UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ACATLÁN

MODELO DE SISTEMA ADAPTIVO DE BÚSQUEDA EN INTERNET OPTIMIZADA

TESIS

QUE PARA OBTENER EL TITULO DE:

LICENCIADO EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN

PRESENTA:

DAVID AARON DÍAZ ESCAMILLA

FES UNAM ACATLÁN

ASESOR DE TESIS:

MTRA. ELVIRA BEATRIZ CLAVEL DÍAZ

MÉXICO, D.F. ENERO 2007

A mi Madre María de la Luz

y su amor ante la vida

Y

A mi Hermano Felipe Israel

que siempre me acompaña.

A mi amada familia

María Teresa
David Aarón
Diego Alberto

Estoy convencido, que cualquier logro en la vida está precedido por una sucesión de esfuerzos y de lideres, que nos muestran con su ejemplo como debemos conducirnos, para alcanzar nuestros objetivos. Por tanto voy a tratar de agradecer a cada una de las personas que me ha apoyado para la consecución de este trabajo y a lo largo de mi vida; sin embargo, cualquier omisión, es solo atribuible a la dificultad de enunciar un conjunto finito no numerable de seres queridos.

Gracias amigos:

Nufo y Tete Cruz (In Memoriam),
María del Carmen Escamilla (In Memoriam)
y Familia,
Rogelio y Maru Cruz Escamilla
e hijos,
Lourdes, Alejandro, Mariana, Zurazi
e Isac Romero (In memoriam),
Miguel y Ady Medina
e hijos,
Omar Y Elvia González
e hijos,
Francisco y Juana Escamilla
e hijos,
Vicente y Gaby Pineda
e hijas,
Steve y Kate Carpenter
e hijos,
Héctor y Rosy Romo
e hijas,
Alejandro y Maries Colín
e hijos,
José y Guille Camacho
e hijos,
Israel y Lupita Ocampo
e hijos,
Vicente y Rosa Pineda
e hija,
Yolanda, Héctor, Carolina, Blanca
y Pabel Robledo,
Dr. Jorge Sevilla (In Memoriam)
y Familia.

Arturo Escamilla,
Vadim Levtchouk,
Alberto Zamarripa,
Jaime Vergara,
Sergio Montes,
Jesús Ruíz,
Salvador Ruíz,
Arturo Ortíz,
Alejandro Ganem,
Nicolás Kemper
Salvador Zamora,
Margarita Azuceno,
Ricardo Ishigami,
Joaquin Gimenes,
Daniel Canudas,
Héctor Garda,
Carlos Díaz,
Juan Antonio Delgado,
José Antonio Sarmiento,
Luís Flores,
David González,
Gerardo Gómez,
Pedro Rico,
Israel Rico,
Mike Trinkle,
Luis Duarte,
Viridiana Pensado,
Víctor, Armando y Oscar,
ICN...

A la Universidad Nacional Autónoma de México.

A la Facultad de Estudios Superiores Acatlán

A mi profesora, Mtra. Elvira Beatriz Clavel Díaz, por su encomiable asesoría en la realización de este trabajo y por su esplendida amistad.

A mis sinodales cuya aportación enriqueció de manera sustancial esta tesis.

Fis. Manuel Valadéz Rodríguez
Lic. Jaime Ramírez Muñoz
Fis. Carlos Pedro Curiel García.
Lic. Jaime Vergara Prado

Tabla de Contenidos.

Prólogo ... iii

Introducción ... vii

Capítulo 1 Búsqueda de Información en Internet. ... 1
1.0 Introducción ... 2

1.1 Información en la Web. ... 6
1.1.1 Motores de Búsqueda. ... 10
1.1.2 Algoritmo de Kleinberg. ... 11

1.2 Búsqueda Distribuida Semántica y Distribuida. ... 16

1.3 Arquitecturas y Aplicaciones de Sistemas Multiagentes. ... 21
1.3.1 Arquitecturas Multiagentes ... 21
1.3.1.1 MIX/MAST ... 22
1.3.1.2 MACRON ... 23
1.3.1.3 MPA ... 25
1.3.1.4 RESTINA ... 25
1.3.1.5 Retriver ... 26
1.3.2 Aplicación de Agentes a Sistemas de Búsqueda ... 28
1.3.2.1 Letizia ... 28
1.3.2.2 Lira ... 29
1.3.2.3 Amalthea ... 29
1.3.2.4 WebWatcher ... 30
1.3.2.5 Syskill & Webert ... 30
1.3.2.6 WebMate ... 30
1.3.2.7 ARACHNID ... 31

Capítulo 2 Teoría de Recuperación de Información, Agentes y Herramientas de clasificación. ... 32

2.0 Introducción ... 33

2.1 Recuperación y clasificación de Información. ... 34
2.1.1 Web Content Mining. ... 34
2.1.2 Web Structure Mining. ... 34
2.1.3 Web Usage Mining. ... 35
2.1.4 Clasificación Automática de Textos (CAT) ... 36
2.1.4.1 Corpus de Entrenamiento ... 38
2.1.4.2 Modelo de Representación ... 38
2.1.4.2.1 Método de Selección de Características. ... 39
2.1.4.3 Modelo de Clasificación ... 43
2.1.4.3.1 Clasificación No Supervisada (CNS) ... 44
2.1.4.3.1.1 Modelo vectorial ... 46
2.1.4.3.2 Algoritmo de Kohonen y la determinación genética de clases. ... 49

2.2 Agentes como Robots de búsqueda. ... 57
2.2.1 Tipos de Agentes ... 58
2.2.1.1 Agentes de Interface. ... 59
2.2.1.2 Agentes Colaborativos. ... 60
2.2.1.3 Agentes Móviles. ... 60
2.2.1.4 Agentes de recuperación de información. ... 61
2.2.2 Lenguajes de agentes. ... 62
2.2.2.1 Java como lenguaje de agentes. ... 63
2.2.2.1.1 Aglets versus applets. ... 63

2.2.2.2 Propuesta multiparadigma para programación de agentes. ... 64
2.2.3 Ventajas e Inconveniente de los agentes móviles ... 66

2.3 Estadístico de Evaluación. ... 67
2.3.1 Índice Kappa ... 69

Capítulo 3 Estructura del Modelo y Aplicación de Herramientas propuestas. ... 78

3.0 Introducción ... 79

3.1 Estructura de SABIO (Sistema Adaptivo de Búsqueda en Internet Optimizada) ... 79
3.1.1 Arquitectura ... 82
3.1.1.1 Agente de Interfaz ... 83
3.1.1.2 Agente Intermedio ... 90
3.1.1.3 Agente de Minado ... 93

3.2 Aplicación del Algoritmo de Clasificación y el Estadístico de Concordancia. ... 98
3.2.1 Codificación de Documentos ... 101
3.2.2 Aplicación del Algoritmo de Kohonen ... 103
3.2.3 Generación del Estadístico Kappa ... 109
3.2.4 Prueba de Hipótesis ... 112

CONCLUSIONES. ... 114

BIBLIOGRAFÍA. ... 116

ANEXO I. ... 120
ANEXO II. ... 127
ANEXO III. ... 128
ANEXO IV. ... 129
ANEXO V. ... 130
ANEXO VI. ... 131
ANEXO VII. ... 132
ANEXO VIII. ... 133

A manera de prólogo, nos permitimos insertar el siguiente ensayo/cuento, publicado en 1998 en la revista española Puertas a la lectura, número cinco y, cuyo propósito, es mostrar los alcances tecnológicos que estamos por vivir; además, de compartir con el lector la motivación que toda obra visionaria despierta e impulsa en nosotros para contribuir en la medida de nuestras fuerzas al logro del bienestar futuro:

De Crusoe y los nuevos exploradores.

"Se arrellanó en el sillón anatómico, preparándose para una nueva sesión de lectura. Dejó vagar su mente por varias ideas intrascendentes, mientras el sillón modificaba automáticamente su nivel de mullido. Había sido un día rutinario, nada práctico, aburrido. La luz del ambiente se fue haciendo cada vez más tenue, hasta alcanzar el tono y calor más adecuado para la sesión de lectura. Las máquinas de ahora no eran como las que habían manejado hacía pocos años los pioneros de la informática, pensó. Las rutinarias tareas de codificación, organización, búsqueda y presentación de datos más o menos elaborados habían sido progresivamente acaparadas por nuevos métodos y herramientas, cada vez más potentes e inteligentes, que eran capaces de imitar, e incluso de anticiparse, a las acciones de sus amos humanos. Poco a poco, al igual que sucediese con las revoluciones industriales, las nuevas máquinas, los nuevos agentes o robots informáticos habían ido sustituyendo al ser humano en los procesamientos más estructurados y masivos de datos. El oficio de teclista o picador dormía el sueño de los justos hacía largo tiempo. Lo más curioso es que estos robots eran capaces de aprender, por imitación, gran parte de las acciones de sus creadores, y llegaban incluso a proponer a sus usuarios nuevas vías para llevar a cabo sus tareas, o satisfacer sus aficiones, Tenían la mala costumbre de husmear en todos los recovecos de los usos y hábitos informáticos de los humanos, e incluso algunos radicales sostenían que arteros, ladinos y oscuros intereses los utilizaban para conocer y controlar el comportamiento de la población. Desechó esos pensamientos, que seguramente le llevarían por sendas demasiado intrincadas para el día que había tenido hoy. La temperatura de la sala ya era agradable, y en los paneles visuales que formaban las paredes de la habitación ya había comenzado a formarse un paisaje de hayedos en una ladera, entre cuyas hojas penetraba tamizada la luz solar, reflejándose en un rápido y saltarín arroyo. No pudo dejar de sonreír ante la comparación con lo lluvioso y sucio que había sido el día. Decididamente, la programación domótica de la casa estaba pensada para equilibrar la vida del propietario, como decía la propaganda que lo había convencido para la compra, además de la compañía de un amable vendedor, que quitaba lo que de impersonal tenían todas las inmobiliarias en la actualidad, empeñadas en una loca carrera por la visita y

el vendedor en realidad virtual. El panel de lectura comenzó a desplegarse ante sus ojos, a una distancia adecuada para la incipiente miopía que sufría. Hacía tiempo que deseaba leer de nuevo Robinson Crusoe, escrita por Daniel Defoe (algunos habían dicho que se trataba de la reconstrucción novelesca de las aventuras de un tal Alexander Selkirk). Un hombre capaz de construir un mundo con sus manos, y defenderlo de todo tipo de amenazas, sin la ayuda de ningún artilugio informático, o infonómico o infomático, como quiera que le llamasen ahora los sesudos cerebros de la inforsociología. El panel ya mostraba la primera información: había satisfecho la petición del humano, y había localizado en diferentes servidores de información 43 versiones (el término ediciones referido a un libro se reservaba para aquellos en soporte papel, que hacía tiempo habían dejado de producirse) de Robinson Crusoe, en varios idiomas, ilustrados, e incluso en cómic. Juiciosamente, había seleccionado 4 de ellos, todos en su propio idioma, y formados sólo por texto. Sonrió. Su robot de lectura sabía que no le gustaba leer en otros idiomas, aunque los conociese, y que rechazaba de plano la inclusión de imágenes. Le atraía el romanticismo de los viejos libros de papel amarillo y lecturas hasta la madrugada. Optó por una versión fiel a la última edición en papel de una conocida editorial de finales del siglo pasado. El robot lector le avisó que esa versión no traía estudio crítico introductorio, como a él le gustaba, a pesar de lo cual se decantó por ella. El robot lector volvió a insistir, ofertándole tres trabajos sobre la obra y el autor, uno de ellos de literatura comparada. Frunció el ceño antes de negarse de nuevo a la lectura de las mismas. Sólo quería leer. Rechazó de plano ésta y futuras sugerencias. A veces resultaba agobiante la perfección y la presión de su robot de lectura, pero era la única forma de leer. Sólo se utilizaba el papel para impresos de valor inalcanzable para las personas normales, destinadas a los regalos entre altas administraciones y grandes corporaciones, y muchos de ellos pasaban a ser objeto de exposición en museos especiales. Las antiguas bibliotecas se habían transformado en salas de lectura automática guiada, y sus fondos impresos habían pasado a formar parte de museos dentro de las mismas, con lo que habían conseguido una fuente de ingresos que les había permitido sobrevivir al tremendo cambio tecnológico y cultural que supuso la desaparición del libro en su tradicional soporte en papel. Por el contrario, eso había supuesto un empobrecimiento de la producción literaria, empeñada ahora en conseguir un impacto sobre los lectores a través de un exceso de integración hipermedia y de diversidad de opciones en un discurso abierto. Ahora, uno podía hablar con su compañero del mismo libro, y comprobar que el desarrollo de la narración era completamente diferente según los gustos y orientaciones de los lectores. Ésta era una de las razones por las cuales los premios literarios habían desaparecido rápidamente, ya que nunca se conseguía la misma obra para todo el jurado. Comenzó la lectura tras servirse un poco de licor del bueno, no de esos baratos basados en química y transgénicos, que reservaba para ocasiones especiales. La lectura le dominó por completo. Enfrascado en

la lucha de Crusoe contra el mar, en su llegada a la playa y en sus primeros paseos por el nuevo hogar, no se dio cuenta de que el paisaje de los paneles había dado paso a un atardecer en una playa de fina arena. El robot de lectura le interrumpió al final del capítulo para advertirle de la hora, y de la necesidad de iniciar un periodo de descanso. Ignoró el aviso, y agradeció haber desconectado con anterioridad sus capacidades sonoras. Decidido, se sumergió en un nuevo capítulo. Tras leer varios párrafos, se levantó para dejar la copa y dar un breve paseo por la habitación. Al sentarse de nuevo se encontró con lo que temía. La central de lectura propietaria del libro había detectado que había dejado de pasar páginas, y rápidamente, al comprobar la avanzada hora, le había enviado un panel de publicidad, con contenidos acordes, según alguna mente preclara, con su insomnio. Evidentemente, no tenía ninguna intención de cambiar la lectura por consejos espirituales o astrológicos, o por otros placeres más mundanos. Solicitó que no le enviasen de nuevo ese tipo de publicidad en ningún caso, y volvió a la lectura. Mientras tanto, Crusoe seguía en su lucha contra el caos, intentando dar un orden a su mundo. Recordó que él también tenía que poner orden en el caos de su trabajo al día siguiente. Ordenó a su robot lector que colocase una señal de lectura para el día siguiente, a la misma hora. Algo había mejorado. Por lo menos, las decisiones tomadas hoy parecían adecuadas. Su mejor humor le hizo entrar en la lectura con más ganas. Tenía intención de terminar la lectura de las aventuras del pionero Crusoe esa noche. Su gasto mensual en lectura era asequible, pero no quería abusar de sus cuentas, porque las entidades bancarias vigilaban (bueno, lo hacían los exploradores de riesgos, otro invento como los robots lectores) con lupa las variaciones y los gastos de sus clientes. El robot de lectura, visto lo sucedido la noche anterior, no le molestó con sugerencias, lo que le permitió ir directamente al enfrentamiento con los antropófagos. Al terminar el capítulo, la central de lectura le interrumpió con una oferta personalizada de materiales sobre Crusoe y su autor. Nuevas y viejas películas, incluso en blanco y negro, documentales, personales y juegos virtuales construidos a medida, etcétera, etcétera. Claro, el comercio era otro de los ingresos que engordaban las acaudaladas cuentas de las centrales de lectura. Al desechar la noche anterior los subproductos típicos de la hora, algún robot habría consultado su perfil, sus lecturas y sus hábitos, y habían decidido ofrecerle una selección de productos que podría clasificarse como culturales. Incluso le habían buscado un viaje de estilo aventurero a una isla del Pacífico, a pagar en módicos y cómodos plazos, así como ofertas bancarias a tal fin. Como no tenía compañero, no se incluían productos de corte romántico. Se preguntó cómo lo sabrían, pero recordó poco después que la lectura era un hábito individual, y que los paneles de lectura estaban preparados para detectar la presencia de varios lectores, porque, claro estaba, no era la misma cuota a pagar cuando leía uno que cuando leían varios al mismo tiempo. Desechó de un plumazo toda la oferta, y pidió que no se le molestase más durante la lectura del libro.

Sabía que la central de lectura respetaría su petición, porque era una de las normas básicas del negocio. A una hora avanzada terminó la lectura y sonrió con satisfacción. Al día siguiente no tenía que ir al trabajo, y podría dormir un poco más. El robot lector se permitió recordarle la necesidad de cuidar sus ojos, y decidió que esta vez tal vez si hiciese caso de su recomendación. Cuando se iba a levantar del sillón, recibió un catálogo de la central de lecturas, que incluía doce libros sobre: náufragos, aventureros solitarios y otras gentes. Desconectó con una sonrisa irónica. Hacía falta valor para comparar a Robinson Crusoe con Indiana Jones."

Dr. Jesús Tramullas Saz.[1]

[1] Refiérase al numeral [44] de la bibliografía para mayor información.

La recuperación de información es una de las actividades más demandantes en un mundo global que comienza a girar en torno al Internet, en este sentido, cada vez más profesionales en todas las áreas, se dedican a la búsqueda de información de calidad, y en cada ejercicio se hace más evidente la desventaja que existe entre la cantidad de las fuentes que ofrecen información y los medios para acceder a ella; generándose millones de horas hombre-máquina. Si este tiempo puede reducirse, es posible obtener beneficios desde el ámbito económico hasta el ecológico.

Por lo anterior, el objetivo de este trabajo es proponer una herramienta que eficiente el acceso a la información en Internet, para ello, se ha desarrollado una extensa investigación respecto a las propuestas tecnológicas de este campo de estudio; donde sobresalen los sistemas multiagentes como una solución probada y adaptable a la que dotaremos de dos herramientas complementarias; capacidad de clasificar los documentos encontrados, utilizando conjuntamente el modelo frecuentista, y algoritmo de Kohonen, y sensibilidad, para determinar las preferencias o perfil del usuario mediante un estadístico de concordancia llamado Kappa.

Es decir, nuestro experimento propone clasificar de manera automática un conjunto de documentos mediante su contenido semántico, y establecer la categoría o tema al que pertenecen. El usuario que acceda a ellos y seleccione una muestra, estará también definiendo su preferencia a un tema de interés, sin embargo, como el ambiente es dinámico, las preferencias pueden cambiar en la siguiente búsqueda, por ello, si contamos con la suficiente evidencia estadística, podemos demostrar más haya del azar si el usuario ha cambiado de preferencia, y esto le permitirá al modelo redefinir su perfil a través del tiempo.

Con el objetivo de presentar con claridad lo antes mencionado se ofrece en el primer capítulo, llamado Sistema de Búsqueda de Información en Internet, el planteamiento del problema, así como un interesante símil entre la pirámide alimenticia y las herramientas de recuperación de información. Posteriormente se presentan las primeras soluciones a través de motores de búsqueda; arquitectura distribuida, Web semántica y Web cooperativa, concluyendo algunos ejemplos de Sistemas Multiagentes

Posteriormente en el segundo capítulo se mostrarán los temas de Teoría de Recuperación de Información, Agentes y Herramientas de Clasificación, abordaremos el marco teórico respecto al concepto de Web Mining. De igual forma en este capítulo se esquematizan los algoritmos de Inteligencia Artificial, como una alternativa para la clasificación de textos, en particular nos

enfocamos en el algoritmo de Kohonen; pero antes abordaremos el tema del preprocesamiento desde un enfoque vectorial y se presentara el fundamento del estadístico de concordancia Kappa.

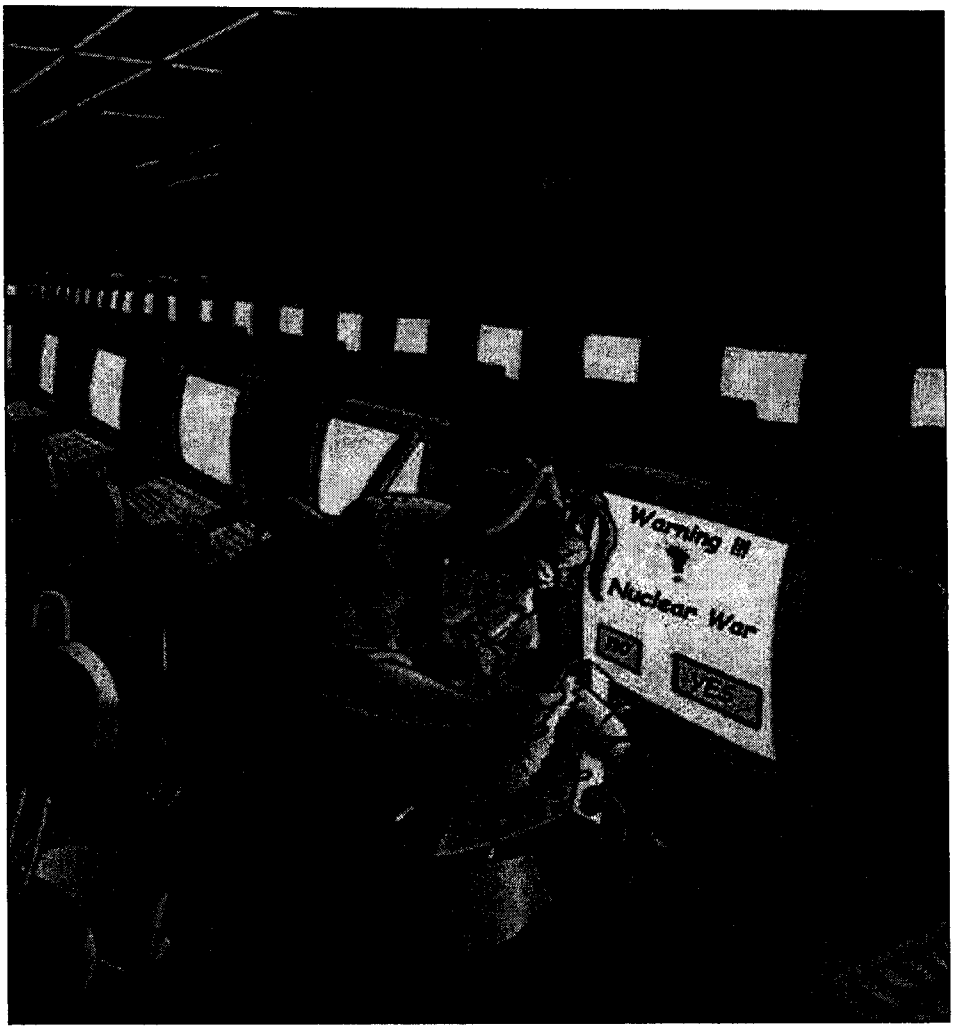
En el capítulo tercero, denominado Estructura del Modelo y Aplicación de Herramientas Propuestas, experimentaremos con la búsqueda de información, usando dos diferentes motores de búsqueda, aplicaremos el preprocesamiento frecuentista a los documentos obtenidos y se clasificaran mediante el uso del algoritmo de Kohonen probándose así su eficiencia y utilizando estas categorías para la aplicación del estadístico Kappa como un efectivo factor de sensibilidad para la selección de ámbitos de búsqueda. Las particularidades y puntos a considerar serán planteados en las conclusiones finales, así como las recomendaciones para el desarrollo de proyectos futuros.

Los anexos incluidos al final de este trabajo de tesis, tienen como objetivo ser un auxiliar didáctico. Por ejemplo, en el anexo I se presenta el código utilizado en las rutinas de preprocesamiento. En el anexo II se presenta la tabla de resultados respecto a la codificación de los documentos seleccionados. En el anexo III se presenta la tabla de resultados arrojados por el algoritmo de Kohonen. En los Anexos IV y V se presentan los cálculos complementarios respecto a la prueba de hipótesis expuesta en el tercer capítulo. Finalmente en el anexo VI se presenta la prueba de Independencia de variables respecto a la tabla de contingencias de donde se obtiene el estimador Kappa.

Búsqueda de Información en Internet.

"Tengo un sueño en dos partes para la Red. Primero veo que se convierte en un medio muy poderoso de comunicación entre los hombres. Luego, en la segunda parte, las computadoras cooperan".

Tim Berners Lee.



Capítulo 1

Introducción.

La información se ha convertido en el estilo de vida predominante en la sociedad de nuestro siglo, actualmente una página del NewYork Times, contiene más datos que los que un hombre en el siglo XIX, pudo conocer en toda su vida. Es la información, materia prima para el trabajo de millones de personas que la generan, codifican, muestran, almacenan y consultan.

El hombre moderno ha podido acumular una cantidad innumerable de información en toda clase de medios, los cuales evolucionan como una máquina autónoma. Una visión futura nos lleva a imaginar que las bibliotecas se transformen en museos donde se exhibirán los libros y documentos como piezas de arte de un valor incalculable. Las consultas históricas, estadísticas, literarias, etc. se realizarán de manera digital. El primer paso hacia este paradigma de acceso al conocimiento es la inagotable fuente "Internet" y todo lo que implica la creación de nuevos sistemas, modelos y arquetipos que darán solución a un vieja pregunta ¿cómo encontrar una aguja en un pajar?.

En el presente capítulo enfrentaremos la necesidad de encontrar información de calidad en Internet, es claro que las herramientas propuestas son sólo unas cuantas piezas en el inmenso rompecabezas del quehacer respecto a uno de los más importantes retos de los profesionales de la matemática y disciplina informática. Para explicarlo plantearemos un problema común en cualquier ámbito de trabajo:

Necesitamos *información* para desarollar nuestras actividades profesionales y personales: la toma de decisiones acerca de una cotización de software, también estamos interesados por algún artículo de tipo científico acerca de los modelos con variables de tipo categórico, y saber si existe un catálogo de direcciones de correo organizado de manera temática, etc. Al analizar nuestra necesidad de información surgen varias preguntas.

Si se lleva algún tiempo investigando sobre los temas mencionados entonces, estamos familiarizados con algunas direcciones Web donde existe tal información. Primer pregunta: ¿mi computadora debería estar familiarizada también con estas fuentes de información?

Es fácil suponer que todo el personal de un departamento se dedica a la realización de tareas similares, por tanto mucha información recabada es de utilidad tanto a unos como a otros, entonces sería conveniente que toda la información de un departamento se almacenara en algún medio y de alguna forma que en el momento que se necesite, pueda ser consultada. Segunda pregunta ¿Cuánto tiempo ahorraría una consulta de esta naturaleza?

Es posible que dentro de una empresa exista una reestructura, y por tanto se reasigne personal de un departamento a otros de la misma dirección, es un hecho que el personal recién instalado va a necesitar información que sabe se encuentra en el compendio de archivos de su anterior departamento. Tercer pregunta, si existe una base de conocimiento en cada departamento, ¿Sería conveniente que estuvieran enlazadas e identificadas por el tipo de información, categoría o ámbitos?

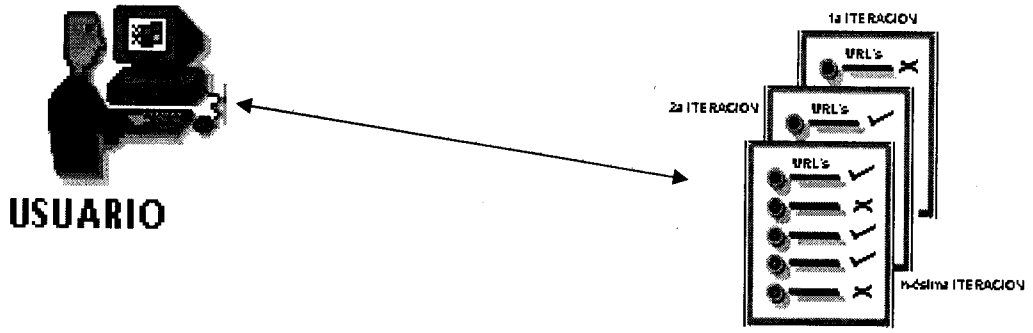
Podemos seguir avanzando, al grado de poner en contacto una base de conocimientos de una empresa nacional con sus filiales en los 5 continentes y el número de preguntas seguiría creciendo.

En la actualidad la fuente de información más amplia y de mayor demanda es Internet, de tal modo que si cada que se encuentra información valiosa en la red se guardan las direcciones Web que interesan y además son marcadas con un distintivo, en el momento que se les requiera serán localizadas de manera oportuna. Parece sencillo, pero en realidad el hecho de recuperar información contenida en páginas Web, implica conocer dichas páginas y saber con seguridad que van a servir y van a estar para cuando se les necesite.

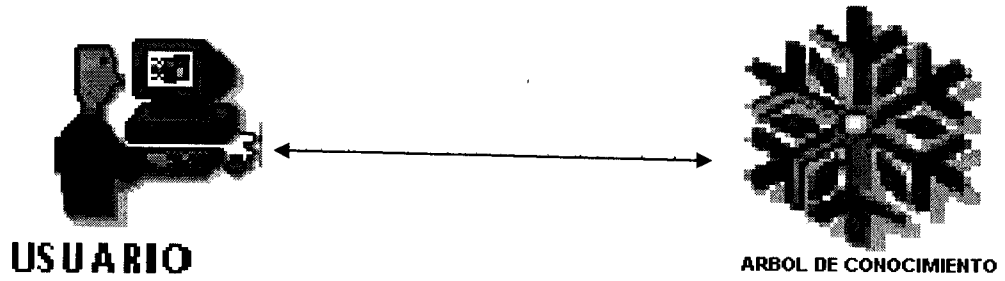
Para llevar a cabo un registro eficaz y oportuno de nuestras búsquedas y hallazgos es necesario, desarrollar un árbol de conocimientos, en donde cada rama represente un tema o ámbito laboral. A continuación, una vez creada la base de nuestro conocimiento, nos interesa cómo extraerlo. Es decir, una manera optimizada de buscar información y si lo que se hallase no es suficiente, iniciaremos la tarea de localizar nueva información pero de manera optimizada, con la ayuda de un sistema inteligente que tome el papel de un asistente que nos evite procesos redundantes, filtre la información y facilite su búsqueda.

Figura 1.0.a Proceso de búsqueda de Información.

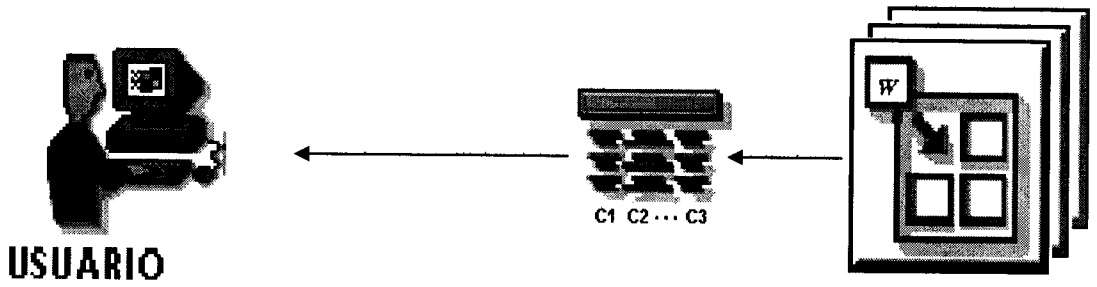
a) búsqueda de información, donde en cada iteración se eligen direcciones Web relevantes



b) El mismo usuario actualiza y consulta información en el árbol de .conocimientos de su departamento



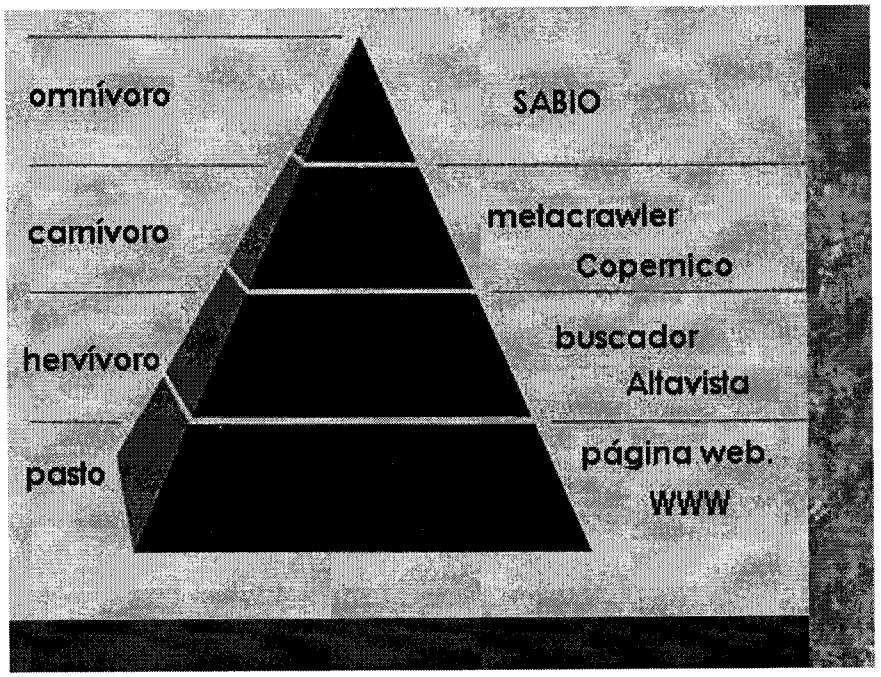
c) Ahora realiza una consulta la cual desencadena una clasificación para definir un ámbito de búsqueda de información.



Actualmente la informática está empezando a poner a disposición de usuarios comunes productos basados en la investigación de Inteligencia Artificial y destinados a facilitarles operaciones frecuentes como la actualización de agenda o la navegación y búsqueda de información por Internet, y en este último aspecto es donde fijaremos nuestro interés para el desarrollo del presente trabajo

Una metáfora muy ilustrativa de la búsqueda en Internet es contemplar la Red como una pirámide alimenticia de información. El conjunto de páginas que la componen están en el nivel inferior de dicho esquema. A continuación, y según ascendemos, tenemos los buscadores de tipo "herbívoro", que "pastan" de las páginas Web, y producen como resultado índices. Aunque hoy en día la gran mayoría de los buscadores están en un nivel muy inferior de la pirámide, la tendencia es ir ascendiendo en la misma, desarrollando buscadores cada vez más "carnívoros", que se alimentan de los resultados producidos por los "herbívoros" (o incluso "carnívoros" que se alimentan a su vez de los "carnívoros" que quedan en un nivel inferior a ellos en la pirámide).

Figura 1.0.b Pirámide alimenticia dentro del contexto de herramientas de búsqueda en Internet.



Los buscadores tradicionales (herbívoros) requieren una gran cantidad de memoria y de ancho de banda para funcionar correctamente. Este gran desembolso sólo es amortizado a costa de recibir multitud de consultas. Aquí no se emplea inteligencia, sencillamente se "ataca" en cada búsqueda como algo nuevo, totalmente independiente de las anteriores ya resueltas.

SABIO es el siguiente paso en la cadena evolutiva, dentro de la pirámide alimenticia de la Información. La arquitectura propuesta es una yuxtaposición de proyectos y trabajos recientes, tanto aplicativos como teóricos, que aportan un medio de búsqueda eficiente de información de calidad, produciendo una solución interdisciplinaria bajo una perspectiva original.

1.1 Información en la Web[1]

Existe la convicción de encontrar en la Web cualquier tipo de información. Creencia firmemente arraigada entre la comunidad científica, puesto que la labor de la mayor parte de los investigadores se ha facilitado desde que Tim Berners-Lee[2] propuso la creación de una red global de *hiperdocumentos* (conocida como "La Web").

Sin embargo, aun cuando la Web permite el acceso a una cantidad enorme de información no es, un mecanismo de localización de información eficiente: no existe un único formato para los documentos[3], las capacidades para proporcionar metainformación son inadecuadas o se utilizan de forma inapropiada, las relaciones (hiperenlaces) que se establecen entre los documentos no siempre aportan información valiosa, etc.

Más aún, la Web por sí sola es completamente inútil como fuente de información puesto que depende de los motores de búsqueda[4], artefactos desarrollados para resolver el problema de localizar un documento en una estructura hipertrofiada y anárquica. Los motores de búsqueda realizan una labor útil pero puede afirmarse que están llegando al límite de sus posibilidades

Puede parecer exagerado afirmar que los motores de búsqueda están siendo rebasados por las necesidades de casi cualquier usuario, pero, como apoyo a esta aseveración presentamos dos relativamente recientes[5] iniciativas del principal buscador de la Web, *Google*. La primera de ellas, *Google Answers*[6] ("Respuestas *Google*"), permite a los usuarios hacer preguntas a otros individuos, expertos en el uso de *Google*, pagando por las respuestas obtenidas. El servicio es definido en los siguientes términos:

---

[1] **The Cooperative Web: A Complement to the Semantic Web** Daniel Gayo Avello, Darío Álvarez Gutiérrez
Department of Informatics, University of Oviedo

[2] ***El británico Tim Berners-Lee es el "inventor y el protector" de la Web, como él mismo dice, definió en 1989, el sistema que daría lugar a la World Wide Web, un conjunto de millones de páginas relacionadas entre sí y que cubre todo el planeta...***

[3] Aunque una gran parte de los documentos disponibles en la Web están escritos en HTML, cada vez es mayor la cantidad disponible en otros formatos (PDF, RTF, PS, etc); este hecho obliga a la conversión de los mismos a texto plano para su procesamiento perdiéndose así cualquier posibilidad de emplear la metainformación existente en el documento original.

[4] Un motor de búsqueda, o simplemente buscador, es un artefacto *software* que explora la Web almacenando en una base de datos el texto de los documentos que analiza. Al ir procesando documentos se crea un índice que emplea las palabras que aparecen en cada página web. Cuando un buscador recibe una consulta toma las palabras utilizadas por el usuario y obtiene los documentos indexados por las mismas. Para presentar los documentos por orden de relevancia se tiene en cuenta el número de apariciones de cada palabra de la consulta en los documentos resultantes y en el cuerpo total de documentos de la base de datos. En definitiva, un buscador Web es una herramienta que trabaja con los documentos a un nivel puramente léxico.

[5] Mayo de 2002.

[6] http://answers.google.com

"El motor de búsqueda de Google es una gran manera de encontrar información en línea. Pero a veces incluso los usuarios experimentados necesitan ayuda para encontrar la respuesta exacta a una pregunta. Google Answers es una forma de conseguir ayuda de expertos en la búsqueda en línea. Al proponer una pregunta usted especifica la cantidad que está dispuesto a pagar por la respuesta y la diligencia con que necesita esa información. Un experto buscará la respuesta y enviará la información que está buscando, así como enlaces útiles a páginas Web sobre el tema. Si usted está satisfecho con la respuesta pagará la cantidad previamente estipulada".

Por otra parte, el *First Annual Google Programming Contest*[7] (Primer Concurso Anual *Google* de Programación), tenía como finalidad *"[...] escribir un programa que haga algo interesante con los datos* [600.000 documentos], *de tal manera que pueda escalar sobre una colección de documentos de un tamaño comparable al de la Web. Tu trabajo* [el del concursante] *es convencernos* [a *Google*] *de que tu programa es interesante y escalable; aparte de esto, puedes hacer cualquier cosa que se te ocurra. (Google, 2002b)"* En este documento se sugieren algunas posibles entradas para el concurso como "detectar páginas prácticamente iguales" o "clasificar páginas por tema".

A la vista de esto, se puede decir que *Google* reconoce implícitamente que las técnicas que emplea en la actualidad ya se han agotado.

Es claro que el futuro de la recuperación de Información en la Web será a través de la clasificación temática, y para esto existen tendencias vertidas hacia la semántica pura, ontologías y clasificadores automáticos a través de palabras clave.

Para situarnos en el origen del problema objeto de estudio nos podemos referir a los sistemas de recuperación de información textual. Dichos sistemas permiten almacenar un gran volumen de documentos escritos en lenguaje natural y facilitan a los usuarios la tarea de localizar documentos específicos mediante la formulación de consultas, que no se realizan en ningún lenguaje especial sino que emplean una o más palabras que describen los intereses del usuario pudiendo utilizar, además, operadores lógicos como AND y OR. Al recibir una consulta de este tipo el sistema proporciona como resultado una lista de documentos que la satisfacen.

[7] http://www.*Google*.com/programming-contest y http://www.*Google*.com/programming-contest/winner.html.

La efectividad de semejantes sistemas puede evaluarse basándose en dos medidas: rememoración (*recall*) y precisión. La primera indica la proporción retornada del material total que satisface la consulta mientras que la segunda indica el grado de relevancia que ese material tiene para el usuario. Idealmente, ambas medidas deberían aproximarse al 100%, sin embargo, en los sistemas reales evolucionan de forma inversa. Es decir, al retornarse más documentos para una consulta la relevancia de los mismos es pequeña y al aumentarse la precisión disminuye el número de documentos retornados (muchos textos relevantes no aparecen en los resultados). La tarea de equilibrar precisión y rememoración queda, generalmente, en manos del usuario a la hora de formular sus consultas con mayor o menor grado de detalle. Sin embargo, Salton y McGill[8] hacen especial hincapié en que debería ser el propio sistema el encargado de establecer mecanismos destinados a aumentar la precisión sin disminuir por ello la tasa de rememoración.

Dejando a un lado su naturaleza distribuida y los detalles tecnológicos, la Web[9] puede entenderse como un sistema de recuperación de información textual. Sería susceptible, por tanto, de ser evaluado en términos de rememoración y precisión y, desgraciadamente, está aquejado de los mismos problemas que los sistemas de hace veinte años aunque a una escala mucho mayor: la tasa de rememoración de la Web es enorme y la necesidad de aumentar la precisión es acuciante.

