Sistema de recomendación de documentos aplicado a un contexto académico

Luis Felipe Borjas Reyes
10611066

19 de marzo de 2010

Universidad Tecnológica Centroamericana
(UNITEC)
Escuela de Ingenierías

Índice general

1 Introducción General

El presente proyecto explora el concepto de un sistema de *recomendación oportuna* de documentos: un sistema que permita la recuperación de documentos relevantes a una necesidad de información *implícita* en una tarea que el usuario está llevando a cabo. Se distingue de un sistema tradicional de recuperación de información, como los motores de búsqueda, en que éstos responden -en propósito y diseño- a una necesidad de información explícitamente proveída por el usuario. Y se puede diferenciar de los sistemas de recomendación convencionales en que éstos se basan en artículos que ya conocen, mientras que la tarea del usuario, en el caso del sistema propuesto, es desconocida para el sistema.

Cuando un docente está planificando una sesión (lección magistral, proyecto, exposición, etc.) la necesidad latente de información se deriva de que quizá el docente se podría beneficiar de documentos que se relacionen con la sesión y con el curso en sí, tanto para referencia propia como para ofrecer como lecturas o recursos suplementarios a los alumnos; y, sin embargo, muchos docentes abdican de ampliar su cátedra de esta manera por varias razones, muchas de las cuales se pueden resumir en desánimo ante el prospecto de buscar tales recursos o el desconocimiento de la existencia de los mismos. Pero ¿qué tal si, en el uso de un sistema en el que administran sus clases, estos documentos útiles les fueran facilitados a los docentes sin necesidad de una búsqueda activa? Es ésta la necesidad que el presente proyecto busca suplir.

1.1. Propósito

Desarrollar un servicio web que recomiende al usuario documentos relevantes a un contexto local, formado éste por la documentación de actividades o sesiones de un curso administrado en línea, expresando esta documentación mediante una descripción de la sesión actual y, opcionalmente, documentos que el usuario proporcione. La recomendación consistirá en la búsqueda de documentos relacionados conceptualmente a la tarea del usuario en un índice de documentos mantenido por el sistema y la consiguiente recomendación de los que se consideren útiles, evaluando la utilidad a partir de un perfil del curso construido y actualizado automáticamente y el contexto inmediato de la sesión actual.

1.2. Objetivos

1. Elegir la naturaleza de la colección documental base para la implementación y pruebas. Ya sea en la forma de documentos obtenidos de internet o mediante la interfaz de búsqueda de una librería digital o motor de búsqueda existente.

2. Desarrollar una interfaz para la expresión de los contextos de usuarios, así como la elección e implementación de la manera en la que se expresarán, construirán y actualizarán los perfiles de los cursos.

3. Implementar un componente de indización e igualación de documentos y consultas; y desarrollar un componente que permita el filtrado de los resultados del primero en base a los perfiles y contextos.

1.3. Contexto

Se puede comparar la tarea que este proyecto llevará a cabo con la que realiza una persona que compra un artículo en `www.amazon.com`: el usuario está interesado en el artículo que está adquiriendo, después de todo, ese es su propósito principal al utilizar el sistema, pero el sitio también le sugiere, cuando el usuario ha elegido un producto, otros que podrían intesarle; eso es precisamente un tipo común de sistema de recomendación: un componente *secundario* en otro contexto mayor -como un sitio de compras o de renta de películas- que, de manera no intrusiva, exacta y rápida, ofrece al usuario opciones que puedan enriquecer su experiencia. Un sistema así, entonces, *intuye* qué podría ser de interés a un usuario en base al contexto en el que está actuando o al perfil que ya ha construido de él (en sitios de recomendación de música -por ejemplo, `www.last.fm`, - se mantiene un perfil de los usuarios para predecir qué podría interesarles después de un tiempo de uso); un sistema así, entonces y como aclara Safran (2005), se vale de la *contextualización* y la *adaptación* para llevar a cabo su tarea. De ahí que estos sistemas se consideren un buen ejemplo de *inteligencia colectiva*, construir conocimiento a partir de información brindada por los usuarios, porque se valen de lo que los mismos usuarios opinan - mediante algún tipo de retroalimentación implícita o comunicación explícita- acercan los artículos en la colección de artefactos del sistema. En suma, un sistema de recomendación, en base al contexto o al usuario, determina qué artículos de la colección de artefactos que maneja podrían interesar a quien lo usa *dado que haya elegido algún artefacto de la colección.* Un sistema de este tipo no puede, ni debería, por definición, recomendar algo en relación a un artículo que no conoce.

Por otro lado, la recuperación de información pretende suplir necesidades de información tratando de asimilar consultas de un usuario a documentos en una colección, retornando el subconjunto de aquéllos que considere relevantes. Se puede decir que las consultas del usuario son *documentos desconocidos* -en cuanto que no están presentes íntegra y textualmente en la colección manejada por el sistema y que, en base a éstos, el sistema busca qué documentos en su colección guardan alguna relación de similitud con ellos. Un sistema de recuperación de información, por excelencia, es *el* componente principal y la tarea primordial de un usuario en cualquier contexto, por lo que pueden darse la licencia de dejar que el usuario examine un conjunto relativamente grande de resultados y que refine o reformule las consultas. Se concluye que un sistema de recuperación de información extrae documentos en base a consultas que *no están en su colección explícitamente.*

El sistema que se pretende construir en este proyecto puede ser visto como un híbrido de recomendador y recuperador de información: del primero toma el importante papel de la inteligencia colectiva y la no interferencia con el usuario -lo que incluye presentar conjuntos de resultados que no sean tan grandes que obliguen al usuario a dejar su tarea principal por explorarlos, menos a cambiar drásticamente su trabajo sólo para obtener otros resultados; y de los sistemas de recuperación de información toma el hecho de búsquedas de consultas y documentos no necesariamente existentes en la colección. Este enfoque no es nuevo, y la investigación en sistemas de este tipo en especial -iniciada por Rhodes (2000a)- se ha valido de servicios ya existentes de recuperación de información, lo cual podría probarse contraproducente: los sistemas de recuperación de información tradicionales asumen que la relevancia de un documento es equivalente a su similitud a la consulta del usuario, y están en lo correcto, para aplicaciones orientadas a la búsqueda activa. Pero, en un sistema que pretenda apoyar una tarea realizada por el usuario ¿sería verdaderamente *útil* proporcionarle documentos que probablemente ya buscó o está buscando fuera del sistema? En realidad, el sistema debería traer a la atención del usuario artefactos que habría pasado por alto por no saber cómo buscar o ni siquiera estar al tanto de su existencia o relación con su contexto local. Para ello, la medida por la cual se debería regir un sistema como el propuesto debería ser la *utilidad*: qué artefactos pueden traerle una ganancia al usuario mediante la reducción de un esfuerzo (al ayudarle a hacer una búsqueda que no sabía cómo realizar) o la serendipia (al mostrarle elementos que le ayuden a ampliar su tarea (Herlocker et al., 2004; Rhodes, 2000a).

1.4. Proyectos relacionados

Se pueden encontrar algunos sistemas surgidos de la investigación en el campo de sistemas de recomendación orientados a la recuperación oportuna de información:

1. Remembrance Agent: Un módulo que se utiliza en el editor de textos *emacs*[1], y que sirve para realizar búsquedas de información relacionada a lo que se escribe en distintos repositorios, desde el correo electrónico del usuario hasta librerías digitales, mostrando los resultados en un apartado de la ventana de edición. Disponible para descarga en `http://www.remem.org/` y descrito en detalle en Rhodes (2000a). Utiliza búsqueda mediante palabras clave y permite buscar explícitamente otros documentos relacionados a las sugerencias.

2. Margin Notes: Un agente que revisa páginas en html y agrega, en un margen derecho, vínculos a documentos locales que podrían ser de interés a las distintas secciones del documento; extrae palabras clave de la página y se basa en la *co-ocurrencia* de éstas en los documentos locales para sugerir. Presentado en Rhodes (2000b).

3. Jimminy: desarrollado para una computadora que se porta en el hombro con una terminal en un casco, recaba datos del contexto físico y social de la persona mediante

[1] *http://www.gnu.org/software/emacs/*

sensores (GPS, infrarrojos, etc) y permite tomar notas mediante un teclado que se porta en el brazo. A partir del contexto local, sugiere notas que podrían ser de utilidad. También introducido por Rhodes (2000a).

4. FIXIT: aplicado al dominio de la reparación de fotocopiadoras y bienes relacionados, permite a los asesores de servicio técnico obtener información de los manuales de usuario en base a la consulta que estén atendiendo. Construye las consultas mediante los síntomas y problemas identificados, se vale de una red de inferencia bayesiana para determinar la probabilidad de relevancia de un documento a una consulta. Definido en Hart & Graham (1997).

5. Watson: un sistema que supervisa el entorno del usuario (su uso de procesadores de texto, navegadores de internet y clientes de correo electrónico) para sugerir documentos útiles, utiliza algoritmos heurísticos que encuentran unidades conceptuales en los documentos, a partir de las cuales realiza búsquedas concurrentes en distintas bases de datos y luego elimina duplicados en los resultados mediante algoritmos de agrupamiento. Definido en Budzik & Hammond (1999).

6. À propos: trata de determinar en qué etapa de la composición de un documento está el usuario para sugerirle lecturas y recursos para ampliar su composición, también se basa en búsquedas basadas en palabras clave en distintas bases de datos. Presentado en Puerta Melquizo et al. (2007).

7. PIRA: un sistema basado en internet que, a medida que se escribe una investigación, sugiere documentos relacionados y términos de búsqueda, a la vez que permite citarlos automáticamente al elegirlos de los resultados, agregándolos también a una bibliografía. Se basa en servicios web de terceros para encontrar los términos clave, ampliar la búsqueda con sinónimos y ofrecer las facilidades de cita automática. Disponible para pruebas en `www.writencite.com` y propuesto en Gruzd & Twidale (2006).

8. Implicit Queries y Stuff I've Seen: sistemas de administración de la información personal que permiten la búsqueda más allá de palabras clave, basándose en tipos de documento, personas y fechas; el sistema Implicit Queries infiere las necesidades de información mientras el usuario escribe un correo electrónico y le sugiere documentos que podrían servirle de apoyo,como correos anteriores, entradas de agendas, etc. Una descripción se puede encontrar en Cutrell et al. (2006); Dumais et al. (2004).

9. AMIDA: supervisa lo que se discute en una reunión de trabajo y recupera textos de reuniones anteriores que puedan servir para fundamentar puntos de la presente. El concepto y la descripción del sistema se pueden encontrar en Popescu-belis et al. (2008).

10. Scienstein: un sistema de recomendación que se basa en lo que se ha escrito y citado para sugerir documentos que puedan ampliar una investigación. Descrito en Gipp & Hentschel (2009).

Los sistemas anteriores tienen en común el uso de servicios de terceros para la búsqueda de información, así como basar las consultas en palabras clave extraídas del contexto local. Sin embargo, el filtrado de los resultados en este tipo de sistemas es poco o nulo: el conjunto de documentos que se presentan al usuario está ordenado de acuerdo a la relevancia que tienen éstos en relación a una consulta que el sistema ha construido. En este proyecto se pretende echar mano del conocimiento que se pueda obtener de cursos y usuarios para implementar un filtrado que reporte resultados más útiles, partiendo de la premisa de que un curso se puede modelar en un perfil y que éste puede ser útil para saber qué documentos mostrar en sesiones individuales de documentación.

2 Marco teórico

2.1. Recuperación de información.

2.1.1. Introducción

Se comienza con una puesta al día en el área de la recuperación de información (RI) , ya que, a pesar de ser los sistemas de recomendación ya toda un área de investigación por mérito propio, es necesario clarificar que lo que subyace a un sistema como el que se presenta en este proyecto es, en mayor medida, una tarea de recuperación de documentos. Se presentan los componentes de un sistema computacional de RI y sus métodos de implementación, además del proceso genérico que sigue un usuario a la hora de utilizar un sistema de esta índole. Para terminar, se determina cómo se podría evaluar un sistema de recuperación de información.

2.1.2. Definiciones y conceptos preliminares

Aunque últimamente, con el extendido uso de los motores de búsqueda, se pueda creer que el área de la recuperación de información es relativamente nueva y su máxima expresión son los susodichos sistemas, la verdad es que la recuperación de información como tal nace mucho antes, quizá desde que comenzaron a existir las primeras grandes colecciones de documentos, pues la necesidad de información, y el consecuente desarrollo de técnicas para buscarla entre lo que otros ya han documentado, es inherente al desarrollo de una cultura escrita.

En un primer intento por definir la recuperación de información, se puede decir que:

> La recuperación de la información es la obtención de material que satisfaga una necesidad de información.

Entendiendo por material cualquier artefacto (video, sonido, texto, etc.) que sirva para resolver una carencia de información en quien emprende la actividad de recuperación. Así, una consulta al directorio de nuestro teléfono celular para recordar el número de un amigo consistiría en una actividad de recuperación de información.

Pero si de definiciones se trata, se puede citar la definición dada en el primer capítulo de (Schütze et al., 2009) como una muy buena para comenzar la discusión de la que tratará esta introducción y esta primera parte en general:

> "La recuperación de la información es encontrar material (usualmente documentos) de una naturaleza no estructurada (por lo general, texto) que satisface una necesidad de información en colecciones grandes (que pueden estar guardadas en computadoras)."

De lo anterior se pueden resaltar algunos conceptos que será de gran utilidad elaborar, nótese que la referencia a sistema no implica uno computacional:

Material de naturaleza no estructurada: Se pueden considerar como la *salida* del sistema. Los textos con los que nos encontraremos en grandes colecciones (e incluso, en librerías digitales) no son, necesariamente, de naturaleza homogénea: unos pueden ser formularios con campos rígidamente definidos, otros, documentos con cierta estructura, como artículos científicos con un título, resumen, palabras clave, etc; o bien en formatos electrónicos estándar, como xhtml o xml; pero la mayor parte de los documentos no tienen una estructura reconocible (lo cual no excluye que puedan tener alguna de naturaleza arbitraria) y esta situación se agrava aún más cuando la colección en uso no es ni más ni menos que la *world wide web*.

Necesidad de información: Se puede considerar como la *entrada* del sistema. Un usuario del sistema (una persona u otro sistema) expresa de alguna manera una carencia de información y se trata de procesar ésta de manera que se le puedan proveer recursos relevantes. Es importante notar que la necesidad de información puede ser explícita -en forma de consulta- o implícita -que se puede inferir del contexto o de las preferencias del usuario.

La definición dada es, a todas luces, una descripción precisa de los sistemas modernos de recuperación de información, pues especifica la noción a los sistemas de recuperación documental contemporáneos.

De las definiciones dadas, se puede apuntar el equívoco nombre que se ha dado a este concepto, o al menos a su implementación práctica: recuperación de información parecería sugerir que, de una gran colección de documentos, se puede extraer *información útil*, pero, como se puede observar, los sistemas de RI no aspiran a tanto: sencillamente dan al usuario documentos que puedan suplir su necesidad, pero es la responsabilidad última del usuario determinar que en efecto es así. Como nota marginal, sin embargo, recuérdese que otra área de las ciencias computacionales, la *extracción de la información, sí* aspira a llenar necesidades estructuradas de información.

(Schütze et al., 2009) Menciona también otros conceptos que vale la pena clarificar:

Documento Por el cual se entiende aquello que consideraremos una unidad en el sistema: pueden ser las páginas de un libro, los mismos libros, los artículos de un periódico o los periódicos de un día.

Relevancia El valor que el usuario le da a un documento en base a su necesidad de información. Es un concepto que es, en realidad, bastante subjetivo, porque para algún usuario documentos que pudieran coincidir exactamente con el tópico que parecía ser el que deseaba conocer serían considerados menos relevantes -por considerarlos quizá demasiado obvios- que otros que, aunque menos relacionados, considere que le aportan más.

2.1.3. Componentes de los sistemas de recuperación de información

Un sistema de recuperación de la información de documentos en grandes colecciones, como los entendemos actualmente, desde bibliotecas hasta motores de búsqueda, a pesar de las grandes diferencias de implementación y arquitectura, tienen algo en común: todos tienen ciertos componentes que, si bien pueden variar en importancia o discernibilidad de otros, definen a un sistema de RI. Antes de discutirlos más a fondo, véase en la figura2.1 el *ciclo de la recuperación de la información* para tener una idea de los roles que juegan los componentes de estos sistemas.

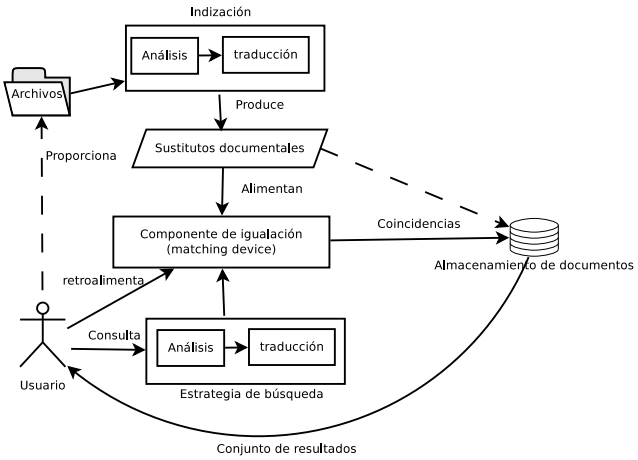


Figura 2.1: El ciclo de la recuperación de la información. Adaptado de (Walker & Janes, 1999)

Los componentes y actividades más notables de un sistema de recuperación de información pueden considerarse, según han determinado (Russell & Norvig, 2003; Walker & Janes, 1999) , los siguientes:

Una colección de documentos: Es el conjunto de todos aquellos documentos que el sistema conoce y puede brindar como resultados a sus usuarios. Constituye el componente central de un sistema de RI, pero sería de poco valor[1] si no fuera por el concepto de los *sustitutos documentales (en inglés, document surrogates),* que son representaciones de los documentos que el sistema conoce y que permiten una obtención más rápida de los mismos. Estos sustitutos documentales se elaboran mediante un proceso de gran importancia: *la indización*; la cual, aunque es una actividad, es también, en muchos de los casos, un componente bien definido del sistema. El propósito de la indización es crear los sustitutos documentales que se almacenarán en la colección y servirán de entrada al componente de igualación. Cada nuevo documento que es agregado al almacenamiento del sistema precisa ser indexado; y esta actividad puede ser tan compleja como se quiera: puede ir desde

[1] Si se prescinde de la indización, la otra alternativa es una búsqueda lineal, nada factible para grandes colecciones documentales.

la simple actualización del índice inverso con el documento, la representación como vector de palabras, o la clasificación del mismo en una taxonomía.

El componente de igualación: Es la parte que sirve de intermediario entre el almacenamiento documental y las consultas del usuario. Es importante notar que la entrada de este componente no es necesariamente la consulta original del usuario, sino que puede haber sido analizada y traducida al *lenguaje interno* del sistema, que es, en última instancia, una *representación del conocimiento*. Una vez en poder de la consulta, este componente trata de asimilar ésta con los sustitutos documentales y luego, si hay coincidencias, solicita al almacenamiento los documentos.

La consulta del usuario: La consulta es la manera del usuario de expresar su necesidad de información. La cual puede ir desde una pregunta en lenguaje natural hasta unos cuantos términos claves de un vocabulario controlado, pasando por consultas escritas con lógica booleana. La consulta se forma mediante la elección de parte del usuario de una *estrategia de búsqueda*.

El conjunto de resultados: Los documentos que el sistema ha considerado relevantes a la búsqueda del usuario, presentados de la manera que se considere idónea, ya sea gráficamente, como los sistemas que se centran en mostrar al usuario no sólo los documentos relevantes sino sus relaciones o como una lista en orden de relevancia según criterios propios del sistema.

2.1.3.1. La estrategia de búsqueda

La necesidad de un *índice* de la colección documental no es algo nuevo ni propio de sistemas computacionales de RI; más bien, como hace notar (Safran, 2005), la recuperación de información nace de la necesidad de determinar si mediante un documento cierta carencia de información se puede suplir sin tener que recurrir a una lectura completa, o al menos significativa, del recurso. Y es de esta necesidad, tanto en documentos de extensión considerable como en colecciones grandes, que se crean los índices en los libros y los sistemas de clasificación bibliotecarios. Así, todo sistema de RI que se precie de ser útil deberá proveer alguna manera de acceder a la información contenida en los documentos; de cualquier manera, los documentos han de ser representados de forma que la técnica elegida pueda valerse de ellos, y en esto radica la importancia de la elaboración de *sustitutos documentales*.

En cualquier sistema de RI, ya sea mediante la ayuda de un profesional en una biblioteca o en un motor de búsqueda, el usuario debe ser capaz de expresar su necesidad de información, es decir, como bien señala (Russell & Norvig, 2003), que la RI es, en el fondo, un asunto de *comunicación*.

Pero el tema de la elaboración de una consulta no es nada sencillo. En primera instancia, se debe discernir qué conceptos podrían satisfacer la necesidad y expresarlos ya sea en un lenguaje natural o en alguna otra representación conceptual de comunicación. Mas precisamente aquí se encuentra el primer obstáculo de la recuperación de la información: la naturaleza subjetiva de los conceptos, y, por tanto, las múltiples maneras de

expresar uno. A esto muchos sistemas de recuperación de información responden con el uso de palabras clave o vocabularios controlados -aquellos donde se da preferencia a una expresión de un concepto sobre otro, por ejemplo, al uso de la palabra automóvil para referirse a un carro. El usuario podría tener acceso a éstos o podría *aprenderlos* mediante la experiencia adquirida en búsquedas anteriores. Esta limitación, si se desea ver así, en los sistemas de recuperación de información es expresada de una manera idónea por (Walker & Janes, 1999):

> Queremos buscar conceptos, pero estamos forzados a buscar palabras.

El otro lado de la comunicación es igualmente importante. Una vez que el usuario ha elegido una consulta, el sistema debería ser capaz de interpretarla y procesarla de manera que el componente de igualación tenga éxito en encontrar coincidencias en los documentos. Este proceso puede ser tan sencillo como simplemente procesarla tal cual se introdujo o tan complicado como se quiera; de hecho, muchos sistemas de recuperación de información pre-procesan las consultas de usuario expresadas en algún lenguaje natural con técnicas que han probado hasta cierto punto ser exitosas:

1. Normalización de mayúsculas y minúsculas: convertir todo en los documentos y consultas a sólo mayúsculas o minúsculas, evitando así que documentos importantes se pierdan porque los términos no coinciden en este aspecto con lo contenido en documentos. Es evidente que esta técnica tiene sus desventajas, pues puede dar lugar a falsos resultados (como documentos que contienen el término en inglés *aids*, refiriéndose quizá a apoyos económicos en el tercer mundo cuando se ha buscado por el término *AIDS*, el síndrome de inmunodeficiencia adquirida).

2. Reducir las palabras a sus raíces: -en inglés *stemming*- que consiste en eliminar terminaciones de plural/singular, femenino/masculino o los sufijos de conjugaciones en verbos regulares. Aunque algoritmos de buena utilidad existen para ello, en muchos de los casos se tiene menor ganancia de la que se podría esperar -sólo de un 2 % en inglés según Russell & Norvig (2003); aunque se ha determinado que es más útil en otros lenguajes como el alemán (Thorsten Brants & Google, 2004).

3. Usar sinónimos para ampliar o normalizar las consultas.

4. Corregir la ortografía. Lo cual podría dar lugar a ganancias, pero, dada la variedad documental en internet y la frecuencia del surgimiento de neologismos, podría probarse hasta cierto punto contraproducente. (Stein & Meyer Zu Eissen, 2003) reporta el uso exitoso de un algoritmo de corrección ortográfica basado en el sonido.

5. Utilizar la estructura y la *meta-data* de los documentos. Los usuarios pensarán en conceptos que probablemente se encuentren en los títulos, resúmenes o encabezados de los recursos. Documentos internamente estructurados como artículos científicos y de periódicos o construidos en formatos organizados como xml pueden ser candidatos a este tipo de refinamiento. Asimismo, el hecho de que documentos hayan sido citados o enlazados por otros mediante términos que no aparecen necesariamente en el documento permite ampliar las posibilidades de encontrarlo.

