Departamento de Ciencias da Computación e
Tecnoloxías da Información
Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN ENXEÑARÍA INFORMÁTICA
MENCIÓN EN ENXEÑARÍA DO SOFTWARE

Plataforma para la etiquetación asistida de casos de riesgo temprano en internet

Estudante: David Otero Freijeiro
Dirección: Javier Parapar López
Daniel Valcarce Silva

A Coruña, setembro de 2019.

A mi familia

Agradecimientos

A Daniel y Javier, por ser unos excelentes directores y por su magnífico trato, tanto personal como profesional. Asimismo, también quiero agradecer al resto de miembros del IRLab el tiempo compartido, especialmente a Álvaro y a Alfonso.

A mi familia, por su constante apoyo, sin el cual no habría llegado hasta aquí. A ti, abuela, allá dónde estés. A vosotros, padrinos, por ser una constante fuente de inspiración.

A Nuria, por ser la mejor compañera de vida.

A todos los amigos que he hecho durante todos estos años, habéis hecho que este camino haya sido mucho más ameno. Anxo y Pablo, es una suerte contar con vosotros.

A todos y cada uno de vosotros, **gracias**.

Este trabajo ha sido posible gracias a la ayuda de la Secretaría de Estado de Universidades, Investigación, Desarrollo e Innovación del Ministerio de Ciencia, Innovación y Universidades bajo el proyecto RTI2018-093336-B-C22.

Resumen

Desde la invención de la World Wide Web, la necesidad de los usuarios por buscar información en Internet no ha parado de aumentar. Esto ha provocado que crezcan de manera continua los sistemas de recuperación de información y los ámbitos donde esta disciplina tiene alguna aplicación. Por esto, es necesario elaborar metodologías y herramientas que permitan realizar una correcta evaluación de estos nuevos sistemas.

El objetivo de este proyecto es diseñar y construir una plataforma que permita la etiquetación de documentos eficiente por parte de los asesores asociados a casos de trastornos psicológicos y mentales. Esta plataforma se usará para construir la colección de prueba en la competición de CLEF eRisk de 2020, que se celebra con el objetivo de evaluar la efectividad de metodologías y métricas para la detección temprana de casos de riesgo en Internet, especialmente aquellos relacionados con la salud, como la anorexia o la depresión. También se busca que la plataforma sea flexible a la hora de poder añadir nuevos modelos de recuperación o nuevas estrategias de *pooling*.

Para poder lograr una correcta consecución de estos objetivos se ha decidido emplear una metodología ágil con ciclos iterativos e incrementales que permiten adaptarse a las circunstancias cambiantes del proyecto. Siguiendo este proceso se ha obtenido una aplicación de calidad que cumple con los objetivos establecidos.

Abstract

Since the invention of the World Wide Web, the need for users to search for information on the Internet has not stopped increasing. This has led to the continuous growth of information retrieval systems and the areas where this discipline has some application. For this reason, it is necessary to develop methodologies and tools that allow a correct evaluation of these new systems.

The aim of this project is to design and build a platform that allows the efficient labeling of documents by asessors associated with cases of psychological and mental disorders. This platform will be used to build the test collection in the 2020 CLEF eRisk competition, which is held with the aim of evaluating the effectiveness of methodologies and metrics for the early detection of risk cases on the Internet, especially those related to health, such as anorexia or depression. The platform is also intended to be flexible when adding new recovery models or new *pooling* strategies.

In order to achieve these objectives correctly, it has been decided to use an agile methodology with iterative and incremental cycles that allow adaptation to the changing circumstances of the project. Following this process has resulted in a quality application that meets the established objectives.

Palabras clave:

- Recuperación de información
- Evaluación
- Pooling
- Colecciones
- CLEF eRisk
- TREC
- Cranfield

Keywords:

- Information Retrieval
- Evaluation
- Pooling
- Collections
- CLEF eRisk
- TREC
- Cranfield

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Históricamente, el ser humano siempre ha necesitado medios para poder representar el conocimiento sobre el mundo que le rodea. Las primeras bibliotecas de la humanidad, que aparecieron hace más de cinco siglos, servían como almacenamiento de tablillas cuyo contenido era mayormente religioso. Desde este primer intento del hombre de poner orden al conocimiento del que disponía, el volumen de información existente no ha parado de crecer hasta el día de hoy. Recientemente, con la invención de los ordenadores y de la World Wide Web, Internet se ha convertido en el mayor repositorio de información que la humanidad haya visto jamás. Con esta ingente cantidad de información disponible, y que no para de crecer, nace la necesidad de desarrollar métodos para poder acceder a ella de una manera sencilla y eficaz.

La Recuperación de Información (*Information Retrieval*, IR) es el área de la ciencia encargada de tratar con la representación, almacenamiento, organización y acceso a elementos de información [1]. El objetivo es proporcionar al usuario información relevante conforme a sus *necesidades*. Definida de esta manera puede parecer que es una actividad realizada solamente por unos pocos: bibliotecarios, documentalistas, o profesionales similares. La realidad es que todo el mundo realiza tareas de recuperación de información diariamente: cuando buscan algo en la web o cuando revisan su correo electrónico. Hoy día, el ejemplo más extendido de un sistema de recuperación de información son los motores de búsqueda en la web como Google[1], Bing[2] o Baidu[3].

De la misma manera en que aumenta el volumen de información al que tenemos acceso, también crece el número de aplicaciones de la recuperación de información: motores de búsqueda, sistemas conversacionales, sistemas de recomendación, etc. Debido a la importancia que estos sistemas tienen en nuestro día a día, es crucial desarrollar métodos para evaluar la calidad y la eficacia de estos.

Este proyecto se enmarca dentro de la evaluación de sistemas de recuperación de infor-

[1] https://www.google.com
[2] https://www.bing.com
[3] https://www.baidu.com

mación, concretamente en la construcción de colecciones de evaluación.

1.1 Motivación

La importancia de los sistemas de búsqueda hoy en día es indudable. En 2019, Google procesa más de 40 000 búsquedas cada segundo, lo que se traduce en más de 3 000 000 000 al día. En la Figura 1.1 [2] se ilustra un histórico del número de consultas que ha recibido Google durante los últimos años.

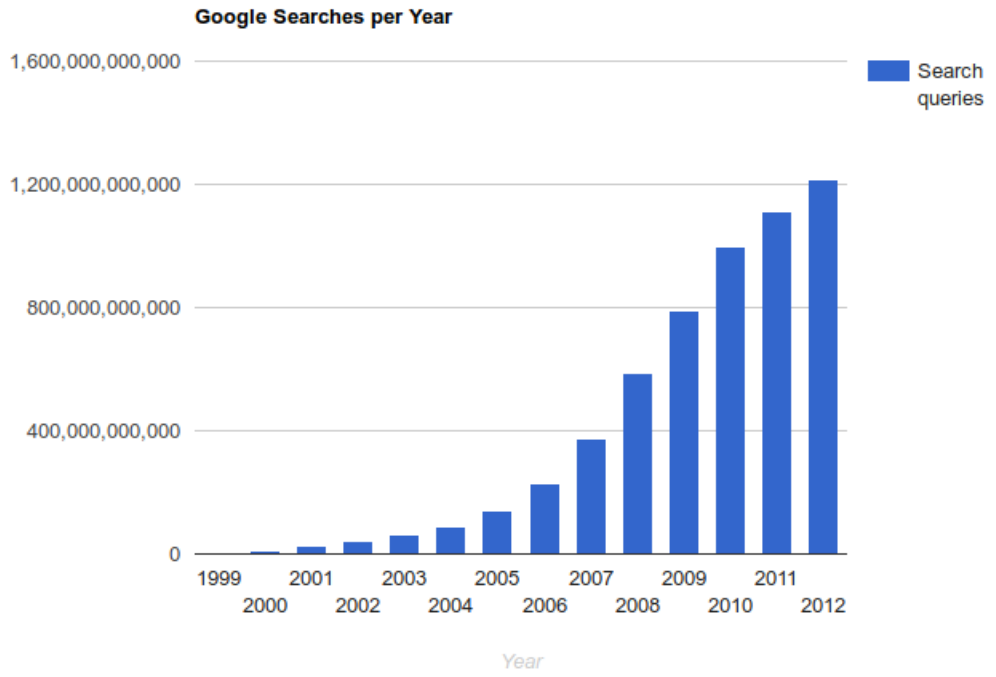

Figura 1.1: Histórico de búsquedas procesadas por Google.

Por otro lado, el origen de estas consultas también está cambiando. Gran parte de estas proceden de dispositivos móviles, más del 50 actualmente, lo que ilustra que el mercado es algo cambiante y que no sigue una línea común, ya que esto cambia la manera en la que se utilizan los buscadores. Además estas consultas ya no se introducen de manera explícita en los sistemas, hoy en día están en auge los sistemas que reciben información de los usuarios de manera implícita, como los sistemas de recomendación. Esto provoca la aparición de nuevas tareas y modelos de recuperación de información, lo que motiva el estudio y desarrollo de metodologías y métricas para poder evaluar con eficacia estos nuevos desarrollos y así adaptarse a las necesidades cambiantes de los usuarios.

Dentro de la disciplina de la Recuperación de Información se enmarcan diferentes tipos

de tareas que tienen diferentes objetivos y que atacan diferentes problemáticas, pero cuyas bases son comunes. Ejemplos de estas tareas son la recuperación ad-hoc, la clasificación de documentos, o la búsqueda de respuestas.

De entre todas, la tarea de recuperación ad-hoc es la más conocida [3], ya que en ella se basan los buscadores web. Esta tarea se realiza sobre una colección de documentos. Estos pueden ser páginas web, vídeos o libros, entre otros. La colección se indexa previamente para crear estructuras que sirvan para hacer una búsqueda eficiente. Esta tarea consiste en buscar esos documentos de la colección que son relevantes para la necesidad de información que tiene el usuario. Esta necesidad que tiene el usuario se introduce en el sistema como un texto corto, que comúnmente se conoce como *query*. El sistema procesa esta *query* utilizando un modelo de recuperación de información y produce una lista ordenada según la relevancia de los documentos. Un documento se considera relevante cuando el usuario advierte que satisface su necesidad de información.

La importancia de los sistemas de este tipo es indudable hoy en día, por eso es vital evaluar la calidad de estos. Actualmente, el paradigma Cranfield y las iniciativas TREC (Text REtrieval Conference) se han convertido en un estándar para la evaluación en la recuperación de información. Las colecciones de prueba que se utilizan para estos propósitos son muy costosas de construir, ya que es necesario juzgar la relevancia de cientos de miles de documentos. Además, estas colecciones, aunque de propósito general, pueden no ser adecuadas para algunas tareas dentro de dominios muy concretos.

La principal motivación de este proyecto nace de la falta de herramientas para poder construir estas colecciones a un bajo coste y sin la necesidad de organizar campañas como las de TREC.

1.2 Objetivos

El propósito de este proyecto es desarrollar una plataforma para asistir a la etiquetación de casos de riesgo temprano sobre usuarios de la red social Reddit. Esta plataforma se utilizará en la competición internacional eRisk para construir las colecciones que se liberan a los participantes, previamente a la organización de la competición.

CLEF eRisk[4] [4, 5, 6] es una iniciativa organizada con el objetivo de proponer metodologías y métricas, y evaluar técnicas y algoritmos para la detección temprana de riesgos en internet, especialmente aquellos relacionados con la salud, como la depresión, la anorexia o los daños autoinfligidos. Con este propósito, se liberan anualmente colecciones de textos escritos por usuarios en redes sociales. En este contexto, este proyecto está principalmente orientado a asistir a los asesores que realizan diagnósticos sobre los usuarios de las redes sociales, así como

[4] https://early.irlab.org

también evaluar la eficacia de diferentes modelos a la hora de construir nuevas colecciones. En años anteriores el objetivo de CLEF eRisk ha sido la detección de depresión (2017[5]) [4], así como la detección de tanto anorexia como depresión (2018[6]) [5]. La tarea de 2019[7] fue sobre anorexia, depresión y daño autoinfligido [6]. Esta plataforma se utilizará para la competición que se organizará en 2020.

El objetivo es diseñar e implementar una aplicación web para le etiquetación asistida de casos de riesgos temprano en Internet. La plataforma permitirá a los asesores emitir juicios sobre los usuarios de la red social Reddit en función de los textos publicados por estos. Además, la aplicación implementará diferentes modelos de recuperación de información a la hora de obtener los casos a juzgar y evaluará su eficacia según los diagnósticos realizados por los jueces sobre los usuarios de Reddit. Esta eficacia se mide por el número de juicios positivos que los asesores han emitido. El objetivo es poder maximizar este número, y para eso la plataforma permite combinar diferentes modelos de recuperación con las diferentes estrategias de presentación de los documentos. Así se consigue maximizar el trabajo de los asesores para reducir los costes de la construcción de la colección. La plataforma permitirá configurar las distintas estrategias de presentación de los usuarios a los asesores por parte del administrador, y mostrará el rendimiento comparado de cada una en base a los juicios positivos emitidos por los mismos. También será posible exportar las colecciones de textos juntos con los juicios emitidos sobre los usuarios para liberar los datos y asistirá a la plataforma de participación de equipos en la competición.

También se contemplan los siguientes objetivos:

- **Eficacia**: la calidad de las colecciones obtenidas con la aplicación es de vital importancia, ya que se utilizarán para hacer evaluaciones de sistemas.
- **Eficiencia**: las colecciones construidas serán de un tamaño bastante grande, por lo que es esencial que la aplicación sepa tratar con estas grandes cantidades de datos.
- **Flexibilidad**: la aplicación debe aportar flexibilidad a la hora de añadir nuevos modelos de recuperación y nuevas estrategias de presentación de los textos a los asesores, de tal manera que no requiera mucho esfuerzo añadir estas nuevas implementaciones.

1.3 Estructura de la memoria

A continuación, se expone la estructura de la presente memoria, explicando brevemente el contenido de cada sección.

[5] https://early.irlab.org/2017
[6] https://early.irlab.org/2018
[7] https://early.irlab.org

- **Introducción**: explica el contexto en el que se enmarca el proyecto, introduce la problemática a tratar y detalla el alcance y objetivos del mismo desde un punto de vista global. También muestra la estructura de la memoria y el plan de trabajo seguido.
- **Fundamentos**: introduce los conceptos en los que se engloba la temática a abordar por el proyecto y el estado del arte de las técnicas a tratar.
- **Tecnologías y herramientas**: describe y justifica las principales tecnologías empleadas para desarrollar el objeto del proyecto atendiendo a los requisitos del mismo.
- **Metodología y gestión de proyecto**: detalla la metodología seguida para la consecución del proyecto, exponiendo la planificación y gestión del mismo.
- **Desarrollo**: detalla la arquitectura de la aplicación, el modelo de datos y el proceso seguido en la fase de desarrollo.
- **Conclusiones y trabajo futuro**: proporciona una evaluación global del producto desarrollado así como futuras líneas de trabajo que se podrían explorar.
- **Apéndices**: está compuesto las siguientes secciones adicionales:
  - **Artículo del FDIA 2019**: reproduce el artículo presentado y aceptado en el FDIA 2019.
  - **Artículo XoveTIC 2019**: reproduce el artículo presentado y aceptado en el II Congreso XoveTIC.
  - **Glosario**: se definen los términos técnicos empleados en el proyecto.
  - **Bibliografía**: recoge la documentación bibliográfica sobre la que se apoya el proyecto.

1.4 Plan de trabajo

En general, durante la elaboración del proyecto se han seguido las siguientes etapas relacionadas con la metodología empleada:

- Estudio del campo de la recuperación de información, y en concreto sobre la evaluación en esta disciplina.
- Realización de un estudio de requisitos funcionales y no funcionales para la consecución de los objetivos establecidos.

- Estudio sobre las tecnologías disponibles para desarrollar los diferentes componentes que forman la aplicación, procediendo a la elección de las mismas y a la configuración del entorno.
- En cada iteración se han realizado las siguientes tareas, resultando en un incremento del software al final de cada una:
  - Propuesta de una aplicación para atacar la problemática.
  - Diseño de la aplicación.
  - Implementación y pruebas.
- Elaboración de la memoria y la documentación.

Capítulo 2

Fundamentos

Este capítulo consta de dos secciones, en la primera se da una breve introducción al campo de la recuperación de información y sus principales aplicaciones y en la segunda se introduce el concepto de evaluación y el proceso de construcción de colecciones de prueba.

2.1 Recuperación de información

La Recuperación de Información es un campo de la ciencia de computadores cuyo objetivo es satisfacer las necesidades de información de los usuarios [1, 3]. Los motores de búsqueda en la web quizá sean los más conocidos ejemplos de un sistema de recuperación de información. De una manera más formal, la recuperación de información se podría definir de la siguiente manera:

> La Recuperación de Información trata con la representación, almacenamiento, organización, y acceso a elementos de información como documentos, páginas web, catálogos, registros estructurados o semiestructurados y objetos multimedia. La representación y organización de la información debe ser tal que permita a los usuarios acceder de una manera fácil a información de su interés [1].

La importancia de la recuperación de información ha crecido de manera exponencial desde la invención de la World Wide Web. El gran aumento del volumen de información disponible ha provocado un gran crecimiento en el desarrollo de nuevos modelos de recuperación para adecuarse a las crecientes necesidades de los usuarios.

2.2 Evaluación

La recuperación de información es una disciplina profundamente enraizada en la evaluación [7]. Muchos sistemas de recuperación de información juegan un papel muy importante

en la vida de las personas y son utilizados por millones de usuarios diariamente. Por esta razón, desarrollar métodos para poder evaluar la calidad de estos sistemas se convierte en una tarea de suma importancia.

Existe un amplio espectro de técnicas sobre como realizar una evaluación sobre un sistema, desde un punto de vista *system-oriented*, donde se estudian los algoritmos utilizados y la implementación de los modelos; hasta un punto de vista *user-oriented*, donde se presta más atención a las interfaces de los sistemas y a la experiencia de los usuarios. Cuando el objetivo es medir como un sistema cumple con las necesidades del usuario, la evaluación *user-based* es mucho más apropiada: es una medida directa del rendimiento global del sistema. Sin embargo, este tipo de evaluación es muy costosa de realizar y difícil de hacer correctamente. Un experimento de este tipo, bien diseñado, debe poseer una cantidad de usuarios suficientemente representativa de los usuarios del sistema. Además, los sistemas comparados deben estar bien diseñados y los experimentos y resultados deben ser reproducibles, lo que es difícil de lograr. Estas dificultades han hecho que los investigadores elijan la evaluación *system-oriented* en muchos casos, utilizando colecciones de prueba con documentos etiquetados por asesores humanos según las necesidades de información.

A la hora de evaluar un sistema de recuperación de información se pueden tener en cuenta dos factores: eficiencia y eficacia. La eficiencia mide cómo se comporta un sistema en términos de tiempo y espacio que necesita para llevar a cabo una búsqueda. Por otro lado, la eficacia mide cómo el sistema es capaz de cubrir las necesidades expresadas por el usuario. Para evaluar la segunda, existe una metodología muy usada en el campo, que se explica a continuación.

2.2.1 Paradigma Cranfield y TREC

El paradigma Cranfield provee una manera estándar de realizar una evaluación sobre un sistema para comprobar cómo este cumple con las necesidades de un usuario. La aparición de esta metodología data de los años sesenta. Los experimentos de Cranfield fueron unas investigaciones llevadas a cabo por Cyril W. Cleverdon, donde se comprobaban diferentes estrategias de indexación de lenguajes [8]. El objetivo de estos era diseñar una situación en la que se pudiera estudiar el rendimiento de sistemas de recuperación de información, evitando la influencia de factores externos que no fueran los propios sistemas, para favorecer la reproducibilidad [9]. Siguiendo este protocolo de evaluación, se utilizan colecciones de prueba que están formadas por una colección de documentos, un conjunto de tópicos y una serie de juicios de relevancia, que indican qué documentos son relevantes para cada tópico. Aunque esta metodología tiene sus detractores [10], otros muchos investigadores adoptaron el uso de colecciones de prueba para comparar el rendimiento de diferentes sistemas [11, 12, 13].

Estos experimentos se basan en tres importantes condiciones: primera, la relevancia de un documento está indicada por su proximidad con un tópico; segunda, un solo conjunto de

juicios es representativo de todo el conjunto de usuarios, es decir, los juicios son válidos para cualquier usuario; y tercera, la lista de los juicios es completa, se conocen todos los documentos relevantes para un tópico. Aunque estas condiciones no siempre son ciertas, pueden ser compensadas de alguna manera [7], por lo que esta metodología se ha convertido en un estándar para analizar la eficacia de los sistemas de recuperación de información [14, 15]. Siguiendo este procedimiento, cada sistema genera una lista de documentos en orden decreciente de relevancia para cada tópico. La eficacia de un sistema para un único tópico se calcula como una función de la lista de documentos relevantes. La eficacia del sistema en general se calcula como la media de todos los tópicos en la colección.

TREC

Text REtrieval Conference (TREC) [14] es una iniciativa organizada por el NIST desde 1992 con el objetivo de proveer soporte e infraestructura a la investigación en la recuperación de información. Con este objetivo, se liberan anualmente colecciones de prueba, es decir, un conjunto de documentos junto con los tópicos y los juicios de relevancia, para posteriormente ser utilizadas por los investigadores en este campo. La mayor parte de estas colecciones están formadas por documentos que contienen, mayormente, extractos de noticias y artículos de opinión, pero existen colecciones y tareas TREC en ámbitos como la química, la medicina, las patentes, etc.

2.2.2 *Pooling*

El principal problema del paradigma Cranfield es que es muy costoso y complejo construir una colección [16, 17]. Como se explicó anteriormente, una de las principales asunciones de los experimentos de Cranfield era que los juicios debían ser completos. Inicialmente, tanto las colecciones de Cranfield como otras que también surgieron en la época, estaban formadas por solamente unos pocos documentos[1], por lo que no suponía un gran esfuerzo juzgarlos todos. Debido al creciente tamaño de las colecciones, dejó de ser factible obtener juicios relevancia completos, es decir, establecer la relevancia de todos los documentos en la colección. Suponiendo un tiempo de treinta segundos para juzgar la relevancia de un documento, llevaría más de novecientos años juzgar todos los documentos para un único tópico en una colección como ClueWeb09[2], que está formada por más de un billón de documentos. Otro ejemplo de este esfuerzo se ilustra con la colección de TREC News Track 2018[3] que, haciendo las mismas asunciones que antes, llevaría mas de siete meses juzgar toda la colección para un solo tópico. Juzgar todos estos documentos se traduce en un coste muy elevado para los organizadores,

[1] http://ir.dcs.gla.ac.uk/resources/test_collections
[2] https://lemurproject.org/clueweb09
[3] https://trec.nist.gov/pubs/trec27/trec2018.html

ya que los asesores son, normalmente, agentes retirados de la Agencia Central de Inteligencia americana (CIA), y son mínimo tres personas diferentes las que juzgan los documentos para cada tópico, lo que requiere muchos recursos costear su trabajo.

Para poder paliar este problema, las colecciones modernas se construyen utilizando un proceso llamado *pooling* [7, 14, 18], que permite obtener bancos de prueba más grandes sin necesidad de juzgar todos los documentos de la colección [19]. Siguiendo esta aproximación, para cada tópico solo se juzga un subconjunto de toda la colección de documentos. A este subconjunto se le llama *pool*. Los documentos que no aparecen en el *pool* y que, por lo tanto, no son juzgados, se asumen como no relevantes. El proceso de construcción del *pool*, que se explica a continuación, hace que este supuesto sea razonable y que los juicios así construidos se puedan asumir como completos.

El Instituto Nacional de Estándares y Tecnología de los Estados Unidos (NIST), el organizador de las campañas TREC, libera un conjunto de documentos y una serie de tópicos a los participantes en la iniciativa. Después, estos participantes ejecutan sus propios modelos de recuperación de información para cada tópico y obtienen una lista ordenada por relevancia de documentos, que es enviada de vuelta a los organizadores. Estos utilizarán los resultados para construir los *pools* y después juzgar los documentos. Es importante destacar la diferencia entre un tópico, que es una representación de la necesidad de información de usuario, y una *query*, que es el tipo de dato que se le da a un sistema para obtener un resultado. Las necesidades de los usuarios se representan en formato de tópico para así poder describir adecuadamente una necesidad de información. El contenido de uno de estos tópicos se ilustra en el Listado 2.1. El conjunto de documentos de una colección debe ser representativo de los tipos de documentos que los usuarios buscarán, por lo que debe reflejar la variedad de estilos, formatos, temas, etc.

```
<head> Tipster Topic Description

<num> Number: 92

<dom> Domain: Military

<title> International Military Equipment Sales

<desc> Description:

Document will identify a proposed or recently concluded sale of
    military equipment in the international arms market.

<smry> Summary:

Document will identify a proposed or recently concluded official
    sale of military equipment between countries in the
```

```
    international arms market.

<narr> Narrative:

To be relevant, document will identify a proposed or recently
    concluded sale of military equipment, to include weapons
    platforms, munitions, spares, ancilliary equipment, technology,
    and services.  The equipment and quantities must be identified,
    and recent should be defined as within the previous 12 months,
    but the price and conditions of sale are optional information.
    Any international sale, including transfers between NATO members
    or equipment being provided under military aid programs, is
    relevant.  However, political discussions of international
    military equipment sales, such as debate of the Iran-Contra
    Affair, discussion of the UN embargo against South Africa, or
    Congressional expressions of hostility against equipment sales
    to Arab states, unless linked to a specific transaction, are not
    relevant.

<con> Concept(s):

1. FMS, foreign military sales

<fac> Factor(s):

<time> Time: current
</fac>

<def> Definition(s):

</top>
```

Listado 2.1: Ejemplo de tópico de TREC.