La Web nació en el *CERN*[10] como un sistema de intercambio de información (Berners-Lee, 1989). Su objetivo básico era evitar la pérdida de información inherente a una gran organización así como facilitar el acceso a la información disponible. Dos características fundamentales de la propuesta han convertido a la Web en lo que es en la actualidad: su naturaleza distribuida y la posibilidad de establecer vínculos entre los documentos.

Por otro lado, la propuesta original de Berners-Lee insistía en la necesidad de hacer el sistema suficientemente atractivo para animar a los usuarios a incorporar información al mismo, de tal forma que su utilidad creciese al añadirse nuevos documentos y esa utilidad creciente impulsase, a su vez, a seguir aumentando la base de documentos.

---

[8] Gerard Salton & Michael J. McGill, "Experiments in automatic thesaurus construction for information retrieval", Ed. McGraw Hill, 1983, Primer libro de Gerard Salton. El gran gurú de la "informática documental" inicia su carrera profesional. Todos los libros del autor resultan muy interesantes para cualquiera interesado en el procesamiento automático de información procedente de documentos en lenguaje natural.

[9] En realidad, el binomio Web y buscadores.

[10] Laboratorio Europeo de Física de las Partículas con sede en Ginebra Suiza.

En ese documento se hacen algunos señalamientos extraordinariamente interesantes sobre los posibles problemas para recuperar información en semejante sistema; algunos de los puntos más interesantes hacen referencia al uso de palabras clave como un método habitual para acceder a documentos cuya localización es desconocida.

Sin embargo también asume que dos usuarios nunca emplean las mismas palabras clave. Asimismo estas palabras clave pueden representar un concepto atado a un nodo, estableciendo así un sistema de enlaces y de esta manera documentos sobre temas similares estarían vinculados en un conjunto reducido de nodos conocidos.

En síntesis se plantea el desarrollo de un sistema con enlaces que permita a los usuarios navegar a través de conceptos, documentos, sistemas y autores, permitiendo, asimismo, almacenar referencias entre documentos.

Así pues, la intención original era construir la Web partiendo de una base semántica empleando "nodos conceptuales" que serían apuntados desde los distintos documentos. Sin embargo, para implementar una "navegación conceptual" que emplease dichos nodos a modo de pasarela sería necesario que los enlaces fuesen bidireccionales dificultando enormemente el desarrollo de la Web[11]. Por otro lado, la propia definición de cada "nodo conceptual", su relación con otros nodos de ese tipo, la resolución de incongruencias, etc., plantean toda una serie de problemas que, sin duda, empujaron a los iniciales desarrolladores de la Web a optar por un esquema más simple, análogo al hipertexto tradicional con la salvedad de su naturaleza distribuida. Y es en ese momento preciso cuando el destino de la Web como sistema de recuperación de información quedó sellado. Al eliminar los "nodos conceptuales" se dispone de un artefacto diseñado para crecer de un modo cada vez más acelerado sin incluir ningún tipo de mecanismo capaz de facilitar la localización de un documento en particular. No obstante, sería un error interpretar esto como una crítica hacia la forma en que se implementó finalmente la Web, esta decisión de diseño facilitó su desarrollo y posterior crecimiento y, a fin de cuentas, desde la puesta en marcha del primer servidor Web aún transcurrieron tres años hasta que la necesidad de un sistema de búsqueda de información para la Web se hizo apremiante.

[11] En la Web actual sólo es necesario conocer el destino para construir un enlace, no es necesario el conocimiento por parte del nodo destino. Sin embargo, los enlaces bidireccionales precisarían que origen y destino se reconocieran mutuamente.

1.1.1 Motores de búsqueda

A continuación, se presenta un esquema evolutivo de los motores de búsqueda más sobresalientes, respecto a la etapa cronológica en que fueron presentados como solución para la búsqueda de información en la Web.

Propuesta	Tipo de buscador	Requiere registro previo de documentos	Información empleada para determinar relevancia
Los primeros motores de búsqueda			
ALIWEB (Koster, 1994)	Robot	SI	Uno de los principales inconvenientes de *ALIWEB* radicaba en que la información almacenada por cada página era muy escasa. (título, descripción y algunas palabras clave)
WebCrawler (Pinkerton, 1994)	Robot	NO	Indexar todo el texto de cada documento. Esta estrategia permite aumentar la capacidad de rememoración del sistema pero, al mismo tiempo, al crecer el volumen de documentos explorados la precisión de las respuestas se reduce.
Lycos (Mauldin y Leavitt, 1994)	Robot	NO	Iniciativa intermedia entre *ALIWEB* y *WebCrawler* puesto que no indexaba ni el texto completo de los documentos ni únicamente el título y la descripción. En su lugar generaba una versión "ligera" de los documentos constituida por el título, las primeras 20 líneas y las 100 palabras más relevantes según el criterio de (Salton y McGill, 1983).
MetaCrawler. Selberg y Etzioni, (1995)	Metabuscador (WebCrawler)	NO	Dicho sistema recibía consultas de los usuarios y las lanzaba contra los "buscadores" más populares de la época (entre los que se encontraban *WebCrawler* y *Lycos*, así como directorios como *Galaxy* y *Yahoo!*). Los resultados recibidos eran filtrados, eliminando los enlaces inaccesibles y los que eran considerados irrelevantes
Motores de búsqueda modernos			
HITS (Chakrabarti *et al*, 1998a) y *CLEVER* (Chakrabarti *et al*, 1998b).	Robot	NO	Utiliza el texto completo junto con algoritmo de Kleinberg; valoración indizada por estructura hipertextual de la Web. (Autoridades y Concentradores)
Google (Brin y Page, 1998)	Robot	NO	Algoritmo PageRank, extiende algoritmo de Kleinberg. El valor PageRank "Fluye" y se distribuye entre los documentos en función de sus vínculos.

Como hemos visto en el esquema anterior existe un elemento de modernidad que separa en dos grupos a los motores de búsqueda: el algoritmo de Kleinberg.

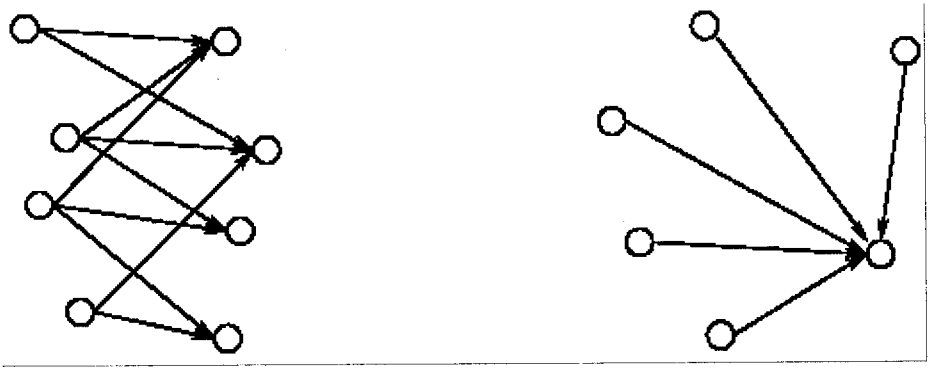
1.1.2 Algoritmo de Kleinberg

Jon Kleinberg sentó en un texto ya clásico (Kleinberg, 1998) las bases sobre las que descansarían los modernos motores de búsqueda al presentar los conceptos de "autoridad" y "concentrador" (*hub*) que aprovechan, precisamente, la estructura global de la Web.

Kleinberg presta atención al problema de la precisión/relevancia de los resultados ofrecidos por los buscadores y señala que semejante característica es totalmente subjetiva y precisa de una evaluación humana. Así pues, ¿sería posible calcular la relevancia, de forma puramente algorítmica?

Para llevar a cabo esta tarea, Kleinberg definió los conceptos de "autoridad" y "concentrador". Una "autoridad" sería un documento fuertemente vinculado, esto es, un documento al que muchos otros documentos apuntan. Cada uno de estos enlaces podría considerarse un "voto" a favor del documento destino y, puesto que cada enlace fue establecido por una persona, dicho voto estimaría indirectamente la calidad del documento desde una perspectiva humana. Analizando los textos empleados como enlaces al documento destino junto con el texto del mismo podría determinarse para qué términos o materias, el documento en cuestión es una autoridad. Por su parte, un "concentrador" sería un documento con enlaces a muchas "autoridades".

Figura 1.1.1.a. Concentradores (izquierda) y autoridad (derecha).



- **MATICES DEL ALGORITMO DE KLEINBERG**

Sin embargo, las técnicas propuestas por Kleinberg, (1998) y Chakrabarti (1998) aun cuando suponen un salto cualitativo frente a los antiguos buscadores como *Lycos* o *WebCrawler* no son ni mucho menos perfectas. Bharat y Henzinger, (1998) plantean tres situaciones en las cuales

dichos algoritmos se comportan de manera pobre: relaciones entre *hosts* mutuamente reforzadoras, enlaces generados automáticamente y documentos no relevantes.

En el primer caso, si varios documentos alojados en un único servidor apuntan a un único documento externo dichos enlaces cuentan en el algoritmo de Kleinberg como votos diferentes aumentando, de manera injusta, la puntuación del documento destino. Bharat y Henzinger plantean la necesidad de reducir el peso otorgado a enlaces que parten desde un único *host* a un único documento.

El segundo caso hace referencia a los enlaces introducidos por herramientas de desarrollo, proveedores de servicios Web, etc. Dichos enlaces no han sido creados por una persona y, en consecuencia, no representan ninguna valoración sobre la calidad del documento apuntado.

El último caso se produce cuando un documento no relevante (por su contenido) está vinculado desde una autoridad o un concentrador, considerándolo entonces y de manera incorrecta como un documento relevante. Un buen ejemplo serían las páginas personales de los autores de documentos muy referenciados.

Para tratar de solucionar estos problemas Bharat y Henzinger complementan el algoritmo de Kleinberg con dos nuevas características. Por un lado, limitan el peso otorgado a enlaces a un documento que parten de un único *host*. Esta solución, sin embargo, sigue planteando problemas puesto que un único *host* puede albergar documentos de distintos autores y, por tanto, sus votos serían independientes e injustamente devaluados con el nuevo algoritmo. Un estudio muy interesante y más riguroso sobre los enlaces nepotistas[12] puede verse en Davison, (2000a).

Por otro lado, realizan un análisis del contenido de los documentos obtenidos por el algoritmo para determinar su relevancia en relación a la consulta realizada. Emplean, para ello, algoritmos clásicos de recuperación de información y para tratar de solventar los inconvenientes de las palabras clave utilizan un algoritmo de *stemming*. Los algoritmos de *stemming* tienen como finalidad reducir una palabra a su raíz, así, por ejemplo, anduve, andaré y ando "colapsarían" a and. La idea es muy atractiva pero plantea el problema de que es total y absolutamente dependiente del idioma. Así, en el prototipo descrito se emplea el algoritmo de Porter, que

[12] Aquellos enlaces que apuntan a un documento no por su mérito sino para "forzar" su popularidad.

implementa directamente en el código las reglas para determinar las raíces de palabras inglesas Figura 1.1.2.c. Por tanto, aún cuando los resultados del prototipo eran, según sus autores, un 45% mejores que los obtenidos con el algoritmo de Kleinberg, el sistema sería muy difícilmente aplicable a la Web de manera global por cuestiones lingüísticas.

De todos modos, también se han señalado algunos inconvenientes de la primera mejora puesto que es muy difícil, por no decir imposible, determinar si dos documentos hospedados en un mismo servidor pertenecen al mismo autor (o si dos documentos hospedados en servidores diferentes pertenecen a autores distintos).

En resumen, esta iniciativa matiza algunos aspectos del algoritmo de Kleinberg y señala la necesidad de analizar el contenido de los documentos. Sin embargo, el enfoque planteado para llevar a cabo este análisis salvando el problema de las palabras clave no es el más adecuado para lograr un sistema independiente del idioma.

Figura 1.1.2.c. Fragmento del algoritmo de *stemming* de Porter.

```
/* step3() deals with -ic-, -full, -ness etc. similar strategy to step2. */
void step3() { switch (b[k])
{
case 'e': if (ends("\05" "icate")) { r("\02" "ic"); break; }
if (ends("\05" "ative")) { r("\00" ""); break; }
if (ends("\05" "alize")) { r("\02" "al"); break; }
break;
case 'i': if (ends("\05" "iciti")) { r("\02" "ic"); break; }
break;
case 'l': if (ends("\04" "ical")) { r("\02" "ic"); break; }
if (ends("\03" "ful")) { r("\00" ""); break; }
break;
case 's': if (ends("\04" "ness")) { r("\00" ""); break; }
break;
} }
```

Obsérvese que el algoritmo de Porter implementa las reglas para obtener las raíces de las palabras en inglés directamente. En el ejemplo que se presenta se muestra la forma en que palabras como usefullness, serían reducidas en una primera iteración a usefull y en una segunda a use. El algoritmo implementa de manera análoga reglas para trabajar con plurales, tiempos verbales, etc. Aunque es factible aplicar esta técnica a otros idiomas, requiere un desarrollo específico para cada uno y, en función de la calidad del algoritmo implementado, los resultados pueden variar mucho de un idioma a otro.

- **GOOGLE**

Si la Web marcó un antes y un después en Internet, *Google* (Brin y Page, 1998) ha tenido un efecto similar sobre la Web. El núcleo de este buscador es el algoritmo de *PageRank* (Page *et al*, 1998) que, aunque muy similar al algoritmo de Kleinberg, aporta ideas innovadoras que permiten evaluar la relevancia de un documento de un modo desconocido hasta aquel momento. El algoritmo otorga a cada documento un valor, también denominado *PageRank*, que indica la relevancia objetiva del mismo. El cálculo de dicho valor extiende las ideas de autoridades y concentradores al no dar el mismo peso a todos los enlaces y propagar el *PageRank*[13] de un documento a los documentos que apunta. Así, documentos muy referenciados (autoridades) tendrán valores altos y, esto es una novedad, documentos escasamente referenciados pero desde documentos autorizados "heredarán" valores de *PageRank* elevados.

El buscador *Google* utiliza el algoritmo de *PageRank* junto con otras medidas (texto de los enlaces, posición de las palabras clave dentro del documento, etc.) para ordenar los resultados obtenidos y presentárselos al usuario. De esta forma, los documentos que se ofrecen en primera instancia serán los más relevantes (al estilo de Kleinberg) pero sin eliminar la posibilidad de consultar otros documentos no considerados tan relevantes por el algoritmo.

Al hacer esto, el buscador vuelve a lograr unas tasas de rememoración elevadas (el algoritmo de Kleinberg no muestra todos los documentos relevantes, sólo las autoridades) a costa, nuevamente, de sacrificar la precisión puesto que los documentos menos relevantes retornados por *Google* podrían ser descartados si se pudiera conocer algo más acerca de los intereses del usuario que realiza la consulta.

- **SOBRECARGA DE INFORMACIÓN**

Como vemos, la Web parece tropezar una y otra vez con el problema de la escasa relevancia de los documentos. En los inicios de la Web bastaba con ser capaces de encontrar el documento que se buscaba (*ALIWEB*, *WebCrawler* o *Lycos*). Posteriormente, el acelerado crecimiento de la Web llevó a un nuevo desequilibrio entre precisión y rememoración obligando a desarrollar nuevas técnicas (*MetaCrawler* –un metabuscador–, algoritmo de Kleinberg,

[13] Normalizado por el número de enlaces de salida.

CLEVER y, por fin, *Google*) que mejoraron parcialmente la situación. Sin embargo, la Web sigue en expansión y las técnicas que facilitan su exploración exhaustiva hacen que cada vez sean más los documentos relevantes para consultas formuladas mediante palabras clave.

El problema ahora, más que una falta de precisión es una sobrecarga de información. Este problema es casi tan antiguo como Internet[14]. Los usuarios comenzaron a sufrir la sobrecarga de información con los mensajes de correo electrónico y, especialmente, con los artículos publicados en *USENET*.

Posteriormente, con la Web se produjo una situación semejante ante los resultados devueltos por los buscadores para las consultas más comunes. Durante la última década se han realizado múltiples propuestas destinadas a aliviar esta situación. Algunas se propusieron específicamente para alguno de los servicios antes mencionados (correo electrónico, *USENET* o Web) mientras que otras pretendían filtrar todo tipo de información procedente de Internet.

En el último punto de este capítulo se presentará un robusto compendio de aplicaciones en las que se pueden encontrar algunas de las iniciativas propuestas para resolver el problema objeto de estudio: la recuperación de información relevante en la Web de forma transparente para el usuario.

[14] Algunos datos curiosos: La primera referencia en *USENET* a la expresión *"information overload"* fue hecha por Warren Montgomery el 22 de diciembre de 1982 en net.auto; este usuario sugería dividir el grupo en tres subgrupos para *"facilitar el problema de la sobrecarga de información (Montgomery, 1982)"*. El primer artículo científico, del que el autor tiene constancia, que hizo referencia al problema fue (Hiltz y Turoff, 1983).

1.2 Búsqueda Distribuida, Semántica y Cooperativa

Se inicia la exposición de *soluciones* para la búsqueda de información, ubicándonos en su primer proveedor, la Web. La estructura de hipertexto de la cual ya hablamos en el punto anterior. Se expondrán tres proyectos hasta cierto punto complementarios para la búsqueda de información. El primero es una solución especifica para ámbitos acotados, el segundo es un enfoque relacionado con la semántica de los documentos y finalmente la tercer propuesta es la base del desarrollo presentado como objetivo de este trabajo y funciona mediante la taxonomía de los documentos y la cooperación de agentes de software; como lo veremos a continuación:

Búsqueda	Concepto
Distribuida	El servicio se orienta a comunidades temáticas que publican información sobre un dominio del conocimiento. El objetivo es instrumentar una red de propagación de mensajes y respuestas, conformada por todos los proveedores de información, a efecto de poder satisfacer consultas de usuarios en forma distribuida y en tiempo real.
Semántica	La idea básica que subyace a esta propuesta es la de marcar los documentos disponibles en la Web mediante "etiquetas semánticas" que proporcionarían metainformación sobre el texto marcado (Por ejemplo, profesión, número de teléfono, dirección postal, etc.) Los textos así etiquetados serían procesados de forma sencilla por agentes software, con el objetivo de desarrollar ontologías, y utilizar las clases y relaciones definidas en una o más de esas ontologías para marcar zonas específicas de un documento.
Cooperativa	La Web Cooperativa, por su parte, pretende utilizar el texto completo del documento, sin ningún tipo de etiquetado, como fuente de semántica. ¿Es esto posible sin "comprender" el significado del texto? Es decir, procesar lenguaje natural para obtener, de manera totalmente automática, una clasificación conceptual de documentos. La construcción de esta propuesta se basa en el siguiente símil: Un documento puede considerarse como un individuo de una población. Entre los seres vivos un individuo está definido por su genoma, el cual se compone de cromosomas que se dividen en genes construidos a partir de bases genéticas. De forma similar, los documentos están compuestos por pasajes que se dividen en sentencias construidas mediante conceptos. Siguiendo esta analogía resulta obvio que dos documentos estarán semánticamente relacionados si sus respectivos <<genomas>> son similares y resulta así mismo evidente que grandes diferencias entre dichos genomas implicarían una relación semántica baja.

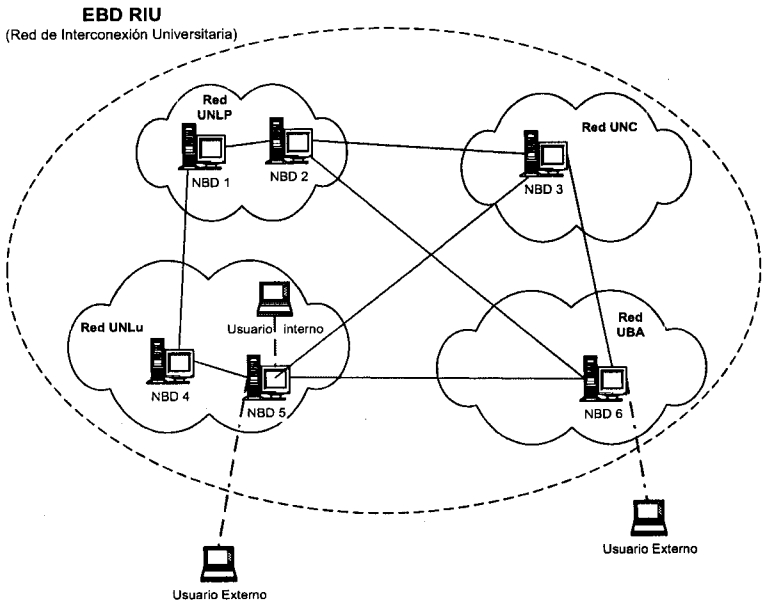
Búsqueda Arquitectura

La diferencia entre los métodos de búsqueda tradicionales y la *arquitectura distribuida*, radica en como se recolecta la información y dónde se ejecutan las búsquedas. Básicamente, se reemplaza una base de datos centralizada de índices por un conjunto de bases de datos distribuidas [Figura 1.2.a], situadas donde reside la información. De esta manera se elimina la necesidad de utilizar un módulo *spyder* para recorrer los sitios, evitando el problema de la frecuencia de actualización. Se pasa entonces a un modelo de búsqueda en tiempo real, es decir, sobre índices completamente actualizados.

La base para soportar esta arquitectura cooperativa es el protocolo compañero a compañero Gnutella[15]. Este protocolo permite establecer una red a nivel de aplicación, basada en nodos cooperantes[16] que intercambian mensajes propios y de terceros.

Figura 1.2.a. . Topología de la arquitectura de búsqueda distribuida

Distribuida



[15] En marzo del año 2000, la empresa Nullsoft que se destaca por ser propietaria del producto Winamp, lanzó públicamente el código de un "clon" del Napster denominado Gnutella, creado por Justin Frankel y Tom Peper.

[16] Se entiende por nodos cooperantes al conjunto de estaciones de trabajo, servidores y demás equipos conectados en una red de trabajo con objetivos comunes, el mejor ejemplo nos lo ofrecen las comunidades universitarias.

Búsqueda **Arquitectura**

Aplicaciones emblemáticas de esta propuesta son: SHOE, Luke y colaboradores (1996) y WebKB (Craven et al, 1998); la primera desarrollan una de las primeras iniciativas destinadas a proporcionar un lenguaje de marcado semántico, una extensión del lenguaje HTML[17] que permite el uso de ontologías[18] [Figura 1.2.b.], clases y relaciones definidas en una o más de esas ontologías para marcar zonas específicas de un documento HTML [Figura 1.2.c.]. Luke y sus colaboradores describen así mismo una herramienta, *Expose*, que explora la Web en busca de páginas anotadas con *SHOE* y almacena los asertos que encuentra en una base de conocimiento. Dicha base de conocimiento puede utilizarse posteriormente para realizar consultas. Posteriormente surge Ontobroker (Fensel et al, 1998) iniciativa muy similar a las anteriores, sin embargo, evolucionaría hacia On2broker (Fensel et al, 1999) cuya principal novedad fue la utilización de tecnologías como XML[19] (Bray et al, 2000) o RDF[20] (Lassila y Swick, 1999).

Figura 1.2.b. Ontología expresada en SHOE.

```
<HTML>
<HEAD>
<META HTTP-EQUIV="SHOE"
CONTENT="VERSION=1.0">
</HEAD>
<BODY>
<ONTOLOGY ID="cs-dept-ontology"
VERSION="1.0">
<USE-ONTOLOGY ID="base-ontology"
VERSION="1.0" PREFIX="base"
URL="http://www.cs.umd.edu/projects/plus/SHOE
/base.html">
<DEF-CATEGORY NAME="Organization"
ISA="base.SHOEEntity">
<DEF-CATEGORY NAME="Person"
ISA="base.SHOEEntity">
<DEF-CATEGORY NAME="Publication"
ISA="base.SHOEEntity">
<DEF-RELATION NAME="member">
<DEF-ARG POS="FROM" TYPE="Organization">
<DEF-ARG POS="TO" TYPE="Person">
</DEF-RELATION>
<DEF-RELATION NAME="publicationAuthor">
<DEF-ARG POS="FROM" TYPE="Publication">
<DEF-ARG POS="TO" TYPE="Person">
</DEF-RELATION>
<DEF-RELATION NAME="publicationDate">
<DEF-ARG POS="FROM" TYPE="Publication">
<DEF-ARG POS="TO" TYPE=".DATE">
</DEF-RELATION>
</ONTOLOGY>
</BODY>
```

Figura 1.2.c. Documento HTML anotado mediante la ontología anterior SHOE

```
<HTML>
<HEAD>
<META HTTP-EQUIV="SHOE" CONTENT="VERSION=1.0">
<TITLE> Página de Tim Berners-Lee </TITLE>
</HEAD>
<BODY>
<P> Esta es la página web de Tim Berners-Lee.
<P> Soy miembro del Consorcio W3C.
<P> Yo inventé la Web, no Al Gore,_
<A HREF="http://www.w3.org/History/1989/_
proposal.html">ésta es la prueba</A>.
<INSTANCE KEY="http://www.w3.org/People/_
Berners-Lee/">
<USE-ONTOLOGY ID="cs-dept-ontology"_
URL="http://www.cs.umd.edu/projects/plus/SHOE_
/onts/cs.html" VERSION="1.0" PREFIX="cs">
<RELATION NAME="member">
<ARG POS=FROM VALUE=http://www.w3.org>
</RELATION>
<RELATION NAME="cs.name">
<ARG POS=TO VALUE="Tim Berners-Lee">
</RELATION>
<RELATION NAME="publicationAuthor">
<ARG POS=FROM
VALUE="http://www.w3.org/History/1989/_
proposal.html">
</RELATION>
</INSTANCE>
</BODY>
</HTML>
```

Semántica

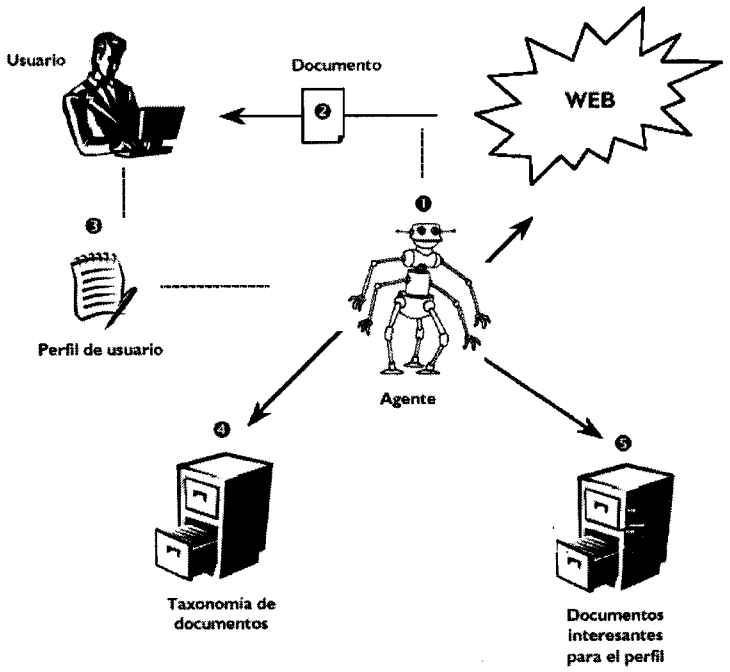
17 Acrónimo inglés de Hyper Text Markup Language (lenguaje de marcación de hipertexto), Hiper Text Meta Lenguaje.
18 Una ontología es la especificación de una conceptualización. Esto es, una descripción de los conceptos y relaciones que pueden existir para un agente o una comunidad de agentes (Gruber, 1993).
19 Acrónimo del inglés eXtensible Markup Language (lenguaje de marcado ampliable o extensible).
20 Siglas de Resource Description Framework, la especificación de un modelo de metadatos desarrollado por W3C.

Búsqueda Arquitectura

Cooperativa

La Web Cooperativa [Figura 1.2.d.] que se sustenta en tres principios básicos:
- La utilización de conceptos, generados automáticamente, como punto intermedio entre las ontologías y las palabras clave.
- La clasificación de documentos en una taxonomía a partir de tales conceptos.
- La cooperación entre usuarios, en realidad, entre agentes que actúan en representación de los usuarios y que no requieren su participación explícita.

Figura 1.2.d. Funcionamiento básico de la Web Cooperativa.



Los agentes de la Web Cooperativa pueden resolver consultas de los usuarios además de explorar en representación de los mismos (recomendar documentos cuyos contenidos pueden ser interesantes). El agente puede examinar el histórico de navegación del usuario o recibir una consulta[21]. Con esta información el agente lleva a cabo una exploración taxonómica, es decir, clasifica dentro de la taxonomía conceptual los datos de partida y obtiene como resultados documentos próximos en el dendrograma. Estos documentos son proporcionados al usuario como recomendaciones en caso de que el agente haya actuado de mutuo propio o como resultados de una consulta.

[21] Una consulta puede estar constituida por una serie de palabras clave, un fragmento de texto o el URL de un documento completo.

Búsqueda	Fortalezas y Oportunidades
Distribuida	El modelo se considera viable para comunidades reducidas (no más de una centena de nodos). A partir de la obtención de un prototipo se estudiará cuál es la cantidad óptima de integrantes de un EBD[22] para que el modelo resulte eficiente. Muchas comunidades universitarias, proyectan realizar la construcción de un prototipo de red de consulta distribuida, a los efectos de validar la idea propuesta y probar el comportamiento en un ambiente de operación real.
Semántica	Como se puede ver, la Web Semántica depende totalmente de las ontologías, razón por la cual se están dedicando grandes esfuerzos tanto a la construcción automática de ontologías y al marcado semántico de los documentos (Erdmann *et al*, 2001). Esta dependencia es la causa de las dos críticas principales que se pueden hacer a la Web Semántica. En primer lugar, la construcción de ontologías realmente útiles con gran número de clases y de relaciones entre las mismas requerirá siempre supervisión humana. En segundo lugar, las ontologías que se vienen desarrollando y el tipo de consultas que mejor puede resolver la Web Semántica son metasemánticas, es decir, literalmente más que semánticas y objetivamente seudosemánticas. En resumen, la Web Semántica y sus ontologías pueden facilitar muchísimo el procesamiento de información en entornos bien definidos (publicaciones científicas, comercio electrónico, etc.) así como la construcción de agentes capaces de deducir nuevos conocimientos en semejantes entornos. Sin embargo, es muy difícil aplicar estos conceptos a la Web en su totalidad.
Cooperativa	La Web Cooperativa es una capa situada directamente sobre la Web actual con el fin de dotarla de semántica de manera global, automática, transparente e independiente del idioma. Requiere la participación de los usuarios pero no de forma consciente y directa sino indirectamente a través de agentes autónomos y cooperantes. La Web Cooperativa se apoya sobre el uso de conceptos y taxonomías documentales, unos y otras pueden obtenerse, sin intervención humana, a partir del texto libre de los documentos.

[22] Espacio de Búsqueda Distribuida

1.3 Arquitecturas y Aplicaciones de Sistemas Multiagentes.

Actualmente "agente" se ha convertido en la palabra de moda en la informática mundial. A cualquier programa especializado en algún tipo de función con interfaz amigable para el usuario, se le denomina agente. Desde la perspectiva de la Inteligencia Artificial Distribuida[23], no tiene sentido hablar de agente si no es multiagente, es decir si no tiene una capacidad de comunicación con otros agentes a través de la red.

La solución multiagente del problema planteado de los buscadores del Web, consiste en crear agentes que juegan un papel similar al de una secretaria electrónica capaz de actuar en representación de su usuario en el "ciberespacio", navegar en él y de preguntar a otros agentes acerca de la información que está buscando; por otra parte, los sistemas servidores tendrán a su vez uno o varios agentes que los representan en la red y que saben dialogar proporcionando información acerca de las características del servidor en cuestión y de la información que posee. Cada vez que se realiza un cambio en un sistema servidor sólo será necesario introducir el conocimiento correspondiente en el agente del servidor y éste se encargará de informar los cambios a los agentes que se contacten con él.

A continuación se examinarán diferentes estructuras y aplicaciones de sistemas que utilizan el nuevo paradigma de programación orientada a agentes, como solución en la búsqueda de información en la Web. Se debe tener en consideración, que aún cuando en la descripción de la aplicación se habla de sólo un agente, éste tiene que interactuar con otros dentro de la red, regresando a la estructura multiagentes, es por esto que el planteamiento de la Inteligencia Artificial Distribuida prevalece como concepto general.

1.3.1 Arquitecturas Multiagentes [24]

Finalmente se establece como Arquitecturas Multiagentes a la estructura de sistemas computacionales en los cuales varios agentes semiautónomos (programas) interactúan entre sí ya sea para colaborar en la solución de un conjunto de problemas o en la consecución de una serie de objetivos individuales o colectivos. Estos agentes informáticos pueden ser homogéneos o heterogéneos, pueden tener metas comunes e involucrarán algún grado de comunicación

[23] Disciplina que estudia la resolución de un problema de forma colaborativa por un grupo distribuido de entidades o agentes inteligentes.
[24] http://www.gsi.dit.upm.es

entre ellos. Cada uno de estos agentes individuales puede o no tener comunicación directa con seres humanos.

La utilidad de pensar en multiagentes reside en que en el entorno de sistemas distribuidos en redes abiertas. En el caso de buscadores de información en la Web, contamos con sistemas capaces de realizar búsquedas especializadas sobre temas que nos interesan utilizando uno o varios buscadores ya disponibles, Pero estos buscadores evolucionarán y ofrecerán nuevos servicios, o si aparecen nuevos buscadores, el sistema que los utiliza no podrá darse cuenta, y es, porque los sistemas abiertos no pueden por naturaleza definirse plenamente de antemano. Por ello cada componente necesita proporcionar mayores niveles de modularidad o autonomía que la que proporciona la tecnología orientada a objetos simple. El tipo de comunicación entre los sistemas debe ser más rica que el simple paso de mensajes, por lo que si bien puede ser deseable que estos sistemas se escriban en lenguajes orientados a objetos, como Java, los esquemas de comunicación entre agentes deben permitir manejar protocolos complejos de comunicación y negociación.

1.3.1.1 MIX / MAST

Uno de los modelos más sencillos es la arquitectura MIX la cual inicia con el proyecto ESPRIT ``*Basic Research Project MIX: Modular Integration of Connectionist and Symbolic Processing in Knowledge Based Systems*". Tiene como intención establecer un marco distribuido de propósito general para la cooperación de múltiples agentes heterogéneos. La arquitectura define dos entidades básicas: los agentes y la red en la que interactúan. Se distingue claramente entre agentes de red (aquellos usados para la comunicación entre agentes), y agentes de aplicación.

El modelo de red ofrece una visión uniforme a los agentes de ésta. El modelo distingue tres capas: la capa de interfaz, la capa de mensajes y la capa de transporte. Los agentes de red, denominados agentes Páginas Amarillas (YP o ANS), se encargarán de controlar el nacimiento/muerte de los demás agentes, así como de la creación de grupos de agentes.

A pesar de que la arquitectura permite la inserción de agentes con diversos modelos, se hace la implementación de agentes siguiendo el modelo de agente orientado a objeto. Los agentes tiene una base de datos interna, un buzón y un componente de control; para definir la estructura se diseñó ADL (*Agent Description Language*) que permite tanto asociaciones ``débiles" de agentes (vía paso de mensajes) como asociaciones ``fuertes" (acceso directo a variables).

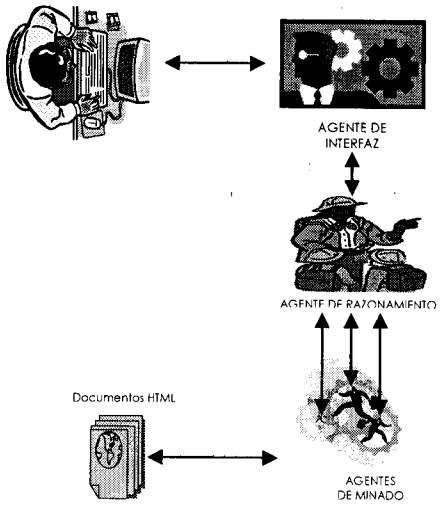
1.3.1.2 MultiAgent Cooperative Information Gathering (MACRON)

Existen muchos trabajos de investigación los cuales proponen la aplicación del paradigma multiagentes para la resolución de problemas involucrados con la búsqueda de Información, sin embargo encontramos en la arquitectura MACRON uno de los mejores ejemplos. Diseñada por el Departamento de Informática de la Universidad de Massachussets como una forma de resolver problemas de sistemas CIG (*Cooperative Information Gathering*). Está por tanto, más enfocada al problema de búsqueda y recopilación de información. Según sus autores es capaz de explotar las interdependencias entre subproblemas, gestionar la incertidumbre inherente a búsquedas en sistemas multiagente, hallar el equilibrio entre calidad y límites en los recursos y, evitar redundancia.

Básicamente el diseño establece tres tipos de agentes autónomos: agentes de interfaz de usuario, agentes de razonamiento y agentes de minado de información en la red. La descomposición en agentes es vertical, como se puede comprobar en la Figura 1.3.1.2.a., pero permitiendo que distintos agentes de una capa accedan a los de otra. Estos se encuentran organizados en unidades funcionales que dan el acceso a un tipo determinado de recurso de información. Estas unidades funcionales están gestionadas por un responsable de la operación (*functional manager*) que debe realizar la asignación de tareas entre los agentes de una unidad y que posee una visión en conjunto de su unidad funcional. Esto le permite considerar en la coordinación de tareas aspectos como límites en los recursos o capacidades especiales de los agentes a su cargo.

La interfaz de usuario se hace vía un navegador de Web y formularios en HTML. El usuario puede comprobar el estado de su petición y tiene acceso a una representación de ésta, pudiendo modificarla si lo desea. Los agentes de bajo nivel realizan la abstracción del modelo a la WWW y son los encargados de la búsqueda recursiva de información HTML (con conocimiento de palabras clave) y el acceso a motores de búsqueda remotos (de los que conocen el medio de acceso). Estos agentes de bajo nivel son llamados por el resto de los agentes anteriores.

Figura 1.3.1.2.a. **Diagrama de MACRON**



Los agentes se comunican entre sí vía KQML y hacen uso de un lenguaje denominado TAEMS (T*ask Analysis, Environment Modeling and Simulation*), diseñado por el mismo grupo de investigación. Cada agente está diseñado según una arquitectura común llamada DECAF (*Distributed Environment Centered Agent Framework*) que divide al agente en un módulo de toma de decisiones, otro de coordinación en tiempo real, otro de monitorización, y otro de planificación; todos comparten una información que es una estructura de tareas y que engloba el estado actual del trabajo del agente y su relación con el trabajo de otros agentes. Cada uno de los módulos modifica esta estructura para llevar a cabo los objetivos (creando subobjetivos, dependencias, etc. según la necesidad).

Este modelo plantea el problema de la búsqueda de información como la resolución de un problema distribuido (cada agente tiene un conocimiento parcial de las fuentes de información) que obliga a la coordinación y comunicación de agentes. Además, al existir un supervisor, esta arquitectura tiene la ventaja, frente a otras, según sus autores, de ofrecer conocimiento de las interrelaciones entre agentes y de tener conciencia de las limitaciones de recursos (y actuar acorde a ello).

1.3.1.3 Multiagente Planning Architecture (MPA)

Esta arquitectura general ha sido desarrollada por el centro de Inteligencia Artificial de SRI Internacional. Su objetivo es el de gestionar un amplio conjunto de componentes dispersos geográficamente, coordinando su interacción, en entornos militares.

MPA define un conjunto de agentes planificadores (AP's) que dan una serie de servicios en respuesta a un conjunto de peticiones. Estos agentes son capaces de realizar progreso incremental, en planes totales o parciales, y de responder a nuevas condiciones, sugerencias y limitaciones. Sus actividades se coordinan por meta-APs, con conocimiento especializado sobre estrategias para repartición de tareas y resolución de conflictos. Cada meta-AP controla su colección de APs en conjuntos denominados ``Células de Planificación".

MPA da un conjunto de *wrappers*' y de librerías de agentes (en C y Lisp) para permitir la construcción de agentes sobre sistemas antiguos. Todos los APs de una misma clase son idénticos salvo por los servicios ofrecidos. El protocolo de comunicación se construye sobre KQML. Los meta-APs también se comunican entre si para cumplir sus objetivos, teniendo cada uno una visión parcial del problema.

Su arquitectura hace uso de un variado número de herramientas, diseñadas originalmente para la NASA, como: un generador de agentes reactivos, un editor de bases de datos de conocimiento genérico y sistemas de planificación.

1.3.1.4 RESTINA

Desarrollado por el Instituto de Robótica de la Universidad Carnegie Mellon, consiste en un sistema multiagente cuyos componentes cooperan para alcanzar unos determinados objetivos. En ella los agentes se dividen en tres tipos: agentes de interfaz, que se relacionan con los agentes de tareas y presentan a un usuario humano los resultados de la tarea que ha encargado; los agentes de tareas, especializados en llevar a cabo una tarea cualquiera; y los agentes de información, especializados en extraer información de una determinada fuente.

