“Aplicación de técnicas de web mining para la búsqueda de patrones de comportamiento en clientes dentro de la web”

HÉCTOR FABIO PANESSO TASCÓN
CARLOS ALBERTO MURILLO MURILLO

CONVENIO
INSTITUTO TECNOLÓGICO DE ESTUDIOS SUPERIORES DE MONTERREY
UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
SANTIAGO DE CALI
2005

”Aplicación de técnicas de web mining para la busqueda de patrones de comportamiento en clientes dentro de la web”

HÉCTOR FABIO PANESSO TASCÓN
CARLOS ALBERTO MURILLO MURILLO

Tesis para optar al título de
Magíster en Ciencias Computacionales

Director
JUAN CARLOS GARCÍA
PHD Ingeniería Industrial

CONVENIO
INSTITUTO TECNOLOGICO DE ESTUDIOS SUPERIORES DE MONTERREY
UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
UNIVERSIDAD AUTONOMA DE OCCIDENTE
SANTIAGO DE CALI
2005

NOTA DE ACEPTACIÓN

_____

Presidente del jurado

_____

Jurado

_____

Jurado

_____

Jurado

Cali, Septiembre del 2005

AGRADECIMIENTOS

- **De los autores**

Las palabras no alcanzarían para expresar nuestra gratitud a todas aquellas personas y entidades que de una u otra manera han colaborado para que este proyecto se convierta en realidad.

Al al doctor Juan Carlos García Díaz, por su dirección y apoyo continuo. A la Universidad Autónoma de Occidente por su colaboración, por su hospitalidad y asistencia en los duros senderos por los que atravesamos y en especial al Al Ingeniero Jorge Pañaloza por su apoyo.

A Dios, quien iluminó nuestro camino, permitio ponernos una meta más y cumplirla con cabalidad, responsabilidad y compromiso, dandonos la salud y paciencia para alcanzarla.

- **De Carlos Alberto Murillo Murillo**

A mi esposa Elsa, por su comprensión, su amor y por apoyarme y a mis hijos Carlos Fernando y Gustavo Adolfo, por ser la luz de mi vida

- **De Héctor Fabio Panesso**

Todo mi agradecimiento va para mi esposa Amparo, quien con su amor y ternura siempre me animó cuando aparecían los obstáculos y a mis hijos, Leonardo y César Augusto, quienes siempre estuvieron apoyándome para llevar a su final este ambicionado proyecto. A mis hermanos Jairo y Alfredo, que siempre estuvieron pendientes de la culminación de esta.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Este trabajo pretende mostrar datos actualizados sobre el *web usage mining* (WUM), incluyendo tanto la investigación académica como la industrial, como una buena oferta comercial; también se presenta las muchas posibles aplicaciones del WUM, que es un proceso de aplicación de técnicas de minería de datos. La minería de datos, que se puede definir como un proceso analítico diseñado para explorar grandes cantidades de datos (generalmente datos de negocios y mercados) con el objetivo de detectar patrones de comportamiento consistentes o relaciones entre las diferentes variables para aplicarlos a nuevos conjuntos de datos. Es importante tener en cuenta que en gran porcentaje el éxito en la aplicabilidad de estas técnicas radica en la información base de la que se disponga; es decir, los datos de entrada (*inputs*).

La WUM esta constituida por tres fases. En este documento se describirán en detalle cada una de estas fases y se encontrara una taxonomía detallada del trabajo en esta área, incluyendo esfuerzos de investigación para un uso comercial.

Asimismo, se describen los objetivos (el general y los específicos) que persigue este trabajo y los tipos de datos de la web que pueden ser usados en el WUM. Luego se muestra se habla del sofware requerido, una aplicación que incluye un informe sobre el macroalgoritmo y los diagramas del software desarrollado como la revisión de software existente en el mercado para generar la información estadistica basada en los datos contenidos en el log del sistema. Finalmente tambien se plantea los resultados de su aplicación, mostrando su análisis estadístico para el caso específico de las necesidades de la Universidad Autonoma de Occidente.

INTRODUCCIÓN

El estudio y aplicación de técnicas avanzadas para el análisis de información almacenada en bases de datos es un tema de investigación en el que ya se han involucrado mucha especialidades, en ingeniería civil y más específicamente en los sistemas de abastecimiento de agua potable aún no han sido lo suficientemente explotados, y es una herramienta que nos puede servir o ser de gran ayuda para resolver problemas de planificación o gestión. Existen muchas formas de almacenar datos, pero no se archivan convenientemente o su entendimiento es nulo.

Este trabajo de investigación, en el primer capítulo, pretende mostrar datos actualizados sobre el ***web usage mining*** (WUM), incluyendo tanto la investigación académica como la industrial, como una buena oferta comercial. También se presenta las muchas posibles aplicaciones del WUM, que es un proceso de aplicación de técnicas de minería de datos para el descubrimiento de patrones dentro de un sitio web, apuntando hacia varias aplicaciones.

En esta parte del documento se describen los antecedentes que hacen que el Data Mining sea importante en nuestra investigación. Asimismo, se describen los objetivos (el general y los específicos) que persigue este trabajo y los tipos de datos de la web que pueden ser usados en el WUM.

El segundo capítulo, o marco teórico, incluye la minería de datos, el cual se puede definir como un proceso analítico diseñado para explorar grandes cantidades de datos (generalmente datos de negocios y mercados) con el objetivo de detectar patrones de comportamiento consistentes o relaciones entre las diferentes variables para aplicarlos a nuevos conjuntos de datos. Como se puede apreciar en la definición anterior, la esencia del problema consiste en "*escarba*r" en la informa-

ción almacenada para "*descubrir*" los elementos de utilidad, término que es asociado con mucha frecuencia al vocablo anglosajón "*knowledge discovery*". En este mismo aparte se presenta una completa taxonomía de los esfuerzos que se han hecho para el uso del WUM, incluyendo el *web*log *sessionnizator*.

También en este segundo capítulo desarrollamos todo el estado del arte. En este punto es importante tener en cuenta que en gran porcentaje el éxito en la aplicabilidad de estas técnicas radica en la información base de la que se disponga; es decir, los datos de entrada (*inputs*). El tema, pese a su gran potencial como herramienta, no ha sido considerado con fuerza para el desarrollo de múltiples aplicaciones, aunque en campos como el de las finanzas, las comunicaciones, los negocios y las páginas web ya se está aplicando desde hace algunos años.

El tercer capítulo se incluye un informe sobre el macroalgoritmo y los diagramas del software desarrollado y se resume las caracteristicas de dos paquetes de software existentes en el mercado.

En el cuarto capítulo se plantea una aplicación, la cual se desarrolla con el software desarrollado y el existente en el mercado, mostrando su análisis estadístico.

El quinto capítulo contempla las conclusiones y recomendaciones para continuar la investigación y se esboza una pequeña discusión sobre políticas de privacidad

Por último tenemos como sexto capítulo el glosario y al final los anexos, cuatro en total.

CAPÍTULO 1
NUESTRO PROYECTO

1.1 HISTORIA

El ***DATA MINING*** o Mineria de Datos (desde ahora DM) es una tecnología de punta que realiza uno de los procesos más viejos de los esfuerzos humanos: el reconocimiento de patrones. Nuestros antepasados confiaban en su capacidad para reconocer los patrones de los depredadores, de los senderos, de las presas y las razones para sobrevivir. El término *web mining* se refiere a la aplicación de técnicas DM para el manejo de grandes volúmenes de datos en la web.

A través del vocablo *web mining* se identifican los siguientes procesos: descubrimiento de información a partir de contenidos en la web (*web content*), descubrimiento de la estructura de los servidores web (*web structure*) y la extracción de patrones de acceso y navegación de usuarios mediante análisis del log (*web usage*).

En la actualidad las empresas y organizaciones que apoyan su gestión en las nuevas tecnologías informáticas, inundadas con millones de datos generados diariamente por las múltiples transacciones de sus cliente (las visitas a las web, códigos de barras, cargos de las tarjetas de crédito y las llamadas telefónicas), afrontan el reto del reconocimiento de patrones de oportunidad y amenaza para su supervivencia. El entorno del negocio asociado a la Internet, hipercompetitivo y en tiempo real, permitirá ver por qué la capacidad de descifrar estos patrones se está convirtiendo rápidamente en un factor crítico para una empresa.

El DM, al contrario de otros métodos de análisis de datos, descubre de un modo fundamental estructuras ocultas, ratios, patrones y firmas. El hecho de que no necesita preparar una búsqueda (query) o configurar el análisis para resolver un problema particular antes de realizar la minería, lo convierten en un método dinámico y efectivo, ya que permite un estudio profundo de las diversas relaciones que se presentan en el manejo de datos. Por ejemplo, el descubrimiento de la salsa caliente y su posterior publicación en la web podría generar que muchos visitantes ocasionales y sin ánimo de comercializar accedieran al *site* mezclándose con "clientes rentistas de altos ingresos" y aquí el análisis que haga el DM es vital ya que permitirá decantar estas relaciones e identificar los compradores más rentables. Así como un investigador forense es capaz de estimar el peso y la altura de una persona que ha dejado una sola huella del pie en una playa, una empresa puede calibrar el valor y las características únicas de los visitantes y clientes más leales y rentables de su sitio web.

A través de una cuidadosa valoración de las pistas o patrones y de las visitas diarias al sitio web se suministran los perfiles de los clientes. Las claves para la identificación de clientes rentables y de potenciales nuevos clientes están contenidas en los datos de los ficheros log, las *cookies* y los formularios de registro. Las herramientas y las técnicas para descubrirlos están en la tecnología DM.

1.2 ANTECEDENTES[1]

La facilidad y la velocidad con que se realizan los negocios en la web, ha propiciado el rápido crecimiento del llamado comercio electrónico. Y, precisamente, son estas cualidades las que hacen que el usuario final experimente un gran cambio a la hora de hacer negocios.

1. URDANETA, Elymir. *El Data Mining*. En: http://www.monografias.com/trabajos7/dami/dami.shtml#funda

La capacidad de seguir la navegación de los usuarios en la web con un solo clic del mouse hacen que el consumidor y el vendedor interactúen de una manera más cercana, esto hace que el vendedor pueda personalizar para ciertos consumidores el mensaje de su producto. En un mercado a gran escala este fenómeno empieza a ser conocido como *arreglo de requisitos particulares en un gran grupo* o *mass customization.*

Las empresas y organizaciones que procesan datos a través de medios computarizados, “coleccionan” cualquier cantidad de información acerca de todas sus operaciones; esto les permite, a la hora de valorar un cliente, obtener su perfil o patrón de comportamiento. Con base en este objetivo, las colecciones de datos han adquirido dimensiones considerables y la posibilidad de utilizarlas es cada vez más alta. Sin embargo, los mecanismos de análisis aún no están claramente definidos, ya que los lenguajes de programación y consulta no fueron concebidos con el propósito de buscar, en forma sistemática, patrones ocultos en los datos.

Es decir, la tecnología necesaria para permitir la búsqueda más o menos automática de patrones en las bases de datos aún no se ha definido completamente. Existen aproximaciones desde diferentes ópticas como la lógica, las matemáticas y la estadística, todas igualmente válidas; y sobre cada una de ellas se han implementado herramientas computacionales que interactúan con algún tipo de base de datos.

Las técnicas del DM son el resultado de un largo proceso de investigación y desarrollo, que se inició cuando los datos de los negocios fueron almacenados por primera vez en computadoras y continuó con mejoras en el acceso a la información; recientemente la incorporación de nuevas tecnologías ha permitido a los usuarios navegar a través de los datos en tiempo real.

La DM es una valiosa manera de capturar información de los modelos subyacentes en los datos comerciales. Con estos modelos es fácil predecir factores de mercadeo e inventar estrategias eficaces en línea que puedan predecir el potencial de un costo eficaz a través de la dirección de inventarios de mercadeo inteligente.

El DM, para obtener información provechosa que no está representada explícitamente en los datos, combina técnicas de inteligencia artificial, análisis estadístico, bases de datos y modelos de visualización gráfica. Asimismo, mediante el uso de redes neuronales, máquinas de aprendizaje y algoritmos genéticos descubre los patrones, perfiles y tendencias de los clientes que acceden a los sitios web. Este es un proceso iterativo de extracción de patrones procedentes de las transacciones de negocios on-line (hacer clic sobre un enlace o sobre un banner, pedir información por e-mail, rellenar un formulario).

En particular, la investigación sobre el uso de la web se ha centrado en el descubrimiento de patrones de acceso a partir de ficheros log. Un patrón de acceso es una secuencia que se repite dentro de los log*s*. La información recogida y almacenada por los servidores web es la principal fuente de datos para analizar patrones de navegación de usuario.

El análisis de los log*s* debería servir para mejorar el diseño, creando sitos web adaptativos.

1.3 RESULTADOS ESPERADOS

- A continuación se presentarán algunos puntos que posibilitan la valoración de los resultados que se pueden obtener cuando se aplican las técnicas de minería de datos:

- Gran libertad a los usuarios profesionales. Estos pueden realizar sus propios modelos sin necesidad de subcontratar ni ponerse de acuerdo con una persona especialista en estadística, ganando tanto en desarrollo como en manejabilidad e incluso en tiempo de trabajo.

- Rapidez y sencillez al manejar grandes cantidades de datos (varios millones). También permite seleccionar una gran cantidad de variables predictivas (hasta varios millares), determinando las más útiles dentro de una gran masa.

Es importante aclarar que no se puede esperar un alto poder predictivo en técnicas de DM, ya que los modelos resultantes tienen un grado alto de legibilidad; es decir, cualquier persona puede interpretarlas fácilmente.

- Identificación, recolección y almacenamiento de información en una base de datos que a su vez es alimentada por otra base de datos.

- Interpretación y depuración de resultados. Para la evaluación de los datos se usarán algunos algoritmos propuestos que ayudarán a entender mejor los datos. Una de las ventajas de estos algoritmos es que la identidad del usuario no es un requisito para predecir las tendencias de navegación. El objetivo del algoritmo es descubrir los caminos frecuentes sin tener en cuenta la identidad del usuario.

1.4 OBJETIVOS

1.4.1 General

Desarrollar un procedimiento para aplicar el DM a la web (*web mining*), dentro de un ambiente real, con los datos provenientes del tráfico del portal de la Universidad Autónoma de Occidente (UAO), usando tecnología estándar para el desarrollo de software de extracción y preparación de datos que permita su alimentación a

una herramienta existente en el mercado que posee la institución con el fin de proporcionar reportes que ayuden a la toma de decisiones, según el patrón de comportamiento de los individuos.

1.4.2 Específicos

- Analizar la información que emana periódicamente de las bases de datos, cruzándola con aquella que existe en la universidad, para conformar alertas e informes oportunos.

- Enfocar el *web usage mining* hacia el descubrimiento de patrones de acceso a partir de ficheros log, haciendo diferentes aproximaciones a la solución del problema.

- Generar una herramienta que tenga la capacidad de tomar la información existente en una base de datos alimentada por medio de la web, para procesarla posteriormente para la producción de los reportes requeridos.

CAPÍTULO 2
MARCO TEÓRICO DEL PROYECTO

2.1 LA W*ORLD WIDE WEB* (WWW) Y EL NAVEGANTE[2]

La web se inicia en marzo de 1989 como una propuesta del investigador Tim Berners-Lee, perteneciente a la CERN (Centro Eupoeo para la Investigación Nuclear) para desarrollar un sistema de creación y distribución de documentos (hipertexto) que permitiera a investigadores ubicados en distintos lugares del mundo compartir información.

Luego se abocó el desarrollo de una solución que permitiera cubrir los siguientes requerimientos:

- Obtener una interfaz consistente; es decir, el sistema debería permitir una conexión que al menos asegurara una transferencia de datos consistente (lo que envío es obtenido intacto).

- Incorporar un amplio rango de tecnologías y distintos tipos de documentos.

- Proveer una herramienta para la lectura, de forma sencilla, de los documentos desde cualquier lugar y por cualquier individuo que esté navegando.

2. Internation conference web. http://www2003.org/cdrom/html/refereed/index.html Web Mining Forum, 2003. http://km.aifb.uni-karlsruhe.de/ws/ewmf03/accept_papers_html
Papers: The PageRank Citation Ranking: Bringing Order to the Web
http://citeseer.nj.nec.com/page98pagerank.html) Semantic Web: An introduction

Luego de esta primera etapa, se comenzó a desarrollar lo que conocemos como browser para poder acceder a la información desde cualquier lugar de forma rápida, sencilla e independiente de la plataforma utilizada por el usuario.

Con el crecimiento explosivo de los recursos de información disponibles en el *world wide web* se ha incrementado la necesidad de utilizar herramientas automatizadas para encontrar los recursos de información deseados. Estos factores aumentan la necesidad de crear sistemas inteligentes, tanto para el servidor como para el cliente.

Las características de conducta del usuario, se han ido volviendo una necesidad prioritaria para los administraores de los sitios Web para determinar sus gustos y poderlos ofrecer reflejardos en el sitio la proxima vez que el navegante, que es el cliente de nuestro sitio. En el caso de la Universidad es una necesidad que la información que se presenta a nustros clientes los estudiantes sea necesaria, agradable y que haga que el sitio sea una verdadera necesidad para la vida universitaria. Determinar esa conducta se vuelve la necesidad y se presentan dos problemas a resolver: Como almacenar y extraer información util y el segundo la información que permita determinar la conducta.

La mayoría de esta información es producida automáticamente por los servidores web y se almacena 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.

La escasez de un nivel básico de análisis dentro de los browser es principalmente una de las mayores dificultades para la recolección de datos, debido al ambiente en el cual se encuentra su formato, dado que no está enfocado hacia un segmento de un usuario particular.

A pesar de estos métodos que restringen las peticiones a través de los *proxies*, el monitoreo de los eventos dentro de los niveles del sistema operativo y el retiro de flaws en varias aplicaciones del navegador permiten que la caracterización pueda ser realizada hoy sin el acceso al código fuente o a los API; ninguna de las técnicas anteriores permite que todos los eventos de la interface del usuario puedan ser capturados en el contexto, limitando el alcance y exactitud de los resultados.

El primer estudio para caracterizar la conducta del cliente en la red fue realizado por Catledge y Pitkow[3] (1995) durante el verano de 1994. El estudio se dirigió por un período de tres semanas, usando una versión totalmente modificada de *Xmosaic* en el Instituto de Tecnología de Georgia. Las 107 personas estudiadas proporcionaron una completa información de la actividad de la interface del usuario, incluso la verificación excesiva de las características de navegación (90%), y los hipervínculos de “siguiente” (52%) y el uso del botón “atrás” (41%). Sólo el 2% de todas las peticiones de URL fueron tecleadas por los usuarios por medio de la caja de diálogo “abrir URL”.

El tiempo más demorado entre los eventos de la interface era 9.3 minutos, con sesiones que se limitaban por intervalos de inactividad de 25 minutos o más. Adicionalmente, el modelo de navegación donde las personas acostumbran tener una página central para explorar se observó en muchos de los registros. Poco después de esto, durante el otoño de 1994, el Boston University Grupo dirigió uno de los más grandes estudio sobre 600 usuarios durante un período de seis meses, usando una versión modificada de *Xmosaic* que se enfocó solamente en eventos de la navegación. Este paquete de datos ha sido usado ampliamente por varios grupos de investigación en sus esfuerzos por caracterizar la web.

Adicionalmente, más del 75% de las URL pedidas estaban afuera de los servidores del campus y el 96% de las URL pedidas por todos los usuarios fueron solici-

3. http://tecnologiaedu.us.es/bibliovir/pdf/18.pdf

tadas más de una vez. Incluso para entonces Yahoo, que residía en http://akebono.stanford.edu. era el sitio más popular.

2.2 LA NECESIDAD DE LA MINERÍA DE DATOS[4] O DATA MINING (DM) Y DEL WEB MINING[5]

En la actualidad las organizaciones se enfrentan a un mundo cada vez más competitivo y, por tanto, las estrategias de administración deben ser flexibles para adaptarse a las condiciones cambiantes del entorno.

