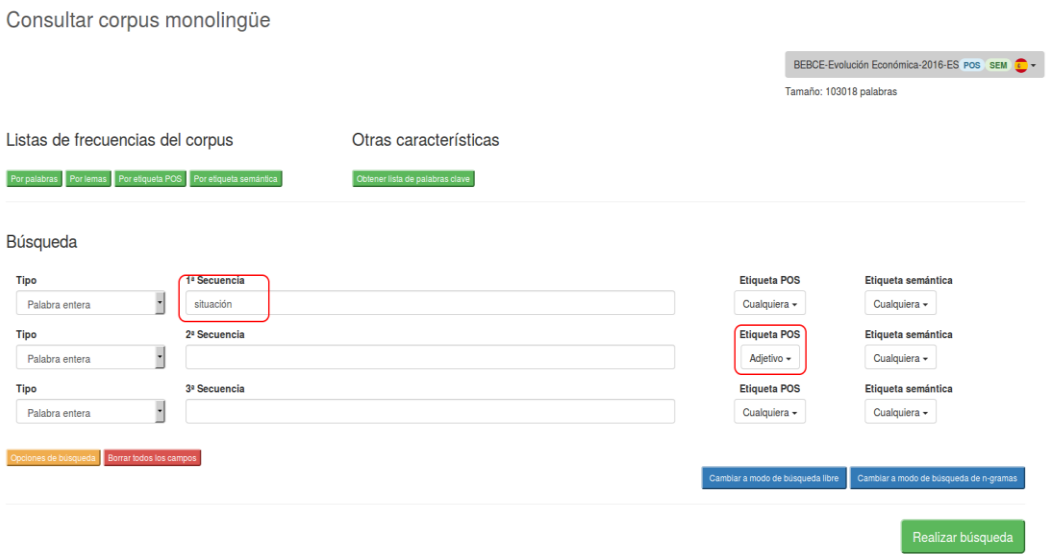


Figure 177 - Image of *ACTRES CORPUS MANAGER*

64 Who may be unfamiliar with OOP concepts, consult (Lavin, 2006)

- Section 7.2 gives the results of the usability test carried out via System Usability Scale (Brooke, 1996).

Chapter 8 discusses what has been attained and the limitations that need further looking into.

- Currently the framework supports English, Spanish, French and Italian. As *ACTRES Corpus Manager* is a prototype, it does not make sense to developed linguistic resources for more languages for demonstration purposes.
- Multiword expressions are poorly recognised because of the Treetagger functionality.
- Rhetorical tagging is only supported for monolingual corpora- or comparable corpora treated separately as monolingual corpora- because it has to be carried out manually. An unsupervised rhetorical tagger will be designed in the future.
- As the priority has been to minimize the pre-treatment of texts and the manual inclusion of data into framework interface, so as to save time to the user, the possibility of including metadata such as author, source, genre, etc. for each text or corpus was not implemented. Work on metadata is already underway and it will be fully implemented in future, upgraded versions.
- The present version of the framework supports four number of sub-corpora for parallel and comparable corpora at most. This a valid number in order to display results properly and get acceptable times for corpus building or querying.
- In this version, only raw text can be used, the plans are that future versions will allow for the use of pre-treated corpora.

Chapter 9 is devoted to the conclusions and further work. A full version in English is showed in 9.1 B Conclusions and 9.2 B Further work.

Additionally, a **glossary** is provided so as to clearly define all the relevant terms and concepts employed throughout the document, and to cater to the interdisciplinary needs of engineers and linguists alike.

Finally, the section **References** lists all the documentation and technical resources consulted and used in the elaboration of the research reported in this dissertation, and the section **Appendix** provides supplementary information

Referencias

Nota: Todos los enlaces a los documentos electrónicos que aparecen en las siguientes referencias han sido comprobados con fecha de junio de 2017

Abercrombie, D. (1965). Pseudo-procedures in linguistics. En D. Abercrombie (Ed.), *Studies in Phonetics and Linguistics* (págs. 114-119). Oxford: Oxford University Press.

ACTRES. (2017a). *Corpus 0 C-GARE: Actas de Reuniones*. Obtenido de http://contraste2.unileon.es/web/es/corpus0_GARE.html

ACTRES. (2017b). *Generadores para escritura*. Obtenido de http://contraste2.unileon.es/web/es/applications.html

ACTRES. (2017c). *Página Web del grupo de investigación ACTRES*. Obtenido de http://actres.unileon.es/

Aijmer, K., Altenberg, B., & Johansson, M. (Eds.). (1996). *Languages in contrast: papers from a symposium on text-based cross-linguistic studies* (Vol. 88). Lund: Lund studies in English.

Andor, J. (2004). The master and his performance: an interview with Noam Chomsky. *Intercultural Pragmatics, 1*(1), 93-112.

Anthony, L. (2005). AntConc: a learner and classroom friendly, multi-platform corpus analysis toolkit. *Proceedings of IWLeL (Interactive Workshop on Language e-Learning)*, (págs. 7-13). Tokio: IWLeL.

Anthony, L. (2013a). A critical look at software tools in corpus linguistics. *Linguistic Research, 30*(2), 141-161.

Anthony, L. (2013b). Developing AntConc for a new generation of corpus linguists. *Proceedings of the Corpus Linguistics conference (CL 2013)*, (págs. 14-16). Lancaster: UCREL.

Anthony, L. (2014a). AntConc (Versión 3.4.3) [Software]. Tokio: Waseda University. Obtenido de http://www.laurenceanthony.net/

Anthony, L. (2014b). AntPConc (Versión 1.1.0) [Software]. Tokio: Waseda University. Obtenido de http://www.laurenceanthony.net/

Anthony, L., & Hardaker, C. (2016). FireAnt (Versión 1.0) [Software]. Tokio: Waseda University. Obtenido de http://www.laurenceanthony.net/

Archer, D., Wilson, A., & Rayson, P. (2002). *Introduction to the USAS Category System.* Obtenido de http://ucrel.lancs.ac.uk/usas/usas%20guide.pdf

Baker, M. (1993). *Corpus linguistics and translation studies: Implications and applications.* Ámsterdam: John Benjamins.

Baker, M. (1995). Corpora in translation studies: An overview and some suggestions for future research. *Target, 7*(2), 223-243.

Baker, M. (1999). The role of corpora in investigating the linguistic behaviour of professional translators. *International Journal of Corpus Linguistics, 4*(2), 281-298.

Baker, P. (2010). *Sociolinguistics and corpus linguistics.* Edimburgo: Edinburgh University Press.

Baker, P., Hardie, A., & McEnery, T. (2006). *A glossary of corpus linguistics.* Edimburgo: Edinburgh University Press.

Baker, P., McEnery, T., Leisher, M., Cunningham, H., & Gaizauskas, R. (2000). Mapping multiple South Asian 8-bit character sets to the Unicode Standard. *Linguistic Exploration: Workshop on Web-Based Language Documentation and Description.* Philadelphia: Institute for Research on Cognitive Science, University of Pennsylvania.

Bamman, D., Eisenstein, J., & Schnoebelen, T. (2014). Gender identity and lexical variation in social media. *Journal of Sociolinguistics, 18*(2), 135-160.

Banco de España. (2015). *Informe Anual*. Obtenido de http://www.bde.es/bde/es/secciones/informes/Publicaciones_an/Informe_anual/

Banco de España. (2016a). *Boletín Económico del BCE.* Obtenido de http://www.bde.es/bde/es/secciones/informes/Publicaciones_de/boletin-economic/index2016.html

Banco de España. (2016b). *ECB Economic Bulletin.* Obtenido de http://www.bde.es/bde/en/secciones/informes/Publicaciones_de/boletin-economic/index2016.html

Bangor, A., Kortum, P., & Miller, J. (2009). Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of usability studies, 4*(3), 114-123.

Bangor, A., Miller, J., & Kortum, T. (2008). An empirical evaluation of the system usability scale. *International Journal of Human-Computer Interaction, 24*(6), 576-594.

Bank, M., & Schierle, M. (2012). A Survey of Text Mining Architectures and the UIMA Standard. *Proceedings of the Language Resources and Evaluation Conference (LREC 2012),* (págs. 3479-3486). Estambul: European Language Resources Association (ELRA).

Barlow, M. (1995). ParaConc Multilingual Concordancer (Versión 1.0.0) [Software]. Houston: Athelstan. Obtenido de http://www.paraconc.com/

Barlow, M. (2003). MonoConc Pro 2.2 [Software]. Houston: Athelstan. Obtenido de http://www.athel.com/mono.html

Baroni, M., Kilgarriff, A., Pomikalek, J., & Rychlý, P. (2006). WebBootCaT: a web tool for instant corpora. *Atti del XII Congresso Internazionale di Lessicografia*, (págs. 121-130). Turín: Edizioni Dell'Orso Srl.

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics, 37*(6), 1554-1563.

Biber, D. (1991). *Variation across speech and writing.* Cambridge: Cambridge University Press.

Biber, D. (1993). Representativeness in corpus design. *Literary and linguistic computing, 8*(4), 243-257.

Biber, D., Connor, U., & Upton, T. (2007). *Discourse on the move.* Ámsterdam: John Benjamins.

Biber, D., Conrad, S., & Reppen, R. (1998). *Corpus Linguistics.* Cambridge: Cambridge University Press.

Biber, D., Johansson, S., Leech, G., Conrad, S., & Finegan, E. (1999). *Longman Grammar of Spoken and Written English.* Harlow: Longman.

Bird, S., Klein, E., & Loper, E. (2008). *Natural Language Processing Version: 0.9.5* . Obtenido de http://www.ling.helsinki.fi/kit/2008s/clt231/nltk-0.9.5/doc/en/book.html

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit.* Sebastopol: O'Reilly Media, Inc.

Blaette, A. (2017). R-package 'polmineR' (Versión 0.7.1) [Software]. GitHub. Obtenido de https://github.com/PolMine/polmineR

BNC Consortium. (2007). *The British National Corpus, version 3 (BNC XML Edition).* Distributed by Oxford University Computing Services on behalf of the BNC Consortium. Obtenido de http://www.natcorp.ox.ac.uk/

Bohnet, B. (2010). Very High Accuracy and Fast Dependency Parsing is not a Contradiction. *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)* (págs. 81-89). Pekín: Coling 2010 Organizing Committee.

Braune, F., & Fraser, A. (2010). Improved unsupervised sentence alignment for symmetrical and asymmetrical parallel corpora. *Proceedings of the 23rd International Conference on Computational Linguistics: Posters (Coling 2010)* (págs. 81-89). Pekín: Coling 2010 Organizing Committee.

Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., & Yergeau, F. (1998). *Extensible markup language (XML).* World Wide Web Consortium Recommendation REC-xml-19980210. Obtenido de https://www.w3.org/TR/1998/REC-xml-19980210

Brezina, V., McEnery, T., & Wattam, T. (2015). Collocations in context: A new perspective on collocation networks. *International Journal of Corpus Linguistics, 20*(2), 139-173.