Como se ve, el sistema debe tratar de entender la consulta del usuario antes de buscar coincidencias. En el capítulo siguiente técnicas relacionadas al procesamiento de la consulta se presentarán más a detalle; cabe notar que, vista como un problema de comunicación, la RI se podría valer de los avances en lingüística computacional, pero, como bien apuntan Thorsten Brants & Google (2004), debido a la naturaleza de corta extensión de muchas consultas en sistemas basados en internet, el uso del procesamiento de lenguaje natural en un sistema de RI debería usar herramientas especialmente confeccionadas para el mismo y no adaptadas de otras áreas de investigación.

2.1.3.2. Presentación de los resultados

Una vez que el sistema ha determinado los documentos que, según el modelo que utiliza, son relevantes a la consulta del usuario, debe presentarlos. Cualquier usuario de motores de búsqueda diría, según su experiencia, que este es un problema trivial: sencillamente se deberían presentar los documentos en orden descendente de relevancia. Para muchos sistemas, esto es suficiente porque los usuarios o sólo desean alguna información somera o no se disponen a investigar más a fondo. Pero, ¿qué sucede con aquellos que desean indagar a conciencia en un tema? Los motores de búsqueda de propósito general más utilizados se quedan cortos en este respecto, pero no significa que no haya otras maneras de representar los resultados de una manera más útil, ni mucho menos que no se hayan implementado.

Una manera es mostrar gráficamente las relaciones entre documentos, entendiéndose éstas, por ejemplo, como que uno cite a otro o conceptos aparecen de maneras similares en los documentos. Un enfoque así se puede encontrar en (Gipp & Hentschel, 2009), donde el sistema muestra de manera gráfica los documentos y sus relaciones (y los documentos más relevantes se muestran de mayor tamaño).

Pero quizá el usuario esté más interesado en ver los documentos agrupados de alguna manera. Para esto se pueden utilizar dos técnicas estrechamente relacionadas con el campo del aprendizaje de máquina:

Clasificación: Dada una categorización preexistente de conceptos, los resultados se presentan agrupados bajo éstos. Así, por ejemplo, en un sistema de recuperación de información en la base de documentos de alguna enciclopedia, los resultados de una búsqueda respecto a "Eco" mostraría alguno resultados agrupados bajo la categoría "Escritores" con documentos refiriéndose a Umberto Eco, otros bajo la categoría "Física" con documentos relacionados al fenómenos físico del eco, etc. Un sistema computacional de RI que desee utilizar clasificación debería utilizar algún tipo de aprendizaje de máquina supervisado: en primera, la categorías se deben decidir de alguna manera (ya sea a criterio de quien compila la colección o mediante otras técnicas de inteligencia artificial), a seguir, se debe elegir un conjunto de documentos y etiquetarlos con la categoría a la que pertenecen; este conjunto se conoce como el *conjunto de entrenamiento*. El algoritmo de aprendizaje debe formular una *hipótesis* que explique por qué a los documentos se les dieron esas etiquetas de categorías. Luego, otro conjunto de documentos etiquetados se alimenta al sistema, esta vez,

sin las etiquetas, para ver si lo puede clasificar con éxito -comparando al final si las etiquetas previamente asignadas coinciden con las dadas por el sistema. Los resultados se evalúan y la hipótesis se puede refinar (con una *matriz de confusión*, por ejemplo). Nótese que la clasificación cuenta con una *taxonomía de tópicos preexistente,* por lo que sólo es factible para colecciones de documentos manejables con categorías claramente definidas. Esta técnica es utilizada por (Gelbukh et al., 2005) en su sistema clasificador.

Agrupación: (En inglés, *clustering*). A diferencia de la clasificación, los documentos se agrupan en categorías determinadas *a la hora de presentar los resultados* -y no preexistentes. La agrupación es un problema de aprendizaje *no supervisado* de máquina (mientras que la clasificación es supervisada) y se vale de técnicas estadísticas para agrupar los documentos según ciertas medidas de distancia (la cual es una medida estadística de frecuencia y reincidencia de palabras en los textos): documentos con poca distancia entre sí se agruparán juntos; luego, se pueden determinar las etiquetas a poner a los grupos, lo cual se puede basar en los títulos de los documentos, según alguna medida estadística, por ejemplo, los más cercanos al *centro* del grupo; o sencillamente etiquetarlos con las palabras más frecuentes. Dada su naturaleza no supervisada, esta técnica, aunque computacionalmente más costosa, es preferida por los investigadores. El sistema presentado en (Stein & Meyer Zu Eissen, 2003) utiliza técnicas de agrupación para presentar los resultados.

Además, el sistema podría valerse de la *retroalimentación* del usuario, ya sea implícita, mediante el acceso a documentos, el tiempo de estancia en ellos -si éstos consisten en hipervínculos- o la impresión; o explícita, mediante evaluación de relevancia. Así, con la retroalimentación, el sistema podría construir un nuevo conjunto de resultados a partir de éste, buscando aquellos documentos que se aproximen a los preferidos por el usuario -es importante resaltar que esta es una característica básica de los sistemas de recomendación.

2.1.3.3. Obtención de la colección documental

Aunque la existencia de la colección se da por sentado al hablar de la implementación de sistemas de recuperación de información, la obtención de ésta es indiscutiblemente fundamental para el sistema. Aunque existen colecciones en internet que se construyen exclusivamente mediante la colaboración de los usuarios, como wikipedia[2] o el *open directory project*[3], y aún cuando esta tendencia es cada vez más común, dado el auge de la *web 2.0* y la *web semántica,* un sistema de RI a gran escala deberá contar con un conjunto de documentos suficientemente grande como para satisfacer la mayor parte de las necesidades de información de sus usuarios, y, para ello, deberá disponer de algún método de obtención automática de nuevos documentos.

Las *arañas web* son tales mecanismos: son programas que, partiendo de un conjunto inicial de páginas web, encuentran en éstas nuevos vínculos a otros documentos, los cuales

[2] `www.wikipedia.org`
[3] `http://www.dmoz.org/`

agregan a su conjunto y exploran para encontrar nuevos vínculos y así sucesivamente. Este enfoque simple de las arañas web se puede encontrar implementado en el lenguaje de programación python por Segaran (2007). Las dificultades del uso de una araña web en un ambiente comercial, como la privacidad y el volumen de datos procesados, se pueden encontrar someramente relatadas en las experiencias de Brin & Page (1998) al desarrollar google[4]. Por su parte, Schütze et al. (2009) menciona las características más importantes que una araña web debería tener: *robustez,* para no caer en trampas de navegación que lo hagan quedarse demasiado tiempo en un solo dominio; *cortesía*, para respetar las políticas de sitios que tienen un límite de visitas automáticas o no desean ser agregados a un índice -téngase presente que existe un estándar[5] para determinar, en un archivo de texto llamado usualmente robots.txt, las direcciones de un sitio que las arañas web no tienen permitido agregar a los índices de sus sistemas anfitriones; *escalabilidad, desempeño, eficiencia y posibilidad de distribución*, para usar óptimamente los recursos de los que dispone; *frescura*, obtener siempre copias nuevas de las páginas; *un estándar de calidad*, para explorar sólo aquellas páginas que podrían verdaderamente ser relevantes, ya sea por su riqueza de contenido, su *fama* entre otras páginas o su apego a convenciones estándar y *extensibilidad*, una araña se debería poder adaptar a nuevos formatos, protocolos y estándares.

Un caso interesante de implementación de arañas web es cuando no se desea obtener cualquier página para agregarla a la colección, sino que sólo aquellas que se refieran a cierto tópico; en este caso, las arañas deben tomar decisiones sobre la relevancia a un tópico y hasta aprender de recolecciones anteriores Menczer et al. (2001); Rungsawang & Angkawattanawit (2005).

2.1.4. Modelos de recuperación de información

El componente más importante de un sistema de RI, como se vio anteriormente, es el tratamiento de la consulta del usuario en el componente de igualación; para que éste devuelva objetos verdaderamente relevantes o útiles, la indización debe haber construido una representación óptima de los mismos, y es en esta etapa en la que entran en juego los distintos modelos presentados en este apartado. En la figura 2.2 se presenta una taxonomía de algunos de los modelos de recuperación de información utilizados comercialmente y en investigación. De éstos, se explorarán someramente unos pocos en este apartado.

El índice invertido Antes de explorar los modelos de recuperación de información, es menester definir una de las técnicas de indización presente en la mayor parte de las implementaciones: el *índice invertido*. Éste consiste en un archivo que contiene todas las palabras de la colección documental y en qué documentos ocurren éstas. Es esta estructura de datos la que permite que los sistemas de recuperación de información computacionales sean factibles, ya que la búsqueda por una palabra no se hace en los documentos en sí, sino en este archivo. Puesto que la construcción de esta estructura

[4] `www.google.com`

[5] definido en `http://www.robotstxt.org/wc/exclusion.html`

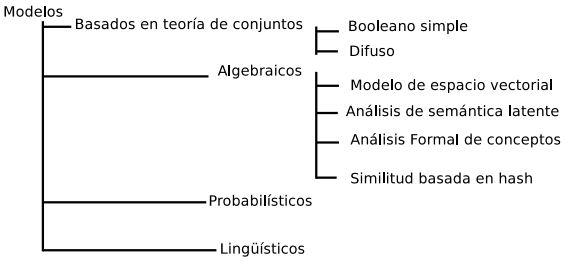


Figura 2.2: Taxonomía de modelos de recuperación de información. Adaptada de Safran (2005); Spoerri (1993a)

ocurre en la etapa de indización, es aquí donde se hacen decisiones fundamentales sobre el tratamiento de los documentos como qué hacer con la puntuación, utilizar las palabras completas o sólo las raíces, si distinguir entre mayúsculas y minúsculas, etc -nótese que estas decisiones deben reflejarse en el tratamiento dado a las consultas, como se define en el apartado 2.1.3.1.

El proceso general de construcción de un índice invertido se puede encontrar explicado en Schütze et al. (2009); Walker & Janes (1999), y se resume a continuación:

1. Juntar los documentos a ser indizados, en alguna estructura de datos.

2. Separar en unidades (*tokens*) cada documento; las unidades son, por lo general, palabras, pero no necesariamente.

3. Hacer pre-procesamiento lingüístico (esto es, aislar raíces, normalizar mayúsculas o minúsculas, ignorar puntuación o palabras vacías, hacer etiquetado gramático, desambiguar palabras mediante análisis semántico, etc.) El pre-procesamiento puede parecer trivial para lenguajes cuya sintaxis y alfabeto lo permiten (como los que utilizan los alfabeto latino, griego y cirílico, por ejemplo; sin embargo, la tarea es más difícil en otros). Para una discusión más a fondo del tema del procesamiento lingüístico de documentos, consúltense Bird et al. (2009); Schütze et al. (2009). Además Thorsten Brants & Google (2004) presenta un sumario de las técnicas más utilizadas en sistemas comerciales en esta etapa y sus resultados.

4. Por cada palabra, construir el índice agregando los documentos en los cuales ocurre ésta y, si se decide considerar la estructura, agregar también en qué parte del documento es la ocurrencia. Los documentos, por lo general, se representan mediante una clave numérica única y las partes mediante una clave o una abreviación.

5. Se ordena el índice alfabéticamente.

Nótese que el índice contiene entradas únicas para las palabras, así que es responsabilidad de quien lo implementa incluir un campo donde se incluya también la cantidad de veces que una palabra ocurre en un documento (o en una parte del documento, si es más granular); esta decisión, como las demás, depende del modelo en uso.

Búsqueda orientada a conceptos vs. la búsqueda orientada a palabras Como ya se apuntó, los sistemas tradicionales parten de la premisa de que las palabras que están en la consulta son suficientes para la búsqueda, sin embargo, como se sabe, la necesidad de información del usuario encierra *conceptos* que trascienden una mera instancia léxica, y por ello se presentarán también técnicas que parten de la premisa de que la consulta sólo es una base semántica para la búsqueda y que ésta se puede aumentar (ya sea mediante sinónimos aportados por tesauros, búsquedas en ontologías o colecciones indexadas por conceptos) o refinar (mediante la desambiguación de términos, por ejemplo). Ambos enfoques se pueden llamar también *búsquedas léxicas* y *búsquedas semánticas*, respectivamente.

La motivación que subyace a la transición de la recuperación de información léxicamente orientada a una semánticamente orientada es la observación de la incertidumbre inherente al lenguaje natural, observable en dos aspectos del mismo: la *sinonimia* y la *polisemia*. Las consultas pueden ser expresadas en palabras que no aparezcan como tales en documentos verdaderamente relevantes sino que en su lugar se utilicen sinónimos, por lo que la exhaustividad del sistema se ve afectada, al no traer todos los artículos relevantes. Asimismo, documentos irrelevantes a una consulta pueden ser obtenidos debido a la polisemia -la variabilidad contextual del significado de una misma palabra- por lo que la precisión se ve afectada (Dumais et al., 1988).

2.1.4.1. Modelo Booleano

Es el modelo más simple de recuperación de información. Podemos definirlo como una función que, dados un documento y una consulta escrita como una expresión booleana de palabras, el documento es relevante a la consulta si y sólo si la expresión evalúa a verdadera en el documento, donde las cláusulas de la expresión son las palabras mismas y la presencia de una palabra en el documento implica que la cláusula es verdadera.

Si se utiliza un índice invertido, basta con hacer búsquedas para las palabras de la consulta y devolver los documentos en los que éstas se encuentran, aplicando operaciones simples de conjuntos a los resultados según la expresión booleana. La consulta se puede hacer en un tiempo lineal respecto a la cantidad de documentos y palabras en la consulta; asimismo, optimizaciones heurísticas se pueden aplicar a las consultas.

Este modelo es tan fácil de implementar y tan eficiente computacionalmente, que se ha usado en sistemas comerciales de búsquedas en bases de datos desde hace casi medio siglo. Pero los sistemas implementados han ido más allá de los simples operadores booleanos y han introducido *operadores de proximidad* y comodines de búsqueda, con el propósito de mejorar los resultados. Un ejemplo de un sistema comercial con una implementación altamente especializada y optimizada del modelo booleano es el sistema *DIALOG*, explorado en detalle en Walker & Janes (1999).

Las desventajas del modelo booleano radican en que no hay manera de ordenar los resultados, puesto que los términos o están o no están, sin manera de saber si la palabra está realmente relacionada al tema central del documento o sólo aparece allí una o pocas veces en contextos distintos. Asimismo, los usuarios podrían no estar familiarizados con la lógica booleana, y aún siendo así, la diferencia de resultados entre una conjunción y

una disyunción de términos puede ser demasiado abismal como para dar al usuario una idea de cómo refinar su consulta.

2.1.4.2. Modelo booleano extendido mediante lógica difusa

Trata de sobreponerse a las limitaciones del modelo booleano tradicional implementado la lógica difusa para las operaciones, así, los resultados de las expresiones no serán sólo verdadero o falso (0 ó 1), sino cualquier valor entre ellos, usualmente representado en el continuo [0, 1].

Los documentos se tratan como conjuntos difusos y la membresía de cada término se puede calcular en base a la frecuencia de su ocurrencia, por lo que los resultados de las operaciones difusas extendidas se pueden utilizar para dar una medida de relevancia a los documentos en el conjunto de resultados. Aunque este es el enfoque más intuitivo, la lógica difusa se puede utilizar para extensiones más drásticas del modelo de conjuntos de recuperación de información (Crestani & Pasi, 1999). Nótese también que este tipo de lógica se asemeja más al lenguaje natural, pues toma en cuenta la incertidumbre y la ambigüedad. La presentación de un modelo de recuperación de información generalizado -que trasciende al mero uso de palabras clave para la búsqueda y se extiende al uso de *conceptos*- se puede ver discutida en Baziz et al. (2004) y los fundamentos de la lógica difusa para la manipulación y representación del conocimiento se encuentran esbozados en Pajares & Santos (1999).

2.1.4.3. Modelo de espacio vectorial

Es el modelo más utilizado en la actualidad, y ha servido de base para otros modelos que han reportado éxito en la investigación. Fue propuesto para la tarea de recuperación de información por Salton et al. (1975) y trata a las consultas y los documentos como vectores multidimensionales, donde cada término es una dimensión. A la hora de la búsqueda, se computa la similitud entre el vector de consulta y los de los documentos, retornando los resultados ordenados según su relevancia (siendo ésta proporcional a la similitud).Por lo general, las dimensiones se ponderan en base a la frecuencia de los términos en los documentos y así se pueden ordenar los resultados por relevancia.

En primer lugar, se debe tener en cuenta la representación de la colección documental. Como ya se dijo, el modelo de espacio vectorial parte de la premisa que los documentos se representan como vectores multidimensionales, tomando todas las palabras de la colección documental como las dimensiones. Nótese que estos vectores se han de construir con los términos que hayan quedado luego del pre-procesamiento lingüístico, en especial de la remoción de palabras vacías, para eliminar dimensiones posiblemente innecesarias.

El vector para un documento D se construye con los *pesos* asociados a cada término del espacio -el cálculo de éstos se discutirá enseguida. Visto de esta manera, entonces, para un vector con t dimensiones, un documento se representa así:

$$D_i =< w_{i1}, w_{i2}, \ldots, w_{it} > \tag{2.1}$$

id	Original	Pre-procesado	Vector
1	"Dijo alguien que decía lo que dícese: Ser o no ser el ser que se es"	"decir[3]"	<3,0,0>
2	"Ser alguien en la vida, es poco decir, mi vida"	"decir[1] vida[2] poco[1] "	<1,3.17,1>
3	"Dije que a poco es vida"	"decir[1] vida[1] poco[1] "	<1,1.58,1>

Cuadro 2.1: Documentos de ejemplo para el modelo de espacio vectorial. El procesamiento lingüístico elimina palabras vacías, puntuación y mayúsculas y reduce a raíces. Por claridad, las frecuencias se muestran contiguas a las palabras, aunque esto es responsabilidad del índice inverso. Las ponderaciones se han calculado con *if*tdf*

A lo largo de este apartado, se utilizarán los documentos presentados en la tabla 2.1 para ilustrar los conceptos clave.

El peso w de un término se puede calcular respecto a ciertas medidas estadísticas y algunas de las más utilizadas en la implementación de este modelo son:

- Peso booleano: asigna 0 ó 1, dependiendo de la presencia o ausencia del término.
- Frecuencia local: el peso es proporcional a la cantidad de veces que la palabra ocurre en el documento. Esta ponderación es fácil de calcular y refleja de manera transparente una premisa común en mayor o menor medida a los modelos basados en palabras: que la posición de la palabra no es importante, suposición heredada de los modelos estadísticos basados en inferencia bayesiana Lewis (1998). La frecuencia del k-ésimo término en el i-ésimo documento de una colección se puede expresar como f_i^k
- Frecuencia local multiplicada por la frecuencia global inversa: conocida en la literatura como *tf*idf*, se basa en la observación que dice que los términos verdaderamente importantes en una colección documental son aquellos que aparecen con la suficiente frecuencia en un documento para definirlo y a la vez son poco frecuentes globalmente. Esta ponderación será proporcional a $f_i^k \cdot IDF_k$ donde, para una frecuencia documental d_k que representa el número de documentos en los que aparece el k-ésimo término, para n documentos, la frecuencia documental inversa se define como:
$$IDF_k = \log \frac{n}{1 + d_k} = \log_2 n - \log_2 d_k + 1 \tag{2.2}$$
- Valores discriminantes: aunque la medida *tf*idf* es ampliamente utilizada, el mismo Salton et al. (1975), reconoció que aún así hay términos que son más decisivos en definir el carácter relevante de los documentos, en especial dentro de colecciones relativamente homogéneas; por lo cual introduce el concepto de *valores discriminantes* en función de la densidad del espacio vectorial: un término es discriminante en la medida en que su ausencia o presencia en la colección afecte qué tan cerca

están, en términos de similitud, los documentos; un buen término, entonces, sería aquel que, al ser eliminado, hace que los documentos queden más cerca entre sí, afectando peyorativamente la precisión de las búsquedas. Puesto que no es factible computacionalmente encontrar el valor discriminante de todos los términos de la colección -ya que habría que calcular la densidad t veces, lo cual sería $O(n^3)$- el mismo autor propone examinar el comportamiento de los términos respecto a las frecuencias, y reporta que buenos términos discriminantes son aquellos que no son demasiado raros ni demasiado frecuentes, por lo que los términos poco frecuentes se deberían agrupar bajo un sinónimo (una clase de tesauro) y los más frecuentes, en *frases*, dado que éstas son, por lo general, menos propensas a tener la misma ocurrencia global que sus componentes individuales.

- Frecuencias de grupo: en el mismo trabajo, Salton et al. (1975), propone un multiplicador similar al *tf*idf*, pero en vez de hacerlo relativo a toda la colección, calcularlo en base a las frecuencias en grupos, luego, claro, de haber agrupado los documentos relacionados mediante algún algoritmo de *clustering* (una explicación más exhaustiva de algoritmos de agrupamiento en la tarea de recuperación de información se puede encontrar en Schütze et al. (2009) y una implementación simple del agrupamiento jerárquico, en Segaran (2007)).

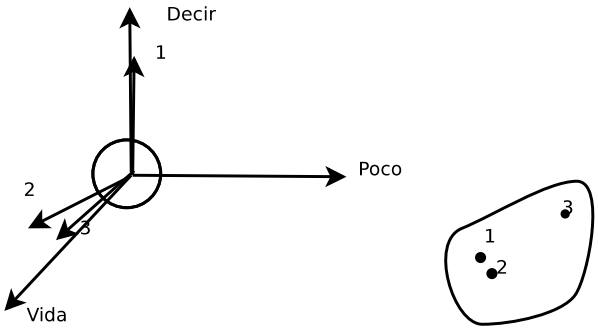
En la enumeración anterior, nótese el uso indistinto de "término" y "palabra" como equivalentes, esto es debido al extendido uso de *unigramas* como los componentes fundamentales de los sistemas de recuperación de información -lo cual no es necesariamente el mejor ni único enfoque, pero es el más simple (Russell & Norvig, 2003).

Una vez obtenidos los vectores, se pueden representar como en la figura2.3a; pero la longitud de los vectores no es importante, sino la *similitud* que hay entre ellos, la cual se puede calcular con un producto interno o como una función inversa al ángulo que hay entre ellos (Levow, 2003). Por lo general, se utiliza la medida de *similitud de coseno* (definida en el apartado 2.2.3.1) porque ésta es independiente de la magnitud del vector, a diferencia de la mera diferencia de magnitudes, que puede ser ambigua, porque dos documentos muy similares pueden tener magnitudes proporcionales y, así, una diferencia significativa (Schütze et al., 2009) que no refleja la verdadera similitud. El modelo se puede simplificar tomando la intersección de cada vector con la esfera unitaria y luego estirando ésta sobre dos dimensiones, de manera que los vectores ahora se aprecian como puntos en un plano y estos puntos estarán cerca unos de otros si los vectores originales estaban angularmente próximos. La figura 2.3b ilustra esta simplificación para el ejemplo de esta sección.

El quid de este modelo estriba en la siguiente hipótesis (Salton et al., 1975):

> "La densidad de un espacio documental y el rendimiento de un sistema de recuperación de información están inversamente relacionados"

Lo cual, llevado al espacio simplificado, implica que los documentos más similares a una consulta estén lo más cerca posible a ésta y unos a otros; y, al mismo tiempo, que los menos relevantes estén lo más alejado posible a la consulta, esto debido a que la



(a) Representación en vectores

(b) Representación en un espacio plano

Figura 2.3: Representaciones en el espacio vectorial para el ejemplo de la tabla2.1

precisión de la búsqueda se puede ver afectada por la densidad del espacio: entre más cerca están los documentos, más posible es que, para una consulta, se obtengan resultados no-relevantes; por lo que es deseable que la densidad se minimice, distinguiendo lo más posible, mediante indización y ponderaciones óptimas, entre los documentos. Técnicas de agrupamiento basadas en la ponderación dada a los términos se pueden aplicar para hacer posibles estos requerimientos, acercando los grupos de documentos similares y alejando los disímiles.

