Universidad de Jaén

Escuela Politécnica Superior (Jaén)

Proyecto Fin de Carrera:

Sistema de Recomendación Actualizable y con Gestión de usuarios

SIREACGU

Alumno:

Francisco Jesús Martínez Mimbrera

Tutores:

Dr.D.Luís Martínez López

Dª.Macarena Espinilla Estévez

Departamento: Informática

Área de Conocimiento: Lenguajes y Sistemas Informáticos

11 de septiembre de 2008

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Dr. D. Luís Martínez López y D^{a}. Macarena Espinilla Estévez, tutores del Proyecto Fin de Carrera titulado: Sistema de Recomendación Actualizable y con Gestión de usuarios.SIREACGU que presenta D. Francisco Jesús Martínez Mimbrera, autorizan su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, Septiembre 2008

El alumno: Los Tutores:

D. Francisco J. Martínez Mimbrera Dr . D. Luís Martínez López

- D^{a} Macarena Espinilla Estévez

Agradecimientos

Mi mayor agradecimiento, como no podía ser de otra manera, a mis padres por sus incansable esfuerzo y apoyo durante estos años.

Como no, reconocer la admirable labor realizada por mis directores de proyecto Luís y Macarena, que a pesar de todas las dificultades por la distancia (Irlanda esta mu lejos) hemos conseguido un buen trabajo.

A todos mis familiares y amigos por apoyarme en todo esto.

GRACIAS A TODOS.

Índice general

Capítulo 1

PREFACIO

1.1. Introducción al Proyecto

El número de empresas que ofrecen sus productos o servicios en Internet se multiplica cada año. Este hecho indiscutible ha hecho que se produzca una tremenda competencia cada vez más encarnizada para asegurar su supervivencia. Debido a esta competencia, las empresas deben ofertar una serie de servicios diferenciadores que les permitan no sólo mantener su clientela sino atraer nuevos clientes desde los competidores.

Una de las estrategias más interesantes para conseguir esto es la de desarrollar servicios de marketing personalizado basados en sistemas de recomendación. Estos sistemas se encargan de suministrar a los usuarios información personalizada y diferenciada sobre determinados productos y/o servicios que pueden ser de interés para ellos. Es decir, se encargan de guiar a un usuario mediante recomendaciones en la búsqueda de aquellos servicios o productos que puedan ser más atractivos para él, modificando el proceso de navegación y búsqueda. Esto es sin duda una gran ventaja para los clientes, que encontrarán lo que necesitan de una forma más rápida, cómoda y fácil dentro de las enormes bases de datos que ofertan las tiendas electrónicas en Internet y además descubrirán nuevos productos o servicios que le puedan ser atractivos y que de otra manera les hubiese sido mucho más difícil y complejo encontrar.

Existen diversos tipos de sistemas de recomendación, siendo los dos más importantes

y utilizados los siguientes:

1. Sistemas de Recomendación Basados en Contenido

   Se basan en la similitud entre objetos, es decir, predicen que para un usuario serán de interés aquellos objetos muy parecidos en su contenido con aquellos que ya sabemos que son de su agrado.

2. Sistemas de Recomendación Colaborativos

   Son más cercanos a la forma de pensar de los seres humanos que los basados en contenido. Son aquellos en los que las recomendaciones se realizan basándose solamente en los términos de similitud entre los usuarios.

El objeto de este proyecto es el estudio y desarrollo de un Sistema de Recomendación Colaborativo. Se podría decir que el funcionamiento de este tipo de sistemas de recomendación tiene los siguientes pasos:

1. El sistema guarda un perfil de cada usuario con sus evaluaciones sobre objetos conocidos por él y que pertenezcan a la base de datos sobre la que se trabaje.

2. Se mide el grado de similitud entre los diferentes usuarios del sistema en base a sus perfiles y se crean grupos de usuarios con características afines.

3. El sistema utilizará toda la información obtenida en los pasos anteriores para realizar las recomendaciones. A cada usuario le recomendará objetos que no haya evaluado y que hayan sido evaluados de manera positiva por los miembros del grupo en el que este incluido.

Por lo tanto, este sistema no toma en consideración el contenido y las características de los productos que se recomiendan sino que sean del gusto de usuarios con gustos semejantes al usuario que solicita la recomendación.

Una vez comentados brevemente la situación actual del mercado de Internet y las características de los novedosos y útiles sistemas de recomendación podemos pasar a introducir brevemente en qué consiste el proyecto que se documenta en esta memoria.

En este proyecto se pretende realizar un sistema de recomendación con filtrado colaborativo que se aplicará al ámbito de la recomendación en tiendas de alquiler o venta de películas o videoclubs. Para ello utilizaremos los datos que proporciona la web http://www.grouplens.org como base del sistema e iremos añadiendo nuevas películas obtenidas a través de la página http://www.imdb.com

Estos datos están en formato texto por lo que nuestro primer paso será formatearlos hasta convertirlos a un formato que nos sea de utilidad. Nosotros hemos elegido un formato de base de datos MYSQL.

Una vez realizado el formateo de la base de datos, pasamos a elegir el mejor algoritmo para nuestro sistema de recomendación. Para ello nos basaremos en el trabajo realizado por Fernando Siles Fernández[1] , donde estudia el mejor algoritmo para implementar este sistema. Concluida dicha fase, nos dispondremos a imlementar el modulo del servidor encargado de obtener los datos automáticamente de imdb y calculará el algoritmo knn cada vez que se incorporen nuevas películas. El proceso de cálculo del algoritmo knn se realizará de manera offline, para no afectar a las recomendaciones de los usuarios. La parte cliente consistirá en una interfaz web para poner a disposición del usuario la posibilidad de registrarse, puntuar películas, buscar películas, alquilar y obtener recomendaciones.

Por último se expondrán una serie de conclusiones tanto sobre los sistemas de recomendación en general como sobre el desarrollado para este proyecto, además adjuntaremos el manual de usuario y de instalación del servidor. Finalmente, para aquellos lectores que estén interesados en el tema, se incluirá una amplia bibliografía.

1.2. Propósito del Proyecto

El propósito de este proyecto es la implementación de un sistema de recomendación colaborativo basado en técnicas de inteligencia artificial para la gestión del alquiler de películas, que utilice el algoritmo que muestre un mejor comportamiento. El sistema deberá ser capaz de actualizar su base de datos de películas automáticamente cada x

tiempo . Cada vez que se actualize la base de datos de películas se procederá al cálculo del algortimmo para poder ofrecer nuevas recomendaciones a los usuarios.

1.3. Objetivos del Proyecto

1. Búsqueda y revisión bibliográfica.

2. Formatear la base de datos de películas,usuario y puntuaciones disponible en http://www.grouplens.org como base de nuestro sistema de recomendación, la cual posteriormente será aumentada con nuevas películas obtenidas de la página http://www.imdb.com y nuevas puntuaciones de usuarios.

3. Implementación de un algoritmo de filtrado colaborativo para encontrar grupos de usuarios con gustos similares previamente estudiado.

4. Desarrollo de un proceso de extracción de conocimiento que permita predecir las valoraciones de un usuario con arreglo a las hechas por los miembros más similares al mismo.

5. Implementación de un módulo para recolección automática de nuevas peliculas, para actualizar el sistema de recomendación.

6. Integración en un sistema de recomendación de los resultados obtenidos en los puntos anteriores con una arquitectura cliente/servidor y una interfaz web amigable que permita al usuario interactuar con facilidad con el sistema.

Capítulo 2

ANTEDECENTES

En este capítulo nos disponemos a hacer una breve introducción a lo que son los sistema de recomendación y sus distintos tipos. Nos vamos a centrar en los sistema de recomendación colaborativos por ser lo que pretendemos implementar en este proyecto. Por último veremos algunos ejemplos reales de sistema de recomendación que existen en la actualidad.

2.1. Sistemas de Recomendación

El ser humano necesita información para tomar decisiones de cualquier tipo pero muchas veces se encuentra con que la información que tiene disponible es demasiado amplia o inconexa, existiendo una sobrecarga de información con lo que es dificil de extraer la información verdaderamente relevante. Es conveniente filtrar esa información diseminada en grandes volúmenes para que los usuarios la encuentren relevante, es en esta situación donde entran con fuerza al rescate del usuario los sistemas de recomendación.

Una definición formal de sistema de recomendación es, aquel sistema que tiene como principal tarea seleccionar ciertos objetos de acuerdo a los requerimientos del usuario.[5]

Otra definición de un sistema de recomendación podría ser la del sistema que utiliza las opiniones de los usuarios de una comunidad para ayudar a usuarios de esa comunidad a encontrar contenidos de su gusto entre un conjunto sobrecargado de posibles

elecciones.[5]

En realidad los sistemas de recomendación no se basan en ningún modelo novedoso sino que, lo hacen en un acto que existe desde que el ser humano tiene conciencia e inteligencia: "pedir consejo o recomendación a expertos en la materia o seguir a aquellos individuos que tienen gustos similares al del usuario, o bien seleccionar objetos que tienen características similares a objetos que le hayan gustado anteriormente o que se parecen al que inicialmente buscaba". Estas dos tendencias milenarias han dado lugar a las dos ramas principales (aunque no las únicas) de los sistemas de recomendación: los colaborativos y los basados en contenido, respectivamente.

Tanto los sistemas de recomendación colaborativos como los basados en contenido necesitan una enorme cantidad de información sobre usuarios y objetos para poder realizar unas recomendaciones de calidad.[10] Debido a esta circunstancia han surgido otros tipos de sistemas de recomendación que pueden trabajar y ofrecer recomendaciones de calidad sin necesitar una cantidad de información tan grande: los sistemas de recomendación basados en conocimiento. Además, en los últimos tiempos se han desarrollado sistemas de recomendación híbridos, los cuales recogen los mejores aspectos de dos o más de los modelos de recomendación anteriores para conseguir mejores resultados que los obtenidos de forma individual.

2.1.1. Esquema Básico de Funcionamiento

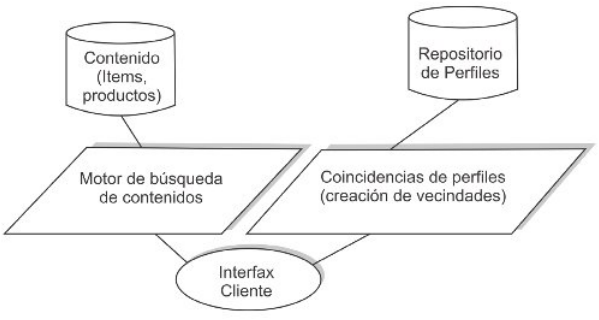


Figura 2.1: Esquema Básico de funcionamiento de un SR

Base de Datos

La calidad de los datos que almacenos en nuestra base de datos jugará un papel fundamental a la hora de poder realizar mejores o peores recomendaciones.

Perfiles

Un usuario se irá creando su perfil conforme más utilice el sistema. El el perfil se irán indicando gustos/preferencias del usuario a la hora de discriminar objetos.

Predicción

Dentro de un esquema básico de un SR la predicción juega un papel fundamental. Esta se apoya en la base de datos con la que contemos y el perfil que nos hallamos ido creando de usuario.

Utilidad de los Sistemas de Recomendación

Los sistemas de recomendación resultan de vital importancia para el marketing personalizado ya que reducen el tiempo de búsqueda de los productos, consiguen una mayor efectividad en las búsquedas y, por lo tanto, una mayor satisfacción en los clientes.

Para lograr estos objetivos, todos los sistemas de recomendación llevan a cabo dos tareas:

- Predecir: los sistemas de recomendación predicen una serie de objetos, servicios o productos en los que un usuario o cliente particular podría estar interesado.

- Recomendar los N-mejores objetos: los sistemas de recomendación identifican los N objetos en los que el usuario estará más interesado.

2.1.2. Clasificación de los Sistemas de Recomendación

Como hemos indicado los sistemas de recomendación se clasifican atendiendo a su funcionamiento, dando lugar a varios tipos de sistemas de recomendación, que revisamos brevemente a continuación: , para que esto llegara a buen puerto

2.1.2.1. Sistemas de Recomendación Basados en Contenido

Un sistema de recomendación basado en contenido es aquel en el cual las recomendaciones son realizadas basándose solamente en un perfil creado a partir del análisis del contenido de los objetos que el usuario ha evaluado en un pasado.[8]

En otras palabras: extraen características de los objetos y las comparan con el perfil del usuario para predecir las preferencias de los usuarios sobre tales objetos. Lo que se pretende es recomendar objetos similares en su contenido a objetos que ya sabemos que son del agrado del usuario en cuestión, o sea, los que forman parte de su perfil.

El filtrado basado en contenido era el más extendido de los sistemas de recomendación hasta la explosión definitiva del filtrado colaborativo. Debido a que los primeros tienen un claro y, en muchos casos, grave problema como es la sobre-especialización. Esta sobre-especialización se da como consecuencia de reducir las recomendaciones a unos contenidos muy similares sin tener en cuenta la posible arbitrariedad de los gustos e intereses de los usuarios. Otro problema de los SRBC es que de un objeto sólo se puede conocer una información parcial, normalmente textual, mientras que la información contextual, visual o semántica es más difícil de conocer y por lo tanto, pueden perderse conexiones no obvias entre objetos similares.

Se han intentado múltiples soluciones para estos problemas como la incorporación de una cierta aleatoriedad a las búsquedas, la indexación semántica latente (LSI) de la información textual[7] o las medidas de similaridad basadas en ontologías pero, sin duda la mejor solución a tales problemas es que exista una buena retro-alimentación entre el sistema y sus usuarios.

2.1.2.2. Sistemas de Recomendación Colaborativos

Los sistemas de recomendación basados en un filtrado colaborativo son aquellos en los que las recomendaciones se realizan basándose en términos de similitud entre los usuarios[8]. Es decir, recomiendan objetos que son del gusto de otros usuarios de intereses similares.

Para la realización de un buen sistema de recomendación colaborativo (es decir, un

sistema que ofrezca recomendaciones de calidad) es necesario utilizar un buen algoritmo de filtrado colaborativo. Estos algoritmos se pueden encuadrar dentro de dos categorías: los algoritmos basados en memoria o usuario y los basados en modelos o ítem.

Conforme la utilización de estos sistemas se ha ido popularizando se han ido detectando una serie de problemas en un modelo de funcionamiento como son: la escasez, la escalabilidad y el problema del ítem nuevo[6] . Multitud de estudios y experimentos se han llevado a cabo en los últimos tiempos con la intención de minimizar estos problemas.

Este tipo de sistemas de recomendación son el eje sobre el que gira este proyecto por lo que se merecen un estudio más detallado en un epígrafe posterior.

2.1.2.3. Sistemas de Recomendación Basados en Conocimiento

Los sistemas de recomendación basados en conocimiento realizan un razonamiento basado en casos entre las necesidades y preferencias de cada usuario para sugerir recomendaciones.[10]

A diferencia de otros sistemas de recomendación, los basados en conocimiento no dependen de grandes cantidades de información sobre objetos puntuados (basados en contenido) y usuarios particulares (colaborativos) sino que lo único que necesitan es tener un conocimiento general sobre el conjunto de objetos y un conocimiento informal de las necesidades del usuario.

El principal problema de estos sistemas de recomendación es que aunque no requieren mucha información si que requieren un gran esfuerzo humano para realizar las recomendaciones mediante todo tipo de heurísticas de inferencia.

2.1.2.4. Híbridos

Todos los modelos de recomendación vistos hasta ahora tienen sus puntos fuertes y sus talón de Aquiles por lo que, es lógico pensar en intentar maximizar sus bonanzas y minimizar sus puntos débiles mediante la hibridación de dos o más de ellos.

Los sistemas híbridos entre los basados en contenido y los colaborativos guardan las preferencias del usuario y las combinan con los objetos más relevantes para realizar las

recomendaciones .

También existen los sistemas híbridos entre los basados en conocimiento y los colaborativos, los basados en contenido y los basados en conocimiento e incluso entre los colaborativos y las redes sociales.

2.1.3. Datos en los Sistemas de Recomendación

2.1.3.1. Realimentación en Sistemas de Recomendación

Un sistema de recomendación no debe ser una entidad estática sino evolucionar en el tiempo en cuanto a la calidad de sus recomendaciones y pronósticos en base a la experiencia y nueva información adquiridas. Para conseguir este objetivo se utilizan mecanismos de realimentación entre el sistema y los gustos de los usuarios. Existen dos tipos de mecanismos de realimentación: los implícitos y los explícitos.

2.1.3.2. Realimentación implicita

Un mecanismo de realimentación implícito es aquel que proporciona información al sistema de recomendación acerca de los gustos de los usuarios sin que éstos sean conscientes de esta situación. Por lo tanto este tipo de realimentaciones no son directas sino que se realizan mediante diversos tipos de medidas como pueden ser el tiempo de visualización del objeto, el número de veces que el objeto es solicitado, etc.

Esta realimentación implícita tiene el problema de depender en demasía del contexto y de ser excesivamente hipotética (podemos suponer que solicitar la visualización de un objeto muchas veces indica un especial interés por parte del usuario pero no tiene porque ser de esa manera) por lo que no resulta ser la más apropiada para todas las situaciones de recomendación.

2.1.3.3. Realimentación explicita

Un mecanismo de realimentación explícito es aquel basado en la acción directa por parte del usuario para indicar que objetos determinados del sistema son de su interés.

Esta interacción directa se puede realizar mediante votaciones numéricas o, más sencillo aún, que el usuario diga si el objeto es o no de su agrado.

Este mecanismo tampoco se encuentra exento de problemas como pueden ser la voluntariedad del cliente o el tiempo consumido.

2.1.3.4. ¿Datos reales o sintetizados?

Otra elección importante es elegir entre un conjunto de datos reales (recopilados de usuarios reales sobre objetos reales) o un conjunto de datos sintetizados (creados específicamente para el sistema de recomendación, sin ninguna base real). Los datos sintéticos son más fáciles de crear que los reales ya que no hay que realizar encuestas ni otros métodos para conseguirlos del mundo real pero sin embargo es recomendable usar estos últimos y solo utilizar los sintetizados en las primeras fases de desarrollo del sistema, siendo sustituidos por los reales cuando haya un número importante de estos recopilados. En nuestro sistema, seguiremos esta idea de sustituir los datos sintetizados iniciales por datos reales.

2.1.3.5. ¿Análisis online u offline?

Es importante decidir como se va a trabajar sobre los datos: si de manera online u offline. En el análisis offline se usa una técnica o algoritmo para predecir ciertos valores retenidos de un conjunto de datos y los resultados son analizados mediante una o varias métricas de error. Este análisis offline tiene la ventaja de ser rápido y económico pero también tiene dos desventajas importantes: el problema de la escasez de datos y el problema de obtener sólo como resultado la bondad de la predicción.

Por contra, el análisis online permite obtener otros resultados como son la actuación de los usuarios participantes, su satisfacción o su participación. En su defecto es más lento y caro que el análisis offline.

2.1.4. Sistemas de Recomendación en Internet: Ejemplos

En esta sección revisaremos distintos sistemas de recomendación (salvo los colaborativos que se verán en su propio epígrafe) que se pueden encontrar en distintos enfoques en internet.

2.1.4.1. Amazon

Figura 2.2: Portal Amazon en Internet

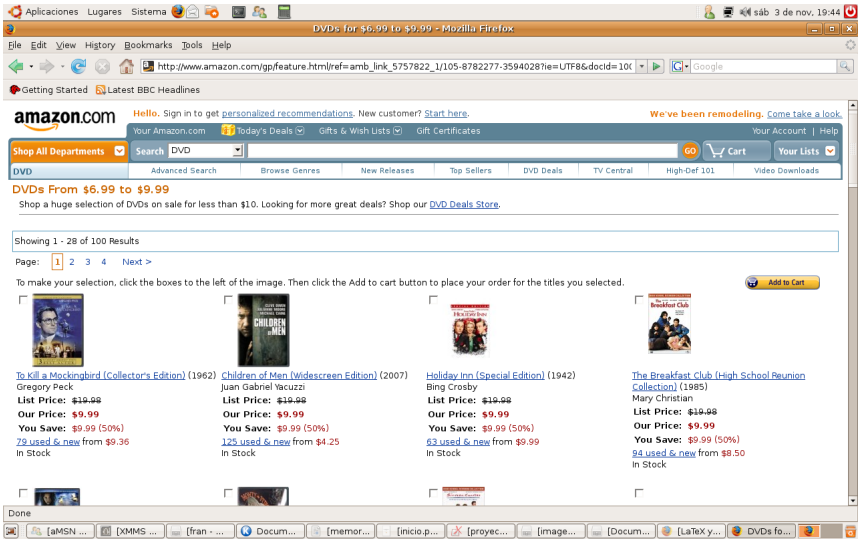


Figura 2.3: Portal Amazon en Internet

La poderosa empresa norteamericana de comercio electrónico que empezó siendo una librería online es un ejemplo paradigmático de sistema de recomendación que mezcla los enfoques basado en contenido y colaborativo. El sistema guarda las preferencias del usuario activo y las combina con objetos relevantes para generar recomendaciones (las ya celebres páginas "People who bought this item... also bought these items..." como la de la figura que acompaña este párrafo).

2.1.4.2. IMDB Recommendation Center

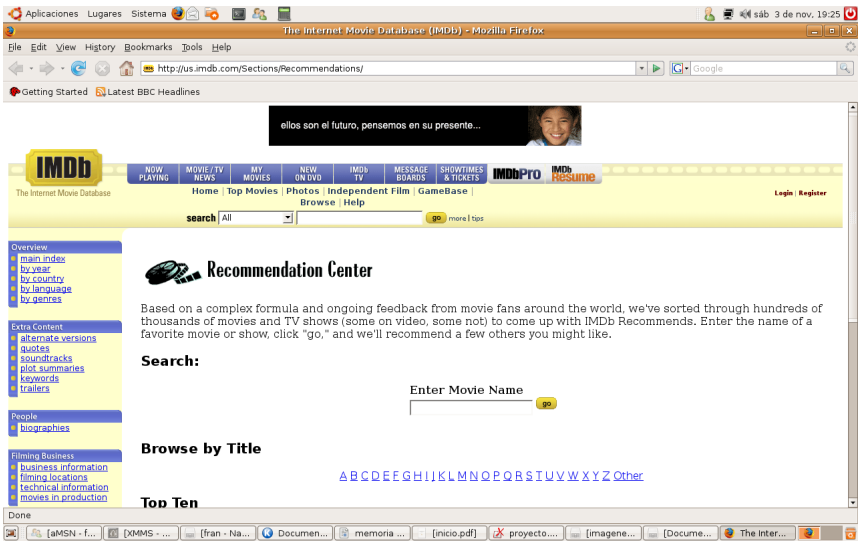


Figura 2.4: nterfaz del Centro de Recomendación de IMDB

La mayor base de datos de cine y televisión del mundo y uno de los sitios web más populares y visitados de todo Internet también ofrece a sus usuarios un sistema de recomendación: se llama Recommendation Center y esta basado en contenido. El usuario introduce la película o show televisivo que más le guste y el sistema le ofrece una lista con diez recomendaciones. Como método de realimentación o feedback con el sistema, el usuario puede señalar las recomendaciones con las que no este de acuerdo y proponer recomendaciones nuevas con lo que el algoritmo se va depurando con la interacción del usuario. De todas formas no es el servicio más exitoso que ofrece IMDB y eso se debe principalmente a que no es un sistema de recomendación especialmente bueno y acertado.

2.1.4.3. Entrée Chicago

Entrée Chicago (Guo; 2006) es un sistema de recomendación basado en conocimiento desarrollado por el Intelligent Information Laboratory (Infolab) de la Universidad de Chicago que ayuda al usuario a decidir entre más de 700 restaurantes de la ciudad de Chicago según sus restaurantes preferidos de otras ciudades norteamericanas. Además

de ofrecer recomendaciones dispone de reviews de todos los restaurantes y de mapas para llegar a ellos.

2.2. Sistemas de Recomendación Colaborativos

Debido a que en este proyecto hacemos uso de un algoritmo de filtrado colaborativo vamos a hacer una revisión más profunda y detallada sobre este tipo de sistemas de recomendación.

2.2.1. Introducción a los Sistemas de Recomendación Colaborativos

Un sistema de recomendación colaborativo es aquel en el que las recomendaciones se realizan basándose solamente en términos de similitud entre los usuarios[8]. Es decir, estos sistemas recomiendan objetos que son del gusto de otros usuarios de intereses similares, en vez de recomendar objetos similares a los que le gustaban en un pasado al usuario activo como sucedía con los basados en contenido. Se podría decir que este tipo de sistemas de recomendación se basan en el concepto del boca a boca entre sus usuarios para realizar sus recomendaciones.

La base teórica de los sistemas de recomendación colaborativos es bastante sencilla: se forman grupos de usuarios más cercanos, es decir, aquellos cuyos perfiles son más parecidos y a un usuario de un grupo se le recomiendan objetos que, él no haya experimentado aún pero que tengan unas puntuaciones positivas por parte del resto de usuarios de ese grupo, es decir, de los más similares a él.

Shardaham y Maes[11] distinguieron tres pasos fundamentales en el funcionamiento de los sistemas de recomendación colaborativos:

1. El sistema guarda un perfil de cada usuario con sus evaluaciones sobre objetos conocidos por él y que pertenecen a la base de datos sobre la que trabajará.

2. Se mide el grado de similitud entre los diferentes usuarios del sistema en base a sus perfiles y se crean grupos de usuarios con características afines.

3. El sistema utiliza toda la información obtenida en los dos pasos anteriores para calcular las predicciones. A cada usuario se le recomendarán objetos que no haya evaluado previamente y que hayan obtenido mayo valor de predicción.

Al igual que los basados en contenido, los sistemas de recomendación colaborativos tienen una serie de problemas que se han detectando conforme se han hecho más populares y utilizados. Dichos problemas son: la escasez, la escalabilidad e ítem nuevo[6]. A continuación veremos una descripcción breve de cada uno de estos problemas:

- Escasez

Los sistemas de recomendación colaborativos necesitan de una gran cantidad de datos, muchos usuarios puntuando muchos ítems similares para así poder calcular los grupos de vecinos y, en base a ellos, realizar las recomendaciones. Si en la base de datos hay pocos usuarios o pocas puntuaciones por parte de cada usuario, nuestra matriz de puntuaciones será muy escasa y los cálculos de vecindad, predicción y recomendación pueden ser realizados sin la suficiente seguridad y exactitud obteniendo recomendaciones de baja calidad.

- Escalabilidad

Los sistemas de recomendación colaborativos usan por norma general *algoritmos de cálculo de los k vecinos más cercanos (knn, K-nearest neighbors)* para obtener la similaridad entre usuarios. Estos algoritmos son costosos computacionalmente y su coste crece linealmente cuanto mayor sea el número de usuarios y de ítems por lo que con bases de datos con millones de elementos, al aumentar el número de datos, el sistema sufrirá graves problemas de eficiencia debido a una falta de escalabilidad.

- Problema del ítem nuevo

En los sistemas de recomendación colaborativos los ítems nuevos, que tienen muy pocas o, incluso, ninguna puntuación no van a ser recomendados prácticamente nunca.

De la misma forma, los nuevos usuarios en el sistema recibirán muy pobres predicciones debido a que ellos han puntuado muy pocos ítems y se hace difícil encuadrarlos en algún grupo de vecinos. Estos dos hechos nos hacen ver que estos sistemas de recomendación requieren un cierto tiempo antes de empezar a hacer predicciones y recomendaciones ciertamente relevantes y acertadas.

Una gran cantidad de experimentos, estudios e investigaciones se han realizado en los últimos años para encontrar técnicas que reduzcan el impacto de estos problemas en los sistemas de recomendación con filtrado colaborativo.

Para reducir el problema de la escasez se han intentado la utilización de puntuaciones implícitas[12], la correlación entre ítems[17] y el filtrado híbrido. Mientras que para tratar de mejorar la escalabilidad se han propuesto la reducción de la dimensionalidad[15] y aproximaciones basadas en modelos. Finalmente, estudios han demostrado que las técnicas de web mining como los árboles de decisión son útiles para paliar el problema de los ítems y usuarios nuevos[13].

2.2.2. Algoritmos de Filtrado Colaborativo

Para realizar un sistema de recomendación colaborativo de calidad es necesario elegir un buen algoritmo de filtrado colaborativo. Existen distintas posibilidades de realizar los algoritmos de filtrado colaborativo. A continuación pasamos a describir brevemente las distintas posibilidades de implementación de este tipo de algoritmos:

- Algoritmos basados en memoria o basados en usuario

Estos algoritmos utilizan la base de datos completa para generar una predicción. El funcionamiento de estos algoritmos es el siguiente: se utilizan técnicas estadísticas para encontrar un conjunto de vecinos al usuario activo y posteriormente se utilizan una serie de algoritmos que combinan las preferencias de esta vecindad para realizar las predicciones y recomendaciones.

Los algoritmos basados en usuario son muy populares y exitosos en la práctica pero son también los que con más ferocidad sufren los problemas de escasez y escalabilidad

vistos anteriormente. Por lo que se hizo necesaria la aparición de otro tipo de algoritmos como los siguientes.

