UNIVERSIDAD DE CASTILLA-LA MANCHA
ESCUELA SUPERIOR DE INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA

TRABAJO FIN DE GRADO

SISTEMA DE RECOMENDACIÓN DE IMÁGENES INTEGRADO EN UNA APLICACIÓN DE MENSAJERÍA

Victor Gualdras de la Cruz

Septiembre, 2016

SISTEMA DE RECOMENDACIÓN DE IMÁGENES INTEGRADO EN UNA APLICACIÓN DE MENSAJERÍA

UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

Tecnologías y Sistemas de Información

TECNOLOGÍA ESPECÍFICA DE COMPUTACIÓN

TRABAJO FIN DE GRADO

SISTEMA DE RECOMENDACIÓN DE IMÁGENES INTEGRADO EN UNA APLICACIÓN DE MENSAJERÍA

Autor: Victor Gualdras de la Cruz

Director: Dr. Jesús Serrano Guerrero

Septiembre, 2016

Victor Gualdras de la Cruz

Ciudad Real – Spain

E-mail: victor.gualdrasla@alu.uclm.es
Teléfono: 679 532 016

TRIBUNAL:

Presidente:

Vocal:

Secretario:

FECHA DE DEFENSA:

CALIFICACIÓN:

PRESIDENTE	VOCAL	SECRETARIO
Fdo.:	Fdo.:	Fdo.:

Resumen

Abstract

English version of the previous page.

Agradecimientos

Escribe aquí algunos chascarrillos simpáticos. Haz buen uso de todos tus recursos literarios porque probablemente será la única página que lean tus amigos y familiares. Debería caber en esta página (esta cara de la hoja).

Juan[1]

[1] Sí, los agradecimientos se firman

A alguien querido y/o respetado

Índice general

Resumen V

Abstract VII

Agradecimientos IX

Índice general XIII

Índice de cuadros XV

Índice de figuras XVII

Índice de listados XIX

Listado de acrónimos XXI

1. Introducción 1

2. Objetivos 3

2.1. Objetivo general 3

2.2. Objetivos específicos 4

2.2.1. Desarrollo de una interfaz para la comunicación entre los usuarios . 4

2.2.2. Diseñar un protocolo de comunicación entre dispositivos 4

2.2.3. Implementar un repositorio flexible de imágenes 4

2.2.4. Desarrollo de un mecanismo para la obtención y análisis de nuevas imágenes 4

2.2.5. Diseño de un algoritmo matemático para la recomendación de imágenes 5

3. Antecedentes 7

3.1. Sistemas de recomendación 7

3.1.1. Clasificación de los sistemas de recomendación 8

3.1.2. Técnicas de recomendación 11
3.1.3. Dilemas de los sistemas de recomendación 13
3.1.4. Sistemas de recomendación híbridos 16

4. Método de trabajo **17**
4.1. Metodología de desarrollo 17
4.1.1. Scrum 17
4.1.2. Aplicación al proyecto 21
4.2. Herramientas utilizadas 21
4.2.1. Herramientas Hardware 21
4.2.2. Herramientas Software 22

A. Ejemplo de anexo **31**

Referencias **33**

Índice de cuadros

3.1. Sitios web y elementos que recomiendan 10
3.2. Técnicas de recomendación 11

4.1. Primera versión del Product Backlog 26
4.2. Diferencia entre Google Datastore y los RDBMS 26

Índice de figuras

4.1. Roles en Scrum. Fuente: https://www.scrum.as/ 19
4.2. Esquema de la estructura un Sprint. Fuente: http://www.i2btech.com/ 20
4.3. Android Studio logo 23
4.4. Google Cloud Platform logo 24

Índice de listados

Listado de acrónimos

HU	Historias de Usuario
TFG	Trabajo Fin de Grado
SO	Sistema Operativo
IDE	Entorno de Desarrollo Integrado
ADT	Android Developments Tools
GCP	Google Cloud Platform
IAAS	Infrastructure as a Service
PAAS	Platform as a Service
SAAS	Software as a Service
ACID	Atomicity Consistency Isolation Durability
CRUD	Create Read Update Delete
RDBMS	Relational DataBase Management System
API	Application Programming Interface
URL	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol
ML	Machine Learning
REST	Representational State Transfer
GCM	Google Cloud Messaging
OO	Object Oriented

Capítulo 1

Introducción

Capítulo 2

Objetivos

En este capítulo se establecerá el principal objetivo que se pretende alcanzar mediante este proyecto, desglosando este a su vez en objetivos específicos que será necesario alcanzar para su consecución.

2.1 Objetivo general

El objetivo principal que se pretende lograr en este Trabajo Fin de Grado es el desarrollo de un sistema de recomendación de imágenes, que integrado en una aplicación de mensajería, sea capaz de a partir de una entrada proporcionada por el usuario, sugerir aquellas imágenes que, en base a ciertos criterios que se especificará a continuación, representen mejor aquello que el usuario desea transmitir. Este sistema estará principalmente orientado para aquellas personas que por un motivo u otro presenten dificultades a la hora de trabajar con las aplicaciones clásicas de mensajería.

Las imágenes a recomendar serán principalmente pictogramas, que resultan más intuitivos que las imágenes tradicionales a la hora de representar conceptos, especialmente para gran parte del público para el que se diseña esta aplicación. Sin embargo, estos pictogramas a veces pueden no ser suficientes, o puede darse el caso de que simplemente no se disponga de un pictograma adecuado para la entrada del usuario. En estos casos será necesario hacer uso de recursos que se encuentren en la red, donde será muy importante hacer uso de fuentes fiables de información.

Respecto al procedimiento a seguir por parte del sistema, lo primero que será necesario es preprocesar la entrada, de manera que la búsqueda de imágenes no quede limitada a la palabra o palabras empleadas por el usuario. A continuación, será necesario establecer qué imágenes tienen más posibilidades de representar aquello que el usuario pretendía transmitir. Para esto se emplearán varias técnicas utilizadas en los sistemas de recomendación. Será necesario establecer perfiles entre los diferentes usuarios de la aplicación. Estos perfiles permitirán relacionar a usuarios con gustos similares de manera que sea más fácil sugerir imágenes que ya han sido previamente seleccionadas por otro usuarios con un perfil similar. De esta manera, el sistema deberá mejorar con la cantidad de usuarios y el uso que se haga de este.

Es necesario recalcar que el objetivo principal del proyecto es el de desarrollar un sistema de recomendación, y no una aplicación de mensajería, por lo que la mayor parte del esfuerzo se centrará en el desarrollo de este sistema, careciendo la aplicación de algunas de las funcionalidades que caracterizan a las populares aplicaciones de mensajería ya existentes en el mercado.

2.2 Objetivos específicos

El objetivo discutido anteriormente será abarcado mediante los siguientes objetivos específicos.

2.2.1 Desarrollo de una interfaz para la comunicación entre los usuarios

Será necesario desarrollar la interfaz de una aplicación de mensajería. Para ello se recurrirá a una aplicación Android. Esta aplicación deberá presentar las características básicas comunes a toda aplicación de mensajería, como una lista con los contactos del usuario que tengan la aplicación. También, contará con un chat que permita la comunicación tanto mediante texto como mediante imágenes, y que permita al usuario realizar la entrada de manera oral y/o escrita. Para el almacenamiento de mensajes intercambiados por el usuario con otros se dispondrá de una base de datos.

2.2.2 Diseñar un protocolo de comunicación entre dispositivos

Se desarrollará un protocolo de comunicaciones, el cual proporcionará una estructura que permita la comunicación entre los diferentes dispositivos y usuarios que hagan uso de la aplicación. Para ello contará con los siguientes componentes. Por un lado, un almacén de datos situado en en la red, el cual contendrá a todos los usuarios de la aplicación junto con los datos de estos, permitiendo a los distintos usuarios descubrir a aquellos de entre sus contactos que también disponen de la aplicación. Por otro lado se deberá establecer un mecanismo que se encargue de hacer llegar los mensajes que envía un determinado usuario de la aplicación a otro permitiendo la comunicación entre estos.