A medida que más empresas basen su negocio en la Internet, las estrategias y técnicas tradicionales para el análisis del mercado deben ser vistas desde un nuevo contexto. Las organizaciones y compañías en la Internet generan y almacenan grandes volúmenes de datos en sus funcionamientos diarios.

Analizar tales datos puede ayudar a las organizaciones a determinar qué usuarios visitan su sitio, permitiendo implementar 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 pueden proporcionar información valiosa de como mejorar la estructura del sitio, creando una presencia en la Internet más eficaz para las organizaciones.

Lo anterior supone un enorme reto, en especial en el manejo de grandes volúmenes de información para conocer el entorno y predecir su evolución. En otras palabras, es necesario conocer quiénes son los principales clientes, qué tipo de servicios son los que más consumen, cómo se agrupan, cómo es su patrón de compra, qué segmentos del mercado se desaprovechan, quiénes son los principales com-

4. CAMPBELL, Richard. *Know your data*. Data based Web Advisor. Oct 1997. v15 n10 p56 (4) In: DELROSSI, Robert A. *Enario packs power as a data-exploration tool: Cognos' program is particularly well-suited to experienced analysts*; InfoWorld. June 2, 1997, V. 19, No. 22, p. 130 (1).
5. R. Cooley, J. Srivastava, B. Mobasher. *Web mining: Information and pattern dicovery on the word wide web*. Nov. 1997.

petidores, en qué mercado se desempeñan, cuáles son sus puntos débiles y fuertes, etc.

De forma general, las organizaciones tienen toda la información necesaria para responder a las anteriores preguntas, pero no son capaces de utilizarla al máximo pues no la tienen organizada adecuadamente y carecen de los métodos necesarios para procesarla y analizarla de la mejor manera.

Tomando en consideración la importancia de extraer los conocimientos “perdidos” en los datos que almacena la organización, ha surgido desde hace un tiempo lo que se conoce como “minería de datos”.

2.2.1 ¿Qué es la minería de datos?

Recordemos que la mineria de datos es un proceso analítico diseñado para explorar grandes cantidades de datos (generalmente de negocio y de mercado) con el objetivo de detectar patrones y modelos de comportamiento o relaciones entre las diferentes variables para aplicarlos a nuevos conjuntos de datos.

El DM permite obtener valores a partir de la información que registran y manejan las empresas, lo que ayuda a dirigir esfuerzos de mejora, respaldados en datos históricos de diversa índole.

Como se puede apreciar en la definición anterior, la esencia del problema consiste en “escarbar” en la información almacenada para “descubrir” los elementos de utilidad. Es por esta razón que el término se asocia con mucha frecuencia a la expresión anglosajona “*knowledge discovery*”.

Las dos metas primarias de la minería de datos son la *predicción* y la *descripción*.

La **predicción** involucra el uso de algunas variables o campos de la base de datos para predecir los valores futuros de otras variables de interés.

La **descripción** se enfoca en encontrar patrones interpretables por humanos para la descripción de los datos. En términos de KDD (Knowledge in Databases) suele ser más importante la descripción que la predicción. Para obtener estas metas se realizan las siguientes tareas primarias:

- Clasificación.
- Regresión.
- Agrupamiento (*clustering*).
- Sumarización.
- Modelamiento de dependencias.
- Detección de cambios y desviaciones.

Los algoritmos de minería de datos tienen tres componentes primarios:

- *Representación del modelo:* específicamente trata de la identificación del lenguaje de representación para la descripción de los patrones descubribles.

- *Evaluación del modelo:* estima que tan bien un patrón particular satisface los criterios del proceso de KDD.

- *Método de búsqueda, compuesto por búsqueda de parámetro y búsqueda de modelo*. En la búsqueda de parámetros, el algoritmo debe hallar los parámetros que optimizan los criterios de evaluación del modelo, dados los datos observados y una representación fija del modelo. La búsqueda del modelo ocurre

un ciclo por encima de la búsqueda de parámetros; es decir, se realiza el proceso de búsqueda de parámetros para toda una familia de modelos.[6]

El proceso de minería de datos consta de tres etapas fundamentales:

1. Integración, limpieza y exploración de los datos.
2. Definición de patrones o construcción de modelos.
3. Validación y verificación de los modelos.

Estas etapas, a grandes rasgos, son similares a las de la metodología general de modelación matemática que puede encontrarse en cualquier texto sobre el tema. Debido a que la información de corte comercial no permite la obtención de modelos con un elevado coeficiente de certeza, las decisiones que se tomen se insertan en el campo de la decisión con incertidumbre.

Por ello la minería de datos, a pesar de basarse en los principios tradicionales de análisis exploratorio de datos, llega más lejos, incluso a incorporar elementos de inteligencia artificial. En ocasiones la minería de datos se considera como una extensión del concepto de almacenamiento de datos.

2.2.2 Herramientas en la minería de datos

Las técnicas de DM son el resultado de un largo proceso de investigación y desarrollo. Esta evolución comenzó cuando las empresas empezaron a almacenar datos en computadoras y continuó con mejoras en el acceso a los mismos. Recientemente se han implementado tecnologías que posibilitan a los usuarios navegar a través de los datos en tiempo real. El DM aborda este proceso de evolución más allá del acceso y la navegación, gracias a las tres tecnologías que lo soportan, las cuales ya están suficientemente maduras: Recolección masiva de datos, Potentes computadoras con multiprocesadores y Algoritmos de DM.

6. Ibid, pp. 16-17.

Las herramientas de DM predicen futuras tendencias y comportamientos, permitiendo la toma de decisiones proactivas. Estas herramientas exploran las bases de datos en busca de patrones ocultos, encontrando información predecible que un experto no puede suministrar porque se encuentra fuera de sus expectativas. Por ellos muchas compañías ya recolectan y refinan cantidades masivas de datos.

Las técnicas de DM pueden ser implementadas rápidamente en plataformas ya existentes de software y hardware para acrecentar el valor de las fuentes de información y pueden ser integradas con nuevos productos y sistemas, pues son traídas en línea (on-line).

- **Almacenamiento de datos (*data warehouse)***

Se define como un proceso de organización de grandes cantidades de datos de diversos tipos, “guardados” en la organización con el objetivo de facilitar la recuperación de la misma con fines analíticos.

Por supuesto, el almacenamiento de información tiene gran importancia en el proceso de minería de datos, pues en cierta medida posibilita la recuperación o al menos la referencia a determinados conjuntos de datos de importancia para un proceso de toma de decisión. En la actualidad existe gran variedad de sistemas comerciales para el almacenamiento de datos entre los que se destacan Oracle, Sybase, MS SQL Server, entre otros.

- **Análisis exploratorio de datos (*Exploratory Data Analysis* –EDA-)**

Como se dijo anteriormente, las técnicas de análisis exploratorio de datos juega un papel muy importante en la minería de datos. Las mismas tienen como objetivo determinar las relaciones entre las variables cuando no hay o no está totalmente definida la naturaleza de estas relaciones.

Las técnicas exploratorias tienen un fuerte componente computacional, abarcando desde los métodos estadísticos simples a los más avanzados como las técnicas de exploración de multivariables, diseñadas para identificar patrones en conjunto de datos multivariables.

Entre las técnicas estadísticas sencillas se incluyen el estudio de distribuciones de las variables, estudio de correlaciones entre matrices, tablas de contingencias, entre otros. Por su parte, entre las técnicas más complejas se incluyen el análisis de factores, el análisis de grupos, el escalado multidimensional, etc., consistentes en algoritmos particulares de búsqueda que, bajo ciertas limitaciones aceptables de eficiencia computacional, producen una enumeración particular de patrones.[7] La minería de datos involucra modelos para ajustar o patrones a determinar sobre los datos observados.

2.2.3 ¿Qué es *web mining*?[8]

El término *web mining* designa la aplicación de técnicas DM a grandes depósitos de datos en la web. Su acepción es tan amplia que se usa para referirse a los procesos de descubrimiento de información a partir de contenidos en la web (*web content*), de descubrimiento de la estructura de los servidores web (*web structure*) y al proceso de extracción de patrones de acceso y navegación de usuarios mediante análisis de log (*web usage*).[9]

Esta tecnología ha surgido en la última década como una estrategia para afrontar los retos que encuentra el hombre moderno a la hora de la generación y almacenamiento masivo de información, la cual sobrepasa la capacidad de las herramientas de análisis de datos tradicionales.

7. CAMPBELL, Richard. *Op. cit.,* p. 9.
8. www.quadernsdigitals.net/index.php?accionMenu=hemeroteca
9. MARBAN GALLEGO, Oscar. *Resumen de su trabajo de investigación.* Universidad Politécnica de Madrid.

Algunos investigadores consideran que el descubrimiento del conocimiento es uno de los resultados finales más deseables de la computación. Y que el hallazgo de nuevos fenómenos o el mejoramiento de nuestro conocimiento acerca de los mismos tienen un valor mayor que la optimización de procesos de producción o de inventarios y está solo en segundo orden de importancia frente a las tareas que ayudan a preservar nuestro mundo y nuestro ambiente. Gio Wiederhold, de la Universidad de Stanford[10], concluye que no es una sorpresa que sea también uno de los retos de la computación más difíciles de cumplir.

2.3 DESCUBRIENDO INFORMACIÓN OCULTA, LA POTENCIALIDAD DE LAS HERRAMIENTAS DE DATA MINING

Las herramientas de DM predicen futuras tendencias y comportamientos, permitiendo en los negocios tomar decisiones proactivas y conducidas por un conocimiento acabado de la información (*knowledge-driven*). Los análisis prospectivos automatizados ofrecidos por un producto así van más allá de los eventos pasados provistos por herramientas retrospectivas típicas de sistemas de soporte de decisión. Las herramientas de DM pueden responder a preguntas de negocios que tradicionalmente consumen demasiado tiempo para poder ser resueltas y a los cuales los usuarios de esta información casi no están dispuestos a aceptar. Estas herramientas exploran las bases de datos en busca de patrones ocultos, encontrando información predecible que un experto no puede llegar a encontrar porque se encuentra fuera de sus expectativas.

El nombre de DM deriva de las similitudes entre buscar valiosa información de negocios en grandes bases de datos por ej.: encontrar información de la venta de un producto entre grandes montos de gigabytes almacenados y minar una montaña para encontrar una veta de metales valiosos. Ambos procesos requieren examinar una inmensa cantidad de material o investigar inteligentemente hasta encontrar

10. http://elies.rediris.es/elies9/4-1-1.htm

exactamente donde residen los valores. Dadas las bases de datos de suficiente tamaño y calidad, la tecnología de DM puede generar nuevas oportunidades de negocios al proveer las siguientes capacidades:

- **Predicción automatizada de tendencias y comportamientos**. DM automatiza el proceso de encontrar información predecible en grandes bases de datos. Preguntas que tradicionalmente requerían un intenso análisis manual, ahora pueden ser contestadas directa y rápidamente desde los datos. Un típico ejemplo de problema predecible es el marketing apuntado a objetivos (*targeted marketing*). DM usa datos en *mailing* promocionales anteriores para identificar posibles objetivos para maximizar los resultados de la inversión en futuros *mailing*.

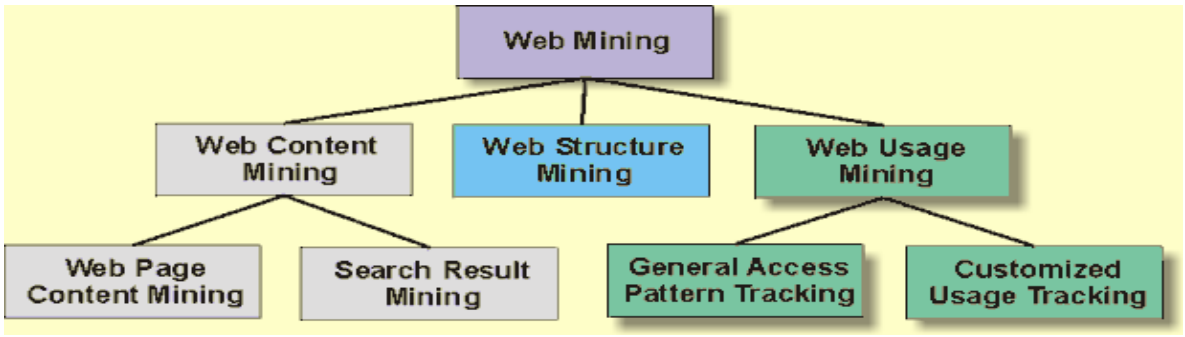
- **Descubrimiento automatizado[11] de modelos previamente desconocidos**. Las herramientas de DM barren las bases de datos e identifican modelos previamente escondidos en un sólo paso. Otros problemas de descubrimiento de modelos incluye detectar transacciones fraudulentas de tarjetas de créditos e identificar datos anormales que pueden representarse.

2.4 TAXONOMÍA DE *WEB MINING*[12]

El *web mining* es la extracción de patrones interesantes y potencialmente útiles e información implícita de artefactos o actividades relacionadas con el *world wide web*.. Este describe la búsqueda automática de recursos informativos disponibles en línea, por ejemplo *web content mining*, y el descubrimiento del acceso del usuario a patrones del *web servers*, por ejemplo *web usage mining*. La taxonomía del *web mining* se representa en el cuadro siguiente:

11. PRESSER CARNE, Cynthia. http://www.monografias.com
12. http://maya.cs.depaul.edu/~MOBASHER/WEBMINER/SURVEY/NODE2.HTML

Figura 1
Taxonomía de la web mining



MOBASHER/WEBMINER/SURVEY/NODE2.HTML

Existen tres dominios relacionados con el descubrimiento del conocimiento que pertenece al *web mining*: *web content mining, web structure mining* y *web usage mining*.

- *web content mining* es el proceso de extraer conocimiento de documentos o de sus descripciones. Los documentos web en texto mining, la búsqueda de recursos basados en conceptos de tecnología de índice o de *agentbased* pueden también incluirse en esta categoría.

- *web structure mining* es el proceso de inferir conocimiento de la *world wide web* y enlaces entre las referencias y referentes en la web.

- *web user mining*, también conocido como el *web* log *mining*, es el proceso de extraer patrones interesantes en *web acces logs.*

El papel del *web mining* en ámbitos ligados al comercio electrónico, adquiere una relevancia determinante y una oportunidad manifiesta de optimizar los procesos comerciales y de marketing, basados en la tecnología web*,* abriendo la posibilidad, al mismo tiempo, de rediseñar dinámicamente la estructura del servicio ofertado a partir de los resultados y tendencias identificados tras el análisis.

La interacción a través de la web constituye, en definitiva, una valiosa herramienta como fuente de información a partir de la cual pueden construirse modelos analíticos que formalicen adecuadamente las pautas de comportamiento y consumo de sus usuarios, posibilitando que el resultado de tales procesos de análisis (*web mining*) facilite la optimización tanto del propio servicio ofrecido a través de la web como de los procesos estratégicos de la empresa, refiriendo ambas vertientes a la personalización de contenidos y oferta, en función de la naturaleza y características detectadas en los clientes del servicio.

2.4.1 *Web content mining*[13]

El *web content mining* es un proceso que va más allá de la extracción de una palabra clave. Debido a que el contenido de un documento no presenta una organización semántica, algunos enfoques han sugerido reestructurarlo en una representación que podría ser aprovechada por máquinas.

El método usual para aprovechar la estructura conocida en documentos es usar *wrappers* para documentos y algunos modelos de datos. Las técnicas de uso del léxico para interpretación de un contenido aún están por llegar.

Existen dos grupos de estrategias de *web content mining*: aquellas que comprenden directamente el contenido de documentos y aquellas que mejoran en la búsqueda de contenidos de otras herramientas como los motores de búsqueda.[14] Este dominio comprende la extracción de conocimiento a partir del contenido de las páginas *web* o de su descripción.

Bajo este epígrafe pueden enmarcarse los mecanismos automáticos para la generación de resúmenes de páginas de contenidos o páginas de resultados de búsquedas generales en el uso del servicio web, proporcionando el conocimiento ne-

13. VEGA ANEIROS, Andrés. w*eb mining en el marco de los negocios*.
14. VEGAS ANEIROS, Andrés. *Web mining en el marco de los procesos de negocio y marketing de la empresa*. Telefónica, investigación y desarrollo.

cesario para su personalización, así como para segmentar y caracterizar la población de clientes a partir de tales patrones de uso.

En el caso de entornos en los que se desarrollen actividades ligadas al comercio electrónico vía web, abarca también la identificación de patrones de consumo, permitiendo segmentar y caracterizar la población de clientes, detectar patrones de tendencias y necesidades y detectar nuevas oportunidades comerciales y de marketing (*cross-selling*, *up-selling*, etc.).

El conocimiento obtenido podrá aplicarse tanto en la implantación de sistemas de personalización de contenidos (navegación guiada, publicidad dinámicamente, etc.) y en la reestructuración del servicio web, con el fin de adecuarse a las pautas transaccionales registradas (*web structure mining* y *web usage mining*), como en la elaboración de planes personalizados de marketing y comercialización (marketing one-to-one), basados en los patrones de comportamiento y consumo extraídos del análisis de las tendencias de uso (*web usage mining*).

- **Wrappers**

TCP wrappers[15] permite controlar y proteger los servicios de red, limitando el acceso como sea posible y registrado todos las conexiones para hacer el trabajo de detectar y resolver problemas de forma más fácil.

Un *wrapper* es un programa para controlar el acceso a un segundo programa. El *wrapper* literalmente cubre la identidad del segundo programa, obteniendo con esto un alto nivel de seguridad.

Los *wrappers* son usados dentro de la seguridad en sistemas UNIX. Estos programas nacieron de la necesidad de modificar el comportamiento del sistema operativo sin tener que modificar su funcionamiento.

15. www.rediris.es/cert/doc/docu_rediris/wrappers.es

Su amplia utilización les ha permitido llegar a formar parte de herramientas de seguridad por las siguientes razones:

- Debido a que la seguridad lógica está concentrada en un solo programa, los *wrappers* son fáciles y simples de validar.

- Debido a que el programa protegido se mantiene como una entidad separada, este puede ser actualizado sin necesidad de cambiar el *wrapper*.

- Dado que los *wrappers* llaman al programa protegido mediante el sistema estándar exec ( ), se puede usar un solo *wrapper* para controlar el acceso a diversos programas que se necesiten proteger.

La historia del TCP *wrappers* se remonta a 1990, cuando una de las máquinas de la Universidad de Eindhoven, en Holanda, era objeto de fuertes ataques por un hacker alemán, que seguido obtenía privilegios de root e introducía el comando "rm-rf/".

2.4.2 *Web structure mining*[16]

La *world wide web* puede revelar más información que la contenida en muchos documentos. Por ejemplo, los enlaces que apuntan a un documento indican su popularidad, mientras que los enlaces que salen de un documento indican la riqueza o quizás la variedad de tópicos o temas tratados en él. Esto puede ser comparado con las citas bibliográficas. Cuando un documento es con frecuencia citado es porque debe ser importante.

La PageRank y los métodos CLEVER toman ventaja de esta información llevados por los enlaces para encontrar las páginas web pertinentes. Por medio de conteos,

16. VEGA ANEIROS, Andrés. *Web mining en el marco de los negocio*.

los niveles más altos acumulan el número de artefactos sometidos por los conceptos que ellos sostienen. Los conteos de hiperenlaces, documentos internos y externos, reanudan la estructura de artefactos web resumidos.[17]

La *web structure mining* intenta descubrir la organización de los enlaces del conjunto de hiperenlaces dentro del documento para generar un informe estructural sobre la página y el sitio web. Según el objetivo a estudiar, se pueden dar tres tipos de informes:

- Basándose en los hiperenlaces, clasifica las páginas web y genera el informe.

- Revelando la estructura del documento web en sí.

- Descubriendo la naturaleza o jerarquía de la red de hiperenlaces del sitio web de un dominio particular.