- Algoritmos basados en modelos o basados en ítem

Estos algoritmos proporcionan recomendaciones de ítems desarrollando primero un modelo (ya sea mediante redes bayesianas, clustering o modelos basados en reglas) de las puntuaciones de los usuarios sobre los ítems.

No se utilizan técnicas estadísticas sino una aproximación probabilística que calcula el valor esperado de una predicción del usuario dados sus puntuaciones sobre otros ítems. Es decir, estos algoritmos miran en el conjunto de ítems que el usuario activo ha puntuado o evaluado y calcula la similitud de estas puntuaciones con respecto al ítem activo con el fin de realizar una predicción para el mismo.

Los sistemas de recomendación colaborativos basados en modelo o ítem son los que centrarán el proyecto que se detalla en esta memoria por lo que veremos con más profundidad estos algoritmos y sus principales aspectos cuando entremos a analizar en profundidad el proyecto realizado.

2.2.3. Sistemas de Recomendación Colaborativos en Internet

2.2.3.1. Tapestry

El pionero. Tapestry, un proyecto de Xerox PARC, esta considerado como el primer sistema de recomendación que implementaba filtrado colaborativo. Tapestry permitía a sus usuarios encontrar documentos basados en comentarios hechos previamente por otros usuarios. Al ser un experimento pionero surgieron muchos problemas ya que sólo funcionaba correctamente con pequeños grupos de personas y eran necesarias consultas de palabras específicas para obtener resultados lo que dificultaba en gran medida el propósito último del filtrado colaborativo. También tenía otras carencias como la falta de privacidad. A pesar de todo fue un sistema que resultó crucial para el posterior fulgurante crecimiento de los sistemas de recomendación colaborativos.

2.2.3.2. zagat.com

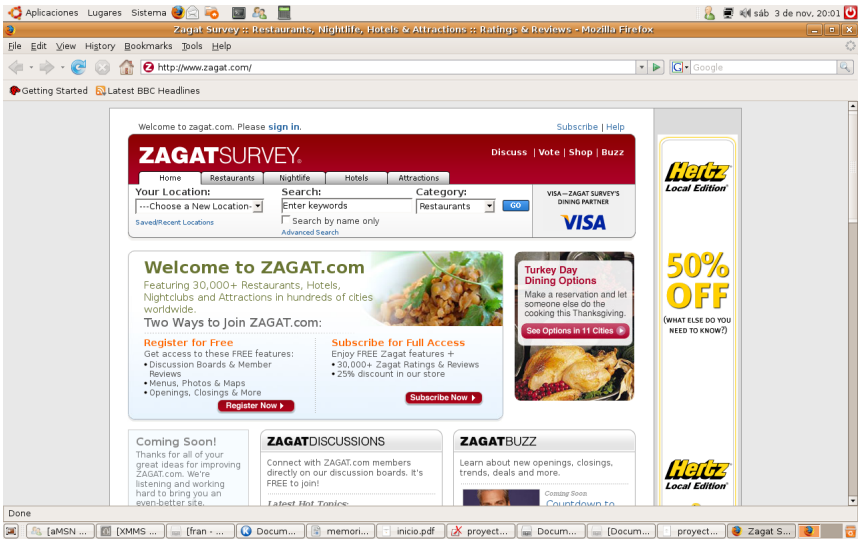


Figura 2.5: Imagen del portal Zagat.com

Zagat Survey es una empresa americana fundada en 1979 que se dedica a la edición de todo tipo de guías de restaurantes, hoteles, clubes o tiendas de distintas ciudades de los Estados Unidos y Canadá. En zagat.com los usuarios registrados pueden votar distintos aspectos (hasta 30) del local referido y, además, introducir pequeños comentarios con su experiencia. En base a estas votaciones los responsables de la empresa asignan sus puntuaciones en sus guías anuales y hacen recomendaciones individuales a sus usuarios a través de su web.

2.2.3.3. MovieLens

Figura 2.6: Imagen del Portal Movilens

MovieLens es un sistema de recomendación de películas online basado en filtrado colaborativo. Desarrollado por el GroupLens Research de la Universidad de Minnesota http://www.grouplens.org recolecta puntuaciones sobre películas de sus usuarios y en base a esos datos agrupa a los usuarios de similares gustos. Atendiendo a las puntuaciones de todos los usuarios dentro de un grupo se intenta predecir para cada usuario individual su opinión sobre películas que todavía no ha visto.

En un principio utilizaba algoritmos basados en usuario para realizar sus predicciones y recomendaciones pero desde hace un tiempo emplea algoritmos basados en ítem porque dan unos mejores resultados.

Los datos sobre sus usuarios y sus puntaciones son privados pero los investigadores de GroupLens mantienen como públicas dos ejemplos de 100000 y un millón de puntaciones respectivamente. Estos ejemplos se pueden descargar desde la propia página web http://www.grouplens.org siendo el de 100000 puntuaciones el usado en este proyecto como conjunto de datos de partida.

Figura 2.7: Imagen del portal GroupsLens de la Universidad de Minesota

Capítulo 3

ESTUDIO ALGORÍTMICO

Una vez presentado el proyecto a grandes rasgos con su propósito y objetivos y realizada la introducción teórica a los sistemas de recomendación en general y a los sistemas de recomendación colaborativos en particular llega el momento de pasar a detallar el desarrollo del proyecto que se ha realizado.

Este proyecto consta de dos partes bien diferenciadas, cada una correspondiente a una de las tareas básicas que deben realizar los sistemas de recomendación para conseguir sus objetivos de marketing personalizado, como son la predicción y recomendación.

1. La primera parte consiste en un estudio comparativo de los distintos tipos de algoritmos de filtrado colaborativo basados en modelo o ítem para la predicción.

2. La segunda parte consiste en realizar un sistema de recomendación basado en una arquitectura cliente/servidor con interfaz web implementando el mejor de los algoritmos previamente estudiados.

3.1. Estudio Comparativo

En este apartado vamos a encargarnos de detallar pormenorizadamente el desarrollo de un estudio comparativo de distintos algoritmos de filtrado colaborativo para elegir el mejor de ellos y, posteriormente, desarrollarlo en la práctica.

En la introducción teórica se presentaban los sistemas de recomendación colaborativos. El primer paso para su realización es formar grupos con los usuarios o ítems de la base de datos más similares entre si. Para formar estos grupos se aplicarán distintas medidas de similaridad y un *algoritmo de clasificación K-nn*.

Una vez creados estos grupos y evaluado el algoritmo K-nn mediante la técnica hold-out se estudiará el comportamiento de distintos algoritmos de predicción. Finalmente se realizarán una serie de pruebas basándose en distintas métricas de evaluación y se compararán sus resultados, para de esta manera obtener el mejor de los algoritmos y pasar a su implementación práctica en capítulo 4 del proyecto.

3.1.1. Notación

Antes de empezar a enumerar la metodología utilizanda en este estudio junto con sus formulas, expresiones y algoritmos utilizados es conveniente dejar clara la notación que se va a emplear para que no se produzcan equívocos entre los lectores:

- Un usuario será aquel elemento representado por $u_i \in U = \{u_1, ..., u_n\}$

- Un ítem será aquel elemento representado por $i_i \in I = \{i_1, ..., i_m\}$

- La similitud entre dos items i_j e i_k se representará como $s\left(i_j, i_k\right)$

- Una evaluación o puntuación de un usuario u_i sobre un ítem i_j se representará como r_{ui}, i_j

- Finalmente, una predicción sobre el ítem i_j del usuario u_i se representará como p_{ui}, i_j

3.1.2. Algoritmo K-nn

Un paso imprescindible para la realización de un sistema de recomendación colaborativo de calidad es la formación de grupos de usuarios (si es un sistema de recomendación colaborativo basado en memoria) o de ítems (si es basado en modelos, como es el caso de este proyecto) de características similares. Esta actividad se puede ver como la primera parte de un **problema de clasificación** de la base de datos y existen multitud de técnicas y algoritmos, llamados clasificadores, que permiten resolverlo. Uno de los clasificadores de mayor difusión y mejores prestaciones es el **algoritmo K-nn**, que es el que se va a utilizar en este proyecto para formar los grupos de ítems más similares para cada uno de los ítems de la base de datos.

El algoritmo K-nn (k nearest neighbors, k vecinos más cercanos) es un tipo de clasificador basado en instancias. Estos clasificadores, que trabajan directamente sobre los datos sin construir ningún tipo de modelo sobre la base de datos, están basados en aprendizaje por analogía encuadrándose dentro del paradigma perezoso del aprendizaje. Este paradigma se caracteriza por:

- El trabajo se retrasa lo máximo posible.

- No se construye ningún modelo, el modelo es la propia base de datos sobre la que se trabaja.

- Se trabaja cuando llega un nuevo objeto a clasificar: se buscan los casos más parecidos y la clasificación se construye en función de la clase a la que dichos casos pertenezcan.

3.1.2.1. Funcionamiento del algoritmo K-nn

Siendo i el objeto a clasificar debemos seleccionar los k objetos con $K = \{i_i, ..., i_k\}$ tal que no existe ningún ejemplo i' fuera de K con $s(i, i') < s(i, i_j), j = 1, ..., k$

Una vez encontrados los k-vecinos más cercanos se puede proceder a la clasificación de dos formas distintas:

- Voto por la mayoría: se clasifica el nuevo objeto en la clase mayoritaria entre los objetos de K.

- Voto compensado según la distancia: se clasifica el objeto según su distancia ponderada con el resto de objetos de k.

3.1.2.2. Descripción algorítmica

Para una mejor comprensión del algoritmo K-NN vamos a escribir una descripción algorítmica simple del mismo:

1. Se separan los datos en dos conjuntos disjuntos: entrenamiento (E) y test (T).
2. Llega un nuevo objeto i_a
3. Se obtienen los k objetos $i_1, ..., i_k$ del conjunto E más cercanos a i_a
4. Se clasifica el objeto i_a de una de las dos maneras siguientes:
   - Voto por la mayoría.
   - Voto ponderado según la distancia.

3.1.2.3. Principales características

- Es un algoritmo robusto frente al ruido cuando el valor de k es moderado ($k > 1$).

- Es bastante eficaz cuando el número de clases posibles es alto.

- Tiene una complejidad temporal de O(dn^2) siendo O(d) la complejidad de la métrica de distancia utilizada.

- El hecho de no utilizar modelos sino la base de datos al completo provoca que sea ineficiente en memoria.

- Es válido para clasificación.

Para este estudio se utilizará un **algoritmo K-nn** para seleccionar los k ítems más similares para cada uno de los ítems que componen nuestra base de datos. Lo que variará de una prueba a otra de nuestro estudio, será la métrica de medida o similitud empleada para calcularlo y el tamaño de los conjuntos de test y entrenamiento utilizados para su evaluación *hold-out*, etc.

3.1.3. Medidas de Similitud

Un elemento crucial para el filtrado colaborativo basado en ítem utilizando el algoritmo k-nn es el cálculo de la similitud entre los distintos ítems y la selección de los k más similares. El concepto de simililud se puede representar de muy diversas formas de entre las cuales hemos elegido para este proyecto la siguiente:

$s\left(x,y\right) : U\text{x}U \rightarrow [0,1]$ midiendo el grado de similitud entre x e y siendo mayor cuanto más cerca de 1 se encuentre.

La similitud cumple con las propiedades:

- Reflexiva: $(s\left(x,y\right) = 1)$

- Simétrica: $(s\left(x,y\right) = s\left(y,x\right))$

Por lo tanto para calcular los ítems más similares a uno dado x, con una medida de similitud entre ítems $s\left(x,y\right) : U\text{x}U \rightarrow [0,1]$, $y \in U - x$, seleccionamos los k ítems y tal que la función sea máxima.

Es conveniente aclarar que, en este proyecto, para calcular la similitud entre dos ítems x e y, sólo se tendrán en cuenta a aquellos usuarios que hayan evaluado a ambos ítems y no siendo tomados en consideración el resto.

Existen multitud de funciones que nos pueden dar una medida de la similitud entre dos elementos pero nosotros hemos elegido para este estudio implementar las tres más estables como son:

3.1.3.1. Coeficiente Coseno

En esta medida se supone que dos ítems x e y son vectores en el espacio y la similaridad entre ellos vendrá dada por el coseno que formen sus ángulos. La expresión para su cálculo es la siguiente:

$$s(x,y) = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} (x_i^2) \sum_{i=1}^{n} (y_i^2)}}$$

Siendo x_i el valor del objeto x para el usuario i, y_i el valor del objeto y para el usuario i y n el número de usuarios que han evaluado tanto x como y

3.1.3.2. Distancia Euclidea

Una forma de calcular la similitud entre dos objetos puede ser calcular la distancia entre los mismos ya que cuanto menor sea esa distancia mayor será la similitud. Una medida de distancia cumple con las siguientes propiedades:

1. $d(x,y) = 0 \leftrightarrow x = y$
2. $d(x,y) = d(y,x)$ (Propiedad simétrica)
3. $d(x,z) \leq d(x,z) + d(y,z)$

Existen múltiples funciones para el cálculo de distancias y una de las más importantes y utilizadas es la distancia euclídea cuya expresión genérica es la siguiente:

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Esta distancia plantea el inconveniente de que el valor resultante no esta restringido a un intervalo entre [0, 1] que es el intervalo en el que hemos comentado que vamos a representar la similitud. Para remediar este inconveniente recurrimos a la normalización

de la fórmula anterior, que quedaría de la siguiente manera:

$$d(x,y) = \frac{\sqrt{\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{A_i^2}}}{\sqrt{n}}$$

Siendo A_i la amplitud del dominio del usuario i, la cual se calcula como: $A_i = b_i - a_i$ donde a y b son los límites superior e inferior respectivamente de tal dominio. Este dominio del usuario, en nuestro proyecto, es el comprendido por el intervalo de enteros entre 1 y 5, ambos inclusive.

La similitud será por tanto: $s(x,y) = 1 - d_n(x,y)$

3.1.3.3. Coeficiente de Correlación de Pearson

Este coeficiente de correlación de Pearson apareció por primera vez en la literatura relacionada con los sistemas de recomendación dentro del contexto del **proyecto GroupLens** donde se empleaba como la base para la asignación de pesos. Este coeficiente es un índice que mide la relación lineal entre dos variables cuantitativas, siendo independiente de la escala de medidas de dichas variables y estando acotado dentro del intervalo [-1, 1]. La expresión que permite calcular este coeficiente es la siguiente:

$$s(x,y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

Siendo $\bar{x}$ e $\bar{y}$ la media de todos los valores de x e y respectivamente.

3.1.4. Algoritmos de Predicción Basados en Ítem

El último pasa para decidir que elemento recomendar al usuario es hacer una predicción sobre los posibles items a recomendar, para ello, una vez calculado el conjunto de los ítems más similares para cada uno de los ítems de la base de datos mediante el algo-

ritmo k-nn y alguna de las medidas de similaridad vistas anteriormente, hay que elegir la técnica o algoritmo más adecuado para realizar la predicción.

No existen algoritmos mejores o peores sino que existen algoritmos que se ajustan mejor o peor al conjunto de datos. Esto se debe a que muchos de los algoritmos de filtrado colaborativo han sido diseñados para un conjunto de datos específico. Si este conjunto de datos específico tiene muchos más usuarios que ítems puede resultar inapropiada su ejecución sobre conjuntos de datos donde se tienen más ítems que usuarios y viceversa.

En este proyecto hemos tomado en consideración dos de los múltiples algoritmos que existen para calcular la predicción para comprobar cual se ajusta mejor a nuestra base de datos:

3.1.4.1. Ítem average + adjustment

Esta técnica presupone que una predicción para un usuario concreto sobre un ítem es igual al valor medio de ese ítem más un ajuste que viene a ser la suma ponderada de las evaluaciones hechas por el usuario y su similaridad con el ítem activo. La expresión para dicha técnica es la siguiente:

$$\mathrm{p}u_a, i_a = \overline{ri_a} + \frac{\sum_{h=1}^{n} s\left(i_a, i_h\right)\left(ru_a, i_h - \overline{ru_a}\right)}{\sum_{h=1}^{n} \left|s\left(i_a, i_h\right)\right|}$$

Siendo u_a el usuario activo, i_a el ítem cuyo valor se quiere predecir y $\overline{ri_a}$ y $\overline{ru_a}$ las puntuaciones medias del usuario y el ítem, respectivamente. Estas medias se calculan de la siguiente manera:

$$\overline{ri_a} = \frac{\sum_{h=1}^{m} ru_h, i_a}{m} \qquad \overline{ru_a} = \frac{\sum_{h=1}^{n} ru_a, i_h}{n}$$

Donde n es el número de ítems que el usuario activo ha puntuado y m es el número

de usuarios que han puntuado el ítem a predecir.

Existen dos enfoques diferenciados para afrontar esta técnica atendiendo al número de valores seleccionados para realizar la predicción:

- Todos menos 1: Se conocen todas las evaluaciones que ha hecho el usuario salvo la que se quiere predecir.

- Dados n: Sólo se conocen n evaluaciones del total que ha hecho el usuario. Esta n suele ser un número potencia de 2.

3.1.4.2. Weighted sum

Este método calcula la predicción de un ítem i por parte del usuario activo u_a como la suma de las evaluaciones del usuario u_a sobre ítems similares a i . Cada una de estas evaluaciones esta ponderada por la correspondiente similitud $s\left(i,j\right)$ entre los ítems i y j. Podemos denotar esta técnica de la siguiente manera:

$$\mathrm{p}(u_a, i_a) = \frac{\sum_{h=1}^{k} s\left(i_a, i_h\right) * ru_a, ih}{\sum_{h=1}^{k} \left|s\left(i_a, i_h\right)\right|}$$

Indicando k los k ítems más similares al ítem i_a.

Básicamente, esta técnica intenta captar como evalúa el usuario activo a ítems similares al que se quiere predecir. Es necesario ponderar estas evaluaciones con la similaridad para asegurarnos de que la predicción entra dentro del rango previamente definido.

3.1.5. Conjunto de Datos

El conjunto de datos utilizado para comenzar a usar el sistema es un ejemplo de la base de datos MovieLens (disponible en http://www.grouplens.org) formado por 943

usuarios, 1682 películas y 100000 puntuaciones habiendo puntuado cada uno de esos 943 usuarios un mínimo de 20 películas y habiendo sido puntuada cada película al menos una vez.

Esta base de datos se encuentra en un formato textual poco manejable y eficiente por lo que la hemos transformado a un formato de base de datos más aconsejable para su tratamiento por parte de un modulo desarrollado en Java a través de JDBC como es MYSQL.

Tablas Creadas:

USUARIOS: una tabla de 943 filas en la que cada una de estas filas esta compuesta por los siguientes campos:

ID_USER: entero. Llave primaria. Identificador numérico y unívoco del usuario.

EDAD: entero. Edad del usuario.

GENERO: cadena de 1 carácter. Género del usuario (M si es hombre, F si es mujer).

EMAIL: cadena de 125 caracteres. Correo Electrónico del usuario.

COD_POSTAL: cadena de 5 caracteres. Código postal del usuario.

NUM_PUNTUACIONES: entero. Campo calculable. Número de películas puntuadas por el usuario.

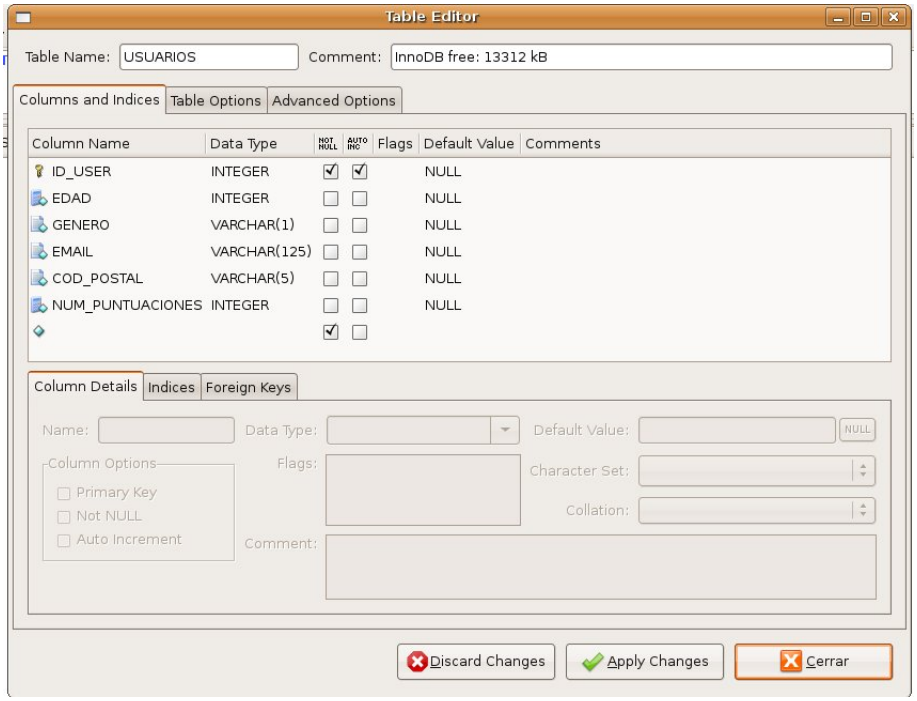


Figura 3.1: Creación de la tabla de usuarios en Mysql

PELICULAS: tabla de 1682 filas con los siguientes campos cada una:

ID_MOVIE: entero. Llave primaria. Identificador númerico y unívoco de la película.

TITULO: cadena de 81 caracteres. Título de la película.

FECHA: fecha. Fecha de estreno de la película.

IMDB_URL: cadena de 134 caracteres. Enlace a la entrada en IMDB de la película.

DESCONOCIDO: booleano. Verdadero si no se conoce el género de la película.

ACCION: booleano. Verdadero si la película es de acción.

AVENTURAS: booelano. Verdadero si la película es de aventuras.

ANIMACION: booleano. Verdadero si la película es de animación.

INFANTIL: booleano. Verdadero si es una película infatil.

COMEDIA: booleano. Verdadero si la película es una comedia.

CRIMEN: booelano. Verdadero si es una película de crimen.

DOCUMENTAL: booleano. Verdadero si la película es un documental.

DRAMA: booleano. Verdadero si la película es un drama.

FANTASIA: booleano. Verdadero si la película es de fantasía.

NEGRO: booleano. Verdadero si la película es de género negro.

TERROR: booleano. Verdadero si la película es de terror.

MUSICAL: booleano. Verdadero si la película es un musical.

MISTERIO: booleano. Verdadero si la película es de misterio.

ROMANTICO: booleano. Verdadero si la película es romántica.

CIENCIA-FICCION: booleano. Verdadero si la película es de ciencia-ficcion.

THRILLER: booleano. Verdadero si la película es un thriller.

GUERRA: booleano. Verdadero si la película es de guerra.

WESTERN: booleano. Verdadero si la película es un western.

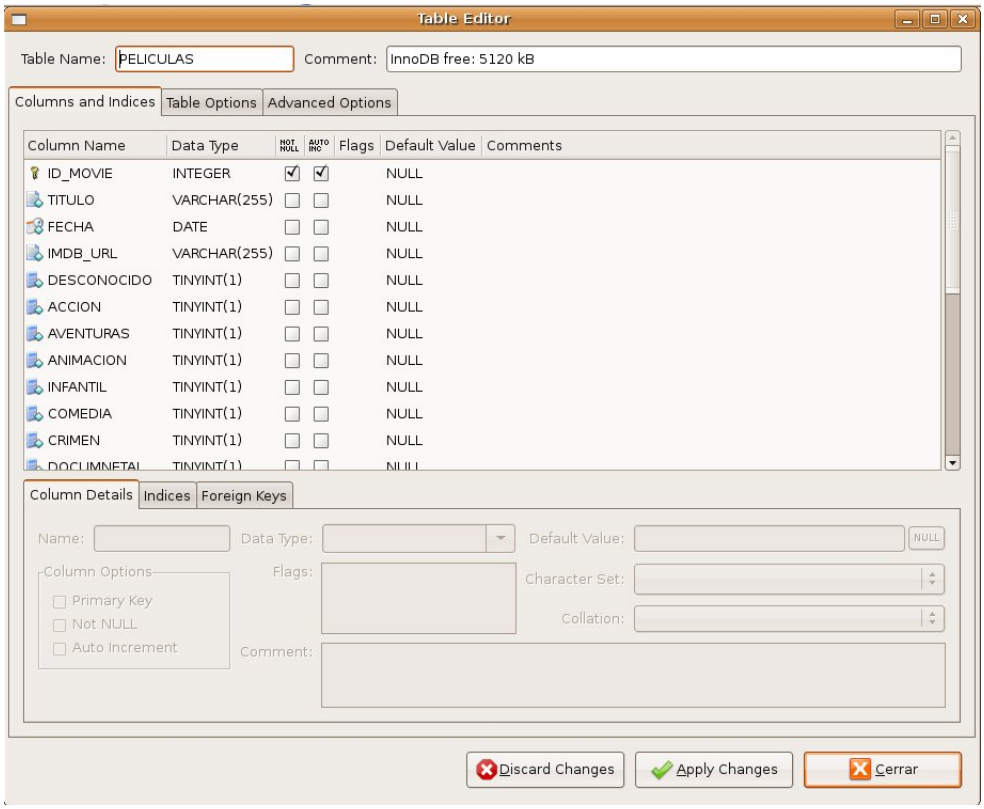


Figura 3.2: Creación de la tabla de peliculas en Mysql

PUNTUACIONES: tabla con 100000 filas con los siguientes campos cada una:

ID_USER: entero. Llave primaria. Llave foránea. Identificador del usuario.

ID_MOVIE: entero. Llave primaria. Llave foránea. Identificador de la película.

RATING: byte. Puntuación (1, 2, 3, 4 o 5) del usuario sobre la película.

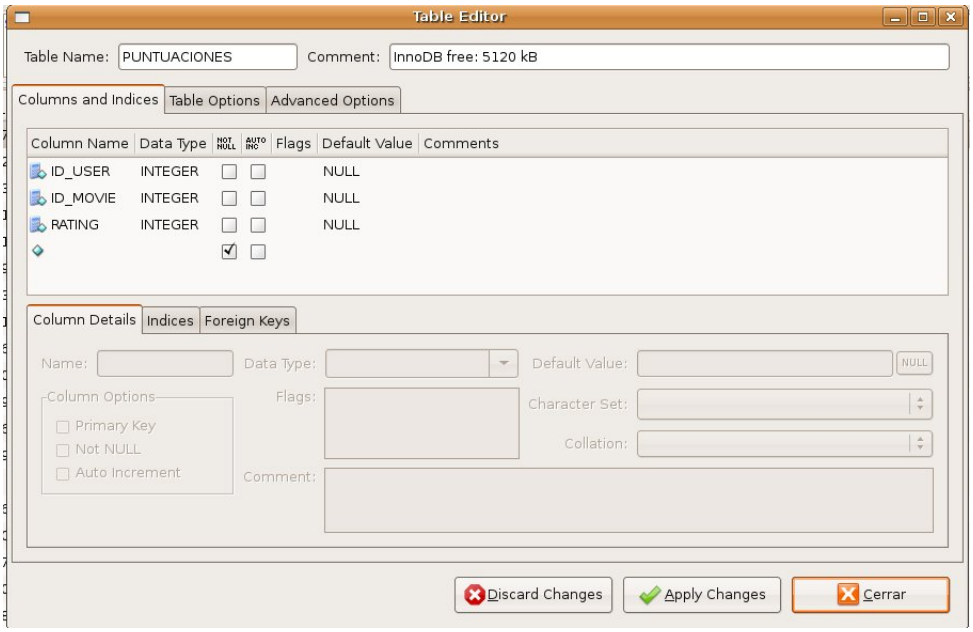


Figura 3.3: Creación de la tabla de puntuaciones en Mysql

Por otra parte, en la base de datos movieranks existe otra tabla auxiliar llamada ALQUILADAS que guarda las películas alquiladas pero todavía no puntuadas de cada usuario. Esta tabla no tiene incidencia en este estudio sino para el posterior desarrollo del sistema de recomendación colaborativo por lo que será alli donde se detalle adecuadamente.

3.1.6. Métricas de evaluación

Para evaluar la bondad de los resultados de las pruebas realizadas existen multitud de métricas. Dentro de estas métricas, un tipo muy importante son las métricas de precisión, de las cuales existen, principalmente dos clases: las métricas de precisión

estadística y las métricas de precisión de apoyo a la decisión.

Métricas de precisión estadística: son aquellas que evalúan la precisión de un sistema de recomendación comparando las predicciones numéricas con las puntuaciones reales para cada ítem que tenga tanto puntuación como predicción. Algunas de estas métricas son el **MAE** (Mean Absolute Error), el **RMSE** (Root Mean Squared Error) o la **correlación**.

Métricas de precisión de apoyo a la decisión: son aquellas que evalúan como de efectivamente las predicciones ayudan a los usuarios a seleccionar ítems adecuados. Algunas de las métricas de esta clase son la puntuación inversa, la sensitividad **ROC** (Receiver Operating Characteristic) o la sensitividad **PRC** (Precision Recall Curve).