2.2.3 Implementar un repositorio flexible de imágenes

Será necesario implementar un sistema capaz de almacenar y gestionar la información relativa a estas imágenes. Deberá permitir la posibilidad de añadir nuevas imágenes junto con información relativa de estas, además de poder modificar información de imágenes ya almacenadas.

2.2.4 Desarrollo de un mecanismo para la obtención y análisis de nuevas imágenes

Se deberá desarrollar un mecanismo capaz de realizar búsquedas de imágenes en sitios de terceros cuando las imágenes propias sean insuficientes. Debe existir la posibilidad de

poder configurar los sitios de terceros sobre los que se realizan esas búsquedas. También será necesario poder realizar un análisis de estas nuevas imágenes, con el fin de extraer la máxima información posible de estas.

2.2.5 Diseño de un algoritmo matemático para la recomendación de imágenes

Se deberán determinar aquellas variables que resultan más determinantes a la hora de recomendar las imágenes. Para ello se valorará si estas variables dependen de características propias de las imágenes únicamente, o si también influyen en ellas características propias de los usuarios. Con todos estos datos será necesario establecer un algoritmo que, valorando todas estos datos en el grado adecuado, sea capaz de determinar que imágenes podrán satisfacer en mayor grado las necesidades del usuario y proceder así a su recomendación.

Capítulo 3

Antecedentes

En este capítulo se discutirán algunos de los conceptos y tecnologías que se han empleado para la elaboración de este proyecto. Por una parte se hablará sobre los sistemas de recomendación, explicando en que consisten y profundizando especialmente en los sistemas de recomendación híbridos.

3.1 Sistemas de recomendación

En la vida diaria, las personas recurren a las recomendaciones o críticas por parte de otras cuando sus experiencias personales o conocimientos sobre el tema son insuficientes. Los sistemas de recomendación actuales desempeñan un papel similar a este proceso. En 1997 Resnick y Varian [RV97] definieron por primera vez los sistemas de recomendación como sistemas en los que "la gente proporciona recomendaciones como entrada al sistema, el cual luego se encarga de agregar y redirigir estas a los destinatarios adecuados". Posteriormente se ha ampliado esta definición, considerando sistemas de recomendación aquellos que generan recomendaciones personalizadas o que son capaces de guiar al usuario hacia elementos que sean de su interés dentro de un amplio abanico de posibilidades. Es esta característica de individualización o recomendación personalizada lo que distingue a estos sistemas de otros como los buscadores simples y los sistemas de recuperación de información.

Estos sistemas son claves hoy en día donde toda la información que se encuentra en la red en prácticamente cualquier portal, trasciende la capacidad de los usuarios para buscar y seleccionar de entre todos los elementos por si mismo. Un ejemplo de sistema de recomendación es el sistema de la compañía Netflix, que proporciona recomendaciones de películas en base a las anteriores visualizaciones de los usuarios. Esta compañía realizó en 2006 una competición en la que retó a la comunidad a desarrollar un sistema de recomendación que fuese capaz de derrotar al de la compañía *Cinematch* [BEL+07]. Para ello ofreció al público una pequeña fracción de sus datos en la que se encontraban valoraciones anónimas de usuarios sobre películas. Se inscribieron 20.000 equipos, de los cuales 2.000 presentaron al menos una solución. En 2009 se concedió un premio de 1.000.000 de dolares a un equipo que mejoró la precisión de Cinematch en un 10 %. Todos estos números pueden servir para hacerse una idea del potencial que ofrecen los sistemas de recomendación hoy en día.

Evolución histórica de los sistemas de recomendación

El primer sistema de recomendación que se desarrollo fue Tapestry [GNOT92] a comienzos de los 90, y sus desarrolladores lo denominaron com el termino *Filtro colaborativo* que fue adoptado por otros desarrolladores pero que en la actualidad ha quedado en desuso debido principalmente a que los actuales sistemas no solo filtran aquellos elementos o productos que no son deseables, si no que tratan de sugerir aquellos que puedan aportar mayor valor, además de que como se verá mas adelante algunos de los tipos de sistemas de recomendación que fueron apareciendo posteriormente no tenía en cuenta las opiniones de estos clientes. Este primer sistema pionero tenía como propósito el de filtrar el correo electrónico, así como artículos de noticias online. Es especialmente en este campo del filtrado de noticias donde los primeros sistemas de recomendación tuvieron mayor auge.

Pese a que este sistema es el primer sistema considerado de recomendación, algunos años antes ya se propusieron algunos trabajos y teorías sobre sistemas para el filtrado de elementos como el trabajo de Housman y Kaskela [HK70] que buscaba mantener a los científicos informados de los nuevos documentos mediante un matching entre las palabras clave que estos establecían de su interés y el contenido de los nuevos artículos, sin embargo esta estrategia no logro los resultados deseados. Otra aproximación fue la de la creación de modelos de usuario que aparece en el trabajo de Allen [All90]. Por otra parte, el sistema *The information Lens* [MGT86] planteo un enfoque diferente hacia los sistemas de recomendación, estableciendo reglas que se apoyaban en la estructura que tienen la mayoría de los mensajes de correo electrónico y en palabras clave de estos, permitiendo a los usuarios utilizar estas estructuras como plantillas de manera que fuese más fácil para estos establecer los mecanismos de filtro. Todos estos sistemas fueron dando forma a los diferentes tipos y categorías de sistemas de recomendación que existen hoy en día.

3.1.1 Clasificación de los sistemas de recomendación

Desde su aparición se han realizado distintas clasificaciones de los sistemas de recomendación. En este caso se va a mostrar la clasificación que realizaron Resnick y Varian [RV97]. De este modo, se establecen una serie de características según las cuales se podría realizar la clasificación de estos sistemas. En primer lugar se establecen las siguientes características técnicas:

- El contenido de la recomendación, es decir la valoración a los distintos elementos a recomendar. Esta valoración puede ser tan simple como un bit (recomendado o no) o algo más complejo como un un porcentaje o un texto.
- El modo en el que se realizan o recogen las recomendaciones. Las recomendaciones pueden ser realizadas de manera explícita, pero también pueden ser recogidas de manera implícita por el sistema, por ejemplo analizando las preferencias de los usuarios, las búsquedas y visitas de estos a diferentes portales o las compras previas de estos.

Existen mecanismos muy útiles para poder recoger estas recomendaciones implícitasm como las cookies en los sistemas web.

- La identidad de los recomendadores. Esta puede ser la identidad real, un pseudónimo o bien una identidad anónima. Como se verá más adelante, los usuarios prefieren conservar su privacidad en numerosas ocasiones, y pueden ser reacios a compartir información de carácter sensible aun cuando esta información puede ser crucial para el sistema de recomendación, por esta razón es necesario ofrecer mecanismos que les permitan conservar dicha privacidad.
- Las técnicas de recomendación, es decir, la manera en la que se relacionan las recomendaciones con aquellos que buscan recomendación. Esta es una de las características que permiten mayor flexibilidad en este tipo de sistemas y será analizada en profundidad más adelante.
- La finalidad de las evaluaciones. Uno de los usos es el de descartar o sugerir elementos. Otra finalidad puede ser la de ordenar los elementos recomendados según un peso, o mostrar para cada elemento el nivel de recomendación.

Además de las características técnicas, otro de los elementos que caracterizan un sistema de recomendación es su dominio, es decir, los elementos o items que se recomiendan, y el público que realiza o recibe las distintas recomendaciones.