Una vez obtenidos los resultados de los participantes, se procede a la creación de los *pools*. Para crearlos, se obtiene la unión de los primeros k documentos de cada resultado enviado por los participantes, que se llaman *runs*. Este proceso se realiza para cada tópico, se contruye un *pool* por cada uno. El parámetro k es la profundidad del *pool* y típicamente tiene un valor de 100. Como los resultados enviados por los participantes están en orden decreciente de relevancia estimada, los documentos más relevantes tienen más posibilidades de ser obtenidos por varios participantes. Debido a esto, el tamaño de un *pool* para un tópico normalmente es más pequeño que la cota máxima teórica, que sería $k \times no.\ runs$ [7]. Por ejemplo, para la colección TREC News 2018, mencionada anteriormente, gracias a este proceso, esta cota máxima es de 2300 documentos, lo que ilustra los beneficios del *pooling*. Una vez obtenidos

todos los *pools*, se juzga la relevancia de los documentos contenidos en ellos. Utilizando estos juicios recién obtenidos, se evalúa la calidad de los resultados enviados por los participantes.

Con el proceso de *pooling* sabemos que obtenemos los juicios de relevancia para los documentos que constituyen el *pool*, pero algunos documentos pueden quedar fuera del *pool*, con lo que se viola el precepto de tener juicios completos. Esto no supone un obstáculo para la evaluación, porque no estamos interesados en el valor absoluto de una métrica, sino que el objetivo es poder comparar cuando un sistema es mejor que otro [20]. Para poder cumplir con este objetivo, es necesario que los juicios de relevancia no estén sesgados. Esto significa que estos juicios produzcan resultados no discriminativos sobre los sistemas de los participantes [21].

2.2.3 Estrategias de *pooling*

Una vez construido un *pool* de documentos, es necesario juzgar la relevancia de estos. Históricamente, en las campañas TREC se han juzgado estos documentos siguiendo una estrategia muy simple, i.e., DocID [14]. Con este método, los documentos son simplemente ordenados por su identificador, y en este orden se presentan a los asesores que juzgan su relevancia. Sin embargo, recientemente se ha realizado mucho trabajo e investigación en el diseño de nuevas estrategias que imponen un orden de juicio para así poder aprovechar al máximo el tiempo de los asesores, haciendo que se juzguen documentos relevantes más temprano en el proceso. En particular, en la iniciativa TREC Common Core Track de 2017 [22] se aplicó por primera vez un algoritmo de *pooling* basado en Bandidos Bayesianos (Bayesian Bandits, en inglés) [23, 24] lo que ha sido demostrado como una buena estrategia de *pooling* que tiene un mejor rendimiento que el estado del arte.

2.2.4 *Query variants*

Otra de las principales desventajas de esta aproximación para construir colecciones de prueba es que existe la necesidad de tener sistemas participantes para construir el *pool*, condición que no siempre se cumple. Por ejemplo, puede darse el caso de que algunos investigadores tengan la necesidad de construir una colección con documentos de un dominio específico [25], y no posean suficientes recursos para contratar un número de asesores que permita obtener los juicios en un tiempo prudencial, y además pudieran necesitar algunos juicios de relevancia antes de la competición para ser ofrecidos a los participantes como datos de entrenamiento para sus modelos.

Para atacar este problema se puede aplicar una estrategia a la hora de obtener los sistemas que posteriormente se utilizarán para constuir el *pool*. Como se ha explicado en secciones anteriores, la entrada a un sistema de recuperación de información convencional es una *query*, que representa la necesidad de información del usuario. Siguiendo esta estrategia, el rol de los

participantes se sustituye por diferentes *query variants* y diferentes modelos de recuperación de información (como BM25 [26, 27], Vector Space Model (VSM) o Language Models (LM)), que se combinan para producir los sistemas utilizados para construir el *pool*. Por ejemplo, si se combinan 5 variantes con 5 sistemas de recuperación diferentes, se podrían obtener 25 sistemas nuevos.

Los diferentes modelos de retrieval empleados pertenecen a diferentes familias de modelos que representan diferentes aproximaciones para resolver el problema de la recuperación. BM25 es un modelo probabilístico, familia de modelos que se basan en el Principio de Clasificación de Probabilidad (Probability Ranking Principle, en inglés) [28]. Vector Space Model es un modelo algebraico que representa los documentos y las consultas como vectores. Finalmente, Language Models son una familia de modelos probabilísticos que utilizando un modelo del languaje (languaje model, en inglés) para estimar la relevancia de un documento.

Estas *query variants* (se mantiene el nombre en inglés, ya que es el utilizado generalmente en la literatura sobre RI) son *queries* reformuladas que se construyen a partir de la *query* original. Existen multitud de estrategias y modelos para construirlas. Por ejemplo, se pueden construir añadiendo a la *query* original nuevos términos extraídos del contenido de los tópicos [13], o especificadas manualmente por un experto [29]

En los objetivos especificados para este proyecto (Sección 1.2), es de suma importancia elaborar una plataforma que pueda construir colecciones sin la necesidad de sistemas participantes. Debido a que la utilidad de la estrategia aquí explicada está demostrada [13, 29], es la que se implementará en esta plataforma.

Capítulo 3

Tecnologías y herramientas

La elección de las tecnologías y herramientas utilizadas durante el desarrollo del proyecto es esencial para poder completarlo con éxito. En este capítulo se explican y se justifican las tecnologías y herramientas empleadas. En primer lugar se exponen las tecnologías utilizadas para la gestión de la persistencia y la base de datos. A continuación se muestran también para la aplicación web. Además, se describen las herramientas de soporte utilizadas a lo largo de todo el desarrollo. Por último, se dedica una sección aparte para detallar la elección de las herramientas utilizadas para realizar la indexación y la búsqueda de documentos de una colección.

3.1 Persistencia

En esta sección se comentan los aspectos relacionados con las tecnologías empleadas para el desarrollo de la capa de persistencia de la aplicación.

3.1.1 RDBMS

Los sistemas de gestión de bases de datos relacionales (*Relational DataBase Management Systems* o RDBMS) permiten la gestión y el trato de información relacional, permiten crear, modificar y consultar bases de datos que siguen un esquema relacional.

Existen varias soluciones de este tipo en el mercado. Dos de las más extendidas son PostgreSQL[1] y MySQL[2].

PostgreSQL

PostgreSQL es un gestor de bases de datos relacionales (RDBMS). Es software libre y es el más potente del mercado. Está orientado a objetos y es multiplataforma, con lo que pue-

[1] https://www.postgresql.org
[2] https://www.mysql.com

de utilizarse en diferentes sistemas operativos. A continuación se enumeran algunas de sus principales características:

- Soporta un lenguaje SQL muy próximo al estándar ISO/IEC.
- Tiene una documentación muy completa.
- Proporciona transacciones 100% ACID.
- Proporciona una alta concurrencia de acceso.

MySQL también es una opción a tener en cuenta, pero existen ciertas diferencias [30] que hacen a PostgreSQL una elección más apropiada para este proyecto:

- Total soporte ACID: mientras que MySQl solo garantiza transacciones ACID bajo algunas condiciones, PostgreSQL las garantiza siempre.
- Rendimiento: PostgreSQL se comporta mejor que MySQL en la ejecución de consultas complejas.
- Seguridad: PostgreSQL ofrece más opciones en este sentido que MySQL. La primera ofrece diferentes roles para gestionar los permisos, así como coneciones SSL y seguridad a nivel de fila.

Todas estas características hacen a PostgreSQL una opción ideal para utilizar en este proyecto.

3.1.2 Hibernate

Hibernate[3] es un mapeador objeto/relacional (ORM), de código abierto, creado por un grupo de desarrolladores a finales de 2001. Es el ORM más utilizado actualmente. Se utiliza con el objetivo de simplificar el desarrollo de la capa de persistencia, evitando utilizar directamente JDBC. Para esto, permite utilizar objetos persistentes en lugar de manipular directamente datos de la BD. En general, ofrece las siguientes características:

- Asocia clases persistencias con tablas de una BD relacional, haciendo transparente la API de JDBC y el lenguaje SQL.
- Soporta relaciones entre objetos y herencia.
- No oculta el tipo de BD, que se sabe que es relacional, pero sí la BD concreta (Oracle, PostgreSQL, etc.).

[3] http://www.hibernate.org

- Implementa la API de persistencia de Java (JPA), y además ofrece una propia por si la primera no fuera suficiente.
- Ofrece un lenguaje de consultas, con semántica orientada a objetos y una sintaxis parecida a SQL.
- Aunque el principal objetivo es no manipular los datos de la BD directamente, si en algún caso fuera necesario, permite lanzar consultas SQL.

Por las características expuestas y la familiaridad del alumno con la tecnología, se ha optado por esta elección.

3.1.3 Spring

Spring[4] es un framework de código abierto, creado por Rod Johnson, para el desarrollo de aplicaciones Java. Su objetivo es facilitar el desarrollo de aplicaciones Java EE, promoviendo buenas prácticas de diseño y de programación. Simplifica el uso de muchas de las APIs de Java EE y ofrece alternativas a algunas de estas.

Es un framework modular, es decir, permite usar diferentes módulos sin comprometerse con el uso del resto, y tiene soporte tanto para la capa modelo como para la capa de interfaz web. En este proyecto se han utilizado las siguientes características de Spring en cuanto a la capa de persistencia:

- Contenedor de inyección de dependencias (DI): es núcleo de Spring. Permite establecer las dependencias entre los objetos Java y se encarga de satisfacerlas, sin tener que ser hecho explícitamente.
- Gestión declarativa de transacciones: ofrece una implementación del paradigma de la programación orientada a aspectos (AOP) con la cual permite que la gestión de transacciones sea transparente al programador.

De igual modo que Hibernate, debido a la familiaridad del alumno con la tecnología y al asentado uso de esta, se ha optado por utilizarla para el desarrollo de los casos de uso de la capa de servicios de la aplicación.

3.2 Aplicación web

La interfaz de interacción con los usuarios es una aplicación web que expone las funcionalidades ofrecidas por la aplicación. A continuación se exponen las tecnologías empleadas para desarrollar este componente.

[4] http://projects.spring.io/spring-framework

3.2.1 Spring MVC

Spring MVC[5] es un framework Java usado para desarrollar aplicaciones web. Está construido sobre la API de *servlet*, y se incluye con el framework de Spring desde las primeras versiones de este. Utiliza el patrón MVC (Modelo-Vista-Controlador), donde el modelo es el encargado de ofrecer la gestión de los datos de la aplicación, la vista se encarga de la representación visual de la aplicación, por último, el controlador se encarga de enlazar la vista con el modelo. Entre sus principales características destacan:

- Ofrece una división limpia entre el modelo, la vista y el controlador.
- Es muy flexible.
- Los controladores de Spring MVC se configuran mediante IoC, lo cual los haces fácilmente testeables en integrables con otros objetos que estén en el contexto de Spring.

Debido a que este framework se integra de manera natural con el framework Spring utilizado para desarrollar la capa modelo y con el empleado para el desarrollo de la vista de la aplicación (Thymeleaf) es un perfecto candidato para ser empleado en este proyecto.

3.2.2 Thymeleaf

Thymeleaf[6] es un motor de plantillas, una tecnología que permite definir una plantilla para obtener un nuevo documento. Para proyectos desarrollados en Java existen numerosas alternativas de este estilo, pero para este proyecto se ha utilizado Thymeleaf debido a que las características que se exponen a continuación lo convierten en una elección adecuada para este trabajo:

- Se integra perfectamente con Spring.
- Ofrece soporte para la internacionalización.
- Facilidad de uso.
- Alto rendimiento, ya que implementa un sistema de caché.
- Diseñado para XML, XHTML y HTML5, aunque es extensible a otros formatos.

Por esto y por ser un framework muy flexible, ha sido utilizado en este proyecto.

[5] https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html

[6] https://www.thymeleaf.org

3.2.3 HTML, CSS y Bootstrap

HTML es un estándar mantenido por el W3C que se ha convertido en el estándar de facto para la visualización de páginas web. Es el lenguaje que todos los navegadores modernos soportan.

CSS es un lenguaje empleado para definir el estilo de presentación de un documento HTML.

Bootstrap es una biblioteca para el diseño de páginas atractivas. Incluye numerosas clases CSS que permiten dotar a los elementos de la web de diferentes estilos, sin tener que volver a desarrollarlos.

Se han utilizado estas tecnologías para el diseño y la presentación de las diferentes vistas de la aplicación web.

3.2.4 D3.js

D3.js es una librería en JavaScript para crear gráficos e infografías interactivos en páginas web. En este proyecto se ha utilizado esta biblioteca para realizar las gráficas que permiten comparar los resultados de los experimentos ejecutados.

3.3 Herramientas de soporte

Anteriormente se han detallado las tecnologías empleadas en los diferentes componentes del sistema. A continuación se detallan las herramientas de soporte que se han utilizado a lo largo de todo el desarrollo.

3.3.1 Gestión, versionado y configuración de software

En este proyecto se han empleado diversas herramientas para poder gestionar el software correctamente y también para llevar un control sobre los cambios implementados.

Eclipse

Eclipse[7] es un entorno de desarrollo integrado (IDE) multilenguaje con un espacio de trabajo y un sistema plug-in ampliable para personalizar el entorno. Como es un IDE, se espera que ofrezca las siguientes características:

- Editor de código adaptado al lenguaje en cuestión, en este caso Java.
- Compilación integrada.

[7] https://www.eclipse.org

- Debugging: Eclipse incluye un debugger para poder analizar la ejecución del código instrucción por instrucción.
- Integración con Git: debido al uso de Git en este proyecto, es deseable un IDE que se integre bien el sistema de control de versiones.
- Refactorización: ofrece herramientas para facilitar la refactorización de código ya existente.
- Ahorro de tiempo: ofrece diferentes opciones que favorecen el ahorro de tiempo a la hora de programar, como el autocompletado o los atajos de teclado.

Además, trabajar con un IDE en el desarrollo del proyecto aporta las siguientes ventajas:

- Facilidad para entender grandes proyectos.
- Facilidad para corregir grandes proyectos.
- Facilidad para mejorar grandes proyectos.
- Hacer el trabajo diario más fácil y productivo.

Debido a esto y a que es el entorno de desarrollo más extendido para el lenguaje de programación Java, se ha optado por utilizarlo en este proyecto.

Git y GitLab

En un proyecto de desarrollo software como este es esencial el uso de prácticas de control del versionado de software. Las principales ventajas de hacer esto son:

- Todos los archivos están etiquetados.
- Todos los artefactos construidos están controlados.
- Realizar tareas de gestión de errores apropiadas.
- Controlar los cambios en el código.
- Controlar los entornos de pruebas.

Además, en un entorno de trabajo en equipo esto provee una manera de que varias personas trabajen sobre lo mismo sin producir interferencias en el trabajo de los demás.

Git[8] es un sistema de control de versiones distribuido desarrollado por Linus Torvalds. Es fácil de aprender y además tiene un rendimiento muy bueno. Hoy en día es el gestor de versiones más empleado. Además, debido a su extendido uso, cuenta con una gran comunidad de desarrolladores, lo que permite su integración con otras herramientas. Este gestor ofrece ciertas ventajas, enumeradas a continuación, sobre otros gestores de control de versiones como SVN[9], que han hecho que Git sea la elección para este proyecto:

- Posibilidad de trabajar sin conexión a internet. Git contempla el uso de repositorios locales y de repositorios remotos, al contrario que SVN, donde solo existe uno remoto, lo cual permite trabajar sin la necesidad de estar conectado a internet.
- Desarrollo no lineal: permite crear diferentes ramas (*branches*), así como herramientas para navegar entre ellas y visualizar el estado del repositorio.

GitHub[10] y GitLab[11] son las dos plataformas de gestión de repositorios que se han posicionado como las más empleadas en el mercado actualmente. No existen diferencias notables entre estas dos plataformas, pero se ha escogido GitLab ya que permite la creación de repositorios privados. Actualmente GitHub también ofrece esta característica, aunque con ciertas restricciones[12], pero en el momento de comenzar el desarrollo de este proyecto, únicamente GitLab lo ofrecía.

GitFlow

En un entorno de trabajo en equipo, desarrollar nuevas funcionalidades de una aplicación puede introducir cierta inestabilidad en el proceso, lo cual puede ser muy destructivo si todos los miembros del equipo trabajan sobre la misma línea de desarrollo. Utilizar diferentes flujos de desarrollo ayuda a paliar estos problemas, haciendo que la organización del desarrollo sea más cómoda.

GitFlow es un modelo de creación de ramas (*branching model*, en inglés) diseñado por Vincent Driessen [31]. Es un flujo que propone diferentes tipos de ramas según el propósito del desarrollo que se haga sobre ellas. Dentro de este marco existen cinco tipos de ramas: *master*, *develop*, *feature branch*, *release branch* y *hotfix branch*. Las nuevas funcionalidades de la aplicación se desarrollan en las *feature branches*, utilizando una rama diferente para cada funcionalidad. Estas se crean a partir de la rama *develop* y se vuelven a integrar en esta una vez terminadas. Una vez se han desarrollado todas las funcionalidades nuevas contempladas para

[8] https://git-scm.com
[9] https://subversion.apache.org
[10] https://github.com
[11] https://about.gitlab.com
[12] https://github.com/pricing

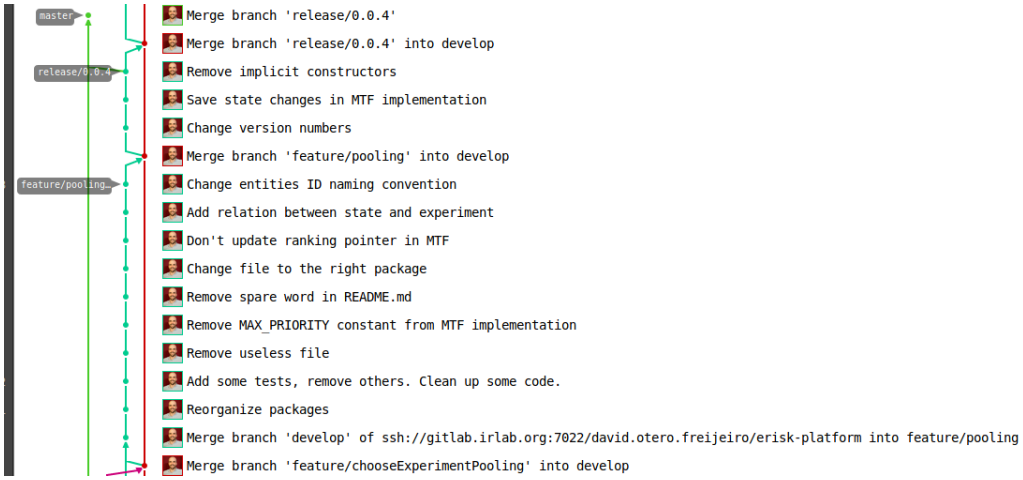

Figura 3.1: Ejemplo de flujo de trabajo seguido con GitFlow - eRisk Platform.

una versión, se crea una *release branch* a partir de la rama *develop* y se despliega en un entorno apropiado de pruebas. Cuando la versión esta finalizada, se integra tanto en *develop* como en *master*. Esta última contiene ahora el código que finalmente será llevado a producción, y contiene siempre la versión más reciente de la aplicación. Por último, si se detectara algún fallo importante en una versión ya integrada en *master*, se puede crear una rama *hotfix* a partir de esta para solucionarlo, y después integrarla otra vez tanto en *master* como en *develop*. En la Figura 3.1 se ilustra un ejemplo de la aplicación de GitFlow en este proyecto.

En este proyecto se ha seguido el flujo aquí presentado. No obstante, debido a ser un proyecto desarrollado por una única persona, no se han podido extraer todas las ventajas de trabajo en equipo.

Apache Maven

En el proceso de desarrollo software existen muchas tareas rutinarias que hacen los desarrolladores, como puede ser la compilación del código, la ejecución de pruebas, el despliegue de la aplicación, entre otras. La automatización de estas tareas permite acelerar todo este proceso, así como también aprovechar mejor el dinero y el tiempo invertidos.

Apache Maven[13] es una de las herramientas más utilizadas para la gestión de software en entornos Java. Promueve la comprensión y productividad en un proyecto aplicando patrones a todo el proceso de construcción del software. No es solo una herramienta de construcción, sino que es una herramienta de gestión y comprensión de proyectos. Ofrece servicios para gestionar la construcción del software, la documentación, las dependencias del proyecto, entre otras cosas. Favorece la reutilización de componentes mediante el uso de arquetipos.

[13] https://maven.apache.org

Maven utiliza un Project Object Model (POM) en formato XML para describir el proyecto, sus dependencias con otros módulos y componentes externos así como el orden de construcción de dichos elementos. Viene con objetivos predefinidos para realizar ciertas tareas comunes en todo proyecto, como la compilación o el empaquetado.

3.3.2 *Testing* e integración continua

La implementación de pruebas automatizadas es muy importante para comprobar por un lado, que el software cumple con los requisitos especificados y, por otro lado, si es útil según las necesidades del usuario de la aplicación. De esta manera, aunque no se pueda afirmar con total seguridad que nuestra aplicación no tiene ningún error, sí que se puede aumentar la confianza en un buen funcionamiento de esta.

JUnit

JUnit[14] es el framework más extendido para realizar pruebas unitarias repetibles en aplicaciones Java. Permite llevar a cabo la ejecución de clases Java de manera controlada, para así poder instrumentalizar estas ejecuciones y evaluar su comportamiento y comprobar si se produce de la manera esperada.

En este proyecto se ha utilizado esta tecnología para construir tests de unidad y de integración sobre la capa modelo de la aplicación.

Selenium

Selenium[15] es un framework para hacer pruebas de una aplicación web, automatizando el proceso que seguiría un usuario final utilizando el navegador. Esta herramienta ofrece soporte para desarrollar pruebas utilizando los principales navegadores existentes en el mercado.

Las pruebas funcionales como estas comprueban el correcto funcionamiento de la aplicación usando directamente su interfaz. Normalmente, cada caso de prueba se corresponde con una secuencia de navegación que representa una funcionalidad concreta que la aplicación ofrece al usuario. A diferencia de las pruebas de unidad o de integración, estas pruebas trabajan directamente contra la interfaz de la aplicación como lo haría un usuario real, por lo que potencialmente son más completas, aunque más costosas de ejecutar y de desarrollar.

En este proyecto se ha utilizado la herramienta Selenium para realizar pruebas funcionalidades contra la interfaz web de la aplicación.

[14] https://junit.org
[15] https://www.seleniumhq.org

Jenkins

La integración continua es una práctica de desarrollo software en la que los miembros del equipo de desarrollo integran su trabajo con frecuencia, normalmente cada día, lo que lleva a tener varias integraciones por día. Esto permite que los errores se detecten lo antes posible, mejorando la calidad del código. Además de esto, esta práctica ofrece ciertas ventajas:

- Los desarrolladores pueden detectar errores y solucionarlos continuamente, evitando errores de última hora antes de una entrega.
- Pruebas inmediatas de todos los cambios hechos en el proyecto.
- Siempre se encuentra disponible una construcción del software libre de errores y lista para ser desplegada, por lo que en caso de fallo siempre se puede revertir a un estado de la aplicación que funcione correctamente.

Jenkins[16] es una herramienta de integración continua (*Continuous Integration* o CI) escrita en Java. Permite la ejecución de proyectos basados en Ant y en Maven, entre otros. Además, ofrece una interfaz para poder configurarlo fácilmente. También ofrece enlaces permanentes (*permalinks*) para acceder a la última integración realizada, la última integración fallida, la última integración exitosa, para que puedan ser fácilmente enlazadas desde otros lugares. En la parte inferior de la Figura 3.2 se pueden ver estos enlaces, además de otra información relacionada con las integraciones del proyecto.

Además de todo esto, en el caso concreto de Jenkins, permite integrarse con herramientas de control de versiones como Git y tiene plugins para integrarse con otro tipo de herramientas, lo que enriquece aún más el uso de la herramienta. Por todo esto, aplicar integración continua en el proceso de desarrollo de software hace que aumente considerablemente la confianza que se puede tener en el producto desarrollado. En la Figura 3.2 se puede ver un ejemplo de la interfaz de Jenkins. En esta vista se ilustra una gráfica con el resultado de todas las integraciones (a la derecha), así como el resultado del análisis de SonarQube, acceso al *workspace* donde se encuentran los artefactos generados, etc.

Por todos estos motivos se ha decido aplicar un flujo de integración continua en este proyecto, utilizando para ello la herramienta Jenkins.

3.3.3 Planificación, seguimiento e inspección continua

Para llevar a buen puerto un proyecto de desarrollo software como este, es necesario, en primer lugar, realizar una buena planificación del trabajo que se hará, empleando las herramientas adecuadas. En segundo lugar, también es imprescindible llevar un seguimiento sobre

[16] https://jenkins.io

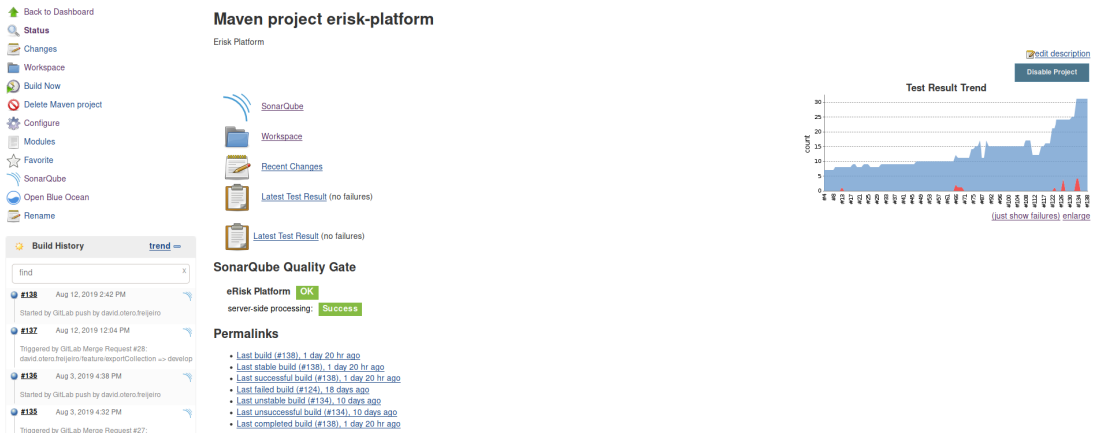

Figura 3.2: Ejemplo de Integración Continua con Jenkins - eRisk Platform.

el proceso de desarrollo del proyecto, para evitar desviaciones no esperadas y cumplir con lo esperado. En este apartado se describen las herramientas utilizadas para estas tareas.