Estos informes suelen dar como resultado representaciones gráficas para una mejor visión del conocimiento obtenido y pueden utilizarse como guía para el usuario en busca de información.

2.4.3 *Web usage mining*[18]

Los servidores graban y acumulan datos sobre las interacciones de un usuario sin importar cuándo sean requeridos los recursos. El análisis del *web acces log* de diferentes sitios web puede ayudar a entender el comportamiento del usuario y la estructura web, debido a esto se mejora el diseño de esta colección colosal de recursos. Hay dos tendencias principales en la *web usage mining* llevadas a cabo por las aplicaciones de los descubrimientos: acceso general a rastreo de patrones (*General Access Pattern Tracking*) y rastreo de uso personalizado (*Customized Usage Tracking*).

17. GARCÍA-SANTIAGO, Lola. *El web mining: una tecnología para la indagación en la world wide web*.
18. VEGA ANEIROS. *Op. cit*.

El acceso general a rastreo de patrones analiza los *web logs* para entender sus tendencias. Estos análisis pueden ser vislumbrados en una mejor estructura y en una agrupación de proveedores de recursos. La técnica de *DATA MINING* en *access logs* muestra accesos a patrones interesantes que pueden ser usados para reestructurar sitios en una agrupación más eficiente, poder ubicar avisos de forma efectiva y usuarios específicos para ventas determinadas.

El rastreo de uso personalizado analiza las tendencias individuales. Su propósito es personalizar los sitios web para los usuarios. La información desplegada, la profundidad de la estructura del sitio y el formato de los recursos pueden ser todos personalizados dinámicamente para cada usuario sobre el tiempo basado en su acceso a los patrones.

Mientras es motivante y excitante ver las diversas aplicaciones potenciales de los archivos de análisis *web logs*, es importante saber que el éxito de tales aplicaciones depende de cuán válido y confiable es el conocimiento que se puede descubrir de las largas filas de datos log. Los servidores web corrientes almacenan información limitada sobre los accesos.

2.5 EL MEDIO: EL ANALISIS DEL LOG

El log es una fuente de información que debemos tener presente para valorar el rendimiento de nuestro site y para extraer información sobre nuestros productos. El log nos da la información sobre qué información están solicitando los usuarios, que información les resulta más interesante. Técnicamente el “log de visitas” es un archivo creado por el servidor donde se registran las acciones que los usuarios generan en la web.

Es muy importante que esta fuente de información esté accesible por todos los departamentos en un formato sencillo de entender y fácil de usar. Es esencial que la comunicación entre los departamentos (o individuos) que tienen la información del log y aquellos que la pueden necesitar sea fluida para no desaprovechar la riqueza de información que se puede extraer del log.

2.5.1 Log del servidor

Desde los primeros tiempos de la red, los servidores web registran de forma incremental todos los accesos a objetos recibidos desde los clientes en el denominado log del servidor. Este log es el que se emplea para los análisis del tráfico y uso siendo periódicamente procesado y analizado por el responsable del sitio web para extraer información sobre el uso de su servidor.

Los ficheros log del servidor contienen entradas irrelevantes o redundantes para tareas de *DATA MINING* como las entradas relativas a ficheros de imágenes o códigos de error que no son necesarios para la obtención de perfil de usuario. Por tanto, estos ficheros deben ser tratados eliminando toda la información innecesaria para cálculo de perfiles. Luego debe identificarse las transacciones de datos y agruparse en sesiones con significado.

El principal problema de los logs del servidor con el protocolo *http*, es que este no tiene indicación de estado (*stateless*) de forma que es imposible saber cuando un usuario deja el servidor, por lo que se debe realizar algunos supuestos para identificar sesiones. La aproximación seguida en la mayoría de los casos es agrupar todas las transacciones con la misma dirección IP de modo que el tiempo transcurrido entre dos accesos consecutivos esté por debajo de cierto umbral.

Una vez que los logs se han preprocesado y se dispone de información de sesiones hay varias clases de descubrimiento de acceso que pueden realizarse depen-

diendo de las necesidades del analista. Por ejemplo análisis de rutas, descubrimiento de reglas de asociación, patrones secuenciales, clustering y clasificación.

Un serio problema al usar el log del servidor son los accesos que no son registrados, ya que mecanismos como las cachés (locales o compartidas) y los servidores proxy pueden distorsionar gravemente la imagen global de los movimientos del usuario por el sitio web. Un objeto listado sólo una vez en un histórico de accesos puede haber sido accedido muchas veces por distintos usuarios, sin que las peticiones lleguen al servidor al estar el objeto almacenado en una de estas cachés intermedias.

El archivo en bruto presenta un formato bastante espeso y su tamaño puede ser un problema a la hora de manejarlo (un log de 20.000 visitas puede pesar tres magabytes).

Ejemplo de línea de un log:

```
200.66.196.132 - - (03/Mar/2002:23:59:20 +0100) “GET /directorio/16.html HTTP/1.1” 304 – “http://www.dominio.com/directorio/index.html” “Mozilla/4.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)”
```

El formato de los logs varía dentro de cada servidor y en su misma configuración.

Cada vez que se produce una llamada a un archivo dentro de nuestra web, el servidor escribirá una línea como la mostrada en el ejemplo. Cada visita, cada HTML, cada imagen dentro de ese HTML generará una línea. Es esta “redundancia” de información lo que hace que el log sea bastante difícil de manejar en bruto.

Explicación de cada elemento dentro del log:

200.66.196.132	I.P. del usuario (número de identificación del ordenador desde el que el usuario accede)
-	Usuario - para sites con identificación</TD< tr>
-	Password
(03/Mar/2002:23:59:20 +0100)	Fecha
“GET /directorio/16.html HTTP/1.1”	Archivo pedido
304	ID de contestación (2xx es OK | 3xx son de redirección | 4xx son de autentificación | 5xx son de errores de servicio)
-	Número de bytes enviados
“http://www.dominio.com/directorio/index.html”	Página desde la que se pide el archivo
“Mozilla/4.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)”	Información sobre versión del navegador, terminal del usuario...

Cómo extraer información. Excel puede ser un primer paso muy útil. Extraer información de un log de visitas en crudo usando una herramienta como Excel no es fácil y presenta muchos problemas, pero nos puede ayudar a entender mejor como funciona el proceso de representación de un log.

- Un primer paso es eliminar la información “redundante”. Las peticiones de gráficos, hojas de estilo, flash... pueden tener su utilidad, pero para analizar el tráfico de nuestra web los documentos “html” serán la referencia válida.

- Eliminando las líneas que no interesan, el tamaño del archivo se puede reducir hasta x6 (un archivo de 20 megas pasó a 3 suprimiendo las líneas de información “no útil”). Al final de este artículo se incluyen enlaces a programas que pueden ayudar a limpiar el log.

Teniendo un archivo limpio, se puede importar a Excel. A la hora de abrir un log, Excel presenta un “wizard” que ayudará a tabular la información creando un documento con la información separada por celdas. Excel presenta una limitación de unas 65.000 filas.

– Dentro de Excel se ordenará la información para extraer conclusiones sobre el log.

Resultados que se obtienen ordenado el log.

	SI ORDENAMOS POR...	...OBTENEMOS.
200.66.196.132	I.P. del usuario	Número de usuarios únicos, rutas de dichos usuarios, páginas de salida...
-	Usuario - para sites con identificación	Número de usuarios registrado frente a usuarios anónimos, ratio de conversión...
-	Password	Número de errores a la hora de introducir el password, longitud de password preferida...
(03/Mar/2002:23:59:20 +0100)	Fecha	Fechas con más visitas, duración de la visita (combinado con el IP), identificación geográfica (global) de los visitantes...
“GET /directorio/16.html http/1.1”	Archivo pedido	Páginas más vistas - si lo combinamos con IPs sacamos las rutas.
304	ID de contestación (2xx es OK | 3xx son de redirección | 4xx son de autentificación | 5xx son de errores de servicio)	Errores de peticiones, movimiento de archivos...
-	Número de bytes enviados	Envió total de datos (transferencia), El tiempo de visita se puede ver afectado por el peso de los archivos solicitados.
“http://www.dominio.com/directorio/index.html”	Página desde la que se pide el archivo	Qué páginas están dirigiendo el tráfico, qué enlaces dentro de qué páginas son los más usados, resultado de búsquedas (internos y externos), páginas externas con enlaces a nuestro site...
“Mozilla/4.0 (compatible; MSIE 5.5; Windows 98; Win 9x 4.90)”	Información sobre versión del navegador, terminal del usuario...	Estadística sobre versiones, target real del site...

La idea de abrir un log en crudo requiere dedicación y algo de tiempo, pero ayudará a entender de forma más clara la información que se está manejando y cómo pedir los resultados que realmente mejorarán el site.

Como alternativa, siempre se puede disponer de alguna aplicación de análisis de log. Disponibles en una amplia gama (gratuitos, de pago, caros, baratos, buenos, malos...). El uso de estas herramientas suele requerir algo de dedicación para poder extraer los datos en un formato que ofrezca resultados válidos. Por lo general todos los fabricantes ofrecen un tiempo de prueba o versiones demo con lo que se puede comparar antes de comprar. Al final de este artículo se consignan algunas recomendaciones.

Qué hacer con los resultados. Para valorar los resultados de un log, lo más normal es comparar los datos de forma interna viendo la evolución en el tiempo (comparando el número de visitas mes a mes, número de páginas vistas, duración de las visitas, etc.). Esta comparación dará una idea sobre la evolución del site.

A la hora de comparar se deben seleccionar elementos que presenten equivalencias. Aparte de mirar valores absolutos en distintos períodos de tiempo, debemos ser capaces de valorar los resultados de páginas similares. Las páginas de menú, las fichas de producto, los formularios, etc., deben ser extraídas del log para ver información sobre el tiempo de permanencia, páginas siguientes, páginas de referencia, etc... Esta comparación resultará más sencilla cuanto más homogéneo sea nuestro site (ya que podremos aislar las variables). Un ejemplo sencillo de comparación es crear X plantillas de resultados de búsquedas, lanzar las páginas en el site y estudiando el log identificar virtudes y debilidades en los diferentes modelos.

Otra forma de utilizar el log es para monitorear campañas. Comprobar la eficacia de una oferta, comparación de productos... el log es una fuente de información válida para poder realizar test de forma rápida y económica en el desarrollo de productos. Al ser información que se colecta sin ser solicitada de forma directa, presenta algo más de validez que datos recogidos bajo formularios o encuestas. El ser capaz de representar los datos de forma comprensible para nuestra audien-

cia es un paso muy importante para mostrar la validez del sistema y poder justificar el desarrollo de esta fuente de información.

Usar capturas de pantalla. Un problema que tienen los programas que analizan los logs es que no enlazan de forma visual los resultados con las pantallas de nuestra web. Para ello, a la hora de presentar los resultados se deben usar capturas de pantalla y gráficas de tal forma que se puedan ver las modificaciones en la web y el resultado que han generado.

Información visual. En esta se concibe la fuente de información idónea para crear representaciones que ayudarán a entender y analizar las mejoras y cambios dentro de la web así como de la información expuesta.

Figura 2
Ejemplo de representación de un path

Los path de un usuario a través de nuestra web son difíciles de identificar y representar. Creando un frame con "*thumbnails*" con las páginas vistas durante la visita indicando el tiempo de estancia en cada página y enlazando estos *thumbnails* a la página real (en el frame inferior) podemos crear un sistema bastante sencillo que presenta la información de forma clara y útil ya que cualquier persona puede recrear la visita del usuario.

Comparación de menús
A la hora de comparar modelos, debemos ser capaces de enlazar la información con el producto. En este ejemplo se representan los hits en cada enlace a modo de gráfica de tarta.

- En la mitad superior se representan los tres enlaces y su distribución de hits.
- En la mita inferior se compara la eficacia en hits de los dos banners (texto Vs. teléfono).
- En el lateral se compara la cantidad de hits de los enlaces en texto frente a los "banners" gráficos.

Antes de lanzar una campaña nacional, un test en la web te puede ayudar a confirmar teorías, pulir ideas...

Distribuir la información de forma "pública". Ya sea a través de la intranet o en memos, es importante que el personal involucrado en el proyecto conozca los datos del log de visitas. También es importante explicar la validez de los datos y qué información se puede extraer de ellos. Diferentes departamentos pueden encontrar nuevas utilidades (conocer que puestos de trabajo son los más vistos, preguntas más frecuentes...), debe servir como incentivo y objetivo**.**

El análisis de* log *no es una ciencia exacta (pero se acerca). Como todo método de recogida de información, el log presenta una tolerancia en su precisión. El principal problema que tiene es que puede dar menos visitas de las que realmente se están produciendo. La caché (del navegador o del servidor proxy) hará que múltiples visitas cuenten como una única vista y que movimientos de avance y retroceso puedan no quedar registrados en él.

El otro problema es el no poder contar con la velocidad de conexión del usuario. Al no tener esta referencia, el tiempo de conexión se convierte en un parámetro algo incierto. Del tiempo de conexión ¿cuánto se dedica a la bajada de datos y cuánto

al tiempo de lectura? Difícil de adivinar, pero de nuevo la comparación nos debería dar algún índice sobre cómo poder crear una referencia.

2.6 EVALUACION DEL ESTADO DEL ARTE

La aplicabilidad de técnicas heurísticas combinadas con lo que genéricamente se ha denominado DM, es un tema de investigación que puede proporcionar una gran utilidad en la búsqueda de patrones no triviales a partir de datos disponibles. Es importante tener en cuenta que en gran porcentaje el éxito de estas técnicas radica en la información base de la que se disponga; es decir, los datos de entrada (*inputs*). Sin embargo, el DM, pese a su gran potencial como herramienta, aún no ha sido muy desarrollado para muchas aplicaciones, aunque en campos como las finanzas, las comunicaciones, los negocios y las páginas web se está aplicando desde hace algunos años.

El DM es el descubrimiento semi-automático de patrones, asociaciones, cambios, anomalías y estructuras estadísticamente significantes y eventos en los datos. Esta herramienta difiere de la estadística tradicional en cuanto esta parte de una hipótesis formulada y validada en los datos, mientras que en el proceso de DM los patrones se extraen automáticamente de los datos; dicho de otra forma, el DM es un manejo de datos mientras que la estadística es un manejo más del hombre.

El KDD es comúnmente definido como “el proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles y entendibles en los datos” (Ohrn, 1999). Involucra investigación de áreas como aprendizaje computacional, reconocimiento de patrones, bases de datos, estadística, inteligencia artificial, adquisición de conocimiento y visualización de datos.

El concepto de minería de datos (*DATA MINING*), que apareció a comienzos de 1990 motivado por el interés en este campo y su explotación en diferentes espe-

cialidades (negocios, finanzas, ingeniería, banca, salud, sistemas de energía, meteorología...), se ha incrementado recientemente debido a la combinación de diferentes factores, los cuales incluyen:

- El surgimiento de gran cantidad de datos (Terabytes= 1.012 Mbytes) debido a la medición y/o recopilación automática de datos, registros digitales, archivos centralizados de datos y simulaciones de software y hardware.
- El abaratamiento de los costos de los medios de almacenamiento.
- El surgimiento y rápido crecimiento del manejo de sistemas de bases de datos.
- Los avances en la tecnología computacional tal como los computadores rápidos y las arquitecturas paralelas.
- Los desarrollos continuos en técnicas de aprendizaje automático.
- La posible presencia de incertidumbre en los datos (ruido, *outliers*, información perdida).
- Las técnicas de minería de datos han ganado gran cantidad de adeptos en el mundo empresarial de hoy.
- Permiten explorar la información almacenada en las organizaciones para detectar patrones y modelo de comportamiento entre las variables.
- La aplicación de estas herramientas ayuda en el proceso de toma de decisión de las organizaciones**.**

Bajo los anteriores preceptos, la minería de datos busca dirigirse a dos requerimientos básicos:

- *Descripción de patrones, asociaciones y grupos de información.* Puede incluir detección de desviaciones, segmentación de bases de datos, agrupamientos, reglas, resúmenes, visualización y minado de textos.

- *Predicción de tendencias y comportamientos futuros.* La predicción incorpora tareas de clasificación, regresión y análisis de series temporales.

Se realizo una revisión de artículos y publicaciones sobre este tema. Se puede observar que la literatura no es muy amplia, y que la mayoría de los artículos provienen de centros de investigación y universidades. Estos artículos se presentarán con un enfoque ascendente, desde los intentos de realizar minería de datos sobre bases de datos relacionales, pasando por la utilización de la teoría de conjuntos aproximados, hasta llegar a lo se ha hecho o planteado para las bases de datos orientadas a objetos.

La experiencia ha mostrado que hay limitaciones en cuanto a los tiempos de respuesta y a los volúmenes de información que se pueden operar en un proceso de descubrimiento de conocimiento en bases de datos. Esto se debe a que existe un nivel débil de acoplamiento entre el manejador de la base de datos y el software que se encarga de realizar la minería de datos. Al no estar implementadas las operaciones de minería dentro del núcleo del manejador de la base de datos, se pierden muchas de las bondades que este puede entregar y se tienen que realizar tareas de transferencia de los datos a otros formatos para su evaluación e interpretación.

Un grupo de investigadores en sistemas de bases de datos de la Escuela de Ciencias de la Computación de la Universidad Simon Fraser,[19] en Canadá, planteó un modelo de lenguaje para operaciones de minería con bases de datos relacionales (DMQL), así como también un conjunto de reglas que describen la filosofía de dicho lenguaje de consulta:

1. El conjunto de datos relevantes debe ser previamente especificado.
2. Los tipos de conocimiento al ser descubiertos deben ser especificados.

19. HAN, Jiawei, *et al. DMQL: A Data Mining query language for relational databases*, 1996.

3. El conocimiento previo debe estar disponible.
4. Los resultados pueden ser expresados en términos de conceptos generalizados o múltiples niveles.
5. Varios tipos de umbrales podrían especificarse en forma flexible para filtrar el conocimiento menos interesante.[20]

Otro grupo de investigadores del Departamento de Ciencias de la Computación de la Universidad de Rutgers,[21] en New Brunswick, Estados Unidos, motivado por la no existencia de una interface de programación de aplicaciones (API, por las siglas en inglés de *Application Program Interface*) ha trabajado en una extensión de SQL, denominada por ellos *M-SQL*, que permita generar y recuperar selectivamente conjuntos de reglas de grandes bases de datos, tanto en forma de consulta libre, como inmerso dentro de un lenguaje anfitrión. M-SQL consta de una extensión a SQL que incluye únicamente la primitiva MINE, un optimizador para compilar los planes de ejecución de M-SQL, y la interface de programación de aplicaciones, donde M-SQL puede ejecutarse desde dentro de programas escritos en el lenguaje C++.

Se puede decir que M-SQL, al ser una extensión de SQL presenta un ambiente de trabajo más amigable o asimilable para un usuario determinado que conozca la estructura de SQL. Sin embargo, la tendencia es tratar de llevar el poder de cómputo al usuario final, y las tareas de minería de datos no pueden ser la excepción, así que se está trabajando también en lograr una buena interface de usuario, de forma gráfica, estilo Windows. El M-SQL es una de las primeras aproximaciones en este sentido, al igual que *DBMiner*, una implementación sobre DMQL.[22]

20. Ibíd.
21. IMIELINSKI, Tomas, *et al. Datamine: application programming interface and query language for database mining*.
22. HAN, Jiawei, *et al. DBMiner: A system for mining knowledge in large relational databases*.

Las dos aproximaciones anteriores representan un buen avance en la definición de una arquitectura de base de datos que permita realizar KDD e incluir un lenguaje de consulta con estas instrucciones. No queda claro en ellas cuáles son los nuevos operadores especializados que han implementado en el álgebra o cuáles cambios han sido necesarios realizar en el optimizador.