En lo referente a los elementos sobre los que se aplican las recomendaciones es importante, en primer lugar, definir el tipo de los elementos que se están recomendando. En el Cuadro 3.1 se muestran algunos ejemplos de sitios y los elementos que recomiendan. Otro factor importante es el volumen de los elementos que se recomiendan, así como la frecuencia con la que se generan y desaparecen. Es necesario conocer y tener en cuenta estos parámetros ya que se deben de tratar de manera muy distinta unos elementos que se generan con gran frecuencia y tienen un tiempo de vida corto, como pueden ser las noticias de un medio electrónico en el que es muy importante poder recomendar dichas noticias en un tiempo acotado, a la necesidad de recomendar por ejemplo películas o libros. Es aquí donde asumen gran importancia las técnicas de recomendación. Por otro lado, se encuentran los costes que implican las diferentes posibilidades en cuanto a la recomendación. Es posible que el coste de fallar en la recomendación de un buen o mal ítem sea alto, o por otro lado el coste de un análisis intensivo puede ser mayor. Esta característica es altamente dependiente de la configuración de las características técnicas mencionadas anteriormente.

En el caso de los usuarios involucrados en el proceso de las recomendaciones, tanto aquellos que las realizan como los que las consumen, es necesario conocer los perfiles de estos. Por ejemplo se debe saber si los usuarios tienden a realizar recomendaciones de numerosos elementos similares, o por el contrario evalúan solo elementos muy específicos, dando lugar a diferentes conjuntos de recomendaciones. También es importante conocer la cantidad de

Sitio	Elementos recomendado
Amazon	Libros/otros productos
Facebook	Amigos
WeFollow	Amigos
MovieLnes	Películas
Nanocrowd	Películas
Jinni	Películas
Findory	Noticias
Digg	Noticias
Zite	Noticias
Meehive	Noticias
Netflix	DVDs
CDNOW	CDs/DVDs
eHarmony	Citas
Chemistry	Citas
True.com	Citas
Perfectmatch	Citas
CareerBuilder	Trabajos
Monster	Trabajos
Pandora	Música
Muffin	Música
StumbleUpon	Páginas Web

Cuadro 3.1: Sitios web y elementos que recomiendan (RESNICK [LMY+12])

usuarios que componen o compondrían el sistema, y la variedad respecto a los gustos de los usuarios, por ejemplo, existiendo una gran cantidad de usuarios con gustos similares.

3.1.2 Técnicas de recomendación

Como se ha mencionado anteriormente existen varias técnicas de recomendación que permiten adaptarse a la situación en cuestión. A la hora de elegir una de estas técnicas, es necesario tener en cuenta principalmente tres factores. Por un lado se encuentran los datos y la información de la que disponemos antes de comenzar el proceso de recomendación, como pueden ser las valoraciones de ciertos usuarios de los items, o información sobre dichos items. Por otro lado está la entrada que realiza el usuario, es decir la información que este comunica al sistema con el propósito de obtener una recomendación. Finalmente es necesario disponer mecanismo o algoritmo que sea capaz de combinar los dos elementos anteriores para poder llevar a cabo la recomendación. Mediante estos tres factores se pueden definir un total de cinco técnicas de recomendación. Considerando **U** como el conjunto de los usuarios de los que se conocen las preferencias, **I** el conjunto de los items sobre los que existen valoraciones, **u** el usuario para el que realizar la recomendación e **i** el elemento en cuestión que se esta considerando para ser recomendado, en el cuadro 3.2 se muestra a grandes rasgos como funcionan estas técnicas en función de los factores anteriores.

Técnica	Información previa	Entrada	Algoritmo
Colaborativo	Evaluaciones de **U** de los elementos en **I**.	Evaluación de **u** de los elementos en **I**.	Identifica a los usuarios en **U** con perfiles similares a **u**, y extrapola sus valoraciones de **i**.
Basada en contenido	Características de los elementos en **I**.	Las valoraciones de **u** de los elementos en **I**.	Generar un clasificador que adapte el comportamiento respecto a las valoracioes de **u** y lo use en **i**.
Demográfica	Información demográfica sobre **U** y sus evaluaciones de los elementos en **I**.	Información demográfica sobre **u**.	Identificar aquellos usuarios que son demográficamente similares a **u** y extrapolar sus evaluaciones de **i**.
Basada en utilidad	Características de los elementos en **I**.	Una función de utilidad sobre los elementos en **I** que describa las preferencias de **u**.	Aplicar la función sobre los elementos y determinar la clasificación de **i**.
Basada en conocimiento	Características de los elementos en **I**. Conocimiento de como estos elementos cumplen las necesidades del usuario.	Una descripción de las necesidades o intereses de **u**	Inferir la afinidad del item **i** con las necesidades de **u**.

Cuadro 3.2: Técnicas de recomendación (BURKE [Bur02])

A continuación se muestran en más detalle las diferentes técnicas.

Colaborativa

La técnica colaborativa es probablemente la más usada dentro de este tipo de sistemas. En los sistemas que usan esta técnica, cada usuario tiene generalmente un perfil en el que se encuentran las evaluaciones que ha realizado este de cada elemento. El funcionamiento de esta técnica consistiría en comparar al usuario, o en este caso su perfil, con el de otros para encontrar las similitudes entre ellos, y usar estas similitudes para extrapolar la información sobre las evaluaciones u opiniones de estos usuarios sobre el item que se pretende evaluar o recomendar. Un ejemplo simple de como sería un perfil de usuario en estos sistemas consiste en una estructura de datos en el que se encuentran los elementos evaluados junto con las valoraciones realizadas por el usuario. Los sistemas que emplean esta técnica están pensados para generar relaciones a largo plazo, ya que por norma general mejoran su funcionamiento conforme aumentan las recomendaciones. Es por esto, que en algunos casos es recomendable establecer mecanismos que sean capaces de modificar estos datos en función del tiempo de manera ajena al usuario. Esto es debido a que el usuario puede haber variado sus gustos u opiniones a lo largo de este tiempo, y las evaluaciones pueden no ser fiables.

Demográfica

En este tipo de recomendadores, es necesario establecer una serie de perfiles o clases definidos de manera previa. Posteriormente se obtiene la información personal de los usuarios y se usa esta para categorizarlos en alguna de las clases demográficas definidas anteriormente. Los mecanismos para obtener estos datos pueden ser variados, usando mecanismos para obtenerlos de manera explícita como puede ser un test, u otros mecanismos que extraigan esta información de manera implícita. Esta técnica es similar a la colaborativa en el uso de los perfiles de los usuarios para sus recomendaciones, sin embargo la ventaja de esta sobre la anterior es que no necesita evaluaciones previas de los items por parte de los usuarios.

Basada en contenido

Esta técnica es una extensión de los estudios en el filtrado de información. A diferencia de la técnica colaborativa en la que se establecían relaciones entre los usuarios, en esta se establecen relaciones entre los los elementos a recomendar. Se comparan aquellas características que se encuentran en los diferentes elementos estableciendo el grado de similitud en los valores de estas. Usando esta información es posible sugerir aquellos elementos que tienen características en común con aquellos que el usuario ha valorado positivamente. De esta manera, el perfil de usuario se establece basándose en aquellos elementos que han gustado, o dicho de otra manera que el usuario ha valorado positivamente. Esta técnica puede ser muy útil en un sistema de recomendación de películas por ejemplo, donde encontramos elementos que comparten las mismas características como el reparto, la temática o la fecha de lanza-

miento. Al igual que los sistemas colaborativos, estos también mejoran su rendimiento con el aumento de las valoraciones de los usuarios.

Basada en utilidad

Los sistemas que emplean esta técnica realizan recomendaciones a partir de un cálculo sobre la utilidad que tiene cada elemento para satisfacer las necesidades del usuario. Es necesario por tanto establecer una función de utilidad para cada usuario, siendo esta función la que componga su perfil. Este tipo de técnica presenta la ventaja de poder incluir en sus cálculos y recomendaciones atributos que no son propios del elemento que se quiere recomendar como tal, como puede ser por ejemplo la confianza en el vendedor de dicho producto en casos de comercio en línea, o el tiempo de entrega. Esto permite que estos sistemas sean capaces de valorar y negociar de acuerdo a distintas características, por ejemplo negociando el precio en función del tiempo de entrega.