Cuando un usuario formula una petición a un agente de interfaz, éste decide a su vez qué tareas debe llevar a cabo. Entonces las propone a uno o más agentes de tareas, quienes a su vez las descomponen en subtareas y encargan a otros, o bien las llevan a cabo ellos mismos mediante llamadas a agentes de información. Los agentes de información recuperan entonces

la información pedida en su parcela específica, y la envían de nuevo a los agentes de tareas, que se ocuparán de enviarla a su vez al de interfaz.

Todos estos tipos de agente son inteligentes, en el sentido de que planifican sus acciones para obtener un mayor rendimiento. Para ello, disponen de información específica para cada clase de agente: el agente de interfaz posee un modelo de usuario (que se compone de las creencias del agente sobre el usuario), el agente de tareas planifica una tarea dada y la cumple paso a paso y el agente de información conoce el método de acceder a la información más relevante de un documento, quizá mediante el uso de resúmenes o extracción de palabras clave. Todos son reactivos, en cuanto que pueden vigilar el estado de otros entes (agentes, fuentes de información, usuarios) y actuar cuando se cumpla una condición determinada. Sin embargo, sólo los agentes de tareas son realmente cooperativos y se comunican con sus pares.

La inteligencia del sistema recae, en su mayoría, en los agentes de tareas. Cada uno de ellos posee unos fragmentos de planificación independientes del dominio de aplicación, y otros dependientes del dominio. Mediante estos fragmentos, el agente es capaz de componer un curso de acción que resuelva la tarea; este curso de acción dependerá del estado de los entes exteriores al agente, y puede modificarse dinámicamente en el caso de que ocurran ciertos eventos. Si queremos distinguir en estos agentes creencias, deseos e intenciones, podemos relacionar los deseos con las tareas impuestas por los agentes de interfaz; las intenciones, con las subtareas que se encomendarán a agentes de tareas o de información; y las creencias, con los datos que se poseen del resto de agentes.

RESTINA es sin duda un modelo que será utilizado en el desarrollo de sistemas inteligentes destinados a la búsqueda de información, sin embargo queda mucho por hacer, en la creación de componentes que provean de sensibilidad a los agentes para interactuar con su medio y actúen de forma inteligente.

1.3.1.5 Retriever[25]

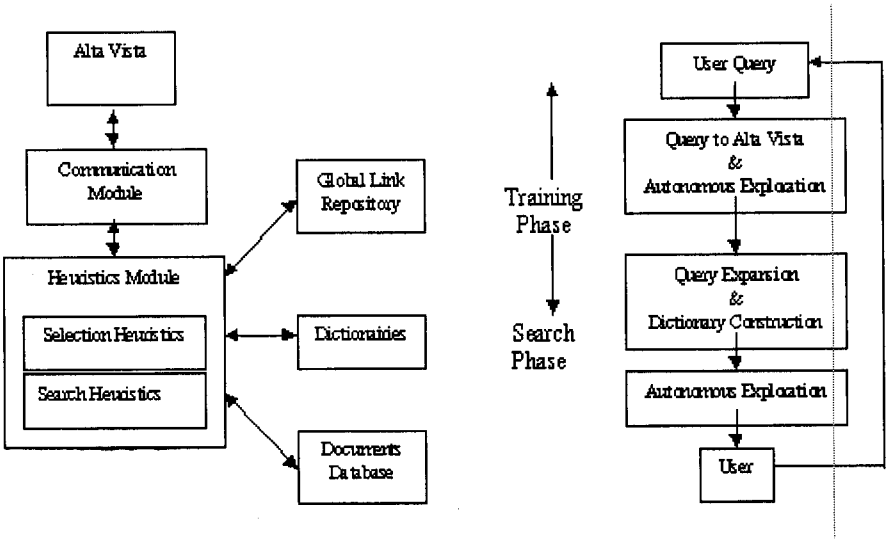
Retriever, es un desarrollo que se encuentra en etapa de prueba, sin embargo en esta sección presentamos solamente su arquitectura; para el siguiente capítulo lo abordaremos como una aplicación en funcionamiento. Para iniciar con la descripción de su diseño diremos que se compone de tres especificaciones: recuperación rápida de información, precisión, y la eficacia.

[25] http://portal.acm.org

El planteamiento inicia bajo el supuesto de cuándo un usuario necesita alguna información, generalmente no hay tiempo ni forma de entrenar un agente. Además, la mayoría de esta búsqueda agobiante no expresa un interés relevante. Cuando se reúne suficiente información, el proceso de búsqueda se termina y ninguna búsqueda adicional se realiza a menos que una nueva necesidad surja.

Todas las arquitecturas descritas en los agentes anteriores requieren un *interés persistente*[26] del usuario en uno o en más dominios. De esta perspectiva, Retriever es la arquitectura novedosa de agentes que difiere radicalmente de todos los paradigmas previos. Retriever es un agente que trata de llevar al máximo su eficiencia en una sola búsqueda. No depende del usuario para valorar la relevancia de los documentos devueltos. La evaluación del documento es generalmente un consumir de tiempo y una tarea frustrante y presupone un período extendido de la búsqueda. La entrada del usuario es restringida a la pregunta de él mismo y a las operaciones en dos fases del agente como se presenta en la siguiente figura:

Figura 1.3.1.5.a. Arquitectura y diagrama de flujo de Retriever



La primera fase consiste en auto entrenamiento que ayudará a Retriever a conocer el dominio de la pregunta. *Dominio de la pregunta*, refiere a los documentos que son pertinentes a la misma en términos de alguna medida de similitud entre documentos. En esta fase, Retriever reúne documentos que son semejantes a la pregunta del usuario. Cuando un número suficiente

[26] Para un usuario experto, es claro que en el proceso de investigación es necesario realizar más de una búsqueda para depurar la información resultante. A manera de ejemplo: Si al disparar sólo un tiro se acierta justo en el centro del blanco, se tendría la evidencia para contar con un nivel de confianza suficiente como para poner una manzana sobre nuestra cabeza y esperar el siguiente disparo.

de documentos pertinentes se recuperan, este proceso se termina y los documentos recuperados se analizan. El análisis del documento se realiza para permitir que el agente aprenda el dominio de la pregunta y aumente su eficiencia en la fase de la búsqueda. Y el documento recuperado como *pertinente* es presentado al usuario.

1.3.2 Aplicación de Agentes a Sistemas de Búsqueda

Para finalizar el presente capítulo, presentaremos un conjunto de aplicaciones de vanguardia basados en agentes, que resultan muy interesantes debido a que son una realidad, se encuentran funcionando y en su mayoría al alcance de los usuarios de Internet. Por otra parte los modelos mostrados a continuación son un punto de arranque o prototipo a tomarse en cuenta para cualquier trabajo que pretenda desarrollar sistemas inteligentes como herramientas de búsqueda de Información.

1.3.2.1 Letizia (Lieberman 1995)

Letizia es un agente de interfaz que ayuda a su usuario en sus búsquedas por la WWW, investigando los enlaces de las páginas que éste visita y recomendando los que, a su juicio, pueden resultar más interesantes.

Este agente desarrolla un modelo del usuario de manera incremental, guiándose tanto por las acciones del usuario como por un sucinto resumen de la página, que sólo contiene, actualmente, una lista de palabras clave.

Las posibles acciones que Letizia juzgará son: examinar una página Web, escoger un enlace a seguir, volver a una página del historial, marcar una página o volver a una página marcada. Obviamente, el marcar una página resulta un indicador de gran interés por parte del usuario. Seguir un enlace en una página indica alto interés en la página e interés moderado en el enlace, pues no podemos decir a priori si la página a la que conduce resultará de utilidad para el usuario. Esta lista de reglas puede aumentarse con otros heurísticos y otras acciones por parte del usuario.

Letizia también mantiene una lista de palabras clave que denotan interés asociadas a los documentos que examina, de manera que puede estimar su valor intrínseco. Esta

lista puede aumentarse realizando un resumen cada página que el usuario selecciona; este resumen será tan detallado como se quiera, y podría obtenerse mediante un análisis más sofisticado que la simple extracción de palabras clave. Esta lista permite que el agente recomiende páginas que podrían tener gran interés para el usuario, aunque el tema al que pertenecen no tenga nada que ver con el de la búsqueda actual. Está implementado en Common Lisp de Macintosh, y no tiene conocimientos del lenguaje natural sólo de palabras clave.

1.3.2.2 Lira (Learning Information Retrieval Agents - Balabanovic y Shoham 1995)

Lira es un sistema que ayuda a mantener a sus usuarios informados proporcionando día a día una selección de páginas de WWW interesantes que el usuario debe evaluar. Con esta información el sistema se adapta e intenta ofrecer mejores páginas cada vez. Trabaja offline obteniendo documentos de interés del usuario.

El método utilizado para evaluar la importancia de la página es el TFIDF[27], que calcula lo interesante que es una palabra basado en lo inusual de ésta. Aunque existen esquemas más complejos que los autores implementan en su versión final.

Para el usuario se mantiene un vector que se modifica con cada evaluación, de forma que se utiliza para ver si una página es o no interesante para el usuario. Imaginando un espacio de una cierta dimensión, sería simplemente comprobar si ambos vectores se encuentran próximos.

Con esta técnica, muy habitual en este tipo de agentes, se obtienen las palabras claves que se incorporan en el perfil del usuario. Al sistema se accede vía WWW a través de formularios, de forma que, según los autores, facilitaría el acceso a Internet de ordenadores móviles (en redes sin cable) que podrían dedicarse a *buscar*, sin tener que *navegar*.

1.3.2.3 Amalthea (Moukas 1997)

Es una co-evolución de un modelo de filtrado y descubrimiento de información, obtenida a través de consultas realizadas a motores de búsqueda existentes en Internet. Se alimenta de la

[27] Acrónimo inglés de Term Frequency Inverse Document Frequency. Asignación de pesos a términos, es decir, El peso de un término i en el documento j, tomando en cuenta un conjunto de Documentos.

información derivada de la aplicabilidad del usuario y adapta su conducta a sus intereses cambiantes. Actuando como asistente en la navegación, sugiere páginas respecto al perfil y crea resúmenes sin intervención del usuario. Utiliza una sociedad de agentes dinámica (algoritmos genéticos) para "acertar" en lo que el usuario desea. Es un sistema multiagente, con dos tipos de agentes: filtros de información y descubridores de información. Utiliza TFIDF para medir los documentos, y ha sido desarrollado basándose en la biblioteca de acceso a Internet *libwww* (para la próxima versión beta prometen que estará implementado en el lenguaje Java).

1.3.2.4 WebWatcher (A learning apprentice for the world wide web - Armstrong. et al. 1995)

Diseñado para ayudar y proporcionar recomendaciones personalizadas al usuario, realizando primero una extensión de búsqueda en las ligas siguientes, su función principal es asistir a la navegación en Internet, mira por delante del usuario y sugiere páginas a las que acceder. Ha sido probado con métodos de aprendizaje diversos: Winnow, Wordstat, TFIDF, Random. Al servir como guía puede aprender la navegación de múltiples usuarios. Esta diseñado en C/perl.

1.3.2.5 Syskill & Webert (Identifying interesting web sites- Pazzani. et al. 1996)

Identifica servidores de WWW interesantes, el usuario debe calificar las páginas para "enseñar" al agente. Para "aprender" utiliza filtros Bayesianos, PEBLS, arboles de decisión, TF-IDF y redes neuronales. Se está pasando la implementación en applets Java.

1.3.2.6 WebMate (Chen y Sycara 1998)

Esta aplicación sale del entorno de los sistemas multiagentes al contar con sólo un agente que compila un conjunto de URLs controlados por preguntas sobre motores de búsqueda para documentos de interés. El agente puede reconocer un número fijo de dominios claros de interés, fungir como asistente en la navegación y ayuda a la generación automática de periódicos (con noticias de interés del usuario). Aprende los intereses del usuario utilizando TFIDF.

1.3.2.7 ARACHNID (Menczer y Belew 1998)

Es un sistema de multiagentes que busca autónomamente en Internet documentos interesantes. Las pruebas del sistema se adaptan al ambiente de la información y pueden operar sin alguna intervención humana. Un enfoque nuevo.

En el presente capítulo se mostraron estrategias y aplicaciones para la búsqueda de Información en Internet. Como hemos visto, cada una tiene cualidades y deficiencias que pueden complementarse y que nos brindan la oportunidad de aportar nuevas ideas a la difícil labor de buscar información de calidad.

La Web Distribuida, es sin duda, puede mostrar su funcionalidad solo para ámbitos locales o de propósito muy específico, y en ese sentido no está pensada para la Web actual. La propuesta esta enfocada a la creación de nuevas comunidades a través de sistemas información distribuidos.

La Web Semántica, se propone el marcaje de documentos para facilitar su clasificación mediante ontologías, sin embargo, se requiere de un gran esfuerzo para desarrollar mecanismos automáticos para la creación de estas últimas.

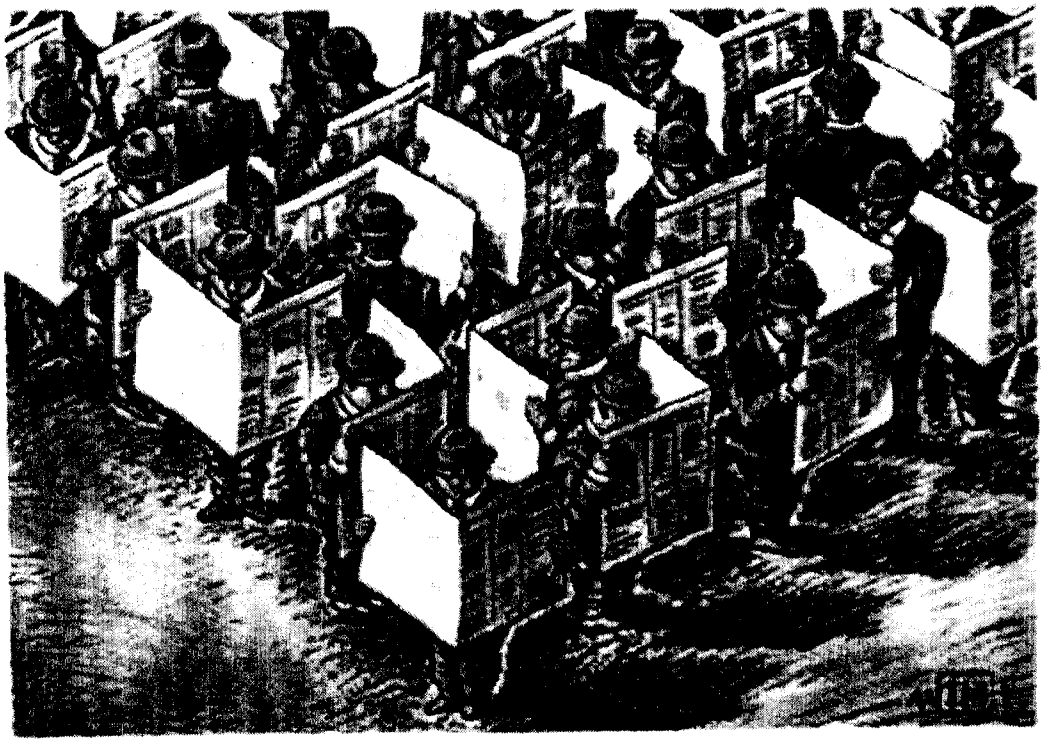
La Web Cooperativa, con base en la taxonomía del documento pretende desarrollar clasificaciones automáticas de los mismos, las cuales serán aprovechadas por una especie de asistente de navegación llamado agente de software, quien nos recomendará sitios o páginas de interés; respecto a estos agentes ya existen diversas aplicaciones, de las cuales se presentaron diversos ejemplos al final del capítulo.

En el capítulo siguiente, nos dedicaremos a mostrar las herramientas matemáticas que darán sustento a la creatividad de nuestra propuesta, es decir, a continuación se mostraran los algoritmos que podrán ser incluidos como elementos de sensibilidad para un agente de software o que lo apoyarán en la difícil tarea de clasificar un conjunto de documentos sin necesidad de la intervención humana.

Teoría de Recuperación de Información, Agentes y Herramientas de Clasificación.

"El problema de la clasificación es uno de los primeros que aparecen en la actividad científica y constituye un proceso consustancial con casi cualquier actividad humana".

Luis M. Molinero.



Capítulo 2

2.0 Introducción.

Como hemos visto en el capítulo anterior una de las tecnologías más prometedoras para aprovechar el creciente potencial de conectividad, velocidad y capacidad de almacenamiento de las redes mundiales es la de los Sistemas Multiagentes (MAS). Esta especialidad nace a principios de los años ochenta como una nueva área de la Inteligencia Artificial llamada Inteligencia Artificial Distribuida, (IAD).

El problema que se planteaba en ese momento era el de cómo hacer para que programas basados en el conocimiento pudiesen comunicarse entre sí a través de una red y negociar para resolver problemas complejos que ninguno de ellos por separado podrían resolver. Sin embargo el concepto de objeto activo (también llamado agente o actor) fue introducido por C.Hewitt (1977) para describir un conjunto de entidades que cooperan y se comunican mediante el envío de mensajes. Este concepto aporta las ventajes de la "programación orientada a objetos" a los entornos distribuidos y agrega a los lenguajes "orientados a objetos" y algunas características de los sistemas abiertos.

A partir de estas ideas iniciales se fue desarrollando un área de investigación muy vasta. A finales de los ochenta se tenía un cuerpo teórico importante, algunos prototipos pero ninguna aplicación real. La explosión de Internet de la primera mitad de la década de los noventa hizo que el campo de la IAD evolucionara rápidamente y que aparecieran las primeras aplicaciones. Nuestro modelo se desprende de la conjugación de varias de estas arquitecturas, sin embargo la tecnología multiagentes, es sólo una parte de la solución, debido a que el ámbito de nuestro problema es mucho más extenso, es decir, involucra objetivos y técnicas de distintas áreas que refieren al proceso global de descubrir información o conocimiento potencialmente útil y previamente desconocido a partir de datos de la Web, lo cual abordaremos mediante el concepto de Web Mining.

En otro orden de ideas, se expondrá también el fundamento teórico de los estadísticos de clasificación que han tenido una gran utilidad en el área biomédica, una de estas aplicaciones es la estimación de la tasa de aciertos de clasificación, la cual se propondrá como una herramienta de diagnóstico para agentes, es decir, un factor que les ayude a cumplir con el carácter adaptivo.

2.1 Recuperación de Información bajo el concepto de Web Mining

Web mining (minería del Web) puede definirse como el descubrimiento y análisis de información útil en la World Wide Web. Dentro de esta amplia definición existen tres áreas o enfoques: Web content mining, Web structure mining y Web usage mining en las cuales a su vez se conjuntan los objetivos y técnicas de distintas áreas, entre ellas la tecnología de Agentes.

2.1.1 Web Content Mining

Web content mining se refiere a la búsqueda automática de información y extracción de conocimiento a partir del contenido y de las descripciones de documentos en la Web. En este enfoque se prioriza La heterogeneidad y la falta de estructura de las fuentes de información, en forma de hipertextos, lo que hace complicado el descubrimiento automático, organización y manejo de la información, motivo por el que los investigadores han clasificado la extracción de información dependiendo del contenido de los documentos en la Web de la siguiente forma:

Text Mining: si los documentos son textuales (planos).

Hypertext Mining: si los documentos contienen enlaces a otros documentos o a sí mismos.

Markup Mining: si los documentos son semiestructurados (con marcas).

Multimedia Mining: para imágenes, audio, vídeo, etc.

2.1.2 Web Structure Mining

Web structure mining se refiere al proceso de inferir conocimiento a partir de la organización y las referencias o links entre documentos de la Web. Por ejemplo, muchos links que apuntan a un documento pueden indicar la popularidad de un documento, mientras links que salen de un documento pueden indicar la riqueza o variedad de temas que cubre el documento. De esta forma, se puede tomar ventaja para encontrar los documentos pertinentes y rastrear una estructura resumida.

Estas estructuras resumidas son comparables a palabras clave que se utilizan para referenciar citas bibliográficas. Existen métodos y software que aprovecha esta información, dando rangos a páginas y haciendo conteos de links para clasificar las páginas Web y facilitar su búsqueda.

2.1.3 Web Usage Mining

Web usage mining es un tipo de Web mining que se refiere al descubrimiento y análisis de patrones de acceso (o hábitos) de los usuarios desde uno o más sitios Web, mediante la extracción de patrones e información implícita en su actividad.

A medida que más empresas basan su negocio en Internet, las estrategias y técnicas tradicionales para el análisis del mercado deben ser vistas desde un nuevo contexto. Las organizaciones y compañías en Internet generan y almacenan grandes volúmenes de datos en sus funcionamientos diarios. La mayoría de esta información es generada automáticamente por los servidores Web y se almacenan en archivos llamados Log files de acceso al servidor. Otras fuentes de información del usuario incluyen la referencia a otros sitios o páginas de la Web y registros de usuario en bases de datos vía formularios en línea.

Analizar tales datos puede ayudar a las organizaciones a determinar qué usuarios visitan su sitio, permitiendo generar estrategias de mercadeo de productos y aumentar la efectividad de sus campañas promocionales, entre otras cosas. El análisis del acceso al servidor y los datos del registro del usuario también puede proporcionar información valiosa de cómo mejorar la estructura del sitio, creando una presencia más eficaz en la red.
Por ejemplo, Web usage mining puede ser usado para responder algunas preguntas, tales como:

- ¿Qué tipo de visitantes navega por el website?
- ¿Qué tipo de visitantes prefiere un determinado contenido?
- ¿Qué tipo de visitantes compran productos y servicios?
- ¿Cuáles son las características o atributos de los clientes online más atractivos?
- ¿Qué rasgos o características tienen en común los visitantes que son atractivos?

- ¿Quiénes son los clientes más atractivos y más fieles?
- ¿Qué hace que un visitante sea más o menos atractivo?
- ¿Qué contenido y estructura es la más aceptada por mis clientes?
- ¿Qué tipo de visitantes responden a la publicidad online?
- ¿Qué tipo de publicidad online produce mayor impacto sobre los visitantes?
- ¿Por qué este producto se vendió con este visitante?
- ¿Qué comprarán los visitantes del website la próxima semana?
- ¿A cuáles visitantes les gustaría comprar un nuevo producto o servicio?

Una vez que el concepto de Web Mining ha dejado atrás las barreras de novedad, ha sido bien acogido por las grandes empresas dedicadas al Internet. Ahora ya no basta con plantar una página Web y esperar sus frutos, se necesita abonarla con dosis de conocimiento y recolectar en bastas extensiones la competitividad.

Hemos presentado, un panorama general del ámbito del problema y el entorno donde se encuentran las técnicas que se usarán para nuestra propuesta de solución, asimismo, el grado de especificación con el que la extracción de información de Internet, es abordada por el concepto llamado Web Mining, Ahora que podemos ver desde las alturas el mapa estratégico, se iniciará una breve explicación acerca de las técnicas de interés para este trabajo, es decir, de las herramientas fundamentales para el diseño de SABIO.

2.1.4 Clasificación Automática de Textos (CAT)

Dentro de la minería de texto se establecen dos etapas principales: una etapa de preprocesamiento y otra de descubrimiento (Tan, 1999). En la primera, los textos se transforman a algún tipo de representación estructurada o semiestructurada que facilite su posterior análisis, mientras que en la segunda etapa las representaciones intermedias se analizan con el objetivo de descubrir en ellas algunos patrones interesantes o nuevos conocimientos.

Figura 2.1.4.a Estado del arte de la minería de texto

Etapa de pre-procesamiento	Tipo de representación	Tipo de descubrimientos
Categorización	Vector de temas	Nivel temático
Full-text	Secuencia de palabras	Patrones de lenguaje
Extracción de información	Tabla de datos	Relaciones entre entidades

Entonces, dependiendo del tipo de métodos aplicados en la etapa de preprocesamiento son el tipo de representaciones intermedias construidas, es decir, que de acuerdo a la calidad de nuestra clasificación, será la calidad en la recuperación de Información y generación de conocimiento.

De forma clásica la Categorización o Clasificación se define como la Asignación de objetos de un universo a dos o más clases o categorías predefinidas, para lo cual precisa los siguientes elementos:

A. Corpus de entrenamiento: Objetos ya clasificados de los que se pretende extraer regularidades (conocimiento).

B. Modelo de representación: Sistema que permite codificar los datos de entrenamiento. Por ejemplo: (x,c) donde x es un vector de medidas y c es la etiqueta asignada.

X. Modelo de clasificación: Un tipo parametrizado de clasificadores. Por ejemplo:

$g(x) = wx + w_0$ (si $g(x) > 0$ se elige C_1, sin no se elige C_2).

Δ. Entrenamiento: Proceso que estima los parámetros del modelo de clasificación (en el caso anterior w y w_0).

E. Conjunto de Prueba: Objetos ya clasificados pero no utilizados en el entrenamiento.

Ahora bien, pensemos en un clasificador binario, y una de sus categorías es "interesante" como en el caso de documentos Web, entonces podemos probar nuestro clasificador mediante una tabla de contingencias que recoge todos los casos posibles definiendo dos categorías A y B:

A_c Objetos clasificados en A correctamente

B_c Objetos clasificados en B correctamente

A_i Objetos clasificados en A incorrectamente

B_i Objetos clasificados en B incorrectamente

La medida obvia para la corrección es: $\frac{(A_c + B_c)}{(A_c + B_c + A_i + B_i)}$

Ahora los clasificamos en relevantes (A) e irrelevantes (B):

Certeza = $\frac{(A_c)}{(A_c + A_i)}$ Cobertura = $\frac{(A_c)}{(A_c + B_i)}$

En el caso de más de dos categorías, se construye una tabla de contingencias (frecuencias) para cada una de ellas (C_i frente a $\neg C_i$), a partir de ellas se pueden hacer dos cosas:

Macro-ponderar: Calcular la medida (p.e. certeza) para cada tabla y extraer la media. Da el mismo peso a cada categoría.
Micro-ponderar: Construir una tabla unificada, sumando las frecuencias, y luego calcular la medida. Da el mismo peso a cada objeto.

2.1.4.1 Corpus de Entrenamiento

Una de las propuestas más novedosas en este trabajo, se encuentra en el uso de las fuentes de información a las que acude el usuario de determinado equipo, es decir, el ordenador puede determinar las preferencias de su operador y de esta manera definir su perfil.

Una de las listas más importantes de información para un usuario, sin duda son las direcciones dentro de sus favoritos, subdirectorios personales o departamentales y de manera más específica un diccionario de términos para un ámbito científico.

2.1.4.2 Modelo de Representación

Sin embargo, aun cuando tengamos un conjunto de entrenamiento, en muchos casos no se conocen a priori las categorías de clasificación, por falta de una teoría sólida establecida para saber cuáles son las características relevantes que permiten discriminar entre diversas categorías.

Bajo esta perspectiva, el problema de clasificar conlleva a seleccionar un subconjunto de **m** características de entre un grupo original de **n** candidatas donde (**m < n**), bajo algún criterio de desempeño. El número de posibilidades crece exponencialmente, haciendo impráctica la búsqueda exhaustiva, aun para valores moderados de **n**.

Desde el punto de vista tradicional un texto se puede representar a través de un grupo de m sustantivos llamados términos índice, obteniéndose una vista lógica conceptual del documento, resulta evidente que al tomar n como el número de palabras totales dentro del documento, se genera mucho "ruido" para la tarea de selección del grupo de términos índice.

El problema de selección de términos índice es equivalente a buscar en un grafo dirigido, donde el nodo raíz corresponde al conjunto de todos los sustantivos del texto. El número total de posibles subconjuntos de un texto de n-sustantivos es $\mathbf{2^n}$. En el grafo, cada nodo corresponde a un subconjunto de términos índice y cada rama representa la inclusión del subconjunto. Los subconjuntos se codifican como tiras binarias, donde el entero 1 indica que un sustantivo está presente en un subconjunto y 0 indica que está ausente.

2.1.4.2.1 Método de Selección de Características.

Un método de selección de características típicamente requiere de los siguientes ingredientes:

- un criterio de evaluación J para comparar subconjuntos de características.
- un procedimiento de búsqueda.
- un criterio de detención, típicamente un umbral de significancia o la dimensión del espacio final de características.

Sea ***Y*** el conjunto original de características, de cardinalidad **n**. Sea **m** el número deseado de términos índice en el subconjunto seleccionado ***X***, Sea J(X) la función objetivo para el conjunto X. Sin pérdida de generalidad se supone que un mayor valor de J indica un mejor subconjunto de características. Un posible criterio es 1-pe, donde pe es la probabilidad de error. Formalmente, el problema de la selección de características consiste en encontrar un subconjunto

$X \subseteq Y$ tal que $|X| = m$, y $J(X) = \max_{Z \subseteq Y, |Z|=m} J(Z)$.

Cover y Van Campenhout[1] demostraron que ningún procedimiento secuencial de selección de características puede garantizar la obtención del subconjunto óptimo, salvo la búsqueda exhaustiva. Estos autores encontraron que cualquier ordenamiento de las probabilidades de error de cada uno de los $(2^n)!$ subconjuntos de características es posible, número que se reduce a 2^n en caso de cumplirse la condición de monotonicidad. Esto quiere decir que el mejor subconjunto de m características no necesariamente está compuesto por las m mejores características individuales.

En la siguiente gráfica 2.1.4.2.1.a se presenta una taxonomía de métodos de selección de características. La primera distinción es entre métodos basados en la teoría estadística de reconocimiento de patrones y aquellos basados en redes neuronales artificiales. Dentro de los primeros, a su vez, se diferencia entre métodos óptimos o subóptimos, con solución única o múltiple, determinístico o estocástico.

Se considera también óptimo el método de ramificación y acotamiento (branch and bound), que es exhaustivo bajo la condición de monotonicidad. Esta condición requiere que una función objetivo ***J*** usada para evaluar subconjuntos de características, crezca monótona sobre una secuencia anidada de subconjuntos de características:

$\{X_1,\ldots, X_k\}$, i.e., $X_1 \subset X_2 \subset \cdots \subset X_k \quad J(X_1) \leq J(X_2) \leq \cdots \leq J(X_k)$ donde

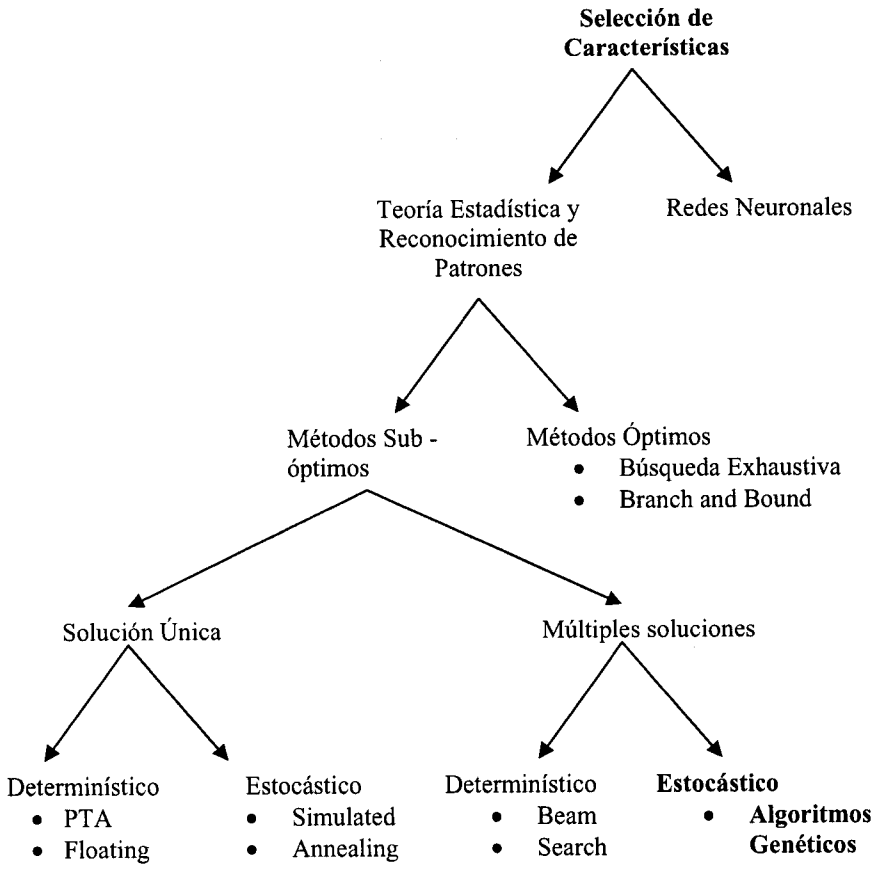
$X_k = \{ x_i,\ 1 \leq i \leq k,\ x_i \in Y\}$ es el conjunto de k características

del conjunto original $Y = \{y_i, 1 \leq i \leq n\}$.

---

[1] Cover, T. and Van Campenhout, J., On the Possible Orderings in the Measurement Selection Problem, *IEEE Trans. Systems, Man and Cybernetics*, Vol. SMC-7, pp. 657-661, Sept. 1977.

Figura 2.1.4.2.1.a **Taxonomía de métodos de selección de características**[2]



Algoritmos Genéticos:

Los algoritmos genéticos (AG's) parecen adecuados para resolver el problema de selección de características debido a su paralelismo inherente, su búsqueda guiada de las regiones más promisorias, su habilidad para encontrar y mantener múltiples óptimos, y además para optimizar criterios no derivables.

Estévez y Caballero,[3] proponen un algoritmo genético de nichos para la selección de características para clasificadores neuronales. Los métodos de nichos permiten la formación de subpoblaciones estables de tiras binarias diferentes dentro de un algoritmo genético, permitiendo encontrar y mantener múltiples óptimos locales. En el método de nichos

[2] Jain, A. and Zongker, D., Feature Selection: Evaluation, Application and Sample Performance, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 19, N° 2, pp. 153-158, February 1997.

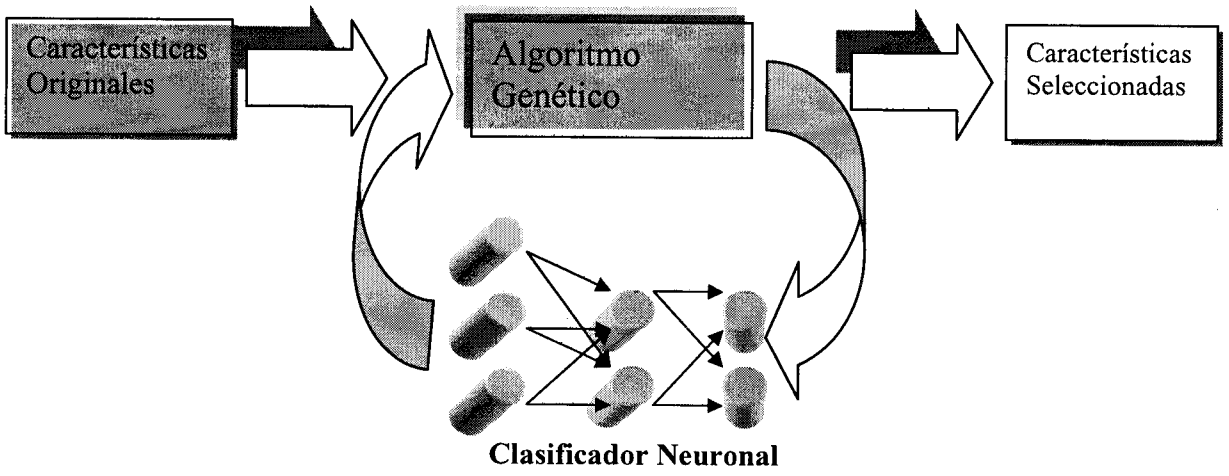
[3] Estévez, P., and Caballero, R., A Niching Genetic Algorithm for Selecting Features for Neural Network Classifiers, Perspectives in Neural Computation (ICANN-98), Vol. 1 , pp. 311-316, Springer-Verlag, 1998.

denominado hacinamiento determinístico (deterministic crowding), todos los elementos de la población son apareados aleatoriamente y recombinados. La descendencia resultante sostiene un torneo con su padre más cercano. Los ganadores se copian a la nueva población para la próxima generación.

Para el problema de selección de características, los individuos de la población se representan como tiras binarias, donde un "0" en la i-ésima posición indica que la i-ésima característica se excluye del subconjunto de características, y un "1" indica que la característica está presente[4].

En la gráfica 2.1.4.2.1.b se ilustra el enfoque híbrido utilizado: el algoritmo genético optimiza la selección de entradas mientras que para la evaluación de los individuos de la población se entrenan clasificadores neuronales.

Figura 2.1.4.2.1 b **Combinación de Algoritmos Genéticos y Redes Neuronales para la Selección de Características.**



Para evaluar la adaptación de un individuo, el subconjunto de características seleccionadas se usa como entrada a un clasificador neuronal de arquitectura fija. La función de adaptación

---

4 Siedlecki, W. and Sklansky, J., A Note on Genetic Algorithms for Large-Scale Feature Selection, *Pattern Recognition Letters*, Vol. 10, pp. 335-347, Nov. 1989.

combina dos criterios de optimización: minimizar la tasa de error del clasificador y minimizar el número de características. La adaptación de un individuo z se expresa como:

$$\textbf{\textit{adaptación(z) = exactitud(z) -}}\ \lambda\left[\frac{ncar\ \ (z)}{L}\right]$$

donde exactitud(z) es la tasa de clasificación por unidad del individuo z en el conjunto de validación, ncar(z) es el número de características seleccionadas y L es el número original de características disponibles. El parámetro λ controla el compromiso entre los dos criterios (típicamente λ =0.01). El cálculo del término exactitud(z) es la parte más costosa del algoritmo.

Para reducir el ruido en la evaluación de la adaptación cada individuo se evalúa un número fijo de veces (típicamente tres veces) y la máxima adaptación alcanzada se escoge como representativa de ese individuo. Por otra parte, para evitar cálculos repetitivos, los individuos sobrevivientes de una generación a otra heredan sus evaluaciones de adaptación. Este procedimiento ahorra tiempo computacional en las generaciones posteriores debido a la convergencia de la mayoría de la población.

Con el propósito de acelerar la convergencia del algoritmo genético, Estévez y Caballero nuevamente proponen un operador de mutación, el cual se basa en un método de eliminación de pesos denominado SSM[5] que elimina los pesos no-significativos de una red neuronal. Este método elimina los pesos pequeños relativos a su desviación estándar estimada. La idea central detrás de este operador de mutación es extender el método SSM a la eliminación de entradas (características) mediante la poda de todos los pesos asociados a tales entradas.

2.1.4.3 Modelo de Clasificación

En el inciso anterior se propone el uso de métodos híbridos para la tarea de seleccionar de manera automática el conjunto de características más representativo de una categoría, y como nos podemos imaginar, éste es un estupendo sistema de clasificación, sin embargo daremos inicio a la explicación de lo que son los métodos no supervisados de clasificación y ahondaremos un poco más en el uso de algoritmos genéticos para la clasificación de términos, puntualizando en una propuesta llamada Algoritmo de Kohonen.

[5] Cottrell, M., Girard, B., Girard, Y., Mangeas., M and Miller, C., Neural Modeling for Time Series: A Statistical Stepwise Method for Weight Elimination, *IEEE Tran. Neural Networks*, Vol. 6, N° 6, pp. 1355-1364, Nov. 1995.

2.1.4.3.1 Clasificación No Supervisada (CNS)

Dentro de la CNS podríamos distinguir dos tipos de clasificaciones, si nos atenemos a si se utilizan números o medidas para describir los objetos del conjunto de datos, o si se utilizan atributos simbólicos. La CNS en la que los objetos son descritos por atributos simbólicos y éstos son utilizados en la clasificación suele ser denominada clasificación conceptual.

Otra posible distinción dentro de la CNS se refiere a si los modelos que se construyen o los métodos que se utilizan para construirlos, están basados en la probabilidad o no. Las técnicas de CNS basadas en modelos probabilistas suponen que los objetos del conjunto de datos han sido obtenidos muestreando una distribución de probabilidad cuya función de densidad suele expresarse como suma de funciones de densidad de distribuciones conocidas. Estas distribuciones conocidas suelen ser habitualmente normales o uniformes. Este tipo de modelos se denominan modelos mixtos. Los algoritmos de clasificación utilizados en los modelos mixtos están basados en hallar los parámetros desconocidos de las funciones de densidad anteriormente citadas.

Centrándonos en la CNS no basada en modelos probabilísticos podemos dividir los tipos de clasificación existentes en función de diferentes parámetros o factores. De esta forma, podríamos distinguir dentro de la CNS entre clasificaciones en las que se obtienen particiones del conjunto de datos (por ejemplo, las clasificaciones particional o jerárquica), y clasificación en las que se obtienen recubrimientos, es decir, las clases no están en general separadas, sino que comparten objetos (por ejemplo, la clasificación piramidal). Otro tipo de clasificación que difícilmente puede introducirse dentro de estos dos grupos es la CNS fuzzy. En este tipo de clasificación no se produce directamente una partición del conjunto de datos sino que a cada dato se le asigna un conjunto de números, uno por cada clase. Dicho número indica el grado de pertenencia del objeto a la clase.

Para nuestro trabajo se utilizarán objetos que vienen descritos por números o por distancias entre ellos, el modelo que se propone no está basado en la probabilidad, y puede generar particiones del conjunto de datos.

En particular nos centraremos en los métodos más clásicos que producen particiones, esto es: la clasificación particional.