Taiga

Taiga[17] es una plataforma creada con el objetivo de prestar soporte a la gestión de proyectos ágiles de desarrollo. Con esta herramienta se pueden emplear dos de las metodologías ágiles más extendidas actualmente: Scrum y Kanban, ambas integradas en la plataforma. Además de esto, ofrece una fácil integración con otros servicios como GitLab. Por esto, y porque está diseñada para trabajar con la metodología seguida en este proyecto (ver 4.1), ha sido la herramienta escogida para hacer la gestión del proyecto.

En la Figura 3.3 se ilustra un ejemplo de un proyecto gestionado con Taiga, donde se puede ver una pequeña lista de historias de usuario, una gráfica sobre el progreso del proyecto (*burndown chart*), así como otra información sobre este: los puntos de historia totales estimados, los asignados hasta ese momento, los completados, y los *sprints* planificados hasta la fecha junto con las historias asignadas a cada uno. En este caso todos los *sprints* están completados, por esta razón esta información no se ve en la columna de la derecha.

SonarQube

SonarQube[18] es una herramienta diseñada para realizar inspección continua (*Continuous Inspection*, CI) sobre la calidad del código del proyecto.

La calidad de un producto software se puede entender desde dos perspectivas: una externa (o funcional), que describe lo bien que el software se adapta a sus requisitos funcionales; y

[17] https://taiga.io
[18] https://www.sonarqube.org

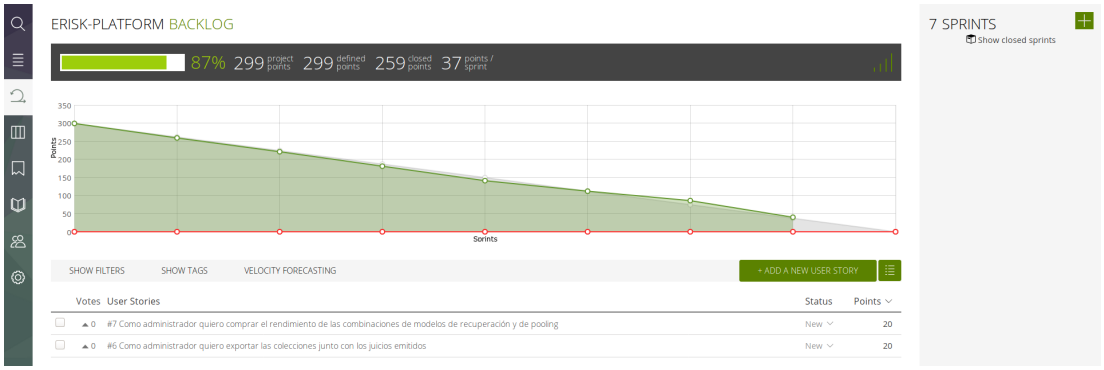

Figura 3.3: Ejemplo de un proyecto gestionado en Taiga - eRisk Platform.

otra interna, que describe la calidad de atributos internos del código: robustez, facilidad de mantenimiento, entre otros. La inspección continua es un práctica cuyo objetivo es hacer la calidad interna del código una parte integrada en todo el ciclo de desarrollo software. Esta práctica permite hacer a todas las personas interesadas en el producto partícipes de su calidad y cómo este avanza. Además, hace posible los ciclos de *feedback* más cortos, haciendo que se resuelvan lo antes posible los problemas de calidad. La inspección continua es un paradigma que convierte a la calidad del producto un factor importante desde el principio y durante todo el ciclo de vida del producto, no solo al final del desarrollo.

En la Figura 3.4 se ilustra parte de toda la información que esta herramienta puede aportar sobre la calidad del proyecto en general: líneas de código totales y desglosadas por lenguaje, cobertura y número de tests de unidad, porcentaje de código duplicado, número de errores graves y *code smells*, etc.

De igual modo que con la integración continua y Jenkins, se ha implementado un flujo de inspección continua en este proyecto, utilizando la herramienta SonarQube.

3.4 Indexación y búsqueda de ficheros

Para cumplir con los requisitos funcionales de esta aplicación sobre la construcción de colecciones de documentos, es necesario implementar la indexación y búsqueda de estos. Para esta tarea, los sistemas de persistencia expuestos (PostgreSQL) no son apropiados, por lo que es necesario escoger otra tecnología o herramienta que se adapte mejor al problema.

3.4.1 Apache Lucene

Apache Lucene[19] es una librería de software libre, escrita en Java, desarrollada para proveer servicios de indexación y búsqueda de documentos. También ofrece una extensión para

[19] https://lucene.apache.org

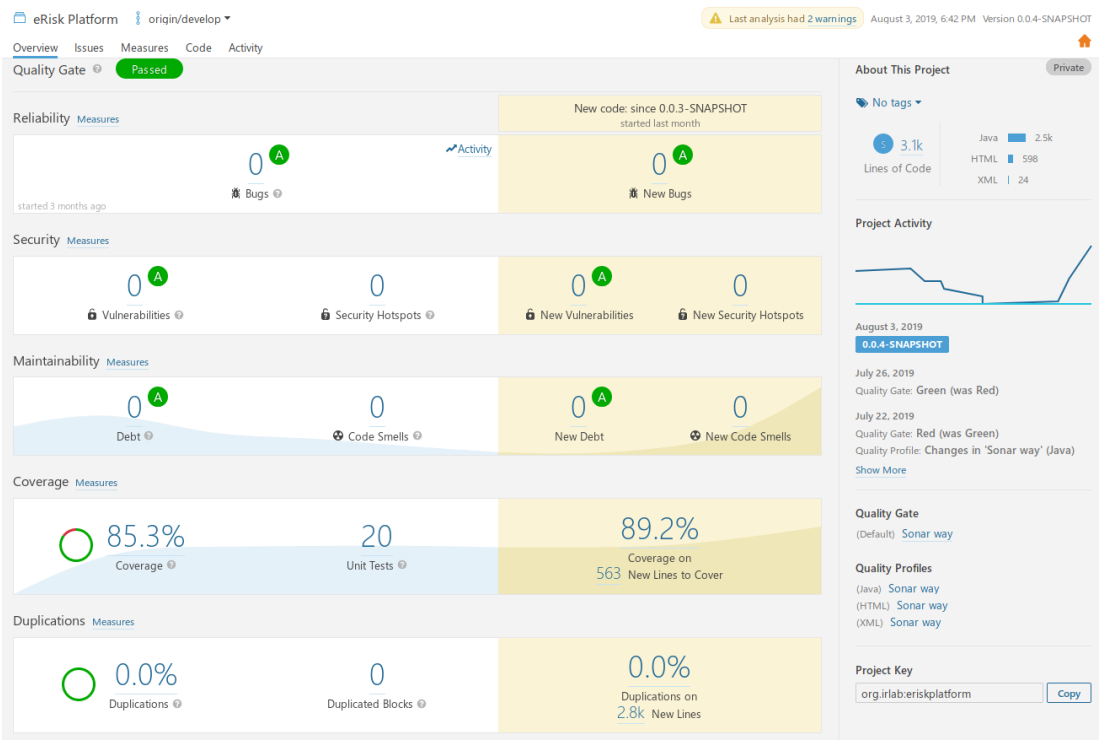

Figura 3.4: Ejemplo de Inspección Continua con Sonar - eRisk Platform.

ser utilizada en Python. Es la tecnología más madura en Java para estas tareas, por eso ha sido la elección para este proyecto. Existen otras tecnologías que podrían servir para el mismo fin, como Elasticsearch[20], que es un servidor web construido sobre Lucene, por lo que ofrece otras muchas funcionalidades que no son necesarias para este proyecto, y Lucene permite utilizar únicamente las funcionalidades necesarias.

Con las tecnologías y herramientas expuestas en este capítulo se han desarrollado los diferentes componentes que conforman el producto final, siguiendo un proceso que se detalla en el siguiente capítulo.

[20] https://www.elastic.co

Capítulo 4

Metodología y gestión de proyecto

En este capítulo se expone lo relacionado con el proceso y la metodología aplicados durante el desarrollo del proyecto. En primer término, se explica y justifica la metodología utilizada, así como algunas adaptaciones al presente proyecto. Por otro lado, se tratan las cuestiones referentes a la gestión del proyecto como la estimación de costes y de la duración, la planificación y la gestión de riesgos.

4.1 Metodología

La metodología de desarrollo establece un marco de trabajo que guía todo el proceso de construcción del proyecto. Esta elección es vital, ya que de ella depende la correcta consecución de los objetivos establecidos. Por lo tanto, es necesario realizar una decisión bien fundamentada.

4.1.1 Elección de la metodología

Existen muchas metodologías desarrolladas para su aplicación en proyectos de desarrollo software. El estudio de estas metodologías es un área dentro de la ingeniería del software. El uso de las metodologías conocidas como metodologías ágiles se ha extendido mucho durante los últimos años. Estas se caracterizan por el uso de ciclos de vida iterativos que se suceden a lo largo del tiempo, para construir un producto funcional al final de cada ciclo. El objetivo de esta división en iteraciones es satisfacer los requisitos fundamentales del sistema en las primeras iteraciones, para dedicar las siguientes fases a añadir funcionalidades o refinar características. Esto supone una serie de ventajas, enumeradas a continuación:

- La construcción de productos intermedios a lo largo de todo el desarrollo permite una mejor interacción con el cliente, que puede ver el estado del producto antes de su total finalización.

- Minimizar los riesgos y los costes no previstos, ya que se pueden detectar en una fase más temprana.
- Desarrollar el software en iteraciones permite realizar una mejor gestión del cambio y adaptarse mejor a los cambios de requisitos.
- La construcción por componentes permite una mayor modularidad y un menor acoplamiento en el software.

Teniendo en cuenta estas ventajas, se ha decidido emplear una metodología ágil. A continuación se explica en líneas generales la filosofía de las metodologías ágiles y, más concretamente, se explica la metodología escogida para este proyecto, Scrum.

Manifiesto Ágil

El Manifiesto Ágil es un documento redactado en 2001 por expertos en el campo que proponían un cambio conforme a los modelos tradicionales. Este documento propone cuatro valores, y sobre estos cuatro valores se sustenta la filosofía de las metodologías ágiles. Estos cuatro postulados son:

- Individuos e iteraciones prevalecen sobre procesos y herramientas.
- Software funcionando prevalece sobre documentación exhaustiva.
- Colaboración con el cliente prevalece sobre negocación contractual.
- Respuesta ante el cambio prevalece sobre seguir un plan.

Estos cuatro valores dan lugar a los siguientes doce principios, que definen el marco de trabajo de cualquier equipo ágil:

1. Satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptar bien los cambios, aun cuando estos ocurran en etapas tardías del desarrollo.
3. Entrega frecuente de software funcional.
4. Los responsables del negocio y los desarrolladores deben estar en constante contacto.
5. Motivación en el trabajo.
6. Conversación cara a cara como método de comunicación, dejando la documentación para casos especiales.
7. La forma para medir el progreso se basa en la cantidad de funcionalidades desarrolladas.

8. Desarrollo y trabajo a un ritmo constante de manera indefinida.

9. Atención continua a la excelencia técnica y al buen diseño.

10. Simplicidad, minimizar la cantidad de carga de trabajo.

11. Equipos auto-organizados.

12. Perfeccionar la efectividad del equipo según la experiencia adquirida con el tiempo.

Aunque puede ser que algunos principios no apliquen en este proyecto debido a factores como que el equipo de desarrollo es de una sola persona, o que el tiempo dedicado no es constante, se ha elegido una metodología ágil debido a la cantidad de ventajas que aporte sobre una metodología tradicional en este caso. A continuación se detalla la metodología escogida, Scrum.

4.1.2 Scrum

Scrum [32] es un framework de desarrollo ágil creado por Ken Schwaber y Jeff Sutherland. Scrum no establece unas prácticas obligatorias, sino que es un marco de trabajo dentro del cual se pueden emplear diferentes procesos y técnicas. Está fundamentado en tres pilares: transparencia, inspección y adaptación. La mayor ventaja de Scrum es que es una metodología facil de aprender.

En la Figura 4.1 se puede ver un diagrama con el proceso Scrum resumido.

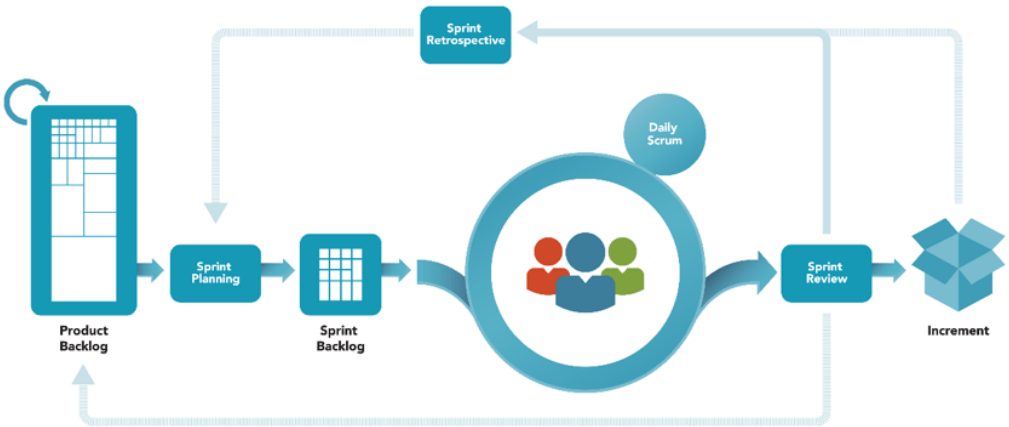

Figura 4.1: Proceso de Scrum.

Sprint

El *sprint* constituye el corazón de Scrum. Es un evento que dura entre una y cuatro semanas, y actúa como contenedor del resto de eventos: *Sprint Planning*, *Daily Scrum*, *Sprint Review* y *Sprint Retrospective*, que se detallarán más adelante. Durante cada *sprint* se desarrolla un incremento del producto, que constituye un elemento funcional y usable, que es potencialmente entregable al cliente. El desarrollo del producto se lleva a cabo en varios *sprints*, que se suceden a lo largo del tiempo. Es recomendable que la duración de estos sea siempre la misma y que no se introduzcan eventos no contemplados, para reducir la aparición de riesgos.

Roles

Todos los miembros del equipo deben desempeñar un rol dentro de este marco de trabajo. Los diferentes roles dentro de Scrum son los siguientes:

- *Product Owner*: es el representante de los intereses del cliente. Todo el equipo debe respetar sus decisiones. Es responsable de: expresar con claridad los requisitos del produto y los elementos del *Product Backlog*, priorizar los elementos del *Product Backlog* para alcanzar los objetivos estipulados y asegurarse de que los elementos del *Product Backlog* son entendidos por todos los miembros del equipo

- *Scrum Master*: es el encargado de promover los valores y las prácticas de Scrum. Debe asegurarse de que estas se aplican de una manera correcta. Además, también debe eliminar los impedimentos que tenga el equipo de desarrollo para poder realizar su trabajo.

- *Equipo de desarrollo*: es el encargado de desarrollar los incrementos del producto en cada *sprint*. Los miembros del equipo son los únicos responsables de establecer el trabajo que se llevará a cabo en cada iteración, es decir, de elaborar el *Sprint Backlog*. Debe ser un equipo que tenga todas las habilidades necesarias para lograr los objetivos y no tiene líder, es un equipo auto-organizado. En cuanto a su tamaño, debe ser lo suficientemente pequeño para permanecer ágil, pero lo suficientemente grande para poder cumplir con los objetivos establecidos. Típicamente tiene entre 3 y 9 miembros.

Eventos

Dentro de Scrum se contemplan diferentes tipos de eventos predefinidos, que se realizan con el objetivo de crear cierta regularidad a lo largo del proceso de desarrollo, para minimizar así la aparición de sucesos no contemplados que suponen un riesgo en cuanto a la consecución de los objetivos. Estos eventos son los siguientes:

- *Sprint Planning*: esta reunión se lleva a cabo al inicio de cada *sprint*. No puede durar más de ocho horas. En ella se planifica el trabajo que realizará durante el *sprint* y se establecen los objetivos del mismo.

- *Daily Scrum*: es una reunión de quince minutos que se realiza todos los días durante el *sprint*. En esta reunión se inspecciona el trabajo realizado y el estado del producto conforme a la consecución de los objetivos. Se plantean diferentes preguntas para ayudar a aclarar dichas cuestiones: ¿qué he hecho hoy que ha ayudado a alcanzar los objetivos?, ¿qué haré mañana que ayude a alcanzar los objetivos?, ¿he tenido algún problema que me haya impedido alcanzar mis objetivos?

- *Sprint Review*: al final de cada *sprint* se organiza una reunión, de un máximo de cuatro horas, en la que se invita a los stakeholders y se presenta el trabajo desarrollado durante el *sprint*.

- *Sprint Retrospective*: además de la anterior, al final de cada *sprint* también se realiza esta reunión, en la que todos los miembros del equipo comparten sus impresiones sobre el *sprint* pasado y se plantean posibles mejoras del proceso.

Artefactos

A lo largo de todo el desarrollo del producto se elaboran diferentes elementos que representan el trabajo o el valor del producto. Estos artefactos están diseñados para aportar la mayor transparencia posible sobre todo el proceso:

- *Product Backlog*: es una lista ordenada que contiene todo lo que se conoce que es necesario para el proyecto. Es la única fuente de requisitos del producto. En ella se establecen todas las funcionalidades, cambios y mejoras que deben ser hechos en un futuro. Es un documento que nunca está completo, evoluciona y cambia a la vez que lo hace el producto. Los elementos más prioritarios constituyen los cambios más tempranos que se harán en el producto, y deben estar bien detallados para que sean transparentes a todo el equipo.

- *Sprint Backlog*: es el subconjunto de elementos del *Product Backlog* que se realizarán durante el *sprint*. Se elabora durante el *Sprint Planning*. El equipo de desarrollo es el encargado de escoger las tareas que se incluirán en esta lista. A lo largo del *sprint* se pueden añadir nuevos elementos si se considera que es necesario para cumplir con el objetivo establecido. El equipo de desarrollo debe ir actualizando el trabajo que va completando.

- *Sprint Burndown chart*: es una gráfica que muestra el trabajo realizado y el trabajo pendiente en relación con la duración del *sprint*.

4.1.3 Adaptación de Scrum al proyecto

Existen diferentes factores que hacen que no sea posible un fiel seguimiento de esta metodología: equipo de desarrollo unipersonal, dedicación no constante, etc. A continuación se detallan las adaptaciones que se han realizado sobre la metodología original para aplicarla a este proyecto:

- El rol de *Product Owner* es desempeñado por el director del proyecto Javier Parapar López.
- El rol de *Scrum Master* es desempeñado por el director del proyecto Daniel Valcarce Silva.
- El equipo de desarrollo esta formado únicamente por una persona, el alumno David Otero Freijeiro.
- No se realizan las reuniones diarias de Scrum, ya que el equipo de desarrollo es de una sola persona. Sin embargo, es el propio alumno el que autoevalúa las cuestiones que se tratarían en estas reuniones.
- Siguiendo los principios de Scrum, se ha intentado que el esfuerzo realizado durante un *sprint* fuese siempre el mismo. Sin embargo, esto no ha sido posible debido a obligaciones académicas del alumno. De todas maneras, sí se ha respetado la duración de los *sprints*.
- La duración de los *sprints* es de 3 semanas y conllevan un esfuerzo de 40 puntos de historia.

Historias de usuario y puntos de historia

Las historias de usuario son descripciones breves y concisas sobre una funcionalidad del sistema, descrita desde la perspectiva del usuario que empleará dicha característica. Generalmente siguen el mismo formato:

Como [rol] quiero [funcionalidad]

Los requisitos funcionales de este proyecto se han recogido en forma de historias de usuario, que ha sido la manera de representar los elementos del *Product Backlog*.

Para estimar el esfuerzo necesario para llevar a cabo una historia de usuario se utilizan los puntos de historia. Estos puntos representan una medida abstracta del esfuerzo necesario para completar una historia, que debe contemplar todo lo que puede afectar a la correcta

consecución de esta: cantidad de trabajo a realizar, complejidad del trabajo, riesgo o incertidumbre, etc. A la hora de asignar estos puntos a una historia se utiliza una sucesión similar a la sucesión de Fibonacci[1], esto se hace porque cuanto más avanza la sucesión más grande es el intervalo entre los números. De igual manera, cuanto más alto es el esfuerzo estimado para una historia, más grandes son el riesgo y la incertidumbre asociados a esta.

4.2 Gestión del proyecto

La gestión del proyecto tiene como objetivos los siguientes puntos:

- Cumplir con la planificación realizada y los costes estimados.
- Optimizar el uso de recursos.
- Tener constancia en cualquier momento del estado real en el que se encuentra el proyecto.

4.2.1 Estimación

Para estimar las historias de usuario se ha seguido la siguiente convención:

1 hora y 30 minutos de trabajo $\simeq$ *1 punto de historia*

Así se puede estimar el coste de realizar una historia de usuario en horas-persona.

Los *sprints* tienen una estimación de 40 puntos de historia y su duración, en circunstancias normales, es de 3 semanas. Sabiendo esto, el tiempo de trabajo ideales de 4 horas al día aproximadamente.

Debido a lo explicado en la Sección 4.1.3, por circunstancias académicas y personales, el esfuerzo realizado en algunos *sprints* no se ha adaptado a lo planificado, pero sí se ha mantenido constante la duración de estos.

4.2.2 Planificación

A continuación se describe brevemente cada uno de los *sprints*:

- **Sprint 0**: Configuración del entorno de desarrollo y búsqueda en Reddit.
- **Sprint 1**: Indexación, búsqueda y publicaciones de usuarios.
- **Sprint 2**: Web de búsqueda y ranking de usuarios.
- **Sprint 3**: Creación de experimentos.

[1] https://es.wikipedia.org/wiki/Sucesion_de_Fibonacci

- **Sprint 4**: Configuración de un experimento.
- **Sprint 5**: Detalle y lista de experimentos.
- **Sprint 6**: *Crawling* y *pooling* en un experimento.
- **Sprint 7**: Exportación de colecciones y gráficas.

4.2.3 Recursos

Como se ha comentado en la Sección 4.1.3, existen tres recursos humanos dedicados al desarrollo del sistema:

- El equipo: David Otero Freijeiro.
- Los directores: Javier Parapar López y Daniel Valcarce Silva.

La mayor parte de los recursos materiales son propiedad del alumno. Los recursos como GitLab, Jenkins, o las herramientas de gestión de proyectos fueron proporcionadas por los directores.

4.2.4 Costes

Para los recursos humanos expuestos en la sección anterior se contemplan los siguientes costes, extraídos del *Estudio salarial - Sector TIC en Galicia 2015-2016* [33]:

Rol	Coste (€/hora)
Equipo de desarrollo	21.28
Director	46.55

Cuadro 4.1: Estimación de costes para los recursos humanos del proyecto.

Además, deben tenerse en cuenta los siguientes aspectos:

- El proyecto cuenta con 8 sprints.
- Se estima que los directores dedican 3 horas por *sprint*.
- De acuerdo con la Sección 4.2.1, el esfuerzo dedicado a cada sprint por parte del equipo de desarrollo es de 60 horas/persona por *sprint*.

Rol	Tiempo (h/sprint)	Dedicación (nº *sprints*)	Coste (€/h)	Total (€)
Equipo de desarrollo	60	8	21.28	10 214.4
Director	6 (ambos directores)	8	46.55	2234.4
			Total	**12 448.8**

Cuadro 4.2: Desglose del coste de los recursos humanos.

En el Cuadro 4.2 se muestra el desglose del coste de los recursos humanos del proyecto. El total de este coste es de 12 448.8 €.

En cuanto a los costes materiales, estos se desglosan en el Cuadro 4.3. Para los equipos informáticos se ha imputado como coste únicamente la parte proporcional del uso respecto a su vida útil.

Recurso	Unidades	Coste (€/unidad)	Vida útil (meses)	Tiempo de uso (meses)	Total (€)
Licencias software	-	-	-	-	0
Estación de trabajo	1	2000	48	6	250
				Total	**250**

Cuadro 4.3: Desglose del coste de los recursos humanos.

El coste total del proyecto es de 12 698.8 €. Es conveniente indicar que esto es solamente el precio de coste, no se contempla ningún margen de beneficio.

4.2.5 Gestión de riesgos

Un aspecto fundamental que debe gestionarse a lo largo del desarrollo de un proyecto son los riesgos. Las metodologías ágiles minimizan estos de manera natural. Sin embargo, siguen existiendo factores difíciles de gestionar si no se los considera explícitamente. Esta sección trata su identificación, clasificación y, en función de la exposición del proyecto a ellos, su prevención, seguimiento o asimilación.

Identificación y clasificación

En primer lugar se identifican los riesgos, su probabilidad de aparición y la exposición del proyecto a ellos. Los riesgos R1, R2, R5 y R6 son ya minimizados por la metodología, por lo que su probabilidad de aparición es baja. El resultado de este análisis se detalla en el siguiente cuadro:

Código	Descripción	Probabilidad	Impacto	Exposición
R1	Requisitos imprecisos	Baja	Medio	Baja
R2	Planificación incorrecta	Baja	Alto	Alta
R3	Falta de recursos	Baja	Alto	Media
R4	Falta de tecnología	Baja	Alto	Media
R5	Mal diseño	Baja	Alto	Media
R6	Mala implementación	Baja	Alto	Media

Cuadro 4.4: Identificación y clasificación de riesgos.

Prevención (planes de contingencia)

Los riesgos de alta exposición deben prevenirse para evitar que estos hagan fracasar la ejecución del proyecto. Por ello son analizados en fases muy tempranas –antes de comenzar el diseño o la implementación– ya que su aparición tardía podría hacer el proyecto insostenible al requerir grandes cambios.

A continuación se detallan los planes de contingencia para los riesgos de alta exposición:

- **R2 - Planificación incorrecta**: dada la escasa experiencia de planificación, es posible que las tareas se desvíen de lo planificado. Por ello, se realizará un seguimiento exhaustivo de la planificación para evitar retrasos.