2.1.4.4. Mejoras del modelo de espacio vectorial

Aunque el modelo de espacio vectorial tiene en cuenta algunos aspectos del lenguaje natural, los términos individuales siguen consistiendo en dimensiones, por lo que documentos semánticamente similares a una consulta, pero léxicamente diferentes, seguirán apareciendo con un bajo puntaje de relevancia -este fenómeno se da aún cuando se usa el valor discriminante bajo clases de sinónimos, puesto que los términos son o equivalentes o no equivalentes a una clase.

Para resolver este problema, el *modelo de espacio vectorial orientado a tópicos* asume que los términos no sólo pueden ser equivalentes o no equivalentes, si no que pueden pertenecer en distintos grados a estas dos clases -tomado de la lógica difusa; este modelo se define y compara con otros basados en el de espacio vectorial clásico en Becker & Kuropka (2003). Por su parte, el *modelo generalizado de espacio vectorial*, asume que los términos individuales de un vector también pueden ser similares entre sí, por lo que se representan también con vectores en un espacio combinatorio. Una implementación de este modelo se puede encontrar explicada en (Tsatsaronis & Panagiotopoulou, 2009). Ambos modelos parten de la idea de aprovechar las relaciones entre términos, y difieren en que el primero utiliza información basada en tesauros y ontologías y el segundo infiere las relaciones entre términos a partir de la colección documental.

Un tercer modelo que extiende al de espacio vectorial clásico es el basado en *Semántica Distribuida* (DSIR, por sus siglas en inglés); éste utiliza la distribución contextual de las palabras, basada en una matriz de co-ocurrencia, para ponderar los términos y así sobreponerse a los problemas de sinonimia y polisemia, dejando alejados en el espacio vectorial aquellos documentos donde los términos sólo sean polisémicamente equivalentes y cercanos aquellos donde haya similitud sinonímica. Este modelo ha sido implementado y probado con una colección documental relativamente grande por Rungsawang (1988).

2.1.4.5. Indización semántica latente

Este modelo parte de la premisa de que existe una "estructura semántica latente que subyace a la elección de palabras específicas de una consulta"Dumais et al. (1988). Es decir, las palabras que el usuario elige para realizar una consulta son instancias de un *concepto* expresable de otras maneras. Este modelo trata de encontrar esta estructura latente mediante una aproximación estadística: parte de una matriz donde se representan todos los términos contra los documentos en los que aparecen (cada entrada de ésta corresponde a la frecuencia del i-ésimo término en el j-ésimo documento) y luego esta matriz se descompone en tres matrices menores mediante la técnica de *descomposición en valores singulares,* reduciendo así la cantidad de dimensiones del espacio documental y dejando que términos sinonímicamente relacionados se representen en una sola dimensión. Esta reducción dimensional basada en la contextualización estadística de los términos es la que permite que documentos semánticamente relacionados a una consulta queden en la vecindad de ésta en el espacio vectorial, mejorando el rendimiento del sistema Levow (2003); Spoerri (1993b). Por su parte, Schütze et al. (2009), resume el proceso de representación documental como la transformación de la matriz original de términos/documentos a un espacio con k dimensiones (donde k es una constante elegida usualmente con un valor entre los pocos cienes) donde éstas son representadas por los k principales *eigenvectores (vectores propios, los que mejor representan un espacio)* del espacio original, haciendo así que sólo aquellas dimensiones que son verdaderamente representativas de los documentos subsistan y que otras que en realidad representaban valores cuasi-sinonímicos desaparezcan y se vean reducidas a las dimensiones propias del espacio. Una explicación más detallada y matemáticamente fundamentada de este modelo se puede encontrar en Deerwester et al. (1990). Y una comparación entre este modelo y otro basado en la expansión de consultas de usuarios mediante el uso de ontologías se puede encontrar en Ozcan & Aslangdogan (2004).

Este modelo no carece de problemas: en primer lugar, el costo computacional relacionado a la transformación matricial -el cual se debe hacer cada vez que la colección documental cambie *drásticamente* Dumais et al. (1988) -énfasis en qué se considere como drástico, pues la adición de pocos documentos nuevos se puede hacer al espacio ya existente, ignorando momentáneamente las nuevas dimensiones que éstos pudieran introducir y sólo redefiniendo el espacio cuando se alcance algún umbral fijo o relativo al rendimiento (Schütze et al., 2009) ; y, a medida que las colecciones crecen, la polisemia de los términos puede afectar negativamente a la transformación (al introducir vectores propios espurios o equívocos). Una solución a ambos problemas puede ser una partición

de la colección documental en segmentos más manejables, como se propone en Bassu & Behrens (2003).

2.1.4.6. Análisis formal de conceptos

Modelo propuesto para recuperación de información por Becker & Eklund (2001), se basa en la construcción de un *retículo* (en inglés, *lattice)* a partir de los conceptos y los documentos, de manera que cada nodo de esta estructura consista en dos conjuntos *A (la intensión) y B (la extensión)* tales que A tenga todos los atributos comunes a los documentos del nodo y B tenga todos los documentos que comparten los atributos de A. De esta manera, en un nodo se agruparían todos los documentos que tengan un concepto en común, definido éste por la intensión del nodo -el conjunto de atributos elegido para representarlo. Dada la construcción de esta estructura, todos los sub-nodos de un nodo consistirán en sub-conceptos del concepto que éste represente.

La tarea de recuperación sencillamente consistirá, entonces, en representar la consulta como un pseudo-nodo del retículo y luego encontrar el conjunto de nodos que tenga la intensión de la la consulta, siendo los documentos más relevantes la extensión combinada de éste. Se pueden sugerir refinamientos de la consulta explorando los sub-nodos del pseudo-nodo de consulta, garantizando que el conjunto de resultados será siempre más preciso (puesto que, dada la naturaleza de la estructura, los sub-nodos son siempre más específicos que sus antecesores). El ordenamiento por relevancia de los resultados se puede hacer en función de la distancia entre los nodos de los conceptos y la magnitud de las extensiones de cada nodo.

Las ventajas de este modelo radican en la facilidad que da al usuario de ampliar o refinar sus consultas: sólo se le sugieren súper-nodos y sub-nodos de la estructura, según sea el caso; además, la magnitud del conjunto de resultados de una consulta refinada o ampliada se puede conocer antes de realizar la recuperación de los documentos, por lo que el usuario puede conocer el tamaño de este conjunto antes de llevar a cabo la consulta en sí, ahorrándose tiempo y costo computacional.

Como se puede observar, el aporte de este enfoque radica más en el mejor rendimiento a la hora de refinar o ampliar las consultas, siendo la primera búsqueda quizá menos eficiente, computacionalmente, que la de otros modelos, puesto que implica búsquedas en todo el espacio reticular.

Por otro lado, se puede utilizar el análisis formal de conceptos para construir jerarquías conceptuales a partir de la colección documental (Cimiano et al., 2005) y expandir la etapa de indización (Schonhofen & Charaf, 2001): la idea básica es mejorar la ponderación de los términos mediante la determinación de clases de sinónimos y reducción de la polisemia -o la identificación de dimensiones en el caso de los modelos generalizados de espacio vectorial.

2.1.4.7. Similitud basada en hash

Este es un modelo relativamente nuevo, al menos en la aplicación a la recuperación documental, cuya idea central es simple pero poderosa: se utilizan funciones *hash* apli-

cadas a los documentos y cuando existe una colisión (se asigna la misma firma *hash* a dos o más documentos) es porque éstos son similares y, por tanto, probablemente igual de relevantes a una consulta. Este método se ha probado computacionalmente más eficiente que la búsqueda en un espacio vectorial (Stein & Potthast, 2007), pero tiene un par de debilidades: en primera, la elección de una función *hash* debe ser en extremo cuidadosa, funciones muy estrictas podrían no asignar las mismas firmas a documentos relacionados y lo contrario para funciones muy permisivas; otra debilidad radica en la premisa misma del modelo: es totalmente basado en palabras clave -puesto que la función hash es intrínsecamente léxica- por lo que es inaplicable para escenarios donde una orientación más semántica que léxica sería deseable.

2.1.4.8. Modelos probabilísticos

Mientras que los modelos basados en el espacio vectorial y otros similares se basan en medidas heurísticas para maximizar la relevancia verdadera de los documentos y así poder obtener buen rendimiento, los modelos basados en probabilidades cuentan con cierta información de relevancia preliminar que utilizan para estimar la probabilidad de que un documento sea relevante a una consulta. Esta información previa puede ser obtenida al momento de construir el índice mediante análisis estadístico similar al de otros modelos: frecuencias, co-ocurrencias o distribuciones. Estadísticamente hablando, que un documento D sea relevante (r) a una consulta Q se puede expresar con la probabilidad condicional expresada en la siguiente probabilidad, la cual se desea maximizar en un sistema de recuperación de información:

$$P(r|D,Q) \tag{2.3}$$

Si se aplica la regla de Bayes, que dice que una probabilidad depende de su inversa (en este caso, la probabilidad de que una consulta sea relevante a un documento) y se aplica la regla de cadena -que dice que se puede estimar una probabilidad conjunta una vez conocido el antecedente de una probabilidad condicional de alguno de los elementos, en el caso de la recuperación de información, que la probabilidad inversa a (2.3) se puede expresar como la probabilidad de que una consulta se dé dado un documento relevante si se sabe que el documento es verdaderamente relevante- y, finalmente, asumiendo que la probabilidad de relevancia es constante (la constante α), ya que ésta sólo puede ser 0 ó 1; se obtiene la siguiente transformación:

$$P(r|D,Q) = \frac{P(D,Q|r)P(r)}{P(D,Q)} = \frac{P(Q|D,r)P(D|r)P(r)}{P(D,Q)} = \frac{\alpha P(Q|D,r)P(r|D)}{P(D,Q)} \tag{2.4}$$

Si además decimos que en vez de maximizar la probabilidad de relevancia de un documento a una consulta queremos maximizar la razón de ésta probabilidad respecto a su negación - es decir, se maximizará la razón $P(r|D,Q)/P(\neg r|D,Q)$ y luego se asume que si un documento es irrelevante a una consulta, la probabilidad de éste es independiente de la consulta, basándonos en (2.4), tenemos la siguiente transformación:

$$\frac{P(r|D,Q)}{P(\neg r|D,Q)} = \frac{P(Q|D,r)P(r|D)}{P(Q|D,\neg r)P(\neg r|D)} = P(Q|D,r)\frac{P(r|D)}{P(\neg r|D)} \tag{2.5}$$

En donde la razón $P(r|D)/P(\neg r|D)$ representa la calidad intrínseca del documento: qué tan probable es que resulte un documento relevante a cualquier consulta, ya sea en base a su riqueza de términos discriminantes o su magnitud. Así, al maximizar la razón de relevancia de los documentos, llegamos a obtener una probabilidad proporcional a su inversa, descubriendo una relación causal entre las consultas y los documentos, lo cual permitiría construir una red de inferencia bayesiana (Turtle & Croft, 1991) que establezca que, cuando ocurran ciertas palabras en las consultas, ciertos documentos serán relevantes.

Este modelo de inferencia asume que el orden de las palabras en un documento no es importante, sino su frecuencia, por lo que la probabilidad $P(Q|D,r)$ se puede expresar como el producto de las probabilidades de cada palabra que conforma la consulta; como ya vimos, esto está lejos de ser verdad, pero es una condición necesaria para el desarrollo de estos modelos y usualmente da buenos resultados (Lewis, 1998).

Las ventajas de este modelo radican en el cálculo preciso de relevancia de los documentos, así como de liberar al usuario de la preocupación de los lenguajes estructurados de consulta (dado que se puede escribir en un lenguaje natural), pero sobre-simplifican el modelo lingüístico, lo cual puede ir en detrimento en la precisión del sistema; además, requieren de estimación de las probabilidades de relevancia de los documentos, que requieren un conocimiento o procesamiento previos de la colección documental (Spoerri, 1993b). Una discusión más detallada del trasfondo estadístico de estos modelos, así como la presentación de otros enfoques de naturaleza similar, se puede encontrar en Fuhr (1992).

2.1.4.9. Modelos lingüísticos y orientados al conocimiento

Como ya se ha observado, la recuperación de la información es, en el fondo, un problema de comunicación, y, si se toma del campo de la inteligencia artificial el concepto de *agente inteligente*, se puede aducir que, si el agente ha aprendido lo suficiente de la colección documental o el dominio del conocimiento como para poder procesar preguntas y contestarlas, un sistema de RI se reduce a la "respuesta de preguntas a partir de una base de conocimiento" (Russell & Norvig, 2003).

Sin embargo, poder representar de manera satisfactoria y utilizable la colección documental como información para agentes inteligentes sigue siendo demasiado costoso, tanto en términos de recursos computacionales como de tiempo. De ahí que la tendencia más común en el campo de la recuperación de la información es combinar la lingüística computacional con los otros modelos: utilizar información contextual para ponderar las palabras -por ejemplo, una palabra en función de sustantivo usualmente es más relevante que la misma palabra en función de objeto indirecto; valerse de *colocaciones* (frases que son más relevantes como tales que sus componentes individuales) a la hora de indexar, desambiguación de palabras según conocimiento previo del dominio, para eliminar la

polisemia, entre otras. Sin embargo, el estudio de Thorsten Brants & Google (2004) respecto a este enfoque híbrido es desalentador: sistemas que lo utilizan son igual o menos eficaces que otros con métodos más ortodoxos, tal falta de ganancia se debe en gran parte a la longitud promedio de las consultas de los usuarios en motores de búsqueda comerciales: son demasiado cortas como para la inferencia de contextos y desambiguación, componentes fundamentales para el procesamiento del lenguaje natural en textos. Por otro lado, sistemas con enfoques más centrados en la lingüística computacional pura, se han implementado con cierto éxito, entre los que se cuentan el famoso sistema *SMART* del promotor del modelo de espacio vectorial, Gerard Salton; el sistema *CLARIT,* que en una etapa de análisis morfológico y sintáctico elegía palabras que fueran candidatos fuertes para ser consideradas dimensiones en el índice y el sistema *SIMPR*, que tiene un enfoque iterativo similar al de CLARIT (Leiva & Muñoz, 1996).

Por su parte, un modelo estadístico basado en un modelado probabilístico del lenguaje se puede incluir aquí. Este modelo, relacionado con los modelos basados en probabilidad, computa una probabilidad distinta: trata de construir un perfil lingüístico de cada documento y luego se pregunta qué tan probable es que un documento pudiera generar una consulta, en base a este perfil; los documentos con mayor probabilidad se devolverán (Schütze et al., 2009).

Un enfoque más general se encuentra en sistemas orientados al conocimiento, que utilizan reglas obtenidas del dominio del conocimiento para elegir términos discriminantes o para establecer la causalidad en el caso del uso de redes de inferencia bayesiana (como en el sistema *FIXIT,* de Hart & Graham (1997)). Estos sistemas son completamente basados en el uso agentes inteligentes o sistemas expertos, por lo que se requiere del uso extenso de técnicas de ingeniería del conocimiento.

2.1.4.10. Comparación de los modelos presentados

Con el fin de ofrecer un sumario sucinto y crítico de los modelos introducidos, se ha elaborado la tabla 2.3 que también evalúa someramente cada uno en términos de la utilidad al proyecto propuesto.

2.1.5. Evaluación de un sistema de recuperación de información.

Como se hace ver en Voorhees (2002), los sistemas de recuperación de información se pueden medir ya sea según el sistema, es decir en base al desempeño del mismo en términos estrictamente computacionales o en base a cómo se ordenan los documentos por relevancia; o se pueden medir según el usuario: el usuario considera que un sistema de RI es útil si los resultados suplen su necesidad de información.

Es definitivamente más difícil conducir pruebas con usuarios, tanto en tiempo como en los recursos de los que se debe disponer. Ambos enfoques se discutirán en esta sección, comenzando con el del usuario, cuya valoración se mide según su satisfacción con el sistema.

Antes de entrar en consideraciones precisas, nótese que los documentos de una colección se pueden clasificar, vistos como los resultados de una búsqueda, como se ilustra en la

Modelo	Orientación	Ventajas	Desventajas
Booleano	Conjuntos, coincidencia exacta, palabras clave.	Fácil implementación. Computacionalmente eficientes.	Demasiado estrictos. Requieren el aprendizaje de un lenguaje de consulta.
Booleano extendido	*ídem*	Concilian la fácil implementación con un enfoque más natural.	Aún requieren de un lenguaje especializado de consulta.
Espacio vectorial (VSM)	Coincidencia aproximada, palabras clave.	Maximiza la similitud léxica entre documentos. Automatiza la indización. Representación óptima de los documentos.	Basado en coincidencias léxicas.
VSM extendido	Coincidencia aproximada, conceptos.	Igual que el VSM, y además permite un tratamiento más eficiente del lenguaje natural.	Los cálculos pueden llegar a ser poco factibles computacionalmente.
Análisis semántico latente	Coincidencia aproximada, conceptos.	Se vale de la estructura semántica de las consultas.	La transformación de la matriz es costosa. Para colecciones muy grandes pierde el rendimiento.
Análisis formal de conceptos	Coincidencia aproximada, conceptos.	Permite consultas más interactivas. Aprovecha la semántica de la colección. Eficiente en términos de estimación de resultados.	La búsqueda en el espacio reticular puede llegar a ser lenta y difícil de computar.
Basado en Hash	Coincidencia exacta. Palabras clave.	El orden de la computación es casi lineal, muy eficiente. La búsqueda se hace en tiempo constante.	Depende de la función de hash seleccionada. Sin posibilidad de consideraciones lingüísticas.
Probabilístico	Coincidencia estimada. Palabras clave.	Puede integrar consideraciones expertas para las probabilidades.	Asume estricta coincidencia léxica y sobre-simplificación lingüística.
Lingüístico	Contexto semántico.	Reduce el problema de la recuperación de información a comunicación, el modelo ideal.	Modelos puros no son actualmente factibles Dependiente del lenguaje.

Cuadro 2.3: Cuadro comparativo de los modelos de recuperación de información

tabla 2.4

	Relevante	No relevante
Obtenido	Positivos verdaderos (vp)	Falsos positivos (fp)
No obtenido	Falsos negativos (fn)	Verdaderos negativos (vn)

Cuadro 2.4: Clasificación de los documentos en una colección, vistos como resultados de una búsqueda

2.1.5.1. Medidas de evaluación

Se puede evaluar un sistema de recuperación de información en términos de precisión y exhaustividad. A continuación, se presentan ambas medidas, definidas con claridad enSchütze et al. (2009):

Precisión Responde a la pregunta: De los documentos encontrados ¿cuántos son relevantes a la necesidad de información?

Y se puede expresar con la siguiente ecuación, para un sistema convencional:

$$p = \frac{documentos\,relevantes\,obtenidos}{documentos\,obtenidos} = \frac{vp}{(vp + fp)} = P(relevante|obtenido) \qquad (2.6)$$

Nótese que la precisión se puede expresar como la probabilidad de que un documento sea relevante dado que fue obtenido.

Exhaustividad Mejor conocida por su nombre en inglés: *recall*. Responde a la pregunta: De los documentos *relevantes* que se encuentran en la colección ¿cuántos se han obtenido?

Y también se puede expresar con una fórmula:

$$r = \frac{documentos\,relevantes\,obtenidos}{documentos\,relevantes\,en\,la\,colección} = \frac{vp}{(vp + fn)} = P(obtenido|relevante) \qquad (2.7)$$

La exhaustividad también se puede expresar como una probabilidad: la probabilidad de que un documento sea obtenido dado que es relevante.

Cuando se trata de evaluar la exhaustividad, surge la pregunta ¿cómo se puede saber cuántos de los documentos en la colección son verdaderamente relevantes, dado que el concepto de relevancia es subjetivo? A esta pregunta, Walker & Janes (1999) responde con el concepto de *exhaustividad normalizada*: varios investigadores hacen búsquedas en la misma colección en base a una misma necesidad de información; juzga cada quien qué documentos son relevantes y el conjunto de todos los documentos relevantes elegidos por los investigadores se considera el conjunto de recursos relevantes en la colección para futuras mediciones.

Se puede ver el problema de encontrar documentos relevantes como una tarea de clasificación: donde relevante y no-relevante son dos etiquetas y el sistema utiliza aprendizaje de máquina supervisado para la labor.

Una tercera medida de evaluación se puede introducir: la *exactitud:* de todos los documentos, ¿qué tan bien se han clasificado, de manera que la mayor parte resulten, en una consulta, o verdaderos positivos (documentos relevantes y obtenidos) o verdaderos negativos (documentos irrelevantes y no obtenidos) -en otras palabras, que la colección tenga muy pocos o ningunos falsos positivos y falsos negativos para cualquier consulta; la exactitud, entonces, se expresa de la siguiente manera:

$$e = \frac{vp + vn}{vp + vn + fp + fn} \tag{2.8}$$

Pero, como ya se vio que la relevancia es en realidad bastante subjetiva, los sistemas obtienen casi 99.9 % de documentos no exactamente relevantes a una consulta (Schütze et al., 2009), por lo que la exactitud, aunque buena para medir el desempeño de tareas de clasificación, es irreal para medir la recuperación de información: ¡un sistema exacto no daría ningún resultado para casi ninguna consulta!

Relación entre la precisión y la exhaustividad Se puede decir preliminarmente, entonces, que un buen sistema es preciso y exhaustivo. Pero, reflexionando un poco más, salta a la vista que la precisión y la exhaustividad no siempre van de la mano, de hecho, en muchos de los casos hay que preferir una a la otra: esto se conoce en inglés como *the precision-recall tradeoff*. Este fenómeno será evidente con un ejemplo, adaptado de Walker & Janes (1999): imagínese a dos usuarios de un sistema de RI, uno, buscando sobre teatros famosos en el mundo para una asignación escolar: este usuario querrá obtener pocos documentos de buena calidad, para terminar su tarea con rapidez; este tipo de búsqueda se conoce como una *búsqueda de alta precisión*. El otro usuario, en cambio, es un investigador preparando su tesis de doctorado, así que querrá una gran cantidad de documentos para tener una considerable base bibliográfica que revisar; esta búsqueda será entonces una *búsqueda de alta exhaustividad*. Ambos tipos de búsqueda son extremos en un continuo: las búsquedas tienen siempre un poco de ambos tipos. Como se menciona en Schütze et al. (2009), se puede entonces ver a la exhaustividad como una función no-decreciente respecto al número de documentos obtenidos (¡se puede obtener un 100 % de exhaustividad si se devuelven todos los documentos como resultado para una consulta!) mientras que la precisión es decreciente respecto al número de documentos obtenidos (Podría tenerse una precisión del 100 % si se retornase sólo un documento muy relevante).

La relación entre ambas medidas se puede representar como un promedio, para así ver en una sola cifra qué sucede al aumentar una o la otra. Para expresar este promedio, un podría recurrir a la media aritmética (el promedio convencional) pero, como es oportunamente llamado a la atención por Schütze et al. (2009), la media aritmética sería engañosa: si obtuviésemos un 100 % de exhaustividad al retornar todos los documentos (un muy mal sistema de recuperación de información) aún así obtendríamos un 50 % de promedio -una medida que, vista con optimismo, podría sugerir que el sistema es bueno.

En consecuencia, se debe recurrir a otra manera de medir el promedio que dé resultados más fidedignos, para ello, se puede pensar en el rendimiento del sistema en términos de precisión y exhaustividad como la *media armónica ponderada* de ambos, siendo la media armónica la menor de las tres medias estándar (la otra es la media geométrica). Esta media armónica ponderada se conoce como *medida F:*

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \, donde \, \beta^2 = \frac{1-\alpha}{\alpha} \, y \, \alpha \in [0,1] \, (\therefore \beta^2 \in [0,\infty]) \quad (2.9)$$

De esta manera, podemos ajustar el valor de alfa para dar prioridad a una u otra medida, por ejemplo, ajustar alfa a 0.5 (equivalente a un valor de beta al cuadrado de 1) resulta en una *medida F balanceada*: se da el mismo valor tanto a la precisión y a la exhaustividad. Por conveniencia, el valor de beta cuadrada se expresa como subíndice de la medida F en uso, así, para la medida F balanceada:

$$F_1 = \frac{2PR}{P+R} \quad (2.10)$$

Claro está que se pueden elegir otros valores en función del tipo de sistema construido: valores de beta mayores a uno corresponden a énfasis en la exhaustividad; mientras que valores de beta menores a uno, en la precisión.