Basada en conocimiento

Esta técnica realiza las recomendaciones a parir del conocimiento de como un determinado item satisface las necesidades o deseos de un usuario. El perfil de usuario en este tipo de reomendadores será cualquier estructura que sea capaz de representar las necesidades de los usuarios. Del mismo modo, el conocimiento sobre los elementos podrá ser representado y almacenado de diversas formas. Al igual que los recomendadores basados en utilidad, los basados en conocimiento no están diseñados para mejorar su funcionamiento con la adición de usuarios.

3.1.3 Dilemas de los sistemas de recomendación

Los sistemas de recomendación presentan principalmente dos tipos de problemas, uno que se deriva de la técnica utilizada y que va ligado a esta debido a su funcionamiento y por tanto que tiene carácter técnico, y otro tipo de problema que depende en mayor medida del dominio, es decir de los elementos que se recomiendan y los usuarios que hacen uso del sistema.

Problemas del dominio

Uno de los problemas en los que los usuarios tienen un papel protagonista es la privacidad, que aunque no es exclusivo de estos sistemas adquiere gran importancia. En estos sistemas suele ser común que la información más valiosa sea también aquella que el usuario es más reacio a compartir. En algunos sistemas como hemos visto anteriormente, es posible la participación en las recomendaciones de manera anónima o bajo un pseudónimo, sin embargo en muchas ocasiones esto puede no ser posible cuando por ejemplo los usuarios quieren reconocimiento al realizr una recomendación, pero no dar a conocer todos los detalles de su información personal. Por esto es necesario en algunos sistemas el proporcionar mecanismos

que presenten un grado intermedio de privacidad, ajustándose a las necesidades y deseos de los usuarios. Otro problema que es común a todas las técnicas es el de la creación del perfil de usuario. Muchos usuarios al principio pueden no ver el potencial que puede llegar a tener este sistema y por tanto considerar la inversión de tiempo necesaria para formar su perfil demasiado alta respecto con el beneficio inmediato que le puede aportar. Es por esto que se requiere que los sistemas presenten incentivos para que estos usuarios contribuyan en la creación de los perfiles, o que por otro lado, estos sistemas de recomendación sean capaces de recopilar estas preferencias o necesidades de los usuarios sin necesidad de intervención explicita de estos. Esto es más fácil en algunas técnicas que en otras.

Cuando las recomendaciones dependen de las opiniones de los usuarios como en el caso de las técnicas colaborativas, surge otro problema conocido como "vote early and often.º votar pronto y a menudo en español. Este fenómeno consiste en manipular de alguna manera los votos o en este caso las recomendaciones. En el caso de que cualquiera pueda realizar todas las recomendaciones que desee, los creadores de contenido o proveedores de distintos productos o servicios recurrirán a votar en favor de sus elementos y en detrimento de sus competidores.

Comparaciónusuarios de técnicas

Existen problemas van ligados a la técnica escogida, y todas las técnicas presentan una serie de ventajas y desventajas entre si.

Uno de los problemas más conocidos es el problema del comienzo frío, o cuesta arriba. Este problema se puede dar en dos casos, cuando aparece un nuevo usuario y cuando aparece un nuevo item. Cuando aparece un nuevo usuario no se tiene información de el, y en el caso de ser necesario relacionarlo con algún otro usuario o con algún perfil concreto resulta complicado. El caso de un nuevo item es similar. Cuando aparece un nuevo item y apenas existen recomendaciones sobre este, aparecen problemas a la hora de relacionar este nuevo item con otros. En ambos casos, si el sistema requiere de recomendaciones, es necesario que este presente algún tipo de incentivo para que los usuarios realicen valoraciones sobre los elementos.

En el caso de los sistemas colaborativos estos se ven afectados por los problemas tanto de nuevo usuario como de nuevo item. Debido a sus características, este sistema presenta problemas en aquellos entornos en los que existen pocos usuarios y estos evalúan continuamente los mismos items (problema conocido como dispersión), o bien los usuarios tienen gustos muy diferentes. En definitiva, este tipo de sistemas resulta útil cuando existe un número suficientemente amplio de usuarios y con una variedad de gustos aceptable dentro del dominio del problema. Una de las ventajas que presentan estos sistemas respecto a otros es principalmente su capacidad para poder recomendar elementos que pese a que no estén dentro del mismo grupo, pueden estar relacionados de alguna manera. Por ejemplo puede darse

la situación de que a todos aquellos a los que les gusta la música Jazz les guste también el mismo grupo de rock, y este sistema sería capaz de recomendar dicho grupo, mientras que otros como por ejemplo el basado en contenido no podría. Otra gran ventaja de esta técnica, es la de ser independiente de la representación a nivel de computador de los elementos que recomienda. Debido a esto es frecuentemente usado en la recomendación de elementos u objetos complejos como libros, música o películas.

Los sistemas basados en contenido sufren de manera menor del problema de la dispersión debido a que solo tienen en cuenta las valoraciones del propio usuario y no de todos los usuarios. A pesar de esto, tienen el mismo problema de comienzo en frío que tenían los sistemas colaborativos. Además estos sistemas, a diferencia de los colaborativos están limitados por las características o los datos de los elementos que se recomiendan. Otro problema que tiene en común con los colaborativos surge como consecuencia de que estos sistemas generalmente no deben recomendar un objeto que el usuario ya ha valorado, y que por tanto se entiende que dispone de el o lo conoce lo suficiente. Aunque esta condición no es siempre necesaria, ya que el puede ser útil recomendar algo que el usuario ya haya seleccionado anteriormente, esta opción puede no ser deseada. El problema que se presenta aquí es el de distinguir cuando dos elementos son lo suficientemente parecidos para ser considerados iguales y no ser recomendados al mismo usuario tras haberlo hecho con uno de ellos.

Los sistemas demográficos también sufren el problema del comienzo frío, pero solo respecto a la aparición de nuevos items. También comparten otro de los problemas de los colaborativos que es el de identificar a usuarios que se salen de lo común, es decir, que o bien no tienen características en común con ninguno de los actuales grupos, o presentan algunas características de cada grupo pero no las suficientes como para categorizarlos en uno concreto. Pese a no tener el problema de comienzo frío con los nuevos usuarios, necesitan recopilar información sobre los distintos perfiles antes de comenzar a operar. Esta recopilación y formación de perfiles es su principal diferencia respecto a los sistemas colaborativos.

Pese a que estos sistemas mencionados anteriormente, que están basados en el aprendizaje durante la ejecución, parecen tener muchos problemas especialmente al inició, presentan la ventaja de ser más flexibles y adaptativos que las dos técnicas que se verán a continuación. Tanto los sistemas basados en utilidad como los basados en conocimiento carecen de los problemas de comienzo frío y de dispersión ya que no aprenden durante su ejecución. Sin embargo en ambos casos es necesario un gran trabajo previo.

En el caso de los sistemas basados en utilidad, pese a carecer del problema del comienzo frío necesitan crear la función de utilidad considerando todas las características del objeto. Esto supone un gran trabajo al inicio y mucha interacción con el usuario, lo cual supone una desventaja para usuarios inexpertos o usuarios que no quieren invertir demasiado tiempo en esto, pero que facilita en gran medida el filtro y selección de aquellos usuarios expertos o que buscan elementos con unas características muy específicas. Como se mencionó en la sección

anterior, la gran ventaja de estos sistemas es poder considerar aquellas propiedades que no son intrínsecas al objeto, como el tiempo de el formato del envío.