Sin duda, elegir unas buenas métricas resulta fundamental. Para este proyecto hemos elegido dos cuyos resultados se pueden complementar bastante bien.

La primera de estas métricas es una métrica de precisión estadística llamada **MAE** (Mean Absolut Error) que es, con diferencia, la métrica de este tipo más utilizada. El MAE es una medida en valor absoluto de la desviación entre las puntuaciones reales (r) y sus predicciones (p) cuya expresión es la siguiente:

$$\text{MAE}= \frac{\sum_{h=1}^{n} |p_h - r_h|}{n}$$

Cuanto menor sea este MAE, que estará acotado por la amplitud del dominio de las puntuaciones, más exactas serán las predicciones permitiendo unas mejores recomendaciones.

La segunda métrica que vamos a utilizar es una métrica temporal ya que de nada nos sirve que el algoritmo de predicción proporcione predicciones muy exactas si el coste en tiempo para ofrecerlas es demasiado alto. Calcularemos el tiempo de ejecución de cada

algoritmo en milisegundos (ms) siendo el mejor de estos algoritmos el que consiga ofrecer unos mejores predicciones en un menor tiempo.

3.1.7. Implementación de Pruebas

Como ya se ha comentado en varias ocasiones esta parte de la memoria esta dedicada al estudio comparativo de distintos algoritmos de filtrado colaborativo mediante la evaluación y análisis de los resultados de diversas pruebas realizadas. Pues bien, en este apartado se va a pasar a detallar el proceso de implementación de dichas pruebas.

El primer paso consiste en dividir el conjunto de datos detallado en el apartado anterior en dos conjuntos disjuntos independientes de usuarios: uno será el denominado conjunto de entrenamiento mientras que el otro, aconsejablemente más pequeño, será el denominado conjunto de test. A este enfoque se le conoce como una técnica de evaluación hold-out y suele emplearse para bases de datos relativamente grandes como la que se utiliza en este estudio.

Sobre el conjunto de entrenamiento se aplicará el algoritmo k-nn para calcular el conjunto de vecinos más similares para cada uno de los ítems. Esta similaridad será calculada mediante uno de las tres medidas de similaridad presentadas anteriormente en esta memoria.

Este conjunto de k-vecinos es indispensable para el cálculo de predicciones, un cálculo que se realizará para los usuarios contenidos en el conjunto de test mediante la utilización de uno de los algoritmos de predicción ya estudiados.

Finalmente obtendremos una medida de la precisión de estas predicciones gracias a la medida de precisión estadística MAE (Mean Absolut Error) y una medida del tiempo empleado para realizarlas.

Estas pruebas no han sido implementadas, sino que presentamos los resultados obtenidos del proyecto “Sistema de Recomendación colaborativo de alquiler de películas”[1] ya que este proyecto no trata de hacer una evaluación del mejor método ya realizada[1] sino de llevar a cabo su implementación en un sistema actualizable y de interfaz gráfica amigable.

3.1.8. Evaluación de resultados

Se realizaron 6 pruebas diferentes, ejecutándose 20 iteraciones de cada una de ellas. Para cada una de estas pruebas se variaron los valores de uno o más de los siguientes parámetros:

Porcentaje Entrenamiento/ Test: indica el porcentaje de la base de datos que se usará como conjunto de entrenamiento en tanto por uno, siendo el conjunto de test el resto de la base de datos.

Número de vecinos: proporciona el valor k del algoritmo k-nn para la construcción del conjunto de k ítems más similares para cada ítem.

Medida de similaridad: indica cual de las tres medidas de similaridad consideradas (distancia euclídea, coeficiente coseno o coeficiente de correlación de Pearson) se ha utilizado.

Algoritmo de predicción: indica cual de los dos algoritmos de predicción considerados se ha utilizado. El algoritmo item+adjustment se identifica en las tablas y gráficos de resultados como Pr1 mientras que el algoritmo weighted sum se identifica como Pr2. Siempre que se utilice el algoritmo item+adjustment se emplearán los dos enfoques vistos previamente: todos menos 1 (identificado como TM1) y dados n con los valores 2, 4 y 8 para dicha n (identificados respectivamente como D2, D4, D8).

Los valores de estos parámetros para cada una de las pruebas son los siguientes:

	PRUEBA1	PRUEBA2	PRUEBA3	PRUEBA4	PRUEBA5	PRUEBA6
ENT/TEST	0.8/0.2	0.8/0.2	0.6/0.4	0.8/0.2	0.8/0.2	0.8/0.2
Nº VEC.	20	20	20	20	10	40
MED. SIMILAR.	Coeficiente Coseno	Coeficiente Coseno	Coeficiente Coseno	Correlación de Pearson	Coeficiente Coseno	Coeficiente Coseno
ALG. PRED.	item+adjust.	weighted sum	item+adjust.	item+adjust.	item+adjust.	item+adjust.

Figura 3.4: Valores de los parámetros de las distintas pruebas

3.1.9. Comparativa entre las Pruebas

Una vez realizadas las pruebas y analizados sus resultados llega el momento de comparar estos resultados y determinar cual prueba los ha obtenido mejores para la posterior implementación de un sistema de recomendación colaborativo con los valores de los parámetros de dicha prueba.

Como en las pruebas que han implementado el algoritmo de predicción item average + adjustment se presentan distintos resultados según el enfoque seguido vamos a recordar cual de estos enfoques es el tomado en consideración (es decir, el que mejores resultados haya obtenido) para cada prueba:

PRUEBA	ENFOQUE
PRUEBA1	Todos Menos 1
PRUEBA3	Todos Menos 1
PRUEBA4	Dados 2
PRUEBA5	Dados 2
PRUEBA6	Todos Menos 1

Figura 3.5: Enfoque elegido por cada prueba

En el siguiente gráfico de barras se muestra con claridad los resultados comparados de todas las pruebas para las métricas MAE y temporal:

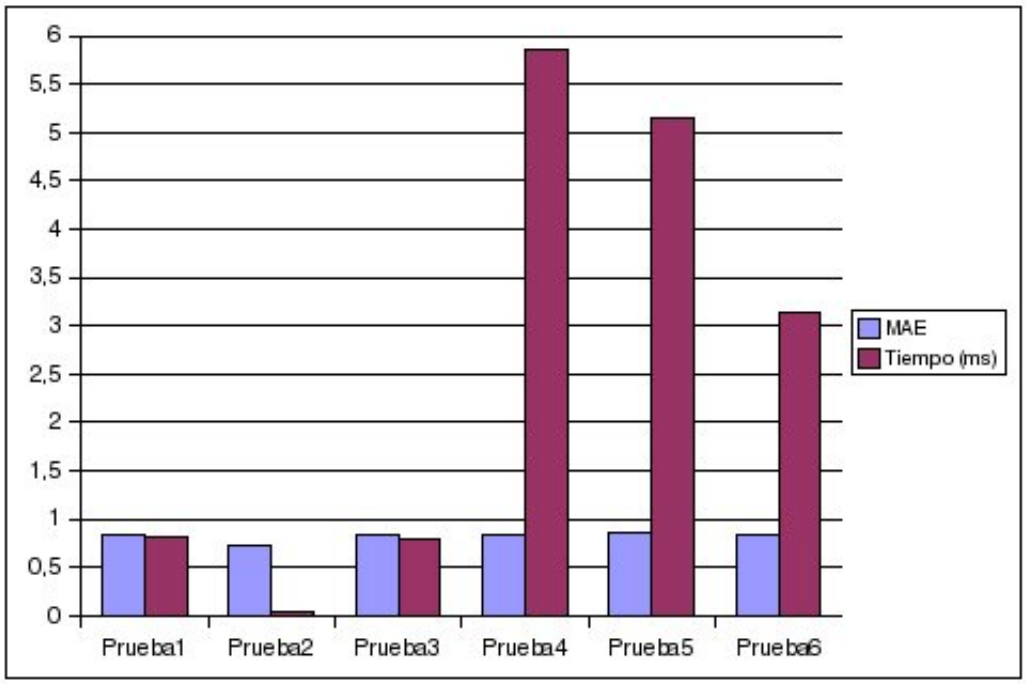


Figura 3.6: Comparativa de todas las pruebas

A la vista del gráfico 3.5 queda claro que tanto para la métrica de precisión MAE como para la métrica temporal es la Prueba 2 la que obtiene unos mejores resultados. Por lo tanto, en la segunda parte de este proyecto se implementará un sistema de recomendación colaborativo basado en una arquitectura cliente/servidor que utilizará un algoritmo de filtrado colaborativo formado por los siguientes valores de los parámetros:

	PRUEBA2
ENT/TEST	0.8/0.2
Nº VEC.	20
MED. SIMILAR.	Coeficiente Coseno
ALG. PRED.	weighted sum

Figura 3.7: Valores del Sistema de recomendación colaborativo

Capítulo 4

DESARROLLO DEL SISTEMA

Una vez presentado el estudio comparativo de algoritmos de filtrado colaborativo que conformaba la primera parte de este proyecto, en este epigrafe se va a detallar el desarrollo de un sistema de recomendación colaborativo basado en una arquitectura cliente/servidor con interfaz web implementando el mejor de los algoritmos previamente estudiados.

Por lo tanto, esta segunda parte es un proyecto de desarrollo software y, como tal, para su desarrollo deben seguirse las actividades de la Ingeniería del Software. No existe una definición única y estandar para la Ingeniería del Software pero, las dos que se presentan a continuación pueden resultar perfectamente válidas para este cometido:

Ingeniería del Software es la construcción de software de calidad con un presupuesto limitado y un plazo de entrega en contextos de cambio continuo.

Ingeniería del Software es el establecimiento y uso de principios y métodos firmes de ingeniería para obtener software económico que sea fiable y funcione de manera eficiente en máquinas reales.

Las actividades que conforman la Ingeniería del Software son las siguientes:

- Especificación de Requerimientos: se obtienen el propósito del sistema y las propiedades y restricciones del mismo.

- Análisis del Sistema: se obtiene un modelo del sistema correcto, completo, consistente, claro y verificable.

- Diseño del Sistema: se definen los objetivos del proyecto y las estrategias a seguir para conseguirlos.

- Implementación: se traduce el modelo a código fuente.

4.1. Especificación de Requerimientos

El primer paso en la Ingeniería del Software debe ser determinar el propósito último del proyecto, las propiedades que debe satisfacer y las restricciones a las que está sometido. Este es, sin duda, una paso de vital importancia dentro del desarrollo de un proyecto software ya que, sin conocer el propósito del proyecto y todas las limitaciones de diversa índole a las que debe hacer frente, difícilmente se podrá realizar una aplicación software que cumpla dicho propósito.

En un proyecto de ámbito comercial para una empresa real, para determinar el propósito del mismo se recurre a una serie de estudios como pueden ser entrevistas con los clientes, encuestas con posibles usuarios, estudios de la situación actual del sistema o estudios de viabilidad. En nuestro caso no nos encontramos ante un proyecto comercial sino ante uno académico por lo que el propósito es conocido desde el mismo momento de la concepción del mismo:

El desarrollo de un sistema de recomendación colaborativo para el alquiler de películas basado en una arquitectura cliente/servidor con interfaz web implementando el mejor algoritmo de filtrado colaborativo de los estudiados previamente.

Habiendo determinado el propósito último del proyecto, el siguiente paso consiste en especificar los requerimientos del mismo. Los requerimientos de un proyecto software son el conjunto de propiedades o restricciones definidas con total precisión, que dicho proyecto software debe satisfacer. Existen dos tipos bien diferenciados de tales requerimientos:

- Requerimientos funcionales: aquellos que se refieren específicamente al funcionamiento de la aplicación o sistema.

- Requerimientos no funcionales: aquellos no referidos al funcionamiento estricto sino a otros factores externos.

En los dos apartados siguientes se pasarán a definir cúales son estos requerimientos (tanto funcionales como no funcionales) para el proyecto del que se ocupa esta memoria. Sin embargo, estas definiciones sólo serán previas ya que en la actividad de análisis del sistema, donde se crearán los casos de uso y sus escenarios, se descubrirán nuevas necesidades que no son observables en esta primera actividad y que permitirán refinar completamente estos requerimientos.

4.1.1. Requerimientos funcionales

Los requerimientos funcionales de un sistema software son aquellos que se encargan de describir las funcionalidades que el sistema debe proporcionar a los usuarios del mismo para cumplir sus expectativas.

Normalmente, estos requerimientos se obtendrían de la interacción con el cliente mediante diversas entrevistas y/o encuestas. En nuestro caso, al tratarse de un proyecto académico, nos encontramos ante la situación de la no existencia de cliente alguno por lo que la información sobre las funcionalidades que debe disponer el sistema se ha obtenido investigando otros sistemas de recomendación colaborativos que ya se encuentran en el mercado, muy especialmente aquellos de reconocido éxito y gran número de usuarios.

En base a estas investigaciones realizadas se ha llegado a la conclusión de que las funcionalidades que el usuario potencial espera de un sistema de recomendación como el nuestro son las siguientes:

- Recibir recomendaciones de objetos que no ha probado y pueden ser de su gusto.

- Puntuar los objetos que ha probado.

- Poder cambiar las puntuaciones de objetos ya puntuados.

- Consultar sus datos personales.
- Modificar su datos en caso de que cambien o sean erróneos.
- Disponer de una lista con todas las objetos disponibles en el sistema y toda la información posible sobre los mismos.
- Disponer de mecanismos de ayuda.

Una vez definidas cuales son las funcionalidades que los usuarios reclaman a un sistema de recomendación, se hace necesario caracterizar de una manera más formal y concreta como va a responder a estas funcionalidades, dentro del ámbito del alquiler de películas, nuestro sistema:

1. Actualizar películas.

   El sistema debe ser capaz de actualizar cada cierto tiempo automáticamente su base de datos de películas para ponerlas a disposición del usuario.

2. Actualizar el modelo de recomendación.

   El sistema deberá actualizar su modelo de recomendación cada cierto tiempo, incorporando las nuevas películas y las nuevas puntuaciones realizadas por los usuarios.

3. Darse de Alta en el Sistema.

   El sistema debe proporcionar un mecanismo por el cual el usuario pueda registrarse en el sistema, para poder ofrecer un servicio personalizado para cada uno.

4. Recibir Recomendaciones de Nuevas Películas.

   El sistema debe proporcionar al usuario, basándose en las películas ya vistas por el mismo y sus puntuaciones, una lista con las películas que no ha visto o alquilado todavía y que pueden ser más de su agrado.

5. Actualizar las recomendaciones.

   El sistema una vez actualizado el modelo de recomendación debe ser capaz de ofrecer nuevas recomendaciones a los usuarios.

6. Puntuar Películas

   El sistema debe permitir al usuario que puntúe películas, tanto aquellas que vaya viendo como aquellas que ya han sido puntuadas pero qué, por alguna razón, el usuario quiera volver a puntuar. Además, debe actualizar la base de datos con esta información.

7. Visualizar Datos Personales

   El sistema debe proporcionar al usuario la posibilidad de visualizar sus datos personales.

8. Modificar Datos Personales

   El sistema debe permitir al usuario modificar sus datos personales y actualizar esta información en la base de datos.

9. Visualizar Todas las Películas

   El sistema debe proporcionar la posibilidad de visualizar un listado con todas las películas disponibles en la base de datos al usuario.

10. Visualizar Sólo las Películas ya Puntuadas

    El sistema debe proporcionar al usuario la posibilidad de visualizar un listado con las películas que ya ha puntuado y cuales son dichas puntuaciones.

11. Visualizar Sólo las Películas Alquiladas pero No Puntuadas

    El sistema debe proporcionar al usuario la posibilidad de visualizar un listado con las películas que ha alquilado pero que todavía no ha puntuado.

12. Consultar ayuda

    El sistema debe proporcionar al usuario algún medio de ayuda para que pueda conocer perfectamente el manejo de la aplicación o resolver cualquier duda puntual que pueda tener

4.1.2. Requerimientos no funcionales

Los requerimientos no funcionales son aquellos que restringen los requerimientos funcionales. Son tan importantes como los propios requerimientos funcionales y pueden incluso a llegar a ser críticos para la aceptación del sistema. Estos requerimientos normalmente especifican propiedades del sistema o del producto en si (plataforma, velocidad, rendimiento...) y del diseño de la interfaz gráfica con el usuario además de todas las restricciones impuestas por la organización (políticas de empresa, estándares, legalidad vigente...).

Al no ser este un proyecto comercial para ninguna organización o empresa real, no debemos someternos a restricciones organizacionales. Por lo tanto, los requerimientos no funcionales que se deben obtener y analizar son los referentes a las necesidades hardware y software de los equipos informáticos para que estos proporcionen al usuario las funcionalidades requeridas de forma eficiente y los referentes a la interfaz gráfica entre la aplicación y el usuario.

- **Requerimientos de Equipo Informático**

Al hablar de los requerimientos del equipo informático y debido a que el marco del desarrollo de la aplicación es una arquitectura cliente/servidor, debemos diferenciar los requerimientos de equipo que necesita el servidor y los que necesita el cliente.

Las **necesidades de equipo informático del cliente** son muy simples ya que tan solo le hace falta un ordenador conectado a Internet (preferiblemente una conexión de banda ancha) y tener instalado un navegador capacitado para visualizar de forma correcta la aplicación (se recomienda Firefox u Opera pero podría ser válido cualquier otro).

Los **requerimientos del equipo informático del servidor**, el cual se aconseja que sea un equipo dedicado, son más amplios y se dividen en dos tipos: los requerimien-

tos de hardware y los requerimientos software.

1)Hardware Servidor

Velocidad: el equipo debe ser lo suficientemente rápido como para ejecutar la aplicación en el menor tiempo posible y con la mayor fiabilidad. Cualquier microprocesador actual es capaz de cumplir con esta labor. RECOMENDABLE: AMD Phenom X4 Quad-Core

Memoria: el equipo debe disponer de la suficiente memoria RAM libre para realizar las operaciones que se soliciten entre la aplicación y la base de datos. RECOMENDABLE: 2GB DE RAM

Almacenamiento: el equipo que haga la labor de servidor debe tener una capacidad de almacenamiento suficiente para almacenar la base de datos con la que trabaja la aplicación y permitir con holgura las transacciones entre ambas entidades. RECOMENDABLE: 1 DISCO DURO SATA II 500GB

Tarjeta gráfica: las tarjetas gráficas de las que disponen los equipos informáticos actuales son de gran potencia por lo que es inutil establecer ningún requerimiento en este aspecto.

Conexión a Internet: el servidor debe encargarse de que la aplicación sea accesible a través de Internet para todos sus usuarios por lo que es indispensable que se encuentre conectado a Internet a través de banda ancha las 24 horas del día.

2)Software Servidor

Sistema Operativo: el servidor de la aplicación trabaja sobre un sistema **Linux**

Ubuntu Hardy Heron 8.04 LTS

Navegador: la aplicación debe poder ser visualizada desde cualquier navegador web actual aunque se recomienda el uso de Firefox en sus últimas versiones.

Sistema Gestor de Bases de Datos: la aplicación trabaja con un gestor de base de datos **Mysql 5**.

El resto del software necesario será: Servidor Apache2, JVM , PHP5

- **Requerimientos de la Interfaz**

Los requerimientos de la interfaz gráfica entre la aplicación y el usuario están íntimamente ligados a la usabilidad y sus principios. La usabilidad se puede definir de varias formas :
- Usabilidad se define coloquialmente como facilidad de uso, ya sea de una página web, una aplicación informática o cualquier otro sistema que interactúe con un usuario.

- Usabilidad se refiere a la capacidad de un software de ser comprendido, aprendido, usado y ser atractivo para el usuario, en condiciones específicas de uso.

- Usabilidad es la efectividad, eficiencia y satisfacción con la que un producto permite alcanzar objetivos específicos a usuarios específicos en un contexto de uso específico.

A partir de estas tres definiciones se pueden obtener los principios básicos de la usabilidad, los cuales se asociarán a los requerimientos no funcionales que deberá cumplir la interfaz gráfica:

* **Facilidad de aprendizaje:** se refiere a la facilidad con la que nuevos usuarios pueden tener una interacción efectiva. Depende de los siguiente factores:

— Predecibilidad: una vez conocida la aplicación, se debe saber en cada momento a que estado se pasará en función de la tarea que se realice.

— Síntesis: los cambios de estado tras una acción deben ser fácilmente captados.

— Generalización: las tareas semejantes se resuelven de modo parecido.

— Familiaridad: el aspecto de la interfaz tiene que resultar conocido y familiar para el usuario.

— Consistencia: siempre se han de seguir una misma serie de pasos para realizar una tarea determinada.

* **Flexibilidad:** relativa a la variedad de posibilidades con las que el usuario y el sistema pueden intercambiar información. También abarca la posibilidad de diálogo, la multiplicidad de vías para realizar la tarea, similitud con tareas anteriores y la optimización entre el usuario y el sistema.

* **Robustez:** es el nivel de apoyo al usuario que facilita el cumplimiento de sus objetivos o, también, la capacidad del sistema para tolerar fallos. Esta relacionada con los siguiente factores:

— Observación: el usuario debe poder observar el estado del sistema sin que esta observación repercuta de forma negativa en él.

— Recuperación de información: la aplicación debe poder deshacer alguna operación y permitir volver a un estado anterior.

— Tiempo de respuesta: es el tiempo necesario para que el sistema pueda mostrar los cambios realizados por el usuario.

4.2. Análisis de Sistema

Una vez conocido el propósito del proyecto software, las propiedades que debe cumplir y las restricciones a las que debe someterse, llega el momento de analizar el sistema y

crear un modelo del mismo que sea correcto, completo, consistente, claro y verificable. Para conseguir ésto se crearán y definirán casos de uso en base a los requerimientos previamente obtenidos y se describirán ciertos escenarios de acción de dichos casos de uso.

4.2.1. Casos de Uso

Un **caso de uso** representa una clase de funcionalidad dada por el sistema como un flujo de eventos. También se puede definir como la representación de una situación o tarea de interacción de un usuario con la aplicación.

Los casos de uso describen como se realiza una tarea de manera exacta y constan de los siguientes elementos:

— Nombre único e unívoco

— Actores participantes

— Condiciones de entrada

— Flujo de eventos

— Condiciones de salida

— Requerimientos especiales

Por lo tanto, es necesario determinar cuales son los actores participantes en cada no de los casos de uso. Un **actor** modela una entidad externa que se comunica con el sistema, es decir, es un tipo de usuario del sistema. Un actor, al igual que un caso de uso, debe tener un nombre único y puede tener una descripción asociada.

En nuestro sistema contamos con los tres actores siguientes:

Cliente: se trata del usuario tipo de la aplicación, el que la va a utilizar para recibir recomendaciones, puntuar películas y demás.

Administrador: se trata del responsable de la aplicación, el que se encarga de actualizar el algoritmo de filtrado colaborativo empleado en base a las nuevas puntuaciones que van realizando los usuarios del sistema.

BBDD: se trata de la base de datos que proporciona los datos a la aplicación.

Una vez definidos cuales van a ser los actores del sistema, es el momento de crear los distintos casos de uso. A la hora de realizar esta acción es importante que cada uno de los requerimientos funcionales ya definidos aparezca en al menos uno de los casos de uso aunque, por otra parte, puede haber casos de uso nuevos, en los que no aparezca ninguno de los requerimientos, ya que estamos en una fase de refinamiento del sistema donde queremos construir un modelo detallado del mismo.

Un paso previo a la creación y descripción de los distintos casos de uso es la obtención de los diversos diagramas de casos de uso de nuestro sistema, al que vamos a llamar SIREACGU.

El primero es un diagrama frontera, es decir, un diagrama que describe completamente la funcionalidad de un sistema:

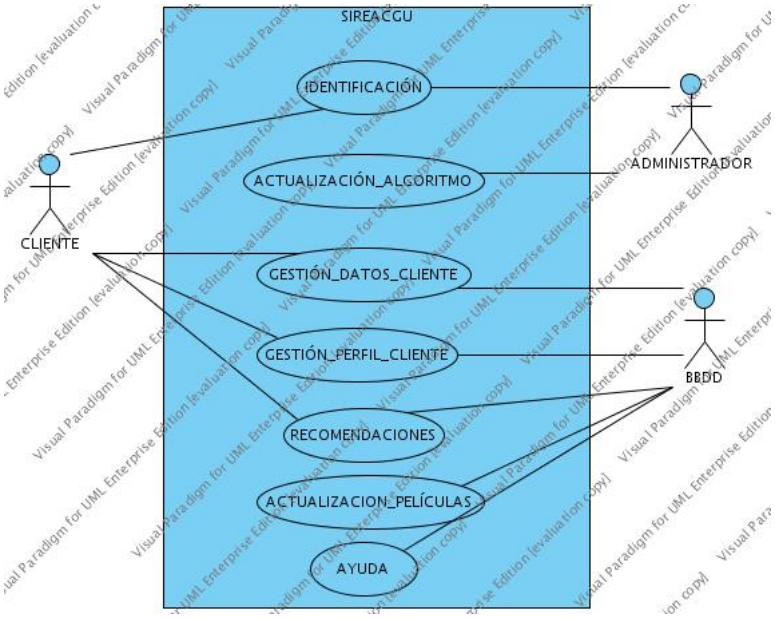


Figura 4.1: Diagrama de frontera del sistema

Los casos de uso mostrados en un diagrama frontera pueden ser lo suficientemente exactos o, por el contrario, pueden ser concretados con un mayor detalle. A la hora de detallar un caso de uso se pueden emplear dos tipos de relaciones:

- «extend»: es una relación cuya dirección es hacia el caso de uso a detallar que representa comportamientos excepcionales del caso de uso.

- «include»: es una relación cuya dirección es contraria a la de la relación «extend» que representa un comportamiento común del caso de uso

En nuestro caso nos encontramos con que los casos de uso GestiónDatosCliente y GestiónPerfilCliente requieren ser detallados en más profundidad:

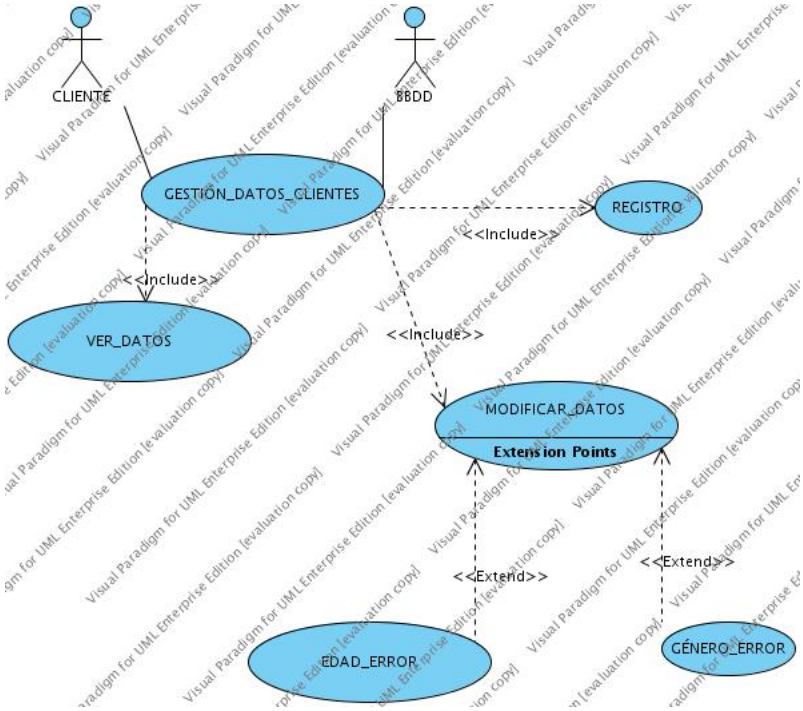


Figura 4.2: Diagrama del caso de uso GestiónDatosCliente

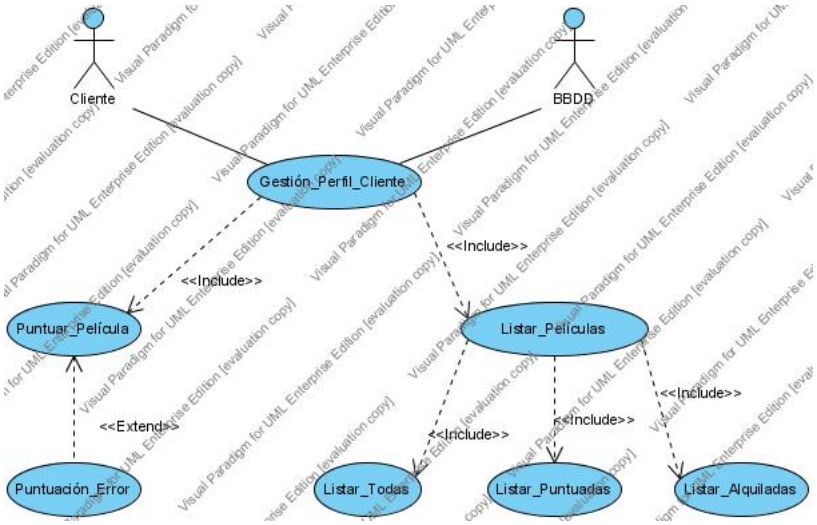


Figura 4.3: Diagrama del caso de uso GestiónPerfilCliente

A continuación, se describen detalladamente cada uno de los casos de uso mostrados

en las figuras anteriores:

- **Caso de Uso 1: Identificación**

Actores participantes: Cliente y Administrador

Condiciones de entrada: Que existan cuentas de usuario en la aplicación

Flujo de eventos:

- El usuario (Cliente o Administrador) inicia la aplicación.
- El sistema muestra un formulario de entrada.
- El usuario introduce su identificador.
- El sistema comprueba que el identificador es válido (E-1).
- Según el identificador sea de: Cliente, entonces el usuario entra al sistema y este le muestra el menú principal. Administrador, entonces el usuario entra al sistema y este le muestra el menú de administrador.