Para ilustrar este intercambio entre la precisión y la exhaustividad, se suele utilizar una ayuda gráfica: la curva de *Características operativas del receptor* (ROC, por sus siglas en inglés) : el eje de las abscisas (x) representa la tasa de falsos positivos y el de las ordenadas (y), la tasa de falsos negativos; el área bajo la curva representará la eficacia del sistema, esto según Russell & Norvig (2003). Por otro lado, Schütze et al. (2009) propone una curva ROC como la gráfica de la *sensitividad* (sinónimo de exhaustividad) en el eje de las ordenadas y la tasa de falsos positivos -que constituye el inverso de la *especificidad* (cuántos de los documentos no obtenidos son, en realidad, no relevantes); como paréntesis, la especificidad se define de la siguiente manera:

$$especificidad = \frac{vn}{fp+vn} \quad (2.11)$$

Así, la tasa de falsos positivos se puede ver como:

$$tasa\, falsos\, positivos = \frac{fp}{fp+tn} \quad (2.12)$$

Pero, claro, la especificidad en la mayor parte de los casos tenderá a la unidad: el número de verdaderos negativos siempre será grande; de ahí que la especificidad no sea una medida tan significativa. En términos de una curva ROC, ésta crecerá con una pendiente pronunciada en el lado izquierdo de la gráfica, es decir, que la tasa de falsos positivos será mínima y la exhaustividad crecerá tempranamente.

Así, a la hora de evaluar el sistema, se debe tener en cuenta a qué extremo del continuo de precisión-exhaustividad tenderán la mayor parte de las búsquedas. En un sistema de recomendación, se puede asumir que los usuarios no desean navegar por páginas y páginas

de documentos que ellos ni siquiera buscaron explícitamente (como se verá después, una de las máximas de un sistema de recomendación es su poca interferencia con la tarea del usuario).

Otras consideraciones Las medidas que se han discutido hasta este punto son para sistemas donde los documentos resultantes para una búsqueda se presentan como un *conjunto* esto es, sin ningún orden específico. Existen otras medidas y consideraciones matemáticas que tener en mente a la hora de presentar los resultados en un orden específico, por rango de relevancia, por ejemplo, ya que otros asuntos como la efectividad del algoritmo de asignación de rango vienen a consideración. Si el lector está interesado, en profundizar en estas medidas puede consultar la sección 8.5 de Schütze et al. (2009).

(Russell & Norvig, 2003) introduce someramente otras medidas, importantes más que todo, para motores de búsqueda: *rango reciproco*, para denotar cuáles de los resultados relevantes para el usuario se encuentran entre los primeros de la lista devuelta por el sistema y *tiempo de respuesta,* refiriéndose al tiempo total en el cual el usuario satisface su necesidad de información.

2.1.5.2. El proceso de evaluación

Para sistematizar la tarea de la evaluación, tanto desde el punto de vista del sistema como del usuario,Schütze et al. (2009) recomienda tener en cuenta los siguientes pasos:

1. Elegir una colección de documentos de dimensiones significativas.
2. Determinar un conjunto de necesidades de información: nótese que esto no es equivalente a un *conjunto de consultas*. La necesidad de información del usuario y cómo esta sea formulada deben considerarse distintas.
3. Obtener un conjunto de juicios de relevancia; usualmente se califica a un documento, dada una consulta, como relevante o no-relevante. Estos juicios se formarán por usuarios que evalúen manualmente los documentos.
4. A partir de los juicios de relevancia, calcular el rendimiento del sistema en términos de precisión, relevancia y cualesquiera otras medidas elegidas.
5. Según los resultados de los cálculos afinar manualmente o mediante técnicas de aprendizaje de máquina el sistema.

Algunas colecciones estándar de prueba se pueden encontrar mencionadas en Schütze et al. (2009); Voorhees (2002) y los lineamientos para construir una colección para efectos de evaluación de un sistema de recuperación de información se pueden encontrar en Cormack et al. (1998). En todo caso, es necesario que las necesidades de información sean coherentes con la colección elegida.

Como es evidente, en la tarea de evaluación de un sistema desde el punto de vista del usuario se debe recurrir a consultar a verdaderos usuarios humanos, pero, ¿cómo hacer para grandes colecciones documentales? Por lo usual, se elige cierta cantidad k de

documentos obtenidos con algún sistema de RI, posiblemente del que se evaluará, como subconjunto de evaluación. De este subconjunto, entonces, se obtendrán los juicios de relevancia de los usuarios.

2.2. Sistemas de Recomendación

2.2.1. Introducción

Se introducirá el concepto general de sistemas de recomendación, las técnicas más comunes para su implementación y los métodos utilizados en su evaluación.

Un sistema de recomendación se puede ver como un caso especial de un sistema de recuperación de información de alta precisión; el quid es, sin embargo, la *recomendación*: ¿cómo un sistema puede deducir qué puede preferir el usuario de entre un conjunto de resultados?

En este punto es idónea una analogía con una situación cotidiana -la cual, dicho sea de paso, es suficientemente popular como para ser el ejemplo por antonomasia de sistemas de recomendación: imagínese el lector que desea ver una película, para ello, decide ver la cartelera del cine más cercano, y elige como opciones unas cuantas que tienen muy buena publicidad en la televisión; indeciso, llama a un amigo que usted sabe que tiene un gusto similar al suyo y le pide que le recomiende una película, su amigo, que le conoce, trata de darle un par de opciones que se adecuen más a sus preferencias; aún indeciso, cuando está en la taquilla decide preguntarle a la persona que vende los boletos por una y esta persona le recomienda la que más se ha vendido en esa tanda, la cual no está entre las que su amigo le recomendó. Al final, usted sopesa las recomendaciones y decide ver una de las que su amigo le recomendó.

Del ejemplo podemos establecer que, de una cantidad considerable de opciones que responden a su necesidad original de información, qué película ver, el sistema *filtra* aquellas que, de alguna manera, considere pueden interesarle más. El amigo y el vendedor de taquilla se pueden ver como dos enfoques distintos de recomendación: el primero filtra la información en base a lo que ya sabe de usted y sus preferencias, además de los gustos que ambos tienen en común, esto tiene su contraparte en sistemas de recomendación en la técnica conocida como *filtrado colaborativo*, donde se recomiendan objetos preferidos por usuarios similares entre sí; el segundo caso también se vale del saber colectivo, pero de una manera más general, a saber, en la *popularidad general* de un artefacto.

En suma, los sistemas de recomendación se pueden ver como *sistemas de filtrado de información* en base a cierto perfil, ya sea construido con las preferencias del usuario y el contexto en el que el usuario se desenvuelve o infiriéndolo del comportamiento de éste y otros usuarios similares .

En las siguientes secciones se presentan los dos grandes tipos de sistemas de recomendación: los *basados en contenido* y los de *filtrado colaborativo*.

2.2.2. Recomendación con filtrado basado en el contenido

Los sistemas de recomendación basados en contenido se centran en el artefacto en el cual el usuario muestre interés en el momento: ya sea un documento propio en el que está trabajando o en un artículo de una colección. Un sistema de este tipo puede construir con el tiempo un perfil del usuario en base los artículos por los cuales se ha decidido antes. O, como se define en (Pazzani & Billsus, 2007):

> "[Los sistemas de recomendación basados en el contenido] sugieren un artículo basado en una descripción del mismo y un perfil de los intereses del usuario. [...] Tienen en común (1) un medio para describir los elementos que se recomendarían, (2) un medio para crear un perfil que describa el tipo de artefactos que un usuario favorece y (3) un medio para comparar elementos con el perfil del usuario para determinar qué recomendar.[6]"

(Baudisch, 1999) propone una analogía basada en las bases de datos relacionales: ver a un sistema orientado al contenido como una relación entre entidades, la cual se muestra en la figura 2.4

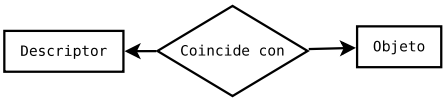


Figura 2.4: E-R representando a los sistemas orientados al contenido, adaptado de (Baudisch, 1999)

De esa relación entre entidades, se puede inferir la tabla 2.5, donde se muestran las posibles combinaciones de entradas y salidas para un sistema de filtrado de información orientado al contenido.

Entradas/Salidas	Descriptores	Objetos
Objetos	Indización	Búsqueda basada en ejemplos
Descriptores	Tesauros asociativos	Recuperación y filtrado de información

Cuadro 2.5: Aplicaciones de sistemas orientados al contenido, con las filas como entradas y las columnas como salidas. Adaptada de (Baudisch, 1999)

2.2.2.1. Componentes

De la definición dada por Pazzani & Billsus (2007) , los tres elementos numerados pueden encontrar su contraparte en los componentes de un sistema de recuperación de información, a saber, la *representación de los artículos* se puede asimilar al concepto de *sustitutos documentales* en la indización de la colección; el *perfil de usuario*, en papel de entrada principal al sistema, se equipararía a la *consulta* de un sistema de RI (nótese que

[6] Numeración del autor

esto refuerza lo antes mencionado como diferencia fundamental entre ambos sistemas: mientras que la consulta contiene todo lo necesario para la búsqueda, en un sistema de recomendación muchos detalles sobre la necesidad de información se deben inferir del perfil del usuario) y, por último, el componente que compara los artículos al perfil sería equivalente al *componente de igualación* de un sistema de recuperación de información.

Hecho el símil, se procede ahora a discutir los detalles y el rol de los componentes:

Representación de los artículos: Al igual que un sistema de RI, se debe elegir la representación de los artefactos que sea más idónea para la labor de recomendación; puesto que los usuarios evalúan los artículos, lo más natural sería elegir los *atributos* más conspicuos que todos los elementos tienen en común, haciendo así sencillo el almacenamiento porque se pueden ver los atributos como las columnas y cada componente como filas en una tabla. Pero para tener verdadera utilidad para el sistema, los valores de estos atributos deberían estar en un *dominio cerrado*, es decir, deberían elegirse de entre opciones previamente definidas; por ejemplo, para el atributo *género* en una película, es mucho más preferible decir "suspenso" a sabiendas de que éste es uno de quizá cinco posibles valores, y no decir "película que da ganas de que no termine nunca", porque tener demasiados atributos de texto libre vuelve inútil al sistema, o innecesariamente complicado, pues todas las ambigüedades inherentes al lenguaje natural se presentan y los análisis de similitud, fundamentales para el sistema, se vuelven difíciles de realizar. Una solución al problema de campos de texto libre es el uso de técnicas de asignación de relevancia y conversión de los términos a otras representaciones más manejables por un sistema computacional -como usar, por ejemplo, el modelo de espacio vectorial propuesto por Salton et al. (1975) y representar los campos de texto libre como vectores de términos.

Definidos los atributos a considerar y los dominios de éstos, resta el no menos importante problema de obtenerlos de los artefactos. Esta tarea es sencilla para cuando los dichos artefactos tienen una naturaleza estructurada (por ejemplo, recursos bibliográficos en formato bibtex[7]) o documentos en xml. Pero la mayor parte de documentos de texto tienen una estructura arbitraria o carecen de cualquier tipo de estructura; como se mencionó antes, técnicas de la indización en la recuperación de información se pueden aplicar a este tipo de entradas para indexarlas; aunque se puede también valer de técnicas de extracción de información para tratar de llenar una tabla de atributos predefinida -por ejemplo, cierto nivel de extracción de información se utilizó en el sistema de Kitamura et al. (2002).

El perfil del usuario Para que un sistema orientado al contenido tenga éxito, debe *conocer* al usuario. Esto lo hace mediante el aprendizaje de las *preferencias* del usuario y el historial de las interacciones de con el sistema. La construcción del perfil se puede tratar desde el punto de vista de la clasificación: partiendo del historial de interacciones con el sistema, llevar un control de qué artículos le interesaron al usuario ya sea mediante retroalimentación explícita (asignar un juicio cualitativo de valor, por ejemplo) o implícita (comprar o utilizar un artefacto); se etiquetan los elementos -por ejemplo, como

[7] `www.bibtex.org`

"interesantes" y "no interesantes"- y se usan estos datos para entrenar a un algoritmo clasificador. De esta manera, se construye una hipótesis de los gustos del usuario para predecir cuáles otros objetos podrían ser de interés. Métodos como Naïve-Bayes, exitosos en otras tareas de clasificación, se podrían aplicar en este tipo de sistemas de recomendación, como proponen Mooney & Roy (1999), otros algoritmos de clasificación más avanzados se han propuesto para estas tareas, como RIPPER (Cohen & Singer, 1999), árboles de decisión y k-vecinos más cercanos, clasificación en base al algoritmo de Rocchio para el modelo de espacio vectorial, el uso de clasificadores lineales -Widrow-Hoff, gradiente exponencial o máquinas de vectores de soporte- todos presentados someramente porPazzani & Billsus (2007). Ahora bien, en vez de aprender el perfil, se puede pedir al usuario que llene formularios sobre sus preferencias (por ejemplo, `www.mendeley.com`, pide al usuario recién registrado que elija su área de investigación y temas de interés para recomendarle lecturas), lo cual, en la práctica, se hace poco o pobremente.

O bien, el sistema puede tener un conjunto de reglas específicas al dominio en el que funciona, las cuales, aunadas al historial del usuario, se usan para inferir las preferencias (por ejemplo, un sitio de venta de computadoras puede recomendar productos de cierta marca a usuarios que han comprado varios artículos de ella).

El componente de recomendación Dado el perfil del usuario y la representación de los elementos, se puede ver a este componente como uno que lleva a cabo una tarea de recuperación de información. Técnicas avanzadas de recuperación de información en un sistema de recomendación orientado al contenido se pueden ver en el trabajo Chen & Wu (2008); técnicas de clasificación para recomendación, utilizadas por Basu et al. (1998) y el uso del procesamiento del lenguaje natural para obtener recomendaciones donde las preferencias del usuario son escasas, mas el contenido es rico en detalles útiles, en el sistema de Fleischman & Hovy (2003).

2.2.2.2. Limitaciones

Para ciertos tipos de artefactos, este tipo de filtrado puede mostrarse muy útil, como, por ejemplo, para entradas bien estructuradas de referencias bibliográficas, puesto que es usual que ciertos autores o citas sean reincidentes en tópicos de interés para el usuario. Sin embargo, para datos escasos, poco estructurados o de valoraciones más subjetivas, como música o sitios web, estos sistemas se quedan atrás, al necesitar suficiente información para construir un clasificador preciso, requiriendo del usuario el uso del sistema por un tiempo considerable o llenar explícitamente un perfil detallado, cosa no factible en ciertos sistemas donde el componente de recomendación es secundario.

2.2.3. Recomendación con filtrado colaborativo.

Con el cambio de paradigma que se ha venido dando en el internet en los últimos años, las personas empiezan a esperar un ambiente más social en los sistemas que utilizan. Esto se ve reflejado en la creciente popularidad del uso de las opiniones de otros usuarios para determinar las recomendaciones en los sistemas.

Un sistema que utilice este enfoque, como lo define Segaran (2007):

> "Busca en un grupo grande de personas y encuentra un conjunto más pequeño cuyos gustos se asemejen a los del usuario. Mira qué otras cosas ellos favorecen y las combina en una lista de sugerencias."

O, más sencillamente, Fox et al. (2010) expresa que un sistema de recomendación de filtrado colaborativo es:

> "Es un método basado en el ambiente social, usado para proponer artículos que usuarios que piensan de manera similar prefieren."

En consecuencia, los sistemas que utilizan filtrado colaborativo trascienden las limitaciones de los basados en contenido: se valen de la inteligencia colectiva para hacer recomendaciones y no tanto en un extenso perfil del usuario, puesto que pueden inferir en menos tiempo de uso del sistema, o con un perfil más reducido, cuáles otros usuarios opinan de manera similar al actual; las recomendaciones dependen de las evaluaciones más que del contenido de los artefactos, por lo se puede omitir la fase de análisis de contenido, la cual es computacionalmente costosa, y particularmente difícil para artefactos con multimedia y, por último, un sistema de filtrado colaborativo se orienta a la comunidad, lo cual permite que los elementos subjetivos y complejos encontrados en cualquier sociedad humana puedan tener cabida en el sistema, como aseguraMortensen (2007).

Evidentemente, debe haber alguna manera de expresar los gustos de las personas cuantitativamente, lo cual se hace usualmente mediante valores numéricos; para sitios como `www.delicious.com`, se puede usar el valor de 1 para indicar que un usuario ha publicado un vínculo y 0 para el caso contrario; para sitios de películas, se podrían usar valores entre uno y cinco para indicar qué tanto disfrutó el usuario cierta película.

Incluso , pueden verse las opiniones cuantitativas de los usuarios como atributos de los elementos, permitiendo así el uso de aprendizaje de máquina para el filtrado (Pazzani & Billsus, 2007).

De manera análoga a la definición dada para los sistemas orientados al contenido, en (Baudisch, 1999) también se puede encontrar una representación de los sistemas colaborativos como una relación entre entidades (figura 2.5) y las distintas combinaciones de entradas y salidas para esta relación (tabla 2.6).

Figura 2.5: E-R de los sistemas colaborativos, adaptado de Baudisch (1999)

De cualquier manera, una vez en posesión de datos cuantitativos de los gustos de los usuarios, se calcula la similitud entre usuarios; las técnicas utilizadas para esto pueden ser bastante complejas: desde el uso de la distancia euclidiana (Segaran, 2007), hasta máquinas de vectores de soporte (Grcar et al., 2005). Algunos de estos algoritmos se discuten a continuación.

Entradas/Salidas	Usuario	Objeto
Objeto	Búsqueda de expertos	*Filtrado colaborativo automatizado*
Usuario	Casamentero/Emparejador	*Filtrado colaborativo activo*

Cuadro 2.6: Combinación de entradas y salidas para un sistema colaborativo, adaptada de Baudisch (1999)

2.2.3.1. Métodos de filtrado colaborativo

El filtrado colaborativo se ha valido tradicionalmente de una matriz que representa las valoraciones de los usuarios respecto a los artefactos en la colección; esta matriz se conoce como *matriz de usuarios y artículos.* Por ejemplo, para un sistema de recomendación de literatura con seis libros y cuatro usuarios, se podría tener la representación mostrada en la tabla 2.7.

Usuarios/Libros	El castillo	La caverna	Fundación	G.E.B	Hamlet	The Silmarillion
Luis	1.0	4.5	4.0	4.9	5.0	4.5
Jorge	0.0	0.0	5.0	4.0	0.0	1.0
Fernando	3.0	3.0	0.0	4.5	1.0	2.0
María	5.0	3.7	3.0	0.0	4.5	4.0

Cuadro 2.7: Ejemplo de matriz de usuarios y artículos. Las valoraciones están en el rango [1-5], 0 equivale a no haber votado

De la matriz, si se separa en distintos vectores para las filas, se obtienen los perfiles de cada usuario (lo que juzgó de cada artículo). Tal separación es la base del primer enfoque de filtrado colaborativo: el *basado en memoria.*

Asimismo, si la matriz se separase en columnas, se obtendrían los votos de todos los usuarios para un artículo en específico, lo cual lleva al segundo enfoque: el *basado en modelos.*

Ambos enfoques pretenden *predecir* cuál sería la opinión del usuario respecto a los artículos relevantes, eligiendo así aquellos que el usuario preferiría.

Filtrado colaborativo basado en memoria

El filtrado basado en memoria trata de simular la situación cotidiana de buscar una opinión: las personas se vuelven a aquellos otros que saben que tienen gustos similares a los propios, pues encuentran de más valor esas opiniones. Para realizarlo computacionalmente, una vez existente la matriz de usuarios-artículos, se debe buscar en toda la base de datos todas aquellas filas que sean similares a la del usuario -es decir, se buscan aquellos perfiles que sean más parecidos al del usuario- y luego, en función de esto, se puede predecir la preferencia respecto a los elementos a recomendar; lo cual tiene también su ejemplo en la vida cotidiana: si sabemos que a los amigos cercanos de una persona les gusta cierto género musical, es probable que a esta persona también. Esta manera de pre-

decir las preferencias del usuario se conoce con el nombre genérico de *cálculo del vecino (o k-vecinos) más cercano(s)*. Es de notar que el algoritmo compara el perfil del usuario con el de todos los demás, lo cual ya vaticina una labor computacionalmente exigente.

Las diferencias en implementaciones radican en la función que se use para calcular la similitud entre los distintos perfiles. A continuación se definen algunas de las funciones más comunes para calcular qué tan similares son dos vectores de usuario (las filas en la matriz):

Distancia euclidiana: Es la función más común para medir las distancias entre dos puntos. Se expresa en la siguiente fórmula:

$$d = \sqrt{(x_2 - x_1)^2 - (y_2 - y_1)^2} \tag{2.13}$$

Sin embargo, la fórmula, tal como está, lo que hace es dar valores inversamente proporcionales a la similitud (esto es evidente ya que entre más similares son dos personas, más corta es la distancia). Lo que se espera, claro, es una función que dé valores *mayores* entre más grande sea la similitud, para este efecto, se puede invertir y ampliar la fórmula de esta manera:

$$\Delta = \frac{1}{1+d} \tag{2.14}$$

Así, se puede hablar del porcentaje de similitud entre personas.

Similitud de coseno: Si se entienden los perfiles de preferencias del usuario como un vector, se puede aplicar esta técnica extensamente utilizada en el modelo de espacio vectorial de recuperación de información: si el ángulo entre los vectores se hace más corto, el ángulo de coseno se acerca a la unidad (García, 2006). Se expresa de la siguiente manera:

$$\cos\theta = \frac{a \cdot b}{\| a \| \| b \|} \tag{2.15}$$

Coeficiente de correlación de Pearson: Definida con sencillez y elegancia en (Segaran, 2007) como

> La relación entre qué tanto cambian las variables juntas e individualmente.

Lo cual se traduce en lenguaje estadístico como la razón entre la covarianza de ambas variables y la desviación estándar de cada una. Se puede escribir así, para dos usuarios i y j que han calificado un conjunto común de elementos E (adaptada de (Sarwar et al., 2001):

$$corr_{i,j} = \frac{\sum_{e \in E}(P_{e,i} - \bar{P}_i)(P_{e,j} - \bar{P}_j)}{\sqrt{\sum_{e \in E}(P_{e,i} - P_i)^2}\sqrt{\sum_{e \in E}(P_{e,j} - P_j)^2}} \tag{2.16}$$

donde $P_{e,i}$ representa la ponderación del usuario i para el elemento e y $\bar{P}_i$ es el promedio de todas las calificaciones del usuario.

Una vez determinados los usuarios más similares, se procede a calcular cuánto le daría el usuario actual a los elementos que los otros usuarios similares han calificado. Para ello, se pueden usar dos métodos: *suma ponderada* y *regresión*.

En el método de suma ponderada, se suman los valores que los usuarios similares han dado al elemento actual, ponderándolos, usualmente, con una multiplicación, con la similitud que exista entre el usuario actual y los demás.

El método de regresión trata de ajustar una línea recta entre las ponderaciones para el elemento actual, lo cual tiene la ventaja de dar resultados más exactos, al tratar de ajustar a un mismo criterio valores que podrían estar muy distantes.

Con cualquier método, el resultado será la predicción del juicio que el usuario actual tendría sobre elementos que él no ha calificado aún; los elementos con mayor calificación consistirán en la recomendación.

Hasta el momento, el filtrado colaborativo discutido ha sido uno *orientado a los usuarios*. Sin embargo, como ya se mencionó, esto puede volverse computacionalmente difícil para grandes colecciones de elementos con un número considerable de usuarios. Una alternativa es calcular periódicamente los elementos que son similares entre sí y, a la hora de recomendar, en lugar de buscar entre todos los usuarios, sólo se busca entre todos los elementos similares y las correspondientes valoraciones. Los algoritmos involucrados son muy similares a su contraparte basada en usuarios, e implementaciones y resultados experimentales de este enfoque alternativo, así como una explicación más detallada, se pueden encontrar en Sarwar et al. (2001).