Seguimiento

Para los riesgos de exposición media se realizará un seguimiento durante el proyecto para controlar que no se conviertan en riesgos de alta exposición:

- **Falta de recursos**: se conocen de antemano los recursos con los que contamos, por lo que la planificación se deberá hacer acorde a ellos.
- **Falta de tecnología**: se deberá realizar un estudio exhaustivo en la fase de elección tecnológica para mantener controlado este riesgo.
- **Mal diseño**: este riesgo está controlado al empleado una metodología iterativa, ya que queda acotado por cada iteración.
- **Mala implementación**: se realizará un seguimiento del correcto desarrollo del diseño proyectado.

Asimilación

Los riesgos de exposición baja son aceptados ya que actuar sobre ellos es más costoso que realizar la gestión de problemas en caso de que lleguen a darse.

En este proyecto se ha identificado un riesgo de exposición baja que se considera poco probable (**R1 - Requisitos imprecisos**), ya que al no tratar con un cliente real los requisitos pueden especificarse en etapas tempranas y permanecer estables a lo largo del desarrollo del proyecto.

Capítulo 5

Desarrollo

En este capítulo se detalla el proceso de desarrollo del proyecto. En primer lugar, se analizarán los requisitos del proyecto para definir la arquitectura global del sistema. Luego se describirá el modelo de datos de la aplicación. Por último, se detallará el trabajo realizado en cada *sprint*.

5.1 Análisis de requisitos

A diferencia de un requisito, un requerimiento es cualquier necesidad especificada por el cliente. Un requerimiento pasa a ser requisito si es factible su implementación y puede satisfacerse. De esta manera, aquellos requerimientos que no sea factible implementar no se consideran requisitos.

A continuación se exponen los requisitos de este sistema, deducidos a partir de diversas reuniones con los directores. Debido a que no entran dentro del alcance del sistema y que, como se ha explicado anteriormente, no se consideran factibles para ser implementados, se obvia el tratamiento de los requerimientos.

5.1.1 Requisitos funcionales

Los requisitos funcionales representan las funcionalidades que debe ofrecer el sistema. A continuación se exponen los requisitos funcionales del proyecto, expresados como historias de usuario, que forman el *Product Backlog*. En el Cuadro 5.1 se puede ver una lista ordenada por prioridad de las historias de usuario estimadas al inicio del proyecto.

ID	Historia de usuario	Puntos
8	Como administrador quiero recuperar textos de Reddit a partir de una consulta	20

9	Como administrador quiero obtener la colección de textos publicados por un usuario	20
24	Como administrador quiero obtener la historia de los usuarios que hayan publicado una consulta	10
10	Como administrador quiero persistir los datos obtenidos sobre los usuarios de Reddit	10
37	Como administrador quiero indexar los usuarios recuperadores de Reddit	5
38	Como administrador quiero obtener un ranking de los usuarios recuperados de Reddit	13
39	Como asesor quiero ver toda la historia publicada por un usuario	20
49	Como administrador quiero ver los rankings obtenidos a partir de diversas consultas	10
48	Como administrador quiero ver los usuarios recuperados a partir de una consulta	10
58	Como administrador quiero crear un experimento	20
57	Como administrador quiero obtener un ranking de la historia de un usuario	20
59	Como administrador quiero elegir el modelo de retrieval de un experimento	8
60	Como administrador quiero elegir las consultas de un experimento	13
61	Como administrador quiero escoger la estrategia de *pooling* de un experimento	8
62	Como asesor quiero ver una lista de los experimentos disponibles	13
73	Como asesor quiero ver el detalle de un experimento	13
74	Como administrador quiero ejecutar un experimento	20
79	Como asesor quiero hacer *pooling* sobre una colección de usuarios	20
3	Como asesor quiero emitir un juicio sobre un usuario de Reddit	20

7	Como administrador quiero comparar el rendimiento de las combinaciones de modelos de recuperación y de *pooling*	20
6	Como administrador quiero exportar las colecciones junto con los juicios emitidos	20

Cuadro 5.1: Historias de usuario al inicio del proyecto.

En la Figura 5.1 se puede ver parte de esta misma estimación pero utilizando la herramienta Taiga.

5.1.2 Requisitos no funcionales

Por otro lado, los requisitos no funcionales especifican diversos criterios que el sistema debe cumplir, en lugar de funciones específicas como indican los requisitos funcionales. Estos condicionan muchas decisiones de diseño de la arquitectura y sus componentes así como en las elecciones tecnológicas detalladas en la Sección 3. En el Cuadro 5.2 se recogen los requisitos no funcionales deducidos tras el proceso de ingeniería de requisitos.

5.2 Arquitectura

Con base en los requisitos enunciados en la sección anterior se propone una arquitectura basada en capas. La arquitectura en capas organiza los componentes del software en capas, donde cada una de estas ofrece una funcionalidad concreta a la capa superior. Entre las ventajas de esta arquitectura cabe destacar:

- Facilita el mantenimiento (RNF-04). Al estar organizado en capas se consigue una localización de los cambios, es decir, los cambios en una capa no afectan a las demás, siempre que se respeten las interfaces.
- Las capas pueden ser desarrolladas en paralelo.
- Favorece el soporte multi-plataforma.
- Facilita la escalabilidad y la tolerancia a fallos (RNF-01 y RNF-05), ya que se pueden destinar más recursos solamente de las capas que lo necesiten.

Votes	User Stories	Status	Points
▲ 0	#8 Como administrador quiero recuperar textos de Reddit a partir de una consulta	New	20
▲ 0	#9 Como administrador quiero obtener la colección de textos publicados por un usuario	New	20
▲ 0	#24 Como administrador quiero obtener la historia de los usuaros que hayan escrito una consulta	New	10
▲ 0	#10 Como administrador quiero persistir los datos obtenidos sobre los usuarios de Reddit	New	10
▲ 0	#37 Como administrador quiero indexar los usuarios recuperados de Reddit	New	5
▲ 0	#38 Como administrador quiero obtener un ranking de los usuarios recuperados de Reddit	New	13
▲ 0	#39 Como asesor quiero ver toda la historia publicada por un usuario	New	20
▲ 0	#49 Como administrador quiero ver los rankings obtenidos a partir de diversas consultas	New	10
▲ 0	#48 Como administrador quiero ver los usuarios recuperados a partir de una consulta	New	10

Figura 5.1: *Product Backlog* en Taiga.

Código	Requisito	Descripción
RNF-01	Fiabilidad	El sistema debe ofrecer la funcionalidad como se necesita. Debe minimizarse tanto la probabilidad de fallo como el número de fallos.
RNF-02	Usabilidad	La aplicación deberá ser lo más fácil, cómoda e intuitiva de usar posible.
RNF-03	Eficiencia	La aplicación debe hacer una buena gestión de los recursos disponibles.
RNF-04	Mantenibilidad	No debe requerir mucho esfuerzo que la aplicación crezca en funcionalidades.
RNF-05	Disponibilidad	El sistema debe ofrecer la funcionalidad cuando se necesita.
RNF-06	Rendimiento	Debe minimizarse la complejidad del sistema y favorecer el rendimiento.

Cuadro 5.2: Requisitos no funcionales del sistema.

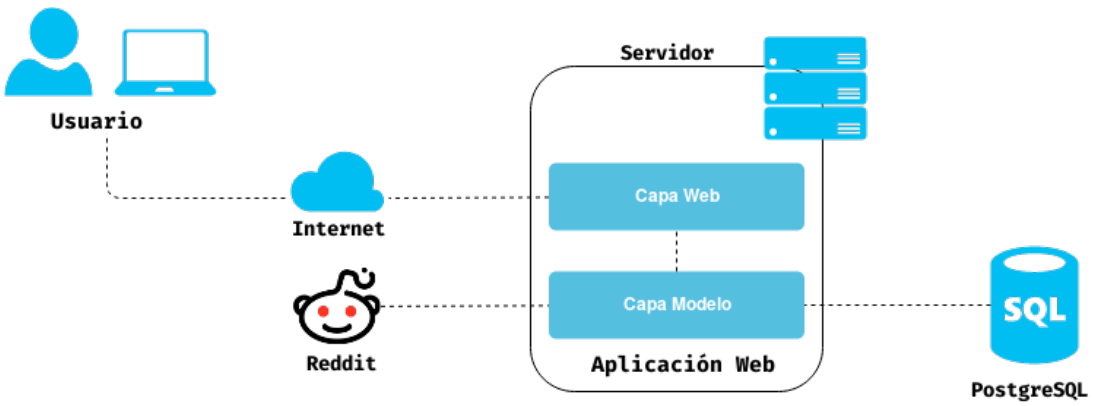

Figura 5.2: Arquitectura general de la plataforma.

5.2.1 Propuesta

En la Figura 5.2 se ilustra la arquitectura propuesta para este sistema.

En esta imagen se ve la clara distinción entre las dos capas del sistema: la capa web y la capa modelo. La primera es la encargada de ofrecer al usuario final la funcionalidad ofrecida por la capa modelo. La segunda implementa las funcionalidades propias de la aplicación y la gestión de los datos de esta. Esta capa, a su vez, se divide en dos capas. La capa de acceso a datos, que es la encargada de gestionar la persistencia de los datos en la base de datos. La segunda, la capa de lógica de negocio, es la que implementa la lógica de los casos de uso, utilizando la capa anterior para gestionar los datos necesarios.

5.3 Modelo de datos

El diagrama entidad-relación (ER) que representa el modelo de datos usa la notación de diagrama de clases UML en lugar de la propia notación ER. El motivo de esta decisión ha sido hacerlo más comprensible, ya que la notación UML es más estándar que la ER y se utiliza principalmente para software orientado a objetos.

En la Figura 5.3 se muestra el modelo de datos de las entidades persistentes de la plataforma.

5.4 Propuesta

En esta sección se comentan algunos detalles específicos de este sistema para poder entender mejor el desarrollo de este.

Esta plataforma está destinada para el uso por dos tipos de roles de usuarios: por un lado, el administrador, que es el encargado de crear y configurar los experimentos y, por otro lado,

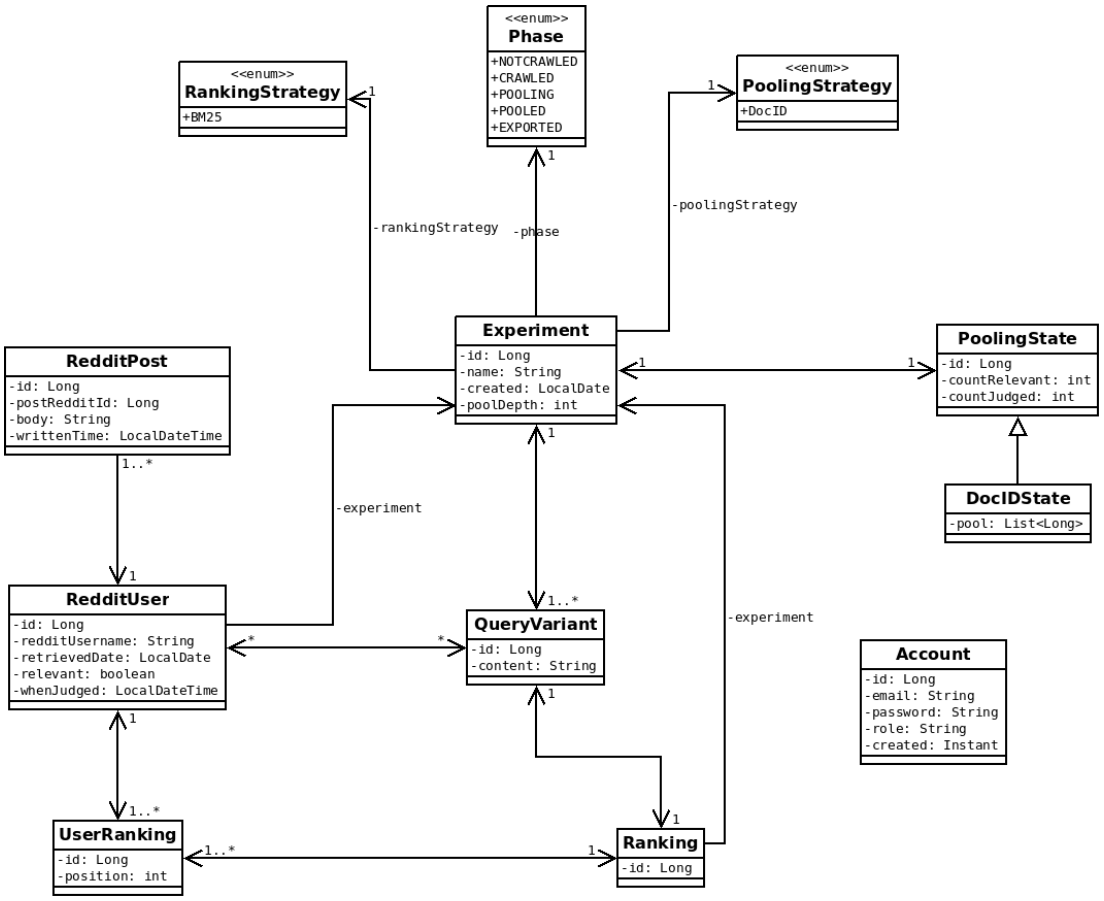

Figura 5.3: Modelo de datos de la aplicación.

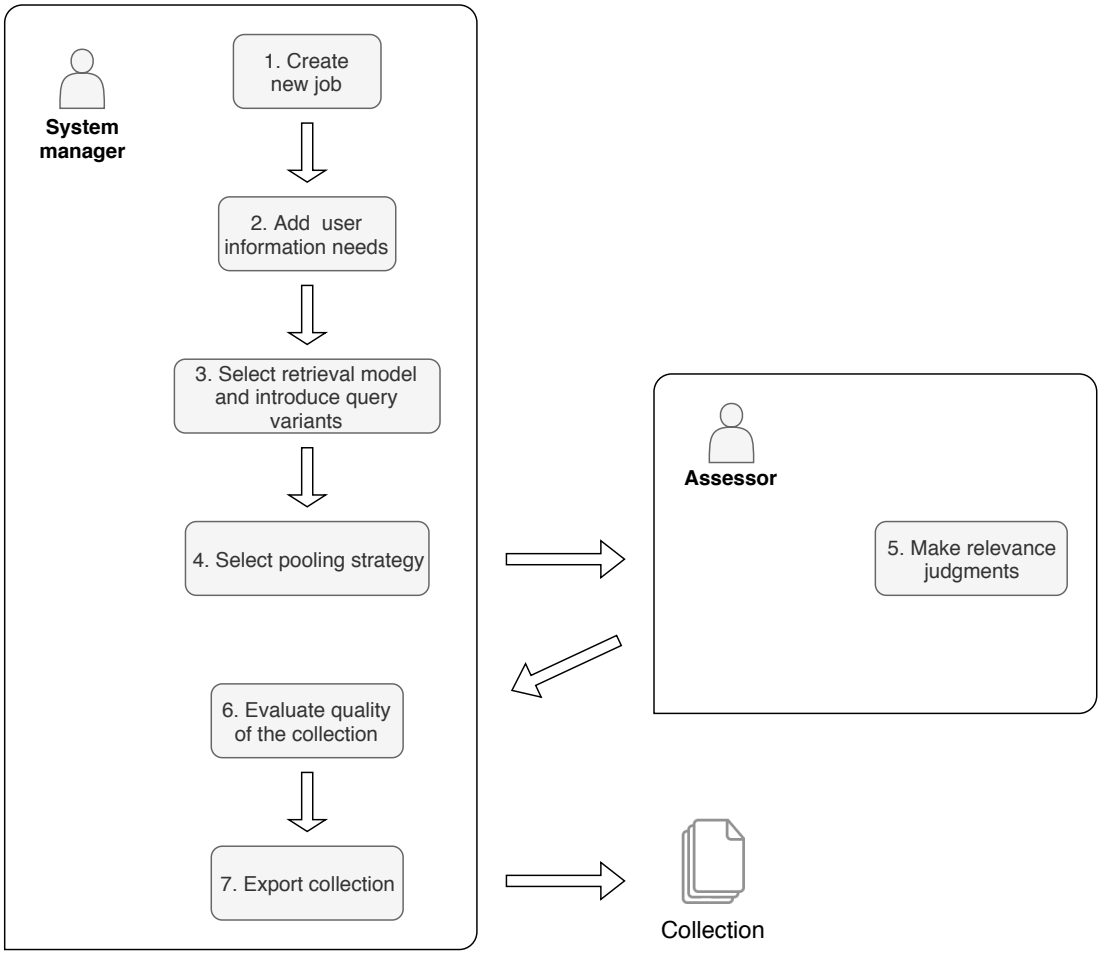

Figura 5.4: Flujo de trabajo con la plataforma.

el asesor, que es el encargado de crear los juicios de relevancia de la colección.

El flujo general de uso de la aplicación se puede ver en la Figura 5.4. Esta imagen se ilustran los dos roles del sistema junto con la labor que tiene que hacer cada uno.

5.5 Desarrollo

En esta sección se describe el proceso de desarrollo, detallando las historias de usuario completadas en cada *sprint*.

Una vez realizada la planificación inicial del proyecto, la estimación total del *Product Backlog* ha sido de 313 puntos de historia y, como el proyecto durará 8 *sprints*, lo ideal será completar aproximadamente 40 puntos en cada sprint.

En la Figura 5.5 se ilustra un ejemplo de esta estimación. En esta imagen se pueden ver varios datos:

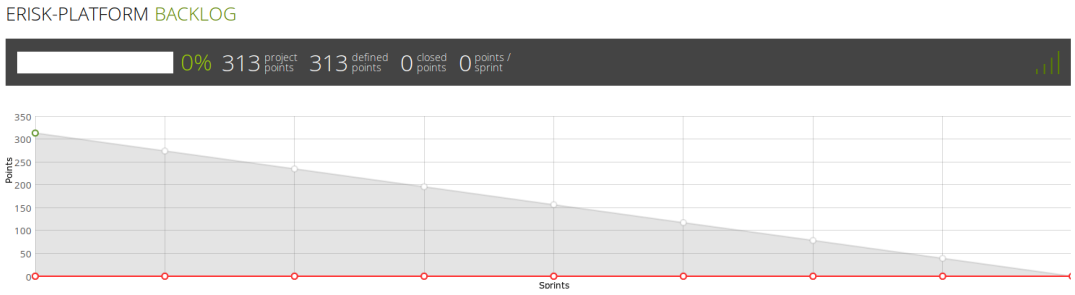

Figura 5.5: Estimación de puntos en Taiga.

- *Project points*: son los puntos totales asignados al proyecto.
- *Defined points*: es la cantidad de puntos que han sido asignados a las historias de usuario.
- *Closed points*: es la cantidad de puntos historia que se han completado.
- *Points/sprint*: es lo que en terminología de Scrum se conoce como *velocity*. Es la media, según datos históricos, de puntos que se han completado en cada *sprint*. Este dato es el que debería mantenerse más o menos en 40 puntos historia por *sprint*, como se ha indicado anteriormente.

5.5.1 Sprint 0: Configuración del entorno de desarrollo y búsqueda en Reddit

En este primer *sprint* se ha comenzado configurando el entorno de desarrollo sobre que el que construirá el proyecto a lo largo de los demás *sprints*. En cuanto a este paso se exponen los siguientes detalles:

- Se ha construido un proyecto en Eclipse utilizando un arquetipo de Maven[1].
- Configuración del proceso de integración continua, mediante las herramientas Git y Jenkins. De esta manera, cada vez que el repositorio remoto recibe alguna actualización, Jenkins lanza una nueva integración. Para poder facilitar este proceso se han configurado diversos perfiles en el POM de Maven, los cuales permiten establecer diferentes variables bajo diferentes condiciones.
- Configuración de la base de datos, tanto la de pruebas como la de producción, utilizando PostgreSQL y H2.

[1]https://github.com/kolorobot/spring-mvc-quickstart-archetype

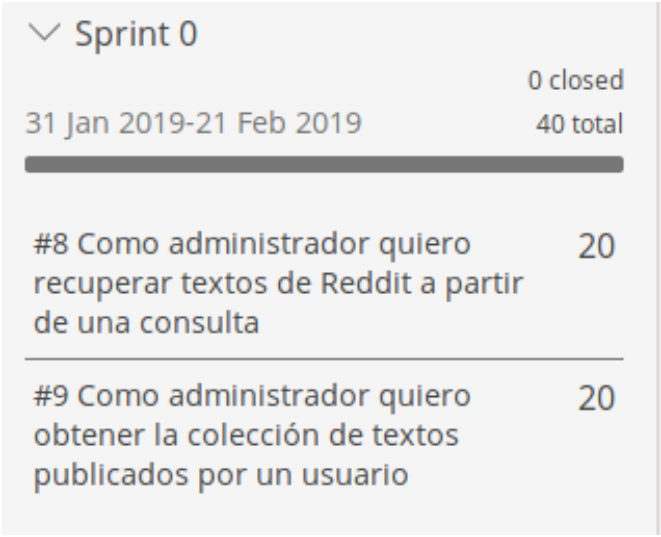

Figura 5.6: Estimación de las historias para el Sprint 0.

- Configuración del proceso de inspección continua con la herramienta SonarQube. Como se ha explicado anteriormente, cuando se producen cambios en el repositorio remoto Jenkins inicia una nueva integración, una vez que esta integración termina exitosamente se ejecuta un análisis de la calidad del código, que es lanzado automáticamente. De esta manera solo debemos preocuparnos de subir los cambios frecuentemente al repositorio, y automáticamente se lanzan la integración y el análisis de calidad del proyecto.
- Integración de las demás herramientas en el flujo de desarrollo: la gestión del proyecto con Taiga y el control de versiones con Git, lo que incluye la creación de los respectivos proyectos dentro de estas herramientas.

Para este *sprint* estaban planeadas las historias 8 y 9 del *Product Backlog*, como se ilustra en la Figura 5.6

A continuación se ha empezado con el desarrollo del proyecto. En este primer *sprint* se ha implementado el componente que consume la API de Reddit. Mediante este componente se pueden recuperar publicaciones de Reddit a partir de una consulta (8), así como toda la historia de un usuario (9). Estas dos historias se completaron cuando el *sprint* solamente había llegado a la mitad de su duración. Por lo tanto, para mantener constante la duración del mismo se decidió establecer que el esfuerzo de estas dos historias pasaría a ser de 10 puntos y se incluyeron dos historias a mayores para continuar con el desarrollo. Estas dos historias son la 24 y la 10. Estas historias están estimadas en 10 puntos ambas, de esta manera también se mantiene el esfuerzo estimado de 40 puntos para este *sprint*.

A partir de las dos funcionalidades ya implementadas también se puede obtener toda la historia de los usuarios que hayan publicado textos que coincidan con una consulta (24), aun-

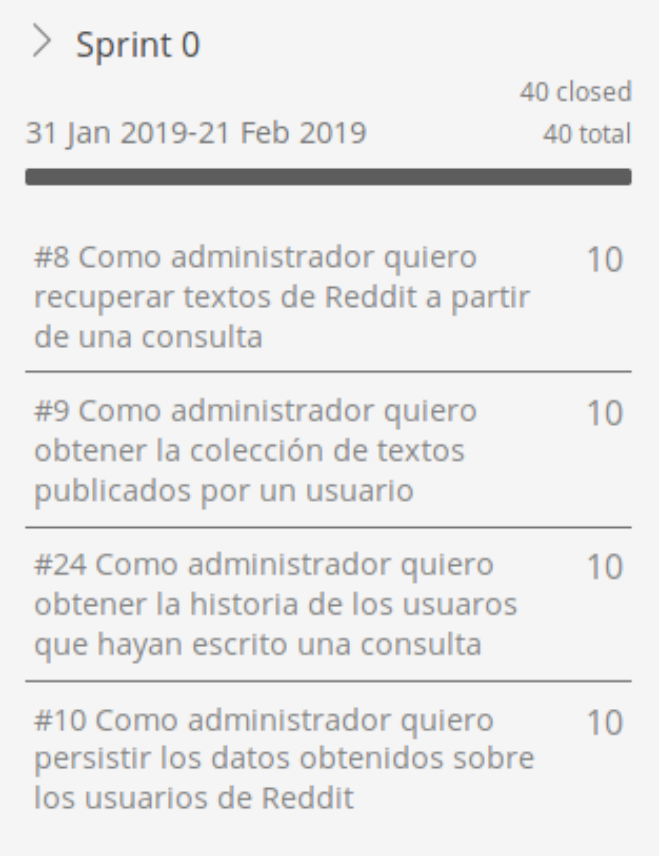

Figura 5.7: Historias completadas en el Sprint 0.

que algunas publicaciones no contengan la consulta especificada, como es lógico. Además, también se ha implementado la parte de la persistencia de estos datos obtenidos de Reddit (10).

Balance del *sprint*

Como se ha indicado anteriormente, las historias inicialmente estimadas para este *sprint* se completaron antes de tiempo. Esto se debe al desconocimiento inicial de la API de Reddit, que posteriormente resultó fácil de consumir, por lo que las tareas duraron menos de lo previsto. De todas formas, al final del *sprint* se logró cumplir con el tiempo y el esfuerzo previstos para este.

En la Figura 5.7 se pueden ver las historias que finalmente se completaron en este *sprint*, junto con el esfuerzo final de cada una.

En la Figura 5.8 se puede ver el progreso al final de este *sprint*. Como se ilustra en la imagen, el proyecto discurre conforme a lo planificado. Además, como los puntos asignados a algunas historias han cambiado, se puede ver también que el número total de puntos asignados

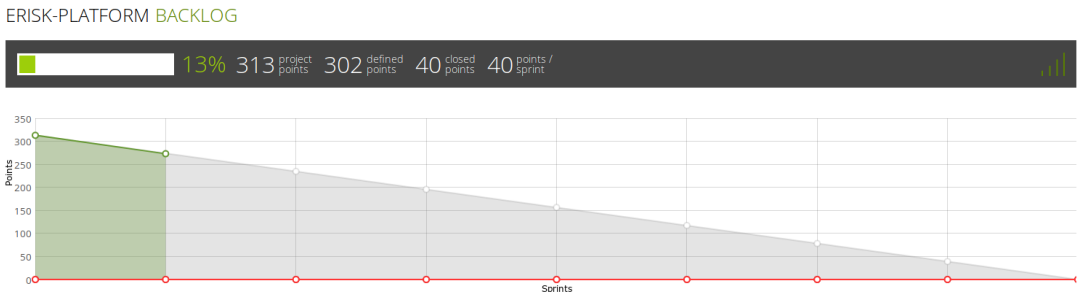

Figura 5.8: Progreso del Sprint 0.

ha cambiado también.