El objetivo de la CNS es: dado un conjunto de datos, descubrir la estructura de grupos subyacente al conjunto si es que esta existe. La CNS ha sido utilizada por investigadores de ramas diversas de la ciencia: biología, geología, arqueología, etc., y dentro de cada ámbito se han realizado aportaciones muy diferentes, lo que la convierte en una rama interdisciplinaria. Esta interdisciplinariedad ha provocado la creación de diferentes tipos de estructuras clasificatorias dentro de la CNS.

- Clasificación Particional
- Clasificación Jerárquica
- Clasificación Piramidal

En la Clasificación Particional, dado un conjunto de datos, trata de dividir dicho conjunto en grupos de manera que los datos que están en el mismo grupo sean lo más parecidos posible entre sí, a la vez que lo más diferentes respecto a datos de otros grupos.

Por su parte la clasificación jerárquica, crea, a partir de un conjunto de datos, una sucesión de grupos encajados, cuya estructura puede representarse por medio de un árbol.

La clasificación piramidal, crea una sucesión de recubrimientos encajados, de tal forma, que dicha estructura puede representarse gráficamente por medio de un grafo, que cumple unas propiedades especiales.

En particular los objetivos buscados en cada uno de los diferentes tipos de CNS fueron los siguientes:

1) En clasificación particional se pretendió utilizar los Algoritmos Genéticos, para llevar a cabo, no sólo la distribución de un conjunto de datos en grupos, de manera óptima, sino que al mismo tiempo se pretenda que el propio algoritmo hallase el número de grupos óptimo.

2) En el caso de la clasificación jerárquica el objetivo fue hallar la jerarquía indexada que mejor representaba la estructura del conjunto de datos. Esta búsqueda se llevó a cabo en el conjunto de las disimilaridades ultramétricas.

3) Al igual que en la clasificación jerárquica, en la clasificación piramidal se intentó hallar la estructura de pirámide indexada que mejor representaba al conjunto de datos.

2.1.4.3.1.1 Modelo vectorial

Antes de la aplicación cualquier método de clasificación no supervisada es necesario deducir dimensiones latentes, para la aplicación de un algoritmo genético como el de Kohonen a un ámbito de clasificación de documentos, es posible asignar pesos a las palabras que intentan describir como importante la representación de un documento. Los pesos pueden estar basados en un concepto probabilista como el modelo de distribución de palabra, usando una distribución Poisson por ejemplo; otro enfoque el cual será el que presentemos a continuación es el modelo vectorial.

Este modelo considera que cada documento es descrito por un conjunto de palabras clave representativas llamadas términos índice, estas palabras tienen una semántica que permite resaltar los principales temas de un documento.

Tomando un conjunto de términos índice para un documento, se tiene que no todos los términos son igualmente usados para describir su contenido. Existen términos que son más ambiguos que otros; decidir la importancia de un término para resumir el contenido de un documento como hemos visto no es un problema trivial. Una manera de resolverlo, es asignar un peso numérico a cada término índice de un documento.

Definición[6]: Sea t el número de términos índice en el sistema y k_i un término índice genérico. $K = \{k_1,\ldots, k_t\}$ es el conjunto de todos los términos índice. Un peso $w_{ij} > 0$ es asociado con cada término índice k_i de un documento d_j. Para un término índice, el cual no aparece en el texto del documento, $w_{ij} = 0$. al documento d_j se asocia un vector término índice $\vec{dj} = (w_{1j}, w_{2j},\ldots, w_{tj})$. Además, sea g_i una función que regresa el peso asociado con el término índice k_i en cualquier vector t-dimensional. Es decir, $g_i(\vec{dj}) = w_{ij}$.

En el modelo vectorial pesos numéricos para términos índice en consultas y en documentos. Los pesos de los términos son usados para calcular el grado de similitud entre los documentos almacenados en el sistema y la consulta.

Definición: Para el modelo vectorial. El peso w_{ij} asociado con el par (k_i , d_j) es positivo y no binario. Además, a los términos índice en la consulta se les asignan pesos. Sea $w_{i,q}$ el peso asociado al par $[k_{i,q}]$, donde $w_{i,q} \geq 0$. Entonces, el vector consulta $\vec{q}$ es definido como $\vec{q} = (w_{1q},$

[6] Judith Jaramillo López.-"Técnicas para la recuperación de información y búsqueda de texto en bases de datos relacionales"- Tesis de posgrado en Ciencias e Ingeniería- página 32. UNAM Junio 2001.

$w_{2q,...,}$ w_{tq}) donde *t* es el número total de términos índice en el sistema. De tal modo, que el vector para un documento d_j es representado por $\vec{dj} = (w_{1j}, w_{2j,...,} w_{tj})$.

Dentro del modelo vectorial se pueden emplear diferentes herramientas de evaluación de similitud del documento d_j con respecto a la consulta *q* como la correlación entre los vectores $\vec{dj}$ y $\vec{q}$. A continuación en la Gráfico 2.1.4.3.2.a se presentan las medidas de similitud más empleadas.

Gráfico 2.1.4.3.2 a Medidas de similitud aplicadas a los modelos booleano y vectorial.

Medida de similitud	Producto escalar	Coeficiente de Dice	Coseno	Coeficiente de Jacard
Modelo booleano	$\lvert X \cap Y \rvert$	$2 * \frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$	$\frac{\lvert X \cap Y \rvert}{\sqrt{\lvert X \rvert} \cdot \sqrt{\lvert Y \rvert}}$	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert - \lvert X \cap Y \rvert}$
Modelo vectorial	$\sum_{i=1}^{t} x_i \cdot y_i$	$\frac{2 \cdot \sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2}$	$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{t} x_i^2 \cdot \sum_{i=1}^{t} y_i^2}}$	$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2 - \sum_{i=1}^{t} x_i \cdot y_i}$

La variable x_i representa el peso asociado con el par (k_i , d_j) es decir w_{ij} y los pesos y_i representan los valores del vector de consulta, w_{iq} se propone este cambio de variable a fin de simplificar el manejo de subíndices.

Dado que $w_{ij} \geq 0$ y $w_{iq} \geq 0$, tendremos que $0 \leq sim(dj , q) \leq 1$, teniendo así, que este modelo permite determinar un rango a los documentos de acierto a su grado de similitud con la consulta; por ello, un documento puede tener un emparejamiento parcial con la consulta.

Ahora para determinar los pesos w_{ij} uno de los procedimientos que mejores resultados ofrece es el modelo frecuentista, denominado TFIDF, que definimos a continuación:

Definimos ***N*** como el número total de documentos en el sistema y sea *D* el número de documentos donde se encuentra el término índice k_i. Sea TF el número de veces que el término k_i se menciona en el texto del documento dj, entonces la frecuencia normalizada f_{ij} del término k_i en el documento dj esta dada por:

$$fij = \frac{TF}{TF\_Máxima}$$

Donde el máximo se calcula sobre todos los términos que se mencionan en el texto del documento d_j. si el término k_i no aparece en el documento d_j entonces $f_{ij} = 0$, Además, sea IDF la frecuencia inversa del documento para k_i, está dada por :

$$IDF = \log\left(\frac{N}{D}\right)$$

De esta manera se considera que el mejor esquema usa pesos con:

$$W_{i,j} = f_{i,j} \log\left(\frac{N}{D}\right)$$

Sin embargo para el cálculo de TFIDF existen varias alternativas, la más utilizada se conoce como variante ATC que calcula el valor en tres pasos consecutivos:

En primer lugar se calcula el valor modificado de TF como:

$$Nueva\_TF = 0.5 + 0.5\frac{TF}{TF\_Máxima}$$

Mediante la introducción de TF_Máxima se pretende ajustar el cálculo para tener en cuenta la longitud de los documentos y obtener un resultado que varíe siempre entre 0.5 y 1.

El segundo paso consiste en el cálculo de TFIDF como:

$$TFIDF = Nueva\_TF * \log_2\left(\frac{N}{D}\right)$$

Finalmente el tercer paso consiste en normalizar el resultado anterior:

$$TFIDF\_final = \frac{TFIDF}{\sqrt{\sum_{i=1}^{t} TFIDF_i^2}}$$

Teóricamente, el modelo vectorial tiene la desventaja de que los términos índice se asumen mutuamente independientes. Pero en la práctica la consideración de los términos dependientes

puede ser una desventaja. En este modelo no sólo se usa un vector como medio para representar adecuadamente la significación de cada término de la colección en un documento determinado sino que, además, se considera que ese vector es un objeto geométrico real en el espacio n-dimensional.

Una vez que contamos con un conjunto de documentos ya clasificados, para integrar nuevos documentos a una de estas categorías o conglomerados, podemos calcular los centroides y compararlos con los vectores de los documentos de entrada para encontrar la medida de mayor similitud.

2.1.4.3.2 Algoritmo de Kohonen y la determinación genética de clases

Antes de profundizar con los conceptos de las redes neuronales no supervisadas donde se encuentra clasificado el algoritmo de Kohonen, se iniciará con un ejemplo a manera de introducción.

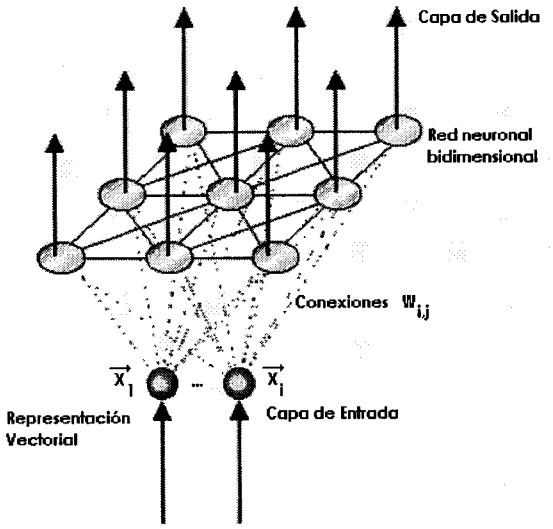


Figura 2.1.4.3.3 a Mapas Autoorganizativos de Kohonen.

En esta ilustración se muestra un Mapa autoorganizativo o SOM[7] de dos dimensiones.

Las neuronas se representan en el mapa bidimensional, mientras que los vectores de datos (representados por X_i) están en la parte inferior. Cada neurona "apunta" a los vectores, de manera que cada neurona tiene tantas coordenadas como rasgos hay en un vector.

Supongamos un conjunto de documentos que se refieren a la aplicación de tres tipos de medicamento, se encuentra que t =13 es decir, que el número de términos índice o coordenadas del vector representativo de cada documento, se caracteriza por 13 palabras clave.

[7] Por sus siglas en inglés SELF ORGANISING MAPS

Ahora suponemos que el mapa tiene una estructura de 4x4, entonces la representación tabular de la primer (1,1) y última (4,4) neurona es la siguiente:

Tablas 2.1.4.3.3 a Representación Tabular de las neuronas (1,1) y (4,4).

i	j	Coordenada	Peso
1	1	1	0.5551268677
1	1	2	0.4912584787
1	1	3	0.5725240301
1	1	4	0.5376965175
1	1	5	0.2996519389
1	1	6	0.2920934172
1	1	7	0.1398930513
1	1	8	0.6301825994
1	1	9	0.2781213684
1	1	10	0.5004732059
1	1	11	0.2149810183
1	1	12	0.2226109995
1	1	13	0.2559089992

i	j	Coordenada	Peso
4	4	1	0.6761664947
4	4	2	0.2570908548
4	4	3	0.5804567616
4	4	4	0.3563307997
4	4	5	0.4030377749
4	4	6	0.6451273643
4	4	7	0.5478053840
4	4	8	0.3396960966
4	4	9	0.4824003531
4	4	10	0.3470270500
4	4	11	0.4842337340
4	4	12	0.6818893122
4	4	13	0.5425087147

Nuestro problema consiste en encontrar los valores de las coordenadas de cada una de las neuronas del mapa de manera que, en el espacio bidimensional de las neuronas, las neuronas vecinas apunten a aquellos datos que forman parte del mismo grupo, como se muestra en la siguiente figura:

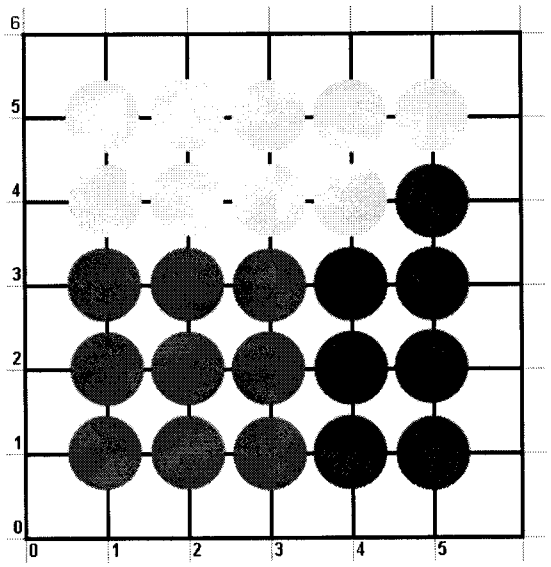


Figura 2.1.4.3.3 b Red Neuronal bidimensional de Kohonen.

En el mapa, cada color o tono de gris, identifica uno de los tres temas; y cada círculo corresponde a una neurona.
Como puede verse, las neuronas correspondientes a cada tipo de tema son vecinas entre si.
Así, lo que pedimos de este tipo de redes, es un mapa del espacio multidimensional (en el ejemplo, de 13 dimensiones) al espacio bi-dimensional de tal manera que haya una cercanía geográfica entre las neuronas que mapean a miembros del mismo grupo.

Ahora sólo nos queda una pregunta, ¿Cómo se consiguen estos resultados?; Teuvo Kohonen intentando conocer y emular la clasificación que hace el cerebro humano, diseñó el algoritmo que presentamos a continuación:

El algoritmo de Kohonen ha sido utilizado para la clasificación topológica como para análisis de conglomerados. Su simulación conlleva cierta complejidad puesto que es necesaria la creación de una capa competitiva donde cada neurona ejerce cierta influencia sobre sí misma y el resto de las neuronas que se hallen a una determinada distancia.

En el algoritmo asociado a este modelo podemos distinguir dos etapas:

1. Etapa de Funcionamiento
2. Etapa de entrenamiento o aprendizaje

Al inicio del algoritmo, la vecindad es muy amplia e incluye una región extensa del mapa. conforme se van realizando las iteraciones este tamaño inicial se va reduciendo hasta llegar a una configuración en la que tan sólo se modifica la neurona específica o ganadora. En el proceso general de aprendizaje se suelen distinguir dos fases:

1. Ordenación global o fase de ordenamiento.
2. Ajuste fino o fase de convergencia.

En la primera se pretende organizar los vectores de pesos en el mapa, se produce el despliegue del mapa. Para ello se comienza con una tasa de aprendizaje y vecindad grandes, para luego ir reduciéndolo a medida que se vaya realizando el aprendizaje.

En la segunda fase las neuronas se van especializando. El proceso suele ser similar al anterior aunque más lento.

No existe un algoritmo que podamos considerar totalmente estándar para los SOM, sin embargo esto no constituye un problema. A continuación los pasos de un posible algoritmo de Kohonen:

1. En primer lugar se inicializan los pesos sinápticos[8] a los que denotaremos por $w_{ijk}(t)$. Para el instante inicial (t = 0) se puede optar por distintas configuraciones: pesos nulos, inicialización con un valor predeterminado (por ejemplo todos con el mismo valor), o bien se les puede inicializar con pequeños valores aleatorios que suele ser la opción elegida. Sin embargo hay que tener en cuenta un aspecto importante de esta inicialización y es que si los valores que se dan son verdaderamente aleatorios puede ocurrir o bien que no tengamos la convergencia deseada, o bien que los ciclos de entrenamiento se vuelvan excesivamente lentos.

2. En cada iteración se muestra un patrón x(t) tomado de acuerdo con una distribución del espacio sensorial de entrada p(*x*).

3. Ahora para cada neurona del mapa se calcula la similitud entre su vector de pesos sinápticos w_{ijk} el vector actual de entradas *x*. Existen diferentes medidas de similitud que explicaremos más adelante.

4. Una vez obtenida esta similitud se debe obtener la neurona ganadora $g \equiv (g1, g2)$ que será aquella cuyo grado de similitud sea mayor (es decir aquella cuyo peso sináptico sea más parecido a la entrada).

[8] Que pertenece o afecta a la zona de contacto entre dos neuronas

5. Una vez determinada la neurona ganadora actualizamos los pesos sinápticos no sólo de ella sino también los de sus neuronas vecinas. La regla mas utilizada para llevarlo a cabo es la siguiente:

$$w_{ijk}(t+1) = w_{ijk}(t) + \alpha(t).h(|i-g|,t).(x_k(t) - w_{ijk}(t))$$

donde: $\alpha(t)$ es el ritmo de aprendizaje , y $h(.)$ es la función de vecindad. Esta función depende de la distancia entre la neurona y la ganadora g. Valdrá cero cuando no pertenezca a la vecindad de la neurona ganadora (con lo que sus pesos no serán actualizados) y tendrá un valor positivo cuando si pertenezca. Tanto α como el radio de la vecindad disminuyen monótonamente con t.

6. Si se ha llegado al número máximo de iteraciones el proceso de aprendizaje finaliza, si no es así se vuelve de nuevo al paso 2 y se repite el proceso.

A continuación se puede realizar la segunda fase de la etapa de entrenamiento: el ajuste fino del mapa. De esta manera se consigue que la distribución de los pesos sinápticos se ajusten más a las entradas. El proceso es similar al explicado anteriormente pero esta vez tomando un valor para $\alpha(t)$ constante y pequeño y un radio de vecindad también constante e igual a uno.

El número de iteraciones que se deben realizar debe cumplir una serie de requisitos: debe ser proporcional al número de neuronas del mapa, de tal manera que a más neuronas serían necesarias más iteraciones, y debe ser independiente del número de componentes de x.

Otra de las consideraciones a tener en cuenta es que el criterio de similitud y la regla de aprendizaje que se empleen deben ser compatibles, ya que si no fuera así estaríamos empleando métricas diferentes lo que nos causaría conflictos en el desarrollo del mapa.

Para comprender un poco mejor las operaciones que realizan los SOM vamos a realizar una pequeña interpretación del algoritmo:

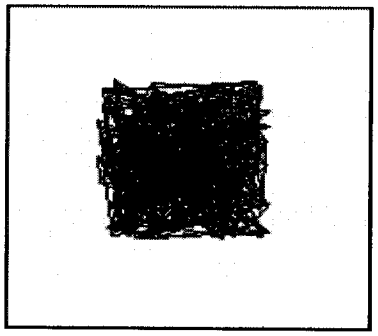
Como se ha indicado en numerosas ocasiones el objetivo de la regla de aprendizaje no es otro sino acercar en cada iteración un poco más el vector de pesos de la neurona ganadora al vector de entrada. Esto se puede representar mediante la formula: $\Delta w(t) = \alpha(x - w)$. Esta cantidad va a depender del ritmo de aprendizaje. Rescribiendo la formula anterior sin más que desarrollándola obtenemos:

$$\Delta w(t) = \alpha(x - w) = \alpha x - \alpha w$$

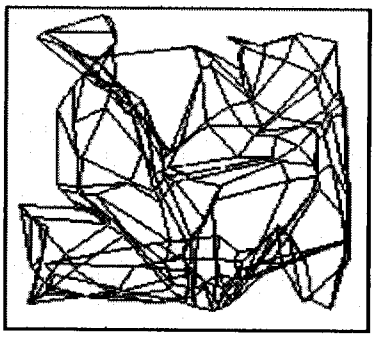
como se puede observar lo que hace es eliminar en cada iteración una pequeña fracción del vector de pesos original, así conseguimos que nuestra neurona se parezca al vector de entradas que la está haciendo ganar.

A continuación se muestra como se produce la evolución en una red de Kohonen:

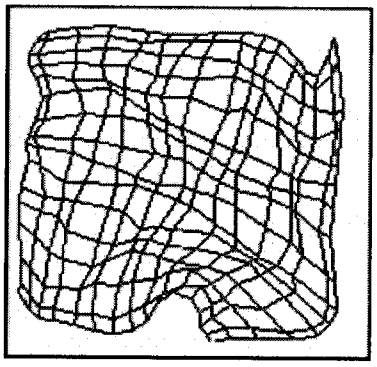
Figura 2.1.4.3.3 c **Evolución de Resultados de red de Kohonen.**



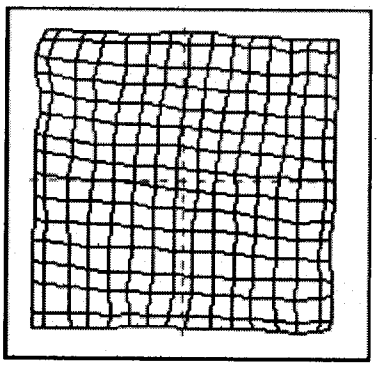
Inicio de la red



Situación trás 100 iteraciones



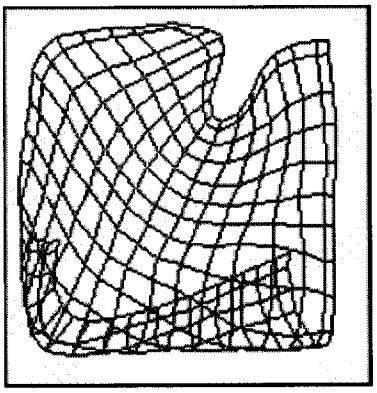
Situación tras 150 iteraciones



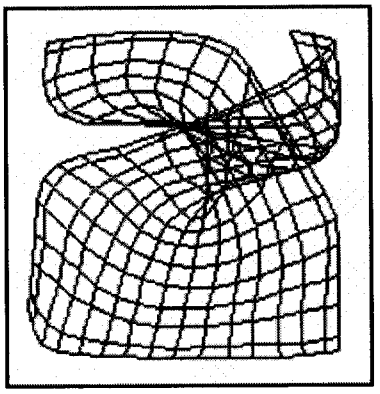
Situación trás 250 iteraciones

Debemos tomar en cuenta que no siempre se obtienen resultados que puedan ser interpretados de manera inmediata.

Figura 2.1.4.3.3 d **Resultados de difícil interpretación de red de Kohonen.**



Situación final trás 250 iteraciones



Situación final trás 250 iteraciones

Ahora puntualizaremos algunas propiedades e inconvenientes del algoritmo:

• Es capaz de realizar el agrupamiento de términos cercanos y también realiza una ordenación topológica de los grupos creados

• Asimismo podemos comprobar que es un algoritmo robusto frente a variaciones en los valores iniciales de los pesos.

• No necesita pares de entrada/salida tan sólo patrones de entrada.

• Se autorganiza de manera autónoma para adaptarse lo mejor posible a los datos utilizados durante el entrenamiento.

• Sin embargo el conjunto de neuronas que constituyen la red tienen una topología fija y rígida de vecindad entre ellas que no se puede modificar durante la etapa de aprendizaje lo que puede ocasionar problemas; por ejemplo cuando el conjunto de patrones de entrada se encuentran concentrados en lugar determinado, y puede ocurrir que los vectores sinápticos de algunas neuronas queden en zonas donde no haya ningún patrón de entrada siendo atraídos por otros patrones que se encuentren situados en zonas periféricas de mayor concentración. Este tipo de problemática se ha intentado solucionar introduciendo pequeñas variantes al algoritmo tradicional. [9]

• Otro de los problemas no triviales es que no hay manera de medir hasta qué momento es buena una neurona ganadora.

[9] http://www.infor.uva.es/~calonso/TemasAlumnosIAII/Redes%20de%20Kohonen.pdf

• El número de neuronas se decide de manera experimental. Si aumentamos su número aumenta su aproximación al espacio de salidas pero por el contrario se incrementa también el tiempo de entrenamiento.

• Para aprender nuevos datos es necesario repetir de nuevo todo el proceso de entrenamiento con todos los patrones y resulta poco aplicable a conjuntos de gran tamaño.

Finalmente, el algoritmo de Kohonen es uno de los más útiles en computación neuronal a pesar de sus limitaciones en cuanto a la duración del aprendizaje y a la imposibilidad de aprender nuevos datos sin tener que volver a repetir todo el proceso. Para nuestro campo de aplicación podemos enfatizar que debido a la manera de conglomerar los datos de entrada, el sistema se puede encargar de analizarlos y buscar el centroide[10] de cada categoría, además cuando se tiene la necesidad de discretizar un espacio continuo se puede transformar un conjunto de vectores n-dimensionales en vectores discretizados, quedando implícito la reducción de dimensionalidad y la cualidad de extraer características comunes a los datos de entrada.

Hasta aquí llegaremos con el estudio de la teoría de clasificación, la cual encuentra su campo de aplicación hecho a la medida en la programación orientada a agentes, como veremos a continuación.

[10] Se hace referencia al vector representativo y ubicado en el centro de un conglomerado de vectores pertenecientes a una categoría

2.2 Agentes como Robots de búsqueda

Ahora que hemos concluido con la exposición de los algoritmos de clasificación aplicados a la recuperación de información, es un buen momento para establecer las tendencias computacionales auxiliares a las teorías recién expuestas. El relativamente nuevo concepto de programación orientada a Agentes, en el cual y adelantándonos un poco a su definición, se establece una diferencia básica con respecto a la programación orientada a Objetos, al agregar autonomía.

La solución presentada por la IAD para la búsqueda de información, pretende la creación de una sociedad de Agentes los cuales se definen a continuación con las siguientes características[11]:

Autónomo.

Debe tener iniciativa y ejercitar sus propias acciones de manera no trivial.

- Orientado a objetivos: debe aceptar peticiones de alto nivel por parte de un humano y saber cómo y dónde satisfacerlas.
- Colaborativo: debe tener la posibilidad de modificar peticiones, preguntar aclaraciones o rehusar satisfacer ciertas peticiones.
- Flexible: sus acciones no deben estar prefijadas, sino permitir una elección dinámica de las acciones que debe realizar y en qué secuencia, en respuesta al estado de su entorno exterior.
- Auto-iniciación: un agente debe sentir los cambios de su entorno y decidir cuándo actuar, al contrario que los programas normales que son ejecutados por el usuario.

Continuidad temporal.

Un agente es un proceso en continua ejecución, no un cálculo que toma una única entrada y produce un resultado antes de terminar.

[11]Taller sobre agentes en Internet de la Universidad Hebrea de Israel.

Carácter.

Un agente debe tener un estado personal y emocional bien definido y creíble.

Comunicativo.

Un agente debe permitir una comunicación compleja con otros agentes, incluyendo personas, para obtener información de ayuda necesaria para cumplir sus objetivos.

Adaptivo.

El agente debe autoconfigurarse según las preferencias de sus usuarios y su experiencia previa. El agente debe adaptarse también a los cambios de su entorno.

Móvil.

Un agente debe permitir ser transportado de una máquina a otra y a través de distintas arquitecturas y plataformas.

Estas características de los agentes pueden sugerir diferentes modelos de SMA, de mayor o menor complejidad. Finalmente podemos definir agente como un programa de software que se ejecuta de forma concurrente, que tiene una noción de estado y es capaz de comunicarse con otros agentes por pase de mensajes y, es algo que se considera como un resultado natural del desarrollo del paradigma de programación concurrente orientada a objetos. Esta noción de agente es la que también se usa en la llamada "ingeniería de software basada en agentes", cuyos productos, llamados "software agents" o "softbots", son agentes que interaccionan con un entorno de software mediante comandos que usan para obtener información o cambiar el estado del entorno.

2.2.1 Tipos de Agentes[12]

Los agentes pueden clasificarse de varias maneras, teniendo en cuenta algunas de las propiedades que poseen o bien haciendo hincapié en alguna en particular. De esta manera puede armarse un árbol taxonómico que abarque todas las combinaciones de propiedades y tareas que se quieran.

[12] NWANA, H. Software Agents: An Overview. Knowledge Engineering Review. Cambridge University Press. v.3, 1996.

Para nuestro propósito resulta más conveniente acercar la clasificación de los Agentes a las aplicaciones que de ellos hemos presentado de acuerdo a líneas de investigación y desarrollos, donde se prioriza la función u objetivo principal del agente.

- Agentes de interface
- Agentes colaborativos
- Agentes móviles
- Agentes de recuperación de información

2.2.1.1 Agentes de Interface

Un agente de interface es un software cuasi-inteligente que asiste a un usuario cuando interactúa con una o más aplicaciones. La motivación es que se les pueda delegar tareas aburridas y laboriosas. Son asistentes personales que reducen el trabajo por la sobrecarga de información, como por ejemplo el filtrado de los mensajes de correo electrónico o la recuperación de archivos de Internet.

Esta categoría de agentes apoyan y proveen asistencia a su usuario. El agente observa y monitorea las acciones que toma el usuario en la interface, aprende nuevos atajos, y sugiere mejores formas de hacer las tareas. La idea es que el agente pueda adaptarse a las preferencias y hábitos de sus usuarios. Enfatizan la autonomía y el aprendizaje para llevar a cabo tareas para sus dueños y trabajan en el mismo ambiente que estos. A su vez, de los agentes de interface pueden encontrarse subdivisiones debido a diferentes tareas para las cuales son construidos. Las más comunes son:

Asistentes: Trabajan realizando tareas típicas como el manejo de la agenda. Estos agentes ayudan al usuario a planificar las reuniones. Sus acciones incluyen negociar, aceptar o rechazar reuniones.

Filtros: Su tarea principal es la de analizar información según un conjunto de reglas dadas por el usuario. La aplicación típica es el filtrado de mensajes de correo electrónico.

Guías: Asisten a los usuarios en el uso de una aplicación. Estos agentes monitorean las acciones de los usuarios e intentan sugerir qué pasos realizar para alcanzar el objetivo.

2.2.1.2 Agentes Colaborativos

Los agentes colaborativos constituyen un sistema multiagentes, es decir existe más de un agente dedicado a satisfacer los requerimientos de sus usuarios. Para ello es necesario contar con esquemas de comunicación entre agentes que posibiliten la cooperación y el intercambio de conocimiento.

Además, deben poseer un alto grado de autonomía para interactuar con sus pares. La motivación detrás de la construcción de agentes colaborativos es que los sistemas construidos con unidades relativamente simples proveen mayor funcionalidad que un ente mayor, pudiendo extender la funcionalidad del sistema mas allá de las capacidades de uno de sus miembros.

Además, estas arquitecturas posibilitan contar con mayor confiabilidad (debido a la redundancia) y mayor velocidad (debido al paralelismo) en el sistema conjunto.

Las áreas de aplicación de este tipo de agentes incluyen:

- Resolución de problemas demasiado grandes.
- Interconexión de múltiples sistemas.
- Manejo de información proveniente de fuentes distribuidas.

2.2.1.3 Agentes Móviles

Los agentes móviles son procesos capaces de "viajar" por una red de computadoras, interactuando con hosts externos, recolectando información en nombre de su dueño y retornando a "casa" luego de completar las tareas establecidas. Los agentes forman un nivel de abstracción más para el usuario, detrás del cual se encuentran soluciones a cuestiones técnicas en algunos casos complicadas. Una de estas cuestiones es la distribución, es decir, como manejar recursos computacionales distribuidos. Con la idea de agentes móviles los recursos distribuidos no son completamente ocultados al usuario pero tampoco completamente expuestos. La noción de movilidad viene del objetivo de reducir el tráfico innecesario dentro de

una red, con lo que se pueden reducir los costos de comunicación. Además, al aportar una nueva forma de computación distribuida posibilita el mejor aprovechamiento de los recursos de la red y permite que los usuarios tengan acceso a una cantidad mayor de recursos. Por ejemplo, debido a que las sesiones en busca de un recurso determinado ciertas veces son largas, la idea de agentes móviles provee una solución. Un usuario delega la tarea de búsqueda de información a un agente, establece una comunicación con la red y "envía" al agente a cumplir con su misión. La próxima vez que el usuario se conecte, el agente "retorna" con los resultados obtenidos. Para soportar la movilidad, debe existir una infraestructura de transporte que mueva el código del agente de una ubicación a otra. Además, se debe contar con un entorno de ejecución de agentes, donde los agentes "viven", compuesto por todas las computadoras que los proveen. Finalmente, para construir sistemas con agentes móviles es necesario resolver algunas cuestiones fundamentales tales como:

- Transporte: ¿Cómo se mueven de lugar en lugar?
- Ejecución: ¿Cómo ejecutar el agente de forma remota?
- Autenticación: ¿Cómo saber si el agente es quien dice ser y a quién representa?
- Privacidad: ¿Cómo asegurar que el agente mantenga resguardado su estado interno?
- Seguridad: ¿Cómo protegerlo de virus? ¿Cómo prevenir que el agente entre a bucles infinitos o falle?

2.2.1.4 Agentes de recuperación de información

Los agentes de recuperación de información poseen métodos para permitir el rápido acceso y recuperación de información relevante. Tienen la tarea de administrar, manipular y juntar información de fuentes distribuidas. Pueden tener mecanismos de búsqueda y navegación flexibles y algoritmos de clasificación poderosos.

El objetivo general de los investigadores dedicados a la recuperación de información, es diseñar una técnica que permita describir los cientos de millones de documentos disponibles de manera precisa, creando un índice de alta calidad, con una forma eficaz y eficiente de acceder a éste (ya sea de manera manual o automática). Una de las soluciones posible se basa en los agentes de recuperación de información. Estos agentes pueden asistir a un usuario novato en la

formulación de consultas avanzadas, con base en sus necesidades de información. Además, permiten acceder e integrar fuentes heterogéneas y manejar diferentes tipos de información.

2.2.2 Lenguajes de Agentes

Un lenguaje de agentes es un sistema que permite programar sistemas computacionales, software o hardware, en términos de algunos conceptos desarrollados en teorías o modelos formales de agentes. Los predecesores de estos lenguajes son los lenguajes concurrentes orientados a objetos, tales como "Actor model"[13]. El primero que propuso un nuevo paradigma para la programación de agentes fue Shoham[14]. La idea principal es la de programar agentes directamente en términos de nociones mentalistas e intencionales. Shoham propone un paradigma de programación orientado a agente basado en tres componentes: un sistema lógico para definir estados mentales, un lenguaje de programación interpretado, y un proceso de "agentificación" para compilar programas de agente en sistemas ejecutables de bajo nivel.

A partir de los trabajos de investigadores como: Kellett (2001), Denti (1999), Fisher, (1994), Poggy (1994) y Weerasooriya (1995), Donde se inicia el uso del paradigma de orientación a objetos en un contexto lógico. Por ejemplo, el lenguaje Metatem[15] que está basado en lógica temporal, encapsulando un conjunto de reglas, se han multiplicado las propuestas de lenguajes dedicados al desarrollo de sistemas multiagentes. En la página siguiente se encuentra un acervo muy importante de lenguajes dedicados al desarrollo de agentes de software: http://www.agentbuilder.com/AgentTools/index.html, agrupados por productos comerciales, proyectos académicos y otros de interés.

2.2.2.1 Java como lenguaje de Agentes.

La idea de los agentes de software (**software agents**) fue materializada hace un tiempo en Java por IBM Japón con los **Aglets,** estos componentes extienden el modelo de movilidad instrumentado en los applets de Java.

---

[13] Hewitt, C. (1977): Viewing control structures as patterns of passing messages. Artificial Intelligence, 8(3):323-364.

[14] Shoham Y. (1990): Agent-oriented programming. Technical Report STAN-CS-1335-90, Computer Science Department, Stanford University.

[15] Fisher, M. (1994): A survey of Concurrent MetateM - The language and its applications. In Gabbay, D. M. and Ohlbach, H. J.

Las applets son programas de Java que se guardan en un servidor de Internet/intranet. Una vez transferidas a las diferentes plataformas cliente, se ejecutan en una máquina virtual (MV) de Java suministrada por el navegador instalado en el ordenador cliente. Esta distribución y ejecución se llevan a cabo bajo la supervisión de un administrador de seguridad. Un administrador de seguridad puede evitar que las applets realicen tareas no deseadas, como el formateo del disco duro o la apertura de conexiones a equipos "no certificados".

Cuando el navegador encuentra una página web con un applet, arranca la MV de Java y le suministra la información de la etiqueta <applet>. El cargador de clases interno de la MV de Java controla qué clases se necesitan para el applet. Como parte del proceso de carga de clases, los archivos de clase se ejecutan a través de un verificador que comprueba que son válidos y no código dañino. Una vez verificado, el applet se ejecuta. Este mecanismo de transferencia es el principal valor de las applets.

2.2.2.1.1 Aglets versus applets

Los aglets son entidades que al igual que los applets de Java, pueden movilizar su código a través de la red, pero con algunas cualidades adicionales, por ejemplo un applet es un programa que al ser detectado por el navegador (Browser) se ejecuta en la máquina local es decir actúa en un arquitectura cliente servidor, sin embargo un aglet puede viajar de servidor en servidor presentando sus credenciales y si el servidor así lo permite ejecutar su labor que una vez terminada, le permite llevar su estado de ejecución, por ejemplo imaginemos un contador numérico de unidades enteras, el cual se inicia en cero y en su primer escala llega a la cuenta de siete para su segunda visita siete será su inicio y de esta manera puede viajar secuencialmente a muchos destinos incluyendo eventualmente el regreso a su nodo de origen.
De forma similar que un applet se ejecuta en uno o múltiples hilos dentro de un contexto del servidor de aplicaciones Java, un aglet requiere de un servidor que le permita hospedarse en una máquina y ejecutarse en ella, el servidor de aglets permitirá establecer restricciones para las actividades de aglets desconocidos y recobrar el estado de los mismos de su servidor anterior. Finalmente mostraremos los eventos que un aglet puede experimentar en su "vida".

Created: Se crea un nuevo aglet, su estado es inicial y es ejecutado su hilo principal.

Cloned: Se crea una copia del aglet actual, asimismo se duplica su estado.

Dispatched: El aglet viaja a un servidor nuevo y su estado va con él.

Retracted: Un aglet es recibido desde un servidor remoto el cual lo devuelve también con su estado.

Deactivated: Un aglet es puesto en estado de espera y su estado es almacenado en alguna parte del disco.

Activated: Un aglet es vuelto a la actividad y su estado es recuperado desde disco.

Disposed of: El aglet muere y su estado se pierde.

Se debe tomar en cuenta que en cada una de las actividades del Aglet se involucra la persistencia de su estado.

Se Propone Java como lenguaje de desarrollo debido a su potencialidad de portabilidad, lo que hace factible ejecutar la aplicación en la mayoría de los sistemas operativos predominantes en el heterogéneo mundo de Internet. Además por compatibilidad con la plataforma de agentes Aglets Workbench.

Y se seleccionó aglets debido a que proporciona la API y el ambiente necesarios para la creación de agentes móviles y estáticos. Además de que trabaja con cualquier versión igual o superior a Java1.1.x. Su similitud con los applets hace que la curva de aprendizaje de la plataforma sea rápida para quienes han trabajado con applets, además cuenta con características de seguridad que son imprescindibles en Internet y con un sistema de mensajería con diferentes tipos de mensajes que hacen fácil la comunicación entre los agentes.

2.2.2.2 Propuesta multiparadigma para programación de agentes.

Ahora bien la programación de agentes involucra además de la portabilidad, el encapsulamiento de sus comportamientos y de su estado mental. Estas características propician el uso de lenguajes orientados a objetos como Java. Sin embargo, cuando estados mentales complejos deben ser administrados, la programación lógica ha mostrado ser una mejor alternativa para la implementación de actitudes mentales.

Una propuesta para utilizar cláusulas lógicas que representan conocimiento interno de objetos, inherentes a la programación de agentes y sistemas multiagentes, es JavaLog[16] un lenguaje

[16] Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial. No.13 (2001), pp. 94-99.

multiparadigma que permite construir agentes a partir de objetos Java capaces de manipular conocimiento en formato de cláusulas lógicas Prolog o extensiones de este lenguaje lógico.

La integración de programación orientada a objetos y programación lógica alcanzada en JavaLog ha sido basada en el concepto de módulo lógico. Un módulo lógico es la encapsulación de una secuencia de cláusulas lógicas. Estos módulos pueden ser ubicados tanto en métodos Java como en variables, para luego ser combinados de diversas maneras.

JavaLog es un lenguaje de programación que combina los paradigmas de orientación a objetos y lógicos a través de la utilización de Java y Prolog. Se ha desarrollado un intérprete Prolog en el lenguaje Java con el fin de posibilitar extensiones del mismo a través de sub-clasificación. Programar agentes con JavaLog es programar un agente como un objeto Java, el cual es instancia de una clase que representa ese tipo de agente. La funcionalidad del agente es implementada en métodos codificados básicamente en Java. Módulos lógicos compuestos por una secuencia de cláusulas lógicas Prolog; son también utilizados en la programación de los agentes. Conocimiento privado a un agente es ubicado en módulos lógicos referenciados por variables de instancia de los agentes. Conocimiento común a los agentes de una clase es ubicado en módulos lógicos que pueden localizarse en los propios métodos de la clase o referenciados por variables accesibles por todos los objetos de la clase.

Las clases que definan algún tipo de agente se asociarán a una clase denominada Brain que permite que cada instancia de esas clases genere una instancia de este Brain que representa una instancia del intérprete Prolog. En otras palabras, cada objeto agente tendrá asociado un objeto Brain que le permitirá manipular cláusulas lógicas.