Filtrado colaborativo basado en modelos

Como se discutirá en el apartado 2.2.3.2, un problema de los sistemas basados en memoria es el escaso uso del historial del usuario. Los sistemas basados en modelos tratan de superar este problema: en vez de calcular las predicciones cada vez, construyen hipótesis de *por qué* los usuarios en una vecindad de preferencias han preferido un sistema y a partir de ésta calculan las predicciones para filtrar los elementos. Técnicas de aprendizaje de máquina, como las mencionadas en el apartado 2.2.2.1 en la página 36 se pueden utilizar para ese fin. La orientación a artículos en lugar de usuarios es un componente central de este tipo de sistemas(Mortensen, 2007).

2.2.3.2. Limitaciones

La orientación a la comunidad es un arma de doble filo: si bien la mayor parte de las recomendaciones son exactas, el sistema se basa en datos que son intrínsecamente inexactos: las opiniones de personas.

Las limitaciones más comunes, presentadas por (Mortensen, 2007; Fox et al., 2010), para este tipo de sistemas son:

Dispersión: Los sistemas implementados para grandes colecciones de productos probablemente tengan una gran cantidad de productos que han sido evaluados por pocos usuarios, resultando así en recomendaciones pobres para éstos.

Escalabilidad: Como se notó antes, los algoritmos convencionales para el cálculo de recomendaciones son computacionalmente costosos, lo cual puede tener un impacto significativo cuando la colección de artículos y usuarios crezca.

Aprendizaje: En los sistemas basados en memoria, nada se aprende del usuario más allá del momento en el que se le recomienda, por lo que cada vez que el usuario utilice el sistema, recibirá recomendaciones genéricas que no aprovechan en realidad su historial de consumo.

Preferencias inusuales: para usuarios que no tienen similitud con otros, las recomendaciones serían también pobres, dado que las funciones de predicción dependen de la existencia de más usuarios con preferencias similares.

Sobre-especialización: artículos de uso más común que otros tendrán más evaluaciones.

Arranque en frío: Sucede cuando el sistema aún no ha podido construir una matriz de usuarios-artículos que provea suficiente información. Esto se puede resolver con algoritmos heurísticos, como proponen Schein et al. (2002).

2.2.4. Recomendación orientada al conocimiento

Los dos grandes enfoques discutidos antes son los más usados a nivel comercial, pero ambos tienen una desventaja fundamental: requieren que se logre obtener suficiente información de los usuarios o artículos para comenzar a dar recomendaciones útiles, lo cual es algo difícil de superar: sin suficientes usuarios, no se tienen muchas evaluaciones de artículos, y, sin recomendaciones de artículos, no se logra conseguir una base suficiente de usuarios. Para superar este problema, se ha propuesto un tercer tipo de sistemas de recomendación: los *orientados al conocimiento* -introducidos en (Burke, 2000). Éstos tratan de *razonar* la motivación que subyace a la elección de cierto artefacto por el usuario, determinando qué atributos, y qué relaciones entre éstos, han influenciado la decisión para así determinar otros artefactos que podrían ser de interés al usuario. Nótese la diferencia entre estos sistemas y los orientados al contenido: mientras que los de contenido infieren un patrón de preferencias del usuario, los orientados al conocimiento no sólo saben del usuario, sino que también de los artefactos, y la combinación de estas dos fuentes de conocimientos permite un razonamiento (y no sólo una conjetura, como lo derivado del mero aprendizaje de máquina) de las necesidades del usuario. Este enfoque se deriva del *razonamiento basado en casos:* dado un problema, se utiliza una base de conocimiento de problemas anteriores para asimilar el problema nuevo con algunos precedentes y derivar una solución similar a la de éstos.

Pero, a diferencia de los otros, estos sistemas requieren una interacción más estrecha con el usuario: luego de las recomendaciones preliminares, ofrecen opciones implícitas de retroalimentación al usuario, como darle más valor a un atributo o ignorar otro, para poder así construir un modelo de conocimiento a medida que se navega entre las sugerencias, convirtiéndose también en agentes de navegación asistida. Como la intromisión está orientada a recabar datos cada vez más precisos del usuario a la vez que ofrece mejores opciones a éste, se compensa con el aporte de cada vez mejores sugerencias.

Otra desventaja de estos sistemas es que requieren del uso de técnicas avanzadas de ingeniería del conocimiento, convirtiéndolos en una opción arquitecturalmente costosa, además de exigir un uso exclusivo, lo cual es indeseable para implementaciones que requieran de poca intrusión en la actividad principal del usuario.

2.2.5. Sistemas de recomendación híbridos

Luego de examinar los dos enfoques convencionales, el orientado a contenido y el colaborativo, se puede notar que ambos tienen sus fortalezas y debilidades: los orientados al contenido son exactos, pero requieren un uso extenso y no toman en cuenta el aspecto social del uso de sistemas basados en internet; por otro lado, los colaborativos pierden exactitud a cambio de la implementación del saber colectivo de los usuarios, además de su característico costo computacional. Sería ideal, entonces, poder combinar la exactitud y eficiencia de los sistemas orientados al contenido con la inteligencia colectiva conseguida mediante un enfoque colaborativo.

2.2.5.1. Sistemas de recomendación colaborativos y orientados al contenido

Mortensen (2007) resume las combinaciones de ambos enfoques en cuatro tipos:

1. Combinar sistemas de recomendación separados: consiste en tomar las listas de sistemas distintos que utilicen cualquier enfoque y luego ordenarlas mediante otras técnicas de ordenamiento por relevancia.

2. Agregar elementos orientados al contenido a sistemas colaborativos: determinando similitud entre usuarios mediante sus perfiles y no en base a artefactos en común; aunque esto requiere que se recabe una base histórica aún mayor, permite recomendaciones personalizadas.

3. Agregar características colaborativas al enfoque orientado a contenido: consiste en representar los perfiles de usuarios como vectores de términos y luego usar técnicas de agrupamiento para asimilarlos.

4. Unificar en un solo sistema: esto incluye las dos técnicas anteriores en una implementación diseñada desde un inicio para este efecto, un ejemplo de un sistema unificado se puede encontrar en el sistema *Fab* de *Balabanović & Shoham (1997)*.

Agregar características de orientación al contenido a un sistema colaborativo se hace tradicionalmente agregando reglas relacionadas con el contenido del dominio a la matriz de usuarios-artículos -por ejemplo, si un artículo tiene mala ortografía, la regla puede decidir darle una puntuación desfavorable; aunque esto resuelve el problema de la dispersión y el arranque en frío -que artículos nuevos no son evaluados, ni recomendados, por carecer de evaluación- al hacer que las reglas sean los primeros evaluadores de los elementos. Baudisch (1999), sin embargo, ofrece una alternativa: en vez de *combinar* las técnicas, propone *unirlas* , en vez de agregar una a la otra[8].

[8] *Join*, el término en inglés para el concepto de unión en b.d. relacionales quizá clarifique el sentido de la frase

Para ilustrar esta unificación, se vale de las definiciones alternas para ambos sistemas -mencionadas en las secciones 2.2.2 y 2.2.3 para hacer una unión de la representación de entidades, que se presenta en la figura 2.6y que ofrece las aplicaciones ilustradas en la tabla 2.8; en ésta, al contar con todos los datos de ambos enfoques en un sistema unificado, se puede ir más allá del filtrado de información, pero la columna pertinente al tópico de la recomendación es la central: nótese que resulta en un filtrado colaborativo orientado a objetos y a usuarios a la vez que permite contar con recuperación de información en base a perfiles.

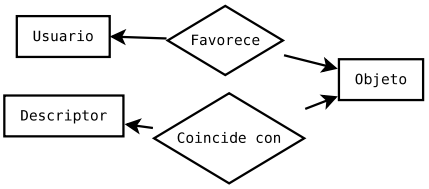


Figura 2.6: E-R Híbrido resultante de la combinación de sistemas colaborativos y orientados al contenido. Adaptado de Baudisch (1999)

E/S	Descriptor	Objeto	Usuario
Usuario	Exportación de perfiles	Filtrado colaborativo activo	Emparejador
Objeto	Indización	Búsqueda mediante ejemplos Filtrado colaborativo automatizado	Búsqueda de expertos
Descriptor	Tesauro	Recuperación y filtrado de información	Búsqueda de expertos Importación de perfiles

Cuadro 2.8: Aplicaciones de un sistema híbrido de recomendación, adaptada de Baudisch (1999). E/S equivale a "Entradas/Salidas"

2.2.5.2. Sistemas de recomendación colaborativos y orientados al conocimiento

Aunque los sistemas orientados al conocimiento pueden encontrar recomendaciones más adecuadas, carecen del importante elemento social de los sistemas colaborativos, filtrando menos elementos que éstos, al carecer del contexto que la comunidad aporta. Por ello(Burke, 1999) propuso, mediante una tabla como la 2.9, un híbrido entre estos

Técnica	Ventajas	Desventajas
Orientada al conocimiento	A. No se necesita mucha información inicial. B. Retroalimentación detallada y cualitativa de preferencias. C. Sensible a la variación a corto plazo.	H. Ingeniería del conocimiento I. La habilidad de sugerir es estática
Filtrado colaborativo	D. Identifica nichos con precisión. E. No se necesita conocimiento específico al dominio. F. La calidad mejora con el tiempo. G. Las recomendaciones son personalizadas.	J. La calidad depende de los datos históricos K. Sujeta a anomalías estadísticas L. Reacciona lentamente a la variación a corto plazo.
Híbrido Ideal	A, B, C, D, F, G	H

Cuadro 2.9: Comparación entre sistemas colaborativos y orientados al conocimiento.

dos enfoques.

Estos híbridos, entonces, paliarían el problema de la poca atención a las tendencias entre varios usuarios, ofreciendo recomendaciones exactas mediante la base del conocimiento y filtradas mediante lo que otros usuarios han hecho con los artefactos del conjunto de resultados, dotando a los sistemas basados en el conocimiento de la alta precisión y la orientación social de la que inherentemente carecen.

2.2.6. Evaluación de un sistema de recomendación

Se puede evaluar un sistema de recomendación en base a los resultados estimados mediante alguna técnica estadística contra los resultados obtenidos, determinando así si los conjuntos de resultados obtenidos por el sistema cumplen con ciertos requerimientos cuantitativos y cualitativos estáticos. Pero existen otras consideraciones a medir, como el nivel de confianza del usuario en el sistema y qué tanta utilidad tienen los resultados para él. Esta distinción es hecha en Hayes et al. (2002)cuando califica al primer tipo de mediciones como *fuera de línea* y a las postreras, como *en línea*.

2.2.6.1. Medidas de evaluación *fuera de línea*

Un componente importante de un sistema de recomendación es el que obtiene de la colección de artefactos aquellos que se filtrarán para luego sugerirlos; como ya se ha mencionado, este componente es, en realidad, un sistema de recuperación de información, de

ahí que parte de la evaluación se pueda hacer en términos de precisión y exhaustividad del subconjunto de resultados. Además, si se tiene en mente que las recomendaciones tienen algún sistema de rango para el filtrado, medidas como la curva ROC, la precisión media, la *precisión-R* y la de *once puntos*, resumidas en Thorsten Brants & Google (2004) y explicadas con mayor detalle en Schütze et al. (2009) pueden ser aplicadas. Es de enfatizar que lo que se mide con la curva ROC es la relación entre precisión y exhaustividad, pero una característica interesante de los sistemas de filtrado de información es que podrían sugerir más elementos de los que un usuario encontraría verdaderamente útiles; para normalizar esta situación, se puede introducir una restricción en la presentación de resultados: sólo mostrar una cantidad constante de recomendaciones a todos los usuarios, las cuales consistirían en las k mejores sugerencias. Un sistema donde se ha introducido tal restricción se puede medir con una curva CROC (curva ROC de cliente), como sugiere Schein et al. (2005).

Asimismo medidas de evaluación de aprendizaje de máquina en sistemas orientados al contenido se podrían aplicar, como la exactitud[9], el afinamiento mediante matrices de confusión o la validación cruzada (más sobre estos métodos de valoración se puede encontrar en los capítulos sobre aprendizaje de máquina de Russell & Norvig (2003)).

Y para los sistemas de colaboración, vistos como un problema de regresión, se podrían usar medidas de correlación entre los resultados esperados y los estimados, así como otros cálculos de error, como ser la media cuadrática. En Herlocker et al. (2004) se examinan a fondo las métricas antes mencionadas para la medida de este tipo de sistemas, específicamente los de filtrado colaborativo.

De las anteriores consideraciones se pueden destacar tres medidas extensamente utilizadas en la investigación de la evaluación de sistemas de recomendación:

Exactitud: que mide qué tan bien clasifica un sistema (o predice) comparando los resultados obtenidos con resultados estimados o evaluados por usuarios.

Cobertura: que mide qué tanto de la colección puede ser sugerido por el sistema, puesto que muchos de los artículos podrían no ser nunca evaluados o no ser semánticamente relevantes a ningún perfil de usuario.

Tasa de aprendizaje: qué tan rápido pueden aprender los algoritmos la cantidad suficiente de información como para empezar a dar recomendaciones significativas.

Una de las mayores limitaciones de este tipo de análisis de desempeño es el sesgo en el conjunto de resultados a evaluar: usualmente, los usuarios sólo han calificado lo que ellos favorecen, y no dan retroalimentación explícita a aquellos artefactos que no encuentran útiles. Se puede mencionar también en contra de las métricas puramente estadísticas que ignoran la *utilidad* de los resultados: aunque ciertos conjuntos puedan significar alta precisión o exactitud, quizá no sean -por ser muy obvios- de gran aporte al usuario como observanAdomavicius & Tuzhilin 2005, esto se relaciona con los conceptos de *novedad* y *serendipia:* si el artículo sugerido es uno que no se le hubiera ocurrido, por no conocer -novedad- o ignorar su utilidad para el contexto presente -serendipia, al usuario.

[9] definida en la sección 2.1.5.1

2.2.6.2. Medidas de evaluación *en línea*

Además de las limitaciones mencionadas, y aunque el análisis puramente sistemático del desempeño de un sistema de recomendación pueda ser suficiente en ciertos casos, el papel del usuario no puede ser dejado de lado del todo, por ello, en la evaluación de su sistema Chen & Wu (2008) se centran en la satisfacción del usuario utilizando dos evaluaciones:

Satisfacción del usuario respecto al sistema: se refiere a la interacción del usuario con el sistema mismo (mejor dicho, con la interfaz de usuario), se proponen cinco medidas:

1. Facilidad de uso.
2. Comprensibilidad de las funciones del sistema.
3. Fiabilidad.
4. Facilidad del aprendizaje de uso.
5. Tiempo de respuesta.

Satisfacción del usuario respecto a la información: o qué tan satisfecho está el usuario con los resultados que el sistema le presenta. Así como en el aspecto anterior, se presentan cinco medidas, respecto a las recomendaciones como tales:

1. Qué tanto de los requerimientos del usuario se ve cubierto.
2. Confiabilidad.
3. No-ambigüedad.
4. Precisión.
5. Inteligibilidad.

Asimismo, se puede argüir que una evaluación fuera de línea puede estar ignorando tanto el contexto de uso del usuario como el nivel de confianza que le tiene al sistema (si las recomendaciones le son útiles y le instan a continuar utilizando el sistema), de ahí que un marco de evaluación en línea basado en la comparación directa de rendimiento entre dos métodos de recomendación y la retroalimentación de usuarios reales sea propuesto por Hayes et al. (2002):

En primera, la evaluación se ha de llevar a cabo en una aplicación en línea utilizada por usuarios reales, en la cual los dos motores de recomendación distintos se utilizan mediante la misma interfaz de usuario, de manera que la satisfacción del usuario respecto al sistema se mantenga constante -ya que el usuario final ignora cuál de los sistemas está utilizando- puesto que, en última instancia, se desea medir solamente el desempeño de las recomendaciones. Para ello se propone una arquitectura -ilustrada en la figura 2.7- que defina:

Los recursos disponibles: una interfaz de aplicación (API) definiendo los recursos a los cuales los sistemas tendrán acceso. Nótese que es importante normalizar esto, puesto que los sistemas orientados la contenido utilizan datos semánticos de los recursos y los de filtrado utilizan datos de evaluaciones de usuarios; esto se puede normalizar mediante el uso de un lenguaje común para representar y estructurar los datos.

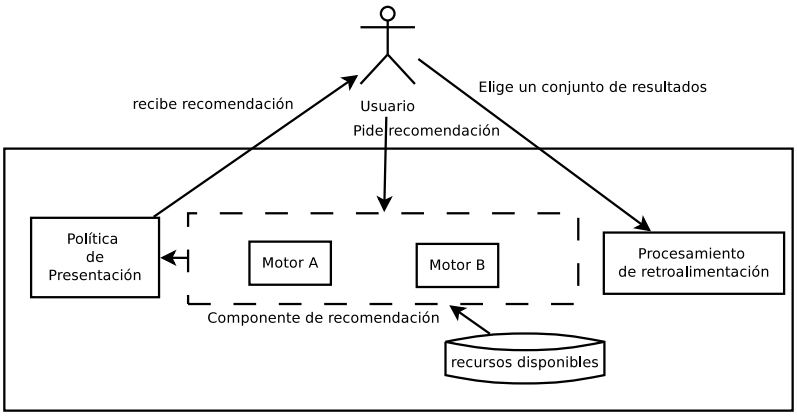


Figura 2.7: Comparación en línea de sistemas de recomendación. Adaptada de Hayes et al. (2002)

Los métodos de sistemas de recomendación: otra interfaz, que defina qué métodos deberá implementar cada sistema.

Una política de presentación: definiendo cómo mostrar los resultados al usuario. Puesto que éstos pueden ser una combinación de las sugerencias de los dos métodos puestos a prueba, se sugieren tres alternativas:

1. Un conjunto combinado de los resultados de ambos.
2. Un conjunto contrastado, mostrando los dos conjuntos de resultados claramente separados -de manera que se sepa que provienen de fuentes distintas, pero sin decir cuál ha dado qué conjunto.
3. Conjuntos en cascada, mostrando, cuando un usuario pide recomendaciones, los resultados de un sistema y, cuando pida recomendaciones de nuevo, los del otro.

Lineamientos de retroalimentación: cuáles acciones del usuario, implícitas o explícitas, se considerarán un signo de preferencia

Una métrica de comparación: mediante la cual analiza la retroalimentación y determinar un ganador.

Téngase en cuenta que, las condiciones en una evaluación de dos métodos debe ser lo más justa posible: técnicas que requieran una sola interacción, como los sistemas colaborativos convencionales, deberían ser comparadas con otras que también requieran una sola interacción; mientras que otros enfoques, como ciertas implementaciones de recomendaciones basadas en el conocimiento -las cuales son más *conversacionales*- deberían ser evaluadas frente a otras de su clase. De igual manera, debe también tenerse en cuenta que existen métodos que requieren un tiempo de adaptación o aprendizaje, como la mayor parte de implementaciones orientadas al contenido o de filtrado colaborativo.

Como conclusión, parece conveniente citar lo que se menciona en Herlocker et al. (2004):

> "Un sistema [de recomendación] es útil en la medida en que ayude a un usuario a llevar a cabo sus tareas."

Por ello, se deben elegir las medidas y experimentos de evaluación más acordes al propósito del sistema.

2.3. Sistemas de recuperación oportuna de la información

Relativamente poca investigación se ha hecho respecto a sistemas de recomendación que asistan recuperando información útil a la tarea del usuario sin obstruirle. Pero poco no es ninguno, y, así, podemos encontrar una definición de este tipo de sistemas en la tesis que sentó las bases de este campo, la de Rhodes (2000a):

> "Un agente de recuperación oportuna de información es un programa que proactivamente encuentra y presenta información basada en el contexto local de una persona de una manera fácilmente accesible y a la vez no intrusiva."

Los agentes descritos en la tesis antes mencionada reciben ya un nombre propio: sistemas de recuperación oportuna de información (en inglés, *JITIRS: Just-in-time Information Retrieval Systems).* Nótese que, como se había establecido antes, la principal utilidad de un sistema de esta especie es encontrar la necesidad latente de información: presentarle información al usuario cuando éste ni siquiera haya pensado en buscarla, o ni supiese que ésta existía.

Un agente de recuperación oportuna de información difiere de un sistema de recuperación de información tradicional en que la búsqueda no es la tarea principal, las consultas son potencialmente más largas -el contexto local del usuario, por el cual entiéndase cualquier contexto, desde lo que está escribiendo en un procesador de texto hasta el lugar geográfico donde está; y que la principal medida de evaluación de un sistema así es la *utilidad* que tienen los documentos -contrario a los sistemas de recuperación de información, que se miden por la relevancia, expresada en la precisión y exhaustividad, de sus resultados.

Asimismo, Rhodes (2000a), establece tres criterios a tener en cuenta a la hora de desarrollar un *JITIR:*

¿Cómo afecta el sistema la manera en la cual una persona busca y usa información?
Tomando la teoría de la ciencia cognitiva, se establece que el sistema debería poder permitir la serendipia: encontrar información que resulta útil y de la cual no se sabía; y poder reducir el costo: encontrar información de la cual se sabía, pero se consideraba demasiado el esfuerzo que hubiera implicado buscarla por cuenta propia.

¿Cómo puede el sistema encontrar información que resultaría útil en base al contexto?
Es en este punto donde entra en juego la recuperación de información: se ha de elegir un modelo que trascienda la mera relevancia para poder ofrecer documentos que

sean de utilidad -lo cual se puede aproximar utilizando técnicas de alta precisión y trayendo a colación el filtrado: basarse en el contenido del contexto actual aunado a las preferencias del usuario o de un grupo de usuarios para eliminar documentos que se consideren, aunque relevantes, inútiles. Enfoques orientados al conocimiento del dominio específico pueden ser de utilidad, por ejemplo, en el sistema *FIXIT* Hart & Graham (1997) se valieron de la existencia de una red de inferencia para el sistema principal -un sistema experto de reparación de artículos de oficina- para poder usar esas mismas reglas en la recuperación de documentación útil a la tarea.

¿Cómo debería el sistema presentar los resultados? El diseño de la interfaz debería ser tal que los resultados sean fácilmente accesibles y evaluables, pero que no interfiera con la tarea principal; así, no debería distraer completamente de ésta pero la atención debería poder dividirse entre ambas interfaces. Es práctica común en el diseño de sitios web el ubicar los elementos más importantes a la izquierda, por lo que, en principio, se podrían ubicar los resultados más a la derecha del lugar donde se realice la tarea principal, práctica adoptada por el sistema *PIRA* (Gruzd & Twidale, 2006). Asimismo, se ha probado útil ofrecer un sumario del documento o una vista previa en caso de que el artefacto consista de multimedia (Schütze et al., 2009).

Nótese que un sistema de recomendación proactivo no es más que un sistema de recomendación fuertemente orientado al contenido que se adapta al contexto del usuario y ofrece sugerencias sin que éste haya elegido antes otros artículos de la colección. El proyecto *À propos (Puerta Melquizo et al., 2007)* es un ejemplo de un sistema de este tipo: asiste a escritores construyendo consultas en base a lo que están escribiendo en el momento, las cuales alimenta a distintos motores de búsqueda; por otro lado, *Scienstein (Gipp & Hentschel, 2009),* otro sistema que se puede considerar de este tipo, se vale también de las citas que el autor ha hecho para recomendar materiales relacionados, además de incorporar técnicas de filtrado de información para presentar los resultados.

2.3.1. Determinación del contexto para sistemas basados en texto

Según parece, la capacidad del sistema para determinar el contexto local del usuario es fundamental para que un agente *JITIR* sea de verdadera utilidad. Aunque para sistemas basados en contextos extraordinarios, como aquellos que se pudieran basar en información geográfica, la información va más allá de lo que se introduzca en una aplicación, para sistemas orientados a tareas puramente documentales, asumir que todo el contexto necesario se encuentra en el texto introducido por el usuario es suficiente.