5.5.2 Sprint 1: Indexación, búsqueda y publicaciones de usuarios.

En la Figura 5.9 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

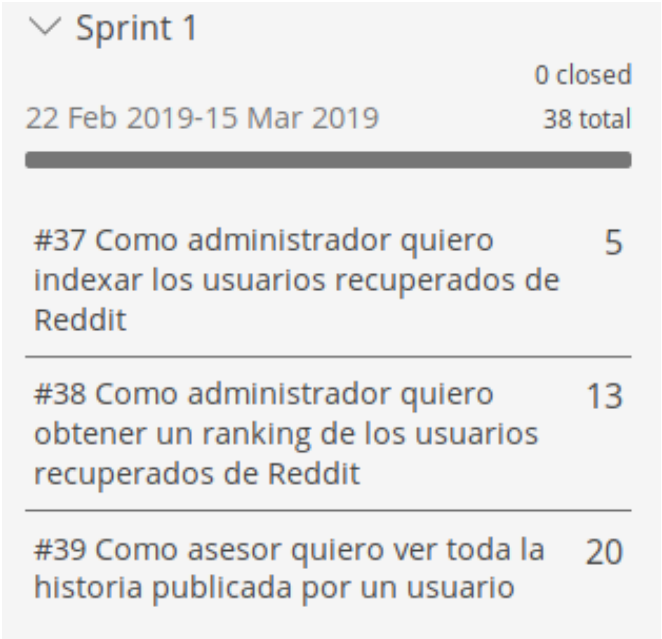

Figura 5.9: Estimación de las historias para el Sprint 1.

En este *sprint* se ha implementado la indexación de los usuarios recuperados de Reddit (37) utilizando la herramienta Lucene, explicada en la Sección 3. La funcionalidad implementada en esta historia se utiliza posteriormente en la siguiente historia desarrollada, la obtención de un ranking de usuarios según relevancia dada una consulta (38).

En un aspecto técnico, a la hora de implementar la historia (37) se ha decidido crear los

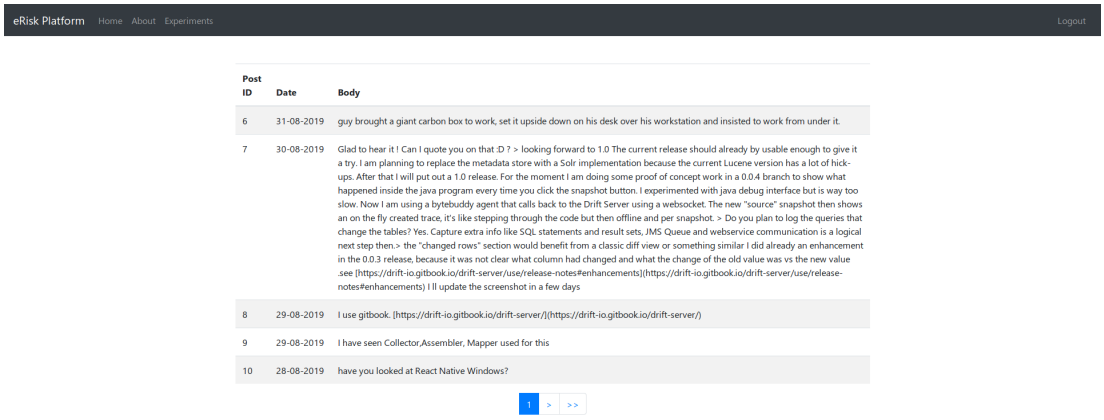

Figura 5.10: Vista de la historia de un usuario.

índices en memoria, esto se debe a que estos índices no son potencialmente grandes, por lo tanto se espera que quepan en memoria. Además, también se hacen pensando que en un futuro, cuando se implemente la ejecución de los experimentos, una vez esta finalice, estos índices ya no se necesitan más, por eso se crean en memoria en tiempo de ejecución. Cambiar la implementación para que se utilicen índices persistidos en disco es trivial, si fuese necesario.

A continuación se implementó la historia (39) para poder ver toda la historia de textos publicados por un usuario en Reddit. En la Figura 5.10 se puede ver esta funcionalidad implementada para un usuario con textos de ejemplo. En esta vista se ha decidido no poner el nombre del usuario que se está visualizando por cuestiones técnicas de la introducción de sesgos a la hora de evaluar la relevancia de usuario, tema que se explica en la Sección 2.

Balance del *sprint*

En este *sprint* se han podido completar todas las historias de usuario que se habían estimado (37, 38, 39). En la Figura 5.11 se pueden ver estas historias completadas, junto con el esfuerzo real de cada una.

El progreso de este *sprint* se puede ver en la Figura 5.12, donde se ilustra que, continuando en la misma tónica que el anterior *sprint*, se ha completado el esfuerzo previsto para cada sprint, que es de 40 puntos historia.

5.5.3 Sprint 2: Web de búsqueda y ranking de usuarios.

En la Figura 5.13 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

En este *sprint* se ha desarrollado la parte de la vista de dos tareas parcialmente desarrolladas en el *sprint* anterior. Estas historias son la 48 y la 49.

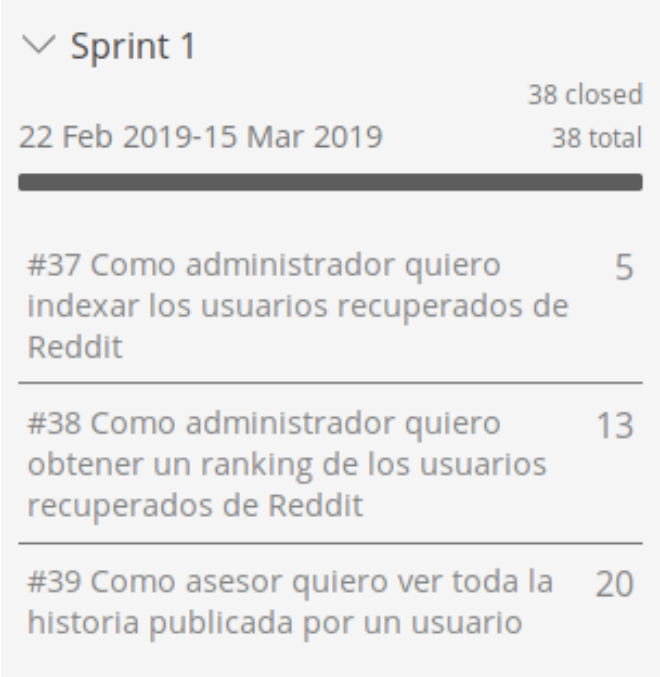

Figura 5.11: Historias completadas en el Sprint 1.

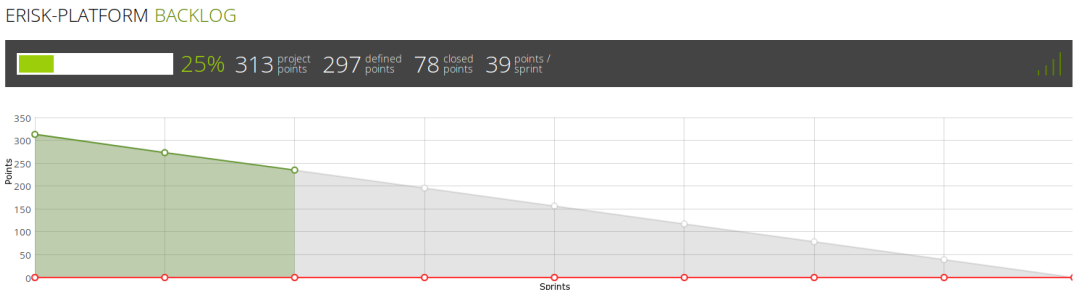

Figura 5.12: Progreso del Sprint 1.

En las figuras 5.14 y 5.15 se ilustra el resultado de la implementación de la historia 48. En la primera imagen se ilustra el formulario donde se puede introducir una consulta para buscar usuarios y el número máximo de usuarios a recuperar. Este segundo parámetro se introduce para poder poner un límite a la búsqueda, sino esta recupera todos los usuarios que encuentra, que potencialmente es un número muy grande.

A continuación se implementó la historia 49. El resultado se puede observar en las figuras 5.16 y 5.17. En la primera se ilustra un simple formulario donde es posible introducir una o más consultas para obtener diferentes rankings sobre los mismos usuarios. En la siguiente figura se observa el resultado de este proceso, donde se indica, para cada ranking, la consulta que lo ha producido y la lista de usuarios ordenada.

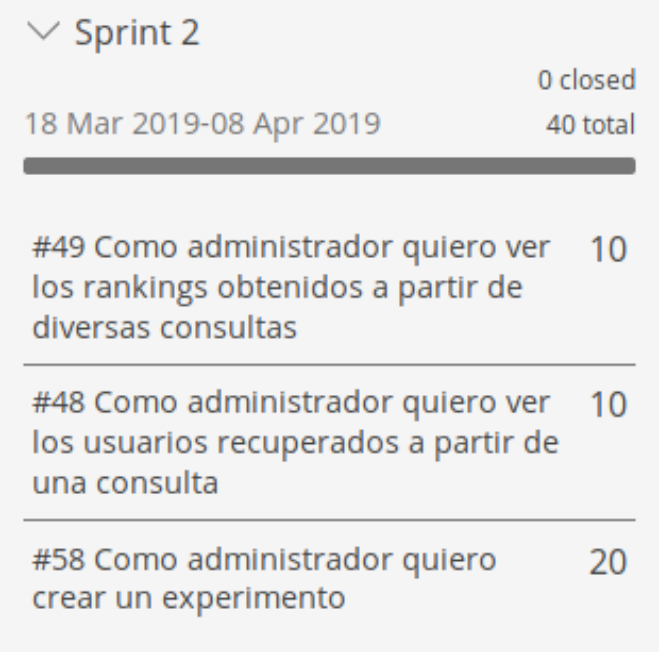

Figura 5.13: Estimación de las historias para el Sprint 2.

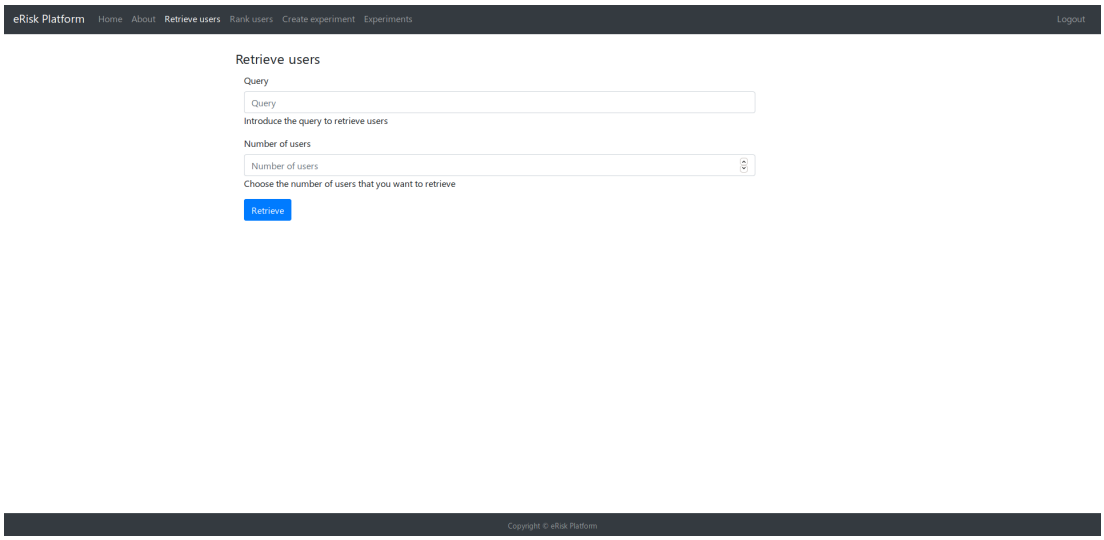

Figura 5.14: Formulario de recuperación de usuarios.

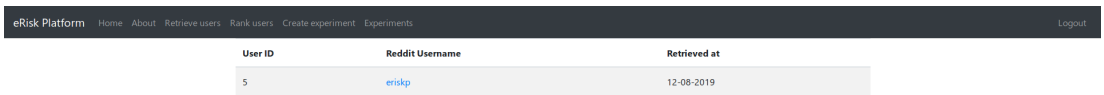

Figura 5.15: Resultados de la recuperación de usuarios.

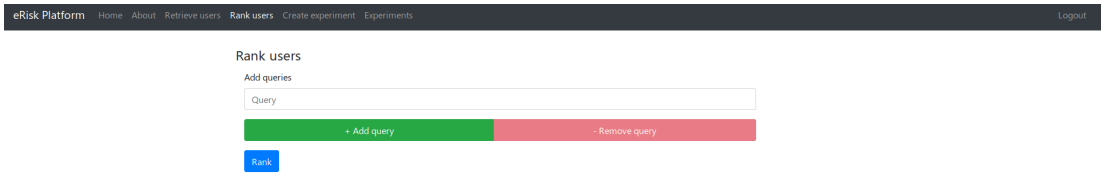

Figura 5.16: Formulario de rankings de usuarios.

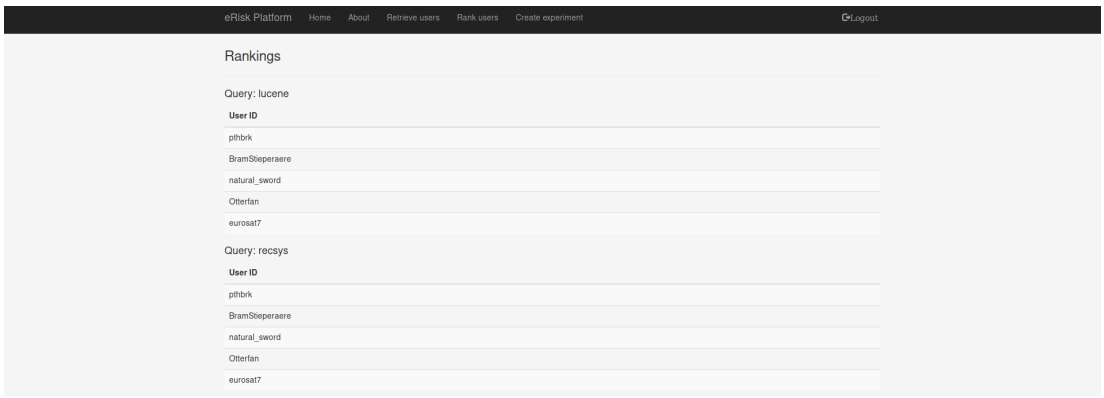

Figura 5.17: Resultados del ranking de usuarios.

Una vez implementadas estas dos historias se había llegado al fin del *sprint*, por lo que estas fueron mal estimadas en un principio. Se cambió el esfuerzo de ambas a 20 puntos, ya que las dos requirieron más o menos el mismo esfuerzo, de esta manera se mantiene el esfuerzo y el tiempo previstos para cada *sprint*.

Balance del *sprint*

Como se ha comentado antes, en este *sprint* se completaron menos historias de las estimadas ya que dos de ellas requirieron más esfuerzo del previsto inicialmente.

En la Figura 5.18 se pueden ver estas historias completadas, junto con el esfuerzo real de cada una.

Sin embargo, el progreso del proyecto, como se ilustra en la Figura 5.19, sigue conforme a lo estimado, ya que el esfuerzo realizado es el mismo. Se han completado aproximadamente los puntos previstos hasta este punto, que son 120.

5.5.4 Sprint 3: Creación de experimentos

En la Figura 5.20 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

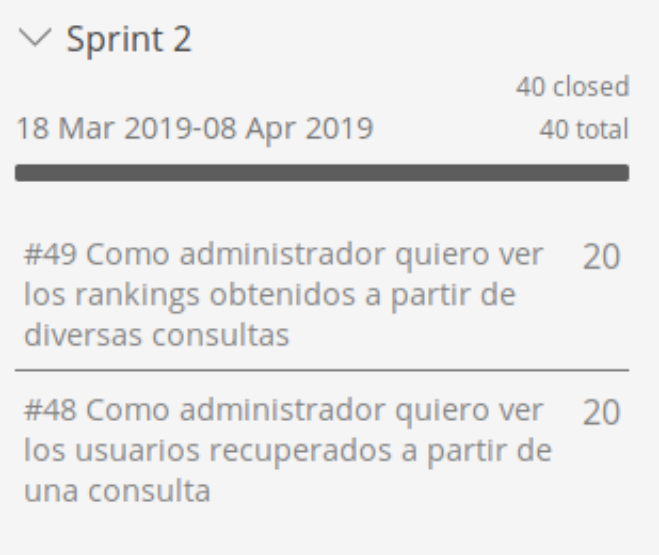

Figura 5.18: Historias completadas en el Sprint 2.

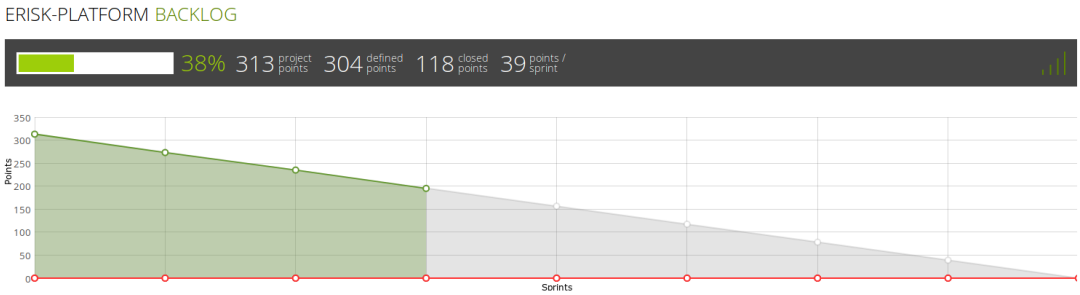

Figura 5.19: Progreso del Sprint 2.

En este *sprint* se completaron las historias 57 y 58. En cuanto a la primera, se introdujo la posibilidad de obtener un ranking de la historia de un usuario dada una consulta. Esto se hizo aprovechando la vista ya expuesta anteriormente donde se ve la historia de usuario. En la Figura 5.21 se ilustra un ejemplo de esta funcionalidad, donde se puede introducir una consulta para obtener un ranking de la historia del usuario en cuestión. El resultado del ranking es la misma vista pero con el orden de los textos cambiado.

A continuación se ha implementado la historia 58. En la Figura 5.22 se ilustra el formulario de creación de un experimento. Aunque en esta imagen se pueden ver que existen diferentes parámetros para configurar dentro de un experimento, inicialmente solo se contemplaba el nombre del mismo. Este nombre representa la necesidad de información asociada al experimento.

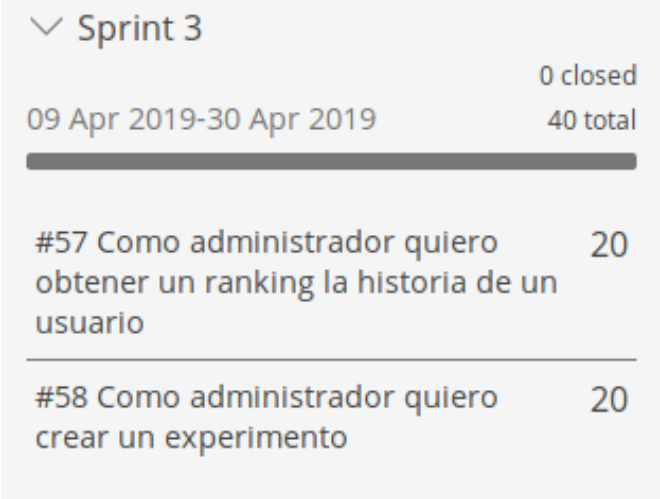

Figura 5.20: Estimación de las historias para el Sprint 3.

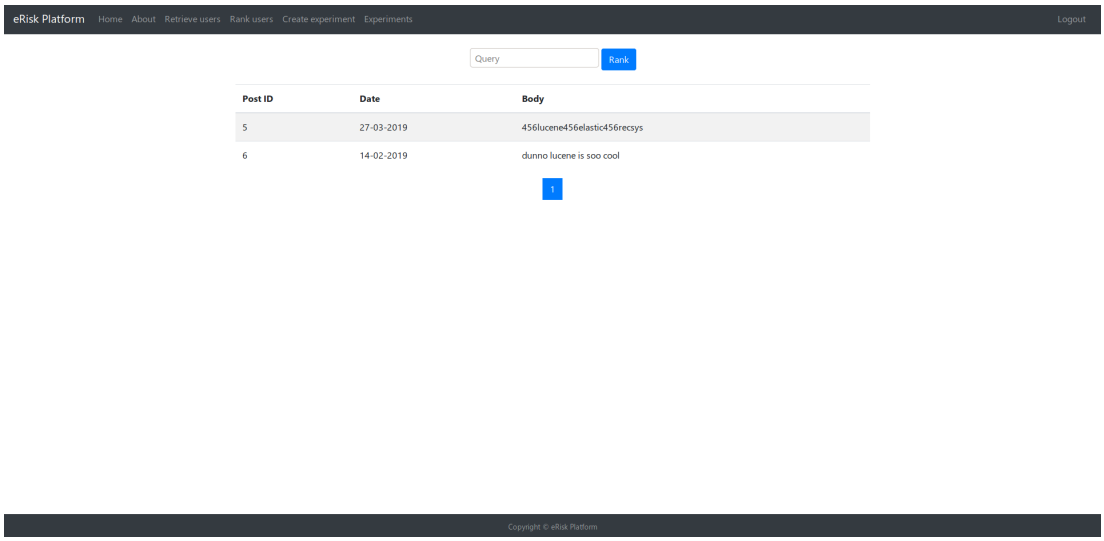

Figura 5.21: Ranking de la historia de un usuario.

Balance del *sprint*

En este *sprint* se ha completado la totalidad de las historias y los puntos estimados para este. Esto quiere que decir que en este punto el estado del proyecto va conforme lo planificado en tiempo.

En la Figura 5.23 se pueden ver estas historias completadas, junto con el esfuerzo real de cada una.

En la Figura 5.24 se puede ver el progreso del proyecto hasta este punto. Se han completado los puntos totales previstos hasta este punto, que son 160. Además, se ha conseguido, hasta ahora, una velocidad media de 40 puntos completados por *sprint*, lo que concuerda con lo

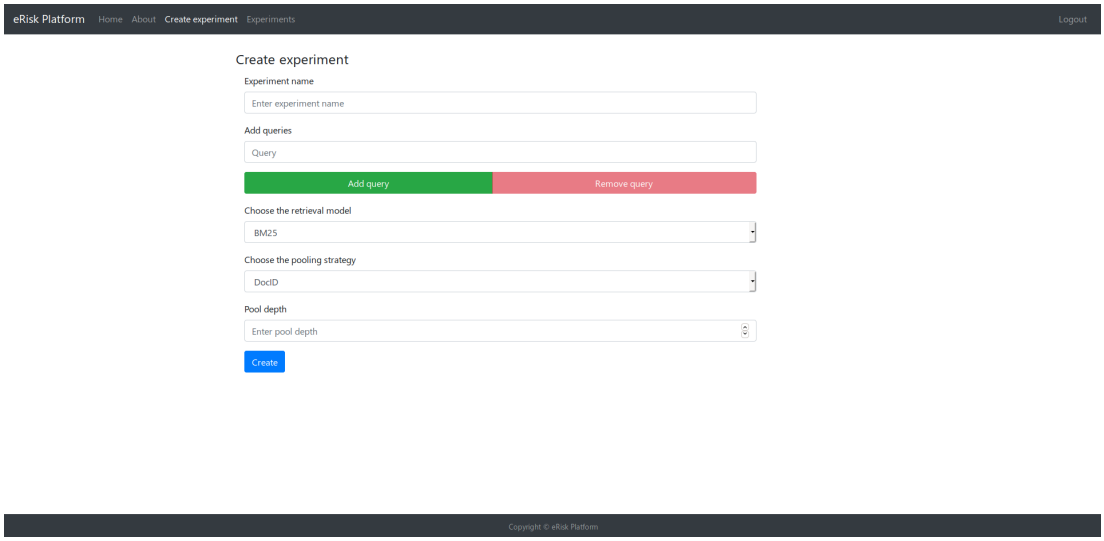

Figura 5.22: Formulario de creación de un experimento.

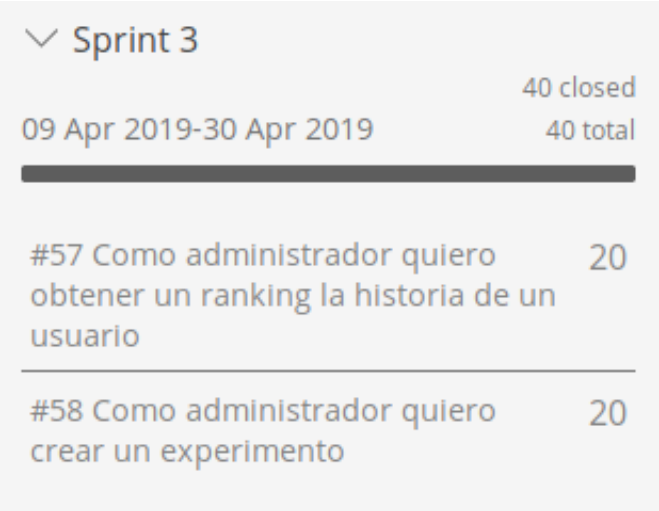

Figura 5.23: Historias completadas en el Sprint 3.

estimado al inicio.