Brooke, J. (1996). SUS - A quick and dirty usability scale. En P. Jordan, B. Thomas, I. McClelland, & B. Weerdmeester (Eds.), *Usability evaluation in industry* (págs. 189-194). Londres: Taylor & Francis.

Burnard, L., & Todd, T. (2010). XAIRA - XML Aware Indexing and Retrieval Architecture (Versión 1.6.0) [Software]. Obtenido de http://xaira.sourceforge.net/

Busa, R. (1974). *Index Thomisticus*. Stuttgart: Frommann-Holzboog.

Cake Software Foundation, Inc. (2017). CakePHP (Versión 3.4 Red Velvet) [Software]. Obtenido de https://cakephp.org/

Calzolari, N., Choukri, K., Fellbaum, C., Hovy, E., & Ide, N. (1999). Multilingual Resources. En M. Palmer (Ed.), *Multilingual Information Management: Current Levels and Future Abilities*. Obtenido de http://www.cs.cmu.edu/~ref/mlim//chapter1.html

Cambridge University Press. (2017). *Cambridge Dictionaries Online*. Obtenido de http://dictionary.cambridge.org/

Carbonell, J. (2010). Intelligence Resource Collection for Low-Density Languages. *Keynote speech of Seventh International Conference on Language Resources and Evaluation (LREC).* La Valeta. Obtenido de http://www.lrec-conf.org/proceedings/lrec2010/keynotes/keynote1.pdf

Carrera-de la Red, M. (1998). *Textos lingüísticos antiguos del romance hispánico.* Obtenido de http://www.vallenajerilla.com/berceo/carreradelared/textosantiguosromanchehispanico.htm

Centre for Translation Studies. (2016). *IntelliText User Guide.* Obtenido de http://smlc12.leeds.ac.uk/itweb/help/IntelliText_User_Guide.pdf

Cer, D., De Marneffe, M., Jurafsky, D., & Manning, C. (2010). Parsing to Stanford Dependencies: Trade-offs between Speed and Accuracy. *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)* (págs. 19-21). La Valeta: European Language Resources Association (ELRA).

Chan, S. W. (2014). *Routledge Encyclopedia of Translation Technology.* Londres: Routledge.

Chandler, B., & Tribble, C. (1989). *Longman mini-concordancer*. Harlow: Longman.

Chanod, J., Hobbs, J., Hovy, E., Jelinek, F., & Rajman, M. (1999). Methods and Techniques of Processing. En N. Ide (Ed.), *Multilingual Information Management: Current Levels and Future Abilities*. Obtenido de http://www.cs.cmu.edu/~ref/mlim//chapter6.html

Chen, D., & Manning, C. (2014). A Fast and Accurate Dependency Parser using Neural Networks. *Conference on Empirical Methods on Natural Language*

Processing (EMNLP 2014), (págs. 740-750). Doha: The Association for Computational Linguistics.

Cheng, W., Greaves, C., & Warren, M. (2006). From n-gram to skipgram to concgram. *International journal of corpus linguistics, 11*(4), 411-433.

Chomsky, N. (1957). *Syntactic Structures.* Mouton: The Hague.

Chomsky, N. (1962). The Logical Basis of Linguistic Theory. En H. Lunt (Ed.), *Proceedings of the 9th International Congress of Linguists* (págs. 914-1008). Cambridge: Mouton.

Chopra, V., Li, S., & Genender, J. (2007). *Professional Apache Tomcat 6.* Hoboken: John Wiley & Sons.

Christ, O. (1994). A modular and flexible architecture for an integrated corpus query system. *Papers in Computational Lexicography (COMPLEX '94)*, (págs. 22-32). Budapest: Linguistics Institute, Hungarian Academy of Sciences.

Clark, R. (1966). *Computers and the Humanities*. *1*(3), 39.

Clarke, T. (2017). TableExport (Versión 5.0.0-rc.8) [Software]. GitHub. Obtenido de https://github.com/clarketm/TableExport

Cortes, V. (2013). "The purpose of this study is to": Connecting lexical bundles and moves in research article introductions. *Journal of English for Academic Purposes*, *12*(1), 33-43.

Cunningham, H.; Maynard, D.; Bontcheva, K.; Tablan, V.; Aswani, N.; Roberts, I.; Gorrell, G.; Funk, A.; Roberts, A.; Damljanovic, D.; Heitz, T.; Greenwood, M.A.; Saggion, H.; Petrak, J.; Li, Y.; Peters, W. (2011). *Text Processing with GATE (Version 6)*. Murphys: Gateway Press CA. Obtenido de http://tinyurl.com/gatebook

Dale, R. (2010). Classical Approaches to Natural Language Processing. En N. Indurkhya, & F. J. Damerau (Eds.), *Handbook of natural language processing* (págs. 3-8). Boca Raton: CRC Press.

Date, C., & Darwen, H. (1993). *A guide to the SQL Standard: a user's guide to the standard relational language SQL*. Boston: Addison-Wesley Longman.

Davies, M. (2008). *The Corpus of Contemporary American English (COCA): 520 million words, 1990-present*. Obtenido de http://corpus.byu.edu/coca/

Davies, M. (2017a). *corpus.byu.edu*. Obtenido de http://corpus.byu.edu/

Davies, M. (2017b). corpus.byu.edu - FAQ/questions. Obtenido de http://corpus.byu.edu/faq.asp

Dearing, V. (1966). *Computers and the Humanities*. *1*(3), 39-40.

Desgraupes, B., & Loiseau, S. (2016). R-Package 'rcqp' (Versión 0.4) [Software]. Obtenido de https://cran.r-project.org/web/packages/rcqp/rcqp.pdf

Down, T. (2017). Rangy Inputs (Versión 1.2.0) [Software]. GitHub. Obtenido de https://github.com/timdown/rangyinputs

Du Bois, J. W., Chafe, W. L., Meyer, C., Thompson, S. A., & Martley, N. (2016). *Santa Barbara Corpus of Spoken American English (SBCSAE)*. Obtenido de http://www.linguistics.ucsb.edu/research/santa-barbara-corpus#access

Dyvik, H. (1998). A Translational Basis for Semantics. En S. Johansson, & S. Oksefjell (Eds.), *Corpora and Crosslinguistic Research: Theory, Method and Case Studies* (págs. 51-86). Ámsterdam & Atlanta: Rodopi.

EAGLES. (1996). *Recommendations for morphosyntactic categories.* Obtenido de EAGLES: http://www.ilc.cnr.it/EAGLES96/annotate/node9.html

ECMA International. (2011). *Standard ECMA-262 ECMAScript Language Specification.* Obtenido de http://www.ecma-international.org/ecma-262/5.1/Ecma-262.pdf

ECMA International. (2013). *Standard ECMA-404 The JSON Data Interchange Format.* Obtenido de http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

Emory NLP. (2017). MLP4J - NLP Toolkit for JVM Languages [Software]. Obtenido de https://emorynlp.github.io/nlp4j/

Evert, S. (2009). *The CQP Query Language Tutorial.* Obtenido de http://cwb.sourceforge.net/files/CQP_Tutorial.pdf

Evert, S. (2014). *[CWB] A question about the aligning using cwb-encoding (CWB mailing list).* Obtenido de http://liste.sslmit.unibo.it/pipermail/cwb/2014-January/001529.html

Evert, S. (2016a). *CQP Query Language Tutorial (CWB Version 3.4).* Obtenido de http://cwb.sourceforge.net/files/CQP_Tutorial.pdf

Evert, S. (2016b). *The IMS Open Corpus Workbench (CWB) - Corpus Encoding Tutorial.* Obtenido de http://cwb.sourceforge.net/files/CWB_Encoding_Tutorial.pdf

Evert, S., & Hardie, A. (2011). Twenty-first century Corpus Workbench: Updating a query architecture for the new millennium. *Proceedings of the Corpus Linguistics 2011 Conference.* Birmingham: University of Birmingham.

Farkas, A. (2010). LF Aligner (Versión 4.1) [Software]. Obtenido de https://sourceforge.net/projects/aligner/

Fielding, R. (2000). *Architectural Styles and the Design of Network-based Software Architectures* [Tesis Doctoral]. Irvine: University of California. Obtenido de http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Fowler, M. (2006). *GUI Architectures.* Obtenido de https://martinfowler.com/eaaDev/uiArchs.html

Free Software Foundation. (2007). *GNU General Public License - Version 3.* Obtenido de http://www.gnu.org/copyleft/gpl.html

Free Software Foundation. (2017). GNU Bash [Software]. Obtenido de https://www.gnu.org/software/bash/

Gale, W. A., & Church, K. W. (1993). A program for aligning sentences in bilingual corpora. *Computational linguistics, 1,* 75-102.

Garabik, R., Zimmer, K., Jurish, B., & Sokirko, A. (2016). DWDS/Dialing Concordance (DDC) [Software]. Obtenido de http://www.ddc-concordance.org/

Garside, R. (1987). The CLAWS Word-tagging System. En R. Garside, G. Leech, & G. Samspon (Eds.), *The Computational Analysis of English: A Corpus-based Approach.* Londres: Longman. Obtenido de http://ucrel.lancs.ac.uk/papers/ClawsWordTaggingSystemRG87.pdf

Geyken, A., Didakowski, J., & Siebert, A. (2008). Generation of Word Profiles on the Basis of a Large and Balanced German Corpus. En E. Bernal, & J. DeCesaris (Eds.), *Proceedings of the 13th EURALEX International Congress* (págs. 371-383). Barcelona: Institut Universitari de Linguistica Aplicada, Universitat Pompeu Fabra. Obtenido de http://euralex.org/publications/generation-of-word-profiles-on-the-basis-of-a-large-and-balanced-german-corpus/

Giesbrecht, E., & Evert, S. (2009). Is part-of-speech tagging a solved task? An evaluation of POS taggers for the German web as corpus. *Proceedings of the fifth Web as Corpus workshop*, (págs. 27-35). Donostia-San Sebastian: Elhuyar Fundazioa.

Gilquin, G. (2015). At the interface of contact linguistics and second language acquisition research: New Englishes and Learner Englishes compared. *English World-Wide, 36*(1), 91-124.

GitHub Inc. (2017). *GitHub*. Obtenido de https://github.com/

Google Inc. (2017a). Angular (Versión 1.6.2) [Software]. Obtenido de https://angular.io/

Google Inc. (2017b). *Traductor de Google*. Obtenido de https://translate.google.com/

Gosling, J., Joy, B., Steele, G., Bracha, G., & Buckley, A. (2014). *The Java® Language Specification (Java SE 8 ed.).* Obtenido de https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf

Gossman, J. (2005). *Introduction to Model/View/ViewModel pattern for building WPF apps.* Obtenido de https://blogs.msdn.microsoft.com/johngossman/2005/10/08/introduction-to-modelviewviewmodel-pattern-for-building-wpf-apps/

Granger, S. (1996). From CA to CIA and back: An integrated contrastive approach to bilingual and learner computerized corpora. *Languages in contrast: Text-based cross-linguistic studies*, 37-51.