Pero ¿cómo determinar, de lo que el usuario escribe, qué se puede considerar importante en la identificación del contexto y la construcción de las consultas para la búsqueda de documentos a recomendar? Esta pregunta parece encontrar su respuesta en un área que, aunque poco conocida, es vital

En primer lugar uno podría argumentar que, dado algún saber previo respecto del usuario o del dominio del conocimiento en el que se desenvuelve, se podrían extraer términos discriminantes del contexto local y con éstos realizar las consultas, ya sea con

técnicas de aprendizaje de máquina -por ejemplo, Uzun (2006), propone el uso de un clasificador bayesiano- o, si se dispone de una colección de documentos en el contexto local o el perfil del usuario, se pueden tomar prestadas técnicas de la recuperación de información: el modelo de vector espacial (Renz et al., 2003), ontologías construidas con el análisis formal de conceptos (Cimiano et al., 2005) o la aplicación del modelo de análisis de semántica latente para determinar los términos más definitivos y hasta se puede disponer de la lingüística computacional, relacionando la importancia de una palabra con su rol gramatical. Una comparación entre los enfoques estadísticos y lingüísticos para la extracción de términos clave se puede encontrar en Kageura & Umino (1996).

Y cuando el conocimiento previo es nulo o escaso, un enfoque lingüístico parece cobrar más importancia, según el trabajo de Matsuo & Ishizuka (2003), quienes demuestran un algoritmo que determina palabras clave con un rendimiento comparable al de la ponderación *tf*idf* (véase 2.1.4.3).

Existen servicios basados en web para la extracción de términos clave, entre los cuales, uno de los más populares es el *Term Extraction Web Service*[10], utilizado por el sistema *PIRA* (Gruzd & Twidale, 2006). Asimismo, existen librerías que implementan esta funcionalidad mediante la identificación de roles gramaticales (una implementación en el lenguaje de programación python se encuentra disponible en `http://pypi.python.org/pypi/topia.termextract/`.

2.4. Construcción de perfiles de usuario

De este tema ya se habló cuando se discutieron los sistemas de recomendación (sección 2.2), pero se considera pertinente ahondar un poco más en el tema, dada la naturaleza de los sistemas que se discuten aquí. En este apartado se mostrará cómo la construcción de un perfil de usuario puede ayudar a la tarea de la recuperación de información, con miras a aportar al área de la recuperación oportuna de la información.

El área de investigación de modelado de perfiles de usuario no es nueva, de hecho, cualquier sistema se crea con un tipo de usuario en mente. McGowan (2003) distingue tres maneras de tratar el problema del perfil de usuario:

1. El enfoque *canónico*. En el cual el diseñador del sistema crea el mismo para un solo tipo de usuario estándar, este prototipo se puede definir arbitrariamente o mediante estudios de grupos de usuarios reales.

2. El enfoque *explícito*. En el cual el usuario tiene la responsabilidad de responder preguntas o ajustar parámetros para definir su perfil, en base a esto, el sistema se adaptará a él. Este enfoque es famoso en sistemas de recomendación comerciales o redes sociales.

3. El enfoque *automático*. En el cual el sistema se basa en las interacciones con el usuario para inferir el perfil. Una primera manera de implementar este enfoque es tener categorías de usuarios preexistentes (como novato, intermedio, y experto,

[10] *http://developer.yahoo.com/search/content/V1/termExtraction.html*

por ejemplo) y luego utilizar tales interacciones para *clasificar* al usuario en la taxonomía dada. Otra manera, la que es relativamente nueva en la investigación, es la de considerar el perfil individual como una base de conocimiento que puede evolucionar y que regirá la interfaz entre usuario y sistema. Es este último enfoque, el de considerar al perfil en sí como la directriz principal para el comportamiento del sistema el que se definirá y evaluará en el resto de esta sección.

2.4.1. Definiciones preliminares

Antes de entrar en materia, es necesario definir ciertos conceptos:

Perfil: de manera general, el *perfil* de un usuario se define como las circunstancias que tienen influencia en su necesidad de información a la hora de utilizar un sistema de RI y en el consiguiente juicio de los resultados de búsquedas. En la literatura, los términos *perfil, contexto y situación* se pueden utilizar indistintamente; sin embargo, en este trabajo se entenderá el perfil como la representación de los intereses a largo y mediano plazo del usuario. El perfil puede consistir de múltiples *dimensiones,* las principales, identificadas por Bouzeghoub & Kostadinov (2005), son:

1. Información Personal: la parte estática del perfil, aquí se encontrarán datos tales como país, lenguaje de preferencia, edad, etc.
2. Dominio de interés: es la dimensión más explorada en la investigación; consiste en una representación de los intereses del usuario, por consiguiente, esta dimensión es dinámica.
3. Preferencias de presentación: los atributos que el usuario favorece en la interfaz (como los colores en los perfiles de algunas redes sociales) o los menús que se muestran, en el caso de muchas aplicaciones de escritorio.
4. Historial de interacciones: toda información recolectada a través de la retroalimentación del usuario, podría utilizarse para construir el dominio de interés.
5. Calidad esperada: alguna medida cuantitativa de la relevancia intrínseca de los resultados; un usuario profesional podría preferir sólo documentos debidamente publicados y revisados mientras que otro prefiera actualizaciones de *blogs* y redes sociales.
6. Seguridad: o las preferencias de privacidad respecto a las otras dimensiones.

Contexto: igual que el perfil, el contexto engloba las circunstancias de la "intención del usuario para búsqueda de información" (Sieg et al., 2007). Pero a diferencia del perfil, el contexto solamente se refiere a la *sesión actual* de uso del sistema, por lo que representa los intereses a corto plazo.

Sesión: se refiere a cada vez que el usuario utilice el sistema para satisfacer una necesidad de información, puede incluir cualquier cantidad de reformulaciones de la consulta, siempre y cuando la tarea tenga siempre la misma finalidad.

Ontología: "es una especificación explícita de conceptos y las relaciones que puedan existir entre ellos" Sieg et al. (2007). En muchas implementaciones prácticas, esto se modela mediante una relación jerárquica en forma de un grafo dirigido acíclico o un árbol, de manera que los conceptos puedan ser sub-conceptos de otros -mediante al relación de herencia "es un".

2.4.2. La definición del perfil

Como se ha notado, los intereses a largo plazo de un usuario son los que permitirán a un sistema de recuperación de información adaptarse a éste y retornar resultados cuya medida de relevancia sea cada vez menos genérica. De ahí que la manera en que el perfil se represente y manipule sea fundamental. Tamine-Lechani et al. (2007) identifican tres etapas para la consecución de esta tarea:

Representación la manera en que se representen internamente las preferencias a largo plazo del usuario tendrá un impacto directo en el sistema, puesto que es la base de cualquier otra operación. La manera más difundida de representación es la vectorial, heredada el modelo de espacio vectorial. Ahora bien, esta representación puede consistir en un solo vector con cada término clave ponderado en base a las interacciones del usuario o en varios vectores (o *clases de vectores*) no relacionadas u organizadas jerárquicamente que representen los distintos dominios de interés del usuario. La representación vectorial es utilizada por Tamine et al. (2006) y las clases de vectores se implementan en el sistema de McGowan (2003). Otra manera de representar un perfil es semánticamente: con el uso de ontologías, para así modelar las relaciones entre los distintos conceptos -y ya no palabras clave. Las ontologías se pueden construir a partir de la colección documental, pero los sistemas que han implementado este enfoque se valen de taxonomías de conceptos ya existentes, como el *open directory project* (Daoud et al., 2008; Sieg et al., 2007), wikipedia (Ramanathan et al., 2008) o wordnet (Semeraro et al., 2005). Un último enfoque utiliza varias dimensiones del perfil, como el modelo de seis dimensiones propuesto por Bouzeghoub & Kostadinov (2005) o el de dos dimensiones -los intereses y el historial de interacciones- implementado por Tamine-Lechani et al. (2007).

Construcción esta etapa se ocupa de la instanciación del perfil en una sesión y de cómo se obtendrá la información que servirá para actualizarlo. En prácticamente todas los sistemas que utilizan modelos de perfil, estos datos se obtienen de la retroalimentación del usuario (en la mayoría de los casos, implícita). Esta retroalimentación se puede tratar mediante el algoritmo de Rocchio o usarse de entrada para un algoritmo de clasificación, con el fin de mejorar los resultados en la sesión actual (Schütze et al., 2009).

Evolución esta etapa es la que utiliza la información obtenida de sesiones de uso para adaptar el perfil a los intereses cambiantes del usuario; y es esta misma etapa la más íntimamente ligada a la representación elegida. En los enfoques semánticos, los conceptos se pueden ponderar mediante *puntajes de preferencia*, actualizando éstos

después de cada sesión, como en el trabajo de Sieg et al. (2007) o con el uso de ponderaciones que "envejecen", como lo proponen Daoud et al. (2008). En el caso de representaciones vectoriales, se puede utilizar un enfoque estadístico y actualizar las probabilidades de relevancia de los documentos en relación a los términos y los intereses que éstos representan, como en el enfoque orientado a diagramas de influencia de **?**.

2.4.3. La utilización del perfil

Se pueden determinar tres enfoques en la recuperación y presentación de la información utilizando los perfiles de usuario. El primero, requiere dos etapas: primero, se obtienen los resultados de un agente externo de recuperación de información con la consulta que el usuario mismo ha construido y luego, los documentos se filtran (o reordenan). Este enfoque en dos pasos ha sido implementado por Sieg et al. (2007) y Daoud et al. (2008). La ventaja de este modelo radica en poder valerse de servicios externos o implementaciones de algoritmos de recuperación de información, concentrándose más bien en la etapa de filtrado; la desventaja estriba en el costo computacional: para contextos complejos, la determinación de la consulta podría volverse computacionalmente costosa y ésto, aunado al tiempo que toma la búsqueda en sí, podría tornarse indeseable. Para resolver este obstáculo, se puede optar por sólo considerar una cantidad fija de los resultados más relevantes devueltos y de éstos sólo tomar en cuenta los resúmenes que el agente devuelve, y no todo el documento.

Por otro lado, se puede hacer la recuperación de información en una sola etapa: reescribir la consulta del usuario, como en el sistema propuesto por Koutrika & Ioannidis (2005) o agregar el perfil al espacio documental, construyendo así el sistema de RI con la personalización incluida desde un principio, como propone Semeraro et al. (2005) al representar los documentos como clases sinonímicas y no como términos individuales, o se puede seguir lo establecido por **?** e incluir las probabilidades entre términos, preferencias y documentos en la evaluación de relevancia. Este enfoque es en realidad, equivalente a un sistema de recomendación orientado al contenido. Con este enfoque se gana en costo computacional, pero se requiere la implementación completa del componente de búsqueda e indización.

Por último, se puede sencillamente utilizar el perfil para encontrar usuarios similares al presente y evaluar los resultados en base a los perfiles de éstos, como se encuentra documentado en el trabajo de Ahn et al. (2005). Como es evidente, este enfoque equivale a los sistemas de recomendación colaborativos.

3 Análisis y Diseño

3.1. Introducción

En este capítulo se documentan los distintos artefactos producidos a la hora de analizar y diseñar el sistema propuesto en este proyecto. Por cuestiones cronológicas, ambas etapas se han agrupado aquí, mereciendo la etapa de implementación y evaluación un capítulo aparte. Como se imaginará, el sistema no consiste en un sistema de información a gran escala, con una larga lista de requerimientos, casos de uso y escenarios de interacción; sin embargo, ninguna empresa de ingeniería debe hacerse sin ningún tipo de definición previa del derrotero a seguir.

El desarrollo clásico de sistemas de información es un proceso lineal que incluye seis etapas: compilación de requisitos, análisis, diseño, implementación, mantenimiento y retiro. Sin embargo, este proceso ideal ha sido desplazado en los últimos años por un enfoque iterativo: *el proceso unificado*. Éste consiste en cuatro fases: iniciación, elaboración, construcción y transición. En cada fase se llevan a cabo, en mayor o menor medida, los flujos de trabajo de revisión de requisitos, análisis, diseño, implementación y pruebas (Schach, 2004). Este enfoque iterativo es mucho más realista y efectivo que el enfoque tradicional, pero requiere de una gran cantidad de tiempo y una organización tales que se puede probar demasiado costoso para proyectos pequeños o de corta duración, características típicas del desarrollo web, que es la clase de desarrollo a llevar a cabo en este proyecto. Por ello, una metodología adaptada al desarrollo de aplicaciones web de este modelo iterativo, como la propuesta en Pressman (2006) se utilizará en el presente proyecto.

3.2. Análisis

En esta etapa se definirán la motivación, alcance, audiencia y contribución del sistema desarrollado. El producto final de esta fase será un resumen de los requisitos funcionales y no funcionales, en la metodología elegida, la identificación de estos se hace en cuatro etapas: los análisis de contenido, de interacción, de función y de configuración. Por mor del orden, y para fundamentar sólidamente las etapas subsiguientes, cada análisis se discutirá brevemente en los apartados siguientes, finalizando con el listado de requerimientos. Antes de proseguir, sin embargo, se contestarán tres preguntas que servirán de trasfondo a estos análisis:

1. ¿Cuál es la principal motivación para la aplicación? Siendo la aplicación la implementación de un proyecto académico, la principal fuerza tras el desarrollo de la misma es determinar si un sistema que pueda recomendar recursos útiles a un usuario que lleva a cabo una tarea de planificación o documentación pedagógica es

verdaderamente una ganancia para los usuarios, tanto los docentes que lo utilicen directamente como los estudiantes que se beneficien de los recursos aportados.

2. ¿Cuáles son los objetivos que debe satisfacer la aplicación? En suma, la aplicación debe trascender los sistemas de recuperación oportuna de la información equivalentes al basarse en una representación más fidedigna del contexto local del usuario, así como en la inteligencia colectiva y las contribuciones activas de los usuarios para determinar los mejores recursos a presentar.

3. ¿Quién utilizará la aplicación? Cualquier usuario del sistema anfitrión - el cual será, para la implementación primera, [1]- que haya creado un curso y se encuentre realizando actividades de documentación en el mismo.

3.2.1. Sumario de requisitos

En base al objetivo y la audiencia del proyecto, se resumen a continuación los requisitos a tener en las siguientes etapas del análisis y diseño, redactados y organizados siguiendo las pautas propuestas en Arlow & Neustadt (2005).

3.2.1.1. Requisitos funcionales

1. El sistema deberá tratar lo que el usuario escriba en la interfaz de documentación como el contexto local, representándolo como los demás documentos de la colección, con el fin de realizar una búsqueda.

2. El sistema deberá tomar los documentos que el usuario aporte como parte de su actividad de planificación y los deberá procesar y representar como los demás documentos, con el fin de agregarlos a la colección y realizar la búsqueda. Los documentos y lo escrito por el usuario en su sesión de planificación consistirán en el *contexto inmediato*.

3. El sistema deberá mantener un perfil de curso y del usuario con el fin de tener éste como base contextual para las recomendaciones.

4. El sistema deberá determinar cuándo es el momento idóneo de iniciar la recomendación o de hacer una nueva, ya sea en base al tiempo pasado desde el inicio de la sesión, la cantidad de información aportada por el usuario o la diferencia entre el último contexto inmediato que creó una búsqueda y el actual.

5. El sistema deberá realizar una búsqueda y filtrado de documentos en base a las representaciones que haya hecho del contexto del usuario, entendiéndose por contexto tanto la base contextual como el contexto inmediato.

6. El sistema deberá determinar cuándo se utiliza un documento recomendado, con el fin de tomar esta acción como retroalimentación y así enriquecer el perfil de la base contextual.

[1] `escolarea.com`

7. El sistema deberá indexar y organizar los documentos que obtenga, ya sea mediante los aportes de los usuarios o una búsqueda periódica de nueva información.

3.2.1.2. Requisitos no funcionales

1. El sistema deberá reflejar la separación entre el sistema anfitrión, el que extrae los datos de entrada y presenta los de salida y el componente principal, el que realiza las búsquedas y filtrado e indexa y mantiene la colección documental. Tal separación se logrará mediante el uso de una interfaz independiente del sistema anfitrión que reciba y devuelva los datos en un formato serializado (json, de preferencia, o xml).

2. El sistema, en el lado del servidor, estará escrito en el lenguaje de programación python. En específico, se utilizará el *framework* de desarrollo web django[2].

3. El sistema, en el lado del cliente, estará escrito en el lenguaje de programación javascript, con la librería jquery[3].

4. El sistema deberá apegarse a los estándares de diseño de w3[4], utilizando (x)html y css válidos.

5. El sistema utilizará, para el almacenamiento, el servidor de bases de datos postgresql[5].

6. El sistema funcionará en un servidor Linux.

7. El sistema deberá devolver los primeros resultados de las recomendaciones en menos de diez segundos, tal cantidad de tiempo es aceptable, dada la naturaleza secundaria y no intrusiva del sistema (Nielsen, 1994).

3.2.2. Casos de uso

Se resumen a continuación los distintos casos de uso del sistema. Nótese que la escasez de éstos se debe a que la mayor funcionalidad del sistema radica en el trasfondo algorítmico de la indización, búsqueda y filtrado. Un diagrama UML que los ilustra se presenta en la figura 3.1.

Inicia sesión: este caso de uso se refiere al comienzo de la sesión de documentación. En este momento, se conoce el curso para el cual se planifica y las sesiones previas, el usuario ha comenzado a escribir los detalles principales de la sesión. El sistema debe tener en cuenta que el usuario quizá se arrepienta y abandone la página o reformule completamente el contexto inmediato.

[2] `www.djangoproject.com`
[3] `http://jquery.com/`
[4] `http://www.w3.org/`
[5] `http://www.postgresql.org/`

Documenta: el usuario ha iniciado sesión, se ha creado o recuperado el perfil y el contexto inmediato ha sido determinado por el sistema para formular consultas a partir de la conjunción de éstos (el *contexto local*). El sistema se vale del componente de búsqueda y filtrado para obtener recomendaciones y mostrarlas. El sistema debe saber cuándo el contexto inmediato difiere significativamente del último al cual se hicieron recomendaciones para volver a iniciar el proceso.

Agrega Documentos: el usuario elige incluir documentos en su sesión. El sistema debe incluirlos como parte del contexto inmediato y realizar nuevas recomendaciones. Dada la posible gran cantidad de texto a procesar aportada por estos documentos y las restricciones de tiempo inherentes al propósito del sistema, es posible que el sistema deba elegir basarse sólo en los títulos o en la existencia previa de los documentos en la colección para la búsqueda y el filtrado -nótese, sin embargo, que, al terminar la sesión, los documentos *sí* son considerados para la construcción del perfil.

Usa Recomendación: el usuario decide incluir un documento de los recomendados a su sesión, el sistema habrá de tener esto en cuenta a la hora de construir el perfil. Dado que en cualquier momento el usuario puede descartar un documento, el sistema debería agregar tal retroalimentación al perfil solamente cuando se considere idóneo (a la hora de cerrar la sesión, por ejemplo).

Cierra sesión: en este momento, el usuario se prepara para abandonar la sesión, por lo que el sistema puede proceder a tomar el contexto de la misma y agregarlo a su conocimiento del perfil del curso. Asimismo, el sistema agregará los documentos que el usuario haya incluido en esta sesión a la colección documental, con la consiguiente indización de los mismos y el procesamiento necesario para la expansión del perfil del curso.

Actualiza Colección: tanto el administrador como el sub-sistema que obtiene documentos externos -el *agente recolector*- pueden agregar una cantidad significativa de éstos a la colección, lo que implica la indización de los mismos.

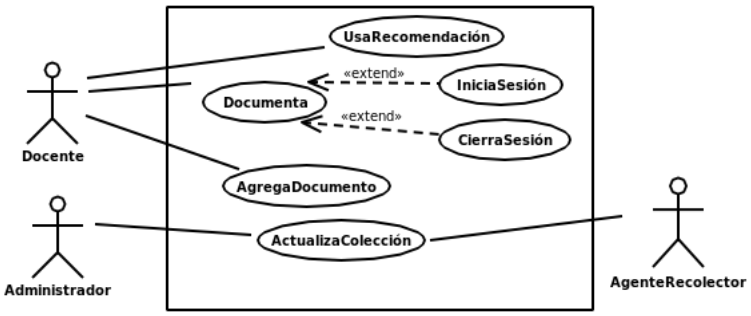


Figura 3.1: Diagrama de casos de uso

3.2.3. Análisis de contenido

3.2.3.1. Objetos del contenido

Como se puede deducir, el único contenido que el sistema mostrará al usuario son las representaciones de los documentos (en inglés, *code snippets*). Éstas consistirán en el título del documento, un sumario -de ser posible- y un enlace para visitar la localización original del documento (o descargarlo).

3.2.3.2. Clases del análisis

Al analizar gramaticalmente los casos de uso, se pueden encontrar los siguientes objetos, cuyas relaciones se ilustran en el diagrama UML de la figura 3.2:

Sesión: consiste en cada uso para documentación del sistema, contendrá una representación del contexto local a la hora de terminarla, los documentos utilizados, el usuario y el curso en cuestión.

Contexto inmediato: es una representación de la tarea del usuario. Representa el estado que ésta tiene hasta la última recomendación hecha.

Perfil: una representación del conocimiento que se tiene de las sesiones anteriores en el curso, a partir de éste y el contexto inmediato se construye el *contexto local*.

Contexto local: una conjunción del contexto inmediato y el contexto base de un curso.

Componente de búsqueda y filtrado: es con el que se comunica el sistema anfitrión para realizar las búsquedas y obtener los resultados de las mismas.

Recomendación: el resultado de la búsqueda y filtrado. Contiene una colección de documentos y una referencia al contexto que la suscitó para que el sistema pueda ser capaz de comparar entre el contexto actual y el de la última recomendación.

Documento: una referencia a documento propiamente dicho, puede consistir en un *documento en crudo*, que es el documento que el usuario provee antes de que éste se agregue a la colección documental o un *sustituto documental*, que es un documento que ya forma parte de la colección. Nótese que las recomendaciones contienen colecciones de éste último tipo.

Usuario: una referencia a la representación de usuarios del sistema anfitrión; deberían poder identificarse únicamente.

Curso: una referencia a la representación de cursos del sistema anfitrión; además de tener un identificador único, sería deseable contar con una descripción y un nombre, para la construcción del perfil.

Agente recolector: el sub-sistema que se encarga de obtener automáticamente nuevos documentos y agregarlos a la colección.

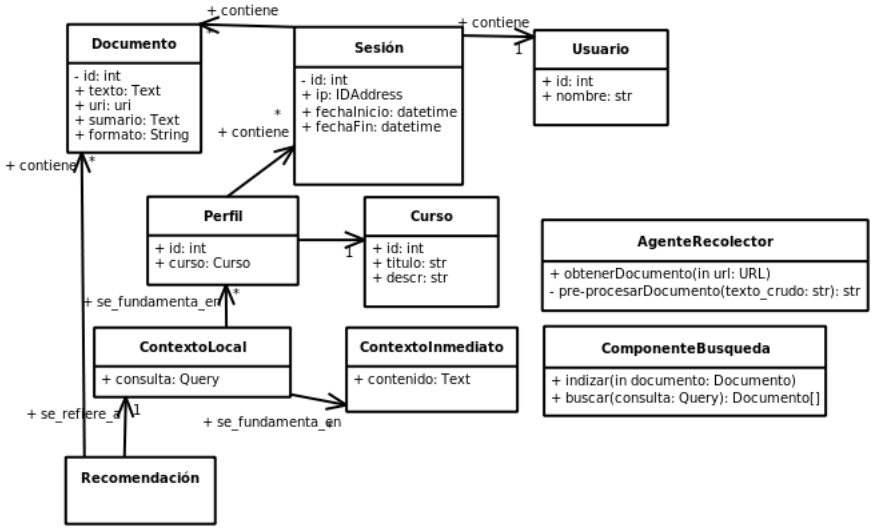


Figura 3.2: Diagrama de clases UML para las clases del análisis

3.2.4. Análisis de interacción y funcional

En esta etapa del análisis, se pretenden estudiar más a fondo las distintas interacciones entre el usuario y el sistema. Éstas ya fueron definidas a grandes rasgos en los casos de uso (3.2.2). Aquí se presentan diagramas de interacción profundizando en los casos de uso (figuras 3.3 y 3.4), diagramas de estado (figura 3.6) y un prototipo[6] de la interfaz de usuario (figura 3.5).