Por otra parte, los sistemas basados en conocimiento comparten el problema de la recolección de datos o conocimiento con los sistemas clásicos basados en el conocimiento, en los que se necesita de un experto del que extraer información y de un ingeniero del conocimiento que sea capaz de extraer dicha información. Respecto a los tipos de conocimiento que estos sistemas requieren se pueden categorizar en tres. Por una parte encontramos el conocimiento sobre los items o elementos que se van a recomendar. También encontramos el conocimiento sobre el usuario y las necesidades de este. Finalmente está el conocimiento funcional, que permite establecer relación entre las necesidades del usuario y las características de los objetos.

3.1.4 Sistemas de recomendación híbridos

En la sección anterior se ha visto que cada tipo de sistema de recomendación tiene unas fortalezas y debilidades, las cuales en muchos casos son complementarias. Es decir, algunos sistemas poseen aquello que les falta a otros y viceversa. Siendo esto así surge una idea, la de combinar varios de estos sistemas tratando de buscar sacar el máximo potencial posible de cada uno.

Capítulo 4

Método de trabajo

A lo largo de este capítulo se planteará la metodología escogida para el desarrollo de este proyecto, justificando su elección y explicando sus principales características así como el primer planteamiento para afrontar el problema. También se mencionarán las herramientas software y hardware empleadas en la realización tanto del proyecto como de esta documentación.

4.1 Metodología de desarrollo

Para la realización de este proyecto se ha escogido una metodología ágil [IIS04], en concreto se va a aplicar el marco Scrum. Las metodologías ágiles se caracterizan por el desarrollo del proyecto mediante interacciones incrementales que finalizan con un entregable, el cual aporta una determinada funcionalidad de manera que pueda ser evaluado por el usuario. Mediante estos entregables tempranos se pretende reducir el riesgo de desarrollar un producto que no se ajuste a las necesidades del usuario, permitiendo añadir cambios y adaptarse a los requisitos de este durante su desarrollo.

4.1.1 Scrum

Scrum es un marco de desarrollo ágil basado en ciclos de desarrollo iterativos e incrementales para la gestión y desarrollo de proyectos [Sch04]. Scrum establece un conjunto de prácticas y roles que se deberán emplear en el desarrollo de un proyecto con el objetivo de minimizar los riesgos que conlleva este desarrollo. Una de las principales características de este modelo, es que se busca el desarrollo colaborativo, en el que todo el equipo trabaja de manera autoorganizada y sin la intervención de agentes externos al proyecto o producto que se desea desarrollar. El equipo de desarrollo de Scrum debe trabajar de manera muy cercana, comunicándose entre si continuamente, y realizando revisiones de todo lo que se ha desarrollado con gran frecuencia para asegurar la mayor calidad posible del producto. Debido a su capacidad de adaptabilidad a las necesidades del usuario y a los cambios, es especialmente útil en entornos donde existe volatilidad en las necesidades de los usuarios, y donde se necesitan obtener resultados tangibles en un tiempo corto.

Como se ha mencionado anteriormente *"Scrum no es un proceso o una técnica para la creación de productos; si no un marco dentro del cual puedes emplear diferentes técnicas y*

procesos" [SS16]. Scrum esta fundamentalmente basado en el enfoque empírico, revisando continuamente el trabajo que se ha llevado a cabo para mejorar. Esta basado principalmente en tres pilares. Por un lado encontramos la **transparencia**, que implica que todos los aspectos importantes del proyecto deben ser conocidos por todos los integrantes del equipo Scrum, y que además deben existir estándares conocidos y seguidos por los implicados, de manera que todos los integrantes entiendan aquello que se esta representando. Por otro lado debe existir una **inspección**, que pese a que no debe ser tan frecuente que se interponga en el desarrollo del trabajo, debe ser suficiente como para asegurar que el producto se desarrolla con la calidad adecuada. Finalmente, el marco Scrum y especialmente el equipo que implementa dicho marco, deben ser capaces de **adaptarse** a los cambios que tras las inspecciones se consideren necesarios para asegurar el correcto desarrollo del producto.

Roles en Scrum

Como se ha mencionado anteriormente, solo aquellos miembros que se encuentren dentro del proyecto serán los encargados de tomar decisiones que afecten a dicho proyecto, sin intervención de terceras personas ajenas a este. Así pues el equipo de Scrum es auto-suficiente y auto-organizado, y es este equipo el encargado de tomar todas las decisiones que afecten a este desarrollo. Scrum define tres grandes roles dentro de su estructura, aunque luego pueda existir una distribución más amplia dentro de estos.

- **Product Owner (Propietario del producto):** Es la persona que representa al cliente y a los stakeholders (o personas interesadas en el producto). Es el encargado de maximizar el valor del producto, asegurándose de que las necesidades se cumplen de la manera más exitosa posible. Para se encarga de redactar las Historias de usuario, dotarlas de prioridad y añadirlas al Product Backlog.
- **Scrum Master (Facilitador):** Es el encargado de facilitar la realización del proyecto por parte del equipo. Debido a la característica autoorganizativa de Scrum no representa el papel clásico de líder de proyecto, sino que actúa como facilitador del equipo, eliminando o minimizando aquellos obstáculos que puedan perjudicar al desarrollo del proyecto. También es el encargado de hacer cumplir el marco de Scrum, así como de liderar o dirigir las reuniones para asegurarse de que son productivas y se desarrollan mediante la idelogía Scrum. Ayuda al Product Owner en la realización del Product Backlog, de manera que se tenga claro que elemento es necesario desarrollar a continuación para que el equipo pueda seguir trabajando continuamente.
- **Development Team (Equipo de desarrollo):** Este equipo es el encargado de materializar las historias de usuario y las necesidades del proyecto en entregables iterativos ya finalizados para que sean evaluados. Dentro de este equipo debe haber personas con las capacidades adecuadas para el desarrollo de todas las funcionalidades necesarias en cada incremento. Debido a esto los equipos se componen de personal con capacidades

variadas y complementarias. Estos equipos deben ser de un tamaño reducido de manera que la autoorganización pueda funcionar, y pueda existir una gran comunicación entre el equipo.

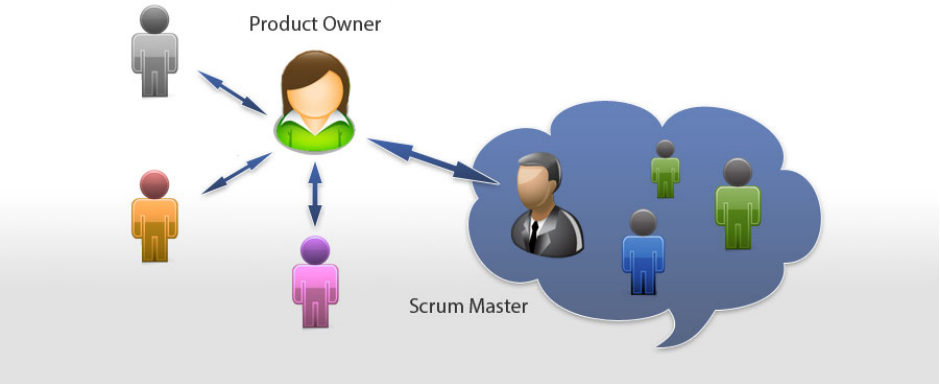

Figura 4.1: Roles en Scrum. Fuente: https://www.scrum.as/

Sprint

El Sprint es la piedra angular en Scrum. Representa un bloque de tiempo de una a cuatro semanas, que debe finalizar con la entrega de un elemento incremental que aporte funcionalidad y que pueda ser evaluado.

Eventos

Del concepto de Sprint se derivan una serie de eventos relacionados con este. En la figura 4.2 se puede observar el esquema que se seguiría para cada Sprint.