Gries, S. (2008). Phraseology and linguistic theory. En S. Granger, & F. Meunier (Eds.), *An Interdisciplinary Perspective* (págs. 3-26). Ámsterdam & Philadelphia: John Benjamins.

Hall, D. (2014). ScalaNLP Project [Software]. GitHub. Obtenido de https://github.com/scalanlp/breeze/wiki/ScalaNLP-Project

Hammarström, H. (2009). *A survey of computational morphological resources for low-density languages.* Obtenido de http://www.csee.ogi.edu/~sproatr/Courses/Linguistics/Readings/hammarstrom.pdf

Hardie, A. (2012). CQPweb—combining power, flexibility and usability in a corpus analysis tool. *International Journal of Corpus Linguistics, 17*(3), 380-409.

Hardie, A. (2016). *CQPweb at Lancaster*. Obtenido de https://cqpweb.lancs.ac.uk/

Hardie, A., Baroni, M., Evert, S., & Sharoff, S. (2016). cqp.inc.php [Software]. Obtenido de https://sourceforge.net/p/cwb/code/HEAD/tree/gui/cqpweb/trunk/lib/cqp.inc.php

Hipp, R. (2017). SQLite [Software]. Obtenido de https://www.sqlite.org/index.html

Hockey, S. (1993). *Micro-OCP.* Oxford: Oxford University Press.

Hoffmann, S., Evert, S., Smith, N., Lee, D., & Berglund-Prytz, Y. (2008). *Corpus linguistics with BNCweb-a practical guide* (Vol. 6). Frankfurt: Peter Lang.

Hofland, K., & Johansson, S. (1998). The Translation Corpus Aligner: A program for automatic alignment of parallel texts. *Language and Computers*, 87-100.

Hopkins, C. (2013). *The MVC Pattern and PHP (The MVC Pattern and PHP, Part 1)*. Obtenido de: https://www.sitepoint.com/the-mvc-pattern-and-php-1/

Hunston, S. (2002). *Corpora in applied linguistics.* Stuttgart: Ernst Klett Sprachen.

IDS. (2017). *The IDS-Text Model.* Obtenido de Institut für Deutsche Sprache: http://www1.ids-mannheim.de/direktion/kl/projekte/korpora/textmodell.html?L=1

Ivaska, I. (2014). The Corpus of Advanced Learner Finnish (LAS2): Database and toolkit to study academic learner Finnish. *Apples: journal of applied language studies, 8*(3), 21-38. Obtenido de http://apples.jyu.fi

Izquierdo, M., Hofland, K., & Reigem, Ø. (2008). The ACTRES parallel corpus: an English-Spanish translation corpus. *Corpora, 3*(1), 31-41.

Jacyntho, M., Schwabe, D., & Rossi, F. (2002). A software architecture for structuring complex web applications. *Journal of Web Engineering, 1*(1), 37-60.

Jiménez-Yáñez, R., Sanjurjo-González, H., Rayson, P., & Piao, S. (próximamente). Building a Spanish lexicon for corpus analysis. *XXXV Congreso*

Internacional de la Asociación Española de Lingüística Aplicada (AESLA 2017). Santiago de Compostela.

Johansson, S. (1998). On the role of corpora in cross-linguistic research. *Corpora and Cross-linguistic Research: Theory, Method and Case Studies, 24*, 3-24.

Johansson, S., & Hofland, K. (1994). Towards an English-Norwegian parallel corpus. En U. Fries, G. Tottie, & P. Schneider (Eds.), *Creating and Using English Language Corpora* (págs. 25-37). Zúrich: Brill.

Kaye, K. (1990). A corpus-builder and real time concordancer browser for IBM-PC. En J. Aarts, & W. Meijs (Eds.), *Theory and Practice in Corpus Linguistics* (págs. 137-162). Ámsterdam: Rodopi.

Kennedy, G. (1998). *An introduction to Corpus Linguistics.* Londres: Longman.

Kenny, D. (2014). *Lexis and Creativity in Translation: A Corpus Based Approach.* Londres & Nueva York: Routledge.

Kilgarriff, A. (2001). Comparing Corpora. *International Journal of Corpus Linguistics, 6*(1), 97-133.

Kilgarriff, A., Baisa, V., Bušta, J., Jakubíček, M., Kovář, V., Michelfeit, J., Rychlý, P., & Suchomel, V. (2014). The Sketch Engine: ten years on. *Lexicography, 1*(1), 7-36.

Knowles, G., Wichmann, A., & Alderson, P. (1996). *Working with Speech: perspectives on research into the Lancaster/IBM Spoken English Corpus.* Reading: Addison-Wesley Longman.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. *Proceedings of Machine Translation Summit X, 5*, págs. 79-86. Phuket: Asia-Pacific Association for Machine Translation.

Kučera, H., & Francis, W. N. (1967). *Computational Analysis of Present-day American English.* Providence: Brown University Press.

Kudo, T. (2005). Mecab: Yet another part-of-speech and morphological analyzer [Software]. Obtenido de https://sourceforge.net/projects/mecab/

Kupietz, M., Belica, C., Keibel, H., & Witt, A. (2010). The German Reference Corpus DeReKo: A Primordial Sample for Linguistic Research. En C. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner & D. Tapias (Eds.), *Proceedings of Language Resources and*

Evaluation Conference (LREC 2010) (págs. 1848-1854). La Valeta: European Language Resources Association (ELRA).

Labrador, N., Ramón, N., Alaiz-Moretón, H., & Sanjurjo-González, H. (2014). Rhetorical structure and persuasive language in the subgenre of online advertisements. *English for Specific Purposes*, *34*, 38-47.

Lavin, P. (2006). *OBJECT-ORIENTED PHP: concepts, techniques, and code.* San Francisco: No Starch Press.

Leech, G. (1991). The state of the art in corpus linguistics. En E. Aijmer, & B. Altenberg (Eds.), *English Corpus Linguistics* (págs. 8-29). Londres: Longman.

Leech, G. (1992). Corpora and theories of linguistic performance. En J. Svartik (Ed.), *Directions in corpus linguistics* (págs. 105-122). Berlín: Mouton de Gruyter.

Leech, G. (1997). Introducing corpus annotation. En R. Garside, G. Leech, & T. McEnery (Eds.), *Corpus annotation: Linguistic Information from Computer Text Corpora* (págs. 1-18). Londres & Nueva York: Routledge: Taylor & Francis Group.

Liu, X., Zhang, S., Wei, F., & Zhou, M. (2011). Recognizing named entities in tweets. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (págs. 359-367). Oregon: The Association for Computational Linguistics.

Lu, X. (2014). *Computational Methods for Corpus Annotation and Analysis.* Heidelberg: Springer.

Maia, B., & Matos, S. (2008). Corpógrafo V4 - Tools for Researchers and Teachers using Comparable Corpora. En P. Zweigenbaum, E. Gaussier, & P. Fung (Eds.), *LREC 2008 Workshop on Comparable Corpora* (págs. 79-82). Marrakech: European Language Resources Association (ELRA).

Manning, C. D. (2011). Part-of-speech tagging from 97% to 100%: is it time for some linguistics? *International Conference on Intelligent Text Processing and Computational Linguistics* (págs. 171-189). Tokio: Springer.

Martin, J. (1991). *Rapid application development.* Nueva York: Macmillan Publishing Co.

Matos, S., & Maia, B. (2008). *NooJ and Corpógrafo - A New Partnership*. Obtenido de http://www.linguateca.pt/documentos/MatosMaiaNooJ08Slides.pdf

Maurer, F., & Martel, S. (2002). Extreme programming. Rapid development for Web-based applications. *IEEE Internet computing*, 86-90.

McArthur, T. (1981). *Longman lexicon of contemporary English.* Reading: Addison-Wesley Longman.

McEnery, T., & Hardie, A. (2012). *Corpus Linguistics: Method, Theory and Practice*. Cambridge: Cambridge University Press.

McEnery, T., & Wilson, A. (2001). *Corpus Linguistics (2ª ed.).* Edimburgo: Edinburgh University Press.

McEnery, T., & Xiao, R. (2007a). Parallel and comparable corpora: The state of play. En Y. Kawaguchi, T. Takagaki, N. Tomimori, & Y. Tsuruga (Eds.), *Corpus-based perspectives in linguistics* (págs. 131-145). Ámsterdam & Philadelphia: John Benjamins.

McEnery, T., & Xiao, R. (2007b). Parallel and comparable corpora: What is happening? En *Incorporating Corpora. The Linguist and the Translator* (págs. 18-31). Clevedon: Multilingual Matters.

McEnery, T., Xiao, R., & Mo, L. (2003). Aspect marking in English and Chinese: using the Lancaster Corpus of Mandarin Chinese for contrastive language study. *Literary and Linguistic Computing*, *18*(4), 361-378.

McEnery, T., Xiao, R., & Tono, Y. (2006). *Corpus-based language studies: An advanced resource book.* Londres & Nueva York: Taylor & Francis.

Meno, M. (2012). DropzoneJS (Versión 5.0.0) [Software]. Obtenido de http://www.dropzonejs.com/

Meurer, P. (2012). Corpuscle – a new corpus management platform for annotated corpora. En G. Andersen (Ed.), *Exploring Newspaper Language: Using the Web to Create and Investigate a large corpus of modern Norwegian* (págs. 31-50). Ámsterdam: John Benjamins.

Meurer, P. (2017). *Corpuscle Home Page*. Obtenido de http://clarino.uib.no/korpuskel/page

Meyers, A. (2009). Compatibility between corpus annotation efforts and its effects on Computational Linguistics. En P. Baker (Ed.), *Contemporary Corpus*

Linguistics (Vol. 16, págs. 105-124). Londres: Continuum International Publishing Group.

Nadkarni, P. M., Ohno-Machado, L., & Chapman, W. W. (2011). Natural language processing: an introduction. *Journal of the American Medical Informatics Association, 15*(5), 544-551.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR), 41*, 1-69.

Navigli, R., Faralli, S., Soroa, A., de Lacalle, O., & Agirre, E. (2011). Two birds with one stone: learning semantic models for text categorization and word sense disambiguation. *Proceedings of the 20th ACM international conference on Information and knowledge management (CIKM - 2011)* (págs. 2317-2320). Glasgow: Association for Computing Machinery.

NLTK Project. (2017). *nltk.tokenize package (NLTK 3.0 documentation)*. Obtenido de http://www.nltk.org/api/nltk.tokenize.html?highlight=treebankwordtokenizer#nltk.tokenize.treebank.TreebankWordTokenizer

npm Inc. (2017). *npm*. Obtenido de https://www.npmjs.com/

Och, F. J., & Ney, H. (2003). A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics, 29*, 19-51.

O'Donnel, M. (2008). The UAM CorpusTool: Software for corpus annotation and exploration. *Applied Linguistics Now: Understanding Language and Mind*, 1433-1447. Obtenido de http://www.uam.es/proyectosinv/woslac/DOCUMENTS/Presentations%20and%20articles/ODonnellAESLA08.pdf

Oliveira, E. (2004). Towards a new authoring environment: overview of some ontology based systems. En J. Engelen (Ed.), *8ª International Conference on Electronic Publishing*, (págs. 121-130). Brasilia: ELPUB 2014 Proceedings.