Se consideran aquí dos enfoques posibles para la implementación del sistema: el primero (figura 3.6a) utiliza la información del perfil del curso y la obtenida del contexto inmediato para construir la consulta -aquí se podría utilizar, por ejemplo, una ontología de dominio derivada del mismo perfil para la extracción de términos clave y la expansión con sinónimos. Por otro lado, el segundo enfoque (figura 3.6b) utiliza la información del perfil para filtrar y ordenar los resultados según su adecuación a éste, mientras que construye la consulta sólo en base al contexto inmediato, extrayendo términos clave con algoritmos heurísticos.

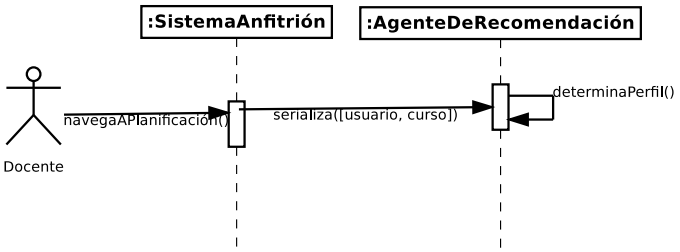
Aunque el análisis funcional merece un apartado aparte, aquí se ha obviado porque esta etapa de análisis ya ha presentado ciertos detalles de la funcionalidad interna y respecto al usuario y una profundización mayor correspondería ya a la etapa de implementación.

3.2.5. Análisis de configuración

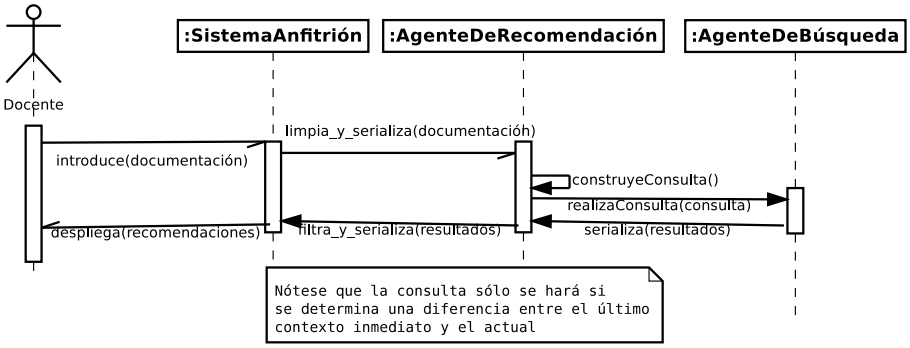
Aunque los requerimientos de configuración ya fueron mostrados en el apartado 3.2.1.2, las razones que los suscitaron se resumirán aquí.

El sistema se ha pensado como un componente independiente del sistema anfitrión. Por ello, aunque ciertos requisitos de compatibilidad son inevitables (como la existencia en la

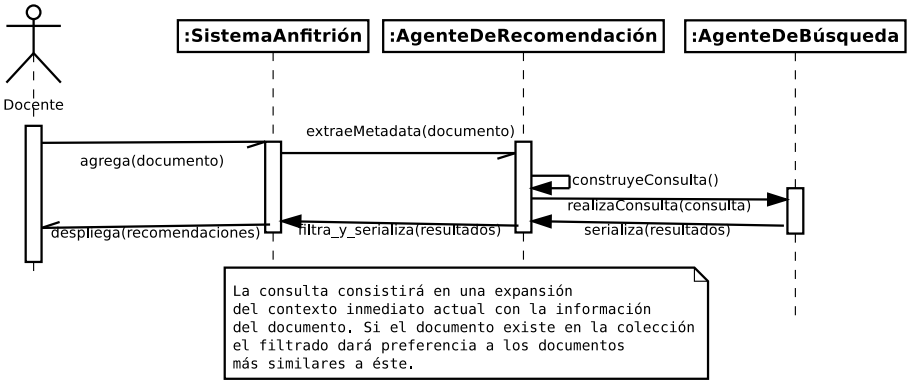
[6]Creado con `gomockingbird.com`



(a) Caso de Uso 1: IniciarSesión

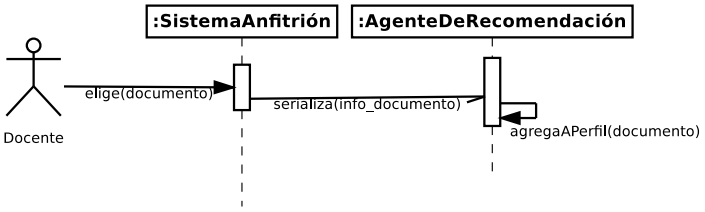


(b) Caso de Uso 2: Documenta

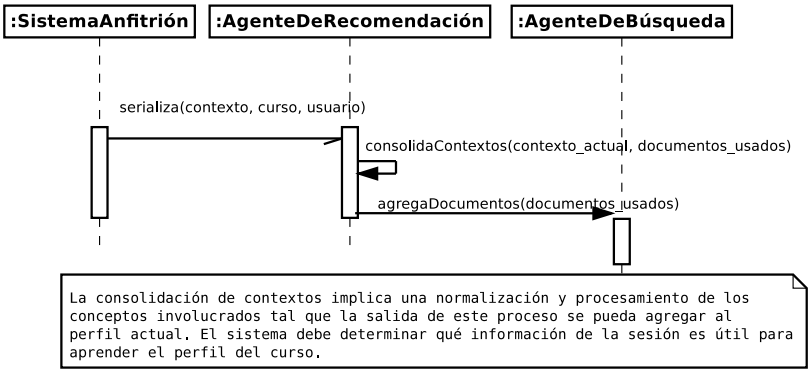


(c) Caso de uso 3: AgregaDocumento

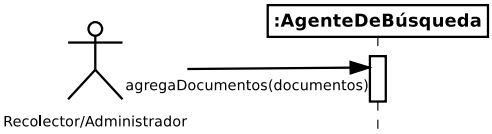
Figura 3.3: Diagramas de interacción para los casos de uso 1 al 3



(a) Caso de uso 4: UsaRecomendación



(b) Caso de uso 5: CierraSesión



(c) Caso de uso 6: ActualizaColección

Figura 3.4: Diagramas de interacción para los casos de uso 4 al 6

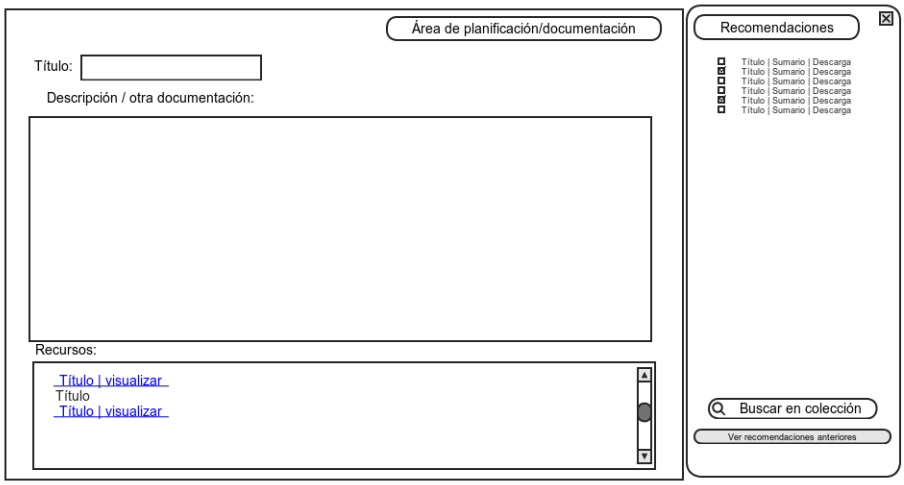


Figura 3.5: Prototipo de interfaz de usuario

base de datos de entidades que representen a los usuarios y los cursos), la comunicación entre ambos debería ser tan independiente de las implementaciones como sea posible, para ello, se proponen dos notaciones de intercambio de datos ampliamente utilizadas en las aplicaciones web: *json* y *xml*. Se ha favorecido el primero por la facilidad de generación y análisis.

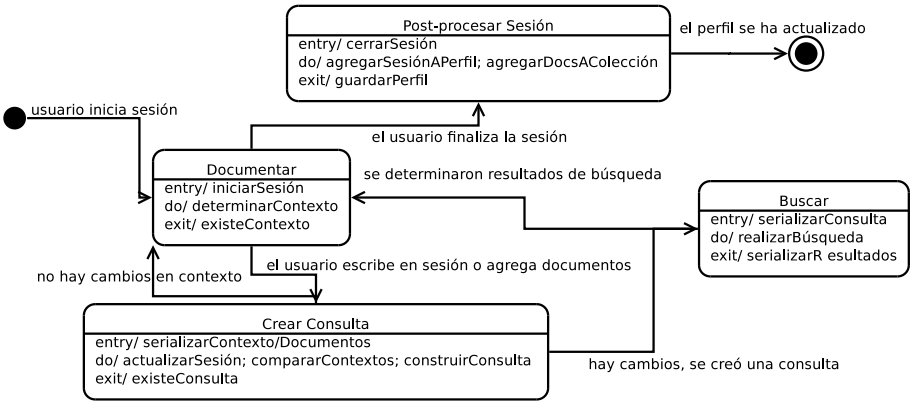
El lenguaje de programación python se ha elegido por su fácil sintaxis, extensa colección de librerías y eficiencia. El marco de desarrollo django se ha favorecido por la experiencia previa del autor con el mismo y por su adhesión a filosofías de desarrollo efectivas, como el uso del modelo de arquitectura MVC y la orientación al desarrollo modular.

Puesto que se desea una interfaz de usuario dinámica e intuitiva, se hizo evidente que se necesitaría agregar funcionalidad al lado del cliente con javascript. Puesto que se desea que el sistema funcione en todos los navegadores web populares, se ha optado por la librería jquery, que se encarga de la compatibilidad y permite al desarrollador preocuparse por escribir código funcional. A pesar de la existencia de diversas librerías javascript que comparten el mismo propósito (como prototype[7] o mootools[8]), jquery tiene una sintaxis más sencilla, mayor uso en aplicaciones comerciales y eficiencia en la manipulación de (x)html[9].

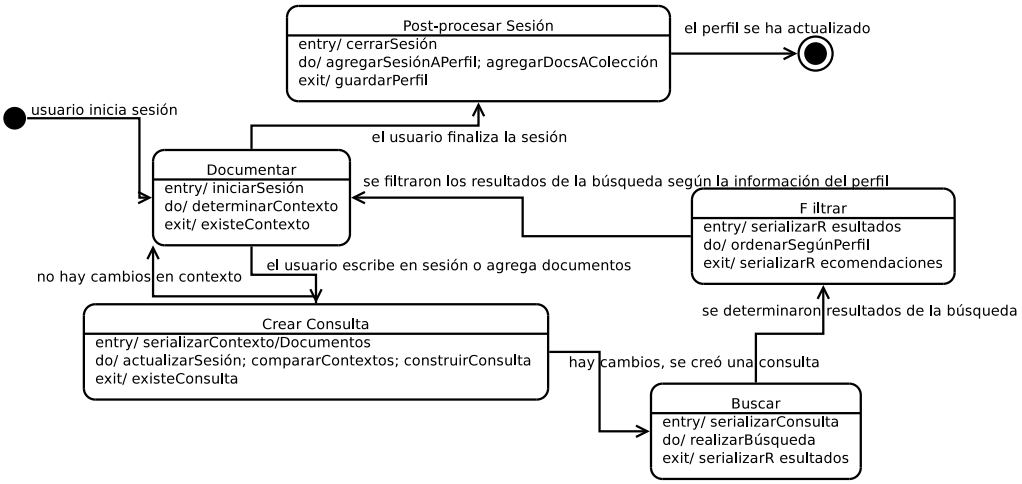
PostgreSQL es un motor de bases de datos eficiente en la manipulación de consultas y sin costo.

Siguiendo la tradición del uso de servidores Linux para aplicaciones que pretendan

[7] `http://www.prototypejs.org/`
[8] `http://mootools.net/`
[9] Según la prueba comparativa de selectores css en `http://mootools.net/slickspeed/`



(a) Enfoque con *pre-filtrado*



(b) Enfoque de *post-filtrado*

Figura 3.6: Diagramas de estado para la interacción con el sistema

escalar y ofrecer el mayor rendimiento, y, dado que el sistema anfitrión de prueba[10] está actualmente siendo servido desde un servidor con el sistema operativo ubuntu, se ha optado por éste sistema operativo para el almacenamiento y servicio.

3.3. Diseño

Siguiendo la pauta sentada en la fase de análisis, se muestra aquí el diseño ordenado según las distintas facetas del mismo propuestas por Pressman (2006), téngase en cuenta que la tercera etapa de ese flujo de diseño, el *diseño de navegación*, se obvia aquí porque el sistema carece de distintas interfaces a las cuales el usuario pudiese navegar; el usuario sólo verá la interfaz de recomendaciones.

3.3.1. Diseño de la interfaz, estético y de contenido

En esta etapa se deben considerar los aspectos de la interacción con el usuario que afecten su apreciación, y consecuente juicio de utilidad, del sistema. Esta interacción se puede considerar mediante los siguientes aspectos: la interfaz en sí, cómo está ésta diseñada estéticamente y qué contenido presenta. Éstos aspectos, aunque íntimamente relacionados, pueden, en el caso de una aplicación web a gran escala, considerarse en etapas independientes y quizá hasta por personas diferentes; en este proyecto, sin embargo, dado el relativamente poco porcentaje de interacción con el usuario final, se abordarán en un solo apartado.

La investigación en el diseño de interfaces se ha convertido en todo un campo de estudio, trascendiendo lineamientos empíricos y subjetivos y tomando elementos de la psicología cognitiva para estudiar las experiencias de usuario. En el caso concreto de los sistemas del tipo propuesto por este proyecto (sistemas de recuperación oportuna de información), la tesis de doctorado de Rhodes (2000a) estudia a fondo la experiencia del usuario y cómo el diseño de la interfaz la afecta, y asegura que:

> "Los *JITIRs* deben permitir a una persona concentrar su atención en su tarea primaria y a la vez dejar que la atención pueda ser dividida entre la tarea primaria y las recomendaciones."

Para que esta aseveración sea realizable, se debe encontrar un equilibrio entre la *concentración de la atención* del usuario y presentar la interfaz de recomendaciones de una manera que permita distinguirla del ambiente de la tarea primaria o ignorarla cuando ya no se desee o necesite información; y, a la vez, la interfaz no debe ser tan disímil como para exigir un cambio mental de contexto tan grande que la tarea principal sea dejada de lado.

Para que este equilibrio sea conseguido, la transición entre la tarea de documentación del docente y la revisión y uso de recomendaciones debe ser lo menos brusca posible.

Desde un punto de vista estético, esto se puede conseguir aprovechando el hecho de que *los usuarios exploran las páginas web desde arriba a la izquierda hasta abajo a la derecha*

[10] `escolarea.com`

(Pressman, 2006). Así, teniendo en cuenta que la interfaz de recomendación debe ser tanto accesible como no intrusiva, se ubicará a la derecha del área de trabajo principal. También, se elegirán colores en el rango cromático de la interfaz principal, de manera que se sepa que está relacionada y que, al mismo tiempo, no se distraiga la atención por ser demasiado distinta.

Y desde el punto de vista funcional, Rhodes (2000a) sugiere el uso de *interfaces incrementales*. En las cuales:

> "Cada etapa provee un poco más de información, al costo de requerir un poco más de atención para leerla y entenderla"

La idea de una interfaz de esta naturaleza es que permita al usuario decidir si vale la pena dividir su atención entre su tarea principal y las recomendaciones, reduciendo así el riesgo de afectar la concentración del usuario en vano. El mismo autor (Rhodes, 2000a) muestra el ejemplo de seis etapas en una interfaz incremental en su sistema *Margin notes*. Una adaptación de éstas al proyecto actual se describe a continuación:

1. *No hay interfaz visible.* En este momento el agente de recomendación está o recuperando información del perfil del curso, decidiendo si la información en el contexto actual es suficiente como para llevar a cabo una búsqueda o realizando la búsqueda en sí. Puesto que no existen recomendaciones que mostrar, sería del todo contraproducente distraer al usuario con cualquier especie de contenido visual. El autor se había sentido tentado a utilizar una barra de progreso o una animación que mostrasen que el sistema estaba trabajando en las recomendaciones, pero esta decisión hubiera sido de mal diseño, porque, como asegura Pressman (2006): "la interfaz [sólo] debe comunicar el estado de cualquier actividad que *el usuario haya iniciado*"[11].

2. *Existen recomendaciones.* Cuando ya existan recomendaciones, se lo hará saber al usuario mediante una discreta pestaña a la derecha de la pantalla, de manera que sea notable en la visión periférica del usuario. Véase la figura 3.7 para un prototipo de cómo se vería la interfaz en este estado.

3. *Gráfico de relevancia.* Una vez que el usuario decida que es hora de evaluar las recomendaciones, dará clic en la pestaña de la interfaz. Esto dará paso a la ventana de recomendaciones, con un gráfico mostrando qué tan relacionadas están éstas, en promedio, al contexto inmediato actual. Así el usuario puede decidir regresar a su tarea sin continuar viendo el resto de la interfaz de recomendaciones. En todo momento el usuario debe poder esconder las recomendaciones.

4. *Muestras documentales ordenadas por relevancia.* Bajo el gráfico de relevancia se encontrarán las representaciones de los documentos encontrados, éstas estarán ordenadas por relevancia con el título resaltado y un corto sumario. Así el usuario puede juzgarlas someramente.

[11] Énfasis del autor

5. *Palabras clave en un dialogo flotante.* Si el usuario decide pasar el mouse por encima de alguna recomendación, una pequeña ventana flotante mostrará los conceptos (o palabras clave) del contexto local que causaron la sugerencia de ese recurso específico, esto permite que el usuario haga juicios más detallados sobre los elementos individuales.

6. *Visualizar el recurso.* Una vez que el usuario ha decidido que un recurso puede serle útil, puede agregarlo inmediatamente a sus documentos de la sesión actual o visualizarlo en una ventana aparte. Nótese que en esta etapa el usuario ha pasado completamente de su tarea principal a la secundaria.

Además de las etapas de interfaz incremental de las recomendaciones en relación a la tarea principal antes discutidas, la interfaz de recomendación en sí debe obedecer lineamientos de diseño que aseguren una alta eficiencia cuando el usuario se decida a dividir su atención hacia ella. Los siguientes han sido tenidos en cuenta:

1. *Reducir el esfuerzo.* La ley de Fitt, citada por Pressman (2006), establece que: "El tiempo para adquirir un objetivo es una función de la distancia a la que se halla y de su tamaño", ésta, aunada a la *ley de los dos segundos*, hecha notar por Rhodes (2000a), que establece que el umbral aceptable para llevar a cabo una sub-tarea - esto es, cualquier tarea que no forme parte de la tarea principal - es de dos segundos, puesto que "incrementar el esfuerzo involucrado en llevar a cabo una tarea, causará un decremento proporcional en el numero de veces que ésta se lleva a cabo", la interfaz de recomendación permitirá seleccionar fácilmente las recomendaciones y agregarlas con un botón que se encuentre en la muestra documental -o agregar varias con un botón cercano a la lista de sugerencias. Como alternativa, el usuario podrá elegir una recomendación y *arrastrarla* al área de trabajo principal.

2. *Facilidad de aprendizaje.* La interfaz de recomendaciones debe ser similar a lo que usuarios hayan usado antes, por ello, las muestras documentales se parecen a resultados de un motor de búsqueda y están acompañadas de cajas de selección que establecen explícitamente que pueden elegirse varias sugerencias a la vez. Además, claro, de lo discutido en el primer punto: las alternativas de uso permiten que tanto usuarios acostumbrados a interfaces dinámicas como neófitos puedan intuir sin mayor esfuerzo cómo usar las recomendaciones.

3. *Reducción de latencia.* Mediante el uso de solicitudes asíncronas al servidor (gracias a la técnica *ajax*[12]) a la hora de hacer las búsquedas de recursos a recomendar.

4. *Mantener la integridad del producto de trabajo.* Los contextos y recomendaciones se guardarán automáticamente en una *sesión* del servidor: el sistema de sesiones[13] de *django* permite asociar una visita de usuario con un objeto que persiste hasta dos semanas en el lado del servidor, evitando así que errores de comunicación con

[12] Del inglés *asynchronous javascript and xml* es una técnica que permite realizar solicitudes asíncronas al servidor sin necesidad de volver a cargar las páginas por completo a la hora de procesar la respuesta.
[13] `http://docs.djangoproject.com/en/dev/topics/http/sessions/`

éste afecten la tarea del usuario. Aunque no está dentro del alcance del sistema, esta misma persistencia se aplicará a la tarea principal en el sistema anfitrión de prueba.

5. *Legibilidad y accesibilidad.* Mediante el uso de html apegado a los estándares de la w3, así como el *minimalismo* a la hora de mostrar los resultados: sólo lo necesario, sin elementos de distracción, con fuentes de tamaño y color legibles, y sumarios lo suficientemente cortos como para ser comprendidos pero no incurrir en una sobrecarga de información. Asimismo, otra motivación para un diseño minimalista es lo poco deseable que, dada alguna configuración de pantalla relativamente estándar, el contenido tenga tan poco espacio que una barra de desplazamiento deba ser agregada por el navegador. Esto porque los usuarios encuentran indeseable este tipo de comportamiento (Pressman, 2006).

6. *Comunicación*: Hacer saber al usuario cuándo, en base a un cambio de contexto, se han determinado nuevas sugerencias. Se parte del supuesto que, una vez que el usuario vuelva a su tarea primaria, la interfaz se esconderá de su vista. En caso de que siempre permaneciese abierta quizá sería mejor, en vez de actualizarla automáticamente, comunicar explícitamente la situación mediante un aviso conspicuo. Asimismo, la interfaz debe permitir al usuario ver recomendaciones anteriores de una manera fácilmente navegable.

7. *Eficiencia y anticipación*: puesto que la tarea principal de la aplicación es incrementar la productividad del usuario, y éste podría estar interesado en recursos que no hay manera de recomendar automáticamente, se ha decidido permitir una búsqueda directa, dispuesta hacia abajo a la derecha de la interfaz para resaltar su naturaleza de último recurso. No obstante, los resultados de una búsqueda directa serán también filtrados según el perfil del curso. Por las mismas razones, se le permitirá al usuario navegar entre las recomendaciones hechas en otros contextos de la sesión.

Para un prototipo de la interfaz de recomendación véase la figura 3.8 , téngase presente que ésta estará a la derecha del área principal de trabajo (como en la figura 3.5), por lo que algunos detalles quizá sean distintos, dada la relativamente poca cantidad de espacio de la cual se dispone.

3.3.2. Diseño de arquitectura

Aunque Pressman (2006) identifica dos partes en esta etapa, a saber, la identificación y modelado de una *arquitectura de contenido* que refleje cómo las distintas páginas de una aplicación web están relacionadas en términos de navegación y una *arquitectura de aplicación* que se refiere al sistema en su totalidad. Dada la inexistencia de la primera arquitectura en el sistema propuesto, se hablará aquí solamente de la postrera.