- **Sprint Planning Meeting (Reunión de planificación del Sprint):** Esta reunión se lleva a cabo al comienzo de cada Sprint, y en ella se decide como se va a organizar el Sprint, estableciendo que elementos del Product Backlog se van a desarrollar, así como periodos y plazos. La duración máxima es de unas cuatro horas para los Sprint de 2 semanas y ocho para los Sprint de un mes.
- **Daily Scrum (Scrum Diario):** Durante esta reunión diaria de unos quince minutos aproximadamente, se evalúa el trabajo realizado desde el ultimo Daily Scrum, se plantea el objetivo para ese día, y se evalúan los posibles problemas o dificultades que puedan afectar a la consecución del objetivo.
- **Sprint Review (Revisión del Sprint):** Este evento se realiza cada vez que se finaliza un Sprint, y durante esta revisión se evalúa el trabajo realizado y se presentan los avances al cliente o stakeholders.
- **Sprint Retrospective (Retrospectiva de Sprint):** Durante este evento el equipo evalúa aquello que se hizo mal o puede mejorarse, y se ponen en marcha las medidas

necesarias para solventar los errores o implementar las mejoras necesarias. Es una revisión que se realiza tras el Sprint Review y antes de que se comience con el siguiente Sprint.

Figura 4.2: Esquema de la estructura un Sprint. Fuente: http://www.i2btech.com/

Artefactos

Los artefactos en Scrum ayudan al equipo a la consecución de los objetivos y a la planificación y estructuración de estos, permitiendo proporcionar la transparencia necesaria para que todos puedan tener una imagen de lo que se necesita, y de la situación del proyecto, permitiendo una adaptación y mejora continua.

- **Product Backlog (Lista de Producto):** El Product Backlog contiene todos los elementos necesarios bien sean características como fallos por resolver, requisitos no funcionales, mejoras o cualquier otro elemento que sea necesario llevar a cabo para lograr llevar a cabo el proyecto. Es visible a todo el equipo, pero solo podrá ser cambiado con el consentimiento del Product Owner. Los items que conforman esta lista se ordenan en función de la prioridad para su resolución, y deben contener además una estimación del tiempo necesario para llevar dicha tarea o elemento a cabo. Es necesario explicar en que consisten los items que se añaden a esta lista, que pese a que no es obligatorio que sean de un formato concreto, suelen consistir en Historias de Usuario o HU. Estas HU describen una necesidad, funcionalidad o característica, que el sistema deberá contener para su desarrollo exitoso. Contienen únicamente aquello que debe hacerse, pero no el cómo, y deben ser lo suficientemente claras y cortas como para ser

fácilmente entendibles. Un ejemplo de plantilla para estas HU se definió como *Como <role>, quiero <objetivo/deseo>para <beneficio>*. También se ha remarcado la posibilidad de suprimir la parte del *para* cuando no sea necesario o no aporte nada de utilidad.

- **Sprint Backlog (Lista de tareas pendientes del Sprint):** Esta lista contiene aquellas tareas que se han seleccionado del Product Backlog para su desarrollo en el Sprint actual. Estas historias se pueden desglosar en tareas a las que se les asigna un tiempo de realización. Generalmente se suele utilizar un mecanismo que se puede implementar mediante una pizarra o mediante herramientas más complejas como herramientas software, en el que se clasifican las tareas o historias según su situación actual dentro del Sprint como *to do* si aún no se ha empezado a trabajar en ella, *in progress* si se esta trabajando y *done* si esta completa.

4.1.2 Aplicación al proyecto

Siendo el marco explicado anteriormente el elegido para el desarrollo de este proyecto, es el momento de explicar como se ha planteado este proyecto de acuerdo al marco Scrum. En primer lugar es preciso identificar al equipo Scrum, el cual pese a no ser recomendable el hecho de que el Scrum Master y el Product Owner sean la misma persona, debido al carácter docente de este trabajo era la única solución. Así pues el equipo queda de la siguiente manera:

- Product Owner: Jesús Serrano Guerrero
- Development Team: Victor Gualdras de la Cruz
- Scrum Master: Jesús Serrano Guerrero

En segundo lugar, se precisa la realización de una primera versión del Product Backlog 4.1 de manera que el grupo de desarrollo pueda comenzar a trabajar en el proyecto.

4.2 Herramientas utilizadas

En esta sección se trataran las principales herramientas utilizadas para el desarrollo de este proyecto, tanto hardware como software, incluyendo las APIs más importantes.

4.2.1 Herramientas Hardware

A continuación se muestran los dispositivos hardware utilizados en el desarrollo de este TFG

- Ordenador portátil personal sobre el que se realizará el desarrollo del proyecto.
- Dos dispositivos Smartphone con Sistema Operativo Android sobre los que se realizarán las pruebas.

4.2.2 Herramientas Software

Las herramientas software más importantes empleadas en este proyecto se muestran a continuación.

Lenguajes de programación

Java

El lenguaje de programación Java es un lenguaje de propósito general y alto nivel, cuyas principales características son las de ser Object Oriented (OO). Además el código generado por Java tiene la característica de que una vez compilado puede ejecutarse en cualquier plataforma que que soporten Java.

Java será el principal lenguaje en el que se desarrolle la aplicación Android, ya que es el lenguaje sobre el que funcionan las aplicaciones para esta plataforma. Si bien se usará algún otro lenguaje como XML para las interfaces, Java es el más importante.

Python

Python es un lenguaje interpretado de alto nivel, que al igual que java es de propósito general. Ofrece la generación de código generalmente más breve que otro que cumpla el mismo propósito en otros lenguajes, y es comúnmente utilizado como lenguaje de prototipado rápido.

Será utilizado para el desarrollo del servidor de este proyecto debido a su facilidad y potencia de uso en el ámbito de las comunicaciones, y especialmente por su fácil integración con Google Cloud Platform (GCP).

LaTeX

LaTeX [Lat] es un procesador de textos para la preparación de documentos de gran calidad de tipografía especialmente diseñado para redactar documentos que pretenden ser publicados en algún medio, especialmente largos artículos técnicos o científicos. Permita que el escritor pueda centrarse en lo que escribe y no en el formato y el diseño que debe tener el texto, encargándose el de esto. Este será el lenguaje utilizado para la redacción de este documento.

JSON

JSON que son las siglas para JavaScript Object Notation es un lenguaje de formato para el intercambio de datos[JSO]. Esta basado en un conjunto del lenguaje de programación JavaScript, y se compone principalmente de dos estructuras de alto nivel, una tupla de nombre valor, y una lista de atributos. Dentro de los atributos que se pueden encontrar en el valor de las tuplas encontramos los datos primitivos característicos de todo lenguaje de programación como cadenas, números, y valores booleanos, así como el elemento nulo. Este será el formato que se usará principalmente para la transmisión y comunicación de datos a través de la

red.

Android Studio

Como se menciono en la sección 1, se pretende desarrollar una aplicación para dispositivos Smartphones con SO Android, y por tanto es recomendable disponer de un IDE que nos ayude en esta tarea, y uno de estos entornos es Android Studio [And]. Pese a que existen otros entornos de desarrollo que nos pueden ayudar también para este propósito como Eclipse utilizando el plugin ADT [Ecl], se ha decidido utilizar Android Studio debido a ser el IDE oficial para el desarrollo de aplicaciones Android y por tanto proporcionar la seguridad de que siempre estará actualizado y contendrá todo lo necesario para poder desarrollar estas aplicaciones, y además por la gran lista de funcionalidades y facilidades que aporta. Algunas de estas utilidades son su integración con Maven para la gestion de dependencias, la automatización de la compilación y el despliegue de las aplicaciones, un editor de código inteligente y auto completado, emulador integrado, herramientas que ayudan al depurado y al testing, y muchas otras opciones como la integración con sistemas de control de versiones.