Condiciones de salida: La contraseña ha sido comprobada.

Excepciones:

E-1:El identificador introducido por el usuario no es válido. El sistema informa al usuario de dicha situación. El usuario puede intentar introducir un identificador válido de Cliente o Administrador o salir del caso de uso.

- **Caso de Uso 2:** Actualización algoritmo

Actores participantes: Administrador

Condiciones de entrada: Administrador identificado correctamente.

Flujo de eventos:

- El sistema muestra el menú de administrador.
- Administrador elige la opción Actualizar Algoritmo de Filtrado del menú.

- El sistema lanza un script para actualizar el algoritmo (E-1). Esta actualización se realiza de manera offline para no afectar a los resultados del sistema.

- Una vez terminada la actualización, el sistema muestra la fecha en la que se ha realizado la misma y un mensaje informativo para Administrador. Seguidamente intercambia las bases de datos con los datos actualizados por el algoritmo.

Condiciones de salida: la actualización se produce satisfactoriamente y se solicita salir del caso de uso.

Excepciones:

E-1: el script lanzado por el sistema ha fallado al actualizar el algoritmo. El sistema se lo comunica a Administrador. Administrador puede volver a intentar lanzar el script o salir del caso de uso.

- **Caso de Uso 3:** Gestión Datos Cliente

Actores participantes: Cliente y BBDD

Condiciones de entrada: Existen Cliente y BBDD.

Flujo de eventos principal:

- El sistema muestra el menú principal.

- Cliente elige la opción Datos Personales del menú principal.

- El sistema muestra un menú con tres opciones y le pide a Cliente que elija:

– Si Cliente elige Ver Datos Personales, se realiza S-1.

– - Si Cliente elige Modificar Datos Personales, se realiza S-2.

– - Si Cliente elige Salir, se termina el caso de uso.

Subflujos de eventos:

`S-1: VerDatos`

El sistema se comunica con BBDD (E-1) para obtener los datos personales de Cliente. El sistema muestra sus datos personales a Cliente. El caso de uso se inicia de nuevo.

`S-2: ModificarDatos`

El sistema le muestra a Cliente un formulario de entrada de datos. Cliente introduce su edad (E-2), su género (E-3), su e-mail y su código postal en el formulario. El sistema actualiza BBDD (E-1) con los nuevos datos. El caso de uso se inicia de nuevo.

Condiciones de salida: Se solicita la salida del caso de uso.

Excepciones:

E-1: Ha habido un error al comunicarse el sistema con BBDD. El sistema informa a Cliente de dicha situación. El caso de uso se inicia de nuevo.

E-2: Cliente ha introducido una edad inválida. El sistema informa a Cliente de dicha situación. Usuario puede intentar introducir de nuevo una edad o salir del caso de uso.

E-3: Cliente ha introducido un género inválido. El sistema informa a Cliente de dicha situación. Usuario puede intentar introducir de nuevo un género o salir del caso de uso.

- **Caso de Uso 4:** Gestión-Perfil-Cliente

Actores participantes: Cliente y BBDD

Condiciones de entrada: Existen Cliente y BBDD.

Flujo de eventos principal:

- El sistema muestra el menú principal.

- Cliente elige la opción Perfil de Usuario del menú principal.

- El sistema muestra un menú con tres opciones y le pide a Cliente que elija:

– Si Cliente elige Puntuar Película, se realiza S-1.

– Si Cliente elige Listados, se realiza S-2.

– Si Cliente elige Salir, se termina el caso de uso.

Subflujos de eventos:

`S-1: Puntuar-Película`

1.1. El sistema se comunica con BBDD (E-1) para obtener el listado de las películas. 1.2. El sistema muestra a Cliente el listado anterior. 1.3. Cliente elige la película que desea puntuar. 1.4. El sistema le muestra a Cliente un formulario de entrada de datos. 1.5. Cliente introduce la puntuación (E-2) que le quiere dar a la película en el formulario. 1.6. El sistema actualiza BBDD (E-1) con los nuevos datos. 1.7. El caso de uso se inicia de nuevo.

`S-2: Listar-Películas`

2.1. El sistema muestra un menú con cuatro opciones y le pide a Cliente que elija: - Si Cliente elige Listar Todas, entonces se realiza S-2-1. - Si Cliente elige Listar Puntuadas, entonces se realiza S-2-2. - Si Cliente elige Listar Alquiladas, entonces se realiza S-2-3. - Si Cliente elige Salir, entonces se inicia el caso de uso.

`S-2-1: Listar-Todas`

2.1.1. El sistema se comunica con BBDD (E-1) para obtener todas las películas. 2.1.2. El sistema muestra un listado con todas las películas a Cliente. 2.1.3. El caso de uso se inicia de nuevo.

`S-2-2: Listar-Puntuadas`

2.2.1. El sistema se comunica con BBDD (E-1) para obtener las películas ya puntuadas por Cliente. 2.2.2. El sistema muestra un listado con las películas obtenidas a Cliente. 2.2.3. El caso de uso se inicia de nuevo.

`S-2-3: Listar-Alquiladas`

2.3.1. El sistema se comunica con BBDD (E-1) para obtener todas las películas que ha alquilado Cliente pero todavía no ha puntuado. 2.3.2. El sistema muestra un listado con las películas obtenidas a Cliente. 2.3.3. El caso de uso se inicia de nuevo.

Condiciones de salida: Se solicita la salida del caso de uso.

`Excepciones:`

E-1: Ha habido un error al comunicarse el sistema con BBDD. El sistema informa a Cliente de dicha situación. El caso de uso se inicia de nuevo.

E-2: Cliente ha introducido una puntuación inválida. El sistema informa a Cliente de dicha situación. Cliente puede intentar introducir de nuevo una puntuación o salir del caso de uso.

- **Caso de Uso 5:** Recomendaciones

Actores participantes: Cliente y BBDD

Condiciones de entrada: Existen Cliente y BBDD. Cliente tiene puntuaciones de películas en su perfil.

Flujo de eventos:

1. El sistema muestra el menú principal.
2. Cliente elige la opción Obtener Recomendaciones del menú principal.
3. El sistema se comunica con BBDD (E-1) y obtiene las puntuaciones hechas por Cliente.
4. El sistema calcula las películas recomendadas para Cliente en base a sus puntuaciones previas utilizando el algoritmo de filtrado colaborativo implementado.
5. El sistema obtiene de BBDD (E-1) la información de las películas recomendadas.
6. El sistema muestra a Cliente una lista con las películas recomendadas.

Condiciones de salida: Cliente visualiza correctamente su lista de películas recomendadas y solicita salida del caso de uso.

Excepciones:

E-1: Que haya error al comunicarse el sistema con la base de datos. El sistema informa a Cliente de dicha situación. El caso de uso se inicia de nuevo.

- **Caso de Uso 6:** Ayuda

Actores participantes: Cliente

Flujo de eventos:

1. Cliente elige Ayuda de la lista de opciones.

2. El sistema muestra el mecanismo de ayuda implementado a Cliente.

Condiciones de salida: Cliente visualiza correctamente la ayuda y solicita salida del caso de uso.

- **Caso de Uso 7:** Actualización de películas

Actores participantes: BBDD
Flujo de eventos:

1. El sistema lanza la orden de actualización de películas

2. Se recogen los datos y procesan, para seguidamente incorporarlos al sistema.

Condiciones de salida: Cliente visualiza nuevas películas en sus búsquedas y novedades.

4.2.2. Escenarios

Un caso de uso es una representación abstracta, una abstracción, de una funcionalidad del sistema a realizar. La representación concreta de un caso de uso se realiza mediante la creación de uno o más escenarios que muestren todas las interacciones posibles entre el sistema y sus usuarios.

Un escenario esta formado por los siguientes elementos:

- Un nombre único y unívoco
- Una descripción
- Los actores participantes
- El flujo de eventos

Como se ha indicado, para cada caso de uso puede haber varios escenarios. Para nuestro proyecto se han creado y descrito una cantidad importante de casos de uso por

lo que no vamos a definir todos los escenarios de cada uno de ellos sino que vamos a definir unos pocos que puedan servir como ejemplo de las principales funcionalidades que el sistema va a desarrollar: realizar puntuaciones y obtener recomendaciones.

- ESCENARIO PUNTUAR PELICULA GOLDEN EYE

Nombre: PuntuarPeliGoldenEye

Descripción: El usuario con identificador 89 quiere puntuar con un 4 la película de título GoldenEye

Actores: Cliente89 y BBDD

Flujo de eventos:

1. El usuario entra al sistema.
2. El sistema muestra el formulario de entrada.
3. El usuario introduce 89 en el formulario.
4. El sistema valida correctamente y el usuario entra a la aplicación como Cliente89.
5. El sistema muestra el menú principal de la aplicación.
6. El usuario elige la opción Perfil de Usuario del menú.
7. El sistema muestra un menú con 3 opciones: Puntuar Película, Listados y Salir.
8. El usuario elige la opción Puntuar Películas.
9. El sistema se comunica con BBDD y obtiene todas las películas disponibles.
10. El sistema muestra una lista con las películas obtenidas.
11. El usuario encuentra en la lista la película GoldenEye y la elige.
12. El sistema le muestra un formulario al usuario para que introduzca la puntuación que le va a conceder a la película goldenEye.
13. El usuario introduce en el formulario un 4.

14. El sistema comprueba que 4 es una puntuación válida
15. El sistema actualiza BBDD con la nueva puntuación.
16. El sistema comunica a usuario que la operación se ha realizado con éxito

- ESCENARIO OBTENER REC CLIENTE 345

Nombre: ObtenerRecCliente345

Descripción: El usuario con identificador 345 quiere recibir del sistema una lista de películas recomendadas

Actores: Cliente345 y BBDD

Flujo de eventos:

1. El usuario entra al sistema.
2. El sistema muestra el formulario de entrada.
3. El usuario introduce 345 en el formulario.
4. El sistema valida correctamente y el usuario entra a la aplicación como Cliente345.
5. El sistema muestra el menú principal de la aplicación.
6. El usuario elige la opción Obtener Recomendaciones del menú.
7. El sistema se comunica con BBDD y obtiene las puntuaciones hechas por el usuario.
8. El sistema calcula las películas recomendadas para el usuario en base a sus puntuaciones previas y utilizando el algoritmo de filtrado colaborativo.
9. El sistema obtiene de BBDD información de las 10 películas recomendadas.
10. El sistema muestra al usuario una lista con sus 10 películas recomendadas.

4.3. Diseño del Sistema

Sin duda, realizar de manera adecuada cada una de las actividades que conlleva la Ingeniería del Software es indispensable para la realización de un proyecto software de calidad. Por lo tanto, no se puede decir que ninguna de estas actividades sea más importante que otra. Sin embargo, si podemos decir que la actividad de diseño es la más delicada y la más laboriosa de llevar a cabo.

Es delicada porque si no se lleva a cabo correctamente se hace imposible el codificar, de manera correcta, en la actividad de implementación el modelo obtenido en el análisis del sistema, lo que puede repercutir en el desperdicio de todo el esfuerzo realizado durante las primeras actividades de la Ingeniería del Software.

Y es laboriosa porque las estrategias a seguir para conseguir que esta traducción entre modelo y código se lleve a cabo correctamente son muy diversas y bastante complejas.

Se puede decir, por tanto, que el diseño del sistema es la actividad de la Ingeniería del Software en la que se identifican los objetivos finales del sistema y se plantean las diversas estrategias para alcanzarlos en la actividad de implementación.

Sin embargo, el sistema no se suele diseñar de una sola vez sino que hay que diferenciar entre el diseño y estructura de los datos que se van a manejar y el diseño de la interfaz entre la aplicación y el usuario. Estas dos fases del diseño no se realizan de forma consecutiva una detrás de la otra sino que lo normal es realizarlas de manera concurrente y finalizarlas a la vez.

4.3.1. Diseño de los Datos

La intención de esta fase del diseño software es determinar la estructura que poseen cada uno de los elementos de información del sistema, es decir, la estructura de los datos sobre los que va a trabajar. Estos elementos son:

- Las películas, de las que conocemos su nombre, su año de producción, su fecha de estreno, el/los géneros a los que se adscribe y la URL de su entrada en IMDB.

- Los usuarios, de los que conocemos su edad, si es hombre o mujer, a que se dedica y su código postal además de su identificador del sistema y el número de películas que ha puntuado.

- Las puntuaciones, de las que conocemos el usuario que las hace, las películas que las reciben y, obviamente, el valor numérico de las mismas.

- Las películas alquiladas por los usuarios pero todavía no puntuadas. Por convención se ha decidido que un usuario no puede haber alquilado más de 20 películas sin realizar ninguna puntuación sobre ellas. Esto se debe a que la realización de puntuaciones es lo que va generando el perfil de cada usuario y es en base a ese perfil sobre lo que se realizan las recomendaciones. Por lo tanto, realizar puntuaciones es el aspecto más importante de la aplicación y hay que fomentar que los usuarios puntuen las películas que alquilan y ven.

Una vez determinados cuales son los elementos de información del sistema, se deben obtener sus representaciones en forma de tablas de una base de datos. Para ello, se debe realizar primeramente un diseño conceptual de la base de datos para, posteriormente, obtener las tablas requeridas. Para realizar este diseño conceptual se utilizará el modelo Entidad-Relación.

MODELO ENTIDAD-RELACIÓN

El modelo Entidad-Relación (también conocido por sus iniciales: E-R) es una técnica de modelado de datos que utiliza diagramas entidad-relación. No es la única técnica de modelado pero si es la más extendida y utilizada.

Un diagrama entidad-relación esta compuesto por tres tipos de elementos principales:

Entidades: objetos (cosas, conceptos o personas) sobre los que se tiene información. Se representan mediante rectángulos etiquetados en su interior con un nombre. Una in-

stancia es cualquier ejemplar concreto de una entidad.

Relaciones: interdependencias entre uno o más entidades. Se representan mediante rombos etiquetados en su interior con un verbo. Si la relación es entre una entidad consigo mismo se denomina reflexiva, si es entre dos entidades se denomina binaria, ternaria si es entre tres y múltiple si es entre más (muy raro).

Atributos: características propias de una entidad o relación. Se representan mediante elipses etiquetados en su interior con un nombre.

En los diagramas entidad-relación también hay que tener en cuenta otros aspectos como pueden ser:

Entidades débiles: son aquellas que no se pueden identificar unívocamente solo con sus atributos, es decir, necesitan de estar relacionadas con otras entidades para existir. Se representan con dos rectángulos concéntricos de distinto tamaño con un nombre en el interior del más pequeño.

Cardinalidad de las relaciones: existen tres tipos de cardinalidades de una relación según el número de instancias de cada entidad que involucren:

`Uno a uno:` una instancia de la entidad A se relaciona solamente con una instancia de la entidad B. (1:1) `Uno a muchos:` cada instancia de la entidad A se relaciona con varias de la entidad B. (1:*) `Muchos a muchos:` cualquier instancia de la entidad A se relaciona con cualquier instancia de la entidad B. (*:*)

Claves: cada entidad de un diagrama entidad-relación debe tener una clave, debe estar formada por uno o más de sus atributos.

Una vez conocidos los elementos que forman parte de un diagrama entidad-relación podemos empezar a desarrollar el modelo entidad-relación. Los pasos a seguir son los siguientes:

1. Convertir el enunciado del problema (o, como es nuestro caso, los elementos del sistema software) en un Esquema Conceptual del mismo.

2. Convertir este Esquema Conceptual (o EC) en uno más refinado conocido como Esquema Conceptual Modificado (ECM).

3. Obtener las tablas de la base de datos a partir del Esquema Conceptual Modificado.

ESQUEMA CONCEPTUAL

Necesitamos convertir nuestros elementos del sistema en entidades o relaciones. De manera obvia se puede llegar a la conclusión de que películas y usuarios se convierten en las entidades PELICULAS y USUARIOS respectivamente y que puntuaciones se transforma en la relación PUNTUAR que une las dos entidades. Por su parte, películas alquiladas se transforma en la relación ALQUILAR entre PELICULAS y USUARIOS. Nuestro EC quedaría de la siguiente forma:

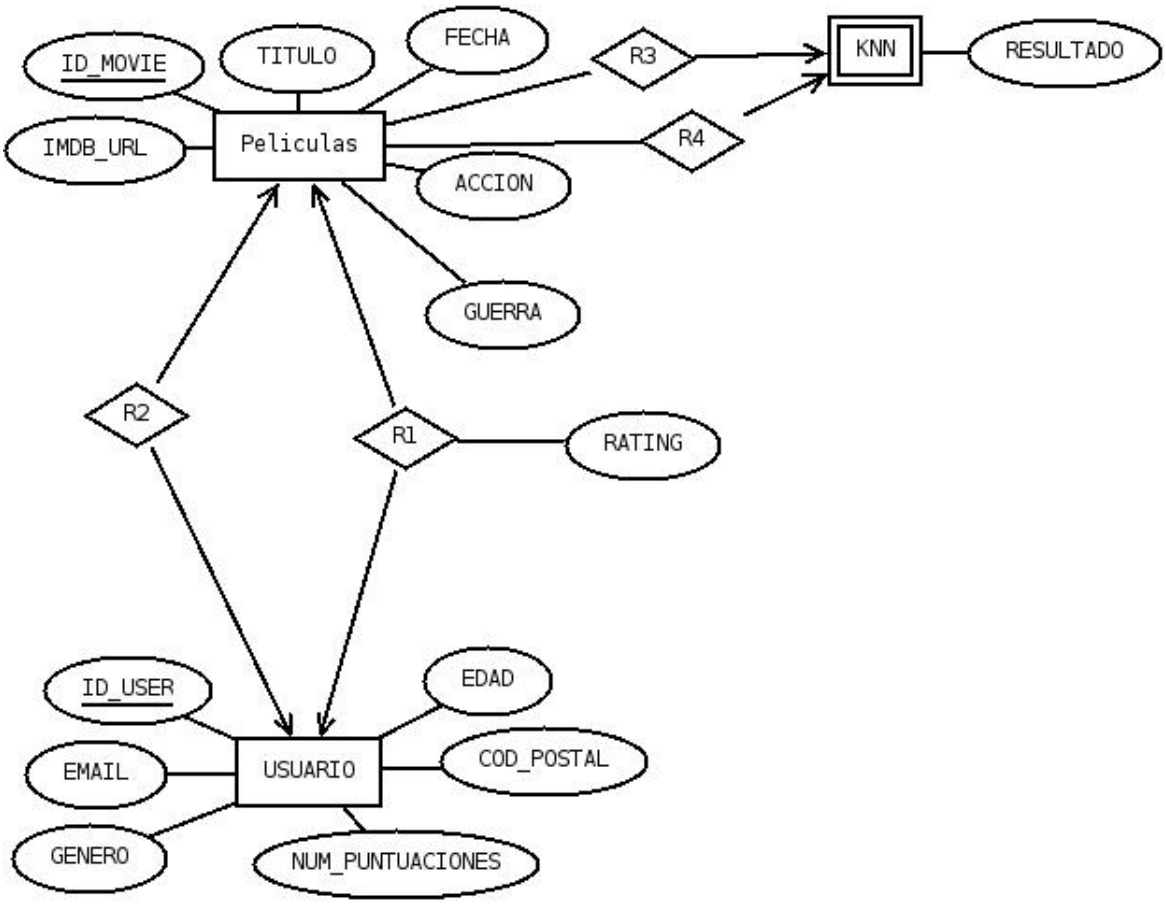


Figura 4.4: Esquema Conceptual

ESQUEMA CONCEPTUAL MODIFICADO

Para obtener el Esquema Conceptual Modificado debemos eliminar todas las entidades débiles, relaciones muchos a muchos y relaciones con atributos que haya en nuestro Esquema Conceptual. Por lo tanto, nuestro ECM queda como sigue:

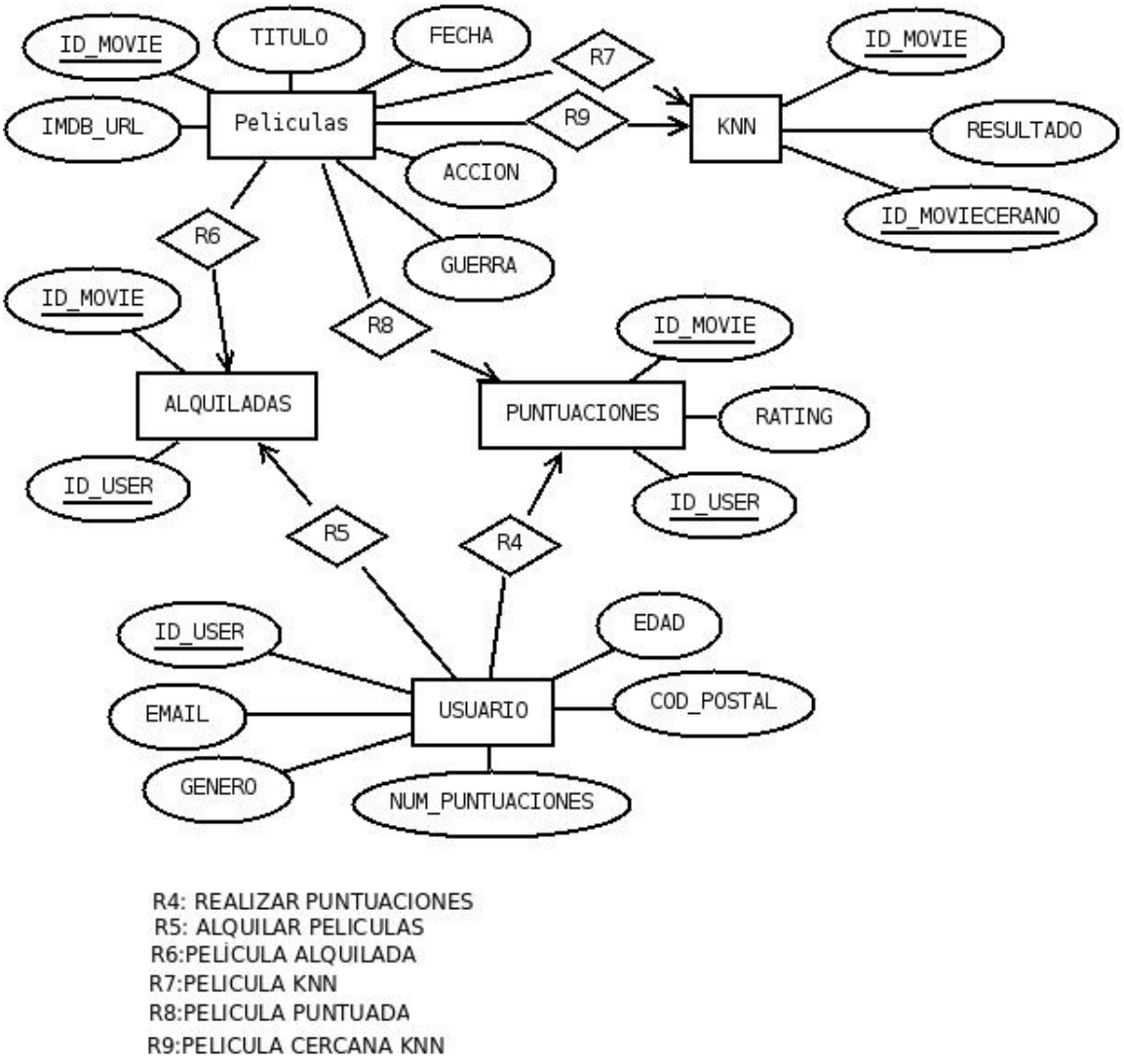


Figura 4.5: Esquema Conceptual Modificado

TABLAS DE LA APLICACIÓN

A partir del ECM obtenido previamente podemos determinar las tablas de la base de datos, teniendo en cuenta que:

Cada entidad del ECM se transforma en una tabla.

Los atributos de una entidad se convierten en los campos de las tablas respectivas.

Por lo tanto, obtendremos las siguientes cuatro tablas: USUARIOS, PUNTUACIONES, PELICULAS y ALQUILADAS. A continuación se detallan cada una de estas tablas.

USUARIOS

Tabla que contiene la información sobre los usuarios de la aplicación. Está formada por los siguientes campos:

CAMPO	TIPO	DESCRIPCIÓN	CLAVE
ID_USER	INTEGER	Identificador unívoco de usuario	*
EDAD	INTEGER	Edad del usuario	
GENERO	VARCHAR(1)	M si el usuario es hombre y F si es mujer	
EMAIL	VARCHAR(125)	Profesión del usuario	
COD_POSTAL	VARCHAR(5)	Código Postal del usuario	
NUM_PUNTUACIONES	INTEGER	Número de películas que ha puntuado el usuario. Campo calculable.	

Figura 4.6: Tabla de Usuarios

PELICULAS

Tabla que contiene la información de todas las películas de la aplicación. Está formada por los siguientes campos:

CAMPO	TIPO	DESCRIPCIÓN	CLAVE
ID_MOVIE	INTEGER	Identificador unívoco de la película	*
TITULO	VARCHAR(81)	Título de la película	
FECHA	DATE	Fecha de estreno de la película	
IMDB_URL	VARCHAR(134)	Dirección URL de la página en IMDB de la película	
DESCONOCIDO	BOOLEAN	La película es de género desconocido	
ACCION	BOOLEAN]	La película es de acción	
AVENTURAS	BOOLEAN	La película es de aventuras	
ANIMACION	BOOLEAN	La película es de animación	
INFANTIL	BOOLEAN	La película es infantil	
COMEDIA	BOOLEAN	La película es de comedia	
CRIMEN	BOOLEAN	La película es de crimen	
DOCUMENTAL	BOOLEAN	La película es documental	
DRAMA	BOOLEAN	La película es de drama	
FANTASIA	BOOLEAN	La película es de fantasía	
NEGRO	BOOLEAN	La película es de género negro	
TERROR	BOOLEAN	La película es de terror	
MUSICAL	BOOLEAN	La película es musical	
MISTERIO	BOOLEAN	La película es de misterio	
ROMANTICO	BOOLEAN	La película es romántica	
CIENCIA-FICCION	BOOLEAN	La película es de ciencia-ficción	
THRILLER	BOOLEAN	La película es un thriller	
GUERRA	BOOLEAN	La película es bélica	
WESTERN	BOOLEAN	La película es un western	

Figura 4.7: Tabla de Películas

PUNTUACIONES

Tabla que contiene la información sobre las puntuaciones que hacen los distintos usuarios de la aplicación sobre las distintas películas. Está formada por los siguientes campos:

CAMPO	TIPO	DESCRIPCIÓN	CLAVE
ID_USER	INTEGER	Identificador unívoco del usuario que realiza la puntuación	*
ID_MOVIE	INTEGER	Identificador unívoco de la película puntuada	*
RATING	INTEGER	Valor de la puntuación. Debe ser 1, 2, 3, 4 o 5. Cualquier otro valor es erróneo	

Figura 4.8: Tabla de Puntuaciones

ALQUILADAS

Tabla que contiene información sobre las películas alquiladas pero todavía no puntuadas por parte de cada usuario de la aplicación. Está formada por los siguientes campos:

CAMPO	TIPO	DESCRIPCIÓN	CLAVE
ID_USER	INTEGER	Identificador unívoco del usuario que alquila la película	*
ID_MOVIE	INTEGER	Identificador unívoco de la película alquilada	*

Figura 4.9: Tabla de Puntuaciones

KNN

Tabla que contiene los resultados obtenidos de aplicar el clasificador knn a la base de datos de películas y usuario junto con sus puntuaciones. Está formada por los siguientes campos:

CAMPO	TIPO	DESCRIPCIÓN	CLAVE
ID_MOVIECERCANO	INTEGER	Identificador de la película con similud más cercana a ID_MOVIE	*
ID_MOVIE	INTEGER	Identificador unívoco de la película	*
RESULTADO	FLOAT	Similitud entre ID_MOVIE e ID_MOVIECERCANO	

Figura 4.10: Tabla de knn

4.3.2. Diseño de la Interfaz

En esta fase del diseño del sistema software se define cual va a ser la apariencia visual de la aplicación, es decir, se define la interfaz visual entre el usuario y la aplicación. Sin duda, realizar un buen diseño de la interfaz resulta primordial ya que esta debe presentarse atractiva al usuario de la aplicación pero a la vez le debe de resultar fácil de entender y trabajar sobre ella.