Entre las ventajas consideramos

- Velocidad de procesamiento: Esta velocidad se alcanza al desarrollar archivos para aplicaciones específicas y sencillas.
- Reducción de problemas de disponibilidad de datos y de seguridad.
- Integridad de los datos.
- Acceso a múltiples usuarios y aplicaciones.
- Flexibilidad y estandarización que permite a los usuarios centrar su atención en los datos y no en la estructura y ubicación física del sistema de almacenamiento.
- Data Mining, tiene sencillez y rapidez, dichas técnicas permiten trabajar con grandes cantidades de ejemplos (varios millones) sin ningún inconveniente.

- El Data Mining parte de la segmentación y análisis de las bases de datos para definir acciones, muy orientadas hacia el cliente; y su objetivo es el estudio consecuente de dicha información para hallar vínculos en un tiempo razonable.
- Permite una mayor flexibilidad y rapidez en el acceso a la información.
- Transforma los datos en información y la información en conocimiento.
- Permite hacer planes de forma más efectiva.
- Reduce los tiempos de respuesta y los costes de operación.

En las desventajas encontramos;

- Falta de potencialidad para evolucionar:el desarrollo de sistemas de archivos va ligado generalmente a soluciones de necesidades inmediatas, y ante estas situaciones de necesidades de información que involucran combinación de

atributos y datos entre archivos separados, no se podrían dar soluciones con este enfoque.

- Duplicidad de datos: La necesidad de integrar la mayor parte de atributos a un archivo en específico, para evitar problemas de consulta posteriores, es común observar la presencia de datos duplicados entre archivos, hecho que convierte la actualización de los datos en un proceso largo y costoso.
- Las desventajas del modelo se centran principalmente en sus costos, debido que en funcionalidad es realmente eficiente**.**
- Requieren una revisión del modelo de datos, objetos, transacciones y además del almacenamiento.
- Tienen un diseño complejo y multidisciplinar.
- Requieren una reestructuración de los sistemas operacionales.
- Requieren sistemas, aplicaciones y almacenamiento específico.

CAPÍTULO 3
EL SOFTWARE UTILIZADO

3.1 EL REQUERIMIENTO

Actualmente, uno de los principales problemas en *web usage* tiene que ver con la fase de preprocesado de los datos, previa aplicación de alguna técnica de *DATA MINING*. Los servidores web generalmente registran una entrada en un fichero log para cada acceso. La mayor parte de la información de acceso al *website* está en el fichero log. La información relevante para uso web se almacena en ficheros que pueden analizarse en diferentes formas. Los componentes más comunes de un fichero log incluyen: dirección IP, hora de acceso, método de interrogación, URL de la página a la que se ha accedo, protocolo.[23]

Para realizar estos estudios se utilizará dos herramientas una desarrollada por el grupo y otra herramienta existente para la producción de la información estadisitica que permita determinar las reglas que definen los hábitos y comportamiento de los usuarios o clientes de su sitio web.

3.2 EL PROGRAMA PROTOTIPO DESARROLLADO PARA HACER LA EXTRACCION

El programa Analogs que hemos implementado fue desarrollado en Lenguaje C para que haga un escaneo sobre el archivo logs.html y que transforme la información que es relevante y la cargue en una lista que deseche lo que no es relevante, tal como las etiquetas html y su estructura. El archivo logs.html es un archivo que

23. MARBAN GALLEGO, Oscar. Resumen del trabajo de investigación.

posee los logs o registros de auditoría del ingreso y salida de usuarios a un servidor y de las acciones que en la Internet ellos realizan.

Lo primero que hace nuestro algoritmo es analizar la estructura de los logs en el archivo html y luego crea una estructura para cargar en la lista cada uno de los logs del archivo. Para poder hacer esto depuramos el archivo html cuatro veces y se va cargando la información relevante en cuatro archivos textos, uno a la vez y el último más depurado que el anterior, el último de estos es el archivo logs.txt que es el que le sirve de base al algoritmo para hacer el DM.

Una vez depurado el archivo html y transformada la información a un archivo texto, el algoritmo procede a construir la estructura de los logs con la información relevante para tal fin. Una vez cargada la lista con los diferentes log, es posible hacer minería de datos y escudriñar hasta encontrar la información que se requiere. Para ello se puede listar cualquiera de los logs o sólo parte de ellos o incluso una parte específica de ellos, Ej.: las fechas. Después de esto se puede saber cuál es la cantidad de log que se construyeron en el proceso y extraer información de los usuarios; por Ej.: si se han *logueado* últimamente, qué acciones realizaron y en qué fecha y hora. También se puede preguntar por un instante determinado para saber qué acciones se realizaron en ese momento, Ej.: si alguna acción se hizo sin permiso en un instante determinado, se puede saber quién la hizo. De esta misma manera se puede agregar cualquier tipo de consulta que se requiera.

La minería de datos se facilita gracias a todo el proceso de depuración y carga de los logs que hace el algoritmo a la lista. En últimas, lo que se logra con el algoritmo es permitir tener un puntero a cada uno de los campos que hacen parte de los logs que se encuentran en la lista, en un momento determinado y según la necesidad del usuario, y como resultado siempre tendremos la información que necesitemos con respecto a tales campos, sin importar la cantidad de log que haya resultado de la depuración.

3.2.1 MACROALGORITMO ANALOGS

- **Estructura del Log**

```
{
        char Mes(4);
        char Dia(3);
        char Hora(3);
        char Min(3);
        char Seg(3);
        char Servidor(23);
        char ID(36);
        char Accion(40);
        char User(10);
};
```

Se incluye la biblioteca lista_ap.h y se declara la lista fecha.

- **Prototipos de las funciones:**

```
void presentacion(void);
void cargar_archivos(void);
void analisis_logs(void);
void consola_administracion(void);
void main(void)
{
        presentacion();
        cargar_archivos();

        fecha = inic_lista();
        analisis_logs();
}
```

void cargar_archivos(void)
{

- Se abre el archive logs.html y se realiza la primera depuración colocando el archivo logs2.txt sin las etiquetas html del archivo original.

- Se hace la segunda depuración colocando los retornos de carro al final de cada uno de los logs desde el archivo logs2.txt al archivo logs3.txt, lo cual permite separarlos unos de otros.

- En las siguientes dos depuraciones se desecha la información que no hace parte de los logs y que se encuentra en el archivo logs3.txt. El resultado son los archivos logs4.txt y logs.txt, este último contiene los logs depurados y que finalmente serán el objetivo de la minería de datos.

}
void analisis_logs(void)
{

Se carga en cada uno de los campos de la estructura logs, la información de cada uno de los logs y uno a uno es almacenado en la lista para que puedan ser analizados.

}
void consola_administracion(void)
{

Se le entrega al usuario la posibilidad de analizar los logs, en este caso tiene 4 opciones diferentes de hacer minería de datos sobre los logs cargados en la lista:

1. Número de logs
2. Listar los logs
3. Mostrar si un usuario se ha logueado.
4. Mostrar un log en una fecha y hora específica.

5. Salir.

}

A continuación nos permitimos presentar los diagramas de flujo correspondiente en las figuras:

- En la figura 5 la estructura del modulo principal de nuestro programa.
- La figura 6 presenta el modulo que permite el calculo de numero de logs.
- La figura 7 el modulo que permite el listar los logs
- La figura 8 presenta el algoritmo para determinar si un usuario ha firmado en el sistema.
- Y la figura 9 el modulo que permite mostrar el log en una fecha y hora especifica.

Para ampliar la información sobre instrucción de cómo se maneja, anexamos el manual de operación del software como tambien el codigo fuente del programa.

3.2.2 DIAGRAMAS DE FLUJO DE ANALOGS

Figura 3
Dagrama principal

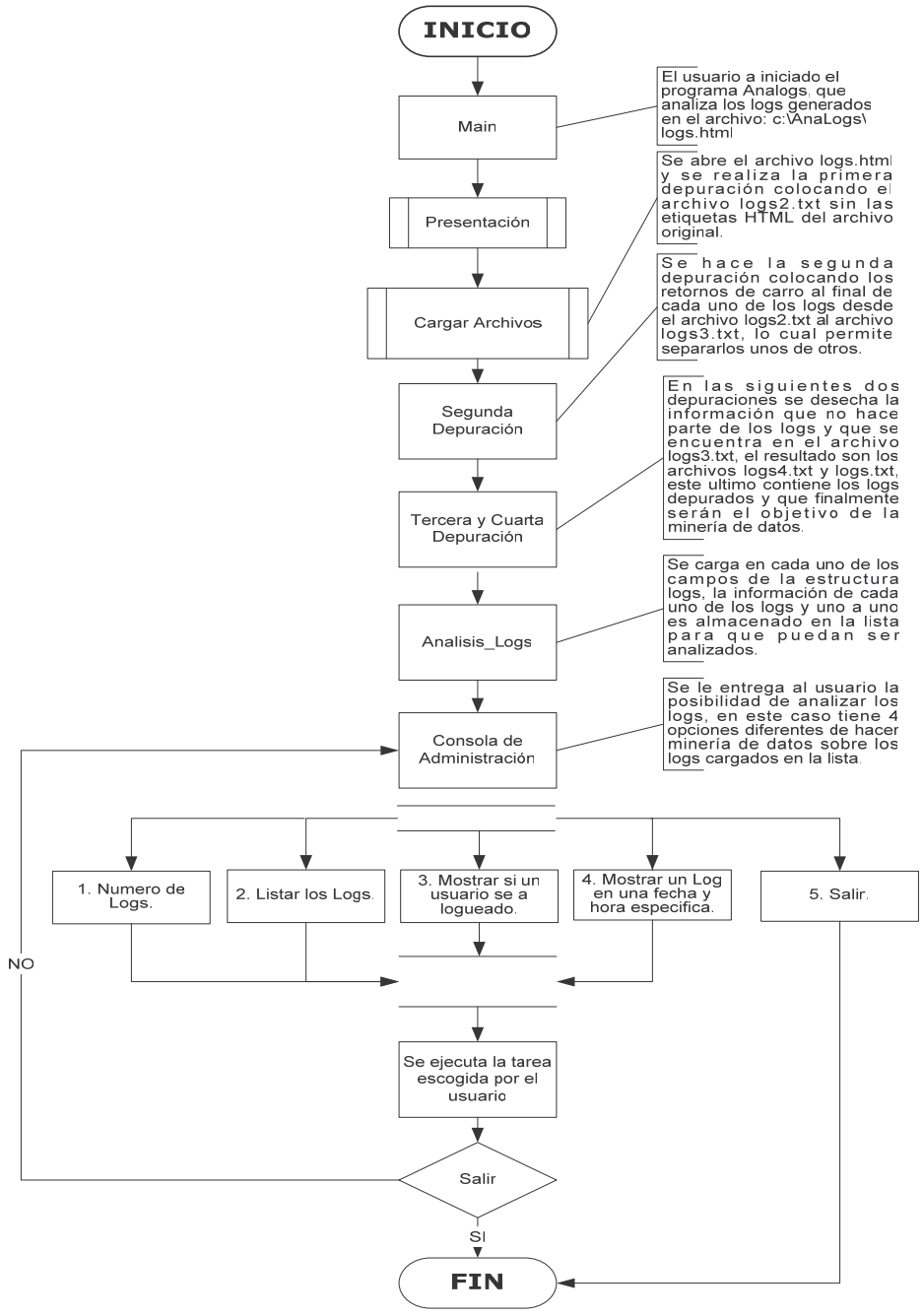


Figura 4
Número de logs

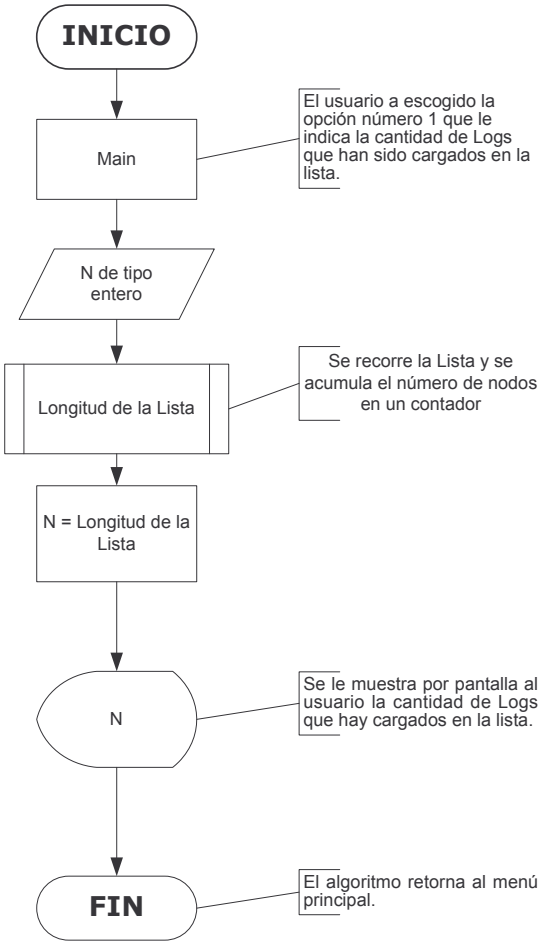


Figura 5
Listar los logs

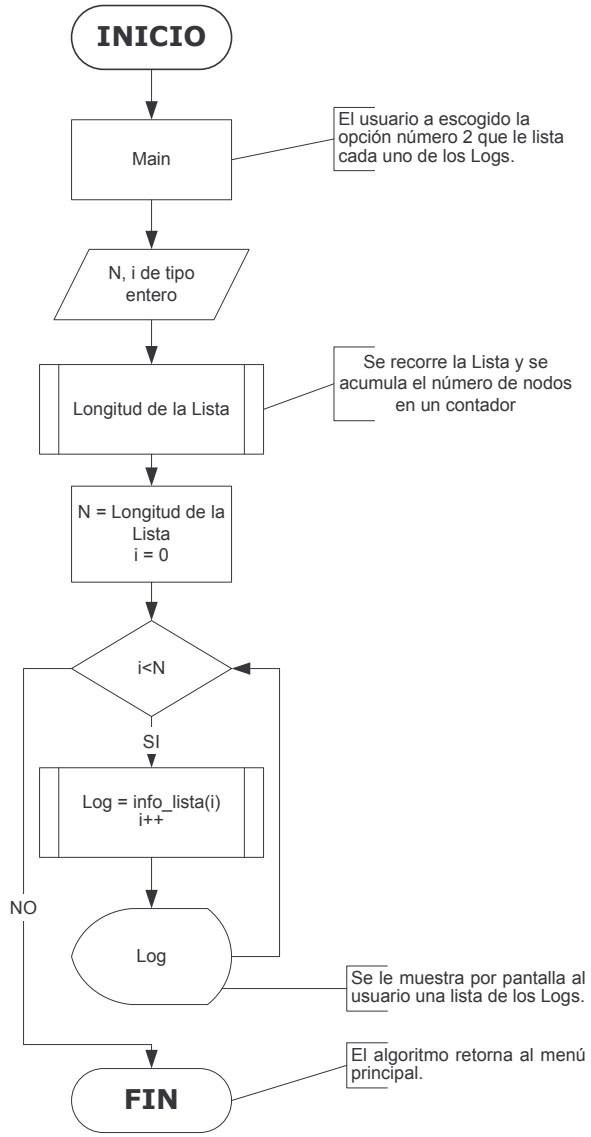


Figura 6
Usuario logueado

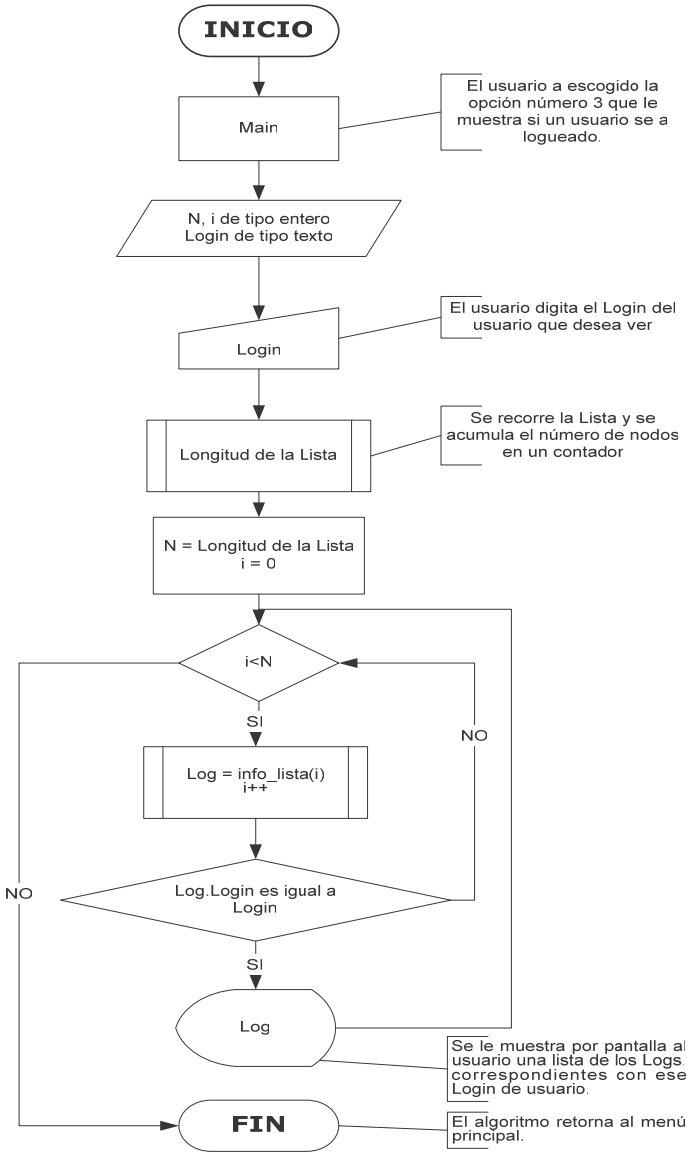
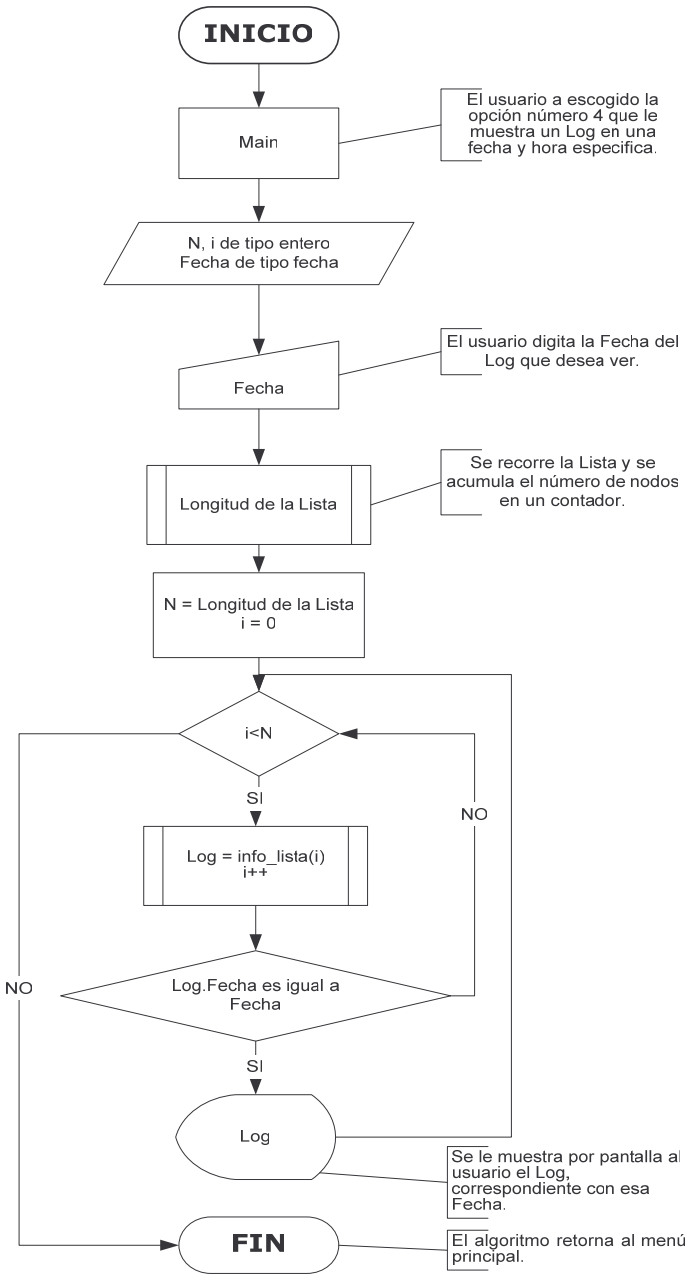


Figura 7
Mostrar un log con fecha y hora



3.3 EL SOFTWARE USADO PARA GENERACION DE REPORTES Y GRAFICOS

3.3.1 El Primer candidato: El ANALOG

El ANALOG es un software de distribución libre, para analizar los archivos log del servidor que funciona con cualquier sistema de Windows y cualquier tipo de servidor web. Es totalmente personalizable, para ajustarlo a las necesidades.