La base de conocimiento de una instancia del intérprete Prolog de un agente cualquiera, estará inicialmente vacía. Los módulos lógicos definidos en variables o métodos de la clase no están ubicados en esta base de conocimiento. Los módulos lógicos referenciados por variables tienen que ser explícitamente agregados o retirados en la base de conocimiento. Los módulos lógicos localizados entre el código Java de métodos serán trasladados temporalmente a la base de conocimiento cuando estos métodos sean invocados y mientras estén siendo ejecutados.

2.2.3 Ventajas e Inconvenientes de los Agentes Móviles

Aun cuando en apartados anteriores hemos mencionado el uso y conveniencias de los agentes móviles, a manera de corolario puntualizaremos algunos comentarios referentes al desarrollo de sistemas involucrados en la creación de agentes móviles.

Actualmente el ancho de banda de la red es limitado debido a la gran cantidad de tráfico que circula a través de ella. Los agentes móviles pueden hacer un uso más efectivo del ancho de banda de la red porque el agente sólo se hace presente en el tráfico de red cuando está viajando de una máquina a otra, y especialmente porque todo el acceso y procesamiento de información lo hace en forma local en la máquina donde reside de acuerdo a su itinerario.

Los agentes pueden configurarse con diferentes perfiles para tomar decisiones de acuerdo a las necesidades del usuario y hacer tareas más específicas y personalizadas.

Al establecer itinerarios, los agentes pueden tener una ruta que les permite viajar en algún esquema particular de búsqueda y acceso de información, yendo a servidores en forma directa o a servidores más cercanos que permitan mayor rapidez en el acceso de la información y el manejo de los recursos.

Al enviar varios agentes a tantos lugares como queramos a hacer tareas específicas, permite aprovechar la potencialidad de las máquinas en paralelo, logrando eficientar el uso de los recursos, además se pueden configurar agentes que migren en forma balanceada de un lugar a otro para distribuir el acceso de información.

La desventaja de la tecnología de agentes reside básicamente en que aun cuando se está utilizando Java como tecnología de desarrollo para ambientes y plataformas de agentes, esto sólo les da la característica de portabilidad de la plataforma, mas no es suficiente para dar la característica de movilidad a los agentes pues estos dependen de la plataforma en que se desarrollen. Dado que la tecnología es nueva, no existen estándares al respecto que permitan garantizar la existencia en Internet de este tipo de plataformas. De modo que las aplicaciones de agentes están restringidas a las redes y sistemas donde se encuentra instalada la respectiva plataforma.

2.3 Estadísticos de Evaluación

Todo proceso de medición (proceso mediante el cual se cuantifica una magnitud) está amenazado por diversas fuentes de error, derivadas tanto de las limitaciones del instrumento de medida, como de la naturaleza de la magnitud a medir. Se distingue entre el error debido a la *precisión* limitada del instrumento que atenta a la reproducibilidad de la medición introduciendo un error aleatorio en la misma y el debido a la *validez*, también limitada, que introduce un error sistemático, denominado *sesgo*. De modo esquemático se puede decir que la validez depende exclusivamente del instrumento y tiene que ver con la cuestión de si el mismo mide lo que debe medir, mientras que la precisión depende tanto del instrumento como del proceso de medición y tiene que ver con cuánto se aproxima la medida al valor real de la magnitud. En ambos casos es siempre una cuestión de grado, pues aún el cerebro humano no puede ser un instrumento infinitamente preciso y válido, hay sólo instrumentos más precisos y/o válidos que otros. El modo habitual de controlar la validez de un instrumento de medida se le denomina calibración, y consiste en comparar las medidas obtenidas con él y sus patrones de referencia (cuanto más se parezcan estas medidas al patrón, más válido es el instrumento), mientras que la manera de controlar la precisión de un instrumento es comparar entre sí medidas repetidas de un mismo objeto y evaluar el grado de acuerdo entre ellas (cuanto más se parezcan estas medidas entre sí, más preciso es).

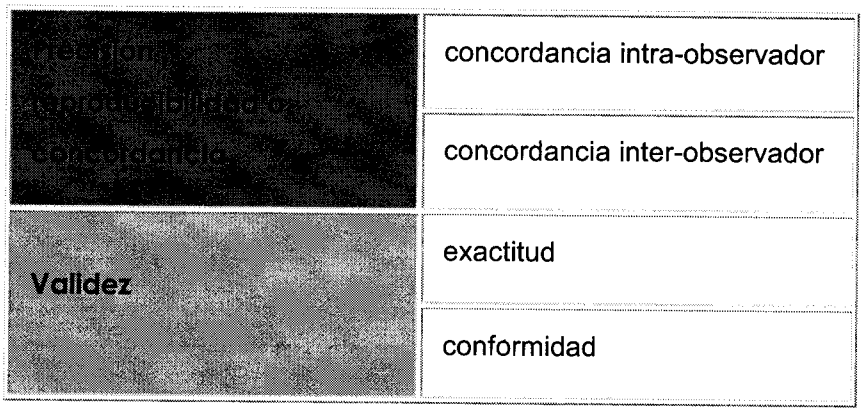
En ciertas situaciones, el control de la precisión y validez de una medida es más complejo que lo esbozado hasta aquí, debido a dos fenómenos que hasta ahora, no se han considerado. El primero es que las magnitudes a medir sean aleatorias, es decir presentan diversos grados de variabilidad propia. Por ejemplo en nuestro trabajo se propone el uso de algoritmos híbridos para la clasificación de documentos, los cuales tienen dentro de su estructura una fuerte carga de factores aleatorios. Aunado a los métodos de clasificación se involucra el criterio del observador o usuario.

Por otra parte, además de magnitudes tales como presión, temperatura, concentración, etc., se trabaja con magnitudes como preferencia, mejoría en un proceso, etc., para las cuales no existe un patrón de referencia claro y objetivo ni escala métrica apropiada y que, por tanto, suelen describirse en escalas ordinales o, incluso, nominales, cuya apreciación puede estar muy distorsionada por influencias subjetivas. Estas magnitudes suelen denominarse variables *blandas* y son las que dan lugar propiamente a clasificaciones en lugar de mediciones en sentido estricto (que implica la existencia de una escala métrica). Evidentemente, existen

también variables objetivas o *duras* que dan lugar a clasificaciones, por ejemplo muerto/vivo. Los procesos de clasificación sufren los mismos problemas de validez y precisión que los de medición, pero con ciertas complicaciones añadidas en el caso de las variables blandas. Para controlar su validez, no suelen existir patrones de referencia, o no son tan objetivos o accesibles como en el caso de una magnitud física. En este sentido se suele distinguir entre dos modos de controlar la validez de un instrumento de medida, cuando se hace con patrones objetivos se habla de *exactitud* (accuracy)[17], mientras que cuando se controla comparando simplemente con una referencia considerada mejor "gold standard" se habla de *conformidad*.

En cuanto a la reproducibilidad, sobre todo con métodos de clasificación, se distingue entre la reproducibilidad del mismo instrumento (típicamente un genético en este caso) en dos instantes de tiempo diferentes y se habla de concordancia (agreement) o consistencia interna o intra-observador, por ejemplo un usuario, ¿se clasifica igual el mismo documento encontrado hoy y dos meses después? y reproducibilidad del mismo usuario calificando en diferentes condiciones, por ejemplo dos métodos de clasificación diferentes ¿agrupan del mismo modo un mismo documento? se habla entonces de concordancia o consistencia externa o Inter-observador.

Figura 2.3.a Esquema de criterios de evaluación de instrumentos de clasificación



Existen varios índices de concordancia propuestos, el más obvio es la proporción de acuerdos observados, es decir $\frac{(a+d)}{N}$ [18]. Este índice es muy intuitivo y fácilmente interpretable: tomará valores entre 0 (total desacuerdo) y 1 (máximo acuerdo).

[17] En la literatura clínico-epidemiológica inglesa.
[18] Es la suma de acuerdos de los observadores entre el total de observaciones.

Figura 2.3.b Tabla de evaluación de concordancia

Análisis de Clasificación de Documentos			
	Algoritmo A		
Algoritmo B	Relevante	No Relevante	**Total**
Relevante	4 **a**	6 **b**	**r = a + b** 10
No Relevante	**c** 10	**d** 80	**s = c + d** 90
Total	**t = a + c** 14	**u = b + d** 86	**N = a + b + c + d** 100

Sin embargo como indicador de reproducibilidad tiene el inconveniente de que aún en el caso de que los dos observadores clasifiquen con criterios independientes se produciría un cierto grado de acuerdo por azar. Por ejemplo, si se tiran dos dados y se registra si sale un cierto número, digamos dos (resultado positivo) u otro cualquiera (resultado negativo), en un cierto número de veces, ambos dados producirán el mismo resultado por azar. Es deseable que un índice de concordancia tenga en cuenta este hecho y que, de algún modo, indique el grado de acuerdo que existe por encima del esperado por azar.

Ciñéndonos a este segundo aspecto, la manera concreta de abordar el problema depende estrechamente de la naturaleza de los datos: si estos son de tipo continuo es habitual la utilización de estimadores del coeficiente de correlación intraclase, mientras que cuando se trata de datos de tipo categórico el estadístico más empleado es el índice kappa, al que dedicamos el resto de este capitulo.

2.3.1 Índice Kappa

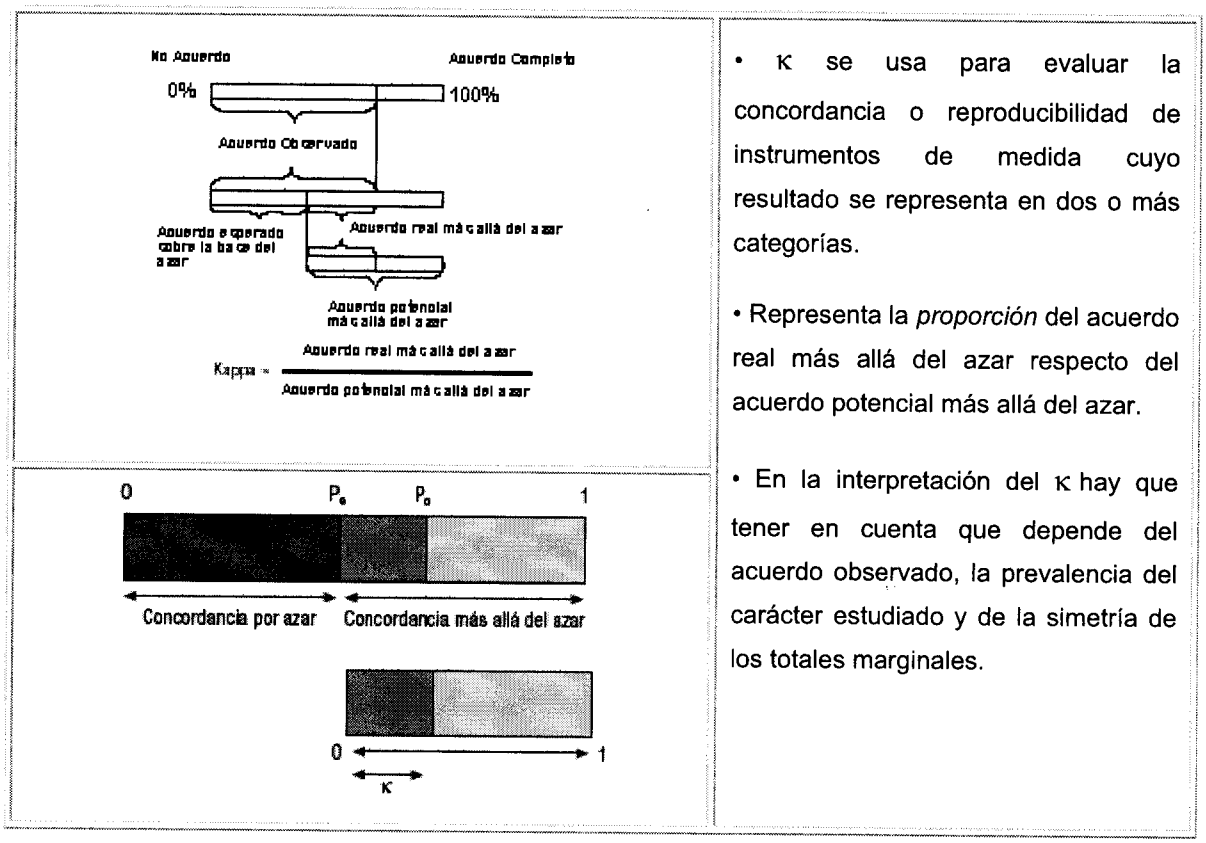
Para valorar el grado de acuerdo o desacuerdo entre dos observadores o la variabilidad de sus mediciones, hay distintos métodos derivados de la naturaleza de las variables consideradas, sin embargo, si se trata de una variable cualitativa o nominal, dicotómica, el método más sencillo y

utilizado es el Índice kappa o test de concordancia, propuesto por Cohen[19] en 1960. En este sentido el denominado índice kappa (κ), se define como:

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

Siendo P_0 la proporción de acuerdos observados y P_e la proporción de acuerdos esperados en la hipótesis de independencia entre los observadores, es decir, de acuerdos por azar. A partir de la figura 2.3.b. de Análisis de Clasificación de Documentos, $P_0 = (a+d)/N$ y $P_e = (rt+su)/N^2$. La interpretación de este índice se facilita mediante su representación gráfica. En la figura 2.3.1.c se observa que el índice κ representa la proporción de concordancia observada más allá del azar, respecto de la máxima concordancia posible más allá del azar.

Figura 2.3.1.c Representación gráfica del índice Kappa



- κ se usa para evaluar la concordancia o reproducibilidad de instrumentos de medida cuyo resultado se representa en dos o más categorías.
- Representa la *proporción* del acuerdo real más allá del azar respecto del acuerdo potencial más allá del azar.
- En la interpretación del κ hay que tener en cuenta que depende del acuerdo observado, la prevalencia del carácter estudiado y de la simetría de los totales marginales.

A fin de explicar de manera intuitiva el concepto en su aplicación más sencilla, obtendremos el índice Kappa siguiendo el ejemplo de la figura 2.3.b.

[19] Cohen J. (1960) A coefficient of agreement for nominal scales. *Educ Psychol Meas* **20**:37-46.

$$P_e = \frac{14 \times 10 + 86 \times 90}{100^2} = 0,788$$

y por lo tanto

$$\kappa = \frac{0,84 - 0,788}{1 - 0,788} = 0,245$$

Sin embargo, la interpretación del Kappa no es tan intuitiva como la proporción de acuerdos observados. Para el coeficiente de Kappa el intervalo se encuentra acotado desde -1 que significa perfecta discordancia y 1 acuerdo perfecto, por ahora dejaremos la escala de valoración del índice para retomarlo más adelante, cuando hayamos generalizado el concepto.

Supongamos que dos observadores distintos clasifican independientemente una muestra de n ítems en un mismo conjunto de C categorías nominales. El resultado de esta clasificación se resume en una tabla 2.3.1.b, en la que cada valor x_{ij} representa el número de ítems que han sido clasificados por el observador 1 en la categoría i y por el observador 2 en la categoría j.

Figura 2.3.1.b Tabla para el planteamiento del modelo Kappa general

Formato de los datos en un estudio de Concordancia					
	Observador 2				
Observador 1	1	2	...	C	**Total**
1	X_{11}	X_{12}	...	X_{1C}	$\mathbf{X_1}$
2	X_{21}	X_{22}	...	X_{2C}	$\mathbf{X_2}$
.	.			.	.
.	.			.	.
.	.			.	.
C	X_{C1}	X_{C2}	...	X_{CC}	$\mathbf{X_C}$
Total	$\mathbf{X_{.1}}$	$\mathbf{X_{.2}}$	**...**	$\mathbf{X_{.C}}$	**n**

Desde un punto de vista típicamente estadístico es más adecuado liberarnos de la muestra concreta (los n ítems que son clasificados por los dos observadores) y pensar en términos de la población de la que se supone que ha sido extraída dicha muestra. La consecuencia práctica de

este cambio de marco es que debemos modificar el esquema de la tabla anterior para sustituir los valores x_{ij} de cada celda por las probabilidades conjuntas, que denotaremos por π_{ij}.

Figura 2.3.1.c Tabla de elementos probabilísticos del modelo Kappa general

Modificación del Esquema General					
	Observador 2				
Observador1	1	2	...	C	**Marginal**
1	π_{11}	π_{12}	...	π_{1}	π_{1}
2	π_{12}	π_{22}	...	π_{2C}	π_{2}
.	.			.	.
.	.			.	.
.	.			.	.
C	π_{C1}	π_{C2}	...	π_{CC}	π c
Marginal	$\pi_{.1}$	$\pi_{.2}$	...	$\pi_{.C}$	1

Con el tipo de esquematización que hemos propuesto en los gráficos anteriores es evidente que las respuestas que indican concordancia son las que se sitúan sobre la diagonal principal. En efecto, si un dato se sitúa sobre dicha diagonal, ello significa que ambos observadores han clasificado el ítem en la misma categoría del sistema de clasificación. De esta observación surge naturalmente la más simple de las medidas, la proporción de concordancia esperada, es decir, la suma de las probabilidades a lo largo de la diagonal principal. Formalmente dicha medida se expresa por: $\pi_0 = \sum_{i=1}^{c} \pi_{i,i}$ cumpliendo $0 \leq \pi_0 \leq 1$.

El uso de esta medida conlleva a ciertos problemas en su interpretación. En la Figura 2.3.1.d. se ilustra el tipo de dificultades que pueden surgir. En el caso A, $\pi_0 = 0.2$, luego la concordancia es mucho menor que en el caso B, donde $\pi_0 = 0.8$. Sin embargo, condicionando por las distribuciones marginales se observa que en el caso A la concordancia es la máxima posible, mientras que en el B es la mínima.

Figura 2.3.1.d Tabla de Paradojas de Concordancia

Ejemplos de concordancia.							
A				B			
	Observador 2				Observador 2		
Observador 1	1	2	Marginal	Observador 1	1	2	Marginal
1	0.1	0.8	**0.9**	1	0.8	0.1	**0.9**
2	0	0.1	**0.1**	2	0.1	0	**0.1**
Marginal	**0.1**	**0.9**	**1**	**Marginal**	**0.9**	**0.1**	**1**

Por lo tanto, se justifica el uso de medidas de concordancia que tengan en cuenta las distribuciones marginales, con el fin de distinguir entre dos aspectos distintos de la concordancia, a los que podríamos aludir informalmente como concordancia absoluta o relativa. El índice kappa representa una aportación en esta dirección, mediante la incorporación en su fórmula de una corrección que excluye la concordancia debida exclusivamente al azar, corrección que como veremos, está relacionada con las distribuciones marginales. Ahora replantearemos la fórmula del índice kappa, κ conforme la notación de la tabla 2.3.1.c:

$$\pi_0 = \sum_{i=1}^{c} \pi_{i,i}$$

$$\pi_e = \sum_{j=1}^{c} \left(\sum_{i=1}^{c} \pi_{i,j} \sum_{i=1}^{c} \pi_{j,i} \right)$$

A fin de simplificar la notación se propone:

$$\pi_e = \sum_{i=1}^{c} \pi_{i,*} \pi_{*,i}$$

$$\kappa = \frac{\sum_{i=1}^{c} \pi_{i,i} - \sum_{i=1}^{c} \pi_{i,*} \pi_{*,i}}{1 - \sum_{i=1}^{c} \pi_{i,*} \pi_{*,i}} \qquad (1)$$

En la expresión anterior observamos que si suponemos la independencia de las variables aleatorias que representan la clasificación de un mismo ítem por los dos observadores, entonces la probabilidad de que un ítem sea clasificado por los dos en la misma categoría i es $\pi_{i,*}\pi_{*,i}$ Por lo tanto, si extendemos la suma a todas las categorías, $\sum \pi_{i,*}\pi_{*,i}$ es precisamente la probabilidad de que los dos observadores concuerden por razones exclusivamente atribuibles al azar. En consecuencia, el valor de κ simplemente es la razón entre la proporción de

concordancia observado más allá del atribuible al azar ($\sum_{i=1}^{c}\pi_{i,i} - \sum_{i=1}^{c}\pi_{i,*}\pi_{*,i}$) y la proporción de concordancia máximo posible más allá del azar ($1-\sum_{i=1}^{c}\pi_{i,*}\pi_{*,i}$).

El valor $\kappa = 0$ se obtiene cuando la concordancia observada es precisamente la que se espera a causa exclusivamente del azar. Si la concordancia es mayor que la esperada simplemente a causa del azar, $\kappa > 0$, mientras que si es menor, $\kappa < 0$. El mínimo valor de κ depende de la simetría de las distribuciones marginales.

En el ejemplo de la tabla 2.3.1.d κ vale 0.024 en el caso A y -0.111 en el B, lo que sugiere una interpretación de la concordancia opuesta a la del índice π_0. Son comprensibles estos resultados paradójicos, debido a que la proporción de acuerdos esperados sobre la base del azar en el caso B (0.82) es más alta que la probabilidad observada (0.80), es decir, el grado de acuerdo una vez corregido el debido al azar no existe, en contraste del 80% de acuerdo "crudo". Según la interpretación anterior, el acuerdo observado está compuesto en 100% del esperado por azar.

Landis y Koch[20] propusieron, y desde entonces ha sido ampliamente usada, la siguiente escala de valoración para κ:

Figura 2.3.1.d Escala de Interpretación del Índice kappa.

Valoración del índice Kappa	
Valor de k	**Fuerza de la concordancia**
< 0.00	Sin Acuerdo
0.00 < 0.20	Pobre
0.21 – 0.40	Débil
0.41 – 0.60	Moderada
0.61 – 0.80	Buena
0.81 – 1.00	Muy buena

La obtención de una simple estimación puntual del valor de κ no nos proporciona ninguna indicación de la precisión de dicha estimación. Desde el punto de vista inferencial es esencial

[20] http://www.hrc.es/bioest/errores_2.html#Landis

conocer la variabilidad de los estimadores y emplear ese conocimiento en la formulación de contrastes de hipótesis y en la construcción de intervalos de confianza.

Asimismo, $\hat{\kappa}$ es una función de variables aleatorias observadas de una muestra proveniente de un experimento binomial, por tanto queda claro que la distribución muestral teórica de $\hat{\kappa}$ es también una binomial. Podemos ahora establecer un estadístico de contraste, que pueda ser usado dentro de una normal estandarizada, siempre y cuando la probabilidad de acuerdo no se aproxime demasiado a 0 o 1 y el tamaño de muestra sea suficiente grande[21], bajo estos supuestos se propone como estadístico de contraste:

$$Z = \frac{|\hat{\kappa} - \kappa|}{\hat{\sigma}_{\kappa}} \qquad (1)$$

Fleiss, Cohen y Everitt [22] dan la expresión de la varianza asintótica del estimador de κ, cuando el verdadero valor de κ es cero:

$$\sigma_0^2(\kappa) = \frac{\sum_{i=1}^{c} \pi_{i,*}\pi_{*,i} + (\sum_{i=1}^{c} \pi_{i,*}\pi_{*,i})^2 - \sum_{i=1}^{c} \pi_{i,*}\pi_{*,i}(\pi_{i,*} + \pi_{*,i})}{n(1 - \sum_{i=1}^{c} \pi_{i,*}\pi_{*,i})^2} \qquad (2)$$

Reemplazando las probabilidades teóricas, que desconocemos, por las proporciones muestrales, obtenemos un estimador de $\sigma_0^2(\kappa)$ que denotaremos por $S_0^2(\kappa)$:

$$S_0^2 = \frac{n\sum_{i=1}^{c} x_{i,*}x_{*,i} + \frac{1}{n}(\sum_{i=1}^{c} x_{i,*}x_{*,i})^2 - \sum_{i=1}^{c} x_{i,*}x_{*,i}(x_{i,*} + x_{*,i})}{(n^2 - \sum_{i=1}^{c} x_{i,*}x_{*,i})^2} \qquad (3)$$

Podemos emplear este resultado para contrastar la hipótesis nula de que κ es cero frente a la alternativa de que no lo es, utilizando como estadístico del contraste el cociente:

$$Z = \frac{|\hat{\kappa}|}{S_0} \qquad (4)$$

Este cociente es la estandarización de una función de variable aleatoria con distribución muestral binomial, sin embargo trabajar bajo el supuesto de que el parámetro estimado sea

[21] Un estimador empírico del tamaño de una muestra, suficiente para representar una distribución normal es n=30.
[22] Fleiss JL, Cohen J, Everitt BS. Large sample standard errors of kappa and weighted kappa. Psychol Bull 1969; 72: 323-327.

cero es decir que κ=0 es de poca utilidad, debido a que es de esperar cierto grado de concordancia más allá del azar, y por tanto nos encontraremos trivialmente con un resultado significativo, rechazando la hipótesis nula κ=0.

Para realizar contrastes de hipótesis más interesantes, es necesario conocer la expresión de la varianza asintótica cuando no se supone que κ es cero. La expresión es sensiblemente más compleja que la anterior

$$\sigma_\kappa^2 = \frac{A + B - C}{n(1-\pi_e)^4} \qquad \text{donde:} \qquad (5)$$

$$A = \sum_{i=1}^{c} \pi_{i,i}\left[(1-\pi_e) - (\pi_{*,i} + \pi_{i,*})(1-\pi_0)\right]^2$$

$$B = (1-\pi_0)^2 \sum_{j \neq i=1}^{c} \sum_{j=1}^{c} \pi_{j,i}(\pi_{*,j} + \pi_{i,*})^2$$

$$C = (\pi_0 \pi_e - 2\pi_e + \pi_0)^2$$

Se puede demostrar que cuando κ es cero la expresión (5) se reduce a la (2). Para contrastar la hipótesis nula de que κ es igual a un valor dado κ_0 frente a una alternativa bilateral, procedemos como en el caso κ = 0, sólo que empleando como estadístico del contraste:

$$Z = \frac{|\hat{\kappa} - \kappa_0|}{\hat{\sigma}_\kappa}$$

donde $\hat{\sigma}_\kappa$ ahora es la raíz cuadrada de $\hat{\sigma}_\kappa^2$, el estimador de σ_κ^2 obtenido sustituyendo en (5) probabilidades por proporciones muestrales. Es obvio que el caso κ = 0 es un caso particular de este contraste con una mejor estimación del error estándar.

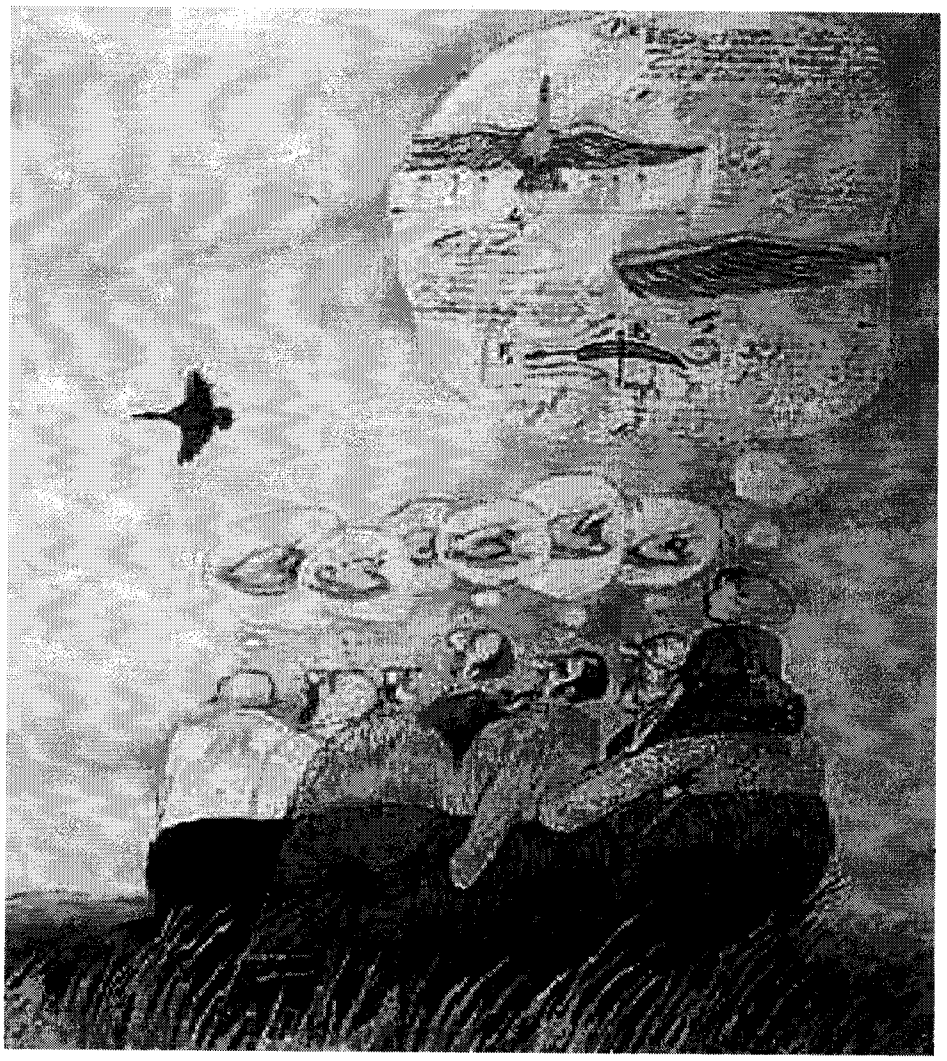
Un enfoque más versátil que el contraste de hipótesis, consiste en dar intervalos de confianza para el verdadero valor de κ. Tomados simultáneamente, $\hat{\kappa}$ y el intervalo de confianza nos dan, además de la mejor estimación de κ, una medida del error que podemos cometer con esa estimación. Un intervalo de confianza aproximado del (1-α)100%, construido por el método estándar, es de la forma: $[\hat{\kappa} - Z_{1-\alpha/2}\hat{\sigma}_\kappa, \hat{\kappa} + Z_{1-\alpha/2}\hat{\sigma}_\kappa]$ donde $z_{1-\alpha/2}$ es el percentil de orden (1-α /2)100% de la distribución normal estándar.

Cuando existen más de dos categorías, existe una alternativa que puede verse como intermedia entre un único kappa global o *K* kappas individuales para cada par de categorías. Es el denominado *kappa ponderado*, en el cual se asignan pesos para cuantificar la importancia relativa entre los desacuerdos. Es decir, no tiene la misma importancia un desacuerdo en la clasificación entre las categorías leve y moderada que entre leve y grave, obviamente la última representa un mayor desacuerdo que la primera, no abundamos más en este método por estar fuera de los fines de este trabajo, sin embargo se puede encontrar más información remitiéndose a Cohen J. (1968) Weighted kappa: Nominal scale agreement with provision for scaled disagreement or parcial credit *Psychol Bull* 70: 213-220.

En este capítulo, se profundizó respecto a la teoría de clasificación y la forma en que se complementan el modelo vectorial con el algoritmo de kohonen, para obtener un mecanismo de clasificación no supervisada. Asimismo, presentamos desde un enfoque más formal que en el capítulo primero, la definición de un agente de software, y lo que significa la minería de datos en la Web, conceptos que apoyan la elección de nuestras herramientas tecnologías de desarrollo. Finalmente, se expone el estadístico de evaluación Kappa, su definición, ámbitos de aplicación y pruebas de su efectividad en las ciencias biológica y médica. En el siguiente capítulo propondremos al estimador Kappa, como un factor de sensibilidad dentro de un agente de software y esto en conjunción con el algoritmo de clasificación dará a nuestro experimento un carácter novedoso.

Estructura del Modelo y Aplicación de Herramientas Propuestas.

Al observar a un niño con el debido cuidado, encontraremos las características básicas de la clase o categoría humana; sensibilidad, curiosidad, objetividad y el proceso primigenio de la inteligencia.



Capítulo 3

3.0 Introducción.

De la famosa cinta Artificial Intelligence de Steven Spielberg, derivaron conjeturas acerca del grado de sofisticación en el diseño de entidades inteligentes. En el film, queda de manifiesto que todos los robots, son de propósito único. Por ejemplo, el Robot Joe, estaba destinado a la satisfacción sexual humana, La niñera en la feria de la carne, mantiene su comportamiento protector y proveedor de cuidados hasta el instante antes de ser destruida, y David vive con el único objetivo de conseguir amor de lo que considera una madre, es sobresaliente que en aún en la visión futurista de "Spielberg" se construyan robots con metas estáticas e imperecederas. Puntualizamos este hecho debido a que nuestra tesis se fundamenta en el uso de un estadístico que provea de sensibilidad a los agentes de software, respecto a las preferencias de su usuario, y en este sentido cambie su ámbito de búsqueda.

Nuestro propósito general es encontrar información de calidad en Internet, lo cual implica una metodología de búsqueda eficiente, propósito nada trivial; ilustremos la idea con un caso de búsqueda rutinaria:

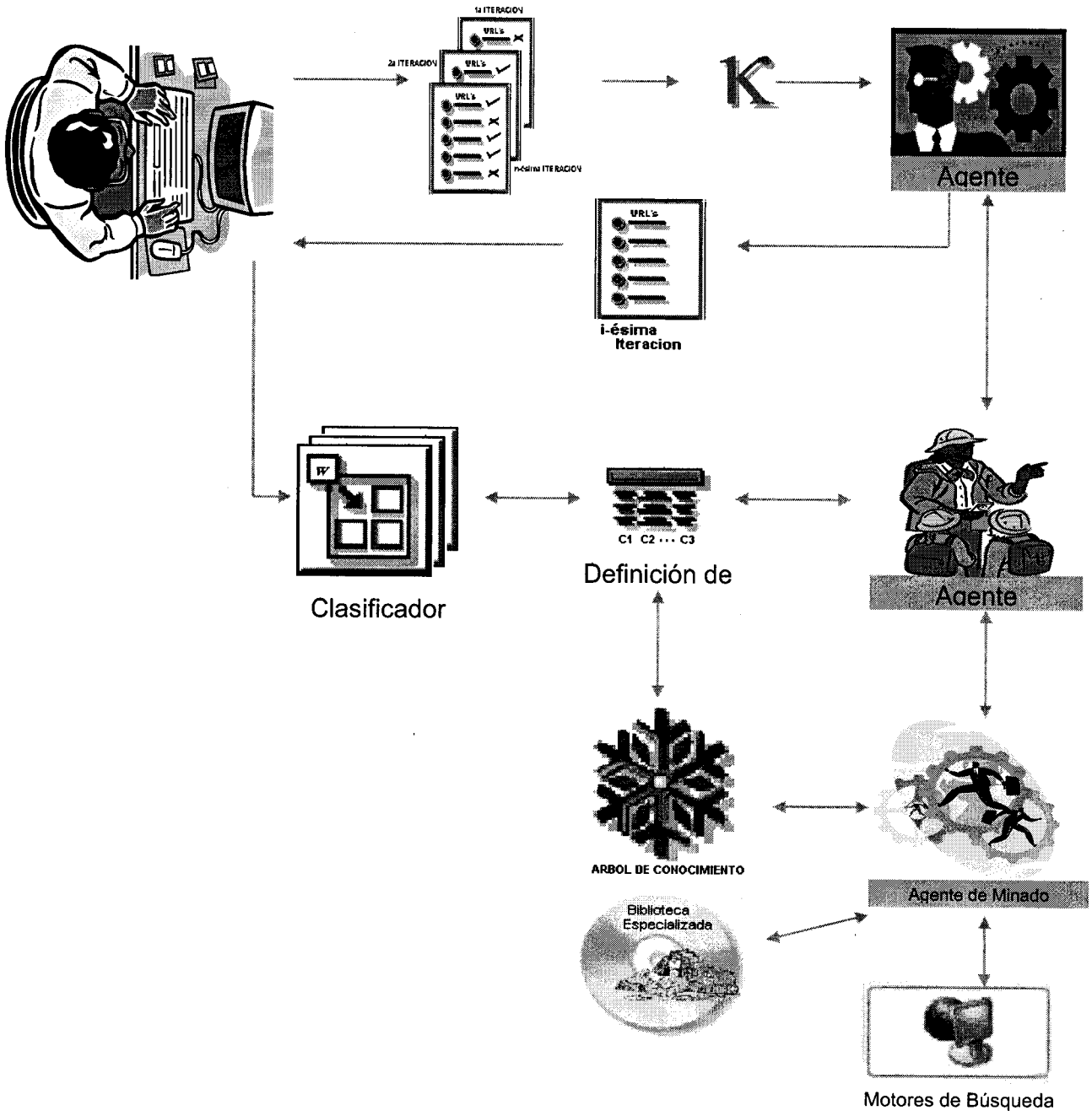
Esta mañana al vestirnos recordamos qué zapatos combinan con la ropa y hacen juego con el cinturón, entonces decidimos que los cafés están bien y nos disponemos a buscarlos, pero no comenzamos por la cocina, lo más obvio es que se encuentren en la recamara y seguramente en el closet y dentro de la zapatera o como alternativa pudieron haber quedado debajo de la cama. Una vez localizados los zapatos decidimos peinarnos y enseguida pensamos en el cepillo y entonces nos dirigimos al tocador o quizá al espejo del baño.

Este ejemplo de clasificación mental que desarrollamos en segundos, es el arquetipo del modelo que se propone a continuación:

3.1 Estructura de SABIO.

La estructura de SABIO tiene tres tipos de agentes: agente central o de Interfaz de usuario, (estos nombres serán usados indistintamente a lo largo del capítulo), Agente Intermedio o de razonamiento y agentes de minado de información. La estructura de los agentes es jerárquica, como se muestra en el gráfico 3.1.a.

Figura 3.1.a Diagrama de SABIO



Dado que las consultas del usuario están construidas, por palabras clave que en la mayoría de los casos representan un conjunto de temas diversos, y sin ninguna relación necesaria entre ellos, es conveniente la partición de tareas, para la especialización de las áreas de interés del usuario. De esta manera la búsqueda de documentos lleva a soluciones basadas en un ambiente multiagentes, cuya estructura es el patrón de relaciones de información y de control que existe entre los agentes, y la distribución de la capacidad de resolver problemas entre ellos.

Iniciaremos el análisis de esta estructura describiendo al agente central. La interfaz de usuario se realizará a través de un navegador y formularios en HTML. El usuario puede comprobar el estado de sus peticiones y modificarlas si lo desea. El agente central se encargar de interpretar las peticiones del usuario y determinar la consistencia de las mismas, mediante el índice de concordancia. Asimismo, trasmitir las peticiones a los agentes de razonamiento, quienes devuelven su respectiva lista de direcciones por tema o categoría. El agente central conjunta y presenta estas listas al usuario quien selecciona las direcciones que le parecen relevantes de la lista presentada y de esta manera se lleva a cabo una calificación de la categoría mejor aceptada, el agente central podrá indicar a cada agente intermedio la clasificación hacia donde debe dirigir la siguiente búsqueda de los agentes de minado.

Los agentes de razonamiento tienen como propósito dirigir la búsqueda del agente de minado a su cargo en dirección de la categoría asignada y en su caso reenviar a este agente de minado a buscar más información o detener su búsqueda derivado de la calificación que hubieran recibido los documentos presentados.

Los agentes de razonamiento además de administrar el tiempo de vida de los agentes de minado, también les indican la ruta y el medio donde habrán de buscar. Como se muestra en el diagrama 3.1.a. el sistema puede contar, además del recurso de la Web, con un servidor de bibliotecas especializadas, y con el árbol de conocimientos local, en el cual se encuentran alojadas y categorizadas las direcciones de documentos relevantes. La clasificación desarrollada en el árbol de conocimientos se realiza mediante un algoritmo genético híbrido, el cual utilizará como corpus de entrenamiento los documentos locales o corporativos según sea el caso.

Los agentes de minado son considerablemente más pequeños y en ese sentido almacenan cantidades pequeñas de información, características necesarias para un programa que tiene que ser trasmitido entre diferentes sitios y que usa recursos de cómputo en servidores ajenos. Un agente de minado se encuentra siempre al servicio de un agente intermedio, y le proporciona los documentos encontrados, generalmente existe un agente de minado para cada medio y categoría.

3.1.1 Arquitectura.

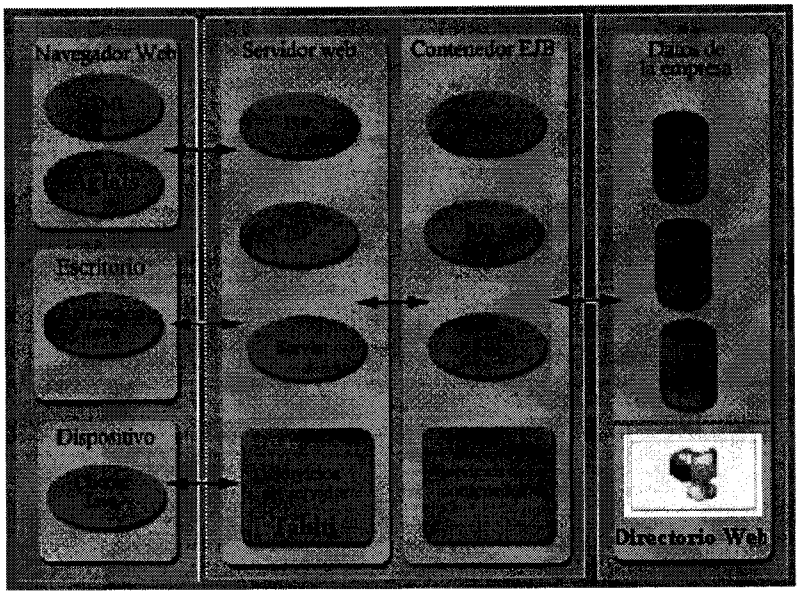
Al inicio de la década de los 90, (y aún hoy) muchos de los sistemas, utilizaban una arquitectura cliente-servidor, donde la interfaz de usuario se ejecuta en un ordenador de escritorio. A lo que llamamos el nivel del cliente. Los datos empresariales a los que tiene acceso la aplicación cliente se encuentran en una base de datos y los "provee" un administrador o servidor de la misma.