Orfanou, K., Tselios, N., & Katsanos, C. (2015). Perceived usability evaluation of learning management systems: Empirical evaluation of the System Usability Scale. *he International Review of Research in Open and Distributed Learning, 16*(2), 227-246.

Otto, M., & Thornton, J. (2016). Bootstrap (Versión 3.3.7) [Software]. Obtenido de http://getbootstrap.com/

OWASP. (2013). *Code Injection - Open Web Application Security Project*. Obtenido de https://www.owasp.org/index.php/Code_Injection

Padró, L., & Stanilovsky, E. (2012). FreeLing 3.0: Towards Wider Multilinguality. *Proceedings of the Language Resources and Evaluation Conference (LREC 2012).* Estambul: European Language Resources Association (ELRA).

Pak, A., & Paroubek, P. (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC 2010)* (págs. 19-21). La Valeta: European Language Resources Association (ELRA).

Paroubek, P. (2007). Evaluating Part-of-Speech Tagging and Parsing: On the Evaluation of Automatic Parsing of Natural Language. En L. Dybkjær, H. Hemsen, & W. Minker (Eds.), *Evaluation of Text and Speech Systems* (págs. 99-124). Dordrecht: Springer Netherlands.

Petasis, G. (2014). *What is Ellogon?*. Obtenido de http://www.ellogon.org/index.php/what-is-ellogon

Petasis, G., Karkaletsis, V., Paliouras, G., Androutsopoulos , I., & Spyropoulos, C. (2002). Ellogon: A New Text Engineering Platform. *Proceedings of the Language Resources and Evaluation Conference (LREC 2002)*, (págs. 72-78). Las Palmas de Gran Canaria: European Language Resources Association (ELRA).

Petrov, S., Das, D., & McDonald, R. (2011). A universal part-of-speech tagset. *arXiv:*1104.2086.

Phillips, M. (1989). *Lexical structure of text [Discourse Analysis Monograph 12]*. Birmingham: University of Birmingham.

Piao, S., Bianchi, F., Dayrell, C., D'Egidio, A., & Rayson, P. (2015). Development of the multilingual semantic annotation system. *2015 Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT 2015)*, (págs. 1268-1274). Denver: The Association for Computational Linguistics.

Pivotal Software. (2017). Spring Framework [Software]. Obtenido de https://projects.spring.io/spring-framework/

Platt, D. (2002). *Introducing Microsoft. Net.* Redmond: Microsoft press.

Princeton University. (2010). *Princeton University "About WordNet."*. Obtenido de: https://wordnet.princeton.edu/

Python Software Foundation. (2017a). *PyPI - the Python Package Index*. Obtenido de https://pypi.python.org/pypi

Python Software Foundation. (2017b). Python (Versión 3.5.2) [Software]. Obtenido de https://www.python.org/

QSR International Pty Ltd. (2017). NVIVO [Software]. Obtenido de http://www.qsrinternational.com/what-is-nvivo

Quinlan, J. R. (2014). *C4. 5: programs for machine learning.* Ámsterdam: Elsevier.

Quirk, R., Greenbaum, S., Leech, G., & Svartvik, J. (1972). *A Grammar of Contemporary English.* Londres: Longman.

R Core Team. (2017). R: A language and environment for statistical (R Foundation for Statistical Computing) (Versión 3.4.0) [Software] . Obtenido de http://www.R-project.org/

Rayson, P. (2003). *Matrix: A statistical method and software tool for linguistic analysis through corpus comparison* [Tesis Doctoral]. Lancaster: Lancaster University. Obtenido de http://ucrel.lancs.ac.uk/pics/pdficon.gif

Rayson, P. (2009). Wmatrix: a web-based corpus processing environment [Software]. Lancaster: Computing Department, Lancaster University. Obtenido de http://ucrel.lancs.ac.uk/wmatrix/

Real Academia Española. (2017a). *Banco de datos (CORPES XXI) [en línea]. Corpus del Español del Siglo XXI*. Obtenido de http://www.rae.es/recursos/banco-de-datos/corpes-xxi

Real Academia Española. (2017b). *Banco de datos (CREA) [en línea]. Corpus de referencia del español actual.* Obtenido de http://corpus.rae.es/creanet.html

Real Academia Española. (2017c). *Diccionario de la lengua española - Real Academia Española*. Obtenido de http://dle.rae.es/

Reed, A. (1977). CLOC: A Collocation Package. *Association for Literary and Linguistic Computing Bulletin, 5*(2), 168-173.

Reenskaug, T. (1979). *The original MVC reports.* Obtenido de http://folk.uio.no/trygver/2007/MVC_Originals.pdf

Refaee, E., & Rieser, V. (2014). An Arabic Twitter Corpus for Subjectivity and Sentiment Analysis. En N. Calzolari, K. Choukri , T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, & S. Piperidis (Eds.), *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC 2014)* (págs. 2268-2273). Reykjavik: European Language Resources Association (ELRA).

Renkema, J. (2004). *Introduction to discourse studies.* Ámsterdam & Philadelphia: John Benjamins.

Rychlý, P. (2000). *Corpus Managers and their effective implementation* [Tesis doctoral]. Brno: Masaryk University. Obtenido de http://www.fi.muni.cz/~pary/dis.pdf

Rychlý, P. (2007). Manatee/Bonito - A Modular Corpus Manager. *1st Workshop on Recent Advances in Slavonic Natural Language Processing* (págs. 65-70). Brno: Masaryk University.

Saber, A. (2012). Phraseological patterns in a large corpus of biomedical articles. En A. Boulton, S. Carter-Thomas, & E. Rowley-Jolivet (Eds.), *Corpus-Informed Research and Learning in ESP: Issues and applications* (págs. 45-82). Ámsterdam & Philadelphia: John Benjamins.

Salazar, D. (2014). *Lexical Bundles in Native and Non-native Scientific Writing.* Ámsterdam & Philadelphia: John Benjamins.

Sánchez-Martínez, F., Forcada, M. L., Martin, R., Ortiz, S., Santos, S., Martineau, V., & Evangelou, Y. (2014). bitext2tmx CAT bitext aligner/converter [Software]. Sourceforge. Obtenido de https://sourceforge.net/projects/bitext2tmx/

Sanjurjo-González, H., & Izquierdo, M. (próximamente). The ACTRES Parallel Corpus (P-ACTRES 2.0). *Parallel Corpora: Creation and Applications International Symposium (PaCor 2016).* Santiago de Compostela.

Sarmento, L., Maia, B., & Santos, D. (2004). The Corpógrafo - a Web-based environment for corpora research. En M. Lino, M. Xavier, F. Ferreira, R. Costa, & R. Silva (Eds.), *4th International Conference on Language Resources and Evaluation (LREC 2004)*, (págs. 449-452). Lisboa: European Language Resources Association (ELRA).

Sarmento, L., Maia, B., Santos, D., Pinto, A., & Cabral, L. (2006). Corpógrafo V3: From Terminological Aid to Semi-automatic Knowledge Engine. En N. Calzolari, A. Gangemi, B. Maegaard, J. Mariani, J. Odjik, & D. Tapias

(Eds.), *5th International Conference on Language Resources and Evaluation (LREC'2006)*, (págs. 1502-1505). Génova: European Language Resources Association (ELRA).

Schmid, H. (1995). Treetagger | a language independent part-of-speech tagger. *Institut für Maschinelle Sprachverarbeitung, 43*, 28.

Scott, M. (1996). *WordSmith Tools.* Oxford: Oxford University Press.

Scott, M. (2012). WordSmith Tools (Versión 6) [Software]. Stroud: Lexical Analysis Software. Obtenido de http://www.lexically.net/wordsmith/

Scott, M., & Johns, T. (1993). MicroConcord [Software]. Obtenido de http://lexically.net/software/index.htm

Scott, M., & Tribble, C. (2006). *Textual Patterns: Keyword and Corpus Analysis in Language Education.* Ámsterdam & Philadelphia: John Benjamins.

Sennrich, R., & Volk, M. (2010). MT-based Sentence Alignment for OCR-generated Parallel Texts. *Proceedings of Association for Machine Translation in the Americas (AMTA 2010).* Denver: AMTA.

SensioLabs. (2017). Symfony [Software]. Obtenido de https://symfony.com/

Sharoff, S. (2006). A uniform interface to large-scale linguistic resources. *Fifth Language Resources and Evaluation Conference (LREC 2006)*, (págs. 539-542). Génova: European Language Resources Association (ELRA).

Sharoff, S., Rapp, R., Zweigenbaum, P., & Fung, P. (Eds.). (2013). *Building and Using Comparable Corpora*. Heidelberg: Springer.

Sharpened Productions. (2017). *The Tech Terms Computer Dictionary*. Obtenido de https://techterms.com/definition/api

Silberztein, M. (2005). NooJ: a linguistic annotation system for corpus processing. *Proceedings of HLT/EMNLP on Interactive Demonstrations* (págs. 10-11). Vancouver: The Association for Computational Linguistics.

Silberztein, M. (2017). NooJ: A Linguistic Development Environment [Software]. Obtenido de http://www.nooj-association.org/

Simov, K., Peev, Z., Kouylekov, M., Simov, A., Dimitrov, M., & Kiryakov, A. (2001). CLaRK - an XML-based System for Corpora Development. *Proceedings of the Corpus Linguistics 2001 Conference*, (págs. 558-560).

Lancaster: UCREL. Obtenido de http://www.bultreebank.org/papers/clark1.pdf

Sinclair, J. (2005). Corpus and Text: Basic Principles. En M. Wynne (Ed.), *Developing Linguistic Corpora: a Guide to Good Practice* (págs. 1-16). Oxford: Oxbow Books.

Sinclair, J., & Ball, J. (1996). *Preliminary recommendations on text typology - EAGLES Document EAG-TCWGTTYP/P*. Expert Advisory Group on Language Engineering Standards (EAGLES). Obtenido de http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=1C86E2BB12ABEAF9539BE818DCFAAAD2?doi=10.1.1.28.1988&rep=rep1&type=pdf

Smith, P. (1966). *Computers and the Humanities*. *1*(2), 39.

Soanes, C., & Stevenson, A. (2005). *Oxford Dictionary of English.* (C. Soanes, & A. Stevenson, Eds.) Oxford: Oxford University Press.

Sommerville, I. (2011). *Software engineering* (9th ed.). Reading: Addison-Wesley Longman.

Sprenger, S. (2003). Fixed expressions and the production of idioms. Nijmegen: Radboud University.

Steele, G. (1990). *Common LISP: the language (2ª ed).* Oxford: Butterworth-Heinemann.

Stellman, A., & Greene, J. (2005). *Applied software project management.* Sebastopol: O'Reilly Media, Inc.

Stroustrup, B. (1995). *The C++ programming language.* Noida: Pearson Education India.

Stubbs, M. (2002). Two quantitative methods of studying phraseology in English. *International Journal of Corpus Linguistics, 7*(2), 215-244.