Esta importancia se acrecienta aun más en nuestro caso ya que la interfaz de nuestro proyecto es una interfaz web y la usabilidad web es un tema candente, foco de cierta

controversia. Esta controversia se debe a que para las aplicaciones con interfaces web no existe una guía de estilo estándar como la puede haber, por ejemplo, para desarrollar interfaces para aplicaciones de escritorio de Windows XP/Linux y que resulten, a la vez, atractivas y familiares. Cada programador, desarrollador o diseñador web debe definir su propia guia de estilo y procurar que, en base a ella, la interfaz resultante consiga unas cotas dignas de atractivo visual, familiaridad y facilidad de uso.

En este apartado vamos a definir nuestra guía de estilo, a describir y analizar las metáforas empleadas y por último mostraremos unos prototipos del sistema:

4.3.2.1. Guia de Estilo

Antes de ponerse a diseñar una interfaz de usuario, se debe definir el estilo de la misma. Esto es de vital importancia cuando el diseño va a ser compartido entre varios diseñadores, ya que ayuda a mantener la coherencia interna de la interfaz.

Sin embargo, en contra de lo que pueda parecer en un principio, también es de mucha utilidad definir una guía de estilo cuando solo hay un diseñador encargado de la interfaz. Esto se debe a varias razones:

A veces es posible que mantener la coherencia y consistencia de una interfaz, si esta es muy grande o muy ambiciosa, sea algo complicado incluso si solo hay un diseñador si no tiene una base.

El diseñador primitivo puede, por las más diversas razones, abandonar el diseño y es de utilidad para sus sustitutos contar con una guía de estilo predefinida para no tener que empezar de cero otra vez. Lo mismo puede aplicarse si no es el diseñador original el que se encarga del mantenimiento o la actualización de la interfaz.

Quedando demostrada la utilidad del uso de guías de estilo podemos pasar a definir las reglas, normas y recomendaciones que contendrá la guía de estilo de nuestra interfaz:

Fuentes:

Tipo: Arial,Times New Roman Tamaño:11px Cabecera: 36px Otras circunstancias:12px

Colores:

Se usarán 3 tonalidades de azul distintas, un tonalidad para la cabecera, otra para

el menu y otra para el pie. La parte central de formulario ira en blanco con las letras en negro o rojo.

Enlaces:

sin subrayado. Fondo blanco. Los enlaces externos se abren en una nueva ventana o pestaña del navegador. Los internos a la aplicación lo hacen en la misma ventana o pestaña. Los colores empleados para los enlaces son: azul para el enlace en si y si el ratón se situa en lo alto se pone de color rojo. Todos los enlaces se rejiran por estos colores menos la ayuda, que estará siempre en color naranja.

Logotipo:

arriba a la derecha (en la Cabecera). Esta presente en todas las páginas del sitio web

Ayuda:

abajo a la izquierda (en el Pie). Esta presente en todas las páginas del sitio web.

4.3.2.2. Metáforas

Una metáfora es el empleo de un objeto con un significado o dentro de un contexto diferente al habitual.Al diseñar una interfaz gráfica, la utilización de metáforas resulta muy útil ya que permiten al usuario, por comparación con otro objeto o concepto, comprender de una manera más intuitiva las diversas tareas que la interfaz permite desarrollar.

Al igual que pasa en el ámbito de la literatura o la lingüística, para que una metáfora cumpla con su cometido, el desarrollador de la aplicación y el usuario final de esta deben tener una base cultural similar. Es muy posible que el uso de un icono de manera metafórica sea entendido de una manera por el usuario occidental y de otra bien distinta por un usuario de extremo oriente. Hay que intentar, por lo tanto, que las metáforas empleadas sean lo más universales posibles para que asi sean comprendidas a la perfección por la mayor parte del público potencial.

Las aplicaciones de escritorio de Linux suelen seguir la Guía de Estilo según estén en entorno GNOME o KDE y utilizan una serie de metáforas con las que el usuario esta plenamente familiarizado (por ejemplo, una lupa con un signo '+én su interior establece

que la función del icono es, inequivocamente, la de realizar un aumento de zoom). En el mundo de las aplicaciones web también existen una cantidad de metáforas de amplia difusión como puede ser, por ejemplo, el celebre carrito de la compra que emplean casi todos los comercios online.

Pero las metáforas no solo dependen del tipo de aplicación (escritorio o web) sino también del ámbito de la misma. Por ejemplo, el carrito de la compra es una metáfora conocida por todos pero si nuestra aplicación no va a vender nada al usuario no resulta conveniente utilizarla ya que puede confundir.

Metaforas Usadas:

P: Cada vez que veamos este simbolo, nos estará indicando que la película puede ser puntuada por el usuario. A: Cada vez que veamos este simbolo, nos estará indicando que la película puede ser alquilada por el usuario. link: Cada vez que veamos este simbolo, nos estará indicando que es un enlace externo a información de la película en imdb.

Aunque no son muy usuales estas metaforas, pretendemos que el usuario lo indentifique con dicha acción.

4.3.2.3. Prototipos

En este apartado vamos a definir la estructura de nuestra interfaz con el usuario, ya que la parte servidor no tendrá interfaz ninguna pues se ejecutará automáticamente sin intervención manual.

Mediante la elaboración de prototipos se pretender esbozar lo que será la interfaz de usuario de nuestra aplicación.

Dichos prototipos no expresan el diseño final, simplemente una día de lo que será nuestro sistema. Estos prototipos serán susceptibles de cambio durante el proceso de implementación.