En su versión 5.22, el generador de estadísticas ANALOG es uno de los analizadores de archivos de log mas potentes. Genera informes rápidos y exactos sobre el sitio Web ofreciendo valiosa información sobre sus visitas. El programa, que soporta más de 30 idiomas distintos (entre ellos la lengua española) no tiene un interfaz de usuario gráfico, pero es relativamente sencillo de controlar.

Los análisis se pueden visualizar con un navegador Web cualquiera .Es muy rápido y ofrece una amplísima variedad de detalles e información sobre el tráfico y rendimiento del servidor y genera un completo informe incluyendo estadísticas mensuales, diarias y horarios, el origen de las visitas, las páginas mas visitadas, los archivos servidores, reporte de errores, palabras claves utilizadas en los buscadores, navegadores y sistemas operativos, informe de peticiones

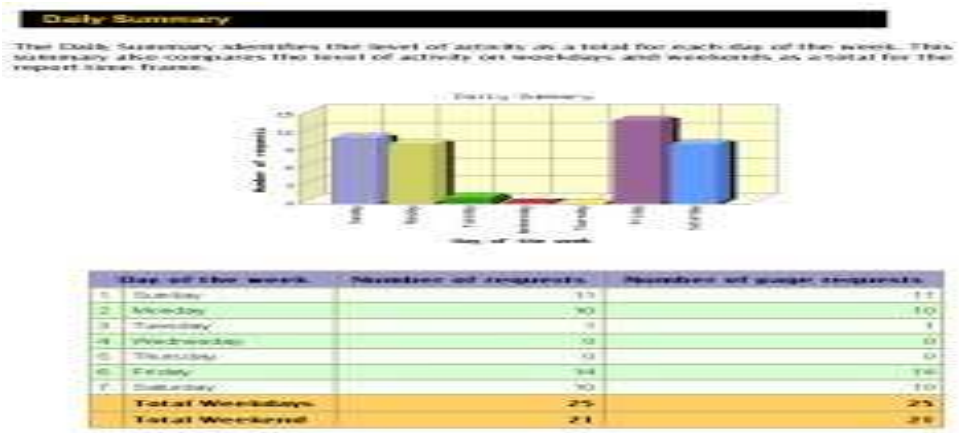
La magia del informe hace estadística usable del análisis del fichero de diario del Web site. Puede contestar preguntas sobre su sitio Web:

¿De dónde vienen los visitantes?
¿Qué palabras y motores de búsqueda se utilizan?
¿Tiempos del tráfico de punta?
¿Las estaciones lentas?
¿Qué browsers los visitantes están utilizando?
¿Qué sistemas operativos utilizan mis visitantes?

¿Hay acoplamientos quebrados en mi sitio u otros sitios que se ligan los míos?
¿Qué páginas los visitantes solicitan en la mayoría?
¿Qué ISP o compañías mis visitantes conectan?

La magia del informe puede contestar a estas preguntas y ayudarle y al resto de la organización a entender cómo se utiliza su Web site.

Muestra Informes en forma grafica como el siguiente:



3.3.2 El Segundo candidato: El WEBALIZER

El WEBALIZER es un software de distribución libre, encontrada en Internet en el sitio http://www.mrunix.net/webalizer/ para analizar los archivos log del servidor. Produce informes altamente detallados, fácilmente configurables en formato HTML, para verlos con un Web Browser estándar. Fue escrito para solucionar varios problemas que se tenían con los paquetes actualmente disponibles. La mayoria de ellos estaba escrita en el Perl o una cierta lengua encriptada. Algunos de ellos no estaban libres para su uso. Algunos producían resultados incorrectos, incluso los producidos, no estaban en un formato que fuera útil.

Para conseguir un formato mas adecuado, se hizo el WEBALIZER, el cual puede ser utilizado por cualquier persona que lo desee, para cualquier propósito. Comenzó como una escritura simple Perl con capacidades limitadas, pero ha crecido como una herramienta robusta y rápida de análisis, siendo utilizado por millares de sistemas alrededor del mundo. Fue escrito en C para ser extremadamente rápido y altamente portable. Se necesita una máquina Pentium de 200 Mhz, se puede procesar en un segundo 10.000 expedientes, con un archivo de 40 megabytes puede tomar 15 segundos (sobre 150.000 expedientes). Maneja registros comunes del servidor del formato del logfile del estándar. Además, varias variaciones del formato combinado del logfile se manejan, permitiendo que la estadística sea generada para referir los tipos de los sitios y del browser también. Los informes generados se pueden configurar en la línea de comando, o por medio de unos o más archivos de la configuración.

Se manejan los tamaños ilimitados del fichero de diario y los registros parciales, permitiendo que los registros sean rotados tan a menudo como se necesite y elimina la necesidad de guardar archivos mensuales enormes en el sistema. Distribuido bajo la licencia libre en general de GNU, el código de fuente completo está disponible, así como las distribuciones binarias para algunas de las plataformas más populares.

3.3.3 El tercer candidato: WEBLOG SESSIONIZTOR Xp 5.0 Beta

El WebLog Sessionizator XPert (WLS XP) es un software de preprocesamiento, transformación y análisis para tareas de Web Mining que involucran la obtención de conocimiento a partir del análisis de la información de comportamiento de los usuarios de un sitio Web contenida en los archivos Log de acceso al servidor (Log Files). WLS XP es la última y más avanzada versión de la familia de productos WLS.

La versión 5.0 de WLS XP incluye una serie de mejoras comparadas con la versión previa de WLS XP

Entre las características más destacadas de WLS XP destacan:

- Mejor Rendimiento gracias a las mejoras de WLS 3.01b que harán ahorrar hasta un 40% de tiempo de procesamiento y sesionización.
- Preprocesamiento más rápido que harán ahorrar hasta un 50% de tiempo comparado con WLS 3.01b.
- Sesionización más rápida que harán ahorrar hasta un 40% de tiempo comparado con WLS 3.01b.
- Nueva Interfaz Modal que le permitirá manejar y administrar sus proyectos de WLS con más facilidad.
- Soporte para CLF y ELF, permitiendo procesar distintos formatos de Log Files, procedentes de distintos servidores.
- Mayor Soporte para URLs largas y con consultas dinámicas (ideal para páginas ASP, JSP, PHP, etc).
- Administración Centralizada de proyectos de WLS. Crea un proyecto nuevo y ud. lo guarda y vuelve a utilizarlo las veces que quiera.
- Nuevo Asistente de Proyecto que facilita la creación de proyectos WLS indicando cada paso.
- Reglas de Asociación. Descubra asociaciones entre sus páginas y sepa lo que verdaderamente está pasando en su sitio Web.

El WLS en tan sólo 3 pasos usted puede transformar los datos en bruto de su sitio Web en información útil para Web Mining, o analizar la actividad de los usuarios de su sitio web mediante la realización de consultas SQL.

Estos son los pasos que se deben seguir para utilizar WLS.

Paso 1 Preprocesamiento: Procese el Archivo Log

Paso 2 Sesionización: Transformación del Archivo Log a Sesiones.

Paso 3 Web Mininig: Obtenga Reglas que describen el comportamiento de los usuarios de su sitio.

3.3.4 Nuestra Decisión

Decidimos usar para la presentación de los informes estadísticos el software ANALOG 5.22, una herramienta encontrada en Internet en el sitio http://analog.gsp.com/ que luego de evaluarlo consideramos que cumplía con las expectativas deseadas sobre los reportes y gráficos que eran de interés para el departamento de redes de la universidad y que se podía ser alimentado con la información que usando nuestro programa ANALOGS habíamos extraído de los archivos de LOGS del servidor Web de la universidad.

CAPÍTULO 4
CASO DE APLICACIÓN: ESTADISTICAS WEB UNIVERSIDAD AUTÓNOMA DE OCCIDENTE

Este informe pretende realizar un análisis de las estadísticas obtenidas del sitio web de la Universidad Autónoma de Occidente y acopiar información valiosa sobre sus tendencias de funcionamiento y uso. Se utiliza el webalizer sobre la base de datos suministrada por la universidad.

- **Análisis de peticiones del Jue/22/Jul/2004 - 1:02 pm al Jue/29/Jul/2004 12:52 pm (6.99 días).**

Resumen general

Este reporte contiene estadísticas globales.

- Peticiones exitosas: 136,121
- Promedio de peticiones exitosas por día: 19,465
- Peticiones exitosas por página: 17,209
- Promedio de peticiones exitosas por página por día: 2,460
- Peticiones no logradas: 8,441
- Peticiones redireccionadas: 3,790
- Archivos diferentes solicitados: 3,826
- Sistemas diferentes atendidos: 4,692
- Líneas no válidas en el archivo de registros: 2
- Transferencia total: 687.88 megabytes
- Promedio de transferencia por día: 98.37 megabytes

El resumen general contiene estadísticas globales sobre todos los datos analizados, siendo los más importantes el número de peticiones exitosas (incluye el número total de archivos descargados y gráficas); el número de requerimientos por páginas; el número de diferentes hosts que hacen peticiones sobre el sitio; y la cantidad de datos transferidos en bytes.

Hay que tener en cuenta que un usuario puede generar muchas peticiones viendo diferentes páginas o imágenes o simplemente viendo la misma página muchas veces.

Informe mensual

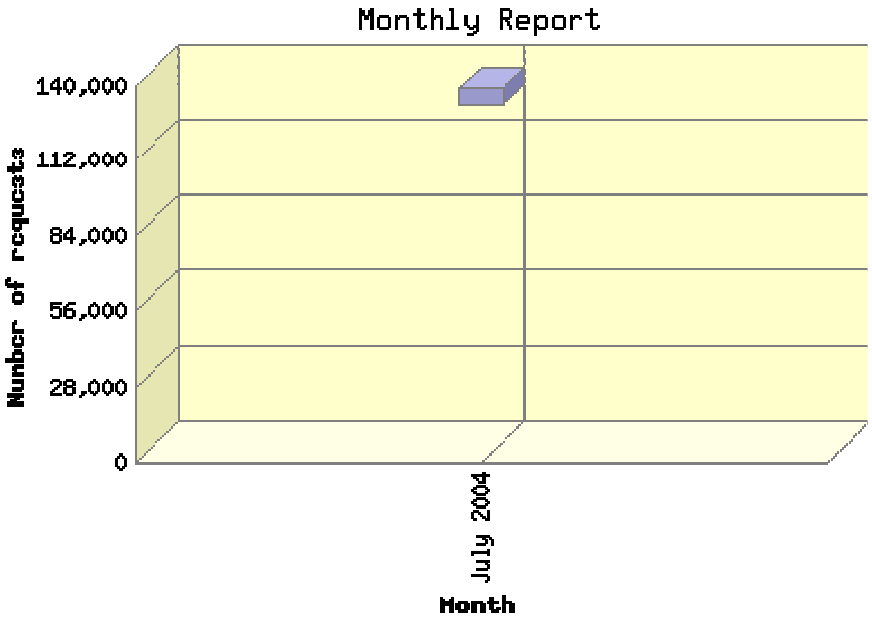
Este reporte enlista la actividad en cada mes.

Cada unidad (▬) representa 500 peticiones por páginas o fracción.

Mes:	No. pet.:	Págs.:
--------:	--------:	--------:
Jul 2004:	136.121:	17.209: ▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬

Mes de mayor tráfico: Jul 2004 (17,209 peticiones por páginas).

Figura 8
Informe mensual



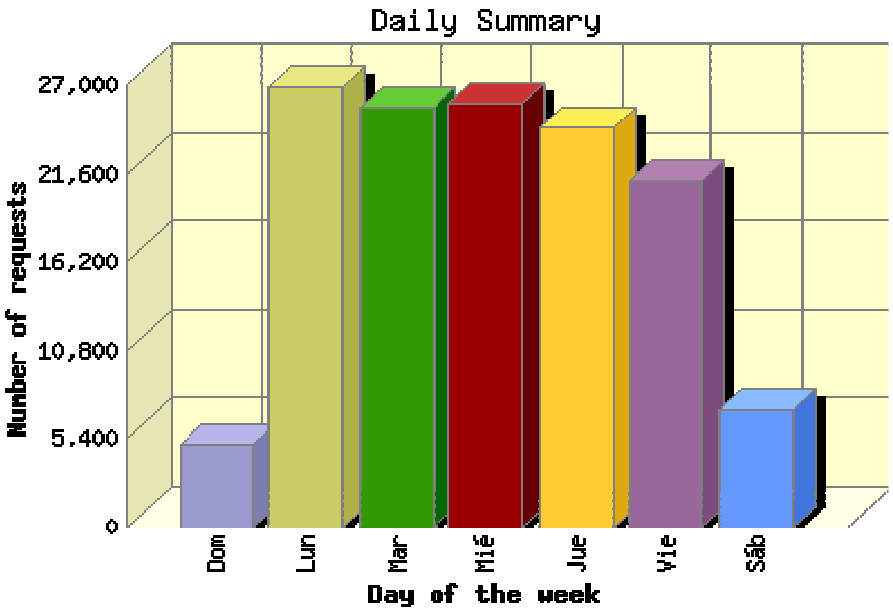
Resumen diario

Este reporte enlista la actividad total por cada día de la semana, sumados por todas las semanas en el reporte.

Cada unidad (▬) representa 80 peticiones por páginas o fracción.

Día:	No. pet.:	Págs.:	
Dom:	5.092:	896:	▬▬▬▬▬▬▬▬
Lun:	26.865:	3.416:	▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬
Mar:	25.612:	3.124:	▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬
Mié:	25.870:	2.959:	▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬
Jue:	24.360:	2.959:	▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬
Vie:	21.186:	2.726:	▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬▬
Sáb:	7.136:	1.129:	▬▬▬▬▬▬▬▬

Figura 9
Resumen diario



Este reporte muestra cuántas peticiones fueron hechas en cada período de tiempo. De acuerdo con lo anterior podemos ver que los días en que más se realizan peticiones son los hábiles (lunes a viernes) y los días que menos peticiones se realizan son los de los fines de semana (sábado y domingo).

La tendencia indica que este sitio es más visitado por personas que cuentan con una alta disponibilidad de acceso a la Internet desde sus lugares de trabajo.

Igualmente, son múltiples las peticiones generadas por la comunidad universitaria de la Cuao (funcionarios, profesores y estudiantes) en días de actividad académica (lunes a viernes), empleando como recurso principal el acceso a la Internet la disponibilidad de salas de sistemas que funcionan al interior de la universidad desde las 7:00 am hasta las 09:30 pm.

Los fines de semana disminuye ostensiblemente las peticiones por ser días de descanso, donde las personas se encuentran realizando actividades diferentes a las laborales o académicas y tanto empresas como universidades se encuentran fuera de servicio.

De acuerdo con lo anterior, se sugiere garantizar una alta disponibilidad del sitio web durante los días de semana (lunes a viernes) y cualquier trabajo de administración y mantenimiento que requiera de la suspensión del servicio, debe ser realizado al final de la semana, de acuerdo con las estadísticas generadas.

Resumen horario

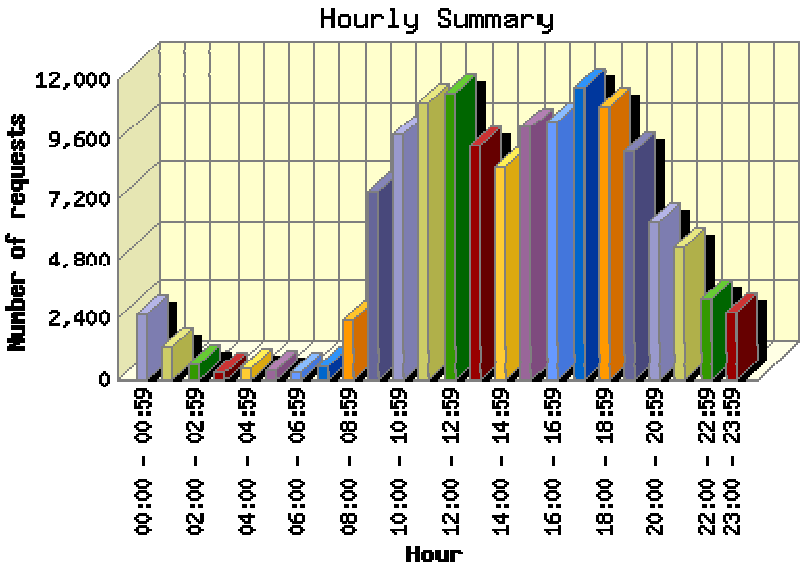
Este reporte enlista la actividad total por cada hora de la semana, sumados por todos los días en el reporte.

Cada unidad (▬) representa 40 peticiones por páginas o fracción.

Hr:	No. pet.:	Págs.:	
0:	2.620:	384:	▬▬▬▬▬▬▬▬▬
1:	1.311:	237:	▬▬▬▬▬▬
2:	577:	149:	▬▬▬▬
3:	292:	106:	▬▬▬
4:	450:	124:	▬▬▬▬
5:	393:	109:	▬▬▬
6:	305:	109:	▬▬▬

7:	528:	136:	
8:	2.355:	336:	
9:	7.528:	907:	
10:	9.790:	1.137:	
11:	11.070:	1.315:	
12:	11.489:	1.410:	
13:	9.338:	1.156:	
14:	8.476:	1.000:	
15:	10.138:	1.272:	
16:	10.325:	1.258:	
17:	11.696:	1.392:	
18:	10.943:	1.314:	
19:	9.080:	1.113:	
20:	6.302:	807:	
21:	5.284:	630:	
22:	3.164:	416:	
23:	2.667:	392:	

Figura 10
Resumen horario



Este reporte nos enseña el número total de peticiones en cada hora del día o en cada período del día, realizando la sumatoria sobre todos los días en el reporte. (Este no es un promedio, tampoco es la información del último día).

De acuerdo con lo anterior se puede ver que las horas en las que más se realizaron peticiones fueron las 12:00 del mediodía y las 5:00 de la tarde. Estas dos horas presentan algo en común y es que en ellas se suspenden las labores cotidianas (estudio o trabajo) bien sea por ser la hora de descanso o porque se finaliza la jornada laboral.