La arquitectura de la aplicación adoptará la propuesta por el marco de desarrollo *django*, una similar a la clásica modelo-vista-controlador: la arquitectura *modelo-plantilla-*

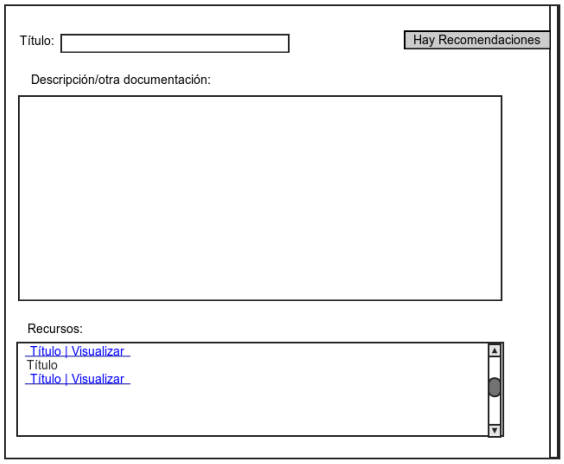


Figura 3.7: Estado oculto de la interfaz de recomendación

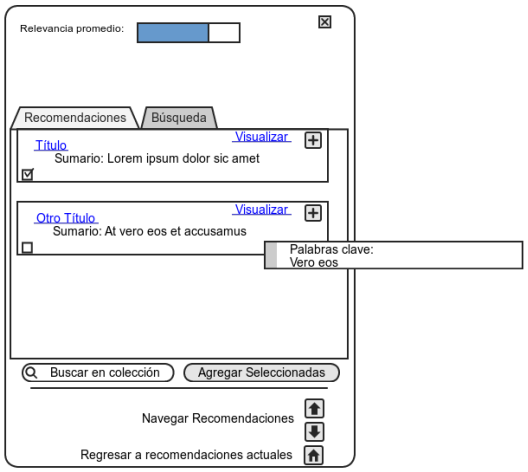


Figura 3.8: Detalle del prototipo de la interfaz de recomendación

vista[14] (MTV, por sus siglas en inglés). Todas las definiciones y descripciones que siguen están fundamentadas en la documentación oficial de django, que se puede encontrar -en inglés- en `http://docs.djangoproject.com/`.

Los componentes de esta arquitectura se describen someramente a continuación:

Modelo: un modelo en la arquitectura MTV es una clase que extiende a una súper-clase definida en el *framework* que abstrae tanto los atributos como la funcionalidad relacionadas a un objeto. Esta clase es independiente del sistema administrador de bases de datos que se utilice. Toda operación que se realice con el almacenamiento se hará a través de métodos implementados en la súper-clase.

Plantilla: (en inglés, *template*) representa la presentación de la aplicación, escrita en una combinación de html y un lenguaje para crear páginas dinámicas definido en el *framework* permite que desarrolladores y diseñadores puedan trabajar en la presentación y el comportamiento de la aplicación de manera separada y ordenada. Corresponde en parte a la *vista* del modelo MVC tradicional, porque define *cómo* se mostrará el contenido.

Vista: Corresponde en parte al controlador y la vista del modelo MVC tradicional; define *qué* contenido se presentará y es donde las interacciones del usuario con los datos almacenados se manejan. Cuando una URL[15] se solicita, django crea una instancia de una clase especial definida en el *framework*: la clase HttpRequest, que contiene meta-datos de la solicitud (como la ruta de la URL que se llamó y los encabezados) y la envía de parámetro a la vista, que es una función. La vista siempre devuelve una instancia de la clase HttpResponse que encapsula los datos que se mostrarán en una plantilla o se enviarán directamente al solicitante (como datos serializados en los formatos xml o json, una respuesta de error o una redirección a una página externa).

La interacción entre las distintas partes de esta arquitectura se puede ver así: el usuario solicita una página, identificada por una URL, el *framework* encuentra la *vista* correspondiente y la llama mediante una solicitud http, la vista puede o no interactuar con los *modelos* -creándolos, modificándolos o sencillamente obteniendo un conjunto de diversas instancias de éstos- y devuelve una respuesta que es mostrada en alguna *plantilla* o devuelta directamente al solicitante. Esta interacción está representada en la figura 3.9.

3.3.3. Diseño de componentes

Para implementar un sistema de recomendación se debe tener en mente qué elementos lo conformarán; según lo mencionado en los capítulos anteriores, se puede esbozar una macro-arquitectura del sistema la cual consistirá de los siguientes elementos:

1. Una colección de documentos.

[14] Comparada con la MVC en `http://docs.djangoproject.com/en/dev/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-ho`

[15] del inglés, *uniform resource locator*

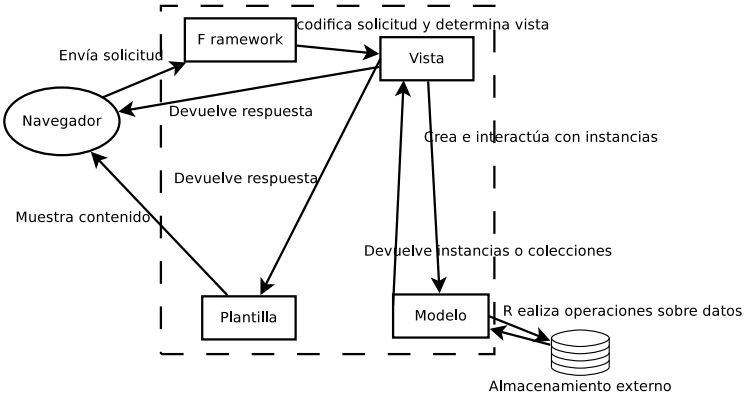


Figura 3.9: La arquitectura MTV

2. Un componente de recuperación de información. El cual incluirá sus respectivos sub-componentes de indización, igualación y obtención de consultas.

3. Un componente de filtrado de información. El cual contempla los aspectos de elección de resultados a mostrar, interfaz de presentación y obtención de retroalimentación.

Esta sección se ampliará cuando se discuta la implementación del sistema

Bibliografía

Adomavicius, G., & Tuzhilin, E. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*, 734–749.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.107.2790

Ahn, J.-w., Brusilovsky, P., & Farzan, R. (2005). Investigating Users' Needs and Behavior for Social Search. In *Proceedings of the Workshop on new Technologies for Personalized Information Access, part of the 10th International Conference on User Modelling (UM'05)*, (pp. 1–12). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.93.4869\&rep=rep1\&type=pdf

Arlow, J., & Neustadt, I. (2005). *UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design*. Addison-Wesley Professional, 2 ed.
URL http://scholar.google.com/scholar?hl=en\&btnG=Search\&q=intitle:UML+2+and+the+Unified+Process#1

Balabanović, M., & Shoham, Y. (1997). Fab: content-based, collaborative recommendation. *Communications of the ACM*, *40*(3), 72.
URL http://portal.acm.org/citation.cfm?id=245124

Bassu, D., & Behrens, C. (2003). Distributed LSI: scalable concept-based information retrieval with high semantic resolution. In *Proceedings of the 3rd SIAM International Conference on Data Mining (Text Mining Workshop)*, vol. 11. San Francisco, CA.
URL http://cchen1.csie.ntust.edu.tw/students/2009/DistributedLSIScalableConcept-basedInformationRetrievalwithHighSemanticResolution.pdf

Basu, C., Hirsh, H., & Cohen, W. (1998). Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *In Proceedings of the Fifteenth National Conference on Artificial Intelligence*, (pp. 714–720). AAAI Press.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.36.4620

Baudisch, P. (1999). Joining collaborative and content-based filtering. In *Proceedings of the ACM CHI Workshop on Interacting with Recommender Systems*, (pp. 1–5). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.16.2812\&rep=rep1\&type=pdf

Baziz, M., Boughanem, M., Pasi, G., & Prade, H. (2004). A Fuzzy set approach to concept-based information retrieval. In *10th Int. Conf. IPMU*, (pp. 1775–1782). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.95.7389\&rep=rep1\&type=pdf

Becker, J., & Kuropka, D. (2003). Topic-based vector space model. In *Proceedings of the 6th International Conference on Business Information Systems*, (pp. 7–12).
URL http://scholar.google.com/scholar?hl=en\&btnG=Search\&q=intitle:Topic-based+Vector+Space+Model#0

Becker, P., & Eklund, P. (2001). Prospects for document retrieval using formal concept analysis. In *Proceedings of the Sixth Australasian Document Computing Symposium, Coffs Harbour, Australia*. Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.17.2644\&rep=rep1\&type=pdf

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. Oreilly and Associates Inc.
URL http://books.google.com/books?hl=en\&lr=\&id=KGIbfiiP1i4C\&oi=fnd\&pg=PT9\&dq=Natural+Language+Processing+with+Python\&ots=YOEmC4GEO3\&sig=8YL7QAi-OxBFtGubIIaphaEDMxo

Bouzeghoub, M., & Kostadinov, D. (2005). Personnalisation de l'information : aperçu de l'état de l'art et définition d'un modèle flexible de profils. In *Conférences Francophone en Recherche d'Information et Applications*, (pp. 31–42). Grenoble, Francia.
URL http://bach2.imag.fr/ARIA/publisparconf.php#2

Brin, S., & Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.3243

Budzik, J., & Hammond, K. (1999). Watson: Anticipating and contextualizing information needs. In *PROCEEDINGS OF THE ANNUAL MEETING-AMERICAN SOCIETY FOR INFORMATION SCIENCE*, vol. 36, (pp. 727–740). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.89.5471\&rep=rep1\&type=pdf

Burke, R. (1999). Integrating knowledge-based and collaborative-filtering recommender systems. In *Proceedings of the Workshop on AI and Electronic Commerce*, (pp. 69–72).
URL http://www.aaai.org/Papers/Workshops/1999/WS-99-01/WS99-01-011.pdf

Burke, R. (2000). Knowledge-based recommender systems. *Encyclopedia of Library and Information Systems*, *69*(Supplement 32), 175–186.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.41.3078\&rep=rep1\&type=pdf

Chen, D.-n., & Wu, C.-h. (2008). An ontology-based document recommendation system: design, implementation, and evaluation. In *PACIS 2008 Proceedings*.

Cimiano, P., Hotho, A., & Staab, S. (2005). Learning concept hierarchies from text corpora using formal concept analysis. *Journal of Artificial Intelligence Research*, *24*(1), 305–339.
URL http://scholar.google.com/scholar?hl=en\&btnG=Search\&q=intitle:Learning+concept+hierarchies+from+text+corpora+using+formal+concept+analysis#0

Cohen, W., & Singer, Y. (1999). Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems (TOIS)*, *17*(2), 141–173.
URL http://portal.acm.org/citation.cfm?id=306688\&dl=GUIDE,

Cormack, G. V., Palmer, C. R., & Clarke, C. L. A. (1998). Efficient construction of large test collections. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '98*, (pp. 282–289).
URL http://portal.acm.org/citation.cfm?doid=290941.291009

Crestani, F., & Pasi, G. (1999). Soft information retrieval: Applications of fuzzy set theory and neural networks. In *Neuro-fuzzy techniques for intelligent information systems*, (pp. 287–315). Physica Verlag.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.112.7820\&rep=rep1\&type=pdf

Cutrell, B. Y. E., Dumais, S. T., & Teevan, J. (2006). Personal Information Management. *Communications of the ACM*, *49*(1), 58–64.

Daoud, M., Tamine-Lechani, L., & Boughanem, M. (2008). Using a concept-based user context for search personalization. In *International Conference of Data Mining and Knowledge Engineering (ICDMKE), London, UK*, vol. I, (pp. 57–64).
URL http://scholar.google.com/scholar?hl=en\&btnG=Search\&q=intitle:Using+A+Concept-based+User+Context+For+Search+Personalization#0

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, *41*(6), 391–407.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.62.1152\&rep=rep1\&type=pdf

Dumais, S., Cutrell, E., Sarin, R., & Horvitz, E. (2004). Implicit queries (IQ) for contextualized search. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, vol. 25, (p. 594). ACM.
URL http://portal.acm.org/citation.cfm?id=1009137

Dumais, S., Furnas, G., Landauer, T., & Deerwester, S. (1988). Using latent semantic analysis to improve information retrieval. *Proceedings of CHI*, (pp. 281–285).

URL http://scholar.google.com/scholar?hl=en\&btnG=Search\&q=intitle:Using+latent+semantic+analysis+to+improve+information+retrieval#1

Fleischman, M., & Hovy, E. (2003). Recommendations without user preferences: a natural language processing approach. In *Proceedings of the 8th international conference on Intelligent user interfaces*, (pp. 242–244). ACM New York, NY, USA.
URL http://portal.acm.org/citation.cfm?id=604045.604087

Fox, E., Palani, A., & Ganesan, V. (2010). Recommender systems.
URL http://en.wikiversity.org/wiki/7-c:_Recommender_systems

Fuhr, N. (1992). Probabilistic models in information retrieval. *The Computer Journal*, *35*(3), 243.
URL http://comjnl.oxfordjournals.org/cgi/content/abstract/35/3/243

García, E. (2006). Cosine Similarity and Term Weight Tutorial.
URL http://www.miislita.com/information-retrieval-tutorial/cosine-similarity-tutorial.html#Cosim

Gelbukh, A., Sidorov, G., & Guzmán-Arenas, A. (2005). Document indexing with a concept hierarchy. *Computación y Sistemas*, *8*(4), 281–292.
URL http://scielo.unam.mx/pdf/cys/v8n4/v8n4a3.pdf

Gipp, B., & Hentschel, C. (2009). Scienstein: A Research Paper Recommender System 1.

Grcar, M., Fortuna, B., Mladenic, D., & Grobelnik, M. (2005). kNN versus SVM in the collaborative filtering framework. In *Proc. of WebKDD*, (pp. 21–24). Springer.
URL http://www.springerlink.com/index/p871073356835258.pdf

Gruzd, A., & Twidale, M. (2006). Write while you search: Ambient searching of a digital library in the context of writing. *Digital Libraries in the Context of Users' Broader Activities*, (p. 13).
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.8745\&rep=rep1\&type=pdf#page=25

Hart, P., & Graham, J. (1997). Query-free information retrieval. *IEEE Expert*, *12*(5), 32–37.
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=621226

Hayes, C., Massa, P., Avesani, P., & Cunningham, P. (2002). An on-line evaluation framework for recommender systems. In *Recommendation and Personalization in eCommerce*, (p. 50). Malaga: Springer-Verlag.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.115.6861\&rep=rep1\&type=pdf#page=56

Herlocker, J. L., Konstan, J. A., Terveen, L. G., John, & Riedl, T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, *22*, 5–53.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.78.8384

Kageura, K., & Umino, B. (1996). Methods of Automatic Term Recognition - A Review.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.25.3740

Kitamura, Y., Sakamoto, T., & Tatsumi, S. (2002). A competitive information recommendation system and its behavior. *Lecture notes in computer science*, (pp. 138–151).
URL http://www.springerlink.com/index/W2EQKX8AQUM64H6V.pdf

Koutrika, G., & Ioannidis, Y. (2005). A Unified User Profile Framework for Query Disambiguation and Personalization. In P. Brusilovsky, C. Callaway, & A. Nürrnberger (Eds.) *Workshop on New Technologies for Personalized Information Access*, (pp. 44–54). Edinburgh, UK.

Leiva, I., & Muñoz, J. (1996). Tendencias en los sistemas de indización automática. Estudio evolutivo. *Revista española de documentación cient\'ifica*, *19*, 273–291.
URL http://webs.um.es/isgil/IndizacionautomaticaAutomaticindexingGILLEIVA.pdf

Levow, G.-A. (2003). Information Retrieval: Models and Methods.
URL http://www.wiziq.com/tutorial/721-Information-Retrieval-Models-and-Methods

Lewis, D. D. (1998). Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval. (pp. 4–15). Springer Verlag.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.11.8264

Matsuo, Y., & Ishizuka, M. (2003). Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.10.3742

McGowan, J. P. (2003). *A Multiple Model Approach to Personalised Information Access*. Masters, University College Dublin.

Menczer, F., Pant, G., Srinivasan, P., & Ruiz, M. (2001). Evaluating topic-driven Web crawlers. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, (pp. 241–249). ACM New York, NY, USA.
URL http://portal.acm.org/citation.cfm?id=383995\&dl=GUIDE,

Mooney, R. J., & Roy, L. (1999). Content-Based Book Recommending Using Learning for Text Categorization. In *IN PROCEEDINGS OF THE FIFTH ACM CONFERENCE ON DIGITAL LIBRARIES*, (pp. 195–204). ACM Press.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.16.4348

Mortensen, M. (2007). Design and evaluation of a recommender system. *Science*.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.103.2726\&rep=rep1\&type=pdf

Nielsen, J. (1994). Response Time Overview.
URL http://www.useit.com/papers/responsetime.html

Ozcan, R., & Aslangdogan, Y. (2004). Concept based information access using ontologies and latent semantic analysis. *Dept. of Computer Science and Engineering, University of Texas at Arlington, Technical Report*, *8*, 2004.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.99.1383\&rep=rep1\&type=pdf

Pajares, G., & Santos, M. (1999). *Inteligencia artificial e ingeniería del conocimiento*. Ra-Ma.

Pazzani, M., & Billsus, D. (2007). Content-based recommendation systems. *Lecture Notes in Computer Science*, *4321*, 325.
URL http://www.springerlink.com/index/qq35wt68l6774261.pdf

Popescu-belis, A., Boertjes, E., Kilgour, J., Poller, P., Castronovo, S., Wilson, T., Jaimes, A., & Carletta, J. (2008). The AMIDA Automatic Content Linking Device : Just-in-Time Document Retrieval in Meetings. In *Machine Learning for Multimodal Interaction*, (pp. 272–283). Utrecht, Holanda: Springer.

Pressman, R. S. (2006). *Ingeniería del Software: un enfoque práctico*. México, D.F: McGraw-Hill, 6 ed.

Puerta Melquizo, M., Bogers, T., L.W.J., B., Desphande, A., Bosch, A. V. D., Cardoso, J., Cordeiro, J., & Filipe, J. (2007). What a proactive recommendation system needs: relevance, non-intrusiveness and a new long-term memory. In *ICEIS 2007: Proceedings of the Ninth International Conference on Enterprise Information Systems*, (pp. 86–91). Setúbal, INSTICC.

Ramanathan, K., Giraudi, J., & Gupta, A. (2008). Creating hierarchical user profiles using Wikipedia. *hpl.hp.com*.
URL http://www.hpl.hp.com/techreports/2008/HPL-2008-127.pdf

Renz, I., Ficzay, A., & Hitzler, H. (2003). Keyword Extraction for Text Characterization. In *Proceedings of NLDB*, (pp. 228–234).
URL http://subs.emis.de/LNI/Proceedings/Proceedings29/GI-Proceedings.29-19.pdf

Rhodes, B. (2000a). *Just-in-time information retrieval*. Ph.D. thesis, Massachusetts Institute of Technology.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.43.4883\&rep=rep1\&type=pdf

Rhodes, B. (2000b). Margin notes: Building a contextually aware associative memory. In *Proceedings of the 5th international conference on Intelligent user interfaces*, vol. 9, (pp. 219–224). ACM New York, NY, USA.
URL http://portal.acm.org/citation.cfm?id=325850

Rungsawang, A. (1988). DSIR: the First TREC-7 Attempt.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.1968

Rungsawang, A., & Angkawattanawit, N. (2005). Learnable topic-specific web crawler. *Journal of Network and Computer Applications*, *28*(2), 97–114.
URL http://linkinghub.elsevier.com/retrieve/pii/S1084804504000086

Russell, S., & Norvig, P. (2003). *Artificial intelligence: a modern approach*. Englewood Cliffs, New Jersey: Prentice-Hall, 2 ed.
URL http://www.philadelphia.edu.jo/university/it/cs/syllabus/751252.pdf

Safran, C. (2005). *A Concept-Based Information Retrieval Approach for User-oriented Knowledge Transfer*. Master's thesis, Graz University of Technology.

Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*(11), 613–620.
URL http://portal.acm.org/citation.cfm?doid=361219.361220

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based Collaborative Filtering Recommendation Algorithms. (pp. 285–295).
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.9079

Schach, S. R. (2004). *Análisis y diseño orientado a objetos con UML y el proceso unificado*. México, D.F: McGraw-Hill, 1 ed.

Schein, A. I., Popescul, A., H., L., Popescul, R., Ungar, L. H., & Pennock, D. M. (2002). Methods and Metrics for Cold-Start Recommendations. In *In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 253–260). ACM Press.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.436

Schein, A. I., Popescul, A., Ungar, L. H., & Pennock, D. M. (2005). CROC: A New Evaluation Criterion for Recommender Systems. *Electronic Commerce Research*, *5*(1), 51–74.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.59.8796

Schonhofen, P., & Charaf, H. (2001). Document Retrieval through Concept Hierarchy Formulation. *PERIODICA POLYTECHNICA ELECTRICAL ENGINEERING*, *45*(2), 91–108.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.4062\&rep=rep1\&type=pdf

Schütze, H., Manning, C. D., & Raghavan, P. (2009). *Introduction to Information Retrieval*. New York: Cambridge University Press.
URL http://nlp.stanford.edu/IR-book/information-retrieval-book.html

Segaran, T. (2007). *Programming collective intelligence*. Sebastopol, Ca; EEUU: O'Reilly, 1 ed.
URL http://portal.acm.org/citation.cfm?id=1406354

Semeraro, G., Degemmis, M., Lops, P., & Palmisano, I. (2005). WordNet-based user profiles for semantic personalization. In *PIA 2005 Workshop on New Technologies for Personalized Information Access*, (p. 74). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.94.7472\&rep=rep1\&type=pdf#page=80

Sieg, A., Mobasher, B., & Burke, R. (2007). Web search personalization with ontological user profiles. *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM '07*, (pp. 5–25).
URL http://portal.acm.org/citation.cfm?doid=1321440.1321515

Spoerri, A. (1993a). InfoCrystal:Â a visual tool for information retrieval & management. *Conference on Information and Knowledge Management*.
URL http://portal.acm.org/citation.cfm?id=170095

Spoerri, A. (1993b). *Information Retrieval Models*, chap. 2. New York, New York, USA: ACM.
URL http://comminfo.rutgers.edu/~{}aspoerri/InfoCrystal/Ch_2.html#2.3.3.1

Stein, B., & Meyer Zu Eissen, S. (2003). AI SEARCH: Category Formation of Web Search Results.
URL http://www.uni-weimar.de/cms/medien/webis/research/projects/aisearch.html

Stein, B., & Potthast, M. (2007). Applying Hash-based Indexing in Text-based Information Retrieval. *Citeseer*.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.116.9323\&rep=rep1\&type=pdf

Tamine, L., Boughanem, M., & Zemirli, W. (2006). Inferring the user's interests using the search history. In *Workshop on information retrieval, Learning, Knowledge and Adaptatbility (LWA), Hildesheim, Germany, Schaaf, Martin, Althoff, Klaus-Dieter*, vol. pp, (pp. 108–110). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.7848\&rep=rep1\&type=pdf

Tamine-Lechani, L., Boughanem, M., & Zemirli, N. (2007). Exploiting multi-evidence from multiple user's interests to personalizing information retrieval. *2007 2nd*

International Conference on Digital Information Management, (pp. 7–12).
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4444192

Thorsten Brants, & Google (2004). Natural Language Processing in Information Retrieval. In *14th meeting of Computational Linguistics in Netherlands*. Antwerp, Bélgica: Google Inc.

Tsatsaronis, G., & Panagiotopoulou, V. (2009). A generalized vector space model for text retrieval based on semantic relatedness. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop on XX*, April, (pp. 70–78). Association for Computational Linguistics.
URL http://portal.acm.org/citation.cfm?id=1609188

Turtle, H., & Croft, W. (1991). Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems (TOIS)*, *9*(3), 187–222.
URL http://portal.acm.org/citation.cfm?id=125188\&coll=portal\&dl=ACM\&C...

Uzun, Y. (2006). Keyword Extraction Using Naive Bayes. In *Bilkent University, Department of Computer Science, Turkey www. cs. bilkent. edu. tr/\˜ guvenir/courses/CS550/Workshop/Yasin_ Uzun. pdf*, (pp. 1–5). Citeseer.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.103.2128\&rep=rep1\&type=pdf

Voorhees, E. (2002). The philosophy of information retrieval evaluation. *Lecture Notes in Computer Science*, (pp. 355–370).
URL http://www.springerlink.com/index/VNQ9W8Q6LBE5PLUE.pdf

Walker, G., & Janes, J. (1999). *Online Retrieval: A dialogue of theory and practice.*. Englewood, Colorado: Libraries Unlimited, 2 ed.