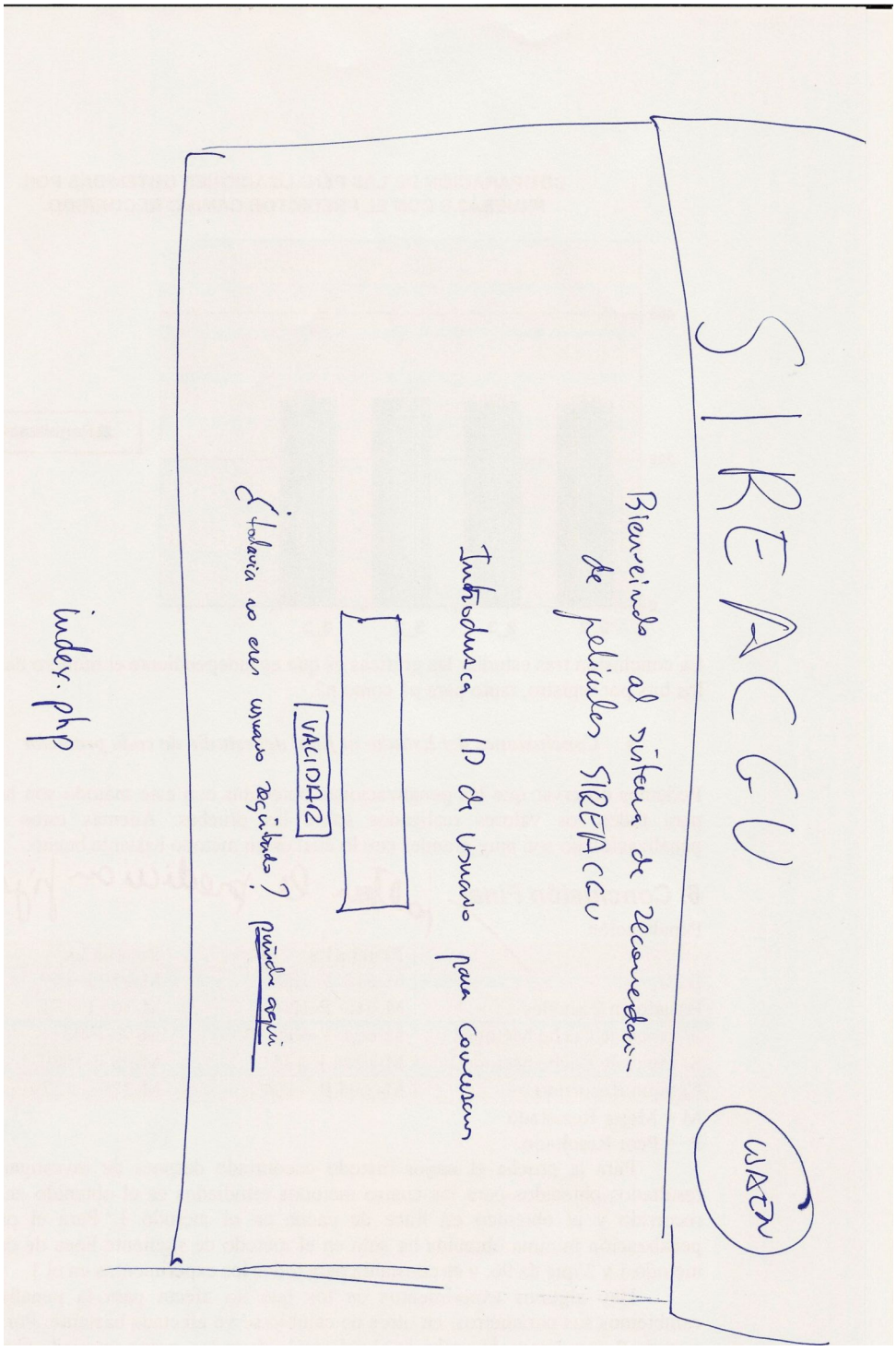


Figura 4.11: Prototipo de entrada al Sistema

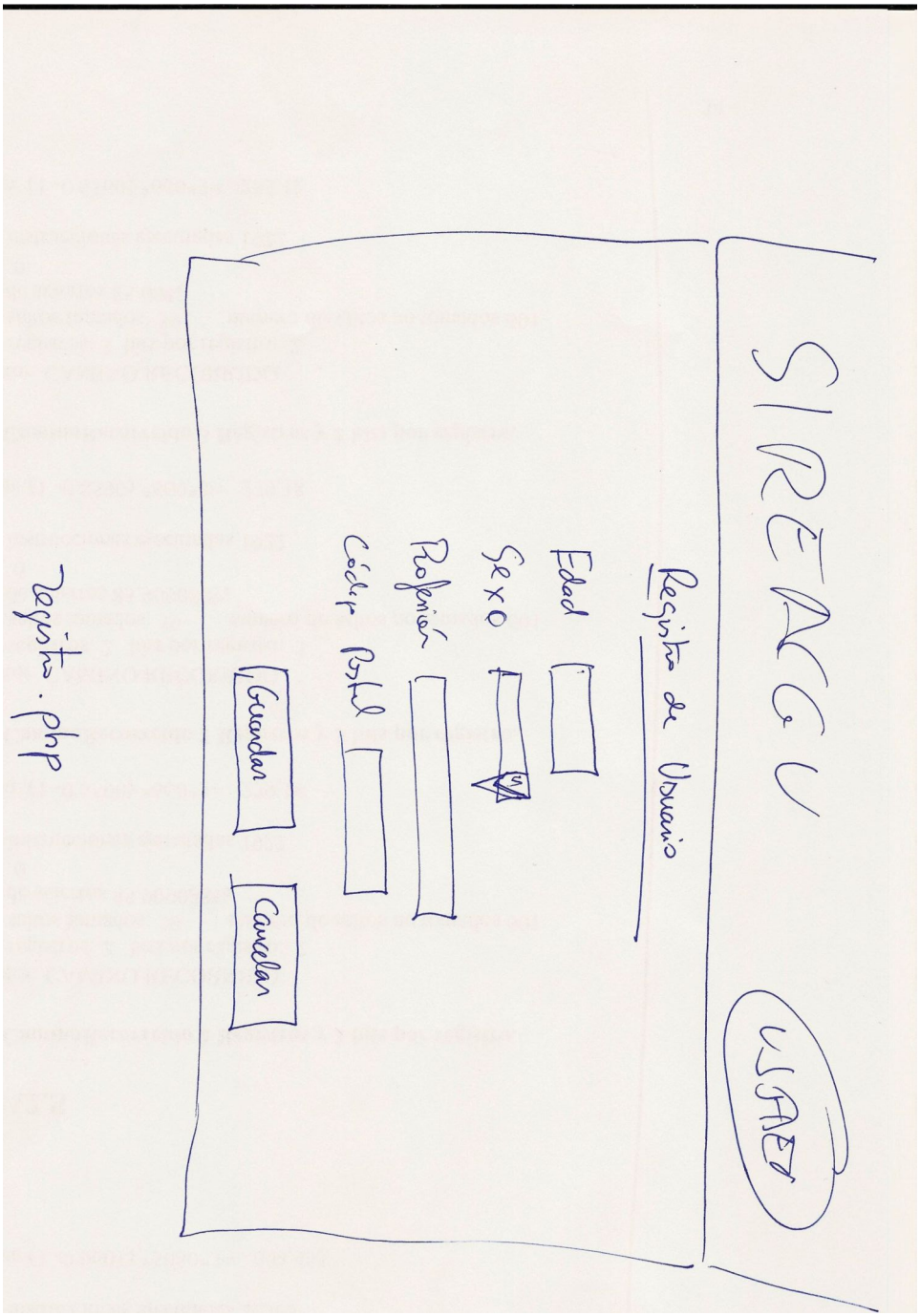


Figura 4.12: Prototipo de Registro de Usuario

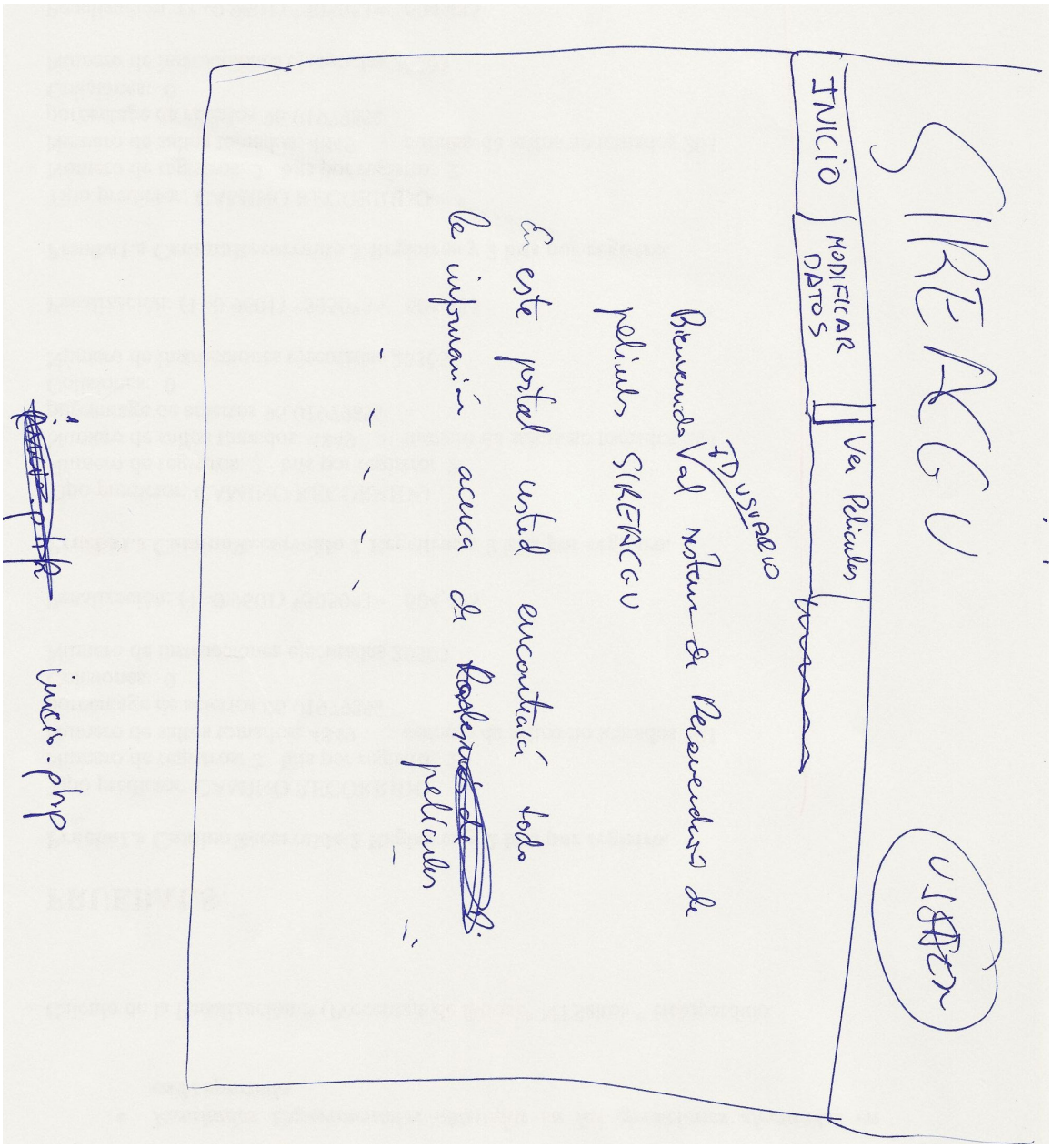


Figura 4.13: Prototipo de Inicio tras validación de usuario

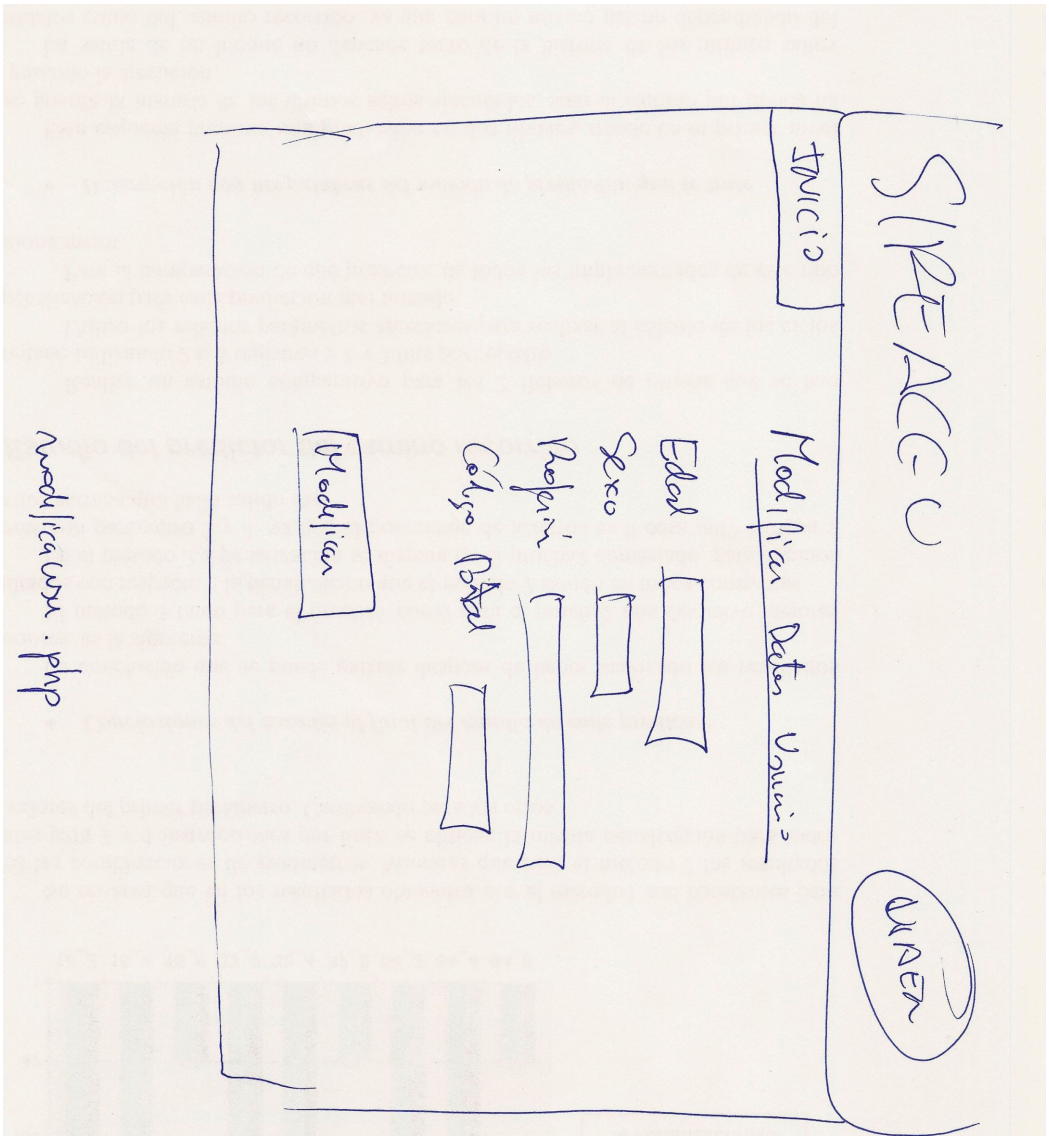


Figura 4.14: Prototipo de Modificación de datos de usuario

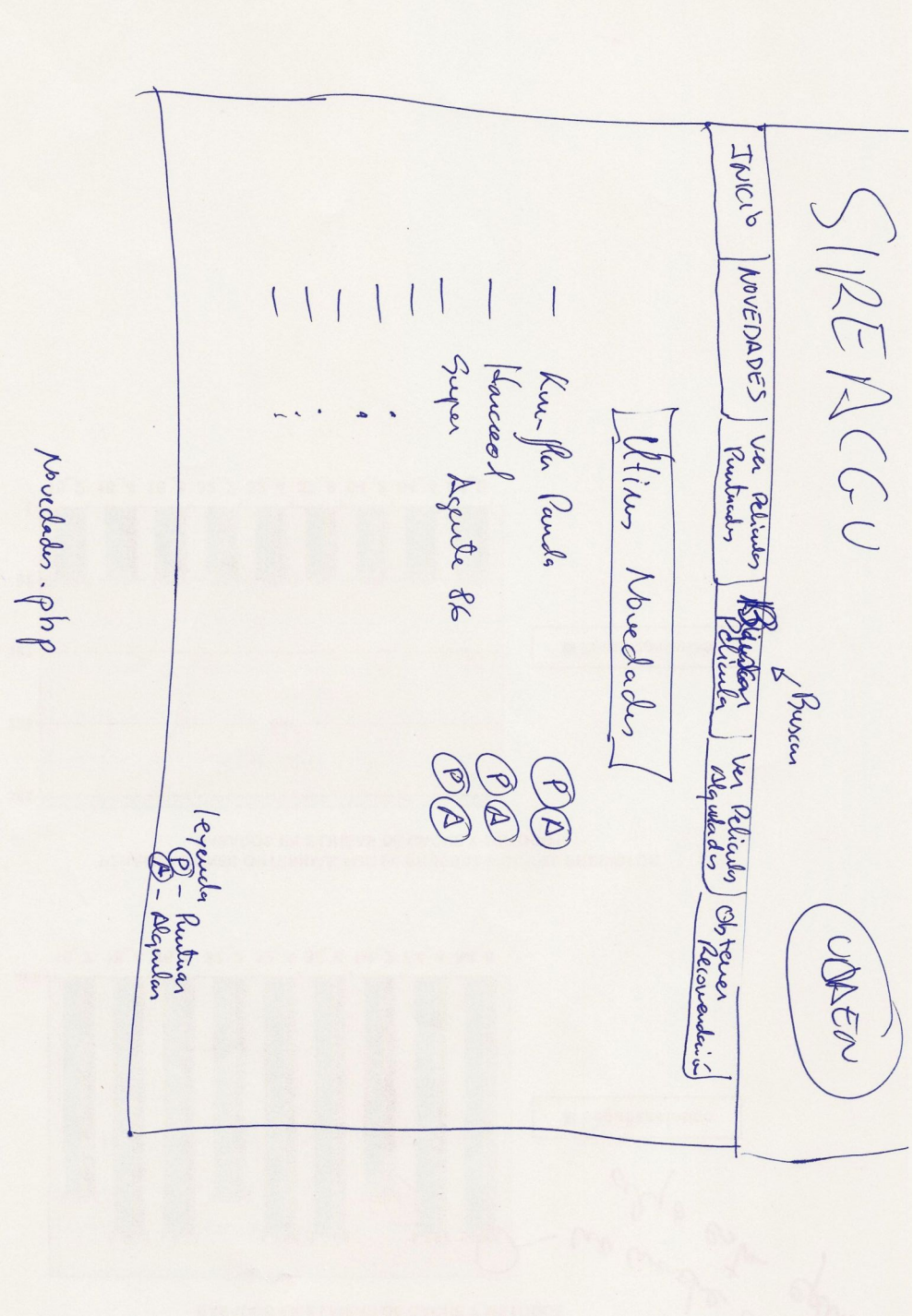


Figura 4.15: Prototipo de pantalla de Novedades

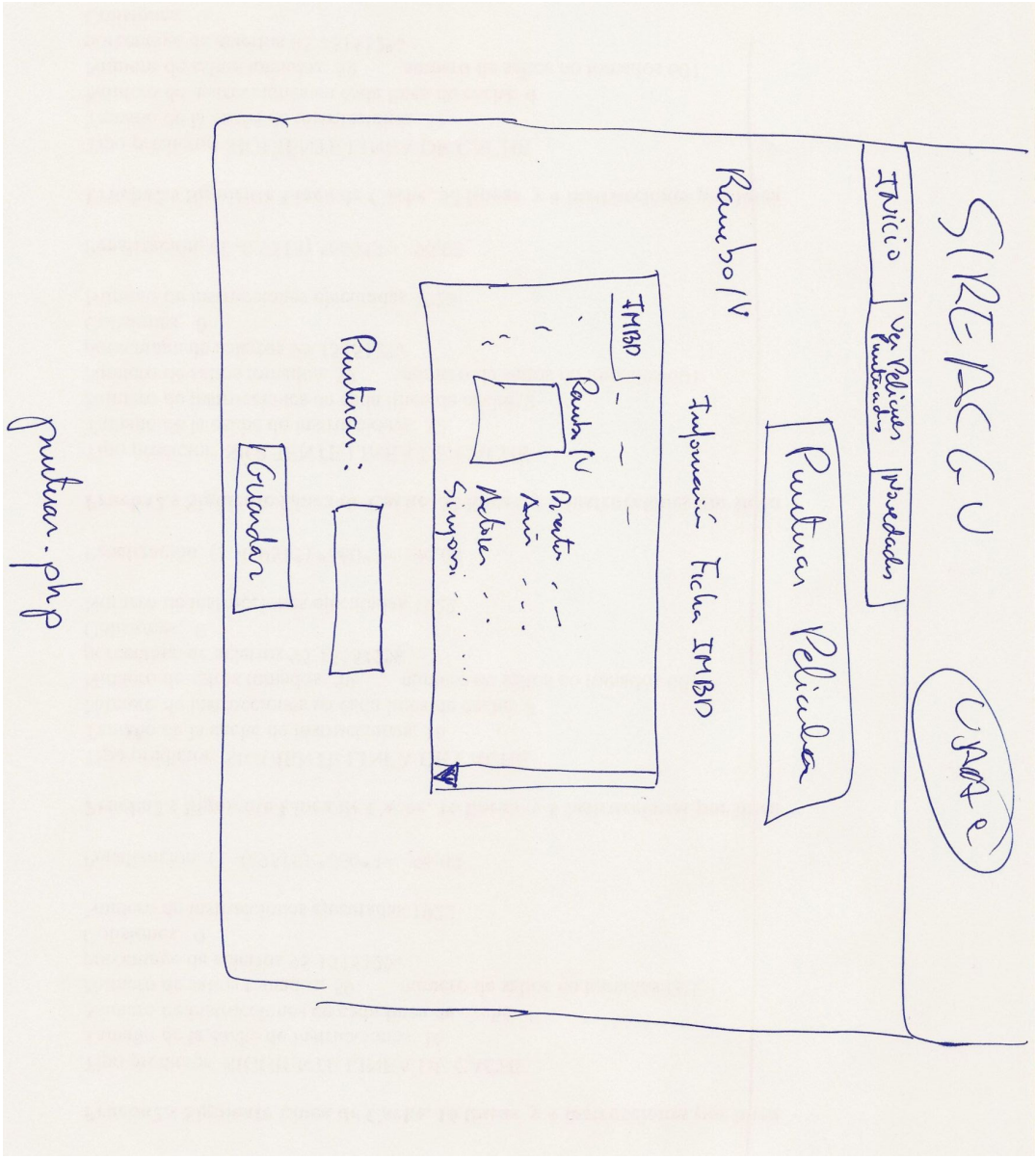


Figura 4.16: Prototipo de pantalla de puntuación de película

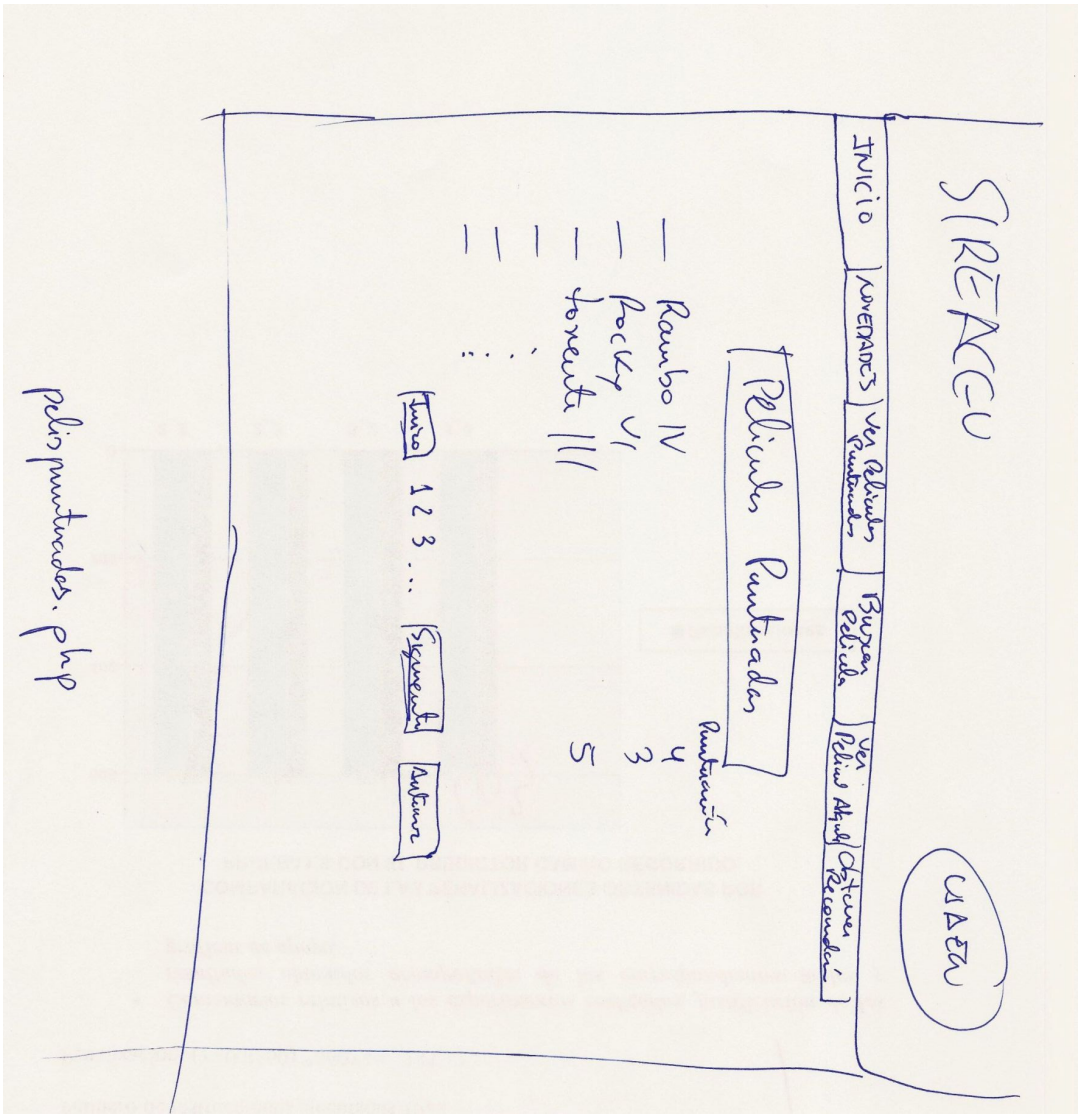


Figura 4.17: Prototipo de pantalla de películas puntuadas

Figura 4.18: Prototipo de pantalla de búsqueda de películas

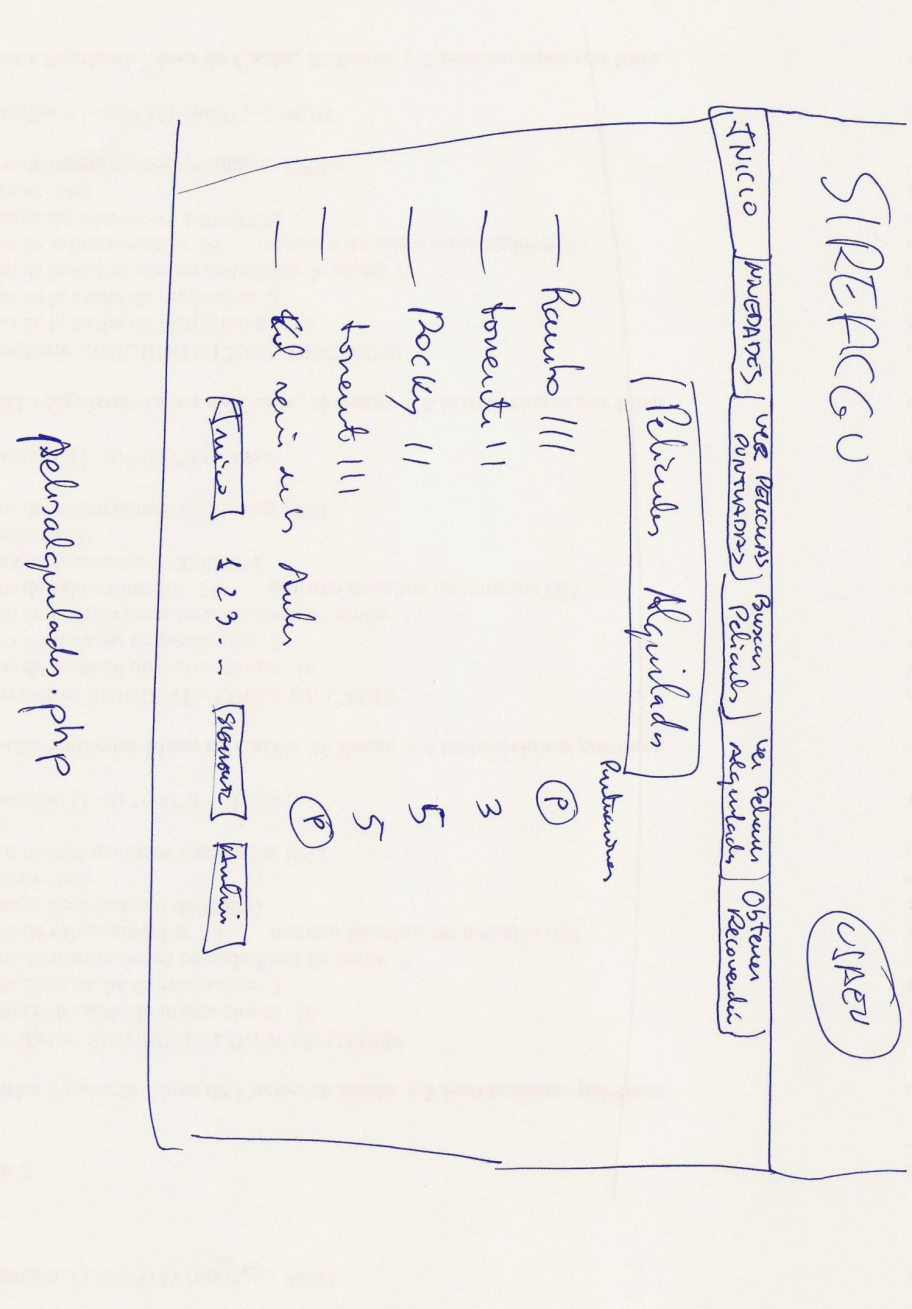


Figura 4.19: Prototipo de pantalla de películas alquiladas

Figura 4.20: Prototipo de pantalla de recomenadación de películas

4.4. Implementación

La implementación es la actividad final de la Ingeniería del Software, aquella en la que el modelo obtenido en las actividades anteriores se debe transformar en código fuente. Para ello se debe ser cuidadoso en la elección del lenguaje de programación empleado para la codificación y de la herramienta utilizada para generarla.

4.4.1. Tipo de arquitectura de la aplicación

En nuestro caso, vamos a desarrollar un sistema de recomendación con una arquitectura cliente/servidor y una interfaz web de comunicación con los usuarios. El funcionamento de las arquitecturas de este tipo es sencilla: la aplicación se encuentra en un servidor central al que los usuarios acceden a través de un software cliente, en nuestro caso un navegador web. Una vez que ha accedido a la aplicación, el usuario realiza peticiones que el servidor tiene que atender para generar una respuesta comprensible para el cliente.

Una arquitectura cliente/servidor libera, por lo tanto, al usuario final de la aplicación de tener que instalarla en su máquina y consigue que cada usuario solo pueda acceder a la información que le corresponde. Además, este tipo de arquitectura, gracias a su diseño modular, es fácilmente escalable y ampliable tanto en nuevos clientes como en servidores añadidos.

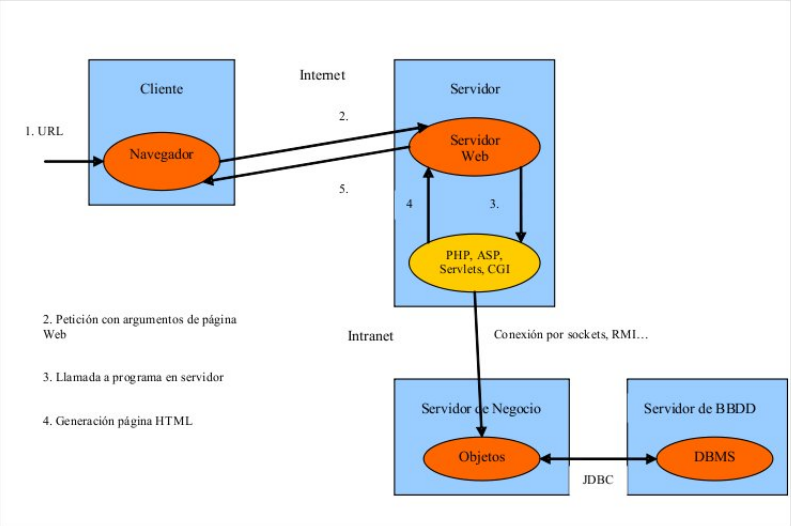


Figura 4.21: Modelo Cliente Servidor

4.4.2. Lenguajes de programación utilizados

- **Parte SERVIDOR**

La implementación de esta parte se ha realizado usando Java y sentencias sql (Mysql). El entorno de desarrollo elegido ha sido Netbeans 6.

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

La implementación original y de referencia del compilador, la máquina virtual y las librerías de clases de Java fueron desarrollados por Sun Microsystems en 1995. Desde

entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre noviembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java todavía no es software libre).

NetBeans se refiere a una plataforma para el desarrollo de aplicaciones de escritorio usando Java y a un entorno de desarrollo integrado (IDE) desarrollado usando la Plataforma NetBeans.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio 2000 y continúa siendo el patrocinador principal de los proyectos.

- **Parte CLIENTE**

Resulta obvio ante la arquitectura y el funcionamiento previsto de nuestra aplicación que el uso de HTML simple y llano no es adecuado sino que se necesita otro lenguaje capaz de generar contenido dinámico desde el servidor de manera transparente al usuario

final. Existen varias alternativas para realizar esto, desde Perl a ASP (Active Server Pages) o JSP (Java Server Pages) pasando por el uso de CGIs (Common Gateway Interface). Sin embargo, finalmente, nos hemos decantado por el uso de PHP.

PHP, acrónimo recursivo de Hypertext Preprocessor, es un lenguaje de programación interpretado, que se ejecuta del lado del servidor y genera contenido dinámico a petición del cliente. Es un lenguaje que tiene una importante serie de ventajas sobre otros lenguajes que realizan funciones parecidas como son las siguientes:

Es libre, abierto y multiplataforma.

Sintaxis similar a lenguajes estructurados como C.

Capacidad de conexión con múltiples gestores de bases de datos.

Cuenta con mecanismos para trabajar con ficheros, tratar textos, generar imágenes de manera dinámica y tratar documentos XML.

Esta ampliamente documentado.

Cuenta con un gran número de extensiones y módulos.

No requiere declaración de variables.

Estas características le hacen ideal para nuestros propósitos:

El cliente solicita cualquier funcionalidad.

El servidor, mediante PHP, conecta con nuestra base de datos en Access y obtiene los datos pertinentes.

También mediante PHP realiza los cálculos y acciones que sean necesarios sobre esos datos.

Finalmente, genera el código xHTML adecuado y se lo presenta al cliente de manera transparente.

Con PHP es suficiente para satisfacer las funcionalidades que debe presentar la aplicación a sus usuarios. Sin embargo, para realizar una implementación de la interfaz

web adecuada se hace necesario el uso de otros dos lenguajes de programación: CSS y Javascript.

CSS, acrónimo de Cascade Style Sheets, es un lenguaje formal que ayuda a separar la estructura interna de un documento de su presentación externa. Las etiquetas de estilo CSS pueden presentarse tanto dentro de un documento HTML como en un documento aparte (con extensión .css) al que el documento HTML se encarga de llamar cuando es necesario. De esta última manera no solo se consigue separar la estructura de la presentación sino que se consigue la centralización del estilo ya que una sola hoja de estilos CSS puede ser invocada por distintas páginas de la aplicación web lo que ayuda de manera muy importante al mantenimiento de la coherencia y consistencia del diseño de la aplicación.

En nuestra aplicación, el uso de hojas de estilo CSS es algo ineludible ya que asi se consigue que las sentencias PHP del servidor generen, simplemente, el código xHTML necesario para responder a la petición del cliente sin entrar en temas del diseño o visualización de esta respuesta, de lo que se encargará el estilo CSS predefinido.

Por su parte, Javascript, lenguaje interpretado de sintaxis similar a lenguajes como Java o C que se ejecuta del lado del cliente, ayuda a PHP de otra manera: filtrando los datos de las peticiones de los clientes, dejando realizar la petición al servidor solo cuando estos son válidos. Si los datos son erroneos informan al cliente de su error mediante mensajes de error o alerta.

Al igual que ocurre con CSS, el código Javascript puede ir incrustado dentro del documento HTML o estar almacenado en ficheros aparte (con extensión .js) y ser invocados por el documento. Para nuestra aplicación, tanto para los estilos CSS como para el código Javascript, nos hemos decantado por la segunda opción.

Para generar código xHTML, CSS, PHP y Javascript no hace falta ninguna herramienta o entorno especifico de desarrollo ya que con un simple editor de textos se pueden escribir las sentencias y etiquetas y guardar el resultado con la extensión correspondiente. Sin embargo, existen multitud de herramientas de desarrollo que facilitan de manera enorme esta tarea de codificación. Las hay tanto libres y gratuitas (como pueden

ser Quanta Plus o NVU) como propietarias. Para este proyecto hemos elegido NVU.

4.4.3. Actualización base de datos Películas

Nuestro sistema está pensado para actualizar la base de datos de películas automáticamente. Para ello extraeremos la información de la página http://www.imdb.com/nowplaying donde semanalmente cuelgan los últimos títulos de las películas en cartelera. La información que se encuentra la página citada está en formato texto (concretamente html), por ello hemos tenido que hacer un módulo para extraer los titulos y sus enlaces del código html. Una vez que hemos tenido los enlaces a cada uno de los títulos, hemos hecho otro proceso de extración de información de cada una de las páginas de los enlaces. En concreto se ha extraido la fecha de salida y el genero. Todas esta información es automáticamente incorporada a nuestra base de datos de películas, aunque no serán recomendadas hasta que se actualice el algoritmo de filtrado. Este proceso de actualización se realiza de manera offline para no molestar a los usuarios del sistema.

4.4.4. Actualización del Algoritmo de Filtrado

La base primordial para desarrollar un sistema de recomendación colaborativo es contar con un buen algoritmo de filtrado colaborativo. Para ello es necesario refinarlo conforme los clientes vayan mejorando sus perfiles puntuando nuevas películas.

Para realizar esto se lanza una aplicación Java en el servidor que se encargará de actualizar el algoritmo así como de insertar las nuevas peliculas que hallan sido publicadas. La aplicación está programada para su ejecución mediante un temporizador (usaremos el cron). Esta actualización requiere de un tiempo importante para realizarse ya que se deben recalcular los grupos de vecinos.

4.4.5. Implementación del Analisis Offline

En este proyecto se ha implementado el análisis offline de datos. Consiste en realizar los cálculos sobre otra base de datos distinta a la que estan usuando los usarios. Una vez

terminado el proceso de cálculo esta se intercambia con la que se está usuando actualmente, quedando el sistema inhabilitado durante 15 segundos.

Para realizar la implementación se ha creado una base de datos auxiliar llamada SIREACGU2 con una sóla tabla llamada knn. En esta tabla se van depositando los datos que va el servidor generando de aplicar el algoritmo knn sobre los datos que contiene de la base de datos original pasados a memoria. Una vez terminado el proceso, se procede a sustituir la tabla knn de la base de datos SIREACGU por la knn de la SIREACGU2. Todo este proceso es totalmente transparente al usuario y no afecta a las recomendaciones.

Capítulo 5

CONCLUSIONES

En la actualidad vivimos en un mundo donde la cantidad de información que recibimos comienza a ser casi improcesable por el ser humano. La rápida explosión de Internet ayudada por la mejora de las comunicaciones y el almacenamiento de datos masivo, ha contribuido a un fortísimo desarrollo de la sociedad de la información. A día de hoy se pueden hacer a través de Internet cosas absolutamente inimaginables hace solo unos años, como pueden ser operaciones bancarias, leer el periódico, ver la televisión, asistir a clase, consultar el catálogo de tu biblioteca, pedir la cena o comprar on-line en una tienda del otro lado del mundo. Millones de usuarios y millones de posibilidades. Esto hace que haya una sobrecarga de información y que el usuario que realiza una búsqueda por el Internet puede llegar a sentirse confundido y frustado y abandone su intento.

Con este panorama es donde empiezan a tener cabida los sistemas de recomendación. Un sistema de recomendación ayuda al usuario a encontrar lo que necesita por medio de una serie de recomendaciones basadas en informaciones proporcionadas por el propio usuario, otros usuarios o expertos en la materia. Se puede empezar a hablar del nacimiento del marketing personalizado a través de las e-shop. Este valor añadido que las e-shop ponen a disposición de los usuarios está orientado básicamente a la fidelización de clientes. Una vez captado el cliente, que vuelva siempre a nuestro sitio cuando necesite algo de su agrado. Sin duda, esta herramienta abre las puertas a los usuarios a un mundo donde no tengan que recibir tanta información sino la que ellos realmente quieran

recibir. Es por ello que en un corto intervalo de tiempo los sistemas de recomendación se han hecho muy populares y tienen una tendencia muy importante a seguir subiendo en los próximos años.

Este proyecto nació con la idea de desarrollar un sistema de recomendación actualizable de películas para video clubs o webs de venta de películas con la intención de tener un sistema centralizado que permitiera a sus clientes dar puntuaciones a películas que fueran alquilando/comprando. En base a estas puntuaciones el sistema crearía un perfil para cada cliente, lo compararía con los perfiles de los demás clientes y ofrecería una lista de películas recomendadas.Con dicha mecánica de funcionamiento, el tipo de sistema de recomendación que mejor se adapta es, sin duda ninguna, el Sistema de Recomendación Colaborativo.

Los sistemas de recomendación colaborativos tienen como base de empleo un algoritmo de filtrado colaborativo. Existen 2 categorías dentro de estos algoritmos: los basados en usuario y los basados en item. Los primeros se caracterizan por utilizar toda la base de datos para realizar sus predicciones y recomendaciones. Los segundos por desarrollar un modelo a partir de las puntuaciones de los usuarios sobre los item.

Ante la imposibilidad de reunir un conjunto de datos reales, como base lo suficientemente amplio, hemos usado un fichero de ejemplo existente en Movilens. Este ejemplo consta de 943 usuario, 1682 películas y 100000 puntuaciones, así como algunos alquieres. Seguidamente nos dispusimos a implementar la parte servidor de este proyecto. Para este desarrollo contamos con el estudio previo realizado en el proyecto "Sistema de Recomendación Colaborativo de Alquiler de Películas"[1], sobre el mejor algoritmo de filtrado colaborativo. Esta parte del proyecto se encarga de recopilar datos de películas automáticamente de la web IMDB. Seguidamente se procede a la actualización del algoritmo de filtrado, dejando preparados los datos para realizar desde la parte cliente la mejor recomendación posible.

Por su parte,la parte cliente cuenta con una interfaz web para su comunicación con los usuarios, donde el usuario puede actualizar sus datos personales, realizar puntuaciones, alquilar peliculas, y obtener recomendaciones.

Para el desarrollo de este proyecto se han seguido todas las fases de la ingeniería del software. Primero determinamos las propiedades que debe satisfascer la aplicación y las restricciones a las que se encuentra sometido. Luego se creó un modelo de sistema correcto,completo consistente,claro y verificable.

Finalmente comentar que para el autor ha sido una gran satisfacción personal contemplar como un proyecto nacido de un idea plasmada en un papel ha sido capaz de llegar a buen puerto y sobre todo ver como mediante la implementación de unas técnicas (totalmente desconocidas para el autor) se consigue dar recomendaciones adecuadas a los usuarios. Para mi el mundo de los sistemas de apoyo a la toma de decisiones era casi desconocido, estudiado muy de pasada en la asignatura del mismo nombre que se cursa en la carrera. Por tanto el reto fue mayor aun si cabe al tener que aprender todo casi desde 0.

ANEXO A

Manual instalación del Servidor

Este manual recoge la instalación del servidor necesario para montar nuestra aplicación. En él, se detallará la instalación de los elementos necesarios como el servidor apache2,tomcat,php y mysql.

Consideraciones previas y material necesario

El servidor vamos a montarlo sobre el sistema operativo **Ubuntu Linux 8.04**. Los paquetes necesarios para su instalación serán, descargados directamente de los repositorios de ubuntu usando el gestor de paquetes synaptic.

La aplicación java se adjunta en el CD del material así como la documentación para manejarla.

A.1. Paso 1: Instalación del Servidor Apache

Apache es un servidor HTTP de código abierto y multiplataforma desarrollado por la Apache Software Foundation, en cuya web http://www.apache.org se pueden conseguir la última versión del servidor, sus múltiples módulos de desarrollo y ampliación y toda la documentación necesaria para su correcto funcionamiento. Se trata, con diferencia, del más popular de los servidores HTTP de la actualidad.

a) Si ya dispone de la versión 2.2.3 o superior de Apache: Dirigase a Paso 2.

b) Si no dispone de Apache:

Dirigase a SISTEMA—>ADMINISTRACION—>GESTOR DE PAQUETES SYNAPTIC

Seguidamente haga click sobre buscar y escriba apache2, a continuación para comenzar la búsqueda pulse buscar en el cuadro de dialogo que muestra la figura.

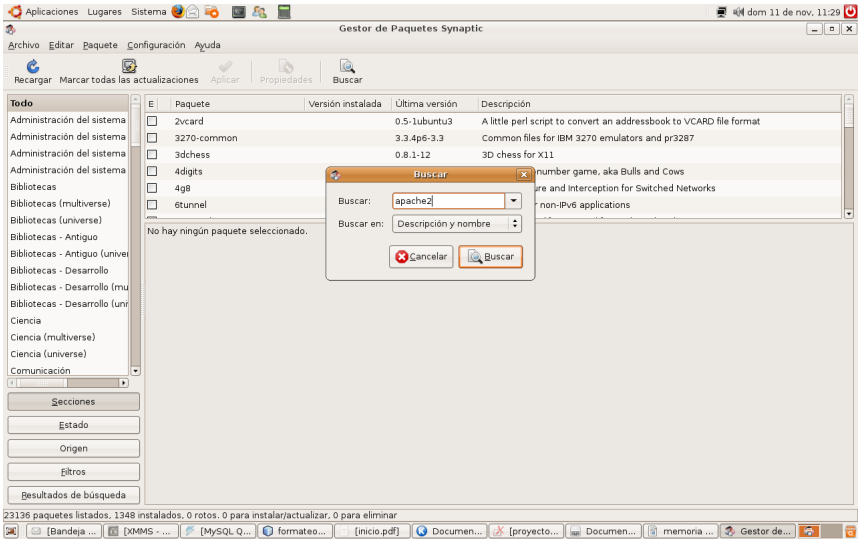


Figura A.1: Imagen del gestor de paquetes de debian-ubuntu synaptic

Tras realizar la búsqueda aparecerá una pantalla como esta

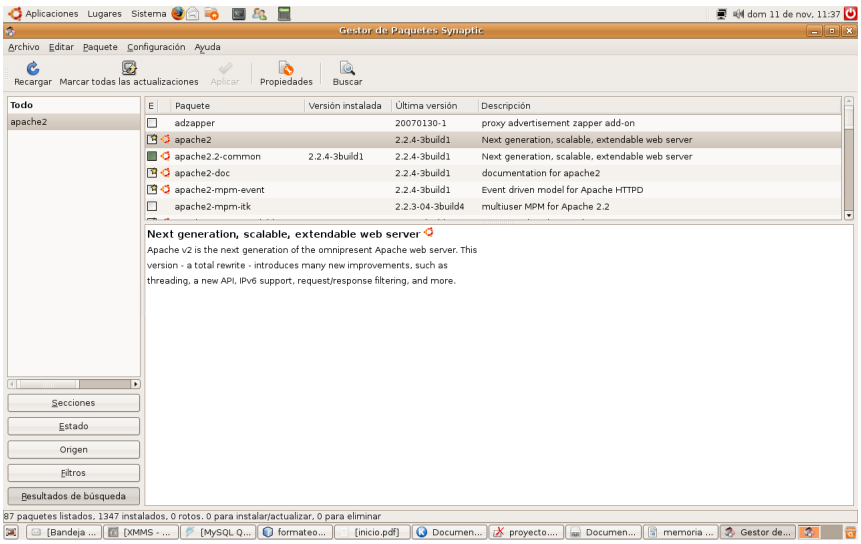


Figura A.2: Búsqueda de Apache2 en synaptic

A continuación estamos en disposición de instalar apache2. Clikcamos con el botón derecho sobre la línea de apache2 y lo marcamos para instalar, seguidamente le daremos al botón aplicar para realizar su instalación.

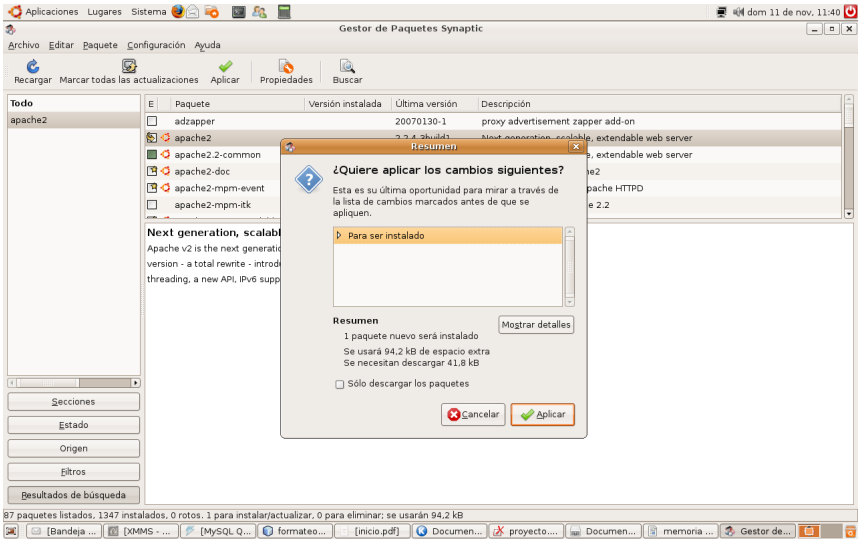


Figura A.3: Instalando Apache2

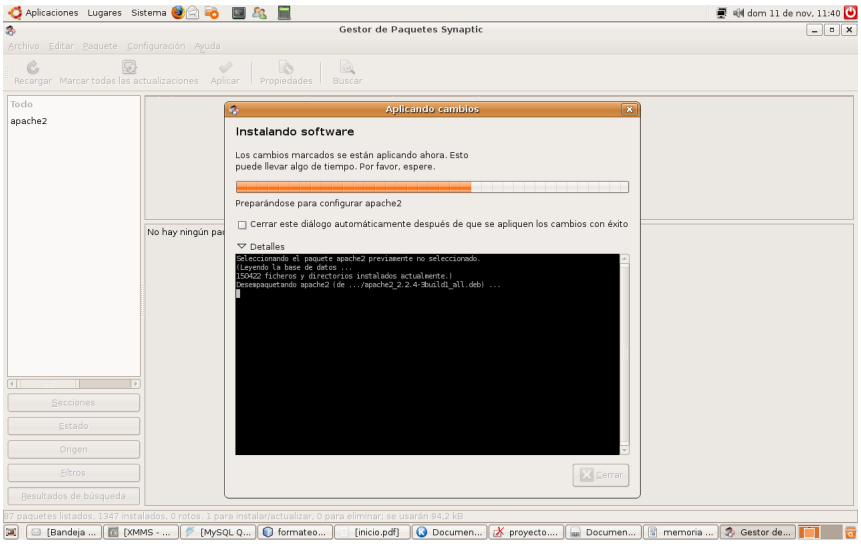


Figura A.4: Instalando Apache2

Para comprobar que el servidor se ha instalado correctamente abrimos el navegador web y escribimos la siguiente dirección

http://localhost/apache2-default/

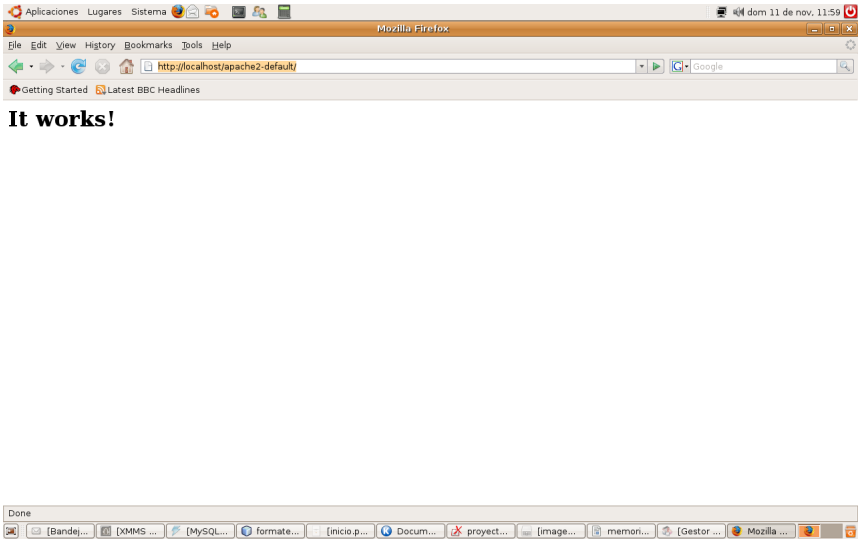


Figura A.5: Apache funcionando

A.2. Paso 2: Instalación de JAVA

Si ya dispone de máquina java instalada pase al **paso 3**.