Swales, J. (1990). *Genre analysis: English in academic and research settings.* Cambridge: Cambridge University Press.

Swales, J. (2004). *Research genres. Explorations and applications.* Cambridge: Cambridge University Press.

TALP-UPC. (2016). *FreeLing User Manual.* Obtenido de http://nlp.lsi.upc.edu/freeling/node/9

Taylor, C. (2008). What is corpus linguistics? What the data says. *ICAME journal, 32*, 179-200.

The Apache Software Foundation. (2016). Apache Struts [Software]. Obtenido de https://struts.apache.org/

The Apache Software Foundation. (2017). Apache OpenNLP [Software]. Obtenido de http://opennlp.apache.org/

The JQuery Foundation. (2017). JQuery (Versión 3.2.1) [Software]. Obtenido de https://jquery.com/

The PHP Group. (2017). PHP: Hypertext Preprocessor (Versión 7.0) [Software]. Obtenido de http://php.net/

The Qt Company. (2017). Qt [Software]. Obtenido de https://www.qt.io/

The Unicode Consortium. (2006). *The Unicode Standard, Version 5.0.* Obtenido de http://www.unicode.org/versions/Unicode5.0.0/

Thomson Reuters. (2017). Open Calais [Software]. Obtenido de http://www.opencalais.com/

Tian, Y., & Lo, D. (2015). A comparative study on the effectiveness of part-of-speech tagging techniques on bug reports. *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)* (págs. 570-574). Québec: IEEE.

Tiedemann, J. (2003). *Recycling Translations -- Extraction of Lexical Data from Parallel Corpora and their Application in Natural Language Processing* [Tesis Doctoral]. Uppsala: Uppsala University. Obtenido de http://uu.diva-portal.org/smash/record.jsf?pid=diva2:163715

Tiedemann, J. (2011). *Bitext alignment* (Vol. 4). (G. Hirst, Ed.) San Rafael, California, USA: Morgan & Claypool (Synthesis Lectures on Human Language Technologies).

Tiedemann, J. (2013). Uplug corpus tools (Versión 0.3.8) [Software]. Uppsala. Obtenido de https://sourceforge.net/projects/uplug/

Toral, A., Poch, M., Pecina, P., & Thurmair, G. (2012). Efficiency-based evaluation of aligners for industrial applications. *Proceedings of the 16th Annual Conference of the European Association for Machine Translation: EAMT 2012* (págs. 28-30). Trento: European Association for Machine Translation.

Toutanova, K., Klein, D., Manning, C., & Singer, Y. (2003). Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. *HLT-NAACL*, (págs. 252-259). Edmonton: The Association for Computational Linguistics.

Tullis, T.S., & Stetson, J. N. (2004). A comparison of questionnaires for assessing website usability. *Proceedings of Usability Professional Association Conference (UPA 2004)*, (págs. 1-12). Minneapolis: UPA.

U.S. Department of Health & Human Services. (2017). *System Usability Scale (SUS)*. Obtenido de https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

UCREL. (2016). *Multilingual-USAS. Lexicons for the Multilingual UCREL Semantic Analysis System*. Obtenido de https://github.com/UCREL/Multilingual-USAS

Unión Europea. (2017). *EuroVoc*. Obtenido de http://eurovoc.europa.eu/

Universidad de León. (2013). Estructuras jerárquicas para la redacción de actas de reuniones en lengua inglesa. Asiento registral: 00/2013/2139. España.

University of Pennsylvania. (2017). *Linguistic Data Consortium - University of Pennsylvania*. Obtenido de Language Resources: https://www.ldc.upenn.edu/language-resources

UPV/EHU. (2013). *ZIO Corpus*. Obtenido de: http://www.ehu.eus/ehg/zio/

Valverde, M. (2011). An evaluation of part of speech tagging on written second language Spanish. *International Conference on Intelligent Text Processing and Computational Linguistics* (págs. 214-226). Tokio: Springer Berlin Heidelberg.

Varga, D. (2015). *The hunalign sentence aligner - partialAlign*. Obtenido de https://github.com/danielvarga/hunalign#partialalign

Varga, D., Németh, L., Halácsy, P., Kornai, A., Trón, V., & Nagy, V. (2005). Parallel corpora for medium density languages. *Proceedings of the Recent Advances in Natural Language Processing (RANLP 2005)*, (págs. 590-596). Borovets: RANLP.

Villayandre-Llamazares, M. (2008). Lingüística con corpus (I). *Estudios Humanísticos Filología, 30*, 329-349.

W3C. (2014). *HTML5 (A vocabulary and associated APIs for HTML and XHTML).* Obtenido de https://www.w3.org/TR/2014/REC-html5-20141028/

W3C. (2015). *CSS Snapshot 2015*. Obtenido de https://www.w3.org/TR/css3-roadmap/

Wall, L. (2017). The Perl Programming Language [Software]. Obtenido de https://www.perl.org/

Wang, Y., Guo, C., & Song, L. (2009). Architecture of E-Commerce Systems Based on J2EE and MVC Pattern. *Management of e-Commerce and e-Government (ICMECG'09)* (págs. 284-287). Nanchang: IEEE.

Wasow, T. (2002). *Postverbal behaviour.* Stanford: CSLI Publications.

Wattam, S. (2015). GraphColl (Versión 1.0.0) [Software]. Obtenido de http://www.extremetomato.com/projects/graphcoll/

Weisser, M. (2016). *Practical Corpus Linguistics: An Introduction to Corpus-Based Language Analysis.* Hoboken: Wiley-Blackwell.

Wikimedia Foundation. (2017). *Wiktionary*. Obtenido de https://www.wiktionary.org/

Wilson, J., Hartley, A., Sharoff, S., & Stephenson, P. (2010). Advanced Corpus Solutions for Humanities Researchers. *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation (PACLIC 24)* (págs. 769-778). Sendai: The PACLIC 24 Organizing Committee and PACLIC Steering Committee.

Winter, T. N. (1999). Roberto Busa, S.J., and the Invention of the Machine-Generated Concordance. *The Classical Bulletin, 75*(1), 3-20. Obtenido de http://digitalcommons.unl.edu/classicsfacpub/70/

Xiao, R. (2007). *Corpus Linguistics: the basics - Lancaster University (Session 2).* Obtenido de www.lancaster.ac.uk/fass/projects/corpus/ZJU/xpresentations/session%202.ppt

Xiao, R. (2010). Corpus Creation. En N. Indurkhya, & F. J. Damerau (Eds.), *Handbook of Natural Language Processing* (2ª ed., págs. 147-166). Boca Raton & Londres & Nueva York: CRC Press.

Zhang, L., & Rettinger, A. (2014). Semantic Annotation, Analysis and Comparison: A Multilingual and Cross-lingual Text Analytics Toolkit. *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics* (págs. 13-16). Gotemburgo: Association for Computational Linguistics.

Anexo

Anexo 1: Casos de uso

A continuación, se presentan los casos de uso y los diagramas diseñados empleando el lenguaje de modelado unificado UML 2.

Caso de uso "Creación de un corpus monolingüe"

Creación de un corpus monolingüe	
Precondiciones	Estar registrado y haber iniciado sesión en el sistema. Documentos del corpus a crear.
Postcondiciones	Confirmación de creación del corpus monolingüe.
Actores	Usuario y framework.
Descripción	Se trata de un proceso asistido por el framework: (1) El framework presenta las opciones disponibles (Framework). (2) El usuario selecciona la opción "Crear Corpus" y posteriormente "Crear corpus monolingüe" (Usuario). (3) El framework presenta los datos que ha de introducir el usuario (Framework). (4) El usuario introduce los datos pertinentes (Usuario): a. Nombre del corpus. b. Idioma del corpus. c. Abreviatura del corpus. (5) El framework presenta una vista para la carga de los documentos que formarán parte del corpus (Framework). (6) El usuario selecciona los documentos desde su equipo (Usuario). (7) El framework presenta los tipos de etiquetados a nivel de palabra disponibles (Framework). (8) El usuario selecciona los etiquetados que desea aplicar (Usuario). (9) El framework presenta toda la información para su confirmación por parte del usuario (Framework).

	(10) El usuario confirma la creación del corpus (Usuario).
Variaciones (escenarios secundarios)	No hay.
Excepciones	La vista controla todas las excepciones y avisa al usuario mediante mensajes por pantalla.

Tabla 61 - Descripción de caso de uso "Crear un corpus monolingüe"

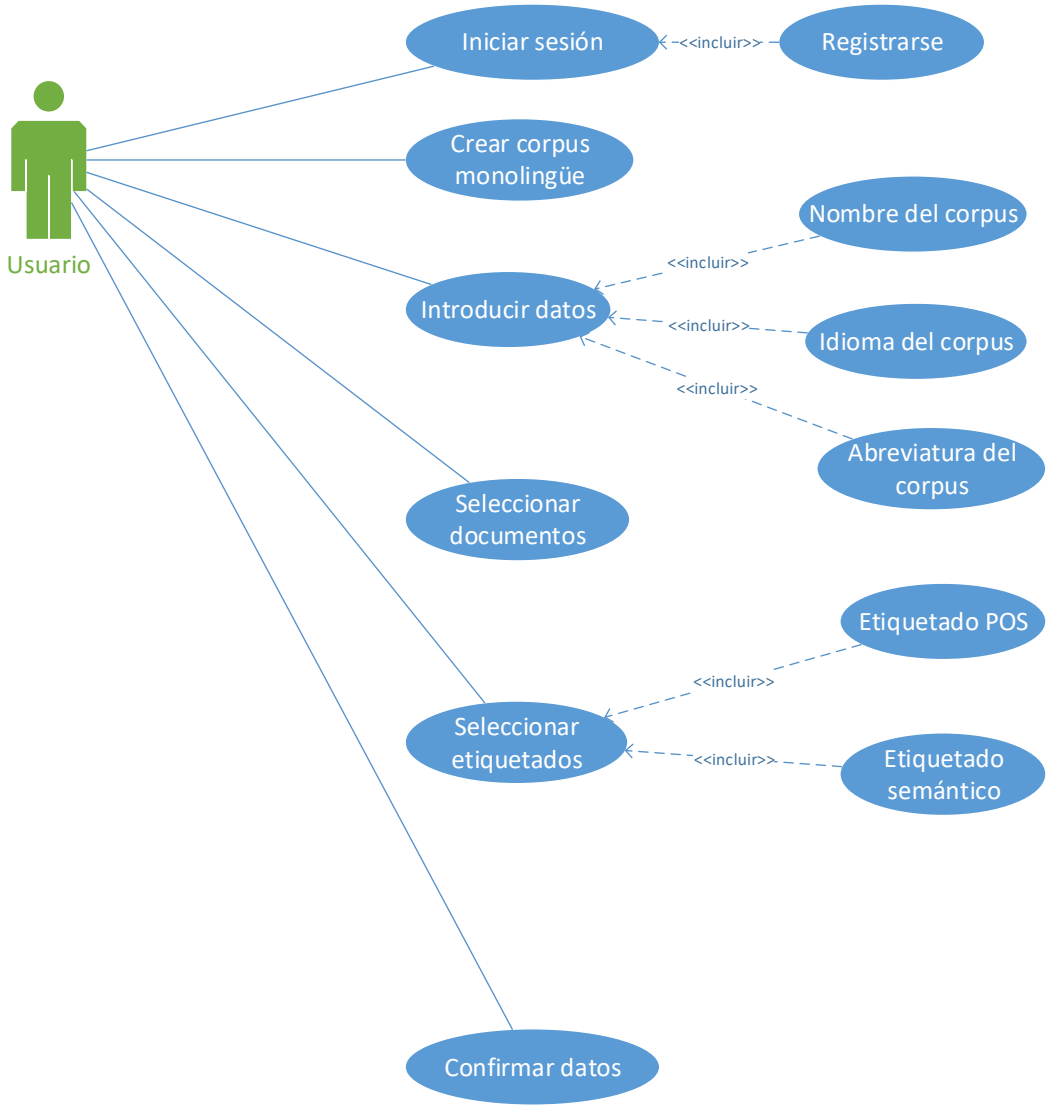


Figura 178 - Diagrama de caso de uso “Creación de un corpus monolingüe”