Tabla 1
Resumen horario

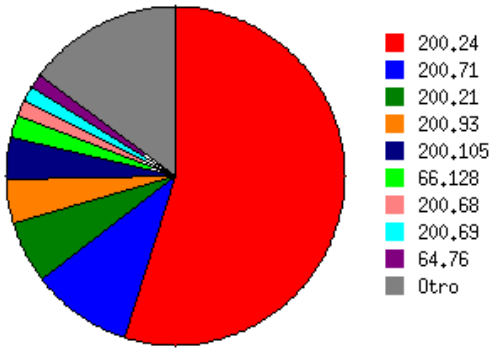
	Hour	Number of requests	Number of page requests
1.	00:00 - 00:59	2,620	384
2.	01:00 - 01:59	1,311	237
3.	02:00 - 02:59	577	149
4.	03:00 - 03:59	292	106
5.	04:00 - 04:59	450	124
6.	05:00 - 05:59	393	109
7.	06:00 - 06:59	305	109
8.	07:00 - 07:59	528	136
9.	08:00 - 08:59	2,355	336
10.	09:00 - 09:59	7,528	907
11.	10:00 - 10:59	9,790	1,137
12.	11:00 - 11:59	11,070	1,315
13.	12:00 - 12:59	11,489	1,410
14.	13:00 - 13:59	9,338	1,156
15.	14:00 - 14:59	8,476	1,000
16.	15:00 - 15:59	10,138	1,272
17.	16:00 - 16:59	10,325	1,258
18.	17:00 - 17:59	11,696	1,392
19.	18:00 - 18:59	10,943	1,314
20.	19:00 - 19:59	9,080	1,113
21.	20:00 - 20:59	6,302	807
22.	21:00 - 21:59	5,284	630
23.	22:00 - 22:59	3,164	416
24.	23:00 - 23:59	2,667	392
	Work hours (8:00 am - 4:59 pm)	80,509	9,791
	After hours (5:00 pm - 7:59 am)	55,612	7,418

En la anterior tabla podemos visualizar un aspecto importante y es que a pesar que las horas donde se generan la mayor cantidad de peticiones son horas no laborales, esto no quiere decir que entre las 8 am y las 5 am se genere menor actividad en el sitio web. El número de peticiones sumadas en todas las horas laborales supera ampliamente al número de peticiones generadas en el horario no laboral, con lo que ratificamos la tesis planteada en el resumen diario donde se plantea el uso de este sitio web desde sitios de trabajo y estudio en días laborales.

Informe de organización

Este reporte enlista las organizaciones quienes pidieron archivos.

Figura 11
Informe de organización



Las áreas están dibujadas por número de peticiones.

Listado de las 20 primeras organizaciones, ordenadas por número de peticiones.

Tabla 2
Informe de organización

	Organization	Number of requests	Percentage of the bytes
1.	200.24	74,652	49.81%
2.	200.71	13,052	4.78%
3.	200.21	8,087	8.09%
4.	200.93	5,744	2.89%
5.	200.105	5,402	2.75%
6.	66.128	2,904	3.06%
7.	200.68	2,162	1.03%

8.	200.69	2,001	0.98%
9.	64.76	1,949	1.37%
10.	200.119	1,221	1.45%
11.	200.91	1,144	1.04%
12.	200.115	1,138	0.87%
13.	200.31	994	0.58%
14.	200.30	894	0.37%
15.	200.74	858	0.55%
16.	200.58	769	0.32%
17.	209.88	585	0.47%
18.	200.14	534	0.91%
19.	200.75	380	0.57%
20.	206.48	351	0.17%
	(no listadas: 922)	11,300	17.96%

Este reporte lista las organizaciones (compañías, instituciones, ISPs - Internet Service Providers, etc.) cuyas computadoras accedieron al sitio web.

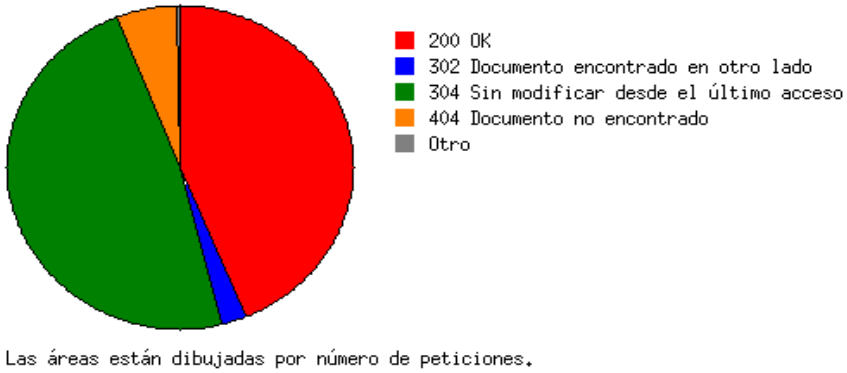
Con esto se puede concluir que la mayoría de las peticiones (49.81%) fueron realizadas desde el interior de la universidad, empleando la red LAN (*Local Area Network*) y atravesaron el servidor proxy. Estas peticiones fueron procesadas a través de NAT (*Network Address Transalation*) y enviadas hacia el servidor web con una dirección válida (200.24.X.X).

Se recomienda verificar los factores que determinan la velocidad de acceso desde la red LAN de la universidad, como los medios de transmisión (cableado), el ancho de banda, equipos de la red de datos (switches, routers, hubs), desempeño del servidor proxy, computadoras y sistemas operativos, pues son los principales clientes de este sitio web.

Informe de códigos de estado

Este reporte enlista los códigos de estado http de todas las peticiones.

Figura 12
Codigos de estado



Listado de los códigos de estado por orden numérico.

Tabla 3
Códigos de estado

	Status Code	Number of requests
1.	200 OK	64,833
2.	206 Contenido parcial	194
3.	301 Documento desplazado permanentemente	238
4.	302 Documento encontrado en otro lado	3,552
5.	304 Sin modificar desde el último acceso	71,094
6.	404 Documento no encontrado	8,431
7.	405 Método no permitido	5
8.	408 petición interrumpida	5

El informe de códigos de estado visualiza las cabeceras html retornadas al cliente desde el servidor. Por ejemplo el estado OK significa que la página requerida fue encontrada en el servidor y será enviada. Un código 404 significa que la página o imagen requerida no pudo ser encontrada sobre este servidor en la ubicación especificada. Esto puede ocurrir si un cliente digita mal un URL o si hace clic sobre un enlace defectuoso.

Igualmente, la mayoría de las peticiones corresponde a componentes del sitio web que han permanecido por largo tiempo sin ser modificados.

Por la arquitectura de sitio estudiado se puede ver como este servidor se encuentra enlazado a información que no está alojada localmente y algunas peticiones deben ser resueltas en otros servidores.

Informe de tamaño de archivos

Este reporte categoriza el tamaño de los archivos que han sido requeridos.

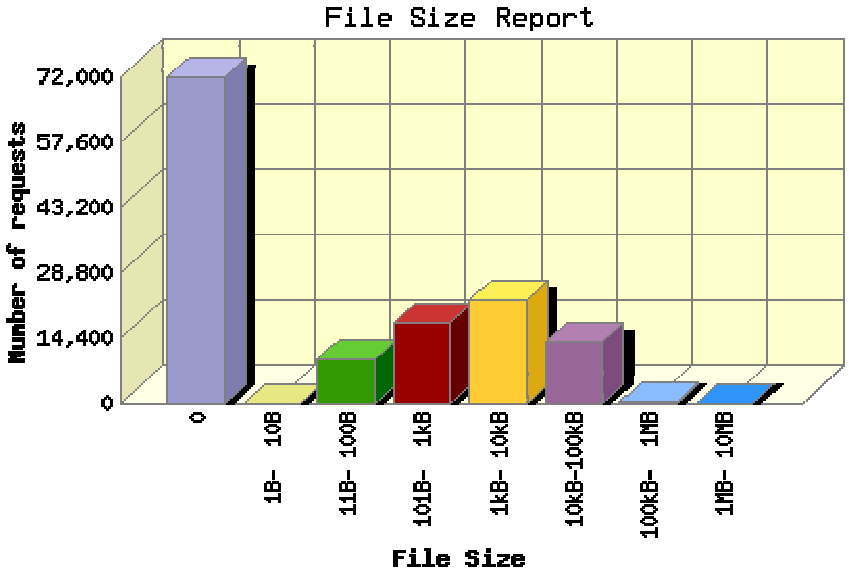
Figura 13
Tamaño de archivos

Tabla 4
Tamaño de archivos

File Size		Number of requests	Percentage of the bytes
1.	0	71,883	0.00%
2.	1B- 10B	14	0.00%
3.	11B- 100B	9,709	0.06%
4.	101B- 1kB	17,676	1.08%
5.	1kB- 10kB	22,900	11.53%
6.	10kB-100kB	13,511	55.80%
7.	100kB- 1MB	380	12.89%
8.	1MB- 10MB	48	18.63%

El elevado número de peticiones a archivos menores a un kilobyte se debe a la configuración particular del sitio web de la universidad, donde las peticiones son recibidas por un servidor front-end y a través de un archivo ligero de flash direcciona las peticiones a su portal.

Figura 14
Tamaño de archivos



Es importante resaltar que hay un porcentaje significativo de peticiones a archivos que tienen un tamaño entre 1 y 10 MB, lo que en algunos momentos puede ocasionar disminución de la velocidad de descarga de los archivos solicitados y en general del sitio web.

Se debe revisar el formato de estos archivos y evaluar la posibilidad de cambiar su formato por uno más liviano, teniendo en cuenta que se encuentran almacenados archivos y presentaciones de multimedia como vídeos, imágenes, sonidos, entre otros.

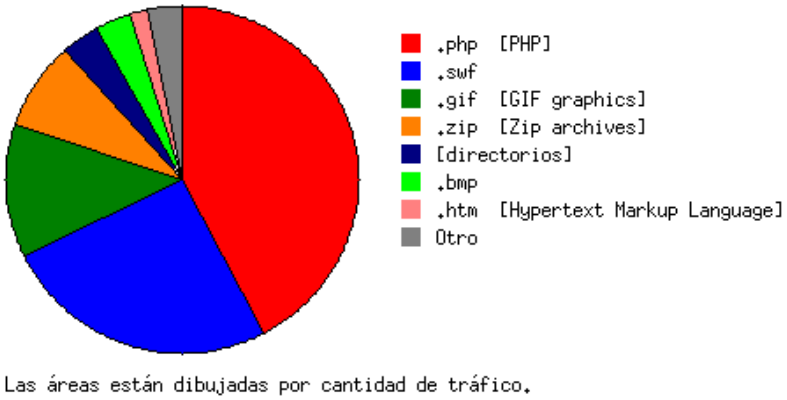
Esta información puede ser usada para optimizar el rendimiento del sitio web, teniendo en cuenta que el equipo de comunicaciones empleado por los visitantes es

un modem a 28.8 Kbps, el cual usa un promedio de 40 segundos para descargar 100 Kb.

Informe de tipos de archivo

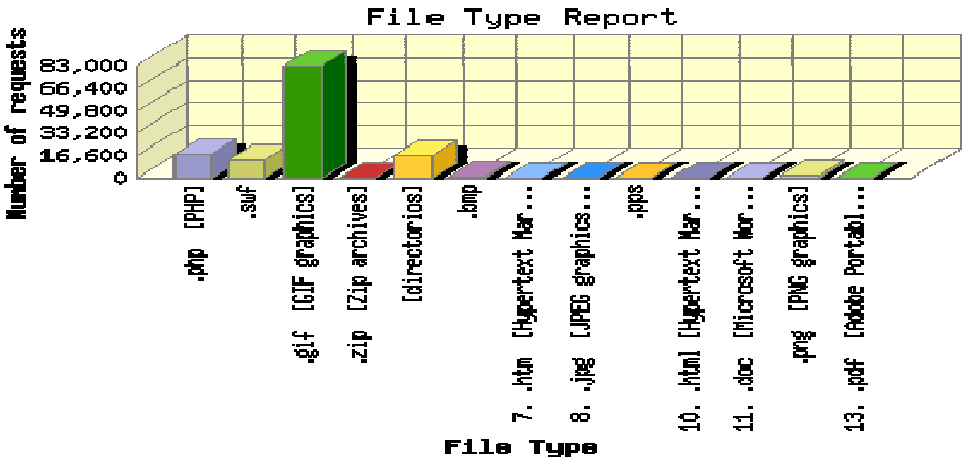
Este reporte identifica el tipo de información que es requerida al sitio web. Gif y jpg son dos clases de gráficos (imágenes) que son comúnmente soportadas por los buscadores web html, asp, etc.

Figura 15
Tipos de archivos



El sitio web se encuentra diseñado con un amplio contenido en imágenes, lo cual justifica la mayoría de peticiones a este tipo de archivos, pero adicional a esto se encuentran archivos con múltiples formatos como componentes de multimedia diseñados en flash (.swf).

Figura 16
Tipos de reportes



Otros archivos aunque con menos porcentaje son importantes para el funcionamiento del sitio y entregan información valiosa en diferentes formatos.

Uno de ellos son los formatos .doc que son empleados para publicar formularios de inscripción, circulares informativas y materiales educativos. Existen otros formatos empleados para fines similares como los .zip, .pdf y .pps.

Tabla 5
Tipos de reporte

File Type		Number of requests	Percentage of the bytes
1.	.php (PHP)	18,477	42.22%
2.	.swf	13,881	25.43%
3.	.gif (GIF graphics)	82,453	12.43%
4.	.zip (Zip archives)	14	8.34%
5.	(directorios)	16,562	3.47%
6.	.bmp	1,368	3.41%
7.	.htm (Hypertext Markup Language)	502	1.60%
8.	.jpg (JPEG graphics)	384	0.89%
9.	.pps	29	0.82%
10.	.html (Hypertext Markup Language)	145	0.50%
11.	.doc (Microsoft Word document)	33	0.27%
12.	.png (PNG graphics)	1,952	0.23%
13.	.pdf (Adobe Portable Document Format)	30	0.16%
	(no listadas: 12)	291	0.24%

Los archivos de formato gráfico no pueden faltar en un sitio web y es por esto que se encuentran algunos .gif, .jpg, .png y .bmp. empleados en el diseño de páginas web.

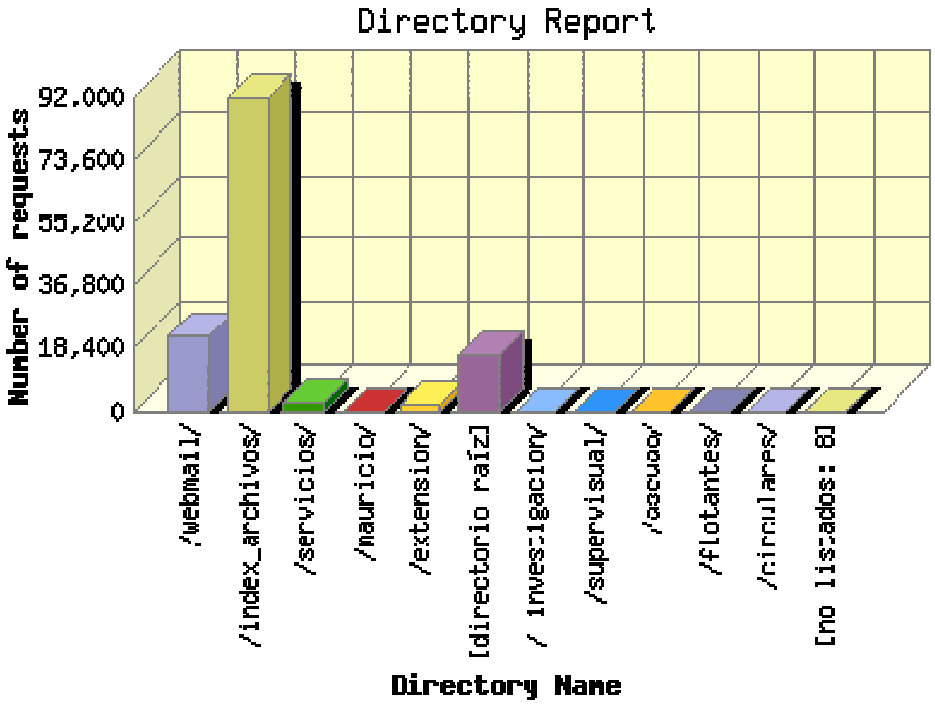
De acuerdo con la evolución en los contenidos de las aplicaciones y sistemas de información se sugiere proveer como mínimo una conexión a 100 Mbps en cada computadora, teniendo en cuenta el alto contenido de formatos de vídeo, imagen y sonido (multimedia).

Informe de directorios

El reporte de directorios analiza el acceso a los directorios del sitio analizado. Esta información puede ser usada para determinar las áreas que son más requeridas.

Figura 17
Informe de directorios

Figura 18
Informe de directorios



En este sitio existe un enlace al servicio de correo electrónico a través de una interface web, es por esto que la mayoría de las peticiones (45.66%) son dirigidas al directorio webmail. En este se encuentran almacenados los archivos que permiten el funcionamiento del webmail *squirrelmail*.

Este directorio almacena uno de los principales servicios que ofrece el sitio, debido a que el correo electrónico se ha convertido en el canal de comunicación oficial entre la comunidad universitaria. Diariamente es visitado por estudiantes, funcionarios y docentes.

Listado de los directorios que representan, por lo menos 0.01% del tráfico, ordenados por cantidad de tráfico.

Tabla 6
Informe de directorios

	Directory Name	Number of requests	Percentage of the bytes
1.	/webmail/	22,028	45.66%
2.	/index_archivos/	91,821	26.89%
3.	/servicios/	2,546	9.43%
4.	/mauricio/	39	8.34%
5.	/□upervisu/	2,285	5.33%
6.	(directorio raíz)	16,497	3.54%
7.	/ □upervisual□n/	246	0.44%
8.	/□upervisual/	29	0.16%
9.	/aseuao/	297	0.10%
10.	/flotantes/	4	0.07%
11.	/circulares/	5	0.03%
	(no listados: 8)	324	0.01%

Debido a su criticidad, se recomienda disponer de una solución alterna en caso de una falla prolongada para mantener publicado el acceso al servicio de correo electrónico y garantizar la disponibilidad del mismo.

Igualmente, se puede observar como la página de entrada al sitio web tiene un alto nivel de uso (26.89% de peticiones) y sus contenidos son visitados frecuentemente, pues los archivos de esta página se encuentran en el directorio */index_archivos*.

La información de uso común, como circulares y comunicados, es enviada a la comunidad universitaria a través de un mensaje de correo electrónico. El contenido de estos mensajes electrónicos contiene un enlace a los directorios de */circulares* o */flotantes*, los cuales almacenan archivos en formato .doc, pdf, pps u otros para ser descargados o visualizados y de esta forma no congestionar el servicio. Otros directorios también son requeridos con un número menor de peticiones como los de */extensión* e */investigación*. El primero almacena información web sobre la división de extensión de la universidad, la cual ofrece cursos de educación no formal y los cuales son publicitados a través del site. El segundo guarda la información de grupos de investigación que se desarrollan al interior de la universidad.

Informe de peticiones

Este reporte enlista los archivos en el sitio.

Figura 19
Informe de peticiones

Listado de los archivos que representan por lo menos 20 peticiones, ordenados por número de peticiones.