5.5.5 Sprint 4: Configuración de un experimento.

En la Figura 5.25 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

En este *sprint* se completaron las historias 59, 60 y 61, todas relacionadas con la configuración de un experimento a la hora de crearlo. Como se ilustra en la Figura 5.26, a la hora de crear un experimento se pueden introducir las consultas que se utilizarán, por un lado para

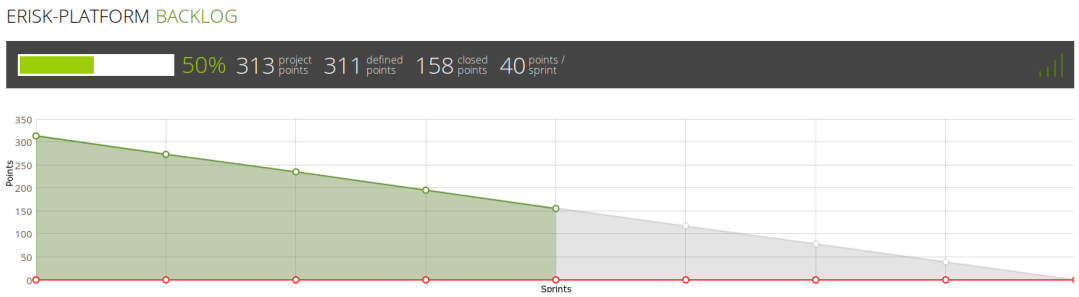

Figura 5.24: Progreso del Sprint 3.

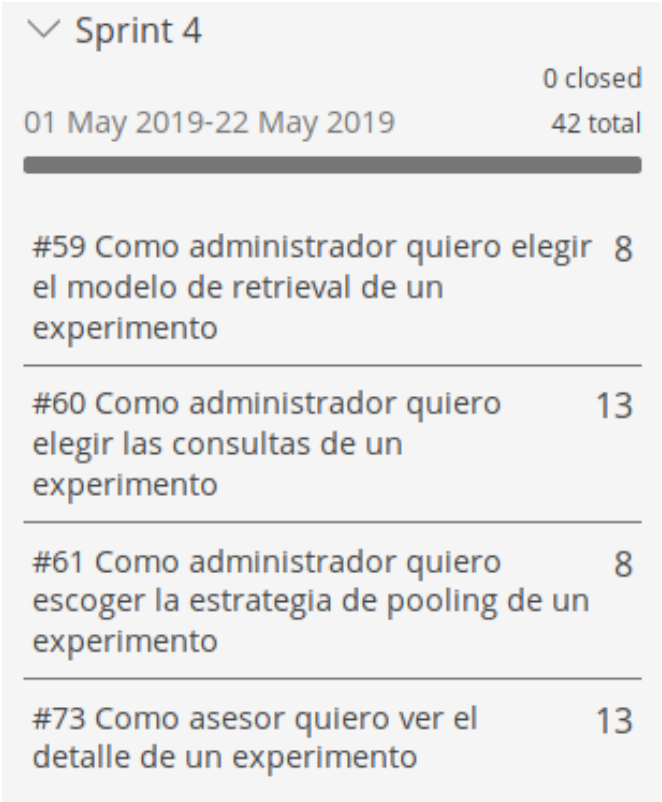

Figura 5.25: Estimación de las historias para el Sprint 4.

obtener los usuarios de Reddit, y por otro lado para construir los rankings que se utilizarán posteriormente en el proceso de *pooling*. Estas consultas representan las *query variants* que se introdujeron en el Capítulo 2. En este caso estas variantes son introducidas directamente por el administrador que crea el experimento. Además, también se puede escoger una estrategia de *retrieval* que se utilizará para construir dichos rankings junto con las consultas. Como se ilustra en esta misma imagen, también se pueden escoger la estrategia de *pooling*, la cual establece el orden en el cual se presentan los documentos a un asesor.

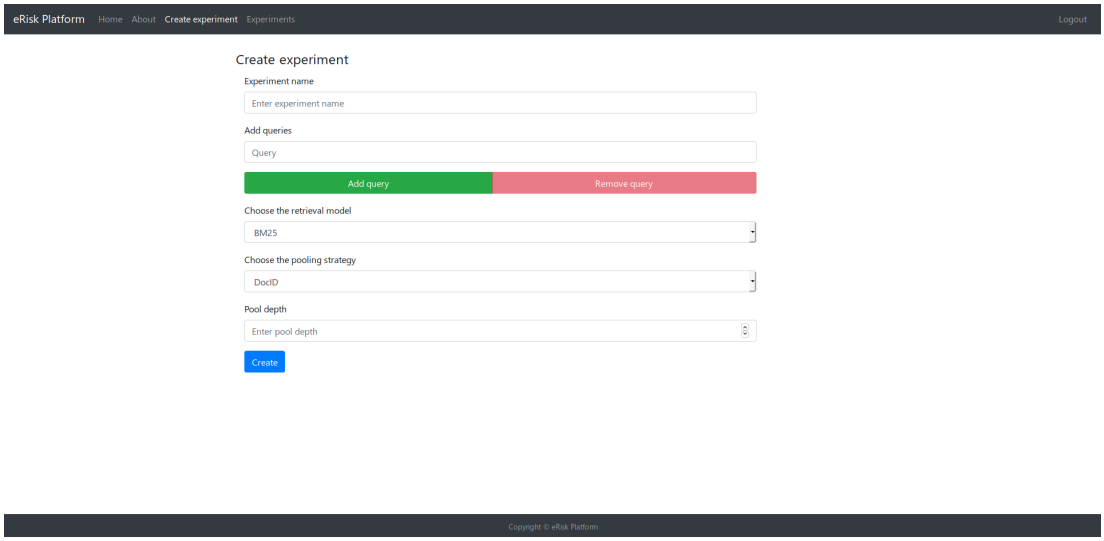

Figura 5.26: Formulario de configuración de un experimento.

Balance del *sprint*

En este *sprint* no se han logrado completar todas las historias estimadas, aunque las implementadas no hayan sido 40 puntos de esfuerzo. Como se explicó en una sección anterior, las obligaciones académicas del alumno han impedido que en este *sprint* se haya completado todo el trabajo estimado. Como consecuencia de esto, el esfuerzo completado al finalizar este *sprint* es menor de lo previsto, por lo que existe una pequeño desviación conforme a lo estimado. La única historia que no ha podido completarse en este *sprint* se dejará para *sprints* futuros.

En la Figura 5.27 se pueden ver estas historias completadas, junto con el esfuerzo real de cada una.

En la Figura 5.28 se ilustran las consecuencias de esta pequeña desviación. En cuanto a los puntos completados, la cifra es ligeramente menor a lo que debería ser en este punto. Además, se puede ver que la velocidad media ha bajado también, estando ahora en 37 puntos por *sprint*.

5.5.6 Sprint 5: Detalle y lista de experimentos.

En la Figura 5.29 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

La primera historia completada en este *sprint* fue la 62. Se implementó una vista para el rol de asesor, en la que este pudiera ver todos los experimentos que hay en el sistema, así como cierta información sobre ellos. En la Figura 5.30 se ilustra esta página. Aquí se puede ver una lista con los experimentos que hay en el sistema. Además, se puede ver alguna información sobre cada uno: el nombre, la fecha de creación y sus estado. Este último puede tener varios

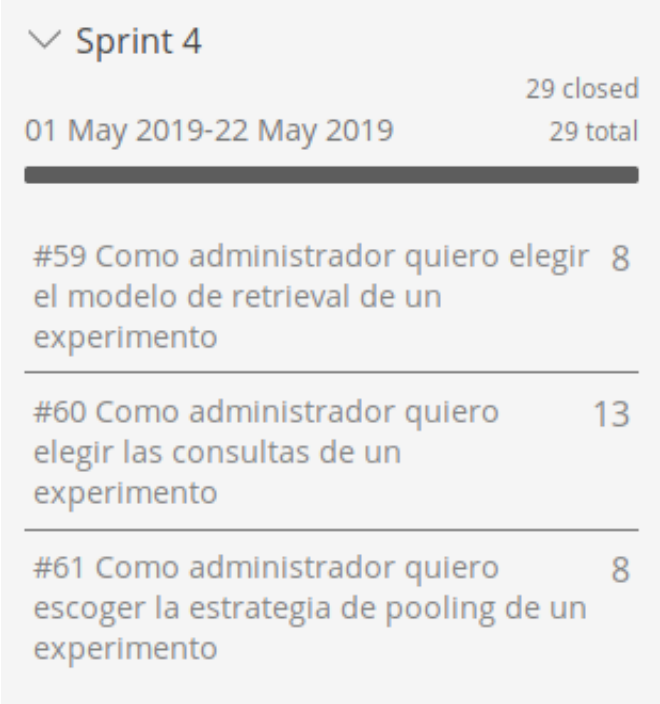

Figura 5.27: Historias completadas en el Sprint 4.

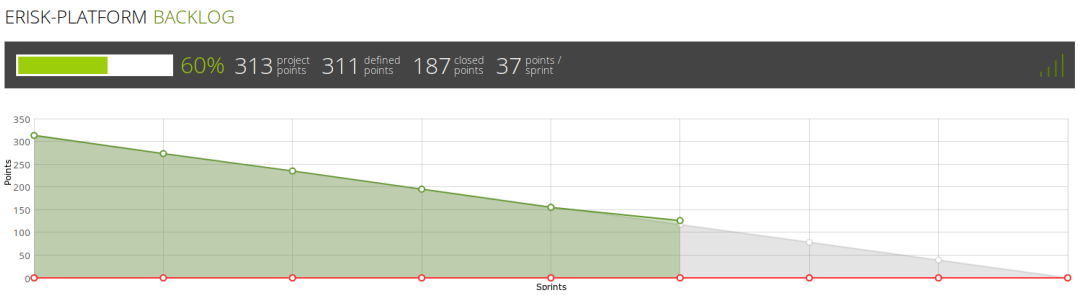

Figura 5.28: Progreso del Sprint 4.

valores, cada uno de ellos expresa una fase diferente en la que se encuentra el experimento:

- *Not crawled*: el experimento simplemente ha sido creado y no se ha hecho nada más.
- *Crawled*: el experimento se creó y se ejecutó para descargar la información de los usuarios y los textos de Reddit, que también ha sido persistido a la base de datos.
- *Pooling*: el proceso de *pooling* ha empezado pero aún no ha terminado.
- *Pooled*: el proceso de *pooling* ya ha terminado.
- *Exported*: la colección asociada al experimento ya ha sido generada al menos una vez.

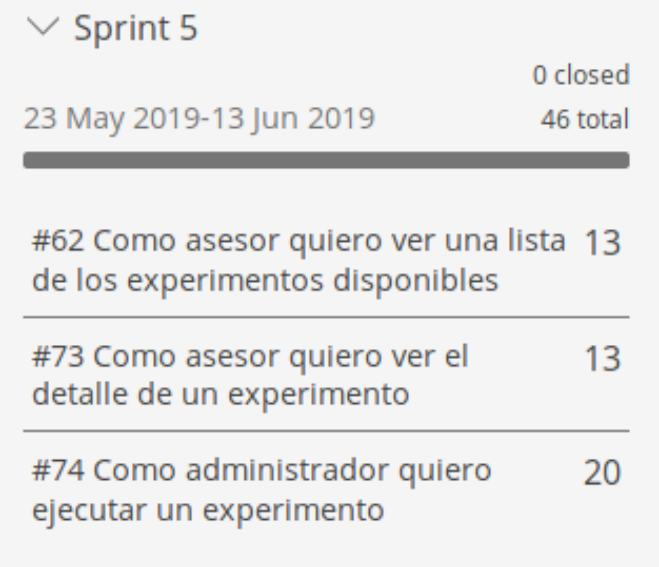

Figura 5.29: Estimación de las historias para el Sprint 5.

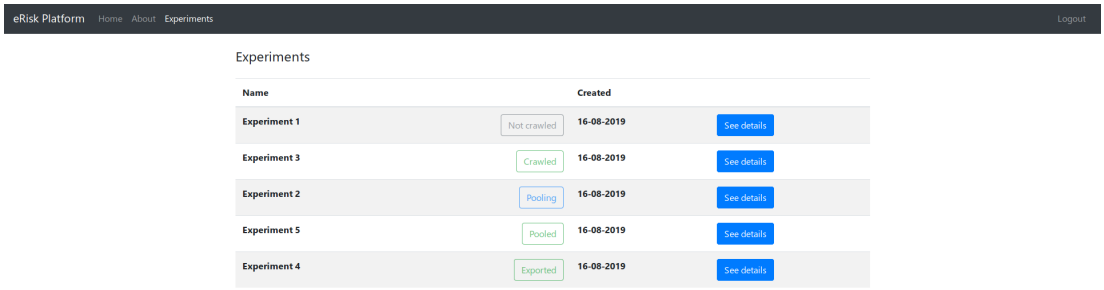

Figura 5.30: Lista de experimentos existentes.

A continuación se realizó la implementación de la historia 73. En las Figuras 5.31 y 5.32 se ilustra el resultado de la implementación de esta historia. En ambas imágenes se puede ver información sobre el estado y la configuración del experimento. En la primera de ellas el proceso de *pooling* ya ha empezado, por lo que el asesor puede continuar este proceso con el botón correspondiente. En la segunda imagen se ilustra un experimento cuya información aún no ha sido descargada, por lo que en este caso el asesor no podría hacer nada, ya que la responsabilidad de ejecutar el experimento es del usuario que lo ha creado.

Balance del *sprint*

Al igual que en el *sprint* anterior, debido a obligaciones del alumno se han implementado menos historias de las estimadas. Debido a esto, la desviación del *sprint* anterior se ha hecho algo más pronunciada.

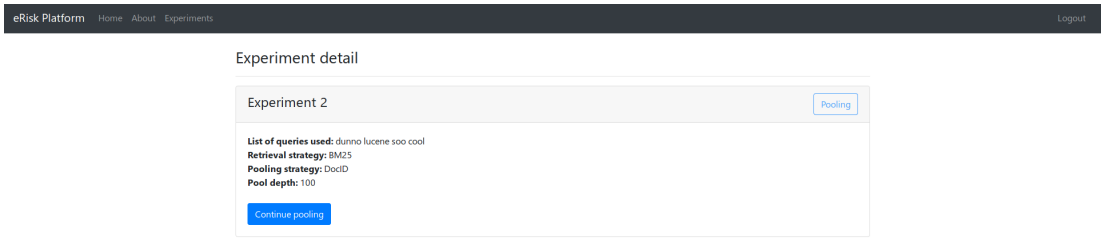

Figura 5.31: Detalle de un experimento que está en proceso de *pooling*.

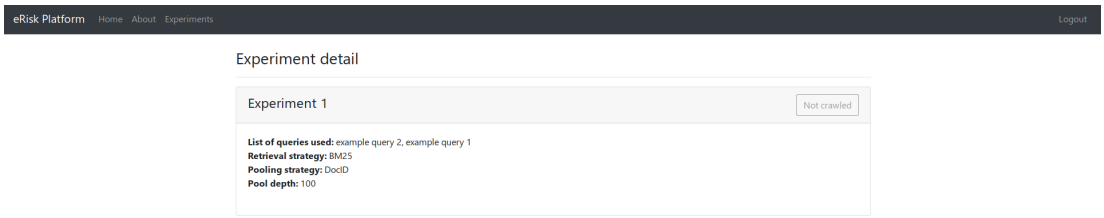

Figura 5.32: Detalle de un experimento que aún no ha sido *crawleado*.

En la Figura 5.33 se pueden ver estas historias completadas, junto con el esfuerzo real de cada una.

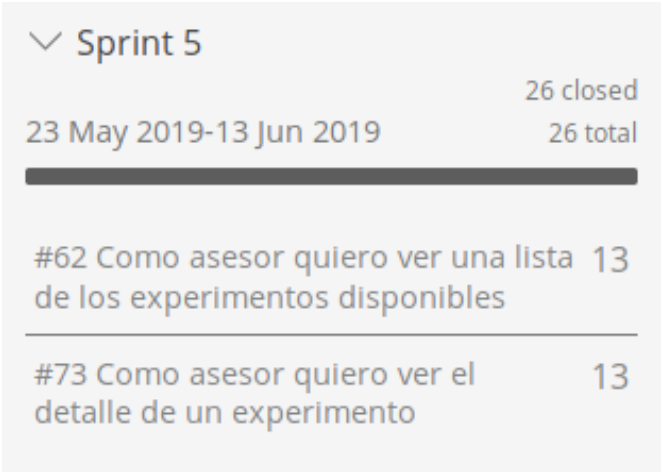

Figura 5.33: Historias completadas en el Sprint 5.

En la Figura 5.34 se expone el progreso del proyecto hasta este punto. Debido a que estos dos últimos *sprints* no han ido como lo estimado, los puntos completados hasta ese punto son menos de los previstos. Deberían haberse completado 240 y solamente se han completado 213. Además, la velocidad también ha bajado con respecto al anterior, estando ahora en una media de 36 puntos por *sprint*.

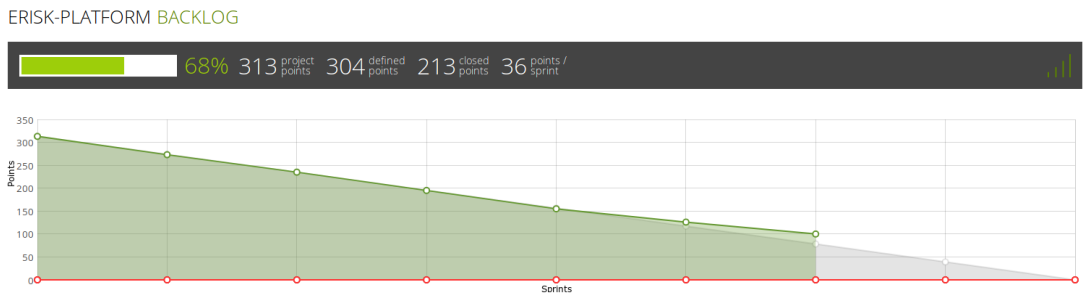

Figura 5.34: Progreso del Sprint 5.

5.5.7 Sprint 6: *Crawling* y *pooling* en un experimento.

En la Figura 5.35 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

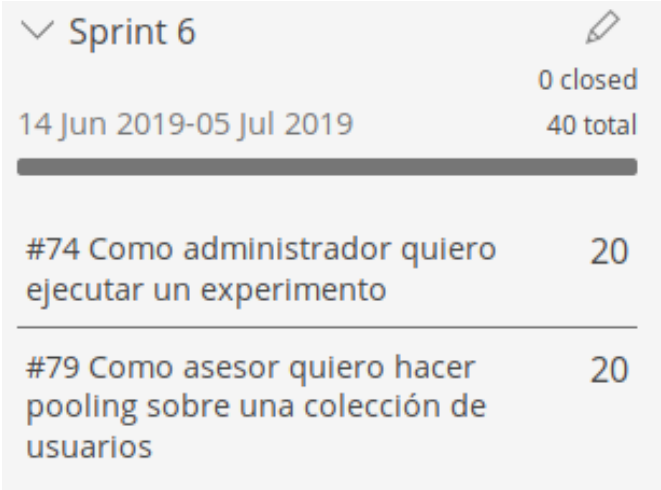

Figura 5.35: Estimación de las historias para el Sprint 6.

Crawling de un experimento

El primer paso después de haber creado un experimento es obtener la información sobre los usuarios y los textos de Reddit para posteriormente poder construir la colección. Para esto, se habilitó un botón en la vista del detalle de un experimento para el rol de usuario experto. Esto se ilustra en la Figura 5.36, en la que se ve el botón que puede accionar el usuario para iniciar el *crawling* de un experimento. Una vez este proceso termina, el experimento pasa al estado de *crawled* y se puede iniciar el proceso de *pooling*.

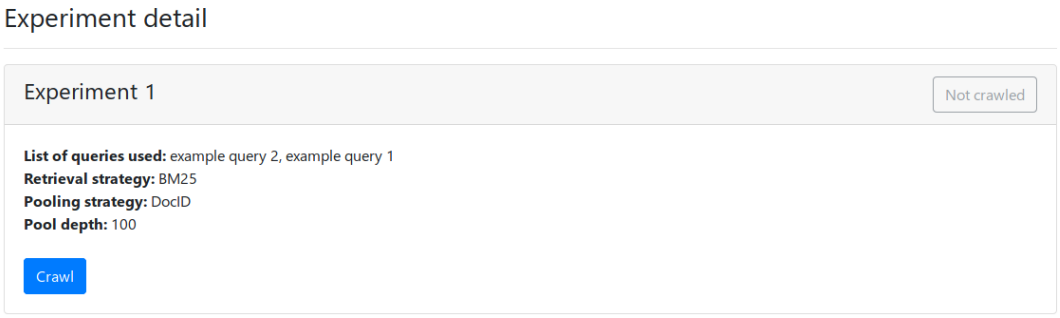

Figura 5.36: Vista de detalle de un experimento, donde se puede iniciar el *crawling*.

Pooling de un experimento

A continuación se implementó el proceso de *pooling*. Para esta historia no existe un reflejo en ninguna vista de la aplicación, ya que durante el proceso de *pooling* el asesor indica la relevancia de un usuario, y esta funcionalidad aún no ha sido implementada.

En este *sprint* se implementaron dos estrategias de *pooling*: DocID [14], que es la estrategia clásica en la que los documentos del *pool* se ordenan por su identificador, y MTF [34], una estrategia en la que los sistemas son priorizados y se juzga el documento más relevante del sistema con más prioridad.

En este momento del *sprint* ya se habían implementado la totalidad de las historias estimadas, pero el *sprint* aún no había terminado, por lo que se añadió una historia más para mantener la duración de este. Esta historia es la 3.

Para implementar la posibilidad de que un asesor emita un juicio sobre un usuario se aprovechó la vista en la que se puede ver toda la historia de un usuario. De esta manera, una vez que el asesor ha leído todas las publicaciones de un usuario, puede decidir si este es relevante o no para el tópico del experimento. Esto se ilustra en la Figura 5.37, donde se ve la historia de un usuario, que en este caso solo ha publicado dos textos. Como esta es la única página de la historia del usuario, en este momento el asesor ya puede juzgar la relevancia de este. Para forzar al asesor a ver toda la historia del usuario estos botones solamente aparecen si el asesor ha llegado a la última página de toda la historia.

Balance del *sprint*

En este *sprint* se han completado más historias de las estimadas inicialmente. Esto se ha debido a que estas historias previstas para este *sprint* han necesitado un esfuerzo menor del previsto. Para mantener la duración del *sprint* se decidió completar una historia más y se ajustó el esfuerzo real de las estimadas inicialmente.

En la Figura 5.38 se pueden ver estas historias completadas, junto con el esfuerzo real de

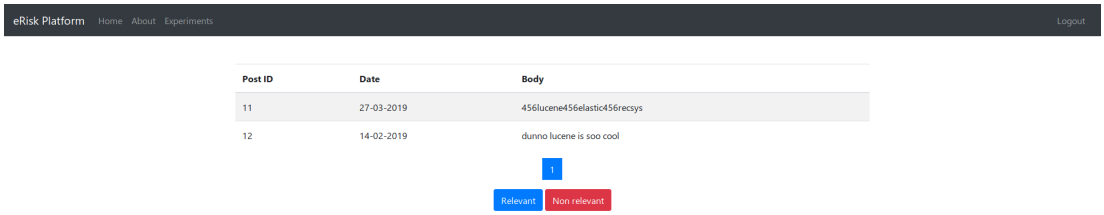

Figura 5.37: Vista de juicio de un asesor.

cada una.

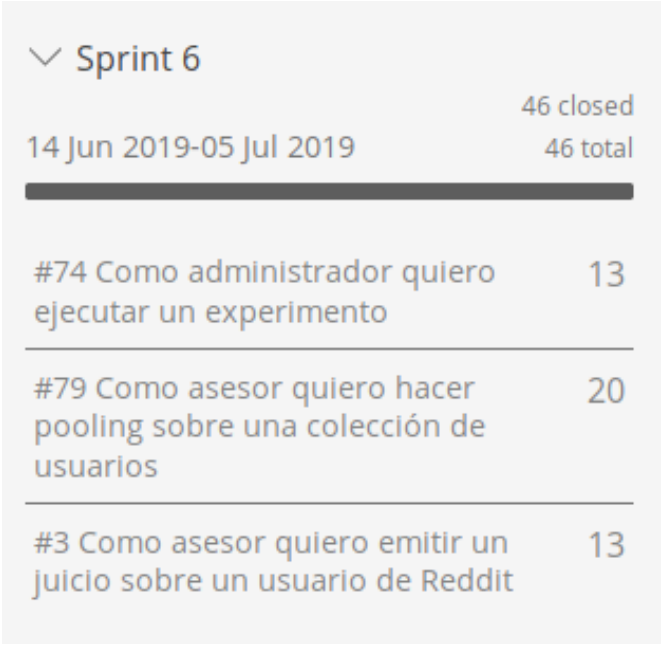

Figura 5.38: Historias completadas en el Sprint 6.

En la Figura 5.39 se ilustra el progreso del proyecto hasta este punto. En este *sprint* se han completado algo más de 40 puntos, por lo que la velocidad ha subido un punto desde el *sprint* anterior. De todas maneras, la desviación acumulada de *sprints* anteriores aún no se ha recuperado del todo.

5.5.8 Sprint 7: Exportación de colecciones y gráficas.

En la Figura 5.40 se pueden ver las historias estimadas para este *sprint* junto con el esfuerzo asociado de cada una.

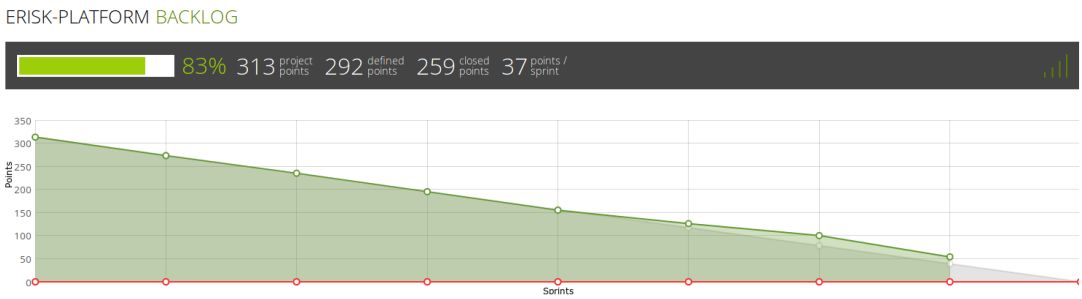

Figura 5.39: Progreso del Sprint 6.