Caso de uso “Creación de un corpus paralelo bi-/multilingüe”

Creación de un corpus paralelo multilingüe	
Precondiciones	Estar registrado y haber iniciado sesión en el sistema. Documentos del corpus a crear.
Postcondiciones	Confirmación de creación del corpus paralelo multilingüe.
Actores	Usuario y framework.
Descripción	Se trata de un proceso asistido por el framework: (1) El framework presenta las opciones disponibles (Framework). (2) El usuario selecciona la opción “Crear Corpus” y posteriormente “Crear corpus paralelo multilingüe” (Usuario). (3) El framework presenta una vista para que el usuario seleccione el número de subcorpus que componen el corpus paralelo multilingüe (Framework). (4) El usuario introduce el número de subcorpus (Usuario). (5) El framework presenta los datos que ha de introducir el usuario (Framework). (6) El usuario introduce los datos pertinentes (Usuario) a. Nombre del corpus paralelo multilingüe. b. Idioma de cada uno de los subcorpus. c. Nombre de cada uno de los subcorpus. d. Abreviatura de cada uno de los corpus. (7) El framework presenta una vista para la carga de los documentos que formarán parte de cada uno de los subcorpus (Framework). (8) El usuario selecciona los documentos desde su equipo (Usuario). (9) El framework presenta los tipos de etiquetados a nivel de palabra disponibles (Framework). (10) El usuario selecciona los etiquetados que desea aplicar (Usuario). (11) El framework presenta toda la información para su confirmación por parte del usuario (Framework). (12) El usuario confirma la creación del corpus (Usuario).
Variaciones (escenarios secundarios)	No hay.

Excepciones	La vista controla todas las excepciones y avisa al usuario mediante mensajes por pantalla.

Tabla 62 - Descripción de caso de uso "Crear un corpus paralelo bi-/multilingüe"

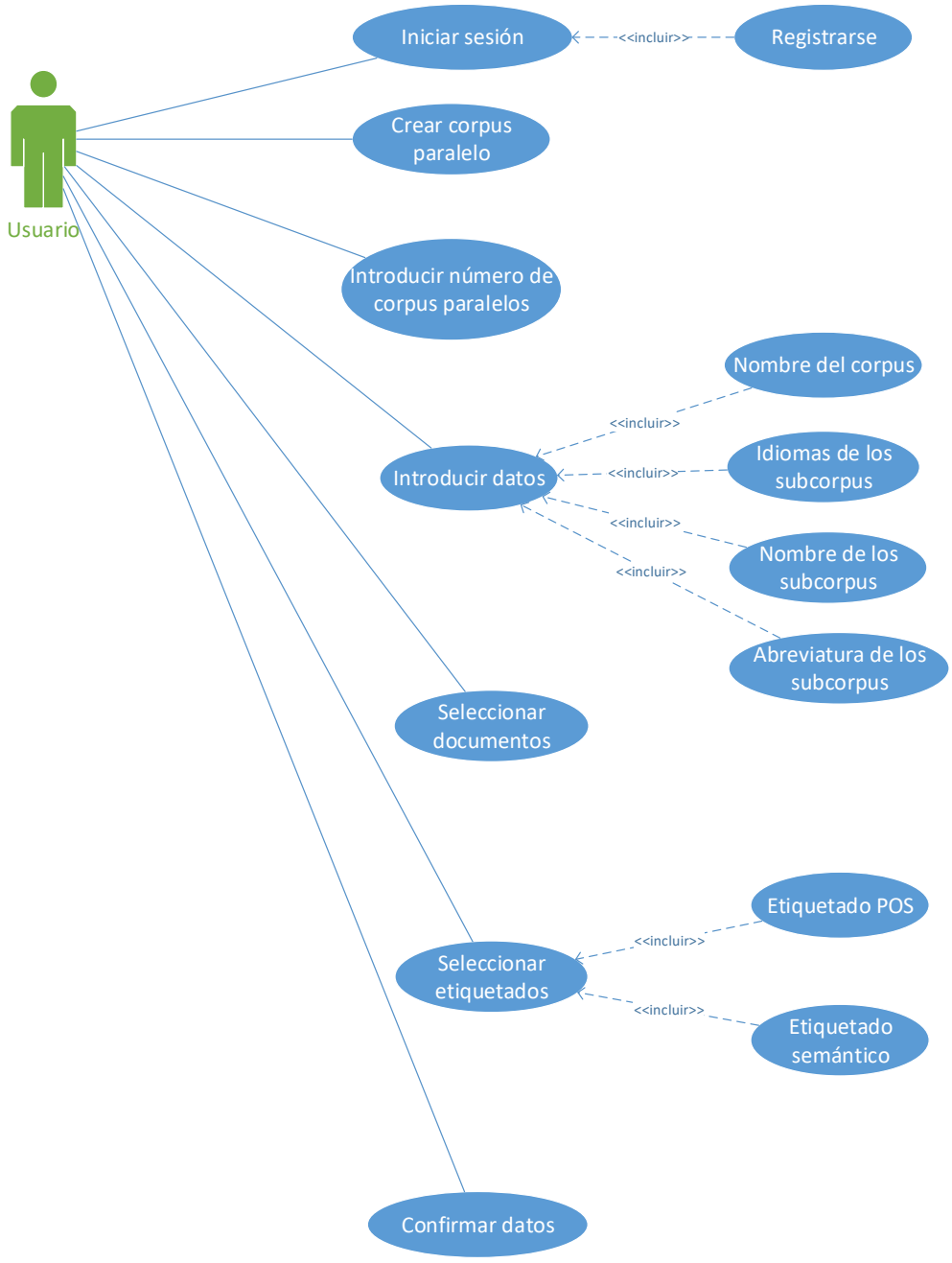


Figura 179 - Diagrama de caso “Creación de un corpus paralelo bi-/multilingüe”

Caso de uso “Creación de un corpus comparable”

Creación de un corpus comparable	
Precondiciones	Estar registrado y haber iniciado sesión en el sistema. Documentos del corpus a crear.
Postcondiciones	Confirmación de creación del corpus comparable.
Actores	Usuario y framework.
Descripción	Se trata de un proceso asistido por el framework: (1) El framework presenta las opciones disponibles (Framework). (2) El usuario selecciona la opción “Crear Corpus” y posteriormente “Crear corpus comparable” (Usuario). (3) El framework presenta una vista para que el usuario seleccione el número de subcorpus que forman parte del corpus comparable (Framework). (4) El usuario introduce el número de subcorpus (Usuario). (5) El framework presenta los datos que ha de introducir el usuario (Framework). (6) El usuario introduce los datos pertinentes (Usuario) a. Nombre del corpus comparable. b. Idioma de cada uno de los subcorpus. c. Nombre de cada uno de los subcorpus. d. Abreviatura de cada uno de los corpus. (7) El framework presenta una vista para la carga de los documentos que formarán parte de cada uno de los subcorpus (Framework). (8) El usuario selecciona los documentos desde su equipo (Usuario). (9) El framework presenta los tipos de etiquetados a nivel de palabra disponibles (Framework). (10) El usuario selecciona los etiquetados que desea aplicar (Usuario). (11) El framework presenta toda la información para su confirmación por parte del usuario (Framework). (12) El usuario confirma la creación del corpus (Usuario).
Variaciones (escenarios secundarios)	No hay.

Excepciones	La vista controla todas las excepciones y avisa al usuario mediante mensajes por pantalla.

Tabla 63 - Descripción de caso de uso "Crear un corpus comparable"

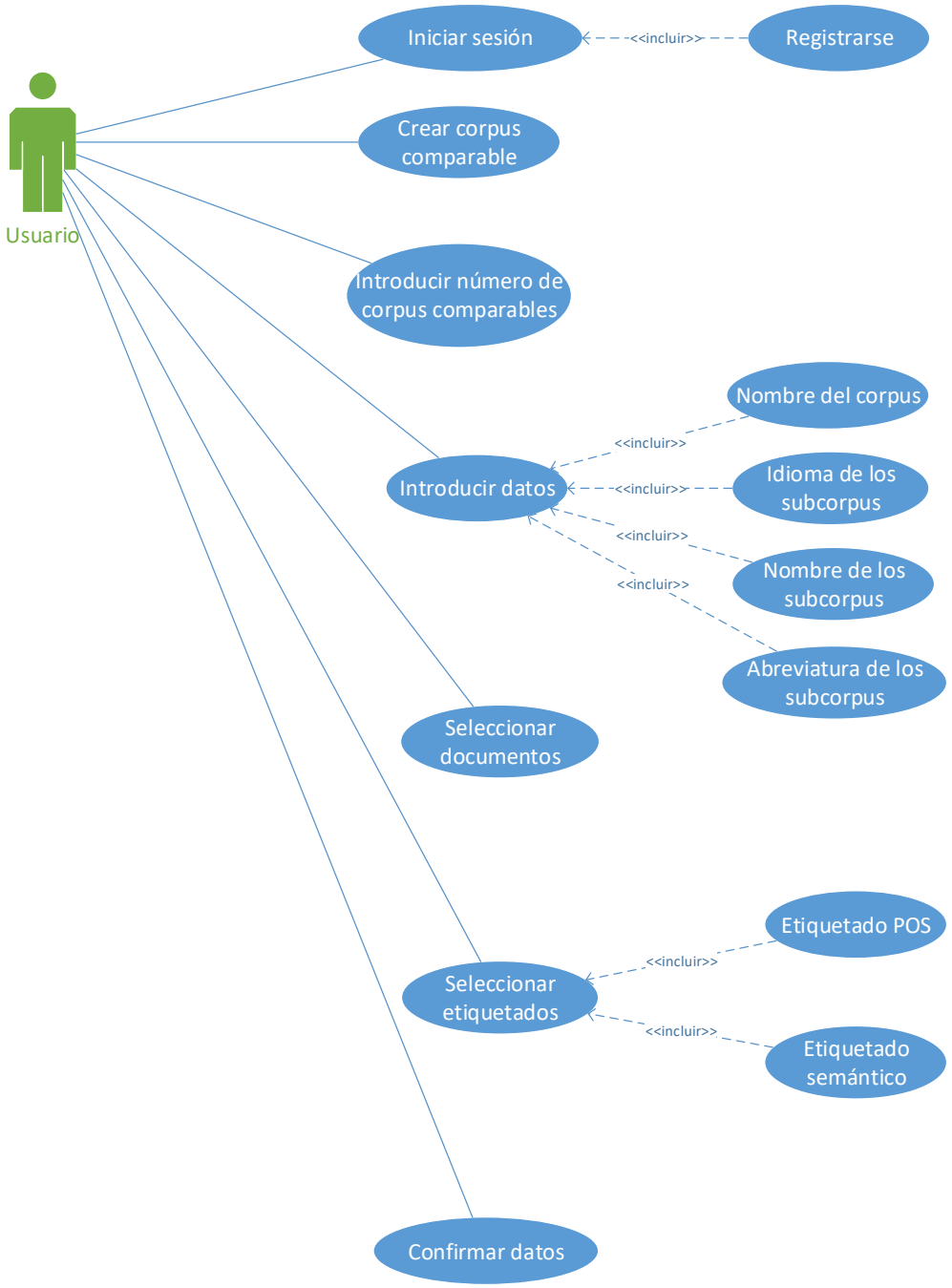


Figura 180 - Diagrama de caso de uso “Creación de un corpus comparable”