No. pet.: %bytes: última hora: archivo

----------: ---------: --------------: ---------

16.484: 3.43%: 29/Jul/2004 12:52 pm: /

13.433: 0.03%: 29/Jul/2004 12:52 pm: /index_archivos/spacer.gif

13.354: 0.37%: 29/Jul/2004 12:52 pm: /index_archivos/index_r1_c1.gif

13.346: 0.03%: 29/Jul/2004 12:52 pm: /index_archivos/fondo.gif

13.310: 0.26%: 29/Jul/2004 12:52 pm: /index_archivos/index_r3_c1.gif

13.243: 0.29%: 29/Jul/2004 12:52 pm: /index_archivos/index_r3_c2.gif

11.361: 0.69%: 29/Jul/2004 12:52 pm: /index_archivos/cuadrit.gif

7.005: 11.62%: 29/Jul/2004 11:19 am: /index_archivos/moviehome.swf

6.765: 13.60%: 29/Jul/2004 12:52 pm: /index_archivos/home1a.swf

5.435: 9.55%: 29/Jul/2004 12:50 pm: /webmail/src/right_main.php

924: 1.50%: 29/Jul/2004 12:39 pm: /webmail/src/right_main.php?PG_SHOWALL=0&sort=0&startMessage=1&mailbox=INBOX

435: 0.74%: 29/Jul/2004 11:57 am: /webmail/src/right_main.php?mailbox=INBOX&startMessage=1

341: 0.57%: 29/Jul/2004 12:46 pm: /webmail/src/right_main.php?mailbox=INBOX&sort=6&startMessage=

300: 0.51%: 29/Jul/2004 12:34 pm: /webmail/src/right_main.php?sort=6&startMessage=1&mailbox=INBOX

234: 0.26%: 29/Jul/2004 12:31 pm: /webmail/src/right_main.php?PG_SHOWALL=0&sort=0&startMessage=1&mailbox=BASURERO

152: 0.16%: 29/Jul/2004 12:28 pm: /webmail/src/right_main.php?PG_SHOWALL=0&sort=0&startMessage=1&mailbox=ENVIADOS

134: 0.30%: 29/Jul/2004 12:38 pm:
/webmail/src/right_main.php?use_mailbox_cache=1&startMessage=16&mailbox=INBOX
86: 0.16%: 29/Jul/2004 12:17 pm:
/webmail/src/right_main.php?use_mailbox_cache=0&startMessage=16&mailbox=INBOX
67: 0.05%: 29/Jul/2004 9:25 am:
/webmail/src/right_main.php?PG_SHOWALL=0&sort=0&startMessage=1&mailbox=BORRADOR
56: 0.31%: 29/Jul/2004 12:25 pm:
/webmail/src/right_main.php?PG_SHOWALL=1&use_mailbox_cache=1&startMessage=1&mailbox=INBOX
55: 0.15%: 29/Jul/2004 12:08 pm:
/webmail/src/right_main.php?mailbox=INBOX&sort=0&startMessage=
52: 0.11%: 29/Jul/2004 11:47 am:
/webmail/src/right_main.php?use_mailbox_cache=1&startMessage=31&mailbox=INBOX
47: 0.12%: 29/Jul/2004 12:36 pm:
/webmail/src/right_main.php?mailbox=INBOX&sort=2&startMessage=
46: 0.09%: 28/Jul/2004 8:14 pm:
/webmail/src/right_main.php?sort=2&startMessage=1&mailbox=INBOX
41: 0.22%: 28/Jul/2004 4:20 pm:
/webmail/src/right_main.php?sort=0&startMessage=1&mailbox=INBOX
32: 0.07%: 29/Jul/2004 12:17 pm:
/webmail/src/right_main.php?use_mailbox_cache=0&startMessage=1&mailbox=INBOX
31: 0.18%: 29/Jul/2004 10:51 am:
/webmail/src/right_main.php?PG_SHOWALL=1&use_mailbox_cache=0&startMessage=1&mailbox=INBOX

23: 0.05%: 29/Jul/2004 12:42 pm:
/webmail/src/right_main.php?use_mailbox_cache=1&startMessage=46&mailbox=INBOX
17: 0.03%: 28/Jul/2004 3:36 pm:
/webmail/src/right_main.php?mailbox=INBOX&sort=3&startMessage=
17: 0.04%: 29/Jul/2004 5:44 am:
/webmail/src/right_main.php?use_mailbox_cache=0&startMessage=31&mailbox=INBOX
17: 0.05%: 29/Jul/2004 10:54 am:
/webmail/src/right_main.php?use_mailbox_cache=1&startMessage=1&mailbox=INBOX
16: 0.03%: 29/Jul/2004 5:45 am:
/webmail/src/right_main.php?use_mailbox_cache=0&startMessage=46&mailbox=INBOX
14: 0.01%: 24/Jul/2004 10:09 am:
/webmail/src/right_main.php?mailbox=BORRADOR&sort=2&startMessage=1&note=Borrador+Guardado
13: 0.02%: 27/Jul/2004 7:55 pm:
/webmail/src/right_main.php?mailbox=INBOX&sort=4&startMessage=
10: 0.02%: 26/Jul/2004 6:42 pm:
/webmail/src/right_main.php?sort=4&startMessage=1&mailbox=INBOX
3.270: 4.91%: 29/Jul/2004 12:51 pm: /webmail/src/read_body.php
22: 0.03%: 29/Jul/2004 9:56 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=882&startMessage=1
16: 0.02%: 28/Jul/2004 1:07 pm:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=883&startMessage=1
15: 27/Jul/2004 6:19 pm:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=1&startMessage=1
14: 0.02%: 28/Jul/2004 11:00 pm:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=216&startMessage=1

13: 0.01%: 28/Jul/2004 7:37 pm:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=176&startMessage=1
12: 0.02%: 26/Jul/2004 9:19 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=173&startMessage=1
12: : 29/Jul/2004 10:44 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=481&startMessage=1
11: 0.02%: 25/Jul/2004 11:51 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=40367&startMessage=1
11: 0.01%: 28/Jul/2004 9:49 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=218&startMessage=1
10: 0.01%: 29/Jul/2004 11:55 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=152&startMessage=1
10: 0.01%: 29/Jul/2004 12:32 pm:
/webmail/src/read_body.php?mailbox=ENVIADOS&passed_id=6&startMessage=1
10: 0.02%: 28/Jul/2004 4:03 pm:
/webmail/src/read_body.php?passed_id=191&mailbox=INBOX&sort=6&startMessage=1&show_more=0&delete_id=190
10: 0.01%: 28/Jul/2004 4:46 pm:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=903&startMessage=1
10: 0.02%: 27/Jul/2004 11:38 am:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=6092&startMessage=1
10: 0.01%: 28/Jul/2004 12:47 pm:
/webmail/src/read_body.php?mailbox=INBOX&passed_id=2&startMessage=1
2.285: 0.57%: 29/Jul/2004 12:50 pm: /webmail/src/left_main.php
2.229: 0.68%: 29/Jul/2004 12:49 pm: /webmail/src/login.php
1.900: 0.08%: 29/Jul/2004 12:50 pm: /webmail/src/webmail.php
1.852: 7.55%: 29/Jul/2004 12:49 pm:
/servicios/imagenes/estudiantiles/taekwondo/fight.gif
1.475: 0.02%: 29/Jul/2004 12:49 pm: /webmail/images/sort_none.png
1.368: 3.41%: 29/Jul/2004 12:47 pm: /webmail/images/logo2004.bmp

1.229: 24.44%: 29/Jul/2004 12:51 pm: /webmail/src/download.php
17: 0.13%: 28/Jul/2004 4:47 pm: /webmail/src/download.php?startMessage=1&passed_id=903&mailbox=INBOX&ent_id=2
1.016: 1.60%: 29/Jul/2004 12:51 pm: /webmail/src/compose.php
261: 0.28%: 29/Jul/2004 12:51 pm: /webmail/src/compose.php?mailbox=INBOX
13: : 28/Jul/2004 11:48 am: /webmail/src/compose.php?passed_id=889&mailbox=INBOX&startMessage=1&passed_ent_id=0&smaction=forward
11: 0.01%: 23/Jul/2004 12:58 pm: /webmail/src/compose.php?passed_id=871&mailbox=INBOX&startMessage=1&passed_ent_id=0&smaction=forward
10: 0.01%: 28/Jul/2004 9:56 am: /webmail/src/compose.php?mailbox=ENVIADOS
379: 0.04%: 29/Jul/2004 12:33 pm: /webmail/src/signout.php
364: : 29/Jul/2004 12:33 pm: /webmail/images/up_pointer.png
362: 0.31%: 29/Jul/2004 12:41 pm: /extension/programacion_archivos/fondologo.gif
230: 0.04%: 29/Jul/2004 12:18 pm: /webmail/src/redirect.php
197: 0.07%: 29/Jul/2004 11:23 am: /webmail/src/image.php
176: : 29/Jul/2004 10:46 am: /webmail/themes/css/sans-10.css
111: 0.08%: 29/Jul/2004 12:10 pm: /webmail/src/view_text.php
92: : 29/Jul/2004 12:49 pm: /webmail/images/down_pointer.png
60: 0.17%: 29/Jul/2004 12:19 pm: /extension/abril/medicion e indicadores de gestion/medigest.htm
59: 0.32%: 29/Jul/2004 9:36 am: /extension/programacion.html
58: 0.14%: 29/Jul/2004 12:19 pm: /extension/abril/medicion e indicadores de gestion/medeindi.gif
57: 0.12%: 29/Jul/2004 12:19 pm: /extension/abril/medicion e indicadores de gestion/informes e inscripciones.gif
56: 0.06%: 28/Jul/2004 7:40 pm: /webmail/src/addressbook.php

54: 0.13%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/educont.gif
54: 0.20%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/f1.jpg
52: 0.08%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/estructura.gif
52: 0.01%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/maildepen.gif
51: 0.20%: 29/Jul/2004 11:21 am: /extension/abril/alta gerencia produc-
cion/altager.htm
50: 0.09%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/f2.jpg
50: 0.14%: 29/Jul/2004 11:21 am: /extension/abril/alta gerencia produc-
cion/altgerenpm.gif
48: 0.24%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/alianzas.gif
48: 0.11%: 29/Jul/2004 11:21 am: /extension/abril/alta gerencia produc-
cion/informes e inscripciones.gif
47: 0.04%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/cede.jpg
46: 0.06%: 28/Jul/2004 11:51 pm: /extension/programacion_archivos/arroba.gif
46: 0.01%: 28/Jul/2004 11:52 pm: /extension/programacion_archivos/academia.gif
46: 0.01%: 28/Jul/2004 11:52 pm: /extension/programacion_archivos/logozeiky.gif
40: 0.03%: 28/Jul/2004 1:02 pm: /servicios/imagenes/estudiantiles/karate.gif
38: : 29/Jul/2004 12:19 pm: /webmail/themes/css/sans-12.css
38: 0.05%: 28/Jul/2004 5:46 pm: /webmail/src/options.php
35: 0.01%: 29/Jul/2004 11:07 am: /webmail/src/move_messages.php
31: : 28/Jul/2004 11:35 am: /webmail/themes/css/serif-10.css
30: 0.07%: 29/Jul/2004 12:41 pm: /extension/marzo/crystal ball/crystal.htm
30: 0.09%: 29/Jul/2004 10:22 am: /extension/marzo/ahorro energia/ahorrener.html
29: 0.07%: 29/Jul/2004 10:23 am: /extension/marzo/ahorro energia/informes e ins-
cripciones.gif
29: 0.82%: 28/Jul/2004 5:26 pm: /servicios/pages/boletin/links_index/consultex.pps
29: 0.08%: 29/Jul/2004 10:23 am: /extension/marzo/ahorro energia/ahorener.gif
28: 0.14%: 29/Jul/2004 3:26 am: /servicios/tramites/hom_conv_espana.pdf
28: 0.09%: 29/Jul/2004 12:41 pm: /extension/marzo/crystal ball/modelsimul.gif

27: 0.06%: 29/Jul/2004 12:41 pm: /extension/marzo/crystal ball/informes e inscripciones.gif
26: 0.05%: 28/Jul/2004 6:35 pm: /extension/abril/gerencia capacitacion/informes e inscripciones.gif
26: 0.06%: 28/Jul/2004 6:35 pm: /extension/abril/gerencia capacitacion/gerencapa.gif
24: 0.09%: 28/Jul/2004 6:35 pm: /extension/abril/gerencia capacitacion/gerencapa.htm
24: 0.06%: 29/Jul/2004 11:45 am: /extension/febrero/calidad sector educativo/informes e inscripcionesexp.gif
24: 0.07%: 29/Jul/2004 11:44 am: /extension/febrero/calidad sector educativo/webcalidadsector.htm
24: 0.02%: 29/Jul/2004 11:45 am: /extension/febrero/calidad sector educativo/fondologo.gif
24: 0.07%: 29/Jul/2004 11:44 am: /extension/febrero/calidad sector educativo/gescalidasecedexp.gif
23: 0.07%: 29/Jul/2004 8:56 am: /extension/abril/ev. impacto ambiental/impactam.htm
23: 0.09%: 29/Jul/2004 10:26 am: /extension/marzo/gestion humana/gesthuman.htm
22: 0.06%: 29/Jul/2004 8:56 am: /extension/abril/ev. impacto ambiental/evaimpamb.gif
22: 0.05%: 29/Jul/2004 8:56 am: /extension/abril/ev. impacto ambiental/informes e inscripciones.gif
22: 0.02%: 29/Jul/2004 11:11 am: /webmail/src/view_header.php
21: 0.05%: 29/Jul/2004 10:26 am: /extension/marzo/gestion humana/informes e inscripciones.gif
21: 0.06%: 29/Jul/2004 10:26 am: /extension/marzo/gestion humana/altgerengh.gif
20: : 29/Jul/2004 12:50 pm: /icons/blank.gif
2158: 11.90%: 29/Jul/2004 12:51 pm: (no listados: 602 archivos).

CAPÍTULO 5
CONCLUSIONES Y RECOMENDACIONES

5.1 SOBRE EL DATA MINING

Consideramos que la capacidad para almacenar datos ha crecido en los últimos años a velocidades exponenciales. En contrapartida, la capacidad para procesarlos y utilizarlos no ha ido a la par. Por este motivo, el DM se presenta como una tecnología de apoyo para explorar, analizar, comprender y aplicar el conocimiento obtenido usando grandes volúmenes de datos. Queremos descubrir nuevos caminos que nos ayuden en la identificación de interesantes estructuras en los datos, que es una de las tareas fundamentales del DM.

Con los informes que arroje el servidor se puede obtener valiosa información sobre tendencias de funcionamiento y uso, además permite realizar un análisis estadístico de todas sus operaciones y accesos.

Un sistema al estilo DM nos permite analizar factores de influencia en determinados procesos, predecir o estimar variables o comportamientos futuros, segmentar o agrupar ítems similares, además de obtener secuencias de eventos que provocan comportamientos específicos.

Se puede concluir que el WUM es una herramienta muy poderosa, con la cual se pueden lograr mejoras significativas tanto en la parte del comercio electrónico, viéndolo desde el punto de vista de los negocios, como desde la búsqueda de patrones de comportamiento en los estudiantes de la universidad, si nos referimos al uso del Log de la UAO, para saber que hacen los usuarios de los equipos de la universidad en sus momentos de navegación.

Las herramientas comerciales de DM que existen actualmente en el mercado son variadas y excelentes. Las hay orientadas al estudio de la web o al análisis de documentos o de clientes de supermercado, mientras que otras son de uso más general. Su correcta elección depende de la necesidad de la empresa y de los objetivos a corto y largo plazo que pretenda alcanzar.

La decisión de utilizar una solución DM no es una tarea simple, es necesario consultar a expertos en el área con vista a seleccionar la más adecuada para el problema de la empresa y tener presente que el data minino (DM) es una tecnología compuesta por etapas que integra varias áreas y que no debe confundirse con un gran software.

5.2 SOBRE NUESTRO CASO

Como objetivo general de nuestro proyecto nos habiamos planteado desarrollar un algoritmo que reflejara el procedimiento a seguir usando el DM aplicado a la *web mining*, dentro de un ambiente real, utilizando los datos provenientes del tráfico del portal de la Universidad Autónoma de Occidente (UAO); para lograrlo se requirió construir un software que permitio la extracción y preparación de datos y posteriormente alimentaron una herramienta existente en el mercado denominada ANALOG, para proporcionar reportes del patrón de comportamiento de los individuos que acceden al sitio de la universidad. Esta herramienta nos permitio para el caso de la universidad determinar las peticiones exitosas, las peticiones no logradas, las redireccionadas, el promedio de peticiones exitosas por día y por pagina por día, los archivos y sistemas diferentes solicitados, la transferencia total de megabytes y su promedio por día, entre otras.

Además el resumen general contiene estadísticas globales sobre todos los datos analizados, siendo los más importantes el número de peticiones exitosas (incluye el número total de archivos descargados y gráficas); el número de requerimientos por páginas; el número de diferentes hosts que hacen peticiones sobre el sitio; y la cantidad de datos transferidos en bytes.

Una de las contribuciones de nuestro proyecto será aportar mayor conocimiento a la implementación e interpretación de resultados de web-mining no solo en instituciones educativas sino en cualquier tipo de organización.

El procedimiento que realizamos dentro nuestra investigación nos permitio cumplir con las metas de nuestro proyecto como era utilizar la información que emana periódicamente de la red de la universidad y que queda recolectada en el archivo de log y a partir de ella generar información grafica sobre estadisticas de comportamiento que pueden ser analizadas. Pudimos de esta manera en forma práctica verificar como el *web usage mining* permite el descubrimiento de patrones de acceso a partir de ficheros o archvivos de log, haciendo diferentes aproximaciones a la solución del problema.

Nos habiamos propuesto originalmente generar una herramienta que tenga la capacidad de tomar la información existente en una base de datos alimentada por medio de la web, la procesara posteriormente para la producción de los reportes requeridos, pero solo realizamos la herramienta para la extracción y preparación de los datos y decidimos usar para la presentación de los informes estadisticos una herramienta encontrada en Internet que luego de evaluarla consideramos que cumplia con las expectativas deseadas sobre los reportes y graficos que eran de interes para el departamento de redes de la universidad.

Para un efectivo desarrollo de los DM será necesario implementar en la universidad mecanismos que garanticen una alta disponibilidad del sitio durante los días de la semana de mayor tráfico y de este modo lograr que cualquier trabajo de administración o mantenimiento que requiera la suspensión del servicio se pueda realizar en los días de menos acceso.

Para esta investigación se planeo la creación de un prototipo de software que por medio de las técnicas de WUM, nos diera como resultado un reporte basado en el Log Web de la Universidad, pero al existir en el mercado software como el ANALOG y el WEBALIZER que son unas herramientas poderosas, se opto por estudiar el manejo de este programa ya existente y profundizar mas en el análisis de los resultados, claro está sin descuidar el estudio de los algoritmos implementados, su funcionamiento y su aplicación en estos. El prototipo desarrollado solo se limita a realizar la extracción y presentación de los datos.

Como una continuidad de nuestro proyecto, nos proponemos presentar a la Institución un plan para desarrollar, con la participación de estudiantes de pregrado de ingenieria de sistemas, una versión mejorada y ampliada del software que no solo permita la extracción y preparación de los datos sino que genere todas las estadisticas y graficas basadas en la información que la herramienta encontrada produce. Además consideramos que este proyecto puede servir de base para la construcción de un software de fabricación nacional que permita no solo que las instituciones educativas, sino en general cualquier organización realice análsis de datos del comportamiento de los visitantes al portal de la empresa.

La detección de sesiones de robots con comportamiento limitado, es decir, que no recorren enlaces externos al sitio web, está siendo estudiada. La identificación de sesiones maliciosas mediante técnicas borrosas se propone como proyecto futuro a realizar a partir de la caracterización de sesiones de robot.

.

CAPÍTULO 6
GLOSARIO

- **HTML (HyperText Mark-up Language)**

Este documento introduce el término HTML explicando como un conjunto de códigos estandarizados que se utilizan para preparar documentos de la *world wide web*. El módulo provee la definición de “página web”, introduce la definición de SGML (Standard Generalized Markup Language) e indica que los navegadores interpretan los códigos HTML para generar una página web legible. Finalmente, el módulo da ejemplos de códigos HTML básicos, con ilustraciones de código fuente y la página web que producen.

- **HTTP (HyperText Transfer Protocol)**

El concepto HTTP, que también introduce los protocolos HTML y links (o conexiones), describe el trabajo cliente/servidor y provee una lista de recursos accesibles vía HTTP. Ejemplos de links (o conexiones) hipertexto.

- **Navegador Netscape**

Este trabajo provee un sumario de las características del navegador Netscape. Los tópicos incluyen navegadores gráficos en general así como aplicaciones auxiliares de Netscape, bookmarks (o marcas de interés) y customización.

- **URL (Uniform Resource Locator)**

Este módulo explica cómo cada recurso www tiene un único URL. Se describe su estructura jerárquica, dando ejemplos y explicando cada componente.

- **Página web**

Este módulo describe lo que son páginas web, explicando que son unidades básicas de información, normalmente llamada “documento”, disponible desde la *world*

wide web. Además, el módulo introduce el código HTML utilizado para crear páginas web. Finalmente, se incluye una discusión de Hypertext Transfer Protocol. También incluye capturas de pantallas de páginas web con conexiones a audio y vídeo**.**

- **Los tags <META> </META>**

Contienen información referida al documento HTML pero no se presentan en la pantalla del navegador. De acuerdo con el nombre que tenga asignado, el contenido de dicha información servirá a distintos propósitos. Los que aquí nos interesan son los que pueden ayudarnos a mejorar nuestra posición en los resultados de búsqueda de los “search engines”.