Sin embargo con la experiencia, la comunidad de desarrolladores se dio cuenta de que es muy complicado crear y mantener un sistema distribuido flexible utilizando el modelo cliente-servidor. Por ejemplo: cuando era necesario modificar la lógica empresarial, esto se tenía que llevar a cabo en cada equipo de escritorio de la empresa. El mantenimiento se convirtió en un proceso demasiado largo y costoso. Estas aplicaciones también tenían que gestionar las transacciones, preocuparse de la seguridad y procesar los datos de forma eficaz, todo ello con una interfaz atractiva y fácil de comprender por parte del usuario. Además, a principio de los noventa como ya hemos mencionado, la WWW da sus primeros pasos y los arquitectos de software comenzaron a interesarse por esa propuesta.

Una vez que se hicieron evidentes las limitaciones del modelo cliente-servidor, la comunidad de desarrolladores empezó a buscar un modelo mejor. El resultado es el modelo multinivel.

En el modelo multinivel, la lógica necesaria para presentar la interfaz de usuario reside en el nivel intermedio. Ahora, la lógica empresarial también se encuentra en el nivel intermedio. Si es necesario realizar cambios, se pueden llevar a cabo solamente en un lugar, sin necesidad de realizarlos en todos los equipos.

Figura 3.1.1.a. Diagrama de los componentes que se ejecutan en cada nivel



La aplicación cliente puede ser una página HTML o un Aglet que se ejecute en un visualizador, El nivel intermedio puede contar con páginas de JavaServer ™ o servlets que se ejecuten en un servidor web. Estos elementos componen la lógica de presentación del servidor. Los contenedores EJB proporcionan un entorno de ejecución para Enterprise JavaBeans™, que contiene la lógica empresarial de la aplicación. Tanto el servidor web como el contenedor Enterprise JavaBeans (EJB) ofrecen servicios a los componentes que se ejecutan en ellos. Gracias a que estos servicios siempre están disponibles, los programadores no necesitan incluirlos en los componentes que crean.

El nivel del Enterprise Information System (EIS - Sistema de información empresarial) es un repositorio de los datos de la empresa. Normalmente EIS se compone de la información contenida en un sistema de base de datos relacional.

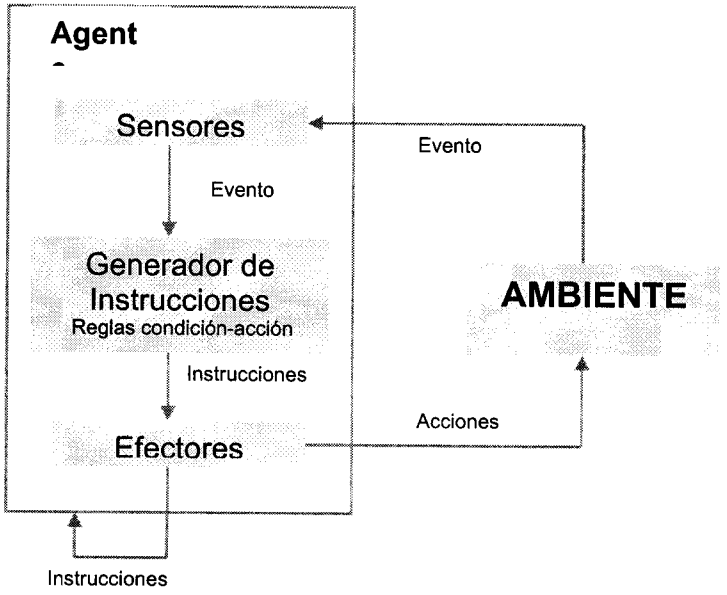
3.1.1.1 Agente de Interfaz.

La trascendencia del agente central radica en dos aspectos fundamentalmente, el primero se refiere a la facilidad para realizar las consultas y presentar los resultados para su selección. El segundo depende de la sensibilidad para poder determinar la preferencia del usuario y, es en este último punto donde concentraremos el objetivo de este trabajo.

El funcionamiento del agente central se encuentra orientado a eventos, es decir, no actúa por iniciativa propia, sino en respuesta a un estímulo externo. Dichos estímulos externos pueden ser originados por otro agente, por el usuario, o por el servidor de agentes y pueden causar una reacción por parte del agente central.

El agente central es la pieza fundamental en este trabajo, debido a la incorporación del estadístico Kappa como un sensor del ámbito de preferencia del usuario. Esta incorporación le da a nuestro agente central otra dimensión, ya que hasta este momento y como lo hemos mostrado en el capítulo primero las aplicaciones multiagentes toman al agente de interfaz como un actor por reflejo, que pasa directamente de las percepciones a las acciones sin realizar ningún proceso deliberativo, como se muestra en el siguiente esquema:

Figura 3.1.1.1.a. Diagrama de Agente de Interfaz.



Los sensores son los mecanismos que le permiten al agente central ser conciente de los eventos generados en el exterior y que le afectan directamente:

- El manejador de mensajes
- La ventana de usuario
- **El estadístico de evaluación Kappa**
- Las interfases de movilidad y persistencia

El manejador de mensajes indica al agente cuando éste ha recibido un nuevo mensaje proveniente de otro agente, en tanto que la ventana de usuario notifica al agente cuándo un evento ha sido originado. Asimismo en dicha ventana el usuario selecciona los documentos relevantes originando la materia prima para la obtención del estadístico de evaluación Kappa.

Las interfaces de movilidad y persistencia indican al agente cuándo un evento de creación, destrucción, movilidad, activación o desactivación, ha sido originado en el servidor de agentes.

Los efectores por otra parte, están construidos por métodos que pueden realizar acciones que tienen su efecto en el ambiente o en los componentes del mismo agente.

Los estímulos provocados por otro agente toman la forma de mensajes dirigidos al agente central, en tanto que los eventos generados por el usuario se originan en la interfaz gráfica del agente.

A continuación se presenta una relación de Mensajes – Acciones y Eventos – Acciones tomadas por el agente central:

Tabla 3.1.1.1.a. Mensajes - Acciones (Agente Central).

Mensaje	Acciones
evaluateDocuments	▪ Incorpora a su base interna los documentos recibidos, es decir, al árbol de conocimientos. ▪ Despliega los nombres en la ventana. ▪ Notifica al usuario que tiene nuevos documentos que sugerirle.
proposedDocuments	▪ Incorpora los documentos propuestos a su base interna de documentos generados por consultas. ▪ Notifica al usuario que responden a una consulta hecha por él.

Tabla 3.1.1.1.b. Evento – Acciones del Usuario hacia el Agente Central.

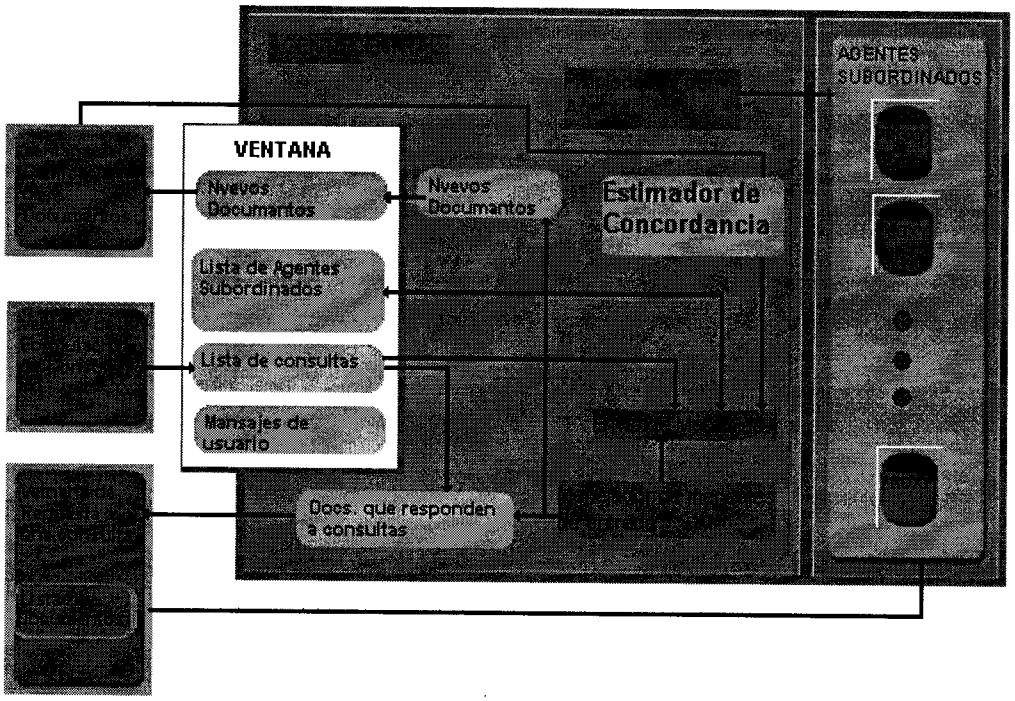
Evento	Acciones
Consulta	▪ Ofrece al usuario una ventana para que ingrese las palabras clave. ▪ Notifica a todos los agentes de razonamiento sobre la consulta que el usuario ha hecho.
Ver documentos propuestos	▪ Muestra una ventana con los documentos propuestos que contestan una consulta
Requerir más documentos	▪ Crea un nuevo agente de razonamiento que trabaje sobre el tema consultado.
Evaluar direcciones de documentos	▪ Permite que el usuario seleccione las direcciones de los documentos sugeridos. ▪ Regresa la categoría mejor evaluada a su agente de razonamiento correspondiente.
Ver agente	▪ Notifica al agente en cuestión que el usuario desea ver su ventana.
Ver información sobre agente	▪ Muestra información técnica sobre el agente en cuestión
Terminar	▪ Cierra la ventana. ▪ Notifica a todos los agentes de razonamiento que el programa está terminado. ▪ Actualiza las propiedades del programa. ▪ Salva las propiedades en el disco.

A continuación se muestra la relación Evento – Acción del agente de interfaz y eventos originados por el aglets server, es decir, plataforma sobre la cual corren los agentes:

Tabla 3.1.1.1.c. Evento – Acciones del Aglet Server hacia el Agente Central.

Evento	Acciones
Create	Crea su ventana Carga las propiedades del sistema en memoria Crea los agentes de razonamiento
Dispatch	Incorpora los documentos propuestos a su base interna de documentos generados por consultas. Notifica al usuario que responden a una consulta hecha por él.
Revert	Arroja una excepción de seguridad que será manejada por el aglets server (Tahiti)
Deactivate	Oculta la ventana
Activate	Muestra o trae al frente la ventana
Dispose	Cierra la ventana. Notifica a todos los agentes de razonamiento que el programa está terminado. Actualiza las propiedades del programa. Salva las propiedades en el disco.

Figura 3.1.1.1.b. Estructura Funcional del agente central.



Dado que el agente central es un agente orientado a eventos, el control del funcionamiento del programa queda a cargo de dos estructuras: el manejador de mensajes de entrada, y la interfaz de usuario.

El manejador de mensajes de entrada se encarga de incorporar a las dos bases internas de documentos, los documentos recibidos de agentes intermediarios a través de mensajes, así como de organizarlos de acuerdo al agente que los obtuvo y a la consulta que los generó, si es el caso.

La interfaz de usuario cuenta con tres ventanas cuyos propósitos se describen a continuación:

- Ventana principal:

- Muestra los nombres de los agentes subordinados y permite pedir al agente central información relativa a él o que llame a alguno de ellos.
- Muestra los documentos que el agente central sugiere en ese momento.
- Muestra las consultas hechas por el usuario y permite pedir al agente central que muestre los documentos que responden a alguna de ellas.

- Despliega mensajes del agente central para el usuario

- Ventana de formulación de consultas:

- Permite formular al agente central una consulta explicita

- Ventana de respuesta a una consulta:

- Despliega en la pantalla los nombres de los documentos que responden a una consulta explicita por parte del usuario.
- Permite la calificación de los documentos sugeridos.
- Indica al agente central los documentos de interés para el usuario y si es necesario aún más documentos relacionados con el tema.

Por otra parte, el agente central cuenta con cuatro respositorios internos de información:

- Nuevos documentos: Documentos que ha recibido de agentes intermediarios y que esperan para ser evaluados por el usuario. Éstos se encuentran organizados de acuerdo al agente que los obtuvo, de manera que el agente central realice fácilmente la tarea de redistribución de los mismos a los agentes correspondientes una vez que hayan sido evaluados por el usuario.

- Documentos que responden a consultas: Documentos que han sido recibidos de otros agentes como respuesta a consultas explícitas del usuario. Se encuentran organizados de acuerdo a la consulta que los generó, de tal manera que el agente central pueda mostrar al usuario únicamente aquellos documentos que corresponden a la consulta cuya solución solicita el usuario en un momento dado.

- Agentes subordinados: Constituyen en conjunto de referencias a los apoderados (proxies) de los agentes Intermedios que se encuentran al servicio del agente central. Esta colección de referencias permite al agente enviar fácilmente mensajes punto a punto a cada uno de los agentes Intermediarios, así como administrar y controlar la cantidad de agentes a su servicio.

- Propiedades del agente: Es un conjunto de parámetros configurables que determinan diversos aspectos sobre el funcionamiento del agente, cuyo propósito es guardar tales parámetros en un medio de almacenamiento secundario para preservar tal información aún después de periodos de inactividad del sistema.

3.1.1.2 Agente Intermedio.

Al igual que en el caso del agente central, las acciones del agente intermedio tienen su origen en eventos provenientes del ambiente exterior, constituido por usuarios, otros agentes y, sistema.

Los estímulos que tienen su origen en acciones del usuario, se transmiten al agente a través de la interfaz gráfica constituida por una ventana de un agente Intermedio no visible, sin embargo el agente Intermedio puede mostrarla si el agente central se lo pide.

La siguiente tabla muestra las acciones del usuario sobre la ventana y acciones correspondientes del agente Intermediario:

Tabla 3.1.1.2.a. Evento – Acciones del Usuario hacia el Agente Intermedio.

Evento	Acciones
Llamar a un agente	▪ Notifica al agente de minado que el usuario quiere ver su ventana
Cerrar	▪ Cierra la ventana notifica a los agentes de minado a su servicio que está cerrando su ventana

A continuación se relacionan los mensajes recibidos de otros agentes con las acciones tomadas por el agente Intermedio:

Tabla 3.1.1.2.b. Mensaje – Acciones de Agentes hacia el Agente Intermedio.

Mensaje	Acciones
Evaluated Document	▪ Si el documento tiene una calificación satisfactoria, lo incorpora a su base de documentos registrados, y modifica su modelo de preferencias del usuario con base

	en información extraída del documento.
Salve Query	▪ Selecciona de su base interna de documentos aquellos que satisfagan la consulta. ▪ Envía copias de tales documentos al agente central
Show Window	▪ Muestra su ventana en pantalla
New Dosuments	▪ Selecciona los documentos que cumplan con las preferencias actuales del usuario. ▪ Envía el resto a otros agentes Intermediarios. ▪ Envía los documentos seleccionados al agente central para mostrarlos al usuario y obtener su avaluación. ▪ Envía las preferencias actuales del usuario al agente Descubridor
SelectDocuments	▪ Selecciona los documentos que le interesen.
terminating	▪ Salva las propiedades y preferencias del usuario en disco. ▪ Cierra su ventana si está abierta

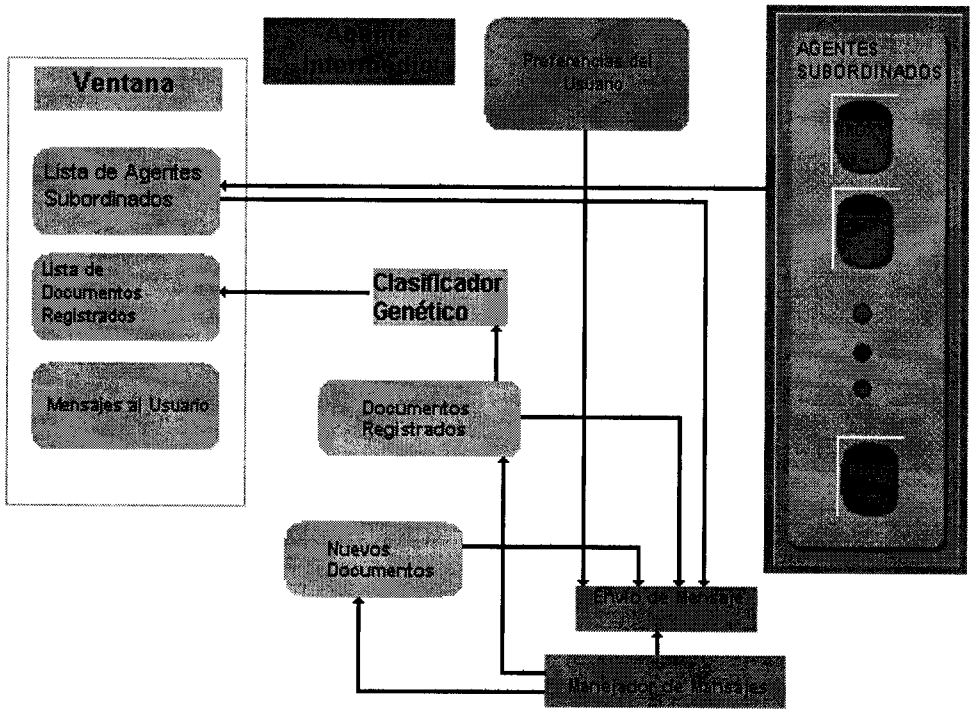
Los eventos originados en el servidor de aglets también causan una reacción por parte del agente intermediario. La siguiente tabla muestra la relación entre tales eventos y las acciones del agente:

Tabla 3.1.1.2.c. Evento – Acciones del Aglet Server hacia el Agente Intermedio.

Evento	Acciones
Create	▪ Carga las propiedades y preferencias del usuario en memoria. ▪ Se subscribe a ciertos mensajes que son difundidos por otros agentes. ▪ Crea su ventana. ▪ Crea los agentes Descubridores necesarios.

Dispatch	▪ Arroja una excepción de seguridad que será manejada por el servidor
Revert	▪ Arroja una excepción de seguridad que será manejada por el servidor
Deactivate	▪ Oculta la ventana
Activate	▪ Muestra o trae al frente la ventana
Dispose	▪ Salva las propiedades y preferencias del usuario en el disco. ▪ Cierra la ventana. ▪ Notifica a todos los agentes de minado que se encuentren presentes que está cerrando su ventana.

Figura 3.1.1.2.a. Estructura Funcional del Agente Intermedio.



3.1.1.3 Agente de Minado.

El agente de minado es el único de los tres tipos de agentes que intervienen en este sistema, que tiene que habitar, no en uno, sino en una gran cantidad de ambientes diferentes, cada uno de ellos de naturaleza altamente complicada. Por tal motivo, los mecanismos que regulan sus actividades requieren de una mayor complejidad.

Podemos caracterizar el ambiente que habitará el agente Descubridor como:

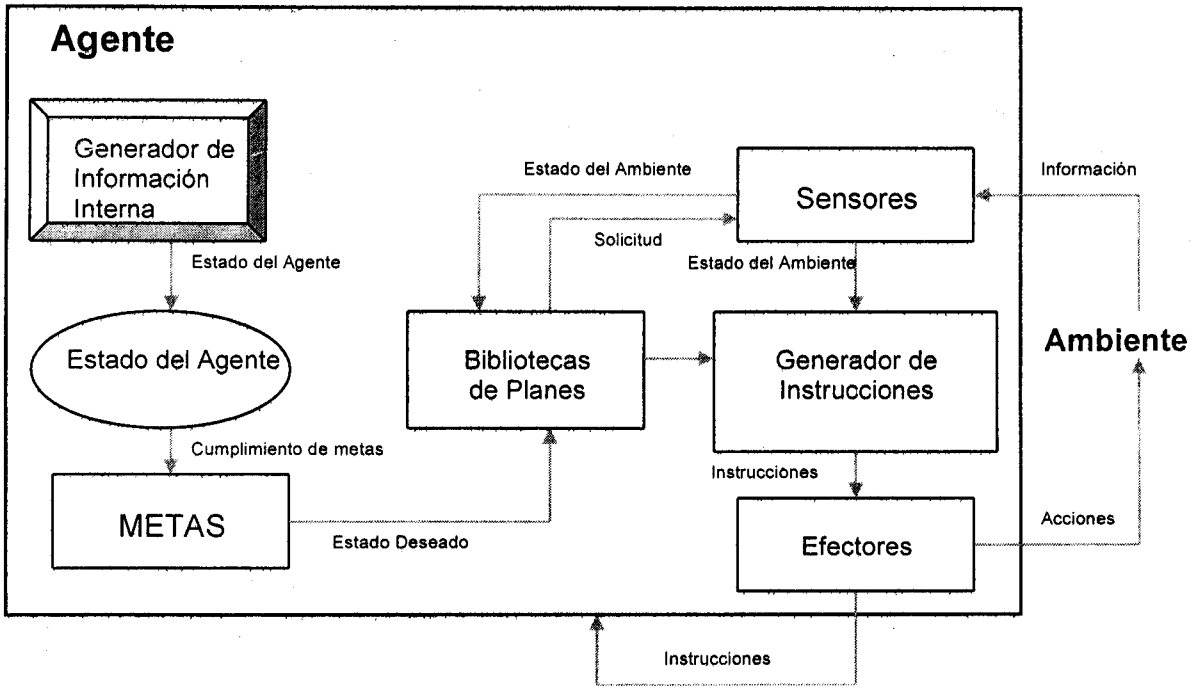
- Inaccesible. Existen atributos del ambiente que intervienen directamente en el éxito del Agente en la realización de sus tareas, y que sin embargo no pueden ser conocidos por el aparato sensor del agente.
- Dinámico. El ambiente cambia a través del tiempo, y es posible que lo haga mientras el agente se encuentra decidiendo su próxima acción.
- No determinístico. Los cambios que se suceden en el ambiente no pueden ser determinados a partir de su estado actual.

Además de las dificultades arriba mencionadas, existe otra característica de este agente que justifica la mayor elaboración de su sistema de toma de decisiones, y que constituye la principal diferencia entre el agente de minado y los otros dos tipos de agentes. Es su proactividad

A diferencia de los agentes Central e Intermedio, el agente de minado no actúa únicamente incitado por estímulos externos, sino que lo hace también con el deseo de cumplir un conjunto de metas que le han sido asignadas desde su creación. Ésa es la razón de que se encuentre siempre en estado de ejecución, mismo que sólo se interrumpe para atender a los eventos externos originales por mensajes, eventos en la interfaz de usuario, o eventos originales en el servidor de agentes.

En los dos tipos de agentes más simples, el sistema de decisión consta de un grupo de reglas condición-acción, en tanto que el agente de minado, este sistema está conformado de varios procesos y repositorios de información que trabajan conjuntamente para proveer al agente de un comportamiento mas flexible como se muestra en la siguiente figura:

Figura 3.1.1.3.a. Estructura Funcional del agente de minado.



Esta arquitectura que permite razonar acerca de las acciones que deben realizarse en ambientes dinámicos, sigue los lineamientos del sistema de razonamiento procedural o PRS, propuesto por Georgeff e Ingrand, en 1989,. En este sistema, el estado del ambiente, el estado del agente, el estado deseado, y las interacciones, se representan explícitamente y juntos determinan las acciones del sistema.

Básicamente, la principal mejora consiste en que el Generador de instrucciones no genera por sí mismo las acciones que los efectores deberán llevar a cabo, sino que utiliza para ello un plan facilitado por una biblioteca de planes.

Un plan es un conjunto de acciones que pretenden realizarse de manera secuencial y ordenada. La biblioteca de planes es capaz de sugerir un plan si conoce cuál es el objetivo del agente, es decir, a dónde quiere llegar. Esa meta que el agente desea lograr está representada por un estado es decir, por un conjunto de atributos del agente o del ambiente con un valor determinado.

Debido a que el ambiente en que existirá el agente de minado no puede ser alterado por éste (y no debe, por cuestiones de seguridad), las metas del agente quedan expresadas en términos de un estado deseado del agente en si, y no en términos de un estado deseado del ambiente, que no puede ser manipulado (la única acción que un agente puede realizar sobre el ambiente es el envío de mensajes a otros agentes, pero eso no se toma en consideración por dos razones: el agente no conoce el efecto de sus mensajes en los demás agentes y, la representación interna del ambiente y del estado del agente, no incluyen una representación del estado de los demás agentes, por lo tanto tal representación no puede ser incluida como una meta.

Si la biblioteca de planes encuentra que existe más que un camino para lograr el estado deseado, decide pedir información relevante acerca del estado actual del ambiente, es decir, factores ambientales de cuyos valores dependa la convivencia de aplicar uno u otro de los posibles planes.

La biblioteca de planes puede tener la siguiente forma:

Estatus Actual	Estatus Deseado	Plan
Estatus	Estatus	$Acción_1$, $Acción_2$, ... $Acción_n$
...	...	...
Estatus	Estatus	$Acción_1$, $Acción_2$, ... $Acción_n$

El agente, por otra parte, tiene sólo una meta que lograr: obtener un reconocimiento por parte del agente Intermediario que lo creó. Sin embargo, toda meta puede estar constituida por un conjunto de submetas, cada una más sencilla de lograr que la meta superior, que a su vez es fácil de alcanzar cuando se ha logrado cada una de las submetas de que consta. De esta manera, la meta del agente es en realidad un árbol de metas y submetas y, es agente trata siempre de satisfacer las metas más básicas.

Podemos representar al árbol de metas con la siguiente tabla:

META

Nombre		
Estado que la representa		
¿Ha sido lograda?		
$Meta_1$	$Meta_2$ ...	$Meta_n$

El agente realiza una revisión periódica del estado actual para identificar las metas que quedan satisfechas con ese estado, de manera que intentente alcanzar las metas subsecuentes.

Los sensores del agente están constituidos por los siguientes mecanismos:

- Interfaz con el sistema remoto
- El Manejador de mensajes
- La ventana de usuario
- Las interfases de movilidad y persistencia

La interfaz con el sistema remoto permite al agente obtener información de ese sistema de manera local, es decir, una vez que el agente ha viajado hasta allí y el sistema ha aceptado hospedarlo. Esta información puede ir desde el nombre de host hasta información sobre la red a la que pertenece. Queda como responsabilidad del diseñador de los agentes decidir la cantidad, calidad y variedad de la información a la que el agente tendrá acceso estando en un host remoto.

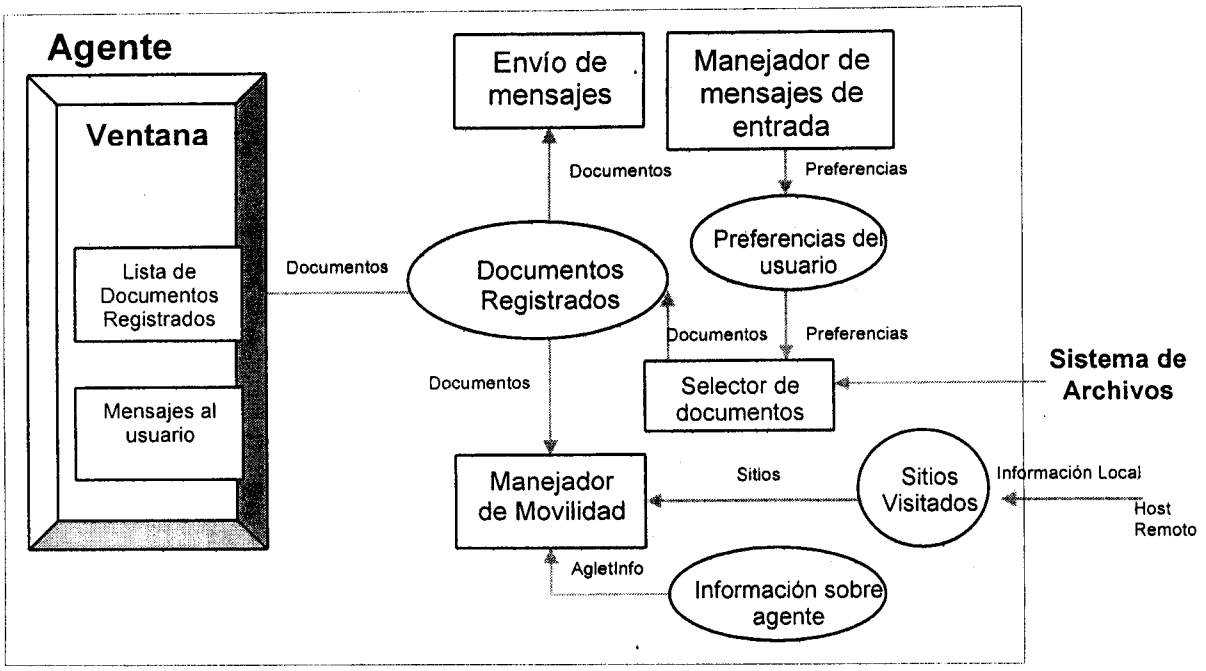
A pesar de que el agente de minado es un agente orientado a metas, es también capaz de reaccionar ante eventos de la manera que se describe a continuación:

- El manejador de mensajes indica al agente cuando éste ha recibido un nuevo mensaje proveniente de otro agente, en tanto que la ventana de usuario notifica al agente cuando un evento ha sido originado por éste.

- Las interfases de movilidad y persistencia indican al agente cuando en evento de creación, destrucción, movilidad, activación o desactivación ha sido originado en el servidor de agentes.

Los efectores, por otra parte, están constituidos por métodos que pueden realizar acciones que tienen su efecto ya sea en el ambiente (envío de mensajes), o en el propio agente y sus componentes internos.

Figura 3.1.1.3.b. Estructura del Flujo de datos en un agente de minado.



- Durante la creación de un nuevo agente de minado el agente Intermedio al cual servirá, le proporciona las preferencias del usuario a las cuales deberá apegarse, y lo vuelve a hacer cada vez que el agente de minado regresa al equipo usuario, de manera que esta última cuente siempre con una representación actualizada de las preferencias del usuario.

- Cuando el agente se encuentra ejecutándose en un host remoto, tiene acceso el sistema de archivos y utiliza tales preferencias para seleccionar los documentos que considere de interés para el usuario. El nombre y la localización de los documentos, así como una descripción de su contenido, son almacenados en un repositorio interno de documentos, y mostrados en la ventana del agente.

- Además, el agente cuenta con una lista interna de los sitios que ya ha visitado, y utiliza dicha información, junto con la lista de documentos e información sobre el agente mismo, para decidir cuáles serán sus próximos movimientos por la red.

3.2 Aplicación del Algoritmo de Clasificación y el Estadístico de Concordancia.

Entre los aspectos que abarca el comportamiento del sistema multiagentes que nos ocupa, sólo nos enfocaremos en dos susceptibles de cambio a lo largo del tiempo, con el fin de ofrecer facilidad de adaptación a su naturaleza dinámica. Estos dos factores son: el número de agentes del que dispone el sistema y el tema al que se orienta cada uno de ellos.

El número de Intermediarios en el sistema responde a la variedad de los temas en los que el usuario muestra interés y corresponde al agente central controlar su número, de manera que se ofrezca al usuario un servicio que cubra todas las áreas de interés, y que no se ocupen de temas en los que no lo hay.

La tarea correspondiente del agente Intermedio es controlar el número de agentes de minado que se encuentran a su servicio, de tal manera que la cantidad de documentos que periódicamente entrega al agente de interfaz como sugerencias, sea conveniente.

El agente intermediario es responsable de mantener actualizada su representación interna de las preferencias del usuario respecto al tema particular que maneja, analizando la manera como el usuario evalúa los documentos sugeridos y registrando los posibles cambios en tal representación.

En un principio, el sistema cuenta sólo con el agente de interfaz. Cuando el usuario realiza una consulta explícita al agente de interfaz, éste a su vez, hace la misma pregunta a cada uno de los agentes Intermediarios para que trate de ofrecer documentos adecuados. En este caso, el agente central no cuenta con ningún agente intermediario que responda a la pregunta, por lo que decide crear uno nuevo, encargado exclusivamente de responder a la consulta.

Una consulta se considera como un conjunto de características que se espera que contenga un documento ofrecido como respuesta. En este sistema, la única característica que compone a una consulta, es una serie de palabras clave. Esta característica es por lo tanto, la misma que se utiliza para obtener la caracterización de un documento.

Las palabras clave constituyen la manera en que los agentes registran las preferencias del usuario, las consultas hechas por él y el contenido de los documentos.

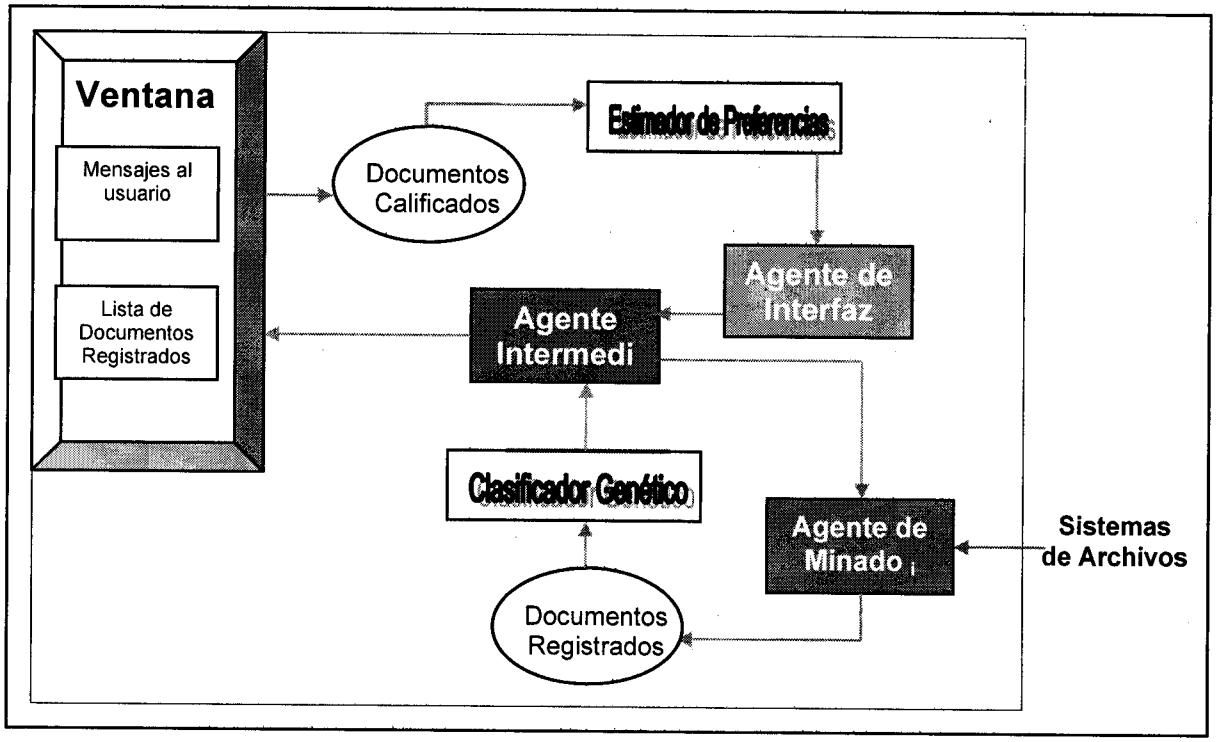
Como se expuso en el primer capítulo existen métodos eficientes (Booleano o Vectorial) para que un agente determine el grado de interés que puede residir en un documento respecto a las preferencias del usuario y los más populares son aquellos que utilizan una representación análoga del documento respecto a las frecuencias de las palabras clave; la selección de un método en particular para la determinación de similitud, dependerá de los recursos con los que se cuente; debemos considerar que viajaremos a un servidor remoto el cual nos alojará y ofrecerá los recursos necesarios para desarrollar nuestras aplicaciones, en ese sentido debemos ser muy cuidadosos; sin embargo, una vez que decidimos qué documentos traer de regreso, es necesario clasificarlos, con el objetivo de preparar la siguiente búsqueda y, para este propósito, se ofrece el algoritmo de Kohonen.

La idea básica, es que un grupo de agentes de minado regresen cada uno con un conjunto de documentos y antes de que sean presentados al usuario por el agente de interfaz, estos documentos sean clasificados por un algoritmo genético, de tal manera que cuando el usuario seleccione los documentos de su interés el sistema esté en posibilidad de identificar el tema asociado a dichos documentos con el objetivo de preparar la siguiente búsqueda y, al mismo tiempo alimentar las bases de datos respecto a las preferencias del usuario.

El siguiente paso que es la aplicación de la herramienta estadística del índice Kappa, el cual nos ayudará a identificar si nuestro usuario a cambiado de preferencias, esta percepción respecto a las preferencias del usuario tiene una característica fundamental; está más allá del azar.

En síntesis, la aplicación del algoritmo genético y el uso del estadístico Kappa es la diferencia fundamental entre el modelo SABIO y los sistemas multiagentes expuestos anteriormente. Ahora con la intención de ser más explícitos, plantearemos el siguiente diagrama de operación, en el cual se presenta la conceptualización para el uso del algoritmo genético de Kohonen y el estimador Kappa, dentro de las funciones de los agentes de interfaz e intermedio.

Figura 3.2.a. Diagrama conceptual de la aplicación del Algoritmo de Kohonen y el indice Kappa dentro de un ámbito de sistemas mutiagentes.



Los agentes de minado presentan el conjunto de todos los documentos registrados y, estos son clasificados por el genético, una vez que los documentos pertenecen a una categoría en particular entonces se notifica al agente Intermedio con la siguiente información:

1.- Grado de concordancia del documento
2.- agente de minado que propone el documento
3.- Categoría a la que pertenece el documento

Esta información prepara el campo para que una vez que las direcciones donde se encuentran los documentos sean calificadas por el usuario, se tenga la posibilidad de realizar la siguiente búsqueda la cual se autogenerará siempre y cuando el usuario no involucre alguna otra palabra clave, de otro modo se tiene que incorporar esta palabra dentro del vector de preferencias y se iniciá el proceso. De este modo contamos con un conjunto de categorías en las cuales es posible clasificar el siguiente conjunto de resultados. Así podemos establecer si las direcciones calificadas siguen considerándose en la misma categoría; si es el caso, el sistema de búsqueda

está en buena dirección y, el agente intermedio enviará a los mismos agentes de minado a buscar más, en la misma zona, por decirlo de manera metafórica; sin embargo es en esta parte donde radica la importancia del estadístico de concordancia, debido a que si nos indica un cambio de preferencias en el usuario no gastaremos recursos enviando a agentes de minado a la casa de información basura y al dispendio de recursos, esta parte da un aspecto crítico y de síntesis operativa a nuestro sistema.

3.2.1 Codificación de Documentos.

El modelo plantea un sistema multiagentes que deberá traer información de la Web, y nos permita realizar búsquedas posteriores con un grado de precisión mayor, es decir, emplearemos una arquitectura de la cual ya se ha probado su eficiencia[1] y factibilidad a lo largo de esta tesis, mediante la exposición de diversos ejemplos.

Dentro del modelo, los agentes son un mecanismo para la recolección de información, a los cuales se pueden incorporar las técnicas de clasificación y estadístico de concordancia; sin embargo, las herramientas propuestas tienen una aplicabilidad de ámbito general, en ese sentido, para el desarrollo de nuestra experimentación se propone un conjunto de páginas Web recolectadas por motores de búsqueda simulando la función de los agentes de minado.
e esta forma nuestro modelo el cual necesita como entrada un conjunto de palabras clave para realizar la búsqueda nos ofrecerá en primera instancia un conjunto de documentos extraídos de la Web por metacrawlers.