Caso de uso “Creación de un corpus retórico”

Creación de un corpus retórico	
Precondiciones	Estar registrado y haber iniciado sesión en el sistema. Documentos del corpus a crear.
Postcondiciones	Confirmación de creación del corpus retórico.
Actores	Usuario y framework.
Descripción	Se trata de un proceso asistido por el framework: (1) El framework presenta las opciones disponibles (Framework). (2) El usuario selecciona la opción “Crear Corpus” y posteriormente “Crear corpus retórico” (Usuario). (3) El framework presenta los datos que ha de introducir el usuario (Framework). (4) El usuario introduce los datos pertinentes (Usuario): a. Nombre del corpus. b. Idioma del corpus. c. Abreviatura del corpus. (5) El framework presenta una vista para la carga de los documentos que formarán parte del corpus (Framework). (6) El usuario selecciona los documentos desde su equipo (Usuario). (7) El framework presenta los documentos seleccionados por el usuario (Framework). (8) El usuario lleva a cabo un etiquetado retórico en cada uno de los documentos (Usuario). (9) El framework presenta los tipos de etiquetados a nivel de palabra disponibles (Framework). (10) El usuario selecciona los etiquetados adicionales que desea aplicar (Usuario). (11) El framework presenta toda la información para su confirmación por parte del usuario (Framework). (12) El usuario confirma la creación del corpus (Usuario).
Variaciones (escenarios secundarios)	No hay.
Excepciones	La vista controla todas las excepciones y avisa al usuario mediante mensajes por pantalla.

Tabla 64 - Descripción de caso de uso "Crear de un corpus monolingüe retórico"

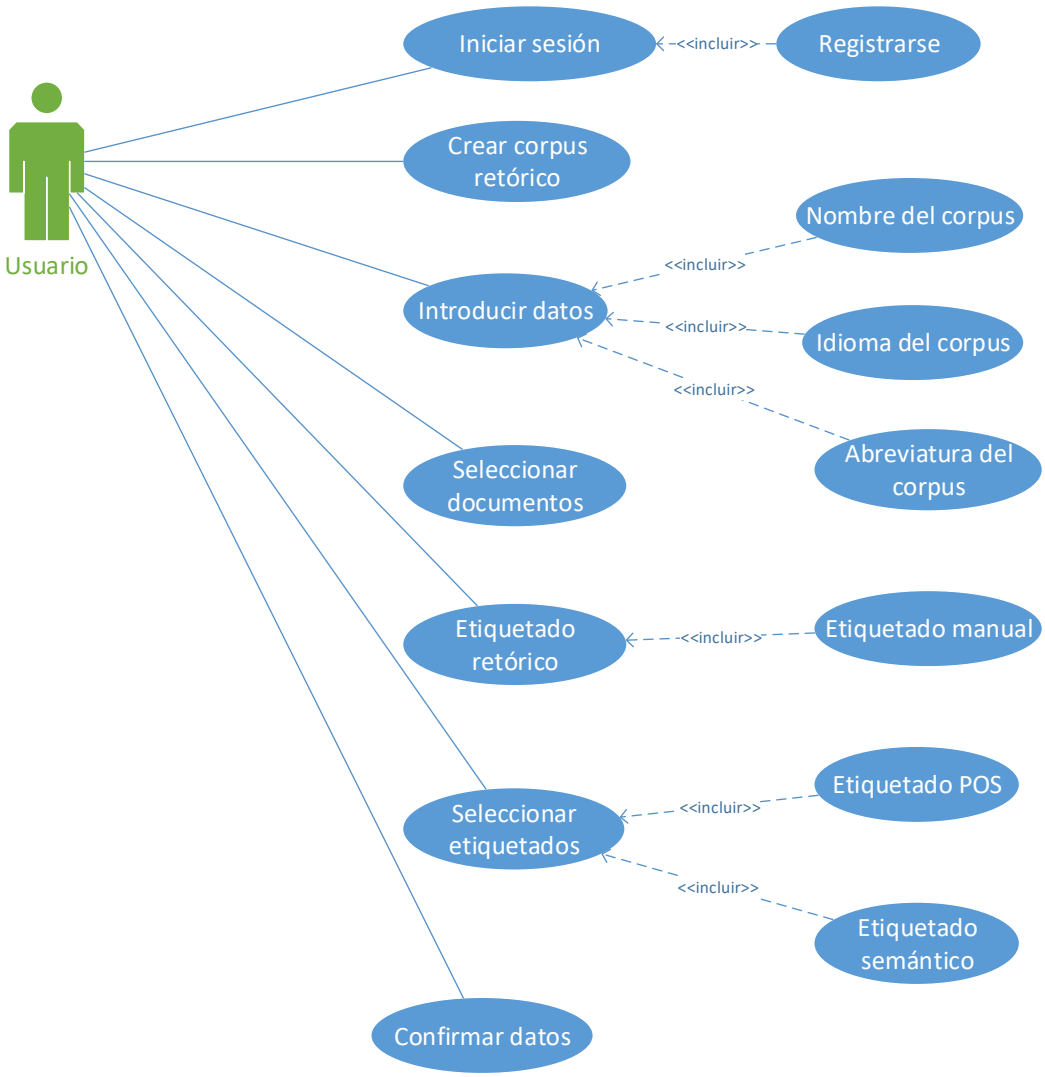


Figura 181 - Diagrama de caso de uso de "Creación de un corpus monolingüe retórico"

Caso de uso "Consulta de un corpus y extracción de estadísticas de la consulta"

Consulta de un corpus y extracción de estadísticas de la consulta	
Precondiciones	Estar registrado y haber iniciado sesión en el sistema. Haber creado un corpus con anterioridad.
Postcondiciones	No hay.
Actores	Usuario y framework.
Descripción	Se trata de un proceso asistido por el framework:

	(1) El framework presenta las opciones disponibles (Framework). (2) El usuario selecciona la opción "Consultar Corpus" (Usuario). (3) El framework presenta una vista para que el usuario seleccione el corpus que desea consultar (Framework). (4) El usuario selecciona el corpus (Usuario). (5) El framework presenta los campos y las opciones de búsqueda junto a las funciones estadísticas del corpus (Framework). (6) El usuario selecciona el tipo y los parámetros de búsqueda (Usuario). (7) El usuario ejecuta la búsqueda (Usuario). (8) El framework presenta una vista para mostrar los resultados de la búsqueda (Framework). (9) El usuario selecciona ver una estadística de la consulta (Usuario). (10) El framework crea un archivo Excel con los resultados de la estadística de la consulta ejecutada (Framework).
Variaciones (escenarios secundarios)	No hay.
Excepciones	La vista controla todas las excepciones y avisa al usuario mediante mensajes por pantalla.

Tabla 65 - Descripción de caso de uso "Consulta de un corpus y extracción de estadísticas de la consulta"

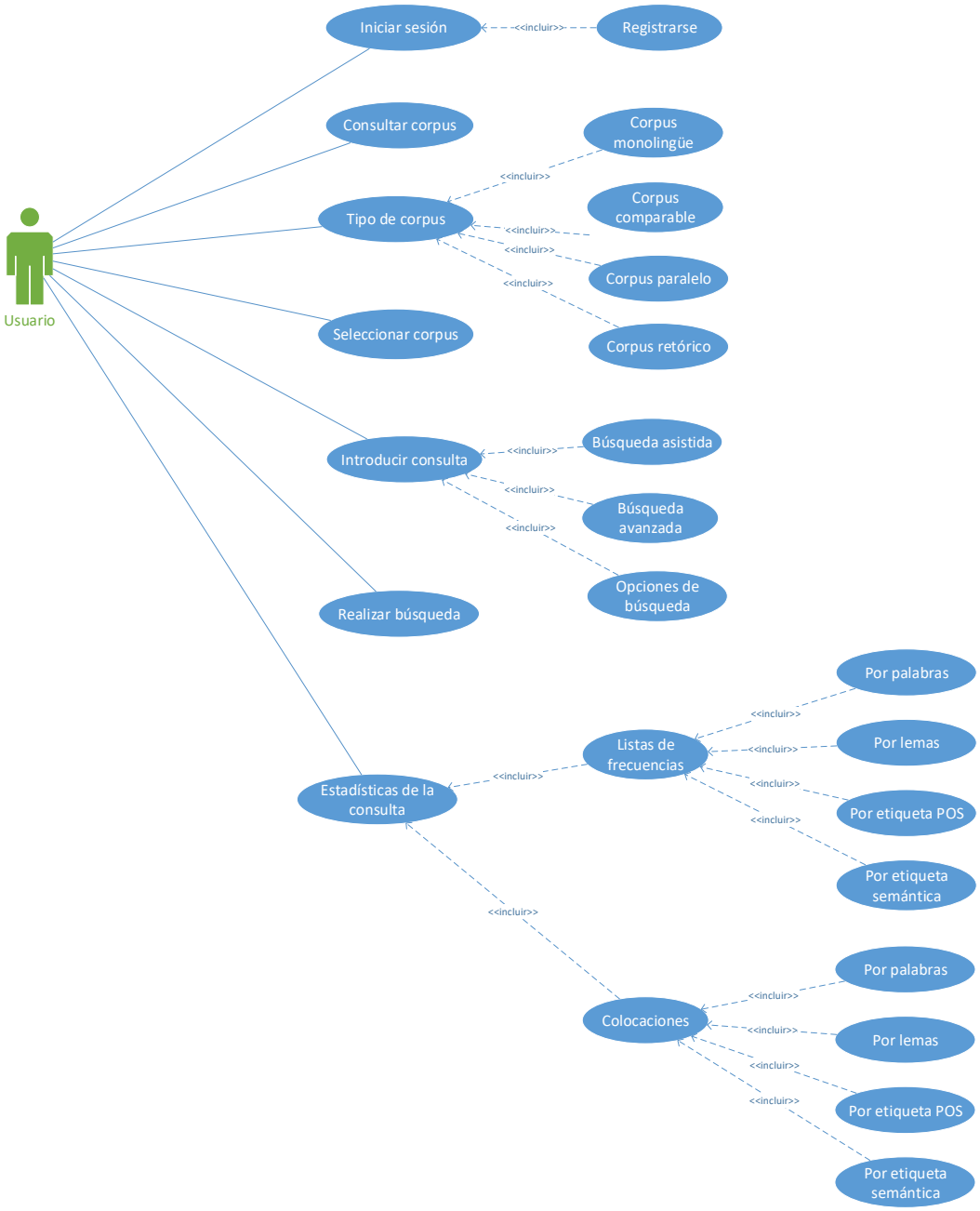


Figura 182 - Diagrama de caso de uso de "Consulta de un corpus y extracción de estadísticas de la consulta"