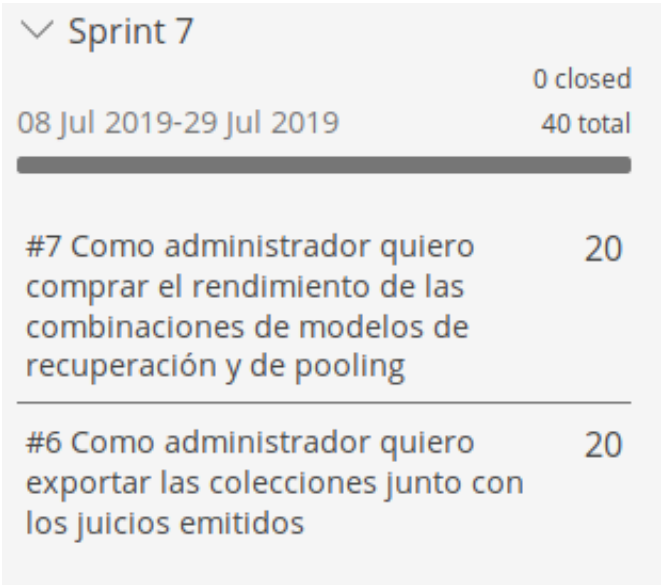

Figura 5.40: Estimación de las historias para el Sprint 7.

Exportación de colecciones

Una vez que el proceso de *pooling* de un experimento ha terminado, se puede proceder a exportar la colección generada, esto es, información sobre los usuarios y los juicios emitidos sobre ellos. Esto se puede hacer desde la vista del detalle de un experimento para un administrador, como se ilustra en la Figura 5.41. Esto es la implementación de la historia 6.

Comparación de diferentes algoritmos

Para poder comparar las diferentes combinaciones de algoritmos de recuperación y estrategias de *pooling* se pueden representar gráficas que ilustran el porcentaje de documentos juzgados como relevantes a lo largo del proceso de *pooling*, como se ve en la Figura 5.42. Esto es la implementación de la historia 7. De esta manera podemos comparar las diferentes combinaciones, ya que las que funcionen mejor podrán encontrar un mayor número de usuarios

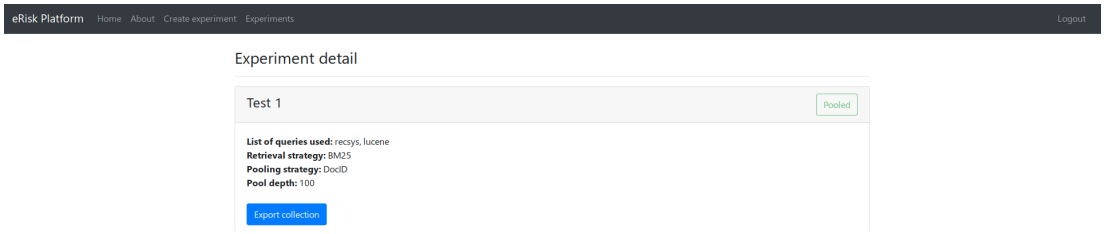

Figura 5.41: Exportación de colecciones.

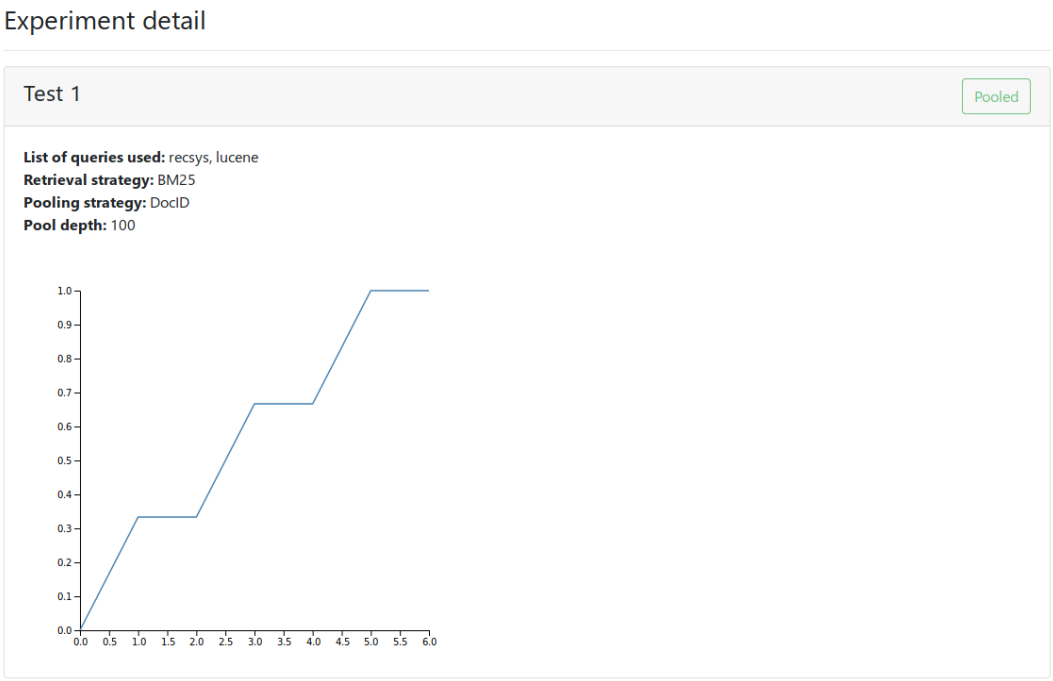

Figura 5.42: Gráfica para comparar algoritmos.

relevantes en una fase más temprana del proceso de *pooling*. Para comparar dos experimentos es necesario crear y ejecutar dos diferentes para así poder obtener y comparar las gráficas de cada uno.

Balance del *sprint*

En este último *sprint* se completó todo el trabajo que se había estimado. En la Figura 5.43 se pueden ver estas historias completadas, junto con el esfuerzo real de cada una.

En la Figura 5.44 se ilustra el progreso del proyecto una vez terminado este último *sprint*. Como ve en esta imagen, se ha conseguido una velocidad media de 37 puntos por *sprint*, que es aproximadamente lo estimado al inicio del proyecto, que era de 40 puntos por *sprint*. Además también se ve que los puntos totales completados son 299, una cifra ligeramente inferior a

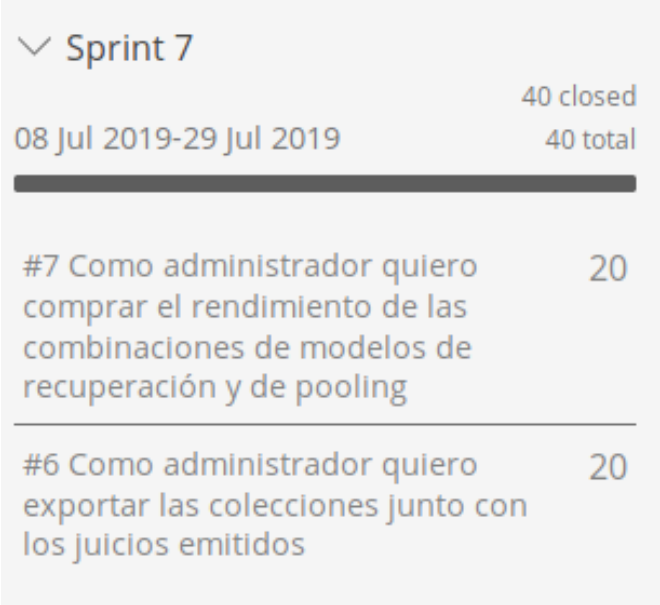

Figura 5.43: Historias completadas en el Sprint 7.

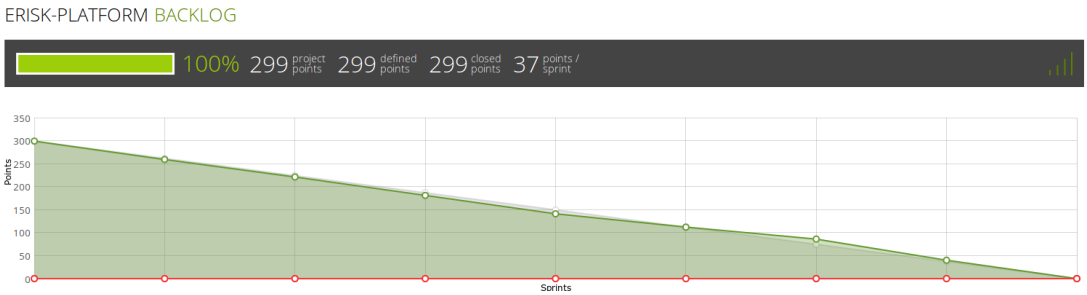

Figura 5.44: Progreso del Sprint 7.

los estimados al principio, 313. Esto indica que el esfuerzo estimado al inicio del proyecto fue algo mayor a lo que realmente resultó al final.

5.5.9 Balance final

Una vez acabado proyecto es necesario hacer un análisis final sobre los resultados obtenidos.

Durante todo el desarrollo del sistema se ha utilizado Taiga como herramienta para gestionar el progreso del mismo: Product Backlog, sprints, tareas... De esta manera, los datos presentados en esta sección con respecto a la planificación y resultados reales son obtenidos por medio de dicha herramienta.

Como se ha comentado anteriormente, en algunos *sprints* no se pudo realizar el esfuerzo estimado debido a otras obligaciones académicas. Aun así, esto se ha compensado con que

algunas tareas se habían sobrestimado al inicio del proyecto, por lo que al final se pudo completar la plataforma en el tiempo estimado. Como se ilustra en el balance final del último *sprint*, los puntos completados totales son menos que los estimados al inicio.

En el Cuadro 5.3 se pueden ver todas las historias completadas durante el desarrollo del presente proyecto. También se indican los puntos historia estimados al inicio y, resaltados en color azul, los puntos historia reales que ocuparon algunas historias que fueron mal estimadas al principio.

ID	**Historia de usuario**	**Puntos**
8	Como administrador quiero recuperar textos de Reddit a partir de una consulta	20 -> 10
9	Como administrador quiero obtener la colección de textos publicados por un usuario	20 -> 10
24	Como administrador quiero obtener la historia de los usuarios que hayan publicado una consulta	10
10	Como administrador quiero persistir los datos obtenidos sobre los usuarios de Reddit	10
37	Como administrador quiero indexar los usuarios recuperadores de Reddit	5
38	Como administrador quiero obtener un ranking de los usuarios recuperados de Reddit	13
39	Como asesor quiero ver toda la historia publicada por un usuario	20
49	Como administrador quiero ver los rankings obtenidos a partir de diversas consultas	10 -> 20
48	Como administrador quiero ver los usuarios recuperados a partir de una consulta	10 -> 20
58	Como administrador quiero crear un experimento	20
57	Como administrador quiero obtener un ranking de la historia de un usuario	20
59	Como administrador quiero elegir el modelo de retrieval de un experimento	8
60	Como administrador quiero elegir las consultas de un experimento	13

61	Como administrador quiero escoger la estrategia de *pooling* de un experimento	8
62	Como asesor quiero ver una lista de los experimentos disponibles	13
73	Como asesor quiero ver el detalle de un experimento	13
74	Como administrador quiero ejecutar un experimento	20 -> 13
79	Como asesor quiero hacer *pooling* sobre una colección de usuarios	20
3	Como asesor quiero emitir un juicio sobre un usuario de Reddit	20 -> 13
7	Como administrador quiero comparar el rendimiento de las combinaciones de modelos de recuperación y de *pooling*	20
6	Como administrador quiero exportar las colecciones junto con los juicios emitidos	20

Cuadro 5.3: Historias de usuario al final del proyecto.

Calidad del proyecto

Un aspecto fundamental a lo largo de todo el desarrollo han sido las herramientas de soporte de integración e inspección continua. Jenkins y Sonar han permitido seguir un flujo desarrollo centrado en hacer la calidad del código un factor importante a lo largo de todo el proceso.

En la Figura 5.45 se puede ver una gráfica con los resultados de las integraciones a lo largo de todo el proceso de desarrollo. En azul aparecen los resultados exitosos y en rojo los fallidos. El eje de abscisas representa el número de integraciones ejecutadas y el eje de ordenadas representa el número de tests implementados.

En cuanto a Sonar, esta herramienta ofrece diferentes métricas que permiten controlar la calidad global del código. Estas métricas se ven en la Figura 5.46, junto con sus valores al final del proyecto.

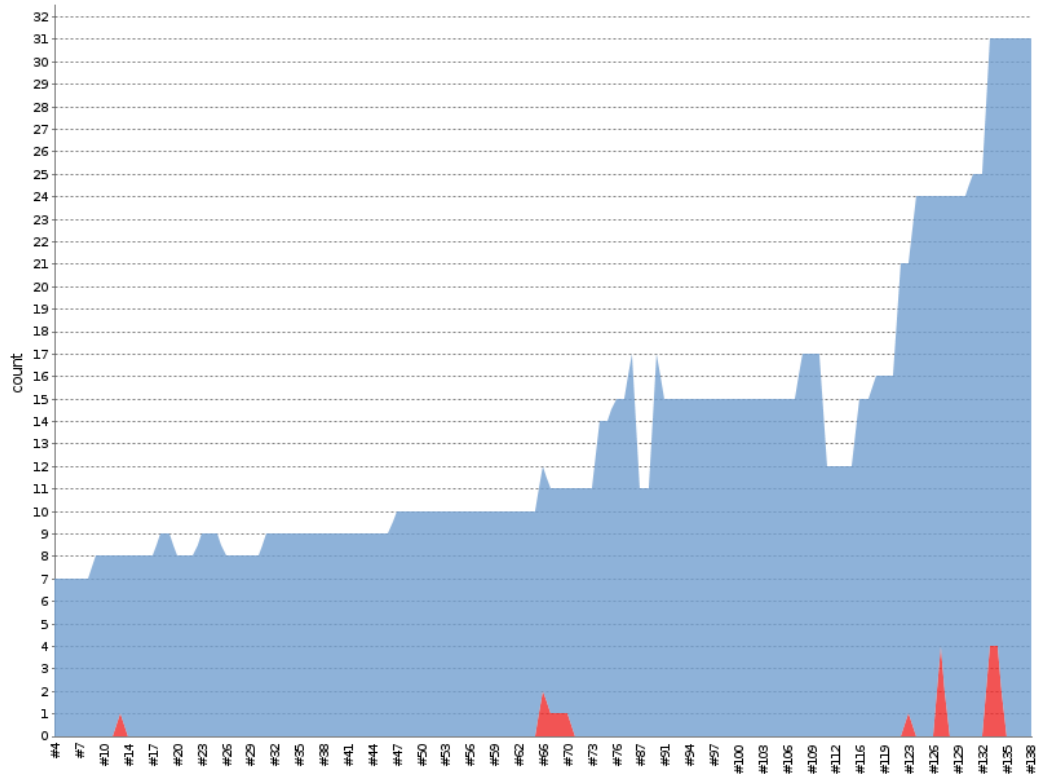

Figura 5.45: Gráfico de integraciones con Jenkins.

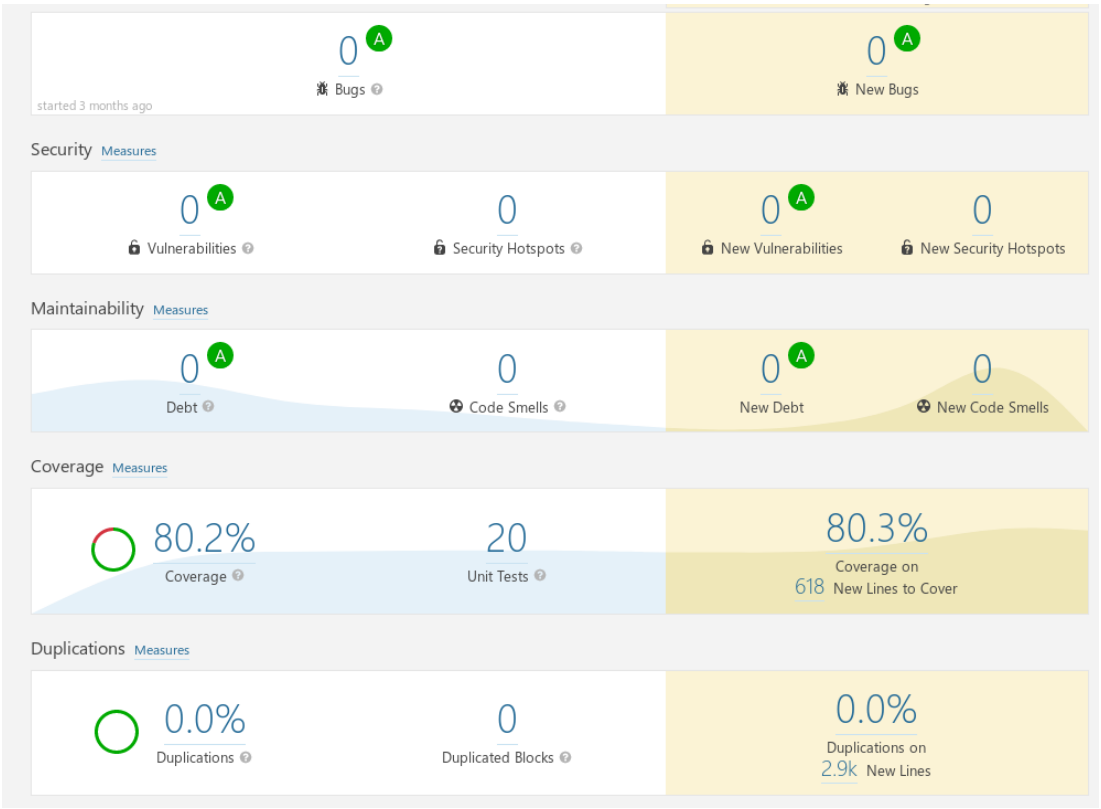

Figura 5.46: Resultados de Sonar al final del proyecto.

Capítulo 6

Conclusiones y trabajo futuro

Con el auge de los sistemas de recuperación que la gente utiliza en su día a día, se convierte en crucial la tarea de elaborar métodos para evaluar la calidad de estos sistemas. Las campañas TREC y el paradigma Cranfield se han convertido en un estándar para realizar esta tarea. Esta metodología requiere de grandes colecciones de documentos que son muy costosas de construir. Hoy en día no existe ninguna metodología que permita configurar y personalizar el proceso de construcción de estas colecciones.

6.1 Conclusiones

El presente proyecto tenía como objetivo diseñar y construir una plataforma que ofrezca una manera fácil de construir colecciones de prueba de calidad. Con este propósito, se ha diseñado un sistema que permite realizar esta tarea de una manera sencilla. Además, se ha conseguido flexibilidad a la hora de poder añadir la implementación de nuevas estrategias de recuperación y métodos de *pooling*, los cuales eran dos objetivos esenciales. Por otra parte, la herramienta permite comprobar la calidad de las combinaciones de los modelos de recuperación con las estrategias de *pooling*, lo que resulta muy útil para el apoyo a la investigación en estos campos. Además, como paso final del proceso de construcción de la colección, la aplicación permite exportar esta.

La plataforma construida es el resultado de todo un proceso de ingeniería y desarrollo. Siguiendo una metodología apropiada y realizando buenas prácticas se ha logrado diseñar e implementar un sistema que cumple con los objetivos especificados, con un alto grado de calidad.

Gracias a la realización de este proyecto se han podido asentar y ampliar muchos conceptos vistos en las clases de teoría y de práctica durante todo el grado. Además, se han aprendido conceptos y tecnologías nuevas, especialmente todo lo relacionado con la disciplina de la Recuperación de Información, un campo que era totalmente nuevo antes de comenzar este

proyecto.

6.2 Publicaciones

Como resultado de todo el trabajo invertido en este proyecto se han podido elaborar dos artículos de investigación que han sido aceptados, mostrando el interés del presente proyecto. El primero de ellos ha sido aceptado en el 9º simposio de *Future Directions in Information Access*, celebrado en la ciudad de Milán. El contenido de este artículo se reproduce en los apéndices de este documento.

Por otro lado, otro segundo artículo fue aceptado en el segundo congreso XoveTIC, que se celebrará los días 5 y 6 de septiembre del año 2019. El contenido de este artículo también se reproduce en los apéndices de este documento.

6.3 Trabajo futuro

Una vez terminado el proyecto, se abren nuevas líneas de trabajo:

- Añadir la implementación de nuevos métodos de recuperación y de *pooling* al sistema.
- Permitir la obtención de los textos de nuevas fuentes de información, no solamente de Reddit.
- Explotación de las colecciones obtenidas mediante el desarrollo de funcionalidades de acceso y búsqueda de información, clustering y clasificación de textos, incluyendo el estudio de la evolución temporal de los tópicos producidos por los usuarios.

Apéndices

Apéndice A

Artículo del FDIA 2019

A continuación se reproduce el artículo íntegro presentado y aceptado en el 9º *Future Directions in Information Access*.

Exploiting Pooling Methods for Building Datasets for Novel Tasks

David Otero[0000−0003−1139−0449]

Information Retrieval Lab
Department of Computer Science
University of A Coruña, Spain
david.otero.freijeiro@udc.es

Abstract. Information Retrieval is not any more exclusively about document ranking. Continuously new tasks are proposed on this and sibling fields. With this proliferation of tasks, it becomes crucial to have a cheap way of constructing test collections to evaluate the new developments. Building test collections is time and resource consuming: it requires time to obtain the documents, to define the user needs and it requires assessors to judge a lot of documents. To reduce the latest, pooling strategies aim to decrease the assessment effort by presenting to the assessors a sample of documents in the corpus with the maximum number of relevant documents in it. The quality of these collections is also crucial, as the value of any evaluation depends on it. In this article, we propose the design of a system for building test collections easily and cheaply by implementing state-of-the-art pooling strategies and simulating competition participants with different retrieval models and query variants. We aim to achieve flexibility in terms of adding new retrieval models and pooling strategies to the system. We want the platform also to be useful to evaluate the obtained collections.

Keywords: Information retrieval · Test collections · Pooling.

1 Introduction

In Information Retrieval, under the Cranfield paradigm, test collections are the most widely used method for evaluating the effectiveness of new systems [15]. These test collections consist of a set of documents, the information needs (topics), and the relevance judgments indicating which documents are relevant to those topics [15]. Collections play a vital role in the process of providing measures to compare the effectiveness of different retrieval models and techniques [14]. However, they are complex and expensive to construct [4, 12]. Some collections of general purpose, such as the ones developed in TREC[1], NTCIR[2]

 FDIA 2019, 17-18 July 2019, Milan, Italy.

[1] https://trec.nist.gov
[2] http://research.nii.ac.jp/ntcir

and CLEF[3], are very useful resources for the evaluation of established tasks, but sometimes research teams need to build their own test collection within a specific domain [6].

When building new collections, it is essential to consider their quality. This aspect is crucial, as they are going to be used to evaluate new developments, and the value of this evaluation depends on it. One common problem is to have biased relevance judgments that unfairly rank some models, as Buckley et al. did observe in TREC AQUAINT 2005 Task [2], or to produce non-discriminative results among systems [13]. Because of this, it is important to have a way of evaluating the collections built.

Nowadays, with the huge growth in the number of novel tasks, it would be convenient to have a cheap way of building the evaluation datasets. When creating an evaluation collection, the most straightforward approach to obtain the relevance judgments is to judge the documents as they are retrieved from the data source. This is a very expensive process because it requires a lot of time from the assessors, as typically they judge many documents that end up not being relevant. This process can be alleviated by using pooling techniques.

Pooling is a well-known approach to extract a sample of documents from the entire document set [15]. Using this technique we avoid judging the entire corpus. When using pooling methods we want to obtain the most complete and unbiased set of relevant documents judged [2]. In community evaluation workshops like TREC, pooling is commonly done over the systems sent by the participants, who run their algorithms on the original dataset and send back their results. [15].

In this article, we present the design of a platform to build test collections. With this platform, we aim to tackle three problems: first, to have an easy and cheap way of building the datasets by reducing the assessor's work; second, to build the most complete and the most unbiased collections that are effective to measure and compare the effectiveness of different systems; finally, we focus also on evaluation as we want the platform to be useful to compare different combinations of retrieval models and pooling strategies to reduce the most of the assessor's work and to evaluate the quality of the obtained collections.

2 Background

System evaluation has been a cornerstone in advance of IR. Building test collections for evaluation is expensive, as it requires the work of human assessors to produce relevance judgments. Pooling strategies aim to reduce this cost, as they allow to build test collections much larger than with *complete* judgments [5]. Pooling allows researches to assume completeness over the judgments with a reasonable degree of certainty. The assessor's work is more profitable when they mark a document as relevant. The documents that are not in the pool are considered being non-relevant. On the other hand, for getting true complete judgments assessors would judge the relevance of every document in the collection. If there

[3] http://www.clef-initiative.eu

are many information needs (queries), they would have to assess the relevance the whole set of documents with respect to every query.

In pooled collections, only a subset –the pool– of the entire corpus is judged. For each topic, the pool of documents is generally constructed by taking the union of the top k –pool depth– documents retrieved by each participant systems, called runs. When we have enough relevant documents in the pool, we can assume that the rest of the documents are non-relevant. These obtained pools are then assessed for relevance.

When we apply pooling strategies, we want to obtain unbiased pools. Unbiased means that the sample of relevant documents obtained does not favour any of model, avoiding to unfairly rank some models over others. Another crucial factor is that when an assessor is judging the obtained pool of documents, the process in which documents are presented can introduce some type of bias to the collection [1]

Historically in TREC, assessors have judged the entire pool following an arbitrary strategy, i.e., by DocID, but a lot of work and research has been done in creating pooling algorithms that impose an order of evaluation intending to reduce the assessment effort without harming the quality of the collection. In particular, in TREC Common Core Track 2017 [1], NIST applied for the first time a pooling algorithm based on Bayesian Bandits [10, 11] which has been demonstrated as an effective and unbiased pooling algorithm which improves the state-of-the-art models.

3 Proposal

In this paper, we present the design of a system for experimenting with the creation of test collections. The main goal of the platform is to address the problem of building test collections for novel tasks at affordable cost.

The main contribution of this platform is that, instead of building the pools with a runs-based approach, we build these systems by combining different query variants and retrieval models. This free us of the need to wait for the participants results. This is very convenient, for example, in competitions where the organizers have to release training data to the participants.

The functionality of our system can be seen from two perspectives: one from the system manager, whose function is to define the user information needs, that are manually created but in a future this process can be automated, and select the retrieval models and the pooling strategies; the second one from an assessor, whose work is to judge the relevance of the documents presented to him.