Una vez que contamos con un conjunto de páginas o documentos, el primer paso en un proceso de clasificación, es tener una interpretación mediante un grupo de variables de los ítems a clasificar. Para nuestros propósitos lo más conveniente es la utilización de una abstracción numérica y el modelo vectorial es una magnifica opción[2]. La representación de un documento a través de un vector **n** dimensional implica ciertas particularidades, la más importante es el establecimiento de **n**, es decir, el número de palabras que deberemos tomar

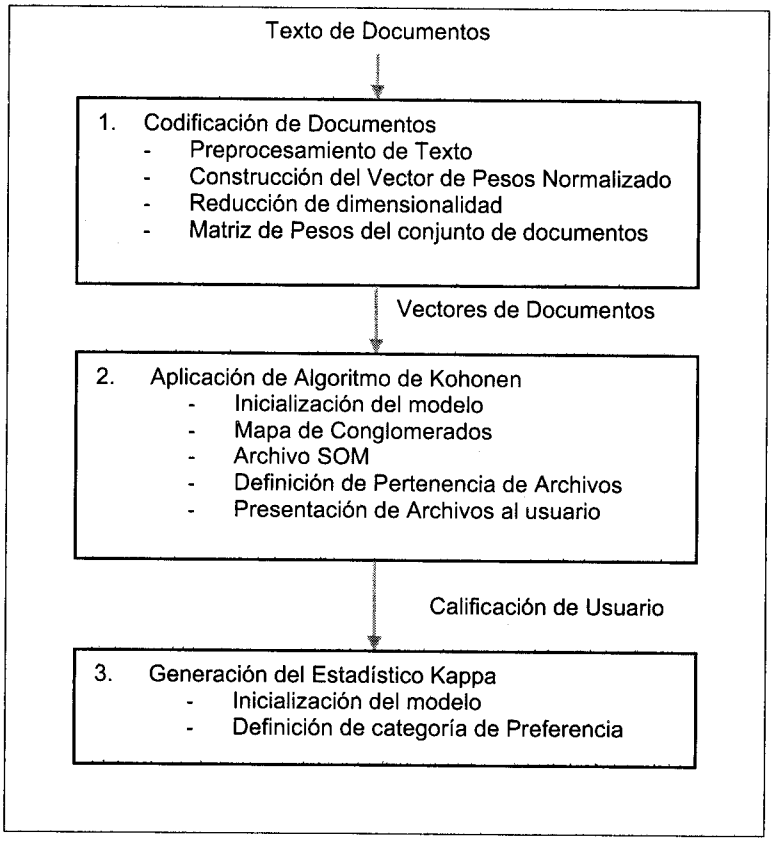
[1] Why Multiagent System? http://www.cs.cmu.edu/afs/cs/pstone/public/papers/96ieee-survey/node5

[2] Text Mining with the WEBSOM.- KRISTA LAGUS.- Helsinki University of Technology
Neural Networks Research Centre.- P.O.Box 5400.-FIN-02015 HUT, Finland
Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 11th ofDecember, 2000, at 12 o'clock noon. Helsinki University of Technology Department of Computer Science and Engineering Laboratory of Computer and Information Science ESPOO 2000

como dimensión suficiente para caracterizar un documento. Debemos recordar en todo momento que nuestra aplicación debe trabajar mediante percepciones sencillas y lo más simple posible, debido a que estamos interactuando en tiempo real con el usuario, además nuestro ámbito de trabajo es la Web.

Dada la naturaleza de nuestro escenario, habremos de definir la dimensión de nuestro vector característico con base en la dimensión del vector de consulta, pero este dato será necesario hasta el momento de clasificar, para lo cual antes deberemos desarrollar una tarea llamada pre-procesamiento la cual describiremos a detalle más adelante; en este momento esquematizaremos el plan general del trabajo de experimentación:

Figura 3.2.1.a. Esquema general de Experimentación.



La metodología que ocuparemos para la codificación de un documento estará basada en el modelo vectorial, la cual fue abordada en el capítulo anterior. A continuación se describen los pasos del algoritmo de codificación:

1. Filtrado de palabras: Se extrae el conjunto total por documento, eliminando las palabras irrelevantes, signos y caracteres extraños.

2. Determinación de pesos $\mathbf{w_{ij}} = \boldsymbol{g_i}(\overrightarrow{dj})$ Al aplicar las ecuaciones del algoritmo vectorial.

3. Para definir el número de palabras características de un documento se toma el doble producto de la dimensión del vector de consulta, que para nuestro experimento se establece como **d =10** palabras por documento, las 10 más altas en $W_{ij.}$

4. Finalmente se conforma una matriz de **Nxd**. Donde se muestra el peso de cada uno de los **d** términos en cada uno de los **N=41** documentos.

Es necesario conforme al diseño del modelo clasificar los documentos antes de mostrar parte de ellos a nuestro usuario y es en este momento que utilizaremos el algoritmo de Kohonen.

3.2.2 Aplicación de Algoritmo de Kohonen.

Además de tener un esquema de aprendizaje competitivo, el modelo SOM aporta una importante novedad, pues incorpora relaciones entre las neuronas próximas en el mapa. Para ello, introduce una función denominada zona de vecindad que define un entorno alrededor de la neurona ganadora actual (vecindad); su efecto es que durante el aprendizaje se actualizan tanto los pesos de la vencedora como los de las neuronas pertenecientes a su vecindad, de esta manera en el modelo SOM se logra que neuronas próximas sintonicen con patrones similares, quedando así reflejada sobre el mapa una cierta imagen del orden topológico presente en el espacio de entrada.

En nuestro caso utilizaremos un paquete disponible en la red[3] al que introduciremos la matriz resultado del proceso anterior y, algunos parámetros los cuales se explican dentro de la siguiente metodología:

[3] http://gepas.bioinfo.cnio.es/tools.html

1. Aplicar un patrón de entrada, es decir, matriz de pesos.

2. Calcular mediante la fórmula de distancia euclidiana o la función gaussiana, la similitud o disimilitud entre las entradas y los pesos de las conexiones.

3. La unidad de salida con los pesos más parecidos al patrón de entrada (es decir, menor Dj) es declarada ganadora. El vector de pesos de la unidad ganadora, Wc, se convierte en el centro de un grupo de vectores cercanos al Wc, es decir, a menos de cierta distancia D.

4. Modificar los pesos de los vectores de pesos Wj "cercanos" a Wc (distancia menor a D), según la fórmula:

$$\Delta W_j = \eta(\chi - W_j)$$

De esta manera conseguimos que los vectores de pesos de la unidad ganadora y de su "vecindario" se parezcan cada vez más al patrón de entrada que hace ganar a esa unidad. La cuantía de la adaptación se puede escalar de acuerdo a una "función de vecindad" preestablecida L (j,c):

$$\Lambda(j,c) = \frac{\exp(-|r_j - r_c|)^2}{2\sigma^2}$$

$$\Delta W_j = \eta\Lambda(j,c)(\chi - W_j)$$

Representa la posición de la neurona j en el espacio de salida. La convergencia del mapa depende de escoger apropiadamente r_j. Una opción es $h = \frac{1}{m}$, siendo m el número de iteración del proceso de aprendizaje. El tamaño del vecindario debería decrecer gradualmente.

5. Repetir los pasos 1 a 4 con todos lo patrones de entrada

Ahora, aplicaremos los valores de nuestro experimento y, obtendremos la clasificación de los documentos a través de un archivo de de texto y un mapa topológico. Utilizaremos el programa que se muestra en la figura 3.2.2.a, en donde se introducen los siguientes parámetros iniciales:

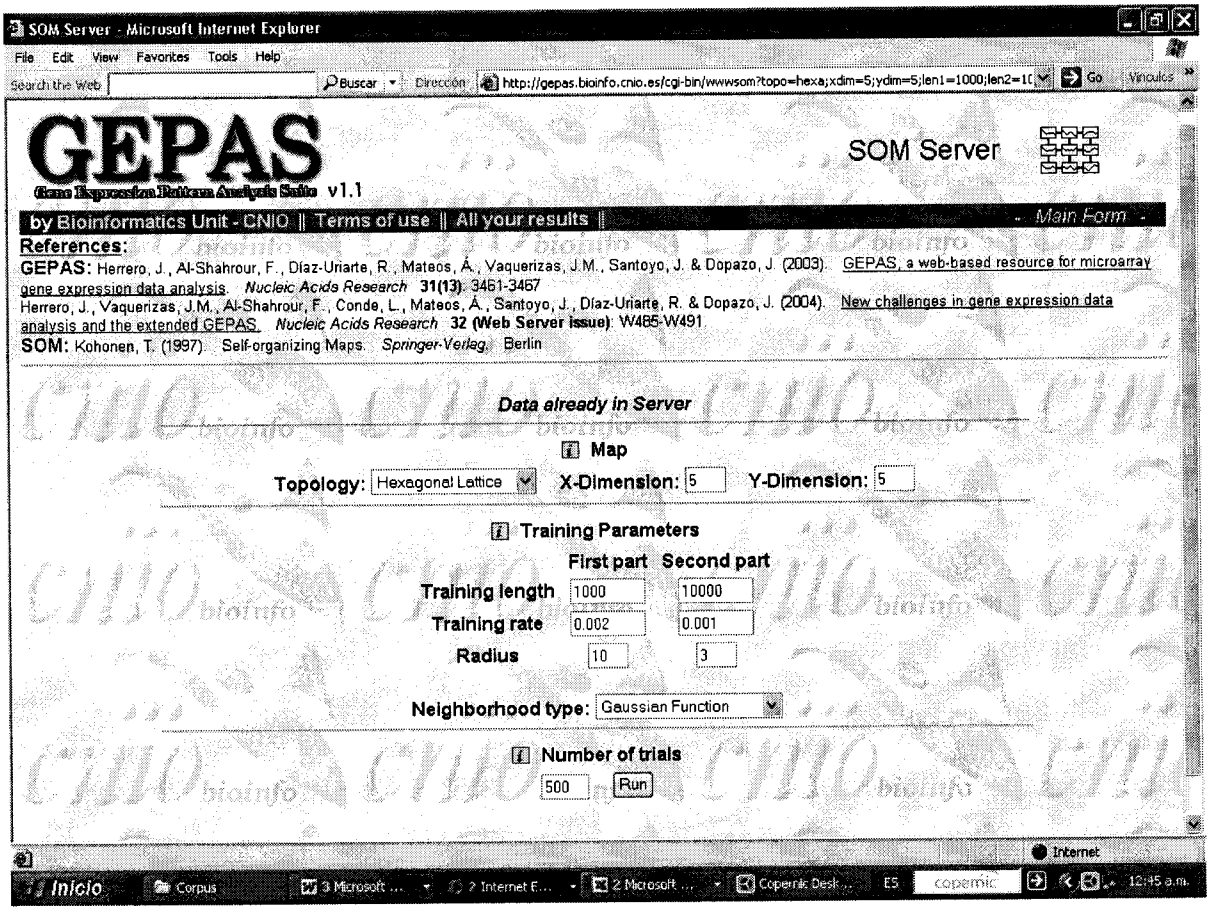
- **Data:** Matriz de pesos obtenida como resultado de la codificación del conjunto de documentos obtenidos en la Web como resultado de una consulta con un vector de 5 palabras.

- **Topology:** El tipo de topología usado en el mapa. Cada topología tiene un forma diferente de conectividad en una topología hexagonal cada neurona tiene 6 vecinos y en una topología de tipo rectangular sólo 4, por lo tanto está demostrado teóricamente que una topología hexagonal es más eficiente.

- **X-Dimension: Es el número de neuronas en la dimensión horizontal.**

- **Y-Dimension:** Es el número de neuronas en la dimensión vertical.

Figura 3.2.2.a. Pantalla de captura de datos para el modelo SOM SERVER.



El entrenamiento tiene dos etapas:

1.- Fase de ordenamiento

2.- Entrenamiento fino.

- **Training length**: Es el número de pasos en cada fase. Usualmente este número es mayor en la segunda etapa.

- **Training rate:** Esta es la tasa de aprendizaje. Se decrementa de forma lineal y tiende a cero durante el entrenamiento. Este parámetro es grande en la etapa de ordenamiento.

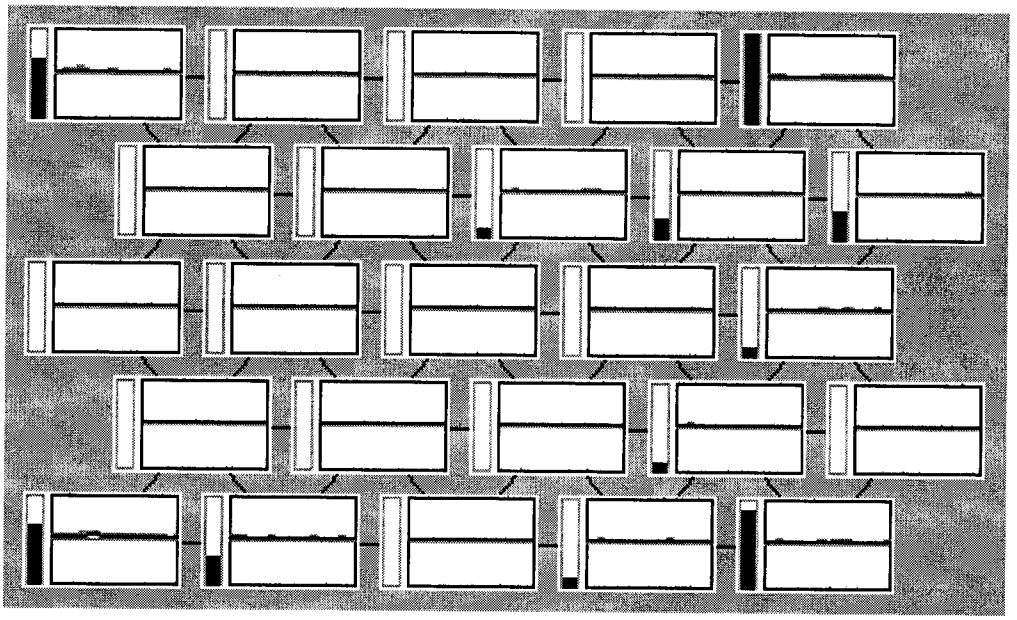
- **Radius**: Este es el radio inicial del área de entrenamiento. Decrece linealmente y tiende a uno durante la etapa del entrenamiento. Usualmente el radio de la primera parte es casi igual al diámetro del mapa mientras que típicamente un valor de 3 es usado en la parte del entrenamiento fino.

- **Neighborhood type:** Ésta es la función de vecindad la cual en nuestro experimento es la función Gaussiana.

- **Number of trials:** Índica el número de ciclos para la función recursiva que define el mapa.

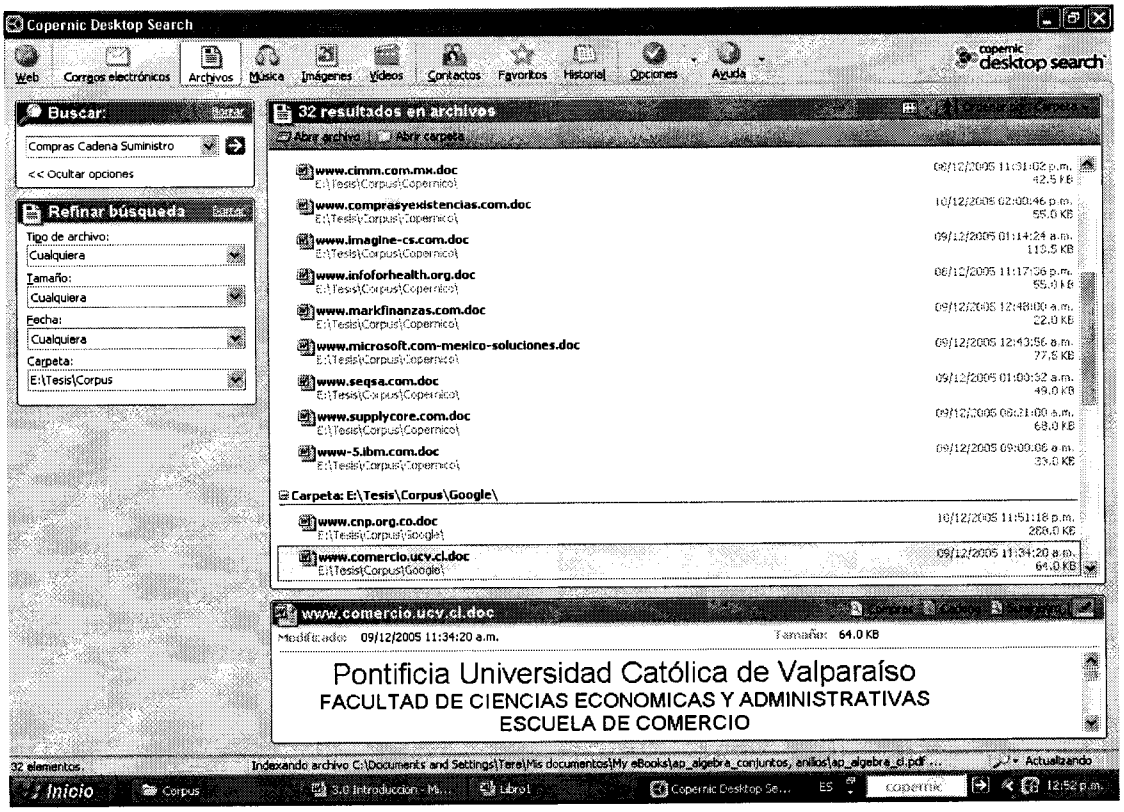
Una vez ingresados los parámetros en el algoritmo debemos interpretar los resultados:

Figura 3.2.2.b. Mapa hexagonal de clusters



Como podemos observar en el mapa existen tres conglomerados y dos categorías de clasificación posibles, en ese sentido hagamos una selección de un conjunto de archivos para presentárselos al usuario. Éste es el papel que jugaría el agente intermedio. Para nuestro experimento simularemos como agente intermedio el programa Copernic Desktop Search. Presentándonos los siguientes resultados:

Figura 3.2.2.c. Pantalla Principal de Búsqueda en Copernic Desktop Search.



Ahora presentaremos una porción de los archivos seleccionados por el usuario[4]:

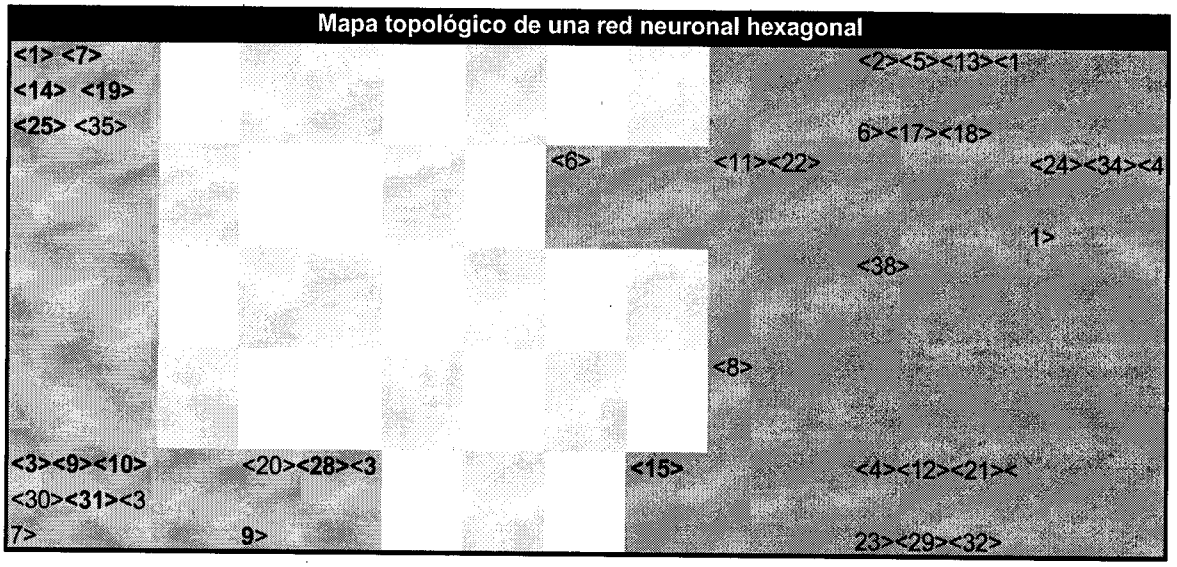
Figura 3.2.2.d. Resultado de selección de archivos por parte del usuario.

Indice	Selección de Documentos Relevantes	
1	*E:\Tesis\Corpus\Copernico\Conferencia del CIES.doc*	✔
2	*E:\Tesis\Corpus\Copernico\mexico.york.com.doc*	✘
3	*E:\Tesis\Corpus\Copernico\segmento.itam.mx.doc*	✔
4	*E:\Tesis\Corpus\Copernico\www.alimarket.es.doc*	✘

[4] La tabla completa se incorpora como el anexo VII.

5	E:\Tesis\Corpus\Copernico\www.aprocal.org.mx.doc	✗
6	E:\Tesis\Corpus\Copernico\www.areaminera.com.doc	✗
7	E:\Tesis\Corpus\Copernico\www.centroccc.com.doc	✓
8	E:\Tesis\Corpus\Copernico\www.centrologistico.cl.doc	✗
9	E:\Tesis\Corpus\Copernico\www.cimm.com.mx.doc	✓
10	E:\Tesis\Corpus\Copernico\www.comprasyexistencias.com.doc	✓
13	E:\Tesis\Corpus\Copernico\www.imagine-cs.com.doc	✗

Figura 3.2.2.e. Gráfico de Documentos seleccionados, a través de la red neuronal de Kohonen .



Porporción de documentos seleccionados en la clase Ganadora:	**0.688**	69%
Porporción de documentos seleccionados en otras clases:	**0.313**	31%

Como se muestra en la figura 3.2.2.e., podemos dividir el mapa en dos clases, una en el costado izquierdo y otra en el costado derecho, la clase ganadora es donde más documentos se encuentran representados es decir la del lado izquierdo, en ese sentido podremos pedir a nuestros agentes de minado que traigan más documentos de la misma ruta donde encontraron la categoría mejor clasificada y de esta manera iniciaríamos la segunda etapa de búsqueda.

Existen documentos que fueron seleccionados y no se encuentran dentro del mapa, esto es muy importante porque debido a la dimensión que asignamos al mapa tuvimos pérdida de información, sin embargo, para nuestro modelo consideraremos que dichos documentos pertenecen a una clase de las no ganadoras y, bajo este supuesto, la proporción de recuperación para la clase ganadora estará dada por 11/16= 0.688 y la clase no ganadora como 5/16= 0.313

Bien, ahora ya sabemos por dónde buscar y el siguiente paso es ir por más información.

Es un buen momento para hacer un alto en el camino y evaluar qué hemos logrado hasta ahora: Nuestro escenario presentó la búsqueda de cualquier usuario de la web que necesita información en ese sentido introdujo un vector de consulta con 3 términos (Compras Cadena Suministro), el sistema le entregó un conjunto de 32 documentos de un total de 41 encontrados en la Web y previamente clasificados. Acto seguido el usuario calificó esos 32 documentos y encontramos con la ayuda del algoritmo de Kohonen que la preferencia del usuario estaba bien definida en uno de los dos grandes conglomerados mostrados por el mapa neuronal. Hasta este momento nosotros contamos con los datos suficientes para proporcionar mayor información al usuario respecto a documentos que tenemos dentro de la categoría ganadora. Para la presentación del siguiente conjunto de documentos, usaremos las proporciones encontradas en el proceso anterior, de esta forma se presentarán el 69% de los documentos que recaben los agentes de minado pertenecientes a la categoría ganadora y el 31% de la otra.

De esta forma los agentes de minado seguirán buscando en las rutas conocidas, hasta que el usuario cambie de vector de consulta.

3.2.3 Generación del Estadístico Kappa.

Como vimos en el punto anterior hemos logrado optimizar la búsqueda de información mediante la clasificación de sus ámbitos; ahora supondremos un escenario algunos días después, donde el usuario buscar información referente a software de logística y recuerda que existen páginas relacionadas al tema y al vector de palabras Compras, Cadena Suministro, en ese sentido llega a su equipo y realiza la consulta exactamente igual que la realizada anteriormente. ¿El sistema tendría que realizar el proceso nuevamente?, La respuesta es no; el sistema ya tiene en memoria todos los datos de esta búsqueda y lo único que haría es presentar el mismo conjunto de resultados de la vez anterior. Sin embargo, el usuario ahora tiene otros intereses; ¿cómo puede el sistema saber que los intereses del usuario han cambiado?, Bueno pues verificando en la categoría en la que caen sus preferencias actuales, pero cómo saber si esta nueva selección no obedece únicamente al azar y él está interesado en el mismo tema de la consulta anterior y no recuerda cuáles documentos son los que tienen la información relevante; bien pues un factor de concordancia entre criterios es el estimador Kappa el cual ya abordamos en el capítulo anterior y que en este momento aplicaremos como una herramienta de ayuda en nuestro experimento:

Figura 3.2.2.f. Resultado de selección de archivos por parte del usuario 2ª Etapa[5].

Indice	Selección de Documentos Relevantes	
1	E:\Tesis\Corpus\Copernico\Conferencia del CIES.doc	✗
2	E:\Tesis\Corpus\Copernico\mexico.york.com.doc	✗
3	E:\Tesis\Corpus\Copernico\segmento.itam.mx.doc	✗
4	E:\Tesis\Corpus\Copernico\www.alimarket.es.doc	✗
5	E:\Tesis\Corpus\Copernico\www.aprocal.org.mx.doc	✗
6	E:\Tesis\Corpus\Copernico\www.areaminera.com.doc	✗
7	E:\Tesis\Corpus\Copernico\www.centroccc.com.doc	✗
8	E:\Tesis\Corpus\Copernico\www.centrologistico.cl.doc	✗
9	E:\Tesis\Corpus\Copernico\www.cimm.com.mx.doc	✓
10	E:\Tesis\Corpus\Copernico\www.comprasyexistencias.com.doc	✗
13	E:\Tesis\Corpus\Copernico\www.imagine-cs.com.doc	✓

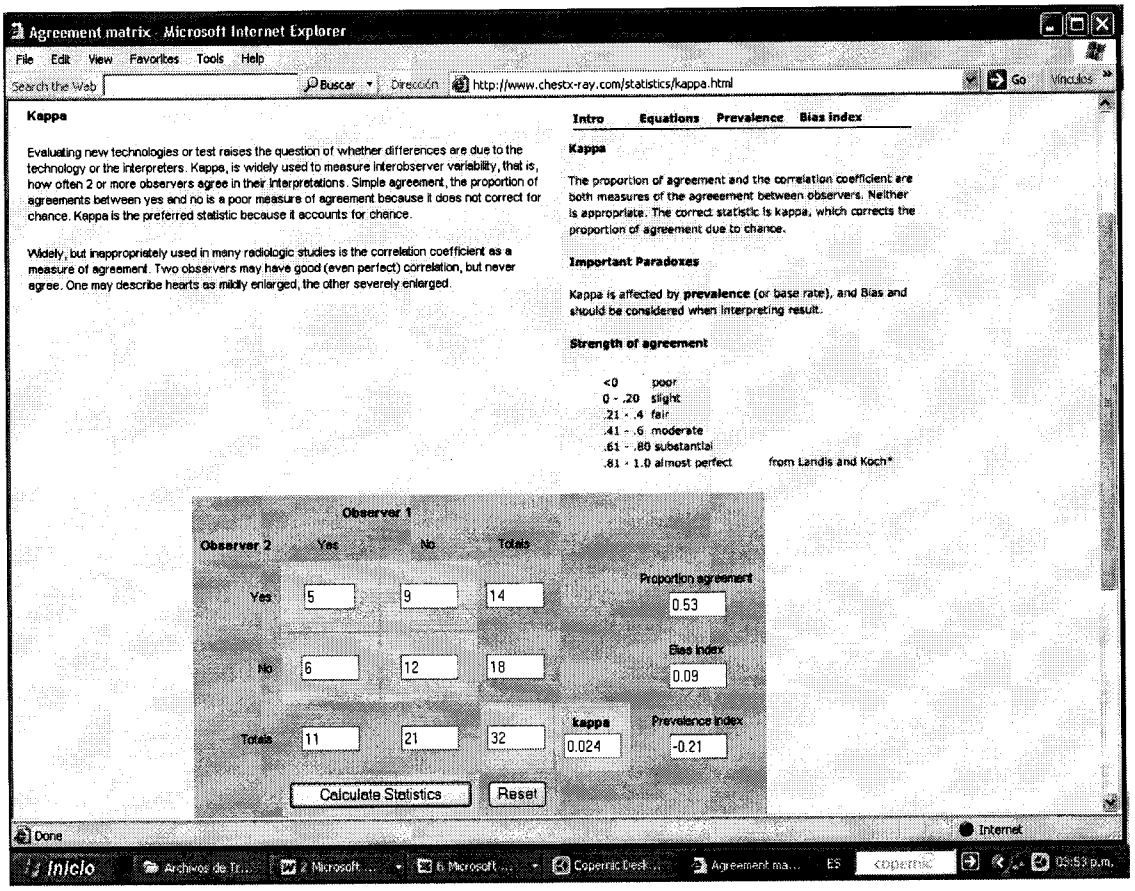
Al igual que en la selección anterior dentro de la tercer columna de la tabla, la paloma marca la aceptación y la cruz el rechazo, con el agregado del color o tono de grís que en el caso de ser azul o más obscuro para las palomas indica la coincidencia con la selección anterior, de igual forma que las cruces rojas; para el caso de que las cruces sean azules o las palomas rojas marcan discrepancia con la selección anterior.

Con los resultados de la tabla anterior, tenemos los datos necesarios para determinar si los criterios de evaluación prevalecen o cambian más allá del azar, es decir, si contrastamos los documentos que fueron seleccionados de forma común en ambas búsquedas mediante el estimador de concordancia Kappa, partiendo de que estamos evaluando el mismo conjunto de documentos entonces, nos encontramos frente a una prueba de congruencia intra-observador. Es decir, tomando el símil de un estudio radiológico, un radiólogo observa el mismo conjunto de radiografías dos instantes de tiempo diferentes; si el estimador Kappa nos indica desacuerdo entre ambas observaciones quiere decir que el radiólogo está evaluando con criterios diferentes el conjunto de imágenes, y en consecuencia hay una inconsistencia de criterios. Así si un método de medición en su primer evaluación da una cifra o cantidad significativamente diferente a una segunda medición entonces estamos ante un caso de inconsistencia en el instrumento de medición. No obstante para nuestro caso ante lo que estamos es un cambio de preferencias de nuestro usuario y el estimador Kappa nos apoya para demostrar que existe un cambio de intención en el tema de búsqueda.

[5] [5] La tabla completa se incorpora como el anexo VIII.

Ahora procederemos a obtener el Kappa no sin antes asegurarnos que existe independencia estadística entre las calificaciones de cada una de las búsquedas[6].

Figura 3.2.2.g. Cálculo de Kappa http://www.chestx-ray.com/statistics/kappa.html.



Como podemos ver en la pantalla anterior en la cual se genera un Kappa igual a 0.024 el grado de acuerdo es muy escaso, lo cual es coherente pues estamos hablando de temas circunscritos en el mismo ámbito; pero que, no obstante son diferentes y el usuario ha decidido cambiar de tema.

El estimador puntual acotado en la tabla de evaluación, nos permite ofrecer una clara evidencia del cambio de preferencia por parte del usuario. Ahora para que podamos hablar de inferencia, deberemos establecer un intervalo de confianza o realizar una prueba de hipótesis.

[6] Los cálculos de la independencia entre las variables de la tabla de contingencias de nuestro experimento se muestran en el anexo VI

3.2.4 Prueba de Hipótesis.

La interpretación que podemos dar al resultado del estimador es que existe una relación de concordancia leve entre ambas selecciones de documentos es decir que nuestro usuario cambió de criterio de búsqueda, lo cual indica a nuestros agentes de minado de información el nuevo camino a seguir, sin embargo un estimador puntual es estadísticamente insuficiente evidencia para tomar una decisión. Sabemos que el estimador Kappa proviene de un experimento binomial por lo tanto puede llegar a obtenerse un estadístico de contraste el cual se use en una distribución normal estandarizada si las condiciones del tamaño de muestra y valor del estadístico sean propicias, en nuestro caso el valor de Kappa no se aproxima demasiado a cero ni a uno y existe una regla empírica que nos permite trabajar con una muestra de un experimento binomial como una normal, siempre y cuando el tamaño de la muestra sea mayor o igual a 30.

$$Z = \frac{|\hat{\kappa} - \kappa|}{\hat{\sigma}_\kappa} \qquad \sigma_\kappa^2 = \frac{A + B - C}{n(1-\pi_e)^4}$$

A=	B=	C=	π_e=	π_0 = 0.53125
0.0222349513	0.4668154651	0.2327768302	0.51953125	

Z_k=	$\hat{\sigma}_\kappa$=	$\hat{\kappa}$=	k_0=0.2	$n = 32$
0.942837509	0.1862566502	0.024390244		

k_0 = 0.2 es límite dentro de la clasificación del estadístico Kappa para mostrar acuerdo leve, en ese sentido si tomamos bajo una probabilidad de error tipo I α=0.01 las siguientes pruebas de hipótesis:

$$H_0 = \hat{k} <= k_0 \qquad \text{vs.} \qquad H_a = \hat{k} > k_0$$

Con 0.01 de probabilidad $Z_{0.01}$ = 2,326 es decir:

$$Z_k = 0.94283 < Z_{0.01} = 2.326$$

El estimador cae en la región de aceptación de la hipótesis nula, es decir que podemos afirmar con un 99% de certeza que el estimador kappa se mantendría por debajo de 0.2 del total de muestras y esto confirma que el usuario ha cambiado de opinión al mostrar acuerdo escaso respecto a la búsqueda anterior[7].

[7] En los Anexos IV y V se presentan los cálculos complementarios respecto a la prueba de hipótesis expuesta en este capítulo.

Conclusiones:

Como se demuestra en nuestro experimento, las herramientas tanto del algoritmo de Kohonen como del índice Kappa, han resultado para conseguir nuestros propósitos, es decir, nos han ofrecido una alternativa que funciona de manera confiable para clasificar un conjunto de documentos y evaluar las preferencias de nuestro usuario a través del tiempo. Ahora se expondrán algunas observaciones fundamentales que debemos dejar en claro si queremos que nuestro modelo se materialice en un sistema multiagentes.

La elección del modelo vectorial como método de preprocesamiento obedeció a un principio de parsimonia respecto a la alternativa probabilista, y a su compatibilidad con el algoritmo genético de clasificación, no obstante, éste fue usado parcialmente. Por lo que es posible mejorar su desempeño, sí se introduce el algoritmo completo o una variante del mismo, es decir, nos permitimos proponer estas alternativas como tema de futuros trabajos de tesis, que ayuden a mejorar la rentabilidad de los recursos invertidos en dicho algoritmo y el beneficio en la clasificación final del genético.

Dentro del algoritmo de Kohonen recomendamos para la evaluación de las distancias entre las neuronas, la función gaussiana respecto de la euclidiana, por dar una mejor representación de la dispersión en el caso de gráficas hexagonales. Además, se demostró la importante aportación del algoritmo para clasificar los documentos de búsqueda, sin la necesidad de una definición previa de clases.

Para insertar el algoritmo de Kohonen, como un sensor de clasificación autónomo dentro de un sistema de búsqueda en la red. Es necesario desarrollar un mecanismo que discrimine las áreas generadas dentro de su mapa y, que a su vez ofrezca de manera automática la categoría dominante. He aquí otro interesante tema de investigaciones futuras.

En el proceso de prueba de hipótesis del estadístico Kappa, es muy importante que tomemos en cuenta el tamaño de la muestra. En ese sentido, debemos dotar al sistema de una función muestral normal estándar. Un criterio empírico útil, es el empleo de muestras de veinticinco a treinta elementos. No se recomiendan muestras más grandes, debido a que el usuario puede verse abrumado si lo sometemos a la calificación de una cantidad elevada de documentos.

Nuestra propuesta, conjunta hallazgos de múltiples trabajos que están actualmente publicados y disponibles en la Red, sin embargo, al momento de llevar a cabo la programación del modelo en un sistema propio, podemos encontrarnos con dificultades de costo, incompatibilidad o escasez de las bibliotecas o códigos del software que tienen las herramientas seleccionadas; en este sentido, si resulta necesario programar el algoritmo de Kohonen, recomendamos el uso del índice Kappa para su prueba, antes de ponerlo en producción. Es decir, aplicar una evaluación de concordancia intra-observador para demostrar la consistencia del método.

El principal cuidado a tomar en cuenta al momento de desarrollar las rutinas de preprocesamiento, son los recursos. Se recomienda que el número de palabras a evaluar por documento no exceda de 2000. Estamos trabajando con páginas Web y los agentes no tienen que recabar más información de la necesaria. La decisión del umbral de palabras permitidas para el preprocesamiento no es trivial, y se sugiere, tomar como referencia el rendimiento del equipo de cómputo con que contemos.

Finalmente, el desempeño del algoritmo de codificación de documentos, depende en gran medida de la definición del conjunto de palabras irrelevantes, como: pronombres, adverbios, preposiciones, artículos, etc. Con base en nuestro objetivo este filtro, elimina las palabras que no deberán ser evaluadas para el conteo de frecuencias. Un ejemplo muy claro es el siguiente: en nuestro experimento, el propósito es encontrar las palabras clave con mayor relevancia en cada documento, para lo cual debimos eliminar palabras que no tienen un significado semántico. Sin embargo, si cambiáramos nuestro objetivo, y necesitamos comprobar si un conjunto de documentos proviene de un mismo escritor, entonces las palabras que involucren un sentido léxico, sintáctico o tiempo verbal son prioritarias y no formarían parte del filtro; puesto que la frecuencia de su uso, es muy importante para definir el estilo literario de un autor.

[1] Abraira, V. (Julio 2003). **El índice kappa**. España: Madrid, Colmenar, km 9,100. 28034, Unidad de Bioestadística Clínica. Hospital Ramón y Cajal. victor.abraira@hrc.es SEMERGEN: 2000; 27: 247-249.

[2] Agresti, A. (1984). Analysis of ordinal categorical data. EUA. John Wiley & Sons.

[3] Amat, Carlos Benito. (Junio 2003). **Recuperación en Internet: cuatro modelos complementarios y una agenda para su integración.** Boletín de la Red Iris". vol. 48 (99), pp., http://www.rediris.es/rediris/boletin/48/enfoque2.html

[4] Armstrong, R., Freitag, D., Joachims, T., y Mitchell, T. (1995). **WebWatcher: A Learning Apprentice for the World Wide Web.** Stanford, California. EE.UU. Proceedings of 1995 AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments,

[5] Balabanovic, M., Shoham, Y., y Yun, Y. (1995). **An adaptive agent for automated web browsing**, Journal of Visual Communication and Image Representation, Vol. 6, No. 4.

[6] Beerud, D. (1994). A Learning Approach to Personalized Information Filtering. EUA: MIT, MSC thesis, Dept. of Electrical Engineering and Computer Science.

[7] Bordignon, F., Tolosa, G. (Julio 2003). **Gnutella: Sistema Distribuido para Almacenamiento y Búsqueda de Información. Descripción del Modelo.** Argentina: Universidad Nacional de La Plata. Reporte de Investigación. Maestría en Redes de Datos Facultad de Informática.

[8] Bray, T., Paoli, J., Sperberg-McQueen, C.M., y Maler, E. (2000). **Extensible Markup Language (XML) 1.0 (Second Edition).** W3C Recommendation, World Wide Web Consortium, http://www.w3.org/TR/2000/REC-xml-20001006. (accedido 14/6/2002).

[9] Brewington, B., Cybenko, G. (2000) **How Dynamic is the Web?**. Proceedings of the Ninth International World Wide Web Conference.

[10] Cohen, J. (1960). A coefficient of agreement for nominal scales. EUA: Educ. Psychol Meas *20:37-46*

[11] Colle, Raymond; (2001). Agentes Autónomos o Softbots. Fragmento de Tesis Doctoral, México: UNAM.

[12] Davison, B.D. (2000a). **Recognizing Nepotistic Links on the Web.** Austin, Texas, EE.UU. Proceedings of AAAI-2000 Workshop on Artificial Intelligence for Web Search.

[13] Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., y Slattery, S. (1998). **Learning to Extract Symbolic Knowledge from the World Wide Web.** Madison, Wisconsin, EE.UU. Proceedings of the 15th National Conference on Artificial Intelligence (AAAI98),

[14] Erdmann, M., Maedche, A., Scnurr, H.P., y Staab, S. (2001). **From Manual to Semiautomatic Semantic Annotation: About Ontology-based Text Annotation Tools.** ETAI Journal - Section on Semantic Web (Linköping Electronic Articles in Computer and Information Science), 6

[15] Etzioni, O. (Agosto 1995). **Intelligent Agents on the Internet**. IEEE Expert, pp.44-49,

[16] Finney, D. J. (1971). Probit Analysis. EUA: Cambridge University Press. Tercera Edición.

[17] Fensel, D., Angele, J., Decker, S., Erdmann, M., Schnurr, H.P., Staab, S., Studer, R., y Witt, A. (1999). **On2broker: Semantic-based access to information sources at the WWW**. Hawai, EE.UU. Proceedings of the World Conference on the WWW and Internet (WebNet 99),

[18] Fraser, D. A. S. (1979). Inference and Linear Models. EUA: McGraw Hill.

[19] Garduño Torres, G., Sierra Martínez, G., Medina Urrea, A. (Junio 2003). **Herramientas de análisis para el Corpus Lingüístico en Ingeniería.** México: UNAM. Grupo de Ingeniería Lingüística Instituto de Ingeniería, 3er cubículo de la Torre de Ingeniería, Circuito Interior, Ciudad Universitaria.

[20] Gruber, T.R. (1993). **Toward principles for the design of ontologies used for knowledge Sharing.** Padua, Italia. Proceedings of International Workshop on Formal Ontology.

[21] Jaramillo López, Judith (2001). Técnicas para la recuperación de información y búsqueda de texto en bases de datos relacionales, Tesis de La Maestría en Ciencias de Ingeniería de la Computación, México: UNAM.

[22] Kazimier, Leonard J. (1988). Estadística Aplicada a la Administración y a la Economía. México: McGraw Hill.