Caso de uso “Extraer estadísticas de un corpus”

Extraer estadísticas de un corpus	
Precondiciones	Estar registrado y haber iniciado sesión en el sistema. Corpus creado con anterioridad.
Postcondiciones	No hay.
Actores	Usuario y framework.
Descripción	Se trata de un proceso asistido por el framework (1) El framework presenta las opciones disponibles (Framework) (2) El usuario selecciona la opción “Consultar Corpus” (Usuario). (3) El framework presenta una vista para que el usuario seleccione el corpus que desea consultar (Framework). (4) El usuario selecciona el corpus (Usuario). (5) El framework presenta los campos y las opciones de búsqueda junto a las funciones estadísticas del corpus (Framework). (6) El usuario selecciona la estadística del corpus que desee (Usuario). (7) El framework crea un archivo Excel con los resultados de la estadística seleccionada (Framework).
Variaciones (escenarios secundarios)	No hay.
Excepciones	La vista controla todas las excepciones y avisa al usuario mediante mensajes por pantalla.

Tabla 66 - Descripción de caso de uso "Extraer estadísticas de un corpus"

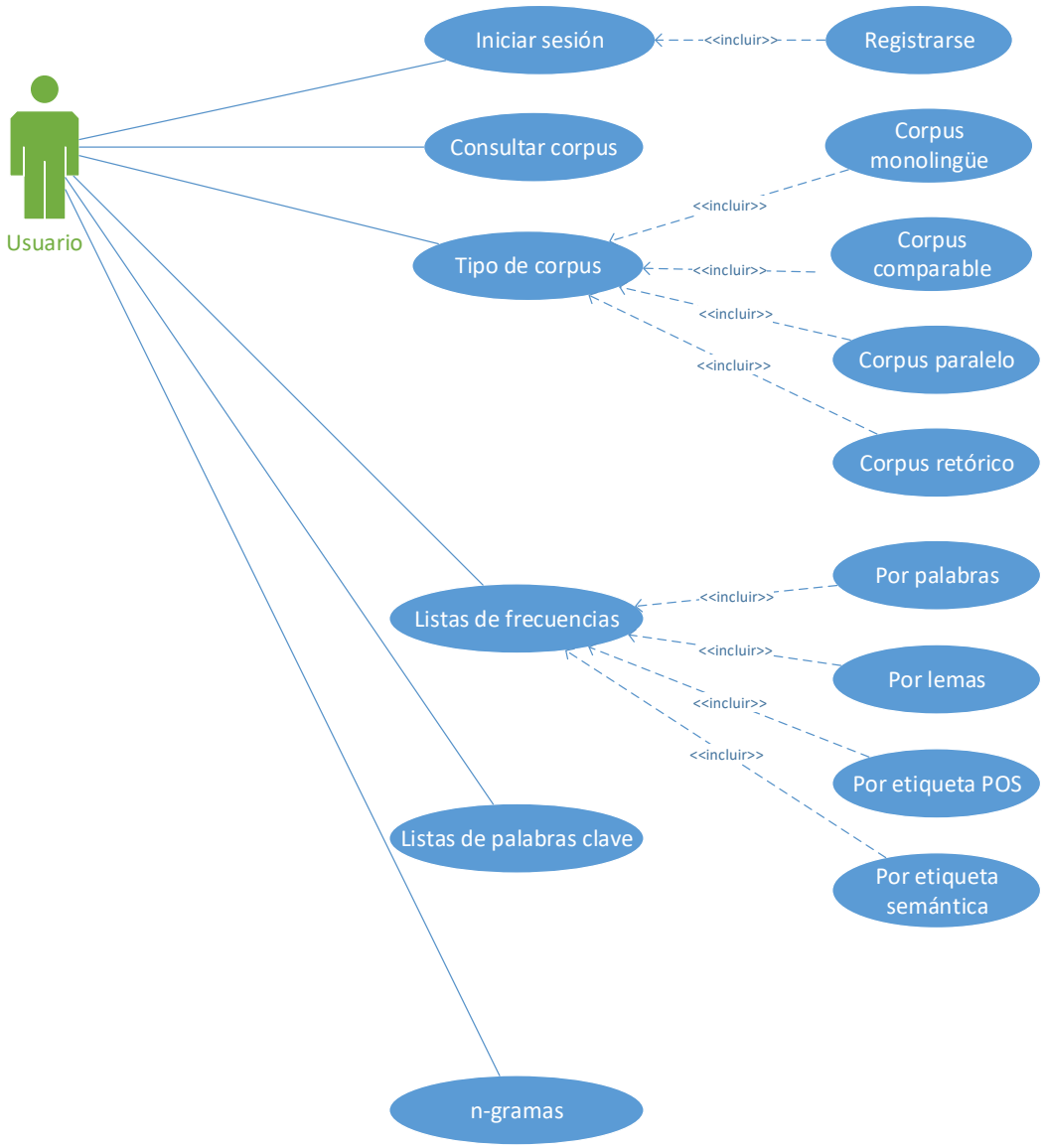


Figura 183 - Diagrama de caso de uso “Estadística de un corpus”

Anexo 2: Diagrama de clases completo

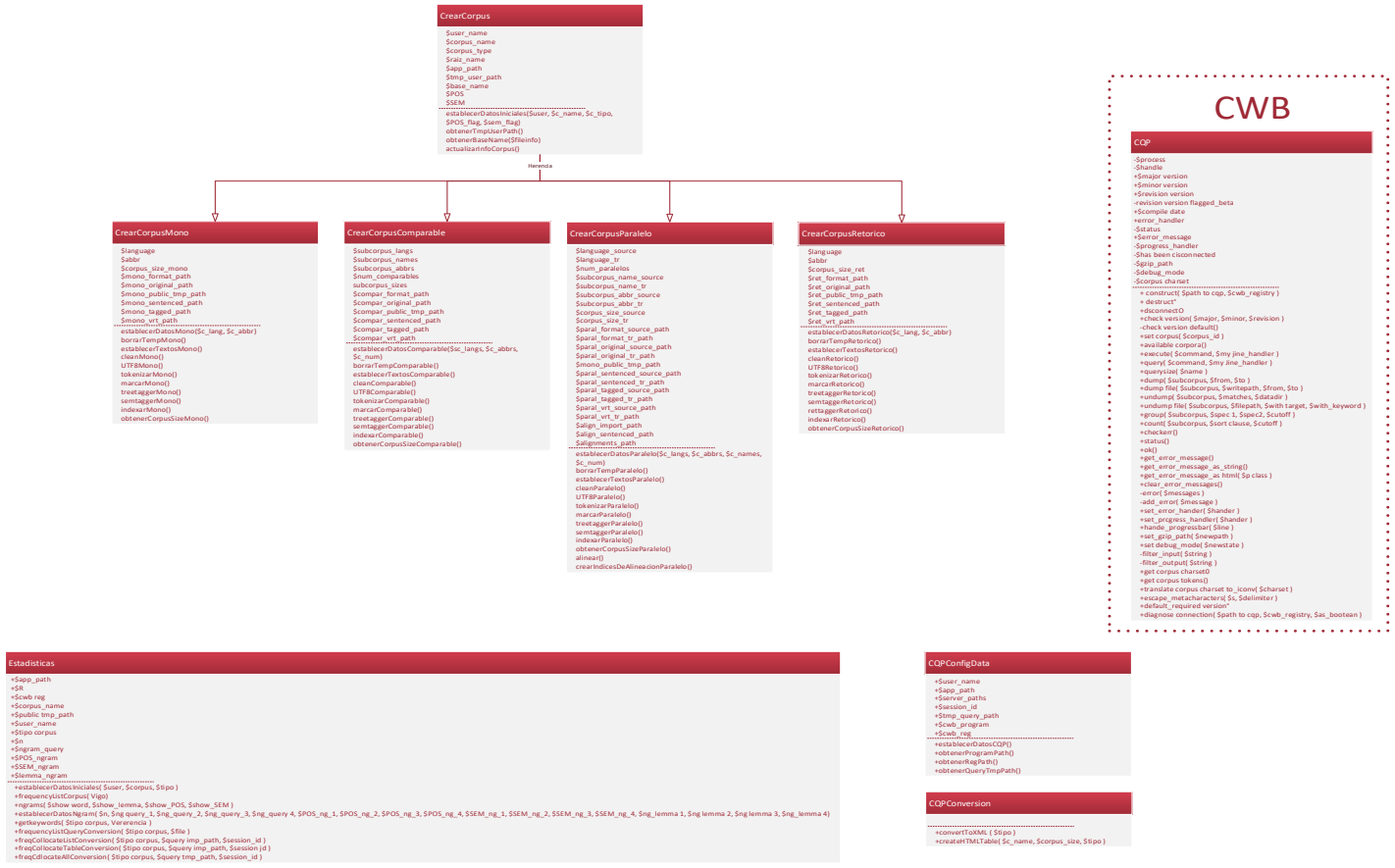


Figura 184 - Diagrama de clases completo

Anexo 3: Cuestionario SUS

1. Creo que me gustaría usar *ACTRES Corpus Manager* frecuentemente.

2. *ACTRES Corpus Manager* me pareció innecesariamente complejo.

3. Creo que *ACTRES Corpus Manager* es fácil de usar.

4. Creo que necesitaría apoyo por parte de personal técnico para ser capaz de usar *ACTRES Corpus Manager*.

5. Las funciones de *ACTRES Corpus Manager* estaban bien integradas.

6. Creo que hay demasiada inconsistencia en *ACTRES Corpus Manager*.

7. Imagino que la mayor parte de la gente aprendería a usar *ACTRES Corpus Manager* rápidamente.

8. *ACTRES Corpus Manager* me pareció difícil de usar.

9. Me sentí muy seguro usando *ACTRES Corpus Manager*.

10. Necesito aprender muchas cosas antes de usar *ACTRES Corpus Manager*.

La imaginación es más importante que el conocimiento.

Albert Einstein