Figura 4.3: Android Studio logo

Google Cloud Platform

Google Cloud Platform o GCP es una plataforma de computación en la nube [AFG+10] que ofrece diversos servicios de Infrastructure as a Service, Platform as a Service y Software as a Service. Estos servicios ofrecen características de escalabilidad, control automático y gestión de los recursos, herramientas de monitorización y sobre todo la simulación o virtualización de recursos hardware sin la necesidad de adquirirlos. Esta última característica es una de las principales para que este tipo de tecnología haya sido seleccionada. Esto permite que no sean necesarias tareas de mantenimiento del hardware, debido a que no habrá necesidad alguna de adquirir hardware específico para el servidor, que no haya que preocuparse por el consumo de energía, o de otras tareas de mantenimiento.Tampoco será necesario preocuparse por la infrautilización de recursos, o la necesidad repentina de tener que aumentar la potencia de procesamiento o almacenamiento según la demanda, ya que esta herramienta permite adaptarse con bastante facilidad a estos eventos. Estos servicios se conocen como IAAS.

Además de estos servicios de virtualización de recursos, Google Cloud Platform también

proporciona como se ha mencionado anteriormente servicios de PaaS. Entre esos servicios se encuentran herramientas tanto para el desarrollo como el despliegue o el mantenimiento del software que se ejecutará en el servidor, además de la posibilidad de monitorizar tanto el tráfico como los datos que se encuentren almacenados en este. Es decir pone gran cantidad de recursos al alcance del desarrollador y el equipo en general sin que este necesite tener que implementar e integrar por si mismo todos estos servicios. Ofrece también mecanismos para acceder a los servicios que se encuentren desplegados bajo los servidores.

Además de todo lo mencionado anteriormente, GCP ofrece la posibilidad de utilizar herramientas software ya desarrolladas de manera que no sea necesario invertir tiempo en desarrollar utilidades que ya existen y que se pueden utilizar mediante el alquiler de ese servicio. Estos servicios son conocidos como SaaS, y nos permite utilizar software de terceros para nuestras necesidades.

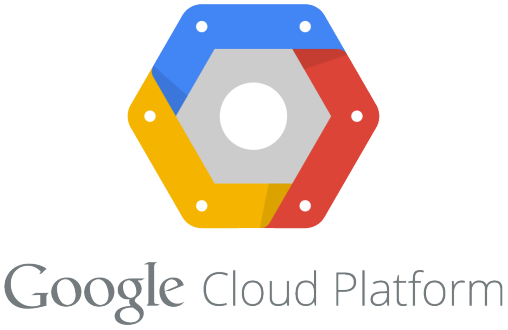

Figura 4.4: Google Cloud Platform logo

De la gran cantidad de recursos que ofrece GCP se explicarán los siguientes por ser los más relevantes en el desarrollo de este TFG. A pesar de eso se han usado otras herramientas y características que proporciona esta herramienta como los registros o el panel de control de Apis.

Datastore

Google Cloud Datastore es una base de datos NoSQL creada para escalar de manera automática, un alto rendimiento y facilitar el desarrollo de aplicaciones [Dat]. Los datos que se encuentran en el Datastore son accedidos mediante las denominadas *transacciones*, que cumplen las características ACID de atomicidad, consistencia, aislamiento y persistencia. Ofrece consistencia fuerte cuando se trata de operaciones que se realizan mediante claves y ancestros y consistencia eventual en el resto de los casos. Estas transacciones también ofrecen toda las operaciones CRUD de crear, leer, actualizar y eliminar. Google Datastore proporciona numerosos mecanismos para realizar estas transacciones, desde una librería JSON, clientes de código abierto o herramientas mantenidas por los usuarios como NDB o Objectify. Además

de las características ACID, ofrece tabmien:

- Disponibilidad: Usa replicación para minimizar el impacto de que uno de los puntos en los que se encuentre falle.
- Escalabilidad: Usa una arquitectura distribuida que maneja de manera automática el escalado.
- Rendimiento: A la vez que es escalable es capaz de mantener un alto rendimiento gracias a su combinación de índices. En una consulta, su rendimiento depende del resultado de la consulta y no del tamaño total del conjunto de datos.
- Almacenamiento y consultas flexibles: Es capaz de trasladar objetos y entidades propias de los lenguajes orientados a objetos y de script a entidades de manera muy natural. También provee de un lenguaje similar a SQL para realizar consultas.
- Encriptación: Se encriptan los datos previo a su almacenado y se desencriptan antes de leídas por los usuarios.
- Administración automática: Google se encarga de todo el mantenimiento.

Como se ha mencionado anteriormente Cloud Datastore es una base de datos NoSQL y que no se corresponde con el sistema relacional de gestión de base de datos. Existen algunas diferencias ya en los conceptos que estos tratan como se puede apreciar en la tabla 4.2. A diferencia de las filas de una misma tabla en los sistemas relacionales, una entidad de Google Datastore puede tener diferentes campos respecto a otra entidad del mismo tipo, o pueden tener propiedades con el mismo nombre y tipos diferentes.

Otras diferencias notables con las bases de datos relacionales son su capacidad de escalado automática, capacidad de balanceo y distribución de datos para mejorar el rendimiento, las únicas consultas que permite son aquellas que permiten que su rendimiento solo dependa del tamaño del resultado proporcionado por la consulta, siendo independiente el número de entidades que existan en total. Esta última es una de las razones por las que algunas operaciones no existen en este sistema.

También es importante destacar que soporta multitud de tipos para sus propiedades, pudiendo incluso contener propiedades estructuradas compuestas por entidades de otros tipos.

Google Datastore se utilizará para almacenar los datos relativos a los usuarios que resulten de interés para la aplicación así como información referente a las imágenes para posteriormente poder ser utilizada en el sistema de recomendación.

Blobstore

Blobstore es un servicio proporcionado por Google Cloud Platform y que accedido mediante una API disponible en los lenguajes Python, Java y Go permite almacenar ficheros de

Identificador	Como	Quiero/Necesito	Estimación	Situación
HU1	Desarrollador	Familiarizarme con la tecnología Android		To Do
HU2	Desarrollador	Familiarizarme con algunos servicios de Google Cloud Platform		To Do
HU3	Usuario	Acceder a la aplicación		To Do
HU4	Usuario	Conocer cual de mis contactos utiliza la aplicación		To Do
HU5	Usuario	Comunicarme con otros usuarios		To Do
HU6	Usuario	Poder enviar imágenes a otros usuarios		To Do
HU7	Usuario	Poder filtrar imágenes mediante palabras clave		To Do
HU8	Usuario	Que la búsqueda no quede limitada a la palabra escrita, sino a su semántica		To Do
HU9	Usuario	Poder seleccionar entre las imágenes que mas se adapten a mis preferencias		To Do

Cuadro 4.1: Primera versión del Product Backlog

Concepto	Google Datastore	RDBMS
Categoría del objeto	Kind/Tipo	Table/Tabla
Objeto único	Entity/Entidad	Row/Fila
Identificador único de objeto	Key/Clave	Primary Key/Clave Primaria
Dato individual sobre un objeto	Property/Propiedad	Field/Campo

Cuadro 4.2: Diferencia entre Google Datastore y los RDBMS

un tamaño superior al permitido por el servicio de Google Datastore [Blo]. Los ficheros se suben en forma de blobs, los cuales se crean indirectamente. En primer lugar es necesario solicitar mediante una llamada a la API la generación de una URL, posteriormente la aplicación cliente podrá proceder a la subida del archivo mediante un formulario web o algún otro tipo de petición HTTP POST. Tras enviar el formulario con el fichero, el servicio Blobstore almacena el archivo y genera una clave para poder recuperar posteriormente el archivo almacenado en el blob. Tras la creación del blob este puede ser modificado o eliminado, y mantiene un registro con metadatos relativos a su creación, tipo y modificaciones.