[23] Kluwer Academic Publishers (Junio 2003). **Agents, Reasoning and Dynamics**. Vol. 6 in Series of Handbooks in Defeasible Reasoning and Uncertainty Management Systems.

[24] Lagus, Krista (2000) Text Mining with the WEBSOM, Finland: Acta Polytechnica Scandinavica, Mathematics and Computing Series No. 110, Espoo 2000, 54 pp. Published by the Finnish Academies of Technology. ISBN 951-666-556-X. ISSN 1456-9418. UDC 004.032.26:025.4.03:004.5

[25] Lassila, O., Swick, R.(1999). **Resource Description Framework (RDF) Model and Syntax Specification.** W3C Recommendation, World Wide Web Consortium, http://www.w3.org/TR/REC-rdf-syntax (accedido 14/6/2002).

[26] Lawrence, S., Giles, C. L. (Agosto 2003). **Searching the World Wide Web**. EUA: Science 1998, vol.280, pp.98-100,

[27] Lieberman, H. (1995). **Letizia: An Agent That Assists Web Browsing**. Montreal, Quebec, Canada. Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, ,

[28] Lloyd, C.J. (1999). Statistical Analysís of Categorical Data. EUA: Wiley.

[29] Mendenhall, W., Scheafer, R. L., Wackerly, D. (1986). Estadística Matemática con Aplicaciones. México, Grupo Editorial Iberoamérica.

[30] Menczer, F., Belew, R.K., y Willuhn, W. (1995). **Artificial Life Applied to Adaptive Information Agents**. Stanford, California, EE.UU. Proceedings of AAAI Spring Symposium on Information Gathering,.

[31] Molinero, Luis M. (Junio 2003). **Métodos estadísticos de clasificación.** España: www.seh-lelha.org/stat1.htm

[32] Moreiro González, José A. (Agosto 2003). **Aplicaciones al análisis automático del contenido provenientes de la teoría matemática de la información**. España: Universidad Carlos III de Madrid, Depto. de Biblioteconomía y Documentación, Anales de documentación, 2002 n.º 5, págs. 273-286.

[33] Moreno Romero, Francisco Javier (Junio 2003) **Pasado presente y futuro de la recuperación de información en Internet.** Congreso Internacional sobre Retos de la Alfabetización tecnológica en un mundo en red. vol. 1, n. 2000), pp. http://168.143.67.65/congreso/ponencias/ponencia-4.pdf

[34] Moreno, A., López Moreno, S., Corcho Berdugo, Alexánder (Agosto 2003). **Salud pública de México** julio-agosto 2000 / vol.42, no.4 México: UNAM, Facultad de Medicina.

[35] Moukas, A. (1996). **Amalthaea: Information Discovery and Filtering using a Multiagent Evolving Ecosystem.** Londres, Reino Unido. Proceedings of the Conference on Practical Applications of Agents and Multiagent Technology.

[36] Muruzabal, J., Velez Langs, O. (Junio 2003) **Mapas Auto-organizados de Kohonen** España: Universidad Rey Juan Carlos, Computación neuronal y evolutiva jorge.muruzabal@urjc.es oswaldo.velez.langs@urjc.es

[37] Pazzani, M., Muramatsu, J., y Billsus, D. (1996), **Syskill & Webert: Identifying interesting web sites**. Portland, Oregon, EE.UU. Proceedings of the American National Conference on Artificial Intelligence (AAAI'96),

[38] Pérez Coutiño, M. A., López López, A. (Octubre 2005) **Un Agente Asistente para la Creación y Mantenimiento de Sitios Web** México: Instituto Nacional de Astrofísica, Óptica y Electrónica Departamento de Ciencias Computacionales. mapco@susu.inaoep.mx, allopez@inaoep.mx

[39] Pérez Lezama V.C. (1998) Agentes Móviles en Bibliotecas Digitales México: Universidad de las Américas Puebla, Tesis de Maestría, Departamento de Ingeniería en sistemas Computacionales.

[40] Poggy, A. (Junio 2003). **Daisy: an Object-Oriented System for Distributed Artificial Intelligence**. In Proceedings of the ECAI-94 Workshop on Agent Theories, Architectures, and Languages. 1994.

[41] Reyes Castañeda, Pedro (1995) Bioestadística Aplicada, México Trillas.

[42] Russell, S., Norvig, M. (1995). Inteligencia Artificial Un enfoque Moderno México: Printece Hall.

[43] Siegel, S., Castellan Jr. N. J. (1998). Nonparametric Statistics for the Behavioral Sience. EUA: McGraw Hil, Segunda Edición.

[44] Tramullas Saz, J. (jueves 23 de marzo de 2006). **De Crusoe y los nuevos exploradores**. Los softbots y la lectura futura. Puertas a la lectura, 5, 1998, 53-58 *http://tramullas.com/papers.html*

[45] Weerasooriya, D., Rao, A., Ramamohanarao. K. (1995). Design of a Concurrent Agent-Oriented Language. EUA, In Wooldridge, M.; Jennings, N. Eds. Intelligent Agents (LNAI 890).

[46] Zellerman, D. (1999). Models for Discrete Data. EUA, Oxford University Press,.

[47] Zhang, T. (Agosto 1997). **Intelligent Agents for Information Retrieval in Internet**. Canada: Calgary, Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing, pp. 327-330.

A continuación se presentan las el código de las funciones en Excel que fueron desarrolladas para la codificación de los documentos a través de vectores característicos.

```
Sub FilesWord()
Dim WordFile As String
Dim s, i As Integer

Windows("Macros TFIDF ver 0.1.xls").Activate
Set fs = Application.FileSearch
For s = 1 To 2
SubDir = IIf(s = 1, "Copernico", "Google")
With fs
   .LookIn = "E:\Tesis\Corpus\" & SubDir
   .FileType = msoFileTypeWordDocuments
   If .Execute > 0 Then
      MsgBox "There were " & .FoundFiles.Count & " Word file(s) found By Agent " & SubDir & "."
      Sheets("Global").Select
      Range("G2").Select
      ActiveCell.Value = ActiveCell.Value + .FoundFiles.Count
      For i = 1 To .FoundFiles.Count
         WordFile = .FoundFiles(i)
         TFIDF (WordFile)
      Next i
   Else
      MsgBox "There were no Binder files found."
   End If
End With
Next
  Call Filtro10(30)
  Call Documentos
  Call FiltroTF_MAX
  Call Selección
End Sub


Sub TFIDF(ByVal WordFile As String)
'
' Macro grabada el 06/12/2005 por Aaron
'
Dim Wordobj As Object    ' Declara la variable para mantener la referencia.
Dim DocumentName, W As String
Dim dTotalWords, i, j As Integer
Dim aWord As Variant
Dim BagWords() As Variant
DocumentName = WordFile
'DocumentName = "E:\Tesis\Corpus\google\" & "mit.ocw.universia.net" & ".doc"
Set Wordobj = GetObject(DocumentName, "Word.document")
Wordobj.Activate
dTotalWords = IIf(Wordobj.Words.Count > 2000, 2000, Wordobj.Words.Count)
iDocumentName = Wordobj.Name
ReDim BagWords(1 To dTotalWords, 1 To dTotalWords)
i = 1
j = 1
With Wordobj.Application.WordBasic
   For Each aWord In Wordobj.Words
      BagWords(j, 1) = DocumentName
      BagWords(j, 2) = Trim(aWord.Text)
      If j = 2000 Then Exit For
      j = j + 1
   Next aWord
End With
Wordobj.Application.Quit   ' Cuando finaliza, utiliza el método Quit para cerrar
Set Wordobj = Nothing   ' la aplicación; después libera la referencia.
```

```
Windows("Macros TFIDF ver 0.1.xls").Activate
Sheets("TFIDF").Select
Sheets("TFIDF").Activate
Range("A2").Select
If ActiveCell.Value <> Empty Then Selection.End(xlDown).Select
  n = 1
  For i = 1 To dTotalWords
    W = Trim(LCase(BagWords(i, 2)))

    If Left(W, 1) <> "□" And Right(W, 1) <> "□" And Val(W) <= 0 And Len(W) > 2 And W <> "0" And W <> "" And W
<> "más" And W <> "también" And W <> "antes" And W <> "con" And W <> "entonces" And W <> "ante" And _
      W <> "es" And W <> "se" And W <> "no" And W <> "si" And W <> "por" And W <> "para" And W <> "de" And
W <> "sobre" And _
      W <> "francés" And W <> "inglés" And W <> "español" And W <> "chino" And W <> "alemán" And W <>
"italiano" And W <> "portugues" And _
      W <> "antes" And W <> "después" And W <> "Luego" And W <> ".-" And W <> ")." And W <> ".," And _
      W <> "-> " And W <> "de " And W <> "allí" And W <> "allá" And W <> "poco" And W <> "mucho" And _
      W <> "aquí" And W <> "algún" And W <> "alguna" And W <> "ello" And W <> "..." And _
      W <> "muy" And W <> "pues" And W <> "puede" And W <> "etapa" And W <> "son" And _
      W <> "han" And W <> "uso" And W <> "con" And W <> "sin" And W <> "través" And W <> "tras" And _
      W <> "caso" And W <> "entre" And W <> "decir" And W <> "amcho" And W <> "largo" And W <> "alto" And _
      W <> ";" And W <> "," And W <> "." And W <> ":" And W <> "_" And W <> "-" And Left(W, 1) <> "?" And _
      W <> "#" And W <> "$" And W <> "%" And W <> "&" And W <> "/" And W <> "!" And _
      W <> "mismo" And W <> "van" And W <> "cuando" And W <> "donde" And W <> "porque" And W <> "fax"
And _
      W <> "|" And W <> "¡" And W <> "¿" And W <> "?" And W <> "=" And W <> "°" And _
      W <> "<" And W <> ">" And W <> "->" And W <> "(" And W <> ")" And W <> "{" And W <> "}" And _
      W <> "[" And W <> "]" And W <> "+" And W <> "*" And W <> "" And W <> "al" And _
      W <> "la" And W <> "las" And W <> "lo" And W <> "los" And W <> "el" And W <> "les" And _
      W <> "etc" And W <> "son" And W <> "en" And W <> "un" And W <> "una" And W <> "unas" And W <> "uno"
And W <> "unos" And W <> "cual" And W <> "ser" And _
      W <> "estará" And W <> "debe" And W <> "además" And W <> "qué" And W <> "sus" And W <> "aún" And W
<> "fue" And W <> "su" And _
      W <> "está" And W <> "este" And W <> "esta" And W <> "esto" And W <> "estas" And W <> "estos" And W
<> "ellas" And W <> "ellos" And _
      W <> "aquellas" And W <> "aquellos" And W <> "otros" And W <> "otras" And W <> "aquel" And W <> "quién"
And W <> "que" And W <> "así" And W <> "como" And _
      W <> "–" And W <> Chr(13) And W <> "???" And W <> "??" And W <> "del" And W <> "vez" And W <>
"todos" And W <> "cada" And W <> "ninguno" And W <> "://" And _
      W <> "puede" And W <> "muy" And W <> "nos" And W <> "hecho" And W <> "edi" And W <> "cómo" And W
<> "día" And W <> "mes" And W <> "año" And W <> "vez" And W <> "lunes" And W <> "martes" And W <>
"miércoles" And W <> "jueves" And W <> "viernes" And W <> "sábado" And W <> "domingo" _
    Then
      ActiveCell.Value = BagWords(i, 1)
      ActiveCell.Offset(0, 1).Value = W
      ActiveCell.Offset(1, 0).Activate
      'n = n + 1
      'If n = 0 Then Exit For
  End If
  Next
  Call pivotTable
End Sub


Sub pivotTable()
'
' Macro grabada el 07/12/2005 por Aaron
'

Dim myRange As Object
Dim rFin As Long

'
  Range("A1:B1").Select
```

```
    Set myRange = Range(Selection, Selection.End(xlDown))
    myRange.Select
    rFin = Str(myRange.Rows(myRange.Rows.Count).Row)

    ActiveWorkbook.PivotCaches.Add(SourceType:=xlDatabase, SourceData:= _
        "TFIDF!R1C1:R" & rFin & "C2").CreatePivotTable TableDestination:="'[Macros TFIDF ver 0.1.xls]Tabla'!R1C1",
TableName:= _
        "Tabla dinámica1", DefaultVersion:=xlPivotTableVersion10
    Sheets("Tabla").Select
    Sheets("Tabla").Activate

     With ActiveSheet.PivotTables("Tabla dinámica1")
        .ColumnGrand = False
        .PreserveFormatting = False
        .RepeatItemsOnEachPrintedPage = False
    End With
    ActiveSheet.PivotTables("Tabla dinámica1").AddFields RowFields:=Array("Agent" _
        , "Word")
    ActiveSheet.PivotTables("Tabla dinámica1").PivotFields("Word").Orientation = _
        xlDataField
    ActiveWorkbook.ShowPivotTableFieldList = True
    ActiveWorkbook.ShowPivotTableFieldList = False
    ActiveSheet.PivotTables("Tabla dinámica1").PivotSelect "", xlDataAndLabel, True
    ActiveSheet.PivotTables("Tabla dinámica1").Format xlReport7
    Application.CommandBars("PivotTable").Visible = False
    ActiveWorkbook.ShowPivotTableFieldList = False

  Call Orden
  Call sinFormato
  Call Enviar


End Sub


Sub Enviar()
'
' Macro grabada el 08/12/2005 por Aaron
'
Dim rFin As Long

    Selection.Cut
    Sheets("Global").Select
    If Cells(2, 1).Value = Empty Then
        Range("A2").Select
    Else
        Range("A2").Select
        Set myRange = Range(Selection, Selection.End(xlDown))
        myRange.Select
        rFin = myRange.Rows(myRange.Rows.Count).Row
        Cells(rFin + 1, 1).Select
    End If

        ActiveSheet.Paste
        Sheets("TFIDF").Select
        Range("A2:B2").Select
        Range(Selection, Selection.End(xlDown)).Select
        Selection.Delete Shift:=xlToLeft


End Sub
```

```
Sub FiltroTF_MAX()
Dim Pivote As String
Dim n, NoDocs As Integer
Dim SumaQ As Double

Windows("Macros TFIDF ver 0.1.xls").Activate
Sheets("Global").Select
Range("A2").Select
NoDocs = ActiveCell.Offset(0, 6).Value

    Set myRange = Range(Selection, Selection.End(xlDown))
    myRange.Select
    rFin = myRange.Rows(myRange.Rows.Count).Row
    Range("A2").Select
    Pivote = Trim(ActiveCell.Value)
    n = 1

    While Pivote <> "" And Pivote <> Empty
      While Pivote = Trim(ActiveCell.Value)
        If n = 1 Then
          TF_MAX = ActiveCell.Offset(0, 2).Value
          ActiveCell.Offset(0, 8).Value = 1
          ren = ActiveCell.Row
          SumaQ = 0
        End If
          ActiveCell.Offset(0, 3).Value = TF_MAX
          ActiveCell.Offset(0, 4).Select
          ActiveCell.FormulaR1C1 = "=0.5+0.5*(RC[-2]/RC[-1])"
          ActiveCell.Offset(0, 1).Select
          ActiveCell.Formula = "=VLOOKUP(B" & Trim(Str(ActiveCell.Row)) & ",TDF!$A$2:$b$2000,2,0)"
          ActiveCell.Offset(0, 2).Select
          ActiveCell.FormulaR1C1 = "=RC[-3]*LOG(" & Str(NoDocs) & "/RC[-2],2)"
          SumaQ = SumaQ + (ActiveCell.Value * ActiveCell.Value)
          ActiveCell.Offset(0, 2).Select
          rSumaQ = IIf(n = 1, "", Trim(Str(-1 * (n - 1))))
          ActiveCell.FormulaR1C1 = "=RC[-2]/R[" & rSumaQ & "]C[-1]"
          ActiveCell.Offset(0, -9).Select
          ActiveCell.Offset(1, 0).Select
          n = n + 1
      Wend
      n = 1
      Cells(ren, 9).FormulaR1C1 = "=SQRT(" & Trim(Str(SumaQ)) & ")"
      Pivote = Trim(ActiveCell.Value)
    Wend
End Sub


Sub Orden()
'
' formaro Macro
' Macro grabada el 08/12/2005 por Aaron
'
  Cells.Select
  Selection.Copy
  Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
    :=False, Transpose:=False
  Range("A1").Select
  Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
    :=False, Transpose:=False
  Range("A2").Select
  Application.CutCopyMode = False
  Selection.Copy
  Range("B3").Select
  Selection.End(xlDown).Select
```

```
    rFin = Trim(Str(ActiveCell.Row))
    Range("A" & rFin).Select
    Range(Selection, Selection.End(xlUp)).Select

    ActiveSheet.Paste
    Application.CutCopyMode = False
    Rows("2:2").Select
    Selection.Delete Shift:=xlUp
    Range("A1").Select
    Selection.End(xlDown).Select
    rFin = Trim(Str(ActiveCell.Row))
    Range("a1", "c" & rFin).Select
    Set myRange = Range(Selection, Selection.End(xlDown))
    myRange.Select

    Selection.Sort Key1:=Range("C2"), Order1:=xlDescending, Header:=xlGuess, _
       OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom, _
       DataOption1:=xlSortNormal

    'Rows(Val(rFin + 2)).Select
    'Selection.Delete Shift:=xlUp

    Rows("1:2").Select
    Selection.Delete Shift:=xlUp

    Range("A1").Select
    Selection.End(xlDown).Select
    rFin = Trim(Str(ActiveCell.Row))
    Range("a1", "c" & rFin).Select


End Sub


Sub sinFormato()
'
' sinFormato Macro
' Macro grabada el 08/12/2005 por Aaron
'
    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
    Selection.Borders(xlEdgeLeft).LineStyle = xlNone
    Selection.Borders(xlEdgeTop).LineStyle = xlNone
    Selection.Borders(xlEdgeBottom).LineStyle = xlNone
    Selection.Borders(xlEdgeRight).LineStyle = xlNone
    Selection.Borders(xlInsideVertical).LineStyle = xlNone
    Selection.Borders(xlInsideHorizontal).LineStyle = xlNone
    Selection.Interior.ColorIndex = xlNone
    Selection.Font.Bold = False
    Selection.Font.ColorIndex = 0
    With Selection.Font
       .Name = "Arial"
       .Size = 8
       .Strikethrough = False
       .Superscript = False
       .Subscript = False
       .OutlineFont = False
       .Shadow = False
       .Underline = xlUnderlineStyleNone
       .ColorIndex = xlAutomatic
    End With

    End Sub
```

```
Sub Documentos()
'
' Documentos Macro
' Macro grabada el 13/12/2005 por Aaron
'
Dim myRange As Object
Dim rFin As Long

Windows("Macros TFIDF ver 0.1.xls").Activate
Sheets("Global").Select
Sheets("Global").Activate
Range("A1").Select

   Set myRange = Range(Selection, Selection.End(xlDown))
   myRange.Select
   rFin = Str(myRange.Rows(myRange.Rows.Count).Row)

'
   Range("A1:C1").Select
   Range(Selection, Selection.End(xlDown)).Select
   ActiveWorkbook.PivotCaches.Add(SourceType:=xlDatabase, SourceData:= _
      "Global!R1C1:R" & rFin & "C3").CreatePivotTable TableDestination:= _
      "'[Macros TFIDF ver 0.1.xls]TDF'!R1C1", TableName:="Tabla dinámica1", _
      DefaultVersion:=xlPivotTableVersion10
   Sheets("TDF").Select
   Sheets("TDF").Activate

    With ActiveSheet.PivotTables("Tabla dinámica1")
      .ColumnGrand = False
      .PreserveFormatting = False
      .RepeatItemsOnEachPrintedPage = False
   End With

   ActiveSheet.PivotTables("Tabla dinámica1").AddFields RowFields:="Palabra"
   ActiveSheet.PivotTables("Tabla dinámica1").PivotFields("Archivo").Orientation _
      = xlDataField
   ActiveWorkbook.ShowPivotTableFieldList = True
   ActiveWorkbook.ShowPivotTableFieldList = False
   ActiveSheet.PivotTables("Tabla dinámica1").PivotSelect "", xlDataAndLabel, True
   ActiveSheet.PivotTables("Tabla dinámica1").Format xlReport9
   Application.CommandBars("PivotTable").Visible = False
   'ActiveWindow.SmallScroll Down:=324
   Cells.Select
   Selection.Copy
   Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
      :=False, Transpose:=False
   'Range("B448").Select
End Sub


Sub Selección()
'
' Macro14 Macro
' Macro grabada el 14/12/2005 por Aaron
'

Windows("Macros TFIDF ver 0.1.xls").Activate
Sheets("Global").Select
Sheets("Global").Activate

   Range("B1:C1").Select
   Range(Selection, Selection.End(xlDown)).Select
   ActiveWorkbook.PivotCaches.Add(SourceType:=xlDatabase, SourceData:= _
      "Global!R1C2:R526C3").CreatePivotTable TableDestination:= _
```

```
        "'[Macros TFIDF ver 0.1.xls]TFIDF_Final'!R2C1", TableName:="Tabla dinámica1", _
        DefaultVersion:=xlPivotTableVersion10

    Sheets("TFIDF_Final").Select
    Sheets("TFIDF_Final").Activate

     With ActiveSheet.PivotTables("Tabla dinámica1")
        .ColumnGrand = False
        .PreserveFormatting = False
        .RepeatItemsOnEachPrintedPage = False
    End With

    ActiveSheet.PivotTables("Tabla dinámica1").AddFields RowFields:="Palabra"
    ActiveSheet.PivotTables("Tabla dinámica1").PivotFields("TF").Orientation _
        = xlDataField
    ActiveWorkbook.ShowPivotTableFieldList = True
    ActiveSheet.PivotTables("Tabla dinámica1").PivotSelect "", xlDataAndLabel, True
    ActiveSheet.PivotTables("Tabla dinámica1").Format xlReport9
    ActiveSheet.PivotTables("Tabla dinámica1").RowGrand = False
    ActiveSheet.PivotTables("Tabla dinámica1").ColumnGrand = False
    Application.CommandBars("PivotTable").Visible = False
    ActiveWorkbook.ShowPivotTableFieldList = False
    Cells.Select
    Selection.Copy
    Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
        :=False, Transpose:=False
    Range("A2:B2").Select
    Range(Selection, Selection.End(xlDown)).Select
    Application.CutCopyMode = False
    Selection.Sort Key1:=Range("B3"), Order1:=xlDescending, Header:=xlGuess, _
        OrderCustom:=1, MatchCase:=False, Orientation:=xlTopToBottom, _
        DataOption1:=xlSortNormal
    Range("A3:A12").Select
    Selection.Copy
    Range("B1").Select
    Selection.PasteSpecial Paste:=xlPasteAll, Operation:=xlNone, SkipBlanks:= _
        False, Transpose:=True
    Range("A2:B2").Select
    Range(Selection, Selection.End(xlDown)).Select
    Application.CutCopyMode = False
    Selection.Delete Shift:=xlToLeft
End Sub
```

A continuación se presenta la tabla de resultados respecto a la codificación del corpus de documentos.

	compras	gestión	logística	procesos	clientes	información	empresas	proveedores	productos	internet
E:\Tesis\Corpus\Copernico\Conferencia del CIES.doc	0	0.066085	0.086779	0	0	0	0	0	0	0
E:\Tesis\Corpus\Copernico\mexico.york.com.doc	0.056657	0	0	0	0	0.04643327	0	0.07912071	0.097081	0
E:\Tesis\Corpus\Copernico\segmento.itam.mx.doc	0	0	0.180246	0.122118	0.08952	0.06407772	0	0.09553796	0.092504	0
E:\Tesis\Corpus\Copernico\www.alimarket.es.doc	0	0	0	0	0	0.0444108	0.08709	0	0	0.087364
E:\Tesis\Corpus\Copernico\www.aprocal.org.mx.doc	0.086179	0	0	0	0	0	0	0	0	0
E:\Tesis\Corpus\Copernico\www.areaminera.com.doc	0	0.086024	0	0	0	0	0.069001	0.07026083	0	0
E:\Tesis\Corpus\Copernico\www.centroccc.com.doc	0.059199	0.066776	0.147681	0	0	0	0	0	0	0
E:\Tesis\Corpus\Copernico\www.centrologistico.cl.doc	0	0.072681	0	0	0	0	0.060458	0	0	0
E:\Tesis\Corpus\Copernico\www.cimm.com.mx.doc	0.069852	0	0.101082	0.184343	0.107302	0	0	0	0.094755	0
E:\Tesis\Corpus\Copernico\www.comprasyexistencias.com.doc	0.115121	0.0908	0.103849	0.099203	0	0	0.068524	0	0	0.096834
E:\Tesis\Corpus\Copernico\www.disheco.es.doc	0	0	0	0	0	0	0	0	0	0
E:\Tesis\Corpus\Copernico\www.exitoexportador.com.doc	0	0	0	0	0	0	0	0	0	0.100387
E:\Tesis\Corpus\Copernico\www.imagine-cs.com.doc	0.100091	0.066849	0	0	0	0	0	0.08408329	0	0
E:\Tesis\Corpus\Copernico\www.infoforhealth.org.doc	0.060235	0	0.095549	0	0.067167	0.04807763	0	0	0.086757	0
E:\Tesis\Corpus\Copernico\www.markfinanzas.com.doc	0	0.071537	0	0	0.059614	0	0.072523	0	0	0
E:\Tesis\Corpus\Copernico\www.microsoft.com-mexico-soluciones.doc	0	0	0	0	0.066084	0.07358188	0	0.09235655	0	0
E:\Tesis\Corpus\Copernico\www.mind.com.co.doc	0	0	0	0	0.092622	0.04972407	0	0.09002362	0.064357	0
E:\Tesis\Corpus\Copernico\www.seqsa.com.doc	0.076378	0	0	0	0.061214	0.04381655	0	0	0	0
E:\Tesis\Corpus\Copernico\www.supplycore.com.doc	0.063357	0	0.088905	0	0	0	0	0	0	0
E:\Tesis\Corpus\Copernico\www-5.ibm.com.doc	0.094472	0.071261	0	0.096373	0	0	0.05835	0	0.069828	0
E:\Tesis\Corpus\Google\amece.org.mx.doc	0	0	0	0	0.062142	0.05150418	0.067334	0	0	0.104122
E:\Tesis\Corpus\Google\cursos.lycos.es.doc	0	0	0	0	0	0	0	0	0	0
E:\Tesis\Corpus\Google\microsites.aprendemas.com.doc	0	0.075036	0	0	0	0	0	0	0	0.09762
E:\Tesis\Corpus\Google\mit.ocw.universia.net.doc	0	0	0	0	0	0.04444378	0	0	0.080595	0
E:\Tesis\Corpus\Google\www.cnp.org.co.doc	0	0.0735	0.084934	0	0	0	0	0	0	0
E:\Tesis\Corpus\Google\www.comercio.ucv.cl.doc	0.105302	0.112527	0	0	0	0.04782455	0	0.092482	0	0
E:\Tesis\Corpus\Google\www.e-global.es.doc	0.059378	0	0	0	0.077552	0.06812779	0.098963	0.09270871	0	0
E:\Tesis\Corpus\Google\www.etc.org.mx.doc	0	0	0	0.119695	0	0	0.073898	0	0.078162	0
E:\Tesis\Corpus\Google\www.gestiopolis.com.doc	0	0	0	0	0.073096	0.05232206	0.068403	0	0	0.099447
E:\Tesis\Corpus\Google\www.hipermarketing.com.doc	0	0	0	0.106886	0.092207	0.07778714	0.067796	0.08660716	0	0.084676
E:\Tesis\Corpus\Google\www.icemd.com.doc	0	0.079793	0	0.098528	0.068988	0	0.066114	0.08173603	0	0.085489
E:\Tesis\Corpus\Google\www.intel.com.doc	0.062399	0	0	0	0	0	0.053039	0.06480961	0	0.069399
E:\Tesis\Corpus\Google\www.intracen.org.doc	0.051692	0.061662	0	0	0	0.04413758	0	0	0.059049	0.083366
E:\Tesis\Corpus\Google\www.isnar.cgiar.org.doc	0	0	0	0	0	0.05086956	0	0	0.063883	0
E:\Tesis\Corpus\Google\www.itc.mx.doc	0	0	0.111474	0	0	0.06335033	0	0	0	0
E:\Tesis\Corpus\Google\www.komintl.com.doc	0.064683	0	0	0	0.067303	0	0	0.08111768	0	0
E:\Tesis\Corpus\Google\www.markfinanzas.com.doc	0	0.122877	0.098605	0.139685	0	0	0	0	0.078446	0
E:\Tesis\Corpus\Google\www.microsoft.com-latam-techne.doc	0	0	0	0	0.067406	0.05474359	0.072782	0	0.080368	0
E:\Tesis\Corpus\Google\www.mor.itesm.mx.doc	0	0	0	0.092629	0	0	0	0	0	0
E:\Tesis\Corpus\Google\www.ntra-net.com.doc	0	0	0	0	0	0.08369192	0.060318	0	0	0.089936
E:\Tesis\Corpus\Google\www.tablero-decomando.com.doc	0	0	0	0	0	0.04066662	0	0	0.060732	0

Anexo III.

A continuación se presenta la tabla de resultados arrojados por el algoritmo de Kohonen.

Representación Vectorial										Representación Neuronal
0.0267198	0.034673	0.0629648	0.0253519	0.0217742	0.0230885	0.0185181	0.0211059	0.0261376	0.0100115	<1><7><14><19><25><35>
0.0301516	0.029057	0.0473196	0.0201274	0.0229593	0.0227581	0.0190388	0.024922	0.0244191	0.0124863	
0.0293869	0.0274611	0.0287923	0.0184227	0.0243373	0.0247039	0.0193153	0.0277165	0.0229815	0.0101298	
0.0310814	0.0222862	0.0190907	0.0130263	0.0248051	0.0281612	0.0191749	0.0337141	0.0249756	0.012277	
0.03437	0.0200472	0.0109472	0.00735559	0.0273912	0.029093	0.0187771	0.0369015	0.0245995	0.0110543	<2><5><13><16><17><18>
0.0303452	0.0325308	0.0472113	0.0296399	0.0241584	0.0233626	0.0244196	0.0227421	0.0260708	0.0158757	
0.0263446	0.0319504	0.0354978	0.0242941	0.0251142	0.0240396	0.0234746	0.0264519	0.0255556	0.0146228	
0.027259	0.0258761	0.0223291	0.0158672	0.0239437	0.0258643	0.0233713	0.0289059	0.0235425	0.0166446	<6>
0.0285998	0.0211822	0.0107663	0.00910364	0.0244993	0.0285515	0.0209023	0.0309883	0.0249398	0.0162797	<11><22>
0.028031	0.0204625	0.0105952	0.0101305	0.0250123	0.0313609	0.019972	0.030955	0.0275343	0.0176573	<24><34><41>
0.0293436	0.0362971	0.0478802	0.0491028	0.0264477	0.0235499	0.0270676	0.0248814	0.0304456	0.0221076	
0.0272692	0.0349199	0.0395885	0.0392741	0.0257432	0.0211133	0.0275415	0.0246377	0.0280298	0.0200935	
0.0265173	0.0293761	0.0241446	0.0276525	0.0243993	0.023089	0.0270171	0.0243466	0.0246355	0.0218333	
0.025702	0.0252112	0.0160946	0.0181643	0.0237631	0.0259257	0.0270933	0.0253142	0.0243684	0.0225018	
0.0257315	0.0208325	0.0100853	0.0109373	0.0236096	0.0298946	0.0268155	0.0252211	0.0243773	0.0243801	<38>
0.0258792	0.0349321	0.0464032	0.0590758	0.0303551	0.02083	0.0322204	0.0257797	0.0319118	0.0260349	
0.0251398	0.0340994	0.0290343	0.0415216	0.0277408	0.0221675	0.0309607	0.0224452	0.028492	0.0259797	
0.0239872	0.0292677	0.0194659	0.0279178	0.0251834	0.0244713	0.0331735	0.0219341	0.0253908	0.0308431	
0.0228376	0.0242966	0.0124078	0.0153879	0.0231094	0.0276147	0.0320184	0.0206806	0.0232	0.0348483	<8>
0.0235325	0.02257	0.0112314	0.0141385	0.0255804	0.0284803	0.0307247	0.0217627	0.0238014	0.0362626	
0.0310585	0.0375698	0.0477162	0.0692614	0.0333203	0.0220409	0.0342486	0.0285819	0.0308186	0.0323011	<3><9><10><30><31><37>
0.0280882	0.0334084	0.0330041	0.0597082	0.0286579	0.0229153	0.0365643	0.0220197	0.029771	0.0284876	<20><28><39>
0.0237508	0.0302399	0.0252814	0.0409578	0.0252063	0.022627	0.0368334	0.0222574	0.0265155	0.0353719	
0.0230479	0.0287831	0.013542	0.0249637	0.0262294	0.0263985	0.0344772	0.0179967	0.0228104	0.0393453	<15>
0.0236148	0.0241298	0.00828556	0.0141829	0.0228057	0.0273455	0.0337831	0.0170223	0.0195939	0.0455333	<4><12><21><23><29><32>

A continuación se presenta la tabla de datos para la realización de la prueba de hipótesis.

Análisis de Clasificación de Documentos			
	Consulta posterior		
Primera consulta	Relevante	No Relevante	**Total**
Relevante	5	9	**14**
NoRelevante	6	12	**18**
Total	**11**	**21**	**32**

Po =	0.53125		
Pe =	0.51953125		
K=	0.024390244	**σ=**	0.1862566502
Ko=	0.2		
α=	0.010	2.326	

Ho K <= .2 **vs.** **Ha k >.2**

$$z = \frac{k-ko}{\sigma} \quad 0.942837509$$

Pvalue = 0.17288201

Anexo V.

A continuación se presenta la tabla de datos con una prueba de hipótesis complementaria para el uso de un intervalo de confianza[58]:

ÍNDICE KAPPA (concordancia entre dos mediciones con k categorías)

datos observados (frecuencias absolutas)

Medida 1	Medida 2 I	II	III	IV	V	VI	Totales
I	5	9					14
II	6	12					18
III							0
IV							0
V							0
VI							0
Totales	11	21	0	0	0	0	32

Concordancia observada		0.5312500	
Concordancia esperada		0.51953125	
Índide kappa de Cohen		0.024390244	
Prueba de significación			
Nivel de signidficación de la prueba (alfa	0.01	"la hipótesis nula: Se dan los resultados por casualidad"	
Valor crítico de la prueba	-2.5758293	0.1406760	2.5758293
Decisión:			
Los Observadores han obtenido una concordancia POR CASUALIDAD			
El p-valor	0.22203147847330200000		
Intervalo de confianza para el índice Kappa		-0.455375094	0.504155581

Cálculos auxiliares	
Valor de A	0.0222349513
Valor de B	0.4668154651
Valor de C	0.2327768302
Desviación estándar del índice kappa	0.1862566502

[58] www.semn.es/educacionyformacion/calculadoras/Concordancia.xls

A continuación se presenta la prueba de independencia entre las variables de la tabla de contingencias usando el estadístico de prueba Chi cuadrado:

Frecuencias Observadas			
	Consulta posterior		
Primera consulta	Relevante	No Relevante	**Total**
Relevante	5	9	**14**
NoRelevante	6	12	**18**
Total	**11**	**21**	**32**

Frecuencias Esperadas			
	Consulta posterior		
Primera consulta	Relevante	No Relevante	**Total**
Relevante	4.8125	9.1875	**14**
NoRelevante	6.1875	11.8125	**18**
Total	**11**	**21**	**32**

gl = 1 **α = 0.01**

X2 = **0.019789734** **<** **6.63**

Ho = **Independencia entre las variables**

Decisión: Ho es aceptada a un nivel de significancia del 0.01

Resultado de selección de archivos por parte del usuario en su primera búsqueda:

Indice	Selección de Documentos Relevantes	
1	E:\Tesis\Corpus\Copernico\Conferencia del CIES.doc	✓
2	E:\Tesis\Corpus\Copernico\mexico.york.com.doc	✗
3	E:\Tesis\Corpus\Copernico\segmento.itam.mx.doc	✓
4	E:\Tesis\Corpus\Copernico\www.alimarket.es.doc	✗
5	E:\Tesis\Corpus\Copernico\www.aprocal.org.mx.doc	✗
6	E:\Tesis\Corpus\Copernico\www.areaminera.com.doc	✗
7	E:\Tesis\Corpus\Copernico\www.centroccc.com.doc	✓
8	E:\Tesis\Corpus\Copernico\www.centrologistico.cl.doc	✗
9	E:\Tesis\Corpus\Copernico\www.cimm.com.mx.doc	✓
10	E:\Tesis\Corpus\Copernico\www.comprasyexistencias.com.doc	✓
13	E:\Tesis\Corpus\Copernico\www.imagine-cs.com.doc	✗
14	E:\Tesis\Corpus\Copernico\www.infoforhealth.org.doc	✓
15	E:\Tesis\Corpus\Copernico\www.markfinanzas.com.doc	✓
16	E:\Tesis\Corpus\Copernico\www.microsoft.com-mexico-soluciones.doc	✗
18	E:\Tesis\Corpus\Copernico\www.seqsa.com.doc	✗
19	E:\Tesis\Corpus\Copernico\www.supplycore.com.doc	✓
20	E:\Tesis\Corpus\Copernico\www-5.ibm.com.doc	✗
25	E:\Tesis\Corpus\Google\www.cnp.org.co.doc	✓
26	E:\Tesis\Corpus\Google\www.comercio.ucv.cl.doc	✗
27	E:\Tesis\Corpus\Google\www.e-global.es.doc	✓
28	E:\Tesis\Corpus\Google\www.etc.org.mx.doc	✓
30	E:\Tesis\Corpus\Google\www.hipermarketing.com.doc	✗
31	E:\Tesis\Corpus\Google\www.icemd.com.doc	✓
32	E:\Tesis\Corpus\Google\www.intel.com.doc	✗
33	E:\Tesis\Corpus\Google\www.intracen.org.doc	✓
34	E:\Tesis\Corpus\Google\www.isnar.cgiar.org.doc	✗
35	E:\Tesis\Corpus\Google\www.itc.mx.doc	✗
36	E:\Tesis\Corpus\Google\www.komintl.com.doc	✓
37	E:\Tesis\Corpus\Google\www.markfinanzas.com.doc	✗
38	E:\Tesis\Corpus\Google\www.microsoft.com-latam-techne.doc	✗
39	E:\Tesis\Corpus\Google\www.mor.itesm.mx.doc	✓
41	E:\Tesis\Corpus\Google\www.tablero-decomando.com.doc	✗

Resultado de selección de archivos por parte del usuario 2ª Etapa:

Selección de Documentos Relevantes	
E:\Tesis\Corpus\Copernico\Conferencia del CIES.doc	✗
E:\Tesis\Corpus\Copernico\mexico.york.com.doc	✗
E:\Tesis\Corpus\Copernico\segmento.itam.mx.doc	✗
E:\Tesis\Corpus\Copernico\www.alimarket.es.doc	✗
E:\Tesis\Corpus\Copernico\www.aprocal.org.mx.doc	✗
E:\Tesis\Corpus\Copernico\www.areaminera.com.doc	✗
E:\Tesis\Corpus\Copernico\www.centroccc.com.doc	✗
E:\Tesis\Corpus\Copernico\www.centrologistico.cl.doc	✗
E:\Tesis\Corpus\Copernico\www.cimm.com.mx.doc	✓
E:\Tesis\Corpus\Copernico\www.comprasyexistencias.com.doc	✗
E:\Tesis\Corpus\Copernico\www.imagine-cs.com.doc	✓
E:\Tesis\Corpus\Copernico\www.infoforhealth.org.doc	✗
E:\Tesis\Corpus\Copernico\www.markfinanzas.com.doc	✗
E:\Tesis\Corpus\Copernico\www.microsoft.com-mexico-soluciones.doc	✓
E:\Tesis\Corpus\Copernico\www.seqsa.com.doc	✓
E:\Tesis\Corpus\Copernico\www.supplycore.com.doc	✗
E:\Tesis\Corpus\Copernico\www-5.ibm.com.doc	✓
E:\Tesis\Corpus\Google\www.cnp.org.co.doc	✓
E:\Tesis\Corpus\Google\www.comercio.ucv.cl.doc	✓
E:\Tesis\Corpus\Google\www.e-global.es.doc	✗
E:\Tesis\Corpus\Google\www.etc.org.mx.doc	✗
E:\Tesis\Corpus\Google\www.hipermarketing.com.doc	✗
E:\Tesis\Corpus\Google\www.icemd.com.doc	✗
E:\Tesis\Corpus\Google\www.intel.com.doc	✗
E:\Tesis\Corpus\Google\www.intracen.org.doc	✗
E:\Tesis\Corpus\Google\www.isnar.cgiar.org.doc	✗
E:\Tesis\Corpus\Google\www.itc.mx.doc	✗
E:\Tesis\Corpus\Google\www.komintl.com.doc	✓
E:\Tesis\Corpus\Google\www.markfinanzas.com.doc	✓
E:\Tesis\Corpus\Google\www.microsoft.com-latam-techne.doc	✗
E:\Tesis\Corpus\Google\www.mor.itesm.mx.doc	✗
E:\Tesis\Corpus\Google\www.tablero-decomando.com.doc	✗