In Figure 1, we can see an overview of the workflow of the platform: the two roles of the system, the system manager –the competition organizer– and the assessor, along with their tasks.

First of all, the platform allows the manager to create different jobs, each one to produce a different collection with its corresponding information needs and relevance assessments. Different types of collections can be built: for example, a multi-topic dataset in which the manager defines one information need per topic;

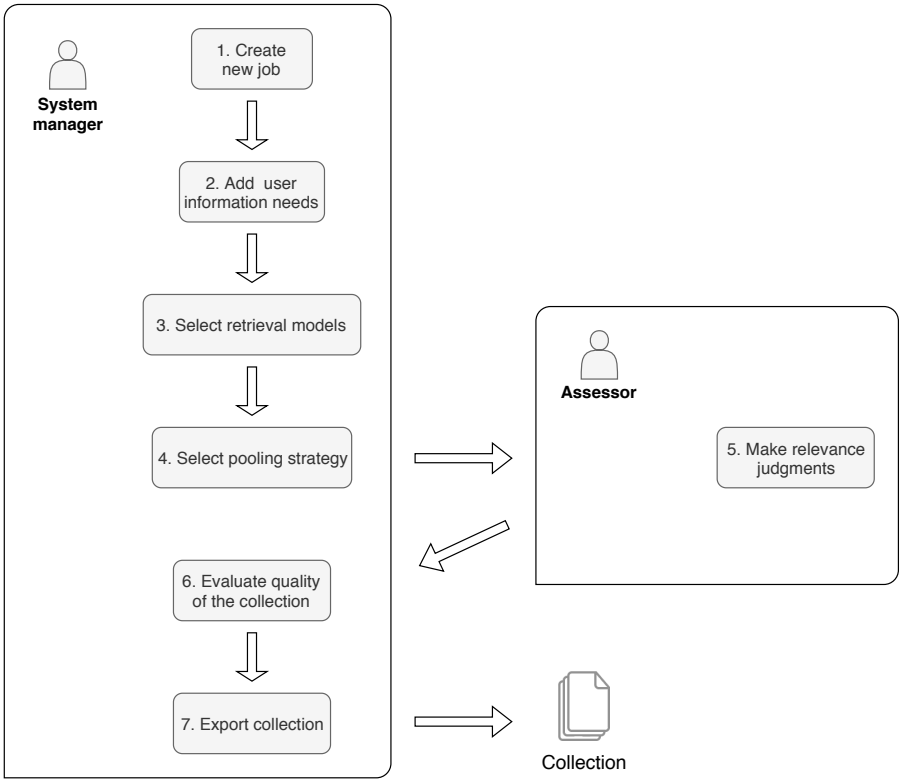

Fig. 1. Platform's workflow overview.

another example is a classification style dataset, in which the manager defines the criteria for the positive cases of each class.

There are two options to obtain the set of documents: the system can use an off-line static collection or can use an API to retrieve documents from an external data source. At this initial stage, we have developed the components to consume documents from the Reddit API. We aim to expand the platform to more data sources soon. We also aim to make the platform flexible to allow the manager to choose among both offline data and different APIs freely.

In TREC-like competitions, each participant sends the results of one or more systems. These results –the runs– are used to build the pool with the top k documents from each system. We propose to build the pool before having runs for participant systems. Here the role of the runs will be played by different query variants and retrieval strategies that the manager can choose to be associated to the job. The top k documents from the runs produced by multiple combinations of query variant and retrieval strategies are used to build the pool.

Our system will allow the manager to select among different state-of-the-art pooling strategies to present the documents to the assessors, such as MTF

[3] and Multi-armed bandits [10, 11]. The function of the assessors is to judge the documents that are presented to them to build the set of judgments of the dataset. Finally, with the documents retrieved, the topics file and the judgments made by the assessors, the platform allows exporting the final collection.

This platform is designed in such a way that is easy to implement and add new retrieval algorithms as well as new pooling strategies. The platform will also be used to analyse the obtained collections. The system will allow the analysis of the different desired properties for a fair evaluation of systems. We want to analyse the combinations of different simulated participants and different pooling strategies in terms of relevant document found at a given budget and the quality of those judgments. The main goal is to reduce the needed time to build the collections drastically. This is achieved by reducing the time that assessor wastes judging non-relevant documents and by allowing faster retrieval of the documents.

3.1 Pilot Task: CLEF eRisk

CLEF eRisk[4] is an initiative organized with the objective of evaluating the effectiveness of methodologies and metrics for the early detection of risks on the Internet, especially those related to health, such as depression, anorexia or self-inflicted harm. For this purpose, collections of texts written by users on social networks are released annually. The lab is mainly oriented to assist advisors who perform diagnoses on users of social networks, as well as to evaluate the effectiveness of different models when building new collections.

Previous tasks have focused on the detection of depression (2017[5]) [7], as well as the detection of anorexia and depression (2018[6]) [8]. The task of 2019 is about anorexia, depression and self-inflicted harm [9]. This lab will serve as pilot task for our systems. We plan to use the platform to build the collections that will be used in the competition in 2020.

4 Conclusions and Future Work

Building cheap and good test collections is crucial for evaluation. We have seen that obtaining the human judgments of these collections is a time and resource consuming task. There are another risks associated with this task: we may end up building collections that have some bias [2] or with incomplete judgments.

In this paper, we have presented the design of an approach whose aim is to tackle those aspects. Our main goal was to have a cheap way of building these datasets by making the most of the assessor's work. We had to leverage that objective with the build of high-quality collections that are *complete* in terms of judgments and, at the same time, unbiased. It was also essential for us to design a flexible platform to include new models and pooling strategies.

[4] http://erisk.irlab.org
[5] https://early.irlab.org/2017
[6] https://early.irlab.org/2018

This work opens an interesting line of future research, which is to compare the quality and usefulness of collections built from participants runs with collections built with other techniques like our approach.

Acknowledgments. This work was supported by projects RTI2018-093336-B-C22 (MCIU/ERDF) and GPC ED431B 2019/03 (Xunta de Galicia/ERDF) and accreditation ED431G/01 (Xunta de Galicia/ERDF). I also would like to thank Daniel Valcarce, Javier Parapar and Álvaro Barreiro for their advise on this work.

References

1. Allan, J., Harman, D., Kanoulas, E., Li, D., Gysel, C.V., Voorhees, E.M.: TREC 2017 Common Core Track Overview. In: Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC 2017, Gaithersburg, Maryland, USA, November 15-17, 2017. vol. Special Pu. NIST (2017)
2. Buckley, C., Dimmick, D., Soboroff, I., Voorhees, E.: Bias and the limits of pooling for large collections. Information Retrieval (2007)
3. Cormack, G.V., Palmer, C.R., Clarke, C.L.A.: Efficient construction of large test collections. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 282–289. SIGIR '98, ACM, New York, NY, USA (1998)
4. Kanoulas, E.: Building Reliable Test and Training Collections in Information Retrieval. Ph.D. thesis, Boston, MA, USA (2009)
5. Kuriyama, K., Kando, N., Nozue, T., Eguchi, K.: Pooling for a Large-Scale Test Collection: An Analysis of the Search Results from the First NTCIR Workshop. Inf. Retr. **5**(1), 41–59 (Jan 2002)
6. Losada, D.E., Crestani, F.: A Test Collection for Research on Depression and Language Use. In: Experimental IR Meets Multilinguality, Multimodality, and Interaction. pp. 28–39. Springer (2016)
7. Losada, D.E., Crestani, F., Parapar, J.: CLEF 2017 eRisk overview: Early Risk prediction on the internet: Experimental foundations. In: CEUR Workshop Proceedings (2017)
8. Losada, D.E., Crestani, F., Parapar, J.: Overview of eRisk 2018: Early Risk Prediction on the Internet (extended lab overview). In: CEUR Workshop Proceedings (2018)
9. Losada, D.E., Crestani, F., Parapar, J.: Early Detection of Risks on the Internet: An Exploratory Campaign. In: Proceedings of the 41st European Conference on Information Retrieval. pp. 259–266. ECIR '19, Springer, Cologne, Germany (2019)
10. Losada, D.E., Parapar, J., Barreiro, Á.: Feeling Lucky?: Multi-armed Bandits for Ordering Judgements in Pooling-based Evaluation. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing. pp. 1027–1034. SAC '16, ACM, New York, NY, USA (2016)
11. Losada, D.E., Parapar, J., Barreiro, A.: Multi-armed bandits for adjudicating documents in pooling-based evaluation of information retrieval systems. Information Processing and Management (2017)
12. Losada, D.E., Parapar, J., Barreiro, A.: Cost-effective Construction of Information Retrieval Test Collections. In: Proceedings of the 5th Spanish Conference on Information Retrieval. pp. 12:1–12:2. CERI '18, ACM, New York, NY, USA (2018)

13. Lu, X., Moffat, A., Culpepper, J.S.: The Effect of Pooling and Evaluation Depth on IR Metrics. Inf. Retr. **19**(4), 416–445 (Aug 2016)
14. Sanderson, M.: Test Collection Based Evaluation of Information Retrieval Systems. Foundations and Trends® in Information Retrieval (2010)
15. Voorhees, E.M., Harman, D.K.: TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing). The MIT Press (2005)

Apéndice B

Artículo del XoveTIC 2019

A continuación se reproduce el artículo íntegro presentado y aceptado en el II Congreso XoveTIC.

proceedings

MDPI

Proceedings

Building High-Quality Datasets for Information Retrieval Evaluation at a Reduced Cost †

David Otero *, Daniel Valcarce, Javier Parapar and Álvaro Barreiro

Information Retrieval Lab, Centro de Investigación en Tecnoloxías da Información e as Comunicacións (CITIC), Universidade da Coruña, 15071 A Coruña, Spain

* Correspondence: david.otero.freijeiro@udc.es; Tel.: +34-881-01-1276

† Presented at II XoveTIC Congress, A Coruña, Spain, 5–6 September 2019.

Published: 1 August 2019

check for updates

Abstract: Information Retrieval is not any more exclusively about document ranking. Continuously new tasks are proposed on this and sibling fields. With this proliferation of tasks, it becomes crucial to have a cheap way of constructing test collections to evaluate the new developments. Building test collections is time and resource consuming: it requires time to obtain the documents, to define the user needs and it requires the assessors to judge a lot of documents. To reduce the latest, pooling strategies aim to decrease the assessment effort by presenting to the assessors a sample of documents in the corpus with the maximum number of relevant documents in it. In this paper, we propose the preliminary design of different techniques to easily and cheapily build high-quality test collections without the need of having participants systems.

Keywords: information retrieval; evaluation; datasets; cost

1. Introduction

In Information Retrieval, test collections are the most widespread technique to evaluate the effectiveness of new developments [1]. These collections are formed by the document set, the information needs (topics) and the human judgments [2]. They are complex to construct because the need of human work to obtain the judgments [3,4]. Datasets of general purpose like TREC (https://trec.nist.gov), NTCIR (http://research.nii.ac.jp/ntcir) and CLEF (http://www.clef-initiative.eu) are useful but sometimes research teams need to build their own collections within a specific task [5].

Pooling methods allow building larger datasets with less effort [6]. When using a pooling approach, only a subset—the pool—of the whole document set is assessed for relevance. The pool is built by taking the union of the top k documents retrieved by each participant system, the runs. In TREC competitions these pools are built using the runs sent by the competition participants, who execute their algorithms on the original dataset and send back their results [2]. Historically, TREC applied the most basic pooling approach (DocID) [2], but recent publications [7,8] have shown that is possible to reduce the assessor's work without harming the quality of the obtained dataset. In particular, in TREC Common Core Track [9] NIST applied these techniques for the first time. The drawback of these techniques is that they are tied to having participant systems, condition that is not always met.

In some cases it may be necessary to obtain collections prior to the competition. Therefore, in these cases, it is not possible to use approaches where the participants are needed, such as CLEF eRisk competition (http://erisk.irlab.org) [10–12], where training data is released.

We propose a method to build the pool before having participant systems. Here the role of the runs will be played by different query variants and out of the box retrieval strategies. The top k documents from the runs produced by multiple combinations of query variants and retrieval strategies are used to build the pool.

2. Experiments

We made a series of experiments to preliminary compare the effectiveness between different pooling approaches. In particular, we want to test if the use of query variants is adequate.

2.1. Systems and Query Variants

We use four different retrieval models: BM25, TF-IDF, LM Jelinek-Mercer and LM Dirichlet. We want to test the effectiveness of this approach having only a few different systems.

To combine with the described models, we build a series of query variants from the original query. With the model×query variant combinations, we can obtain larger pools, ideally having more relevant documents in them. To build a query variant we combine the original query with one of the five terms from the topic description with the highest IDF, i.e., the more specific terms.

Combining these variants along with the systems, we end obtaining a number of different runs equal to $no.\,systems(4) \times no.\,variants(5) = 20$.

2.2. Pooling Algorithms

To perform these experiments we use two pooling algorithms. The first one is the traditional pooling strategy used in TREC competitions [2], i.e., DocID. The second one, DocPoolFreq, is a simple adaptation of the former, where we order the documents by the number of times they appear in the pool and if they tie, by DocID. This is based on the intuition that if a document appears on more systems is it going to be more relevant than other that appears less in all the systems, which is part of many complex pooling algorithms [13].

3. Results

We performed these experiments using the TREC5 dataset (disks 2 & 4, topics 251-300). The results can be observed in Figure 1. When it comes to finding more relevant documents we can observe that the approaches that use the query variants outperform the other two. This is because when using the variants we have more systems, which results in having more relevant documents.

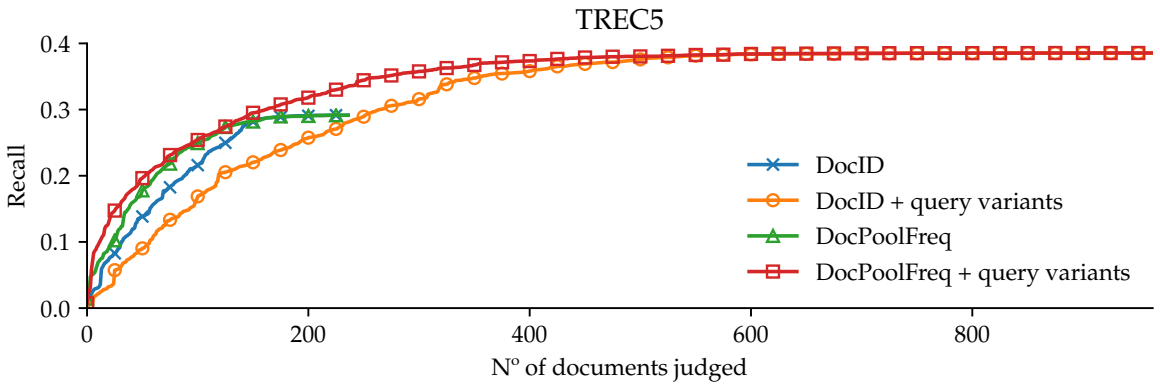

Figure 1. Comparison between pooling strategies.

We also can observe that DocPoolFreq outperforms DocID as it finds relevant documents earlier in the process. This confirms that with only four models and query variants it is possible to obtain the 40% of the relevant documents found in TREC 5 where 61 systems where used to build the pool.

4. Discussion

Results show that our research direction is promising. We also open a line of investigation which is to compare the quality of the datasets built with a participants-based approach with techniques that do not need the participant system, like the presented in this paper.

Funding: This work was supported by projects RTI2018-093336-B-C22 (MCIU & ERDF) and GPC ED431B 2019/03 (Xunta de Galicia & ERDF) and accreditation ED431G/01 (Xunta de Galicia & ERDF).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Sanderson, M. Test Collection Based Evaluation of Information Retrieval Systems. *Found. Trends® Inf. Retr.* **2010**, *4*, 247–375.
2. Voorhees, E.M.; Harman, D.K. *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*; The MIT Press: Cambridge, MA, USA, 2005.
3. Kanoulas, E. Building Reliable Test and Training Collections in Information Retrieval. Ph.D. Thesis, Northeastern University, Boston, MA, USA, 2009.
4. Losada, D.E.; Parapar, J.; Barreiro, A. Cost-effective Construction of Information Retrieval Test Collections. In Proceedings of the 5th Spanish Conference on Information Retrieval, Zaragoza, Spain, 26–27 June 2018; ACM: New York, NY, USA, 2018; pp. 12:1–12:2.
5. Losada, D.E.; Crestani, F. A Test Collection for Research on Depression and Language Use. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 7th International Conference of the CLEF Association, CLEF 2016, Évora, Portugal, 5–8 September 2016*; Springer: Berlin, Germany, 2016; pp. 28–39.
6. Kuriyama, K.; Kando, N.; Nozue, T.; Eguchi, K. Pooling for a Large-Scale Test Collection: An Analysis of the Search Results from the First NTCIR Workshop. *Inf. Retr.* **2002**, *5*, 41–59.
7. Losada, D.E.; Parapar, J.; Barreiro, Á. Feeling Lucky?: Multi-armed Bandits for Ordering Judgements in Pooling-based Evaluation. In Proceedings of the 31st Annual ACM Symposium on Applied Computing, Pisa, Italy, 4–8 April 2016; ACM: New York, NY, USA, 2016; pp. 1027–1034.
8. Losada, D.E.; Parapar, J.; Barreiro, A. Multi-armed bandits for adjudicating documents in pooling-based evaluation of information retrieval systems. *Inf. Process. Manag.* **2017**, *53*, 1005–1025.
9. Allan, J.; Harman, D.; Kanoulas, E.; Li, D.; Gysel, C.V.; Voorhees, E.M. TREC 2017 Common Core Track Overview. In Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC 2017, Gaithersburg, MD, USA, 15–17 November 2017.
10. Losada, D.E.; Crestani, F.; Parapar, J. eRISK 2017: CLEF Lab on Early Risk Prediction on the Internet: Experimental Foundations. Proceedings of 18th Conference and Labs of the Evaluation Forum; Springer: Dublin, Ireland, 2017; CLEF '17, pp. 346–360.
11. Losada, D.E.; Crestani, F.; Parapar, J. Overview of eRisk: Early Risk Prediction on the Internet. Proceedings of 19th Conference and Labs of the Evaluation Forum; Springer: Avignon, France, 2018; CLEF '18, pp. 343–361.
12. Losada, D.E.; Crestani, F.; Parapar, J. Early Detection of Risks on the Internet: An Exploratory Campaign. In Proceedings of the 41st European Conference on Information Retrieval, Cologne, Germany, 14–18 April 2019; pp. 259–266.
13. Losada, D.E.; Parapar, J.; Barreiro, A. A rank fusion approach based on score distributions for prioritizing relevance assessments in information retrieval evaluation. *Inf. Fusion* **2018**, *39*, 56–71.

Glosario

RDBMS *Relational DataBase Management System*: un software para la gestión de bases de datos relacionales.

SQL *Structured Query Language*: lenguaje utilizado para administrar sistemas de bases de datos relacionales.

ACID *Atomicity, Consistency, Isolation and Durability*: son un conjunto de características que garantizan la fiabilidad de las transacciones en una base de datos.

ORM *Object-Relational Mapping*: es una herramienta de programación que establece una correspondencia entre las clases y los objetos de un lenguaje de programación orientado a objetos con las tablas y las filas de una base de datos relacional.

API *Application Programming Interface*: es una práctica de programación para facilitar la comunicación entre componentes de software.

JDBC *Java DataBase Connectivity*: es una API que permite la ejecución de operadores sobre bases de datos desde el lenguaje de programación Java utilizando SQL.

Stakeholder En el marco del desarrollo de un proyecto software, se conoce como stakeholder a cualquier persona y organización que tiene interés en el proyecto.

IoC *Inversion of Control*: es un patrón de diseño en el que se invierto el flujo típico de la programación imperativa. En este caso, en vez de especificar una secuencia de acciones a realizar, se solicita un resultado o recurso que será proporcionado por otro elemento externo.

Bibliografía

[1] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999.

[2] "Google Search Statistics," accedido: 2019-01-10. [En línea]. Disponible en: https://www.internetlivestats.com/google-search-statistics/#trend

[3] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[4] D. E. Losada, F. Crestani, and J. Parapar, "CLEF 2017 eRisk overview: Early Risk prediction on the internet: Experimental foundations," in *CEUR Workshop Proceedings*, 2017.

[5] ——, "Overview of eRisk 2018: Early Risk Prediction on the Internet (extended lab overview)," in *CEUR Workshop Proceedings*, 2018.

[6] ——, "Early Detection of Risks on the Internet: An Exploratory Campaign," in *Proceedings of the 41st European Conference on Information Retrieval*, ser. ECIR '19. Cologne, Germany: Springer, 2019, pp. 259–266.

[7] E. M. Voorhees, "The Philosophy of Information Retrieval Evaluation," in *Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems*, ser. CLEF '01. London, UK, UK: Springer-Verlag, 2002, pp. 355–370. [En línea]. Disponible en: http://dl.acm.org/citation.cfm?id=648264.753539

[8] C. W. Cleverdon, "The Cranfield Tests on Index Language Devices," in *Readings in Information Retrieval*, K. Sparck Jones and P. Willett, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 47–59. [En línea]. Disponible en: http://dl.acm.org/citation.cfm?id=275537.275544

[9] ——, "The Significance of the Cranfield Tests on Index Languages," in *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '91. New York, NY, USA: ACM, 1991, pp. 3–12. [En línea]. Disponible en: http://doi.acm.org/10.1145/122860.122861

[10] K. S. Jones, "The Cranfield tests," in *Information Retrieval Experiment*. Newton, MA, USA: Butterworth-Heinemann, 1981, ch. 13, pp. 256–284.

[11] ——, *Information Retrieval Experiment*. Newton, MA, USA: Butterworth-Heinemann, 1981.

[12] G. Salton, *The SMART Retrieval System—Experiments in Automatic Document Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1971.

[13] D. Otero, D. Valcarce, J. Parapar, and Á. Barreiro, "Building High-Quality Datasets for Information Retrieval Evaluation at a Reduced Cost," *Proceedings*, vol. 21, no. 1, 2019. [En línea]. Disponible en: https://www.mdpi.com/2504-3900/21/1/33

[14] E. M. Voorhees and D. K. Harman, *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press, 2005.

[15] M. Sanderson, "Test Collection Based Evaluation of Information Retrieval Systems," *Foundations and Trends® in Information Retrieval*, 2010.

[16] E. Kanoulas, "Building Reliable Test and Training Collections in Information Retrieval," Ph.D. dissertation, Northeastern University, Boston, MA, USA, 2009.

[17] D. E. Losada, J. Parapar, and A. Barreiro, "Cost-effective Construction of Information Retrieval Test Collections," in *Proceedings of the 5th Spanish Conference on Information Retrieval*, ser. CERI '18. New York, NY, USA: ACM, 2018, pp. 12:1–12:2.

[18] K. S. Jones and C. J. van Rijsbergen, "Report on the need for and provision of an 'ideal' information retrieval test collection," *Computer Laboratory*, 1975.

[19] K. Kuriyama, N. Kando, T. Nozue, and K. Eguchi, "Pooling for a Large-Scale Test Collection: An Analysis of the Search Results from the First NTCIR Workshop," *Inf. Retr.*, vol. 5, no. 1, pp. 41–59, Jan. 2002.

[20] D. Valcarce, "Information Retrieval Models for Recommneder Systems," Ph.D. dissertation, University of A Coruña, A Coruña, Spain, 2019.

[21] X. Lu, A. Moffat, and J. S. Culpepper, "The Effect of Pooling and Evaluation Depth on IR Metrics," *Inf. Retr.*, vol. 19, no. 4, pp. 416–445, Aug. 2016.

[22] J. Allan, D. Harman, E. Kanoulas, D. Li, C. V. Gysel, and E. M. Voorhees, "TREC 2017 Common Core Track Overview," in *Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC 2017, Gaithersburg, Maryland, USA, November 15-17, 2017*, vol. Special Pu. NIST, 2017.

[23] D. E. Losada, J. Parapar, and Á. Barreiro, "Feeling Lucky?: Multi-armed Bandits for Ordering Judgements in Pooling-based Evaluation," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, ser. SAC '16. New York, NY, USA: ACM, 2016, pp. 1027–1034.

[24] D. E. Losada, J. Parapar, and A. Barreiro, "Multi-armed bandits for adjudicating documents in pooling-based evaluation of information retrieval systems," *Information Processing and Management*, 2017.

[25] D. E. Losada and F. Crestani, "A Test Collection for Research on Depression and Language Use," in *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Springer, 2016, pp. 28–39.

[26] K. S. Jones, S. Walker, and S. Robertson, "A probabilistic model of information retrieval: development and comparative experiments: Part 1," *Information Processing & Management*, vol. 36, no. 6, pp. 779 – 808, 2000. [En línea]. Disponible en: http://www.sciencedirect.com/science/article/pii/S0306457300000157

[27] ——, "A probabilistic model of information retrieval: development and comparative experiments: Part 2," *Information Processing & Management*, vol. 36, no. 6, pp. 809 – 840, 2000. [En línea]. Disponible en: http://www.sciencedirect.com/science/article/pii/S0306457300000169

[28] S. Robertson, "The Probability Ranking Principle in IR," *Journal of Documentation*, vol. 33, pp. 294–304, 12 1977.

[29] D. Otero, "Exploiting Pooling Methods for Building Datasets for Novel Tasks," to be published.

[30] "PostgreSQL vs MySQL," accedido: 2019-01-14. [En línea]. Disponible en: https://www.2ndquadrant.com/es/postgresql/postgresql-vs-mysql

[31] V. Driessen, "A successful Git branching model," January 2010. [En línea]. Disponible en: https://nvie.com/posts/a-successful-git-branching-model

[32] K. Schwaber and M. Beedle, *Agile Software Development with Scrum*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[33] "Estudio salarial - Sector TIC Galicia 2015-2016." [En línea]. Disponible en: https://www.scribd.com/document/288511179/Guia-Salarial-Sector-TI-Galicia-2015-2016

[34] G. V. Cormack, C. R. Palmer, and C. L. A. Clarke, "Efficient construction of large test collections," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '98. New York, NY, USA: ACM, 1998, pp. 282–289.