Este servicio se usará en la aplicación para almacenar las imágenes que los usuarios envíen unos a otros y para almacenar las imágenes propias de las que se dispondrá antes de que la aplicación comience a ser utilizada por los usuarios.

Vision

Google Cloud Vision es una API que permite analizar imágenes [Vis]. Funciona a partir de modelos de Machine Learning (ML) mediante una API REST. Ofrece diferentes posibilidades en el análisis de dichas imágenes, entre ellas estan la clasificación de imágenes entre categorías, aportando también una estimación de la posibilidad de acierto, detecta objetos individuales y caras en las imágenes, pudiendo indicar por ejemplo el número de ciertos objetos o los sentimientos que reflejan dichas caras. Otra de sus funcionalidades es la de detectar texto que se encuentre de una manera u otra en las imágenes.

Para el desarrollo de este Trabajo Fin de Grado (TFG) interesa especialmente la funcionalidad de detectar la categoría con la que se corresponde la imagen para de esta forma poder extraer información de las imágenes seleccionadas por los usuarios y poder utilizar esta información extra en el sistema de recomendación.

Google Cloud Messaging

Google Cloud Messaging (GCM) es un servicio proporcionado por Google que permite enviar datos desde un servidor a diferentes dispositivos y que además permite a estos dispositivos comunicarse entre ellos mediante mensajes. GCM proporciona la funcionalidad de gestionar todos los aspectos relacionados con el encolamiento y la entrega. Además a diferencia de los servicios anteriores, este es completamente gratuito sin importar la densidad del tráfico o la cantidad de usuarios. Además, permite la comunicación entre dispositivos de diferentes plataformas como Android, iOS y Chrome. Tiene la restricción de no permitir envíos superiores a los 4KB en los mensajes, por lo que es necesario recurrir a otros servicios como el Blobstore para enviar las imágenes.

Esta servicio se utilizará en este proyecto para gestionar el envío y recepción de mensajes entre los diferentes usuarios. Esta herramienta es muy útil, ya que se encarga de gestionar todo lo relativo con la distribución de los mensajes, como gestionar el hecho de que el dis-

positivo para el que esta destinado el mensaje no este disponible y no sea posible su envío en ese mismo momento. En este caso, este servicio se encargará de seguir intentando comunicarse con el dispositivo destinatario del mensaje de manera independiente al usuario. Además, ofrece también posibilidades de comunicación en grupos si en un futuro decidiese añadir dicha funcionalidad.

Google Custom Search Engine

Google Custom Search [GCS] permite crear un buscador personalizado para realizar búsquedas sobre una página web o un conjunto de estas. Además, permite grandes opciones de configuración entre las que destacan para la realización de este TFG la de poder filtrar de manera que solo se recuperen imágenes. Esta opción permitirá poder ampliar el abanico de posibles imágenes a utilizar a no solo las imágenes disponibles previo a comenzar a utilizar la aplicación, si no a imágenes de terceros que aumentarán el espectro de opciones disponibles y mejorarán la aplicación. Además proporcionan información y metadatos que resultan de utilidad a la hora de aplicar los sistemas de recomendación.

GitHub

GitHub [Gitb] es una herramienta web para la gestión de repositorios basada en Git[Gita]. Permite almacenar y gestionar código, además de llevar un control de versiones que permiten poder consultar los cambios que han ocurrido durante el desarrollo del proyecto. También facilita la tarea del trabajo en equipo, permitiendo el desarrollo en paralelo y de manera distribuida por varios miembros de un equipo de un mismo trabajo. Proporciona una interfaz web que facilita el uso por parte del equipo de desarrollo del sistema de control de versiones Git, además de añadir algunas otras funcionalidades propias.

Es especialmente útil para este proyecto su funcionalidad de control de versiones, así como para cualquier otro proyecto software que se componga de una complejidad notable. Es especialmente útil en un proyecto que se desarrolla de manera incremental, ya que siempre es útil tener conocimiento de los cambios que se han ido realizando.

TexMaker

Texmaker es una herramienta de edición de texto pensada para trabajar con LaTeX. Ofrece algunas características muy útiles para trabajar con el lenguaje LaTeX como herramientas de corrección ortográfica, autocompletado en el código, y otras herramientas como asistentes para la elaboración de tablas o automatizadores de la compilación.

ANEXOS

Anexo A

Ejemplo de anexo

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Referencias

[AFG+10] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, 2010.

[All90] Robert B Allen. User models: theory, method, and practice. *International Journal of man-machine Studies*, 32(5):511–543, 1990.

[And] Android Studio. `https://developer.android.com/studio/index.html`. Retrieved June 12, 2016.

[BEL+07] James Bennett, Charles Elkan, Bing Liu, Padhraic Smyth, y Domonkos Tikk. KDD Cup and Workshop 2007. *SIGKDD Explor. Newsl.*, 9(2):51–52, Diciembre 2007. url: `http://doi.acm.org/10.1145/1345448.1345459`.

[Blo] Google Blobstore. `https://cloud.google.com/appengine/docs/python/blobstore/`. Retrieved June 13, 2016.

[Bur02] Robin Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Noviembre 2002. url: `http://dx.doi.org/10.1023/A:1021240730564`.

[Dat] Google Cloud Datastore. `https://cloud.google.com/datastore/docs/`. Retrieved June 12, 2016.

[Ecl] Eclipse ADT. `https://marketplace.eclipse.org/content/android-development-tools-eclipse`. Retrieved June 12, 2016.

[GCS] Google Blobstore. `https://developers.google.com/custom-search/`. Retrieved June 13, 2016.

[Gita] Git. `https://git-scm.com/`. Retrieved June 13, 2016.

[Gitb] GitHub. `https://github.com/`. Retrieved June 13, 2016.

[GNOT92] David Goldberg, David Nichols, Brian M. Oki, y Douglas Terry. Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM*, 35(12):61–70, Diciembre 1992. url: http://doi.acm.org/10.1145/138859.138867.

[HK70] E. M. Housman y E. D. Kaskela. State of the Art in Selective Dissemination of Information. *IEEE Transactions on Engineering Writing and Speech*, 13(2):78–83, Sept 1970.

[IIS04] Sylvia Ilieva, Penko Ivanov, y Eliza Stefanova. Analyses of an agile methodology implementation. En *Euromicro Conference, 2004. Proceedings. 30th*, páginas 326–333. IEEE, 2004.

[JSO] JSON. http://www.json.org/. Retrieved June 13, 2016.

[Lat] LaTeX. https://www.latex-project.org/. Retrieved June 13, 2016.

[LMY+12] Linyuan Lü, Matúš Medo, Chi Ho Yeung, Yi-Cheng Zhang, Zi-Ke Zhang, y Tao Zhou. Recommender systems. *Physics Reports*, 519(1):1 – 49, 2012. Recommender Systems. url: http://www.sciencedirect.com/science/article/pii/S0370157312000828.

[MGT86] T. W. Malone, K. R. Grant, y F. A. Turbak. The Information Lens: An Intelligent System for Information Sharing in Organizations. *SIGCHI Bull.*, 17(4):1–8, Abril 1986. url: http://doi.acm.org/10.1145/22339.22340.

[RV97] Paul Resnick y Hal R. Varian. Recommender Systems. *Commun. ACM*, 40(3):56–58, Marzo 1997. url: http://doi.acm.org/10.1145/245108.245121.

[Sch04] Ken Schwaber. *Agile project management with Scrum*. Microsoft press, 2004.

[SS16] Ken Schwaber y Jeff Sutherland. The Scrum Guide. http://www.scrumguides.org/scrum-guide.html, July 2016. Retrieved Agosto 03, 2016.

[Vis] Google Blobstore. https://cloud.google.com/vision/. Retrieved June 13, 2016.

Este documento fue editado y tipografiado con LaTeX empleando
la clase **esi-tfg** (versión 0.20160805) que se puede encontrar en:
`https://bitbucket.org/arco_group/esi-tfg`