Si no dispone de una máquina virtual de Java instalada tiene 2 opciones:

- Puede conseguirla en la dirección http://java.sun.com/javase/downloads/index.jsp Solo tiene que ejecutar el archivo y seguir los sencillos pasos de instalación para instalar las últimas versiones de no solo el kit de desarrollo Java (JDK) sino el entorno de ejecución Java (JRE) y, obviamente, la máquina virtual Java.

- En nuestro caos vamos a instalarla de los repositorios de ubuntu.

Siguiendo los mismos pasos que para instalar apache2(Paso 1), introducimos en la caja de busqueda la palabra sun-java.

De toda la lista que obtenemos debemos de seleccionar sun-java6-bin,sun-java6-jdk,sun-java6-jre,sun-java6-plugin y acontinuación le damos a aplicar.

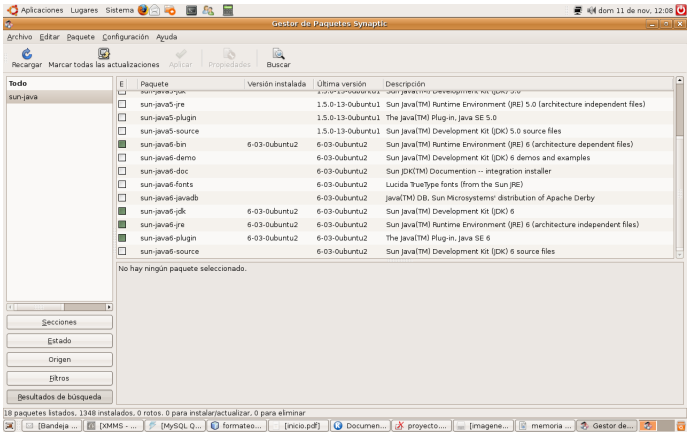


Figura A.6: Gestor de paquetes Synaptic tras la busqueda de sun-java

Para comprobar que la instalación ha sido satisfactoria, abrimos un terminal (APLICACIONES–>ACCESORIOS–>TERMINAL) y escribimos el siguiente comando

java -version

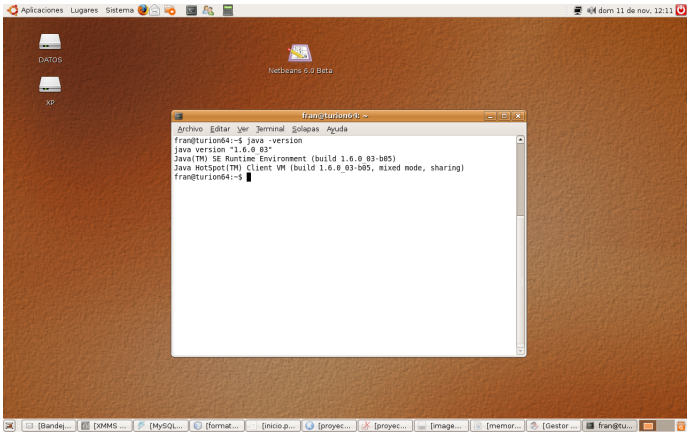


Figura A.7: Terminal de ubuntu, donde consultamos por la versión de java

A.3. Paso 3: Instalación del Sistema de Gestión de Bases de Datos MYSQL 5

Si dispone de Mysql 5 instalado pasar al **paso 4**

Siguiendo la metodología de la instalación de apache2, nos dirigimos al gestor de paquetes synaptic para proceder a la instalación de mysql.

Debemos seleccionar los paquetes mysql-server 5.0, mysql-query-browser,mysql-admin

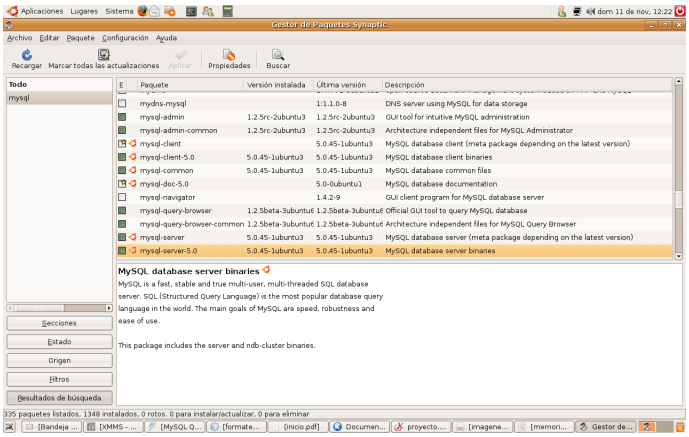


Figura A.8: Gestor de paquetes Synaptic tras la busqueda e instalación de mysql-server

Tras la instalación crearemos el usuario que nos va a servir para trabajar con la base de datos. así como la base de datos que albergará los datos de nuestra aplicación.

Para crear el usuario nos dirigimos a APLICACIONES–>PROGRAMACIÓN–>MYSQL-ADMINISTRATOR

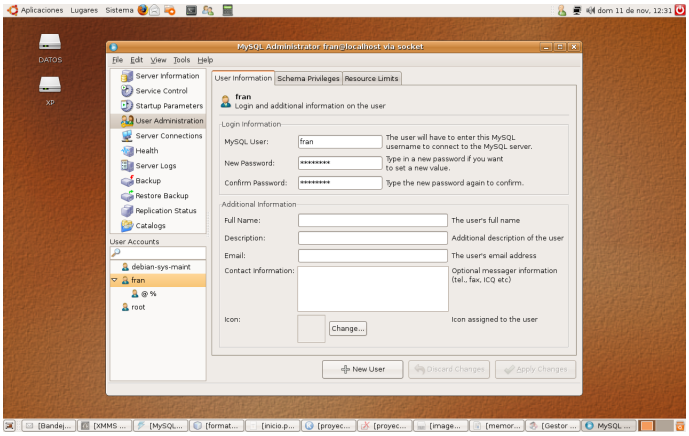


Figura A.9: Panel de administración de usuarios y permisos de mysql-server

Una vez que disponemos de usuario y tenemos permisos asignados, nos disponemos

a crear la base de datos. Para ello abriremos APLICACIONES–>PROGRAMACIÓN–>MYSQL-QUERY-BROWSER

Seguidamente ingresamos en el sistema con nuestro usuario y contraseña. A la derecha encontramos una pestaña que pone schemadata en la cual estan las bases de datos con las que cuenta el servidor. Con el botón derecho pulsamos y aparecerá un menú en el cual nos da varias opciones. La opción a tomar es create schema (Crear base de datos). El cuadro de dialogo que sale es para introducir el nombre que queremos darle. Una vez que tenemos la base de datos creada, desde la aplicación crearemos tablas y manejaremos los datos.

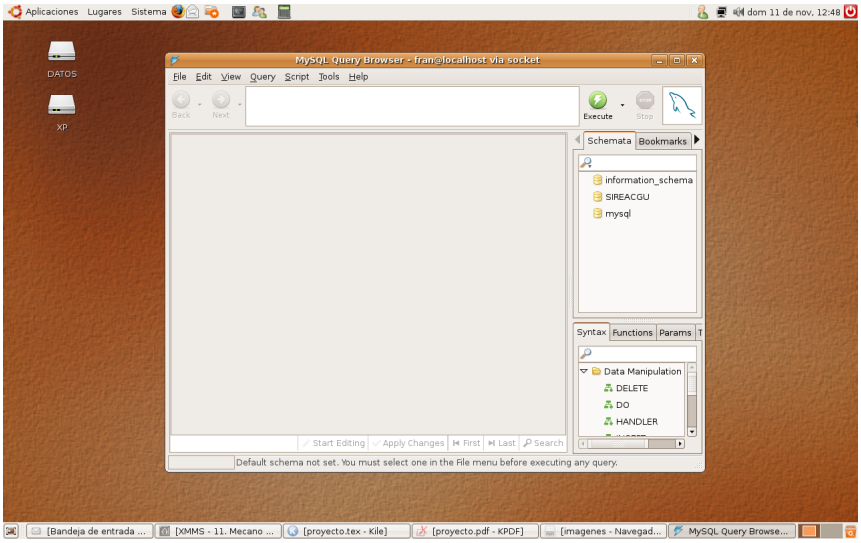


Figura A.10: Creación de la base de datos SIREACGU

A.4. Paso 4: Interconector JAVA-MYSQL

Para la conexión desde java con la base de datos mysql necesitamos lo que se llama un j-conector.

Se encuentra disponible la última versión en la página web
http://download.softagency.net/MySQL/Downloads/Connector-J/

Tras la descarga nos encontramos con un fichero comprimido en el cual el archivo

que nos hace falta viene en formato jar. Ese archivo será el que tendremos que incluir en el classhpath de nuestro proyecto para que funcione la conexión java-mysql.

A.5. Paso 5: Configuración CRON Linux

Para la ejecución automática de la aplicación nos disponemos a registrarla en el CRON, para que en la fecha y hora que indiquemos la aplicación de actualización sea lanzada.

Antes de disponernos a hacer el registro debemos de contar con la aplicación de administración a través de web llamada Webmin.

Esta aplicación(webmin) se adjunta en el cd.

Para su instalación necesitamos permisos de superusuario. Tenemos que descomprimir el fichero adjunto en el cd (webmin.tar.gz). Para ello abrimos una consola y escribimos tar -xvfz webmin.tar.gz y se nos descomprimirá en la ruta actual donde estemos. Una vez descomprimido, nos metemos dentro de la carpeta webmin recien creada por el descompresor y seguidamente ejecutamos desde consola el comando ./setup.sh. Todo lo que nos pregunte le damos a aceptar, hasta que nos pregunte el usuario y la clave de administrador, donde meteremos la que queramos.

Una vez instalado webmin para ejecutarlo abrimos un navegador con la siguiente dirección http://localhost:10000

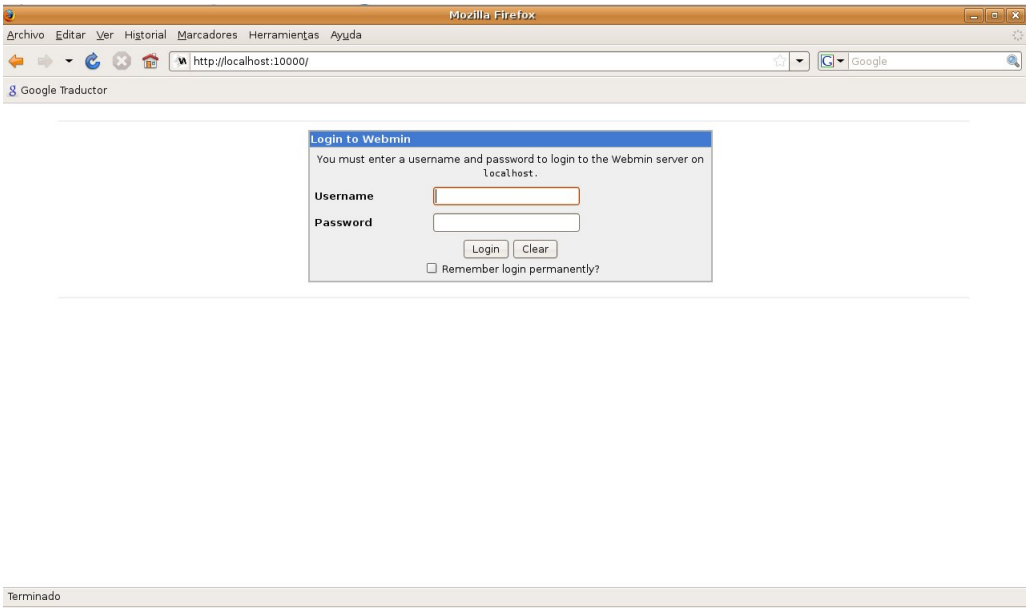


Figura A.11: Administrador web Webmin

Después de introducir nuestro usuario y contraseña, nos disponemos a buscar el acceso al cron, el cual se encuentra dentro de las utiliades de system (marcado en la figura con un punto negro) como se observa:

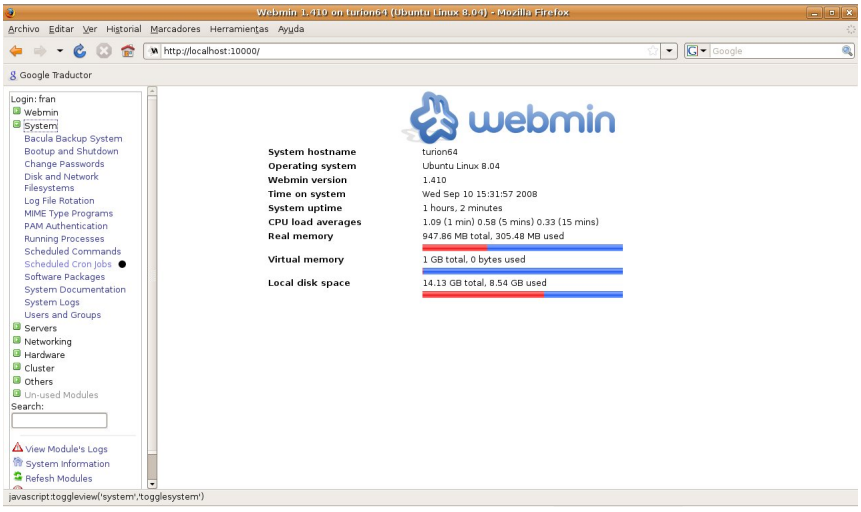


Figura A.12: Administrador web Webmin

Una vez dentro de la pantalla de Scheduled Cron Jobs marcaremos la opción en el menú superior de *create a new scheduled cron job* y aparecerá una pantalla como la siguiente:

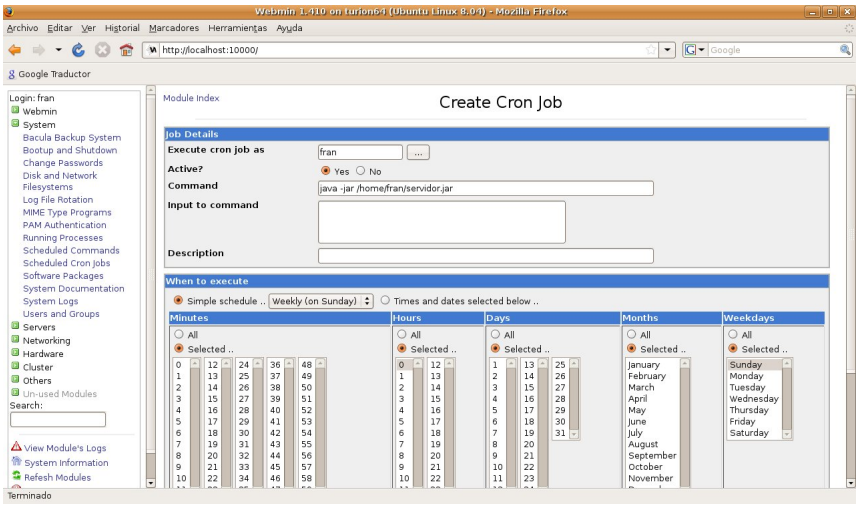


Figura A.13: Administrador web Webmin

En ella hemos puesto quien es el usuario que quiere ejecutar la orden y que comando ejecutar. También se ha planificado que sea semanalmente los domingos a las 0:00 horas. Seguidamente pulsamos el boton create que esta en la parte inferior de la pantalla.

ANEXO B

Manual de Usuario

Este manual de usuario esta organizado como una visita guiada por la aplicación pero antes de embarcarse en ella es conveniente que el usuario tenga claros algunos aspectos:

SIREACGU es una aplicación web optimizada para su visualización en un navegador Firefox (a ser posible su versión más reciente, la cual se puede descargar en http://www.mozilla-europe.org/es/) y con una resolución no inferior a 1024x768 pixels. Si se utiliza otro navegador o una resolución inferior a la recomendada se pueden producir fallos de visualización aunque la funcionalidad de la aplicación esta completamente asegurada.

Inicialmente el usuario de SIREACGU accederá a la dirección (http://turion64/index.php o http://localhost/index.php) página de inicio de la aplicación como la que se puede observar en la figuraB.1:

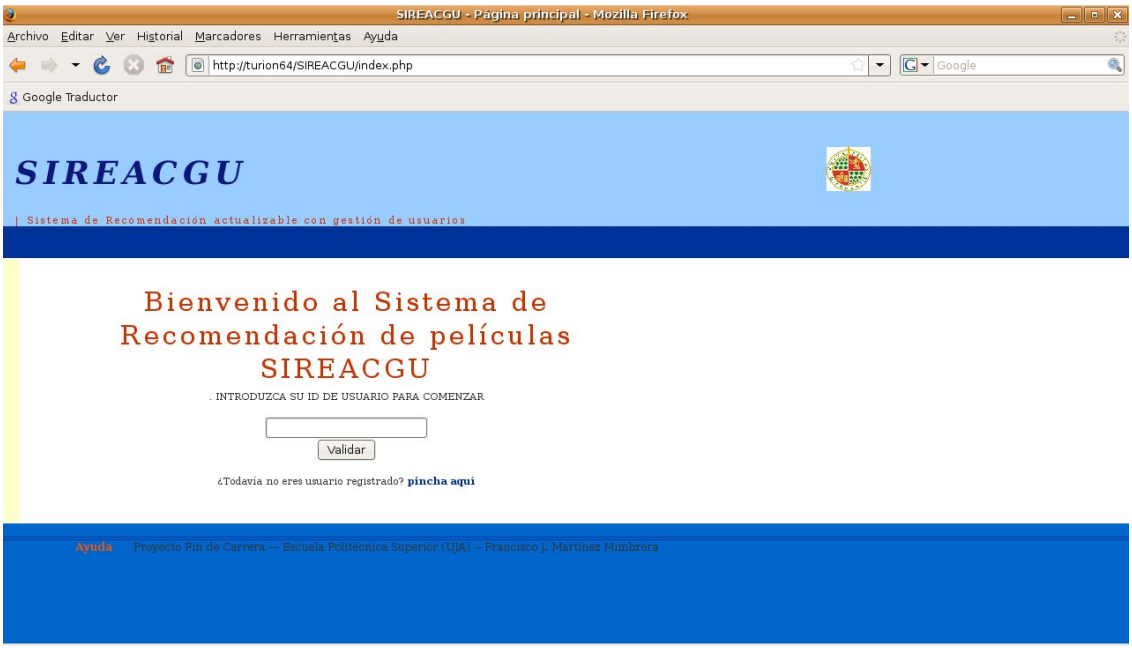


Figura B.1: Página de inicio de SIREACGU

El usuario deberá introducir su identificador de usuario y pulsar el botón login. Llegados a este punto pueden ocurrir 2 cosas. Si el ID introducido es correcto el sistema nos mostraría la página de bienvenida, figura B.2 :

Figura B.2: Página de bienvenida de SIREACGU

Si el ID introducido no es correcto nos devolverá a una página de error para volver a introducir el ID. figura B.3

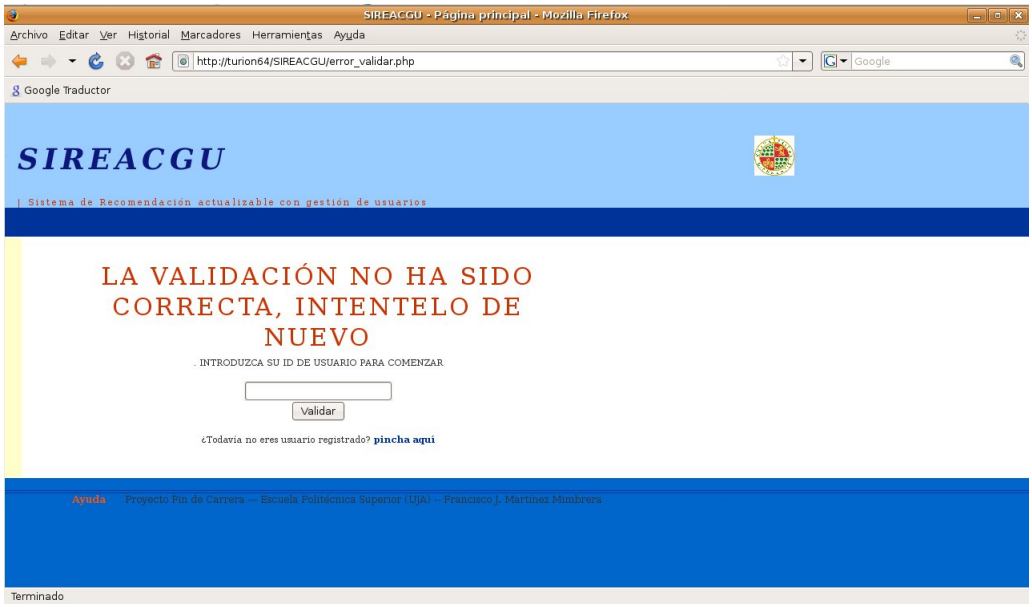


Figura B.3: Página de error de validación de SIREACGU

Si no se dispone de ID de usuario, el sistema permite registrarse en tiempo de ejecución como usuario del sistema como se muestra en la figuraB.4.

Figura B.4: Acceso a registro de usuario SIREACGU

Para registrarse es tan fácil como rellenar los campos que se muestran a continuaciónB.5 y hacer click a guardar. Al guardar aparecerá una pantalla indicando cual es su nuevo ID de usuario. Si por el contrario desea volver a la página anterior pulse el botón de cancelar.

Figura B.5: Registro de usuario SIREACGU

Una vez correctamente validados en el sistema nos disponemos a utilizarlo. Tras pasar la validación nos encontramos en la pantalla de bienvenida, en la cual se nos ofrecen 3 posibilidades distintas B.6.

Datos Personales:

Permite acceder a los datos personales del usuario registrado y modificarlos.

Ver Películas:

Nos da acceso al menú de gestión de películas, donde podremos ver las últimas novedades así como puntuar películas, alquilarlas, realizar búsquedas y obtener recomendaciones.

Salir:

Nos permite salir del sistema con el Id de usuario actual.

Figura B.6: Página de bienvenida de SIREACGU

Si elegimos la opción de datos personales, se nos mostrará la figuraB.7:

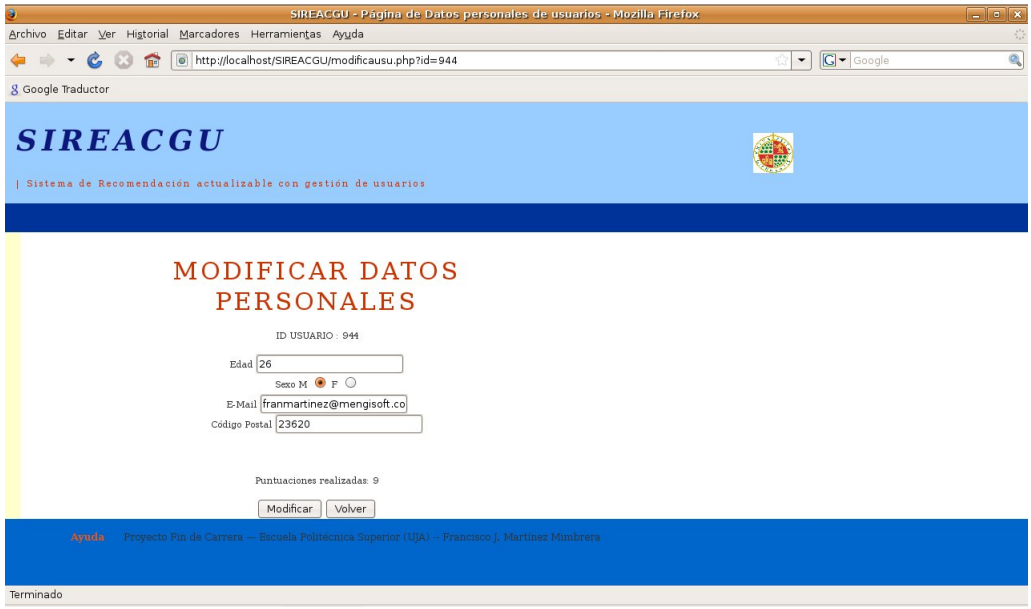


Figura B.7: Página de Modificación de usuarios SIREACGU

En ella podremos modificar los datos o simplemente visualizarlos y dejarlos intactos.

Para volver a la página de inicio sin realizar cambios pulsaremos sobre el botón VOLVER.

La opción VER PELÍCULAS nos lleva a la pantalla mostrada en la figuraB.8 donde se va a realizar la activadad principal con el sistema SIREACGU. Lo primero que se nos muestran son las novedades (películas recien incorporadas) del sistema.

Además se nos ofrecen un amplio abanico de posibilidades que iremos desgranando a continuación.

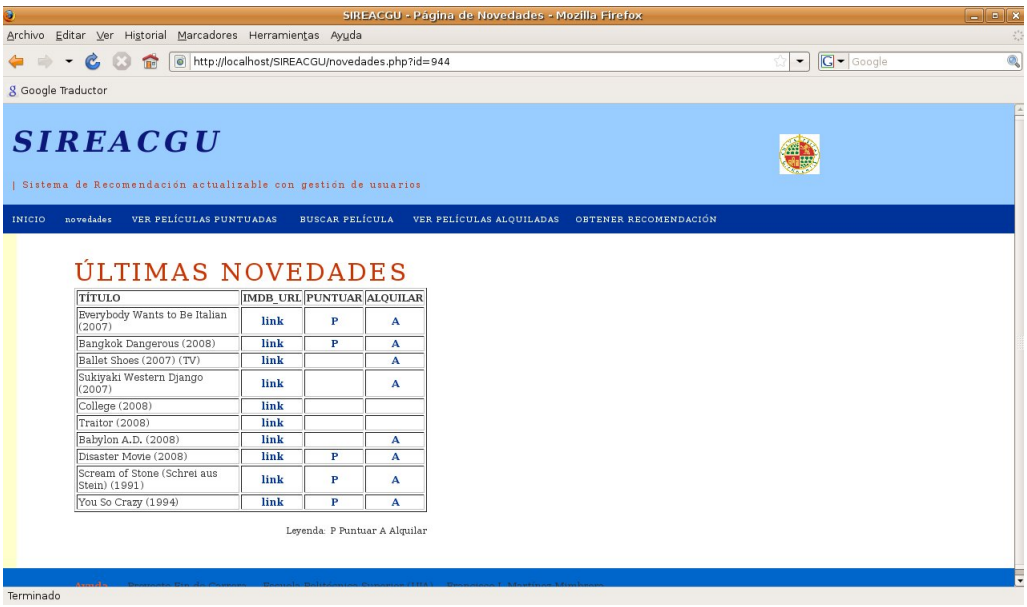


Figura B.8: Página de Novedades SIREACGU

Como se observa en la figuraB.9 el sistema nos muestra el título de las películas así como un enlace a la información referente a ella que se encuentra la página web imdb. También se nos permite puntuar la película si ya la hemos visto y alquilarla. Si una película ya ha sido puntuada o alquilada, el sistema dejará ese hueco en blanco para que no se pueda repetir la operación.

Figura B.9: Detalle de película SIREACGU

Para realizar una puntuación de una película es tan sencillo como cada vez que se nos permita hacer click sobre la P situada justamente al lado de la película pulsar y nos llevará la pantalla que se muestra a continuación. En ella mediante radio button podemos elegir la puntuación que deseamos darle, además, podremos ver más detalles acerca de la película desde la propia página de imdb.

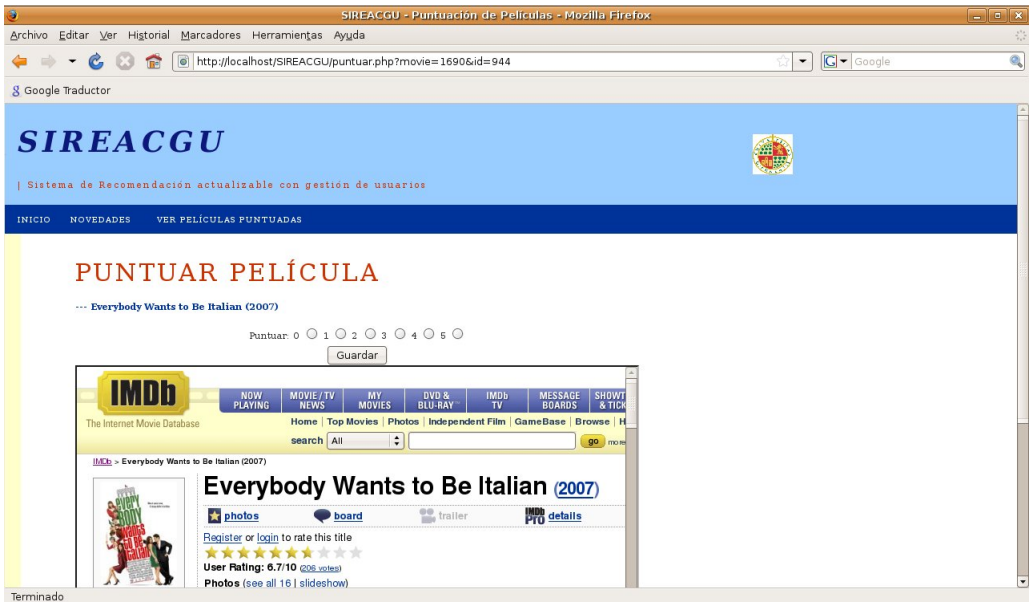
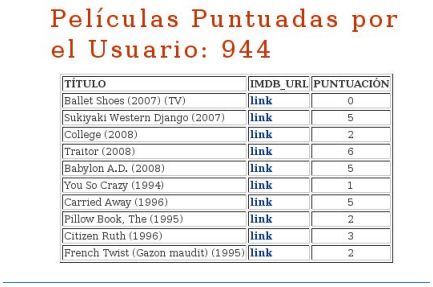


Figura B.10: Puntuar película SIREACGU

Al realizar la puntuación el sistema nos lleva automáticamente a la pantalla mostrada en la figuraB.11 de VER PELÍCULAS PUNTUADAS.

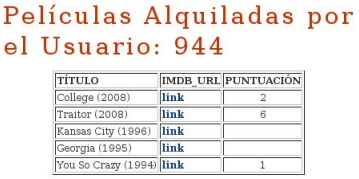


Películas Puntuadas por el Usuario: 944

TÍTULO	IMDB_URL	PUNTUACIÓN
Ballet Shoes (2007) (TV)	link	0
Sukiyaki Western Django (2007)	link	5
College (2008)	link	2
Traitor (2008)	link	6
Babylon A.D. (2008)	link	5
You So Crazy (1994)	link	1
Carried Away (1996)	link	5
Pillow Book, The (1995)	link	2
Citizen Ruth (1996)	link	3
French Twist (Gazon maudit) (1995)	link	2

Figura B.11: Películas Puntuadas por el usuario SIREACGU

Para realizar un alquiler basta con pulsar sobre la letra A que acompaña al título de la película. Al realizar el alquiler automáticamente nos llevará a la pantalla de VER PELÍCULAS ALQUILADAS representada por la figuraB.12.



Películas Alquiladas por el Usuario: 944

TÍTULO	IMDB_URL	PUNTUACIÓN
College (2008)	link	2
Traitor (2008)	link	6
Kansas City (1996)	link	
Georgia (1995)	link	
You So Crazy (1994)	link	1

Figura B.12: Películas Alquiladas por el usuario SIREACGU

La siguiente sección con la que nos encontramos es con el buscador de películas. Podemos hacer busquedas tanto por título como por año de lanzamiento mostramos unos ejemplos

Figura B.13: Buscador de Películas SIREACGU

Ejemplo de búsqueda figuraB.14: Queremos obtener la película de dibujos animados Toy Story. La vamos a buscar por su título. Introducimos pues la palabra toy en la caja de texto título. El buscador hacer tanto busquedas por delante como por detras de la palabra solicitada, para encontrar coincidencias.

Figura B.14: Buscador de Películas por título SIREACGU

Por ejemplo si ahora realizamos una búsqueda por año, nos saldrán todos los títulos de películas editados durante ese año disponible en nuestra base de datos. Ejemplo año 2008.

Figura B.15: Buscador de Películas por año SIREACGU

Por último nos queda comentar la opción de RECOMENDACIÓN. El sistema nos recomendará las mejores películas que se adecuan a nuestro perfil.

Figura B.16: Recomendador de Películas SIREACGU

La opción de ayuda de nuestro sistema simpre permanecerá activa desde el primer momento en la barra de tareas de abajo, para en cualquier momento el usuario pueda consultar como ejecutar una acción. Deberá de dispone de un lector de pdfs

.

Figura B.17: AYUDA SIREACGU

ANEXO C

Código fuente del módulo de extracción de información de peliculas IMDB

.

Contenido del fichero Formateo.java:

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package servidor;

import java.net.*;
import java.io.*;
import java.text.*;
import java.util.*;
import java.sql.*;
/**
 *
 * author Francisco Jesús Martínez Mimbrera
*/
public class ExtraerInformacion {

   ManejaBd basedatos;

    public ExtraerInformacion(ManejaBd b){
        try{
        //Runtime.getRuntime().exec("");
        basedatos=b;
        leeURL("http://www.imdb.com/nowplaying/","/home/fran/principal.txt");
        }catch (Exception ex) {
            System.err.println("No puedo extraer la información");
```

```
            return;
        }
        parserprincipal("/home/fran/principal.txt");

    }
    /*****************************************************************/
    /**Procedimiento para leer el codigo fuente de una dirección url**/
    /*****************************************************************/
    void leeURL(String urlCompleta, String destino)throws Exception{
        URL url = new URL (urlCompleta);

        URLConnection urlc = url.openConnection();
        InputStream is = urlc.getInputStream();
        FileOutputStream os = new FileOutputStream(destino);
        byte[] buf = new byte[16384];
        int c;
        while ((c=is.read (buf))>0)
        {
        os.write(buf, 0, c);
        }
        os.close(); is.close();

    }
    /*****************************************************************/
    /**Procedimiento para dado un fichero de texto extraer informacion**/
    /**y guardar en Base de datos dicha información sobre películas    **/
    /*****************************************************************/
    void parserindividual(String nombrefichero,String direurl){
```

```
BufferedReader entrada;
String titulo;
String fecha;

boolean bdes=false, bac=false, bav=false, bani=false, binf=false, bcom=false;
boolean bcri=false, bdoc=false, bdra=false, bfan=false, bne=false, bte=false;
boolean bmu=false, bmis=false, bro=false, bcfi=false, bthri=false, bgu=false, bwes=f

try {
    FileReader fr = new FileReader(nombrefichero);
    entrada = new BufferedReader(fr);
    String cadenaleida=new String();
    StringTokenizer elementos;
    String word;
    titulo= new String();
    String genero;

    fecha = "2008-1-1";
    int numero = 0;
    while((cadenaleida = entrada.readLine()) != null){
        numero++;
        if (cadenaleida.length() > 14){

            if (cadenaleida.substring(0, 7).compareTo("<title>")==0){
                titulo = cadenaleida.substring(7,cadenaleida.length()-8);
            }
            if (cadenaleida.contains("<h5>Release Date:</h5>")==true){
```

```
        cadenaleida=entrada.readLine();
        fecha= formateoFecha(cadenaleida);
    }
    if (cadenaleida.contains("<h5>Genre:</h5>")==true){
        cadenaleida=entrada.readLine();
        elementos = new StringTokenizer(cadenaleida,"<>");
        while(elementos.hasMoreTokens()){
          word = elementos.nextToken();

          if (word.contains("a href=")==true){
              genero=word.substring(25, word.length()-2);

              if (genero.contains("Horror")==true){
                    bte=true;
              }
              if (genero.contains("Mystery")==true){
                    bmis=true;
              }
              if (genero.contains("Thriller")==true){
                    bthri=true;
              }
              if (genero.contains("Drama")==true){
                    bdra=true;
              }
              if (genero.contains("Action")==true){
                    bac=true;
              }
              if (genero.contains("Adventure")==true){
                    bav=true;
```

```
}
if (genero.contains("Animation")==true){
        bani=true;
}
if (genero.contains("Comedy")==true){
        bcom=true;
}
if (genero.contains("Crime")==true){
        bcri=true;
}
if (genero.contains("Documentary")==true){
        bdoc=true;
}
if (genero.contains("Romance")==true){
        bro=true;
}
if (genero.contains("War")==true){
        bgu=true;
}
if (genero.contains("Western")==true){
        bwes=true;
}
if (genero.contains("Fantasy")==true){
        bfan=true;
}
if (genero.contains("Musical")==true){
        bmu=true;
}
if (genero.contains("Family")==true){
```

```
                        binf=true;
                    }
                    if (genero.contains("Sci-Fi")==true){
                        bcfi=true;
                    }
                     if (genero.contains("Film-Noir")==true){
                        bne=true;
                    }
                  }
                   genero=""; //Vacio la variable
                }

              }


          }


      }
```

```
        String res = "INSERT INTO PELICULAS (TITULO,FECHA,IMDB_URL,DESCONOCIDO,ACCION,
AVENTURAS,ANIMACION,INFANTIL,COMEDIA,CRIMEN,DOCUMENTAL,DRAMA,FANTASIA,
NEGRO,TERROR,MUSICAL,MISTERIO,ROMANTICO,CIENCIA_FICCION,THRILLER,GUERRA,WESTERN)" +
                "VALUES('"+titulo+"','"+fecha+"','"+direurl+"',"+
                bdes+","+bac+","+bav+","+bani+","+
                binf+","+bcom+","+bcri+","+bdoc+","+
                bdra+","+bfan+","+bne+","+bte+","+bmu+","+
                bmis+","+bro+","+bcfi+","+bthri+","+
                bgu+","+bwes+")";
```

```
                basedatos.conectaBd();
                basedatos.insertaBd(res);
                basedatos.cierraConexionBd();

        entrada.close();
                                                                    180
    }
    catch(java.io.FileNotFoundException fnfex) {
    }
    catch(java.io.IOException ioex) {
    }


}
    /*********************************************************************/
    /**Procedimiento para dado un fichero de texto extraer informacion**/    190
    /**y guardar en ficheros individuales el codigo fuente de cada url de
     **película                                                    **/
    /*********************************************************************/
    void parserprincipal(String nombrefichero){

    BufferedReader entrada;
    String palabrucha=new String();

    try {
        FileReader fr = new FileReader(nombrefichero);                  200
        entrada = new BufferedReader(fr);
```

```
String cadenaleida=new String();
StringTokenizer elementos;
String word;
int fila=0;


while((cadenaleida = entrada.readLine()) != null ){


                                                                    210
    elementos = new StringTokenizer(cadenaleida,"<>");
    while(elementos.hasMoreTokens()){
    word = elementos.nextToken();
  if (word.contains("a name=\"topten\"")){
    return;
  }

if ((word.contains("td class=\"movie even\" align=center")==true)
  || (word.contains("td class=\"movie \" align=center")==true)) {
   word = elementos.nextToken();                                    220
   if (word.contains("a href")==true){
     palabrucha=new String();
      palabrucha = "/home/fran/"+word.substring(15,24)+".txt" ;
       System.out.print(palabrucha);
      try{
        leeURL("http://www.imdb.com/"+word.substring(9,24),palabrucha);
       }catch (Exception ex) {
       System.err.println("No puedo extraer la información");
       return;
       }                                                            230
```

```
                parserindividual(palabrucha,"http://www.imdb.com/"+word.substring(9,24));
              }
            }
        }
        }



            entrada.close();
        }
        catch(java.io.FileNotFoundException fnfex) {
        }
        catch(java.io.IOException ioex) {
        }


    }

    private String formateoFecha(String s) {
            String smes="00", res;

            StringTokenizer elementoscadena;
            elementoscadena= new StringTokenizer(s);


               String sdia = elementoscadena.nextToken();
               String messucio=elementoscadena.nextToken();
               String sanio = elementoscadena.nextToken();



            if ((messucio.compareTo("January"))==0) smes="01";
```

```
        if ((messucio.compareTo("February"))==0) smes="02";
        if ((messucio.compareTo("March"))==0) smes="03";
        if ((messucio.compareTo("April"))==0) smes="04";
        if ((messucio.compareTo("May"))==0) smes="05";
        if ((messucio.compareTo("June"))==0) smes="06";
        if ((messucio.compareTo("July"))==0) smes="07";
        if ((messucio.compareTo("August"))==0) smes="08";
        if ((messucio.compareTo("September"))==0) smes="09";
        if ((messucio.compareTo("October"))==0) smes="10";
        if ((messucio.compareTo("November"))==0) smes="11";
        if ((messucio.compareTo("December"))==0) smes="12";



        res =sanio+"/"+smes+"/"+sdia;
        return res;
    }

}
```

ANEXO D

Código fuente del módulo de cálculo del Knn

.

Contenido del fichero Formateo.java:

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package servidor;
import java.util.*;
import java.io.*;
import java.sql.*;
/**
 *
 * author Francisco Jesús Martínez Mimbrera
 */
public class Knn {

    ManejaBd basedatos;
    Matrix m;
    int med;
    Vector m80;
    registro[][] arrknn;


    public Knn(int nfilas,int ncolumnas,ManejaBd b){


        m = new Matrix(nfilas,ncolumnas,b);
                System.out.println("Matrix Terminada");
```

```
        m80 = calConjunto80(m);
        basedatos=b;

                                                                    30
        calcularKnn(m,m80);

    }

    public Vector calConjunto80(Matrix m) {
            System.out.println("Empezando matrix80%");
        int aux,i=0;
        int filas80;
        Random r;
        Vector v;                                                   40
        v = new Vector();

        filas80=Math.round(m.numFilas() * Float.parseFloat("0.80"));
        v.setSize(filas80);
        int x=v.size();
        //elegimos aleatoriamente los integrantes del conjunto
        r = new Random();

        for (i=0;i<x;i++){

                                                                    50
            aux = r.nextInt(m.numFilas());
            //Comprabamos que ningun elemento sea repetido
            while (v.contains(aux)==true){
                aux = r.nextInt(m.numFilas());
            }
            v.add(aux);
```

```
        }

         System.out.println("Ya he terminado el matriz80%");
        return v;
    }

     public void calcularKnn(Matrix m, Vector m80) {
        float sim;
        ArrayList cola;
        ArrayList colb;
        ArrayList vecinos;
        //El tamaño es 20 porque hemos elegido k-vecinos =20

         System.out.println("Empezando knn");
        for (int i=0;i<m.numColumnas();i++){
            //obtengo la primera columna
            cola = new ArrayList();
            for(int fila=0;fila<m.numFilas();fila++) {
                 if (m80.contains(fila)==true){
                      cola.add(m.getCell(fila, i));
                 }
            }
            //obtengo las demas columna
            vecinos = new ArrayList();
            for(int aux=0;aux<m.numColumnas();aux++) {
                colb = new ArrayList();
                if (aux!=i) {
                    for(int fila=0;fila<m.numFilas();fila++) {
                        if (m80.contains(fila)==true){
```

```
                        colb.add(m.getCell(fila, aux));
                    }
                }
                //calculamos similaridad
                sim = aplicarMedida(cola, colb);
                //calculamos si entra en la k-vecindad
                registro raux = new registro(aux+1, sim);
                vecinos = comprobarVecinos(vecinos, raux);
            }
        }
          System.out.println("Ya he terminado knn");
        //introducimos en la matriz de resultados
        basedatos.conectaBd();

        for (int aux2=0;aux2<20;aux2++) {


            registro a;
            a=(registro)vecinos.get(aux2);

            int peli= a.leeIndex();
            float numero= a.leeValor();
            int k=i+1;
 basedatos.insertaBd("INSERT INTO KNN(PELICULA,RESULTADO,K)
VALUES("+peli+","+numero+","+k+")");
              System.out.println("k: "+k +" Peli: " +peli+ " Resul: "+numero);
        }
        basedatos.cierraConexionBd();
```

```
    }


}

/** Metodo que aplica la medida de similaridad requerida a los dos dos
 * conjuntos dados */
public float aplicarMedida(ArrayList a, ArrayList b) {
    double f = 0, f1 = 0, f2 = 0, aux;
    float sim=0;

    //Calculamos el coeficiente del coseno
    if (a.size()!=b.size()) {
        System.out.println("Error");
    }
    else {
        for(int i=0;i<a.size();i++) {
            if ((Integer)a.get(i)!=0 && (Integer)b.get(i)!=0) {
                aux = (Integer)a.get(i) * (Integer)b.get(i);
                f = f + aux;
            }
        }
        for(int i=0;i<a.size();i++) {
            if ((Integer)a.get(i)!=0) {
                aux = Math.pow((Integer)a.get(i), 2);
                f1 = f1 + aux;
            }
        }
        for(int i=0;i<b.size();i++) {
```

```
                if ((Integer)b.get(i)!=0) {
                    aux = Math.pow((Integer)b.get(i), 2);
                    f2 = f2 + aux;
                }
            }
            if (f1==0 || f2==0){
                f = 0;
            }
            else{
                f = f / Math.sqrt(f1 * f2);
            }

            sim = (float)f;
        }


        return sim;
    }

    /** Metodo que devuelve la matriz de resultados del entrenamiento */
    public registro[][] getKnn() {
        return arrknn;
    }

    /** Metodo que comprueba que el registro dado pertenece a la k-vecindad */
    private ArrayList comprobarVecinos(ArrayList a, registro r) {
        //si a no esta llena, insertamos al final y ordenamos
        if (a.size()<20) {
            a.add(r);
```

```
            burbuja(a);
        }

        //si a esta llena, comprobamos la similaridad y ordenamos
        else {
            registro tmp = (registro)a.get(20-1);
            if (r.leeValor()>=tmp.leeValor()) {
                a.set(20-1, r);
                burbuja(a);
            }
        }

        return a;
    }

    /* ordenamos mediante el metodo de la burbuja el array
     * de los vecinos mas cercanos */
    private void burbuja(ArrayList a) {
        int i, j;
        registro r1, r2, tmp;

        for(j=a.size()-1; j>0; j--){
            for (i=0; i<j; i++) {
                r1 = (registro)a.get(i);
                r2 = (registro)a.get(i+1);
                if (r1.leeValor()<r2.leeValor()) {
                    //intercambio
                    tmp = (registro)r1;
                    a.set(i, r2);
```

```
                    a.set(j, tmp);
                }
            }
        }

    }

}
```

ANEXO E

Código fuente del algoritmo de predicción weighted sum

.

```
<?
 include "conexion.php";
$iduser=$_GET['id'];
$idcilloo=$iduser;

        $resulot = mysql_query("SELECT * FROM PELICULAS");
        $num_peliculas = mysql_num_rows($resulot);

        $resulot = mysql_query("SELECT * FROM USUARIOS");
        $num_user = mysql_num_rows($resulot);                    10



        //volcamos los resultados de knn ha memoriaM^

        $resulot = mysql_query("SELECT * FROM KNN");
        $i=0;
        while (($row = mysql_fetch_row($resulot)) ){

                $knn_array_movies[$i]=$row[0];
                $knn_array_resultado[$i]=$row[1];                20
                $i++;


        }
        M^

        //volcamos la tabla PUNTUACIONES en memoria

        $resulot = mysql_query("SELECT * FROM PUNTUACIONES");
```

```
$i=0;
$j=0;
for ($i=0;$i<$num_user;$i++){
  for ($j=0;$j<$num_peliculas;$j++){
        $punt_array[$i][$j] = 0;
  }
}

while (($row = mysql_fetch_row($resu\ellt)) ){

      $punt_array[$row[0]-1][$row[1]-1]=$row[2];
}

M̂
//obtenemos las pelis ya votadas por el usuario      M̂
$i = 0;
for($i=0;$i<$num_peliculas;$i++){
  if ($punt_array[$iduser-1][$i] != 0)
      $aux_array[] = $i+1;

}M̂

for($i=1;$i<$num_peliculas;$i++){

      $produc_total = 0;M̂
      $sim_total = 0;M̂
      $encontrado = false;
      //nos aseguramos de que la peli i no haya sido votada
      M̂
```

```
$j = 0;M̂
            while (($j < count($aux_array)) && ($encontrado==false)) {M̂
            if ($i == $aux_array[$j]) $encontrado = true;M̂
            $j++;M̂
        }
            if ($encontrado == false) {
            //empezamos a predecirM̂
            $jau = 20 * ($i - 1);M̂
            for($ℓ=0;$ℓ<20;$ℓ++){
        $item = $knn_array_movies[$jau];
        $sim= $knn_array_resultado[$jau];
        $punt = $punt_array[$iduser−1][$item−1];
        if ($punt != 0) {M̂
                        //dividendoM̂
            $produc_total = $produc_total + ($sim * $punt);M̂
            //divisorM̂
            if ($sim < 0) $sim = $sim * (−1);M̂
                $sim_total = $sim_total + $sim;M̂
             }M̂
                                    M̂
            $jau++;
        }
        //obtenemos la predicción basada en item
        if ($sim_total == 0) $pbi = 0;
        else $pbi = $produc_total / $sim_total;
            $pred_array[$i] = $pbi;      M̂

        }
```

```
}

//ordenamos las predicciones y obtenemos las 10 mejores
arsort($pred_array);
foreach ($pred_array as $key => $val) {
        $topten_array[] = $key;
}
$m = 0;
   echo " <table border=\"1\" align=\"center\"> ";

        echo "<tr>";
        echo "<td> <b>TÍTULO </b></td>";
        echo "<td> <b>FECHA </b></td>";
        echo "<td> <b>URL IMDB </b></td>";
        echo "<td> <b>PUNTUAR </b></td>";
        echo "<td> <b>ALQUILAR </b></td>";
        echo "</tr>";
for($m=0;$m<10;$m++){
        $movie = $topten_array[$m];

$result3 = mysql_query("SELECT * FROM PELICULAS WHERE ID_MOVIE='".$movie."' ");

        while (($row = mysql_fetch_row($result3)) ){
                echo " <tr> ";
                echo "<td> $row[1] </td>";
                echo "<td> $row[2] </td>";
                echo "<td> <a href=\"$row[3]\" target=\"_blank\">link</a> </td>";
```

```
$idmovie=$row[0];
$esta=0;
$esta2=0;
 $resuℓt2 = mysql_query("SELECT * FROM PUNTUACIONES
WHERE ID_USER='".$idciℓℓo."' AND ID_MOVIE='".$idmovie."' ");

while ($row1 = mysql_fetch_row($resuℓt2)){
$esta=1;
}

$resuℓt3 = mysql_query("SELECT * FROM ALQUILADAS
WHERE ID_USER='".$idciℓℓo."' AND ID_MOVIE='".$idmovie."' ");

while ($row2 = mysql_fetch_row($resuℓt3)){
$esta2=1;
}
        if ($esta==1 and $esta2==1){
                echo " <td> ";
                echo "  ";
                echo " </td> ";
                echo " <td> ";
                echo " ";
                echo " </td> ";
        }
```

```
if ($esta==0 and $esta2==0){
        echo " <td align=\"center\"> ";
        echo " <a href= puntuar.php?movie=$idmovie&id=$idcillo>P</a> ";
        echo " </td> ";
        echo " <td align=\"center\"> ";
        echo "<a href= alquilar.php?movie=$idmovie&id=$idcillo>A</a> ";
        echo " </td> ";
}
else{

        if ($esta2==0){

                echo " <td> ";
                echo " ";
                echo " </td> ";
                echo " <td align=\"center\"> ";
                echo " <a href= alquilar.php?movie=$idmovie&id=$idcillo>A</a> ";
                echo " </td> ";
        }
        else{
                if ($esta==0){

                        echo " <td align=\"center\"> ";
                        echo " <a href= puntuar.php?movie=$idmovie&id=$idcillo>P</a> ";
                         echo " </td> ";
echo " <td> ";
                         echo " ";
```

```
                                echo " </td> ";
                            }
                        }
                    }

                                                                    180

                                echo "</tr>";

                    }M̂

            }M̂
            M̂                                                      190
            echo "</table>";

    ?>
```

Bibliografía

[1] Siles Fernández, F.(2007), *"Sistema de Recomendación colaborativo de alquiler de películas"* Proyecto Fin de Carrera . Escuela Politécnica Superior (Universidad de Jaén).

[2] Albín Rodríguez, Pedro A.(2007), *"Sistema de Recomendación colaborativo basado en algoritmos de filtrado mejorados"* Proyecto Fin de Carrera . Escuela Politécnica Superior (Universidad de Jaén).

[3] Karypis, G. (2001), *"Evaluation of item-based top-N recommendation algorithms"* en Proceedings of ACM 10th International Conference on Information and Knowledge Management, Atlanta, Georgia.

[4] Breese, J. S., Heckerman, D. y Kadie, C. (1998), *"Empirical analysis of predictive algorithms for collaborative filtering"* en Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, Madison, Wisconsin, USA.

[5] Herlocker, J., Konstan, J., Terveen, L. y Riedl, J. (2004), *"Evaluating Collaborative Filtering Recommender Systems"* en ACM Vol. Transactions on Information Systems, vol. 22, pp. 5-53.

[6] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. y Sartin, M. (1999), *"Combining content-based and collaborative filters in an online newspaper"* en Proceedings of ACM SIGIR'99 Workshp on Recommender Systems: Algorithms and Evaluation, Berkeley, CA.

[7] Dai, H. y Mobasher, B., *"Integrated semantic knowledge with web usage mining and personalization"* en Web Mining: Aplications and Techniques,Ed. IRM Press

[8] Balabanovic, M. y Shoham, Y. (1997), *"Content-based, collaborative recomendation"* en Comunications of ACM 40.

[9] Papagelis, M., Plexousakis, D., Rousadis, I. y Theorapoulos, E. (2004), *"Qualitative Analysis of User-based and Item-based Prediction Algorithms for Recommendation Systems"*, Proceedings of the 3rd Hellenic Data Management Symposium.

[10] Guo, X. (2006), *Personalized Government Online Services with Recommendation Techniques*, tesis Phd, University Graduate School, University of Technology Sydney.

[11] Shardanan, U. y Maes, P. (1995), *"Social information filtering: algorithms for automating 'word of mouth'"*, en Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems.

[12] Sarwar, B., Konstan, J., Riedl, J., Borchers, A., Herlocker, J. y Miller, B. (1998), *"Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system"* en Proceedings of the 1998 ACM Conference on Computer Support Cooperative Work, Seattle, Washington, USA.

[13] Cho, Y. H., Kim, J. K. Y Kim, S. H. (2002), *"A personalized recommender system based on web usage mining and decision tree induction"* en Expert Systems with Applications, vol. 23, pp. 233-342.

[14] Dawson, C. W. y Martin, G. (2002), *El Proyecto Fin de Carrera en Ingeniería Informática: Una guía para el Estudiante*, Prentice Hall, Madrid.

[15] Sarwar, B., Konstan, J., Terveen, L. y Riedl, J. (2000), *"Analysis of recommendation algorithms for e-commerce"* en Proceedings of the 2nd ACM Conference on Electronic Commerce, Minneapolis, Minnessota, USA.

[16] W.Clay Richardson,Donald Avondolio... (2007) *Profesional Java JDK6*, Editorial AnayaMultimedia.

[17] Sarwar, B., Konstan, J., Terveen, L. y Riedl, J. (2001), *"Item-Based Collaborative Filtering Recommendation"* en Proceedings of the 10th International World Wide Web Conference, Hong Kong, China.

[18] H.Deitel, P.Deitel (2004) *Como Programar Java*, Editorial Prentice Hall.

[19] Gutiérrez, A. y Bravo, G. (2005) *PHP5 a través de ejemplos*, Editorial Ra-Ma

[20] Krug, S. (2000),*No me hagas pensar. Una aproximación a la usabilidad en la Web*, Prentice Hall, Madrid.

[21] Manchón, E. (2002),*¿Qué es usabilidad?* (*http* : $//www.ainda.info/que_es_usabilidad.htm$).

[22] García Gómez, J. C. y Saorín Pérez, T. (2006),*Usabilidad para principiantes*, disponible en Internet (http://usalo.es/?p=117).

[23] José Hernández Orallo, Mªjosé Ramiez Quintana, César Ferri Ramirez..(2005) *Introducción a la Minería de datos*, Editorial Prentice Hall.

[24] Francisco Feito Higueruela,Juan Ruíz de Miras, Andrés Molina Aguilar..(1996) *Análisis y Gestión de Datos*, Editorial Universidad de Jaén.