Estos son:

<META NAME=“keywords”>

Este nos permite incluir en su CONTENT las palabras clave (keywords) mediante las cuales deseamos que nuestro sitio sea hallado. Cuando el motor de búsqueda lea nuestras páginas, pondrá especial atención a este tag e indexará también su contenido. Así, y como ejemplo, si tenemos un sitio dedicado al marketing online, puede incluir keywords tales como: marketing, marketing online, marketing internet, cyber marketing, negocios online, etc. Todo aquello que puede estar buscando un posible visitante interesado en nuestro producto o servicio. No se trata, sin embargo, de hacer una lista enorme: sólo debemos incluir las palabras relativas necesarias. La sintaxis es:

<META NAME=“keywords” CONTENT=“marketing,negocios,estrategias, ...”>
<META NAME=“description">
Con este podemos controlar la forma en que aparecerá nuestro enlace en los resultados de búsqueda, esto es, la descripción del sitio que aparece a continuación del título de la página.

Siguiendo el ejemplo anterior, la sintaxis sería: <META NAME="description" CONTENT= "¿No sabe cómo empezar con su negocio online? Aquí le presentamos estrategias online que le serán muy útiles">

La utilización de los tags META es recomendable y aunque de por sí no aseguran posicionamiento alguno en los resultados de búsqueda, un trabajo que combine los META con un título adecuado y un texto descriptivo, nos permitirán mejorar el que obtendríamos con el mismo sitio si no los utilizáramos. Cabe tener en cuenta que algunos buscadores los ignoran y otros lo leen solamente como parte del contenido de una página, sin darle una relevancia particular.

En lo que se refiere a la inclusión de los tags META dentro del código HTML de una página, deben estar ubicados dentro del espacio comprendido entre los tags <HEAD> y </HEAD>.

<IMG ALT=" ">

El atributo ALT, utilizado dentro del tag IMG, permite que los navegadores configurados para no cargar las imágenes, puedan exhibir un texto en reemplazo de estas. También es accesible por los navegadores más antiguos que trabajan sólo con texto. En lo que respecta a algunos motores de búsqueda, este texto alternativo forma parte del contenido de una página y por lo tanto es indexado. Conviene, entonces, incluir aquí también las palabras clave seleccionadas.

BIBLIOGRAFÍA

- DISCOVERING DATA MINING FROM CONCEPT TO IMPLEMENTATION / Cabena, Hadjinian, Stadler, Verhees, Zanasi.

- Mining Navigation Patterns with Hypertext Probabilistic Grammars/ Jose Borges y Mark Levene / Departamento de Ciencias Computacionales / Universidad de London/199.

- MINEit Software Ltd, Facultad de Informatica/ Universidad de Ulster Shore Road, Newtownabbey, Co. / A.G. Buchner, S.S. Anand, M.D. Mulvenna y J.G. Hughes/1999.

- webSIFT: The web Site Information Filter System / Robert Cooley / Departamento de Ciencias Computacionales / Universidad de Minnesota / 1999.

- Learning to Extract Symbolic Knowledge from World wide web / Mark Craven , Dan DiPasquo / Escuela de Ciencias Computacionales Universidad Carnegie Mellon / Pittsburgh / 1999.

- Discovering DATA MINING: From Concept to Implementation, 1/e / Peter Cabena, Almaden Research Ctr., San Jose, California.

- INSTITUTO COLOMBIANO DE NORMAS TECNICAS. Normas Colombianas para la presentación de trabajos de investigación. Segunda actualización. Santafe de Bogotá D.C. ICONTEC, 2000.

- http://www.geocities.com/CapeCanaveral/Launchpad/7651/dminits.htm / Agosto / 2000

- http://megamarketing.com.mx/datam.htm / Agosto / 2000

- www.DATA MINING.com / Agosto / 2000

- www.vermontinc.com / Agosto /2000

- www.themeasurementgroup.com/Septiembre/2000

- www.http://www.angoss.com/angoss.html/Septiembre/2000

- http://www.imparcial.com.mx/Notas/Semanal/informatica/n14.htm / Agosto30 / 2000

- www.webminer.com/ Agosto / 2000

- http://usability.gov/serverlog/index.html Excelente artículo sobre circunstancias a la hora de extraer información de un log de visitas.

- http://www.analog.cx/docs/webworks.html. Errores comunes a la hora de leer un log de visitas.

- http://www.infovis.net/Revista/num_65.htm. Artículo dentro de infovis.net sobre visualización de logs. Directorio de google sobre analizadores de tráfico.

- http://directory.google.com/Top/Computers/Software/Internet/Site_Management/Log_Analysis/ Archivos de ACM.org sobre logs

- http://usability.gov/serverlog/index.html Excelente Artículo sobre circunstancias a la hora de extraer información de un log de visitas.

- http://www.microsoft.com/usability/webconf/fuller/fuller.htm Juan C. Dursteler recomienda un par de artículos:

- http://www.analog.cx/docs/webworks.html. Errores comunes a la hora de leer un log de visitas. Un tono algo negativo.

- http://www.infovis.net/Revista/num_65.htm. Artículo dentro de infovis.net sobre visualización de logs. Directorio de google sobre analizadores de tráfico.

- http://directory.google.com/Top/Computers/Software/Internet/Site_Management/Log_Analysis/ Archivos de ACM.org sobre logs.

- http://www.acm.org/archives Archivos de Cadius sobre preferencias a la hora de usar herramientas de log.

ANEXO 1
PROGRAMA IMPLEMENTADO

ANALOGS.C

```
/*************************************************************************
                                    *****
*
* INTERFAZ E IMPLEMENTACION DEL MODULO: AnaLogs.c
*
* VERSION: 1.0
*
* LUGAR Y FECHA DE CREACION: Santiago de Cali, Octubre de 2003 V. 1.0
*
* Descripci¢n: Programa que analisa los logs generados en el archivo:
* c:\AnaLogs\logs.html
*
* Requerimientos: Este programa lee un archivo logs.html desde el
* directorio: c:\AnaLogs. Luego se eliminan las etiquetas
* de HTML y con los registros que quedan se crea un nuevo
* archivo llamado Logs.txt, donde cada fila es un LOG.
* Con este archivo el AnaLogs hace su analisis.
*
*************************************************************************/

/*_____________________________________________________________________*

       Archivos de interfaz de librerias estandar y otros modulos
*_____________________________________________________________________*/
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
/*_____________________________________________________________________*

            Declaracion de Constantes y Macros, publicas
*_____________________________________________________________________*/
#define Esc 27
#define Tab 9
/*_____________________________________________________________________*

       Declaracion de Estructuras y Redefiniciones de Tipo, publicas
*_____________________________________________________________________*/
struct Log
{
      char Mes[4];
```

```
	char Dia[3];
	char Hora[3];
	char Min[3];
	char Seg[3];
	char Servidor[23];
	char ID[36];
	char Accion[40];
	char User[10];
};

/*_____________________________________________________________________*

		Definicion de Variables Publicas
*_____________________________________________________________________*/
typedef struct Log elemento;
/*_____________________________________________________________________*

	Archivos de interfaz de librerias estandar y otros modulos
*_____________________________________________________________________*/
#include "c:\analogs\lista_ap.h"
lista fecha;
/*_____________________________________________________________________*

		Prototipos de Funciones Publicas
*_____________________________________________________________________*/
void presentacion(void);
void cargar_archivos(void);
void analisis_logs(void);
void consola_administracion(void);
void salir(void);
/*_____________________________________________________________________*

		Prototipos de Funciones Privadas
*_____________________________________________________________________*/

/*_____________________________________________________________________*

		Implementacion de Funciones Publicas
*_____________________________________________________________________*/
void main(void)
{
	presentacion();
	printf("Espere, procesando los Logs.");
	cargar_archivos();
	fecha = inic_lista();
	analisis_logs();
}

void presentacion(void)
{
Definicion de la pantalla de presentacion.

	clrscr();
	textbackground(LIGHTBLUE);
```

```
        window( 10, 5, 70, 20);
        clrscr();
        textcolor(YELLOW);
        gotoxy(14,3);
        printf(" AnaLogs 1.0");
        gotoxy(14,5);
        printf(" HECTOR FABIO PANESSO");
        gotoxy(14,6);
        printf(" CARLOS ALBERTO MURILLO");
        gotoxy(14,9);
        printf(" ESTE PROGRAMA ANALIZA LOS LOGS");
        gotoxy(14,10);
        printf("GENERADOS EN EL ARCHIVO logs.html");
        gotoxy(14,12);
        printf(" BIENVENIDO");
        gotoxy(14,14);
        printf(" OCTUBRE DE 2003");
        gotoxy(22,16);
        printf(" DIGITE UNA TECLA PARA CONTINUAR...");
        getch();
        textbackground(BLACK);
        textcolor(WHITE);
        window(1,1,80,25);
        clrscr();
}

void cargar_archivos(void)
{

        FILE *archivoHTML;
        FILE *archivoTXT;
        FILE *archivoTXT2;
        FILE *archivoTXT3;
        FILE *archivoTXT4;
        int i, opcion, tecla, copiar, num1, num2, num3;
        char caracter, charAnt, charPen, charAnte, charAnteAnte;
        char cadena1[2], cadena2[2], cadena3[2], cadena4[2];

Confirma que el archivo d:\\tc\\tmp\\logs~1.html exista. Si no es asi,
muestra mensaje de error.

if ( (archivoHTML=fopen("d:\\tc\\tmp\\logs~1.html","rt")) == NULL )
        {
                printf("\n\n\nERROR!!! EL ARCHIVO NO EXISTE\n");
                printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
                getch();
                fclose(archivoHTML);
                salir();
                return;
        }
Confirma que el archivo d:\\tc\\tmp\\logs2.txt exista. Si no es asi, mues-
tra mensaje de error.

        if ( (archivoTXT2=fopen("d:\\tc\\tmp\\logs2.txt","wt")) == NULL )
```

```
	{
		printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER CREADO\n");
		printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
		getch();
		fclose(archivoTXT2);
		salir();
		return;
	}

	fseek(archivoHTML,0L,SEEK_END);
```

Confirma que el archivo html tenga datos en su interior.

```
	if ( ftell(archivoHTML)==0 )
	{
		printf("\n\n\nERROR! EL ARCHIVO ESTA VACIO\n");
		printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
		getch();
		fclose(archivoHTML);
		salir();
	}

	rewind(archivoHTML);
	copiar=0;
```

Recorre el archivo html de logs hasta que encuentre el fin de archivo.

```
	while ( !feof(archivoHTML) )
	{
		caracter=fgetc(archivoHTML);
		Obtiene el caracter y verifica que sea el carácter '<'

		if(caracter=='<')
		{
			caracter=fgetc(archivoHTML);
			Si es '<', obtiene el siguiente caracter y verifica que
			sea el carácter 's'.
			if(caracter=='s')
				caracter=fgetc(archivoHTML);
				Si es 's', obtiene el siguiente caracter y verifi-
				ca que sea el carácter 'p'.
					if(caracter=='p')
					{
						copiar=1;
						Si el caracter es 'p' escribe un enter
						en el archivoTXT2.
						fputc('\n', archivoTXT2);
					}
					else
						copiar=0;
			Si el caracter es diferente de '>' sigue tomando el si-
			guiente carácter.
			while(caracter!='>')
				caracter=fgetc(archivoHTML);
```

```
	}
	else
	{
		Si el caracter es diferente de '<' copia el carácter
		obtenido al archivoTXT2.
		if(copiar==1)
			fputc(caracter, archivoTXT2);
	}
}
Cierra los archivos HTML y archivoTXT2.
fclose(archivoHTML);
fclose(archivoTXT2);

if ( (archivoTXT=fopen("d:\\tc\\tmp\\logs3.txt","wt")) == NULL )
{
	printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER CREADO\n");
	printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
	getch();
	fclose(archivoTXT);
	salir();
	return;
}

if ( (archivoTXT2=fopen("d:\\tc\\tmp\\logs2.txt","rt")) == NULL )
{
	printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIERTO\n");
	printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
	getch();
	fclose(archivoTXT2);
	salir();
	return;
}

fseek(archivoTXT2,0L,SEEK_END);
if ( ftell(archivoTXT2)==0 )
{
	printf("\n\n\nERROR! EL ARCHIVO ESTA VACIO\n");
	printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
	getch();
	fclose(archivoTXT2);
	salir();
}
rewind(archivoTXT2);

while ( !feof(archivoTXT2) )
{
	caracter=fgetc(archivoTXT2);
	if(caracter!='\n')
		fputc(caracter, archivoTXT);
}
fclose(archivoTXT2);
fclose(archivoTXT);

if ( (archivoTXT4=fopen("d:\\tc\\tmp\\logs4.txt","wt")) == NULL )
```

```
	{
		printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIERTO\n");
		printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
		getch();
		fclose(archivoTXT4);
		salir();
		return;
	}

	if ( (archivoTXT3=fopen("d:\\tc\\tmp\\logs3.txt","rt")) == NULL )
	{
		printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIERTO\n");
		printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
		getch();
		fclose(archivoTXT3);
		salir();
		return;
	}

	fseek(archivoTXT3,0L,SEEK_END);
	if ( ftell(archivoTXT3)==0 )
	{
		printf("\n\n\nERROR! EL ARCHIVO ESTA VACIO\n");
		printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
		getch();
		fclose(archivoTXT3);
		salir();
	}
	rewind(archivoTXT3);

	while ( !feof(archivoTXT3) )
	{
	Mientras el existan datos en el archivoTXT3 toma cada uno de los
	caracteres.
		caracter=fgetc(archivoTXT3);
		charAnteAnte=charAnte;
		charAnte=charPen;
		charPen=charAnt;
		charAnt=caracter;

	Si las variables que contienen los caracteres cumplen las condicio-
	nes de ser igual o desigual a '.' o al codigo ASCII de 0-9 o :;
	if( (charAnteAnte=='.') && (charPen>=48 && charPen<=59) && (charAn-
te>=48 && charAnte<59) && (charAnt!='.') && !(charAnt>=48 && charAnt<=59)
)
		{
			Si el caracter anterior es igual a ']' escribe el ca-
			rácter en archivoTXT4 e imprime un enter. Sino imprime
			un enter y luego escribe el carácter en el archivoTXT4.
			if(charAnt==']')
			{
				fputc(caracter, archivoTXT4);
				fputc('\n', archivoTXT4);
			}
```

```
                  else
                  {
                        fputc('\n', archivoTXT4);
                        fputc(caracter, archivoTXT4);
                  }
            }
            else
            {

                  if( (charAnte=='.') && (charPen>=48 && charPen<=59) &&
(charAnt!='.') && !(charAnt>=48 && charAnt<=59) )
                  {
                        Si el caracter anterior es igual a ']' escribe el
                        carácter en archivoTXT4 e imprime un enter. Sino
                        imprime un enter y luego escribe el carácter en el
                        archivoTXT4.
                        if(charAnt==']')
                        {
                              fputc(caracter, archivoTXT4);
                              fputc('\n', archivoTXT4);
                        }
                        else
                        {
                              fputc('\n', archivoTXT4);
                              fputc(caracter, archivoTXT4);
                        }
                  }
                  else
                  Si no cumple con la condicion escribe el caracte en ar-
                  chivoTXT4
                        fputc(caracter, archivoTXT4);
            }
      }
      Cierra los archivos archivoTXT3 y archivoTXT4
      fclose(archivoTXT3);
      fclose(archivoTXT4);

      if ( (archivoTXT=fopen("d:\\tc\\tmp\\logs.txt","wt")) == NULL )
      {
            printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIERTO\n");
            printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
            getch();
            fclose(archivoTXT);
            salir();
            return;
      }

      if ( (archivoTXT4=fopen("d:\\tc\\tmp\\logs4.txt","rt")) == NULL )
      {
            printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIERTO\n");
            printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
            getch();
            fclose(archivoTXT4);
            salir();
```

```
            return;
      }

      fseek(archivoTXT4,0L,SEEK_END);
      if ( ftell(archivoTXT3)==0 )
      {
            printf("\n\n\nERROR! EL ARCHIVO ESTA VACIO\n");
            printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
            getch();
            fclose(archivoTXT4);
            salir();
      }
      rewind(archivoTXT4);

      while ( !feof(archivoTXT4) )
      {
            Mientras no se termine el archivoTXT4 se toma cada character
            y se compara si el character obtenido esta entre los codigos
            ASCII de 65 a 90.
            caracter=fgetc(archivoTXT4);
            if(caracter>=65 && caracter<=90)
            {
                  Si es asi, mientras no sea fin de archivoTXT4 y el ca-
                  rácter obtenido no sea un enter, se copia el carácter a
                  archivoTXT y se obtiene el siguiente carácter de archi-
                  voTXT4.
                  while ( caracter!='\n' && !feof(archivoTXT4) )
                  {
                        fputc(caracter, archivoTXT);
                        caracter=fgetc(archivoTXT4);
                  }
                  fputc(caracter, archivoTXT);
            }
            else
            {
            Si el caracter no esta entre los ASCII 65 y 90, y mientras no
            sea enter o fin de archivo se obtiene el siguiente carácter
            de archivoTXT4.
                  while ( caracter!='\n'&& !feof(archivoTXT4) )
                        caracter=fgetc(archivoTXT4);
            }

      }
      clrscr();

}

void analisis_logs(void)
{

      struct Log Log;
      struct Log2 Log2;
      FILE *archivoTXT;
      int num, j, pos=0, FechaValida=1, ServidorValido=1, UserValido;
```

```
char caracter;

if ( (archivoTXT=fopen("d:\\tc\\tmp\\logs.txt","rt")) == NULL )
{
      printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIERTO\n");
      printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
      getch();
      fclose(archivoTXT);
      salir();
      return;
}

fseek(archivoTXT,0L,SEEK_END);
if ( ftell(archivoTXT)==0 )
{
      printf("\n\n\nERROR! EL ARCHIVO ESTA VACIO\n");
      printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");
      getch();
      fclose(archivoTXT);
      salir();
}

rewind(archivoTXT);

while ( !feof(archivoTXT) )
{
      Mientras no se termine el archivoTXT se obtiene el carácter y
      se compara si es un enter.
      caracter=fgetc(archivoTXT);
      if(caracter=='\n')
      {
            caracter=fgetc(archivoTXT);
            Si es un enter '\n', se obtiene el siguiente carácter y
            se compara a ver si es igual al codigo ASCII entre 65 y
            90
            if(caracter>=65 && caracter<=90)
            {
                  j=0;
                  Si es asi se llena la estructura Log que tiene
                  Mes, Dia, Hora, Min, Seg, Servidor, ID, Accion,
                  User.

                  while(j<3)
                  {
                        Log.Mes[j]=caracter;
                        caracter=fgetc(archivoTXT);
                        j++;
                  }
                  Log.Mes[3]='\0';
                  j=0;
                  if(caracter>=65 && caracter<=90)
                  {
                        FechaValida=0;
                        pos--;
```

```
                        }
                        while(j<2)
                        {
                              Log.Dia[j]=caracter;
                              caracter=fgetc(archivoTXT);
                              j++;
                        }
                        Log.Dia[2]='\0';
                        caracter=fgetc(archivoTXT);
                        j=0;
                        while(j<2)
                        {
                              Log.Hora[j]=caracter;
                              caracter=fgetc(archivoTXT);
                              j++;
                        }
                        Log.Hora[2]='\0';
                        caracter=fgetc(archivoTXT);
                        j=0;
                        while(j<2)
                        {
                              Log.Min[j]=caracter;
                              caracter=fgetc(archivoTXT);
                              j++;
                        }
                        Log.Min[2]='\0';
                        caracter=fgetc(archivoTXT);
                        j=0;
                        while(j<2)
                        {
                              Log.Seg[j]=caracter;
                              caracter=fgetc(archivoTXT);
                              j++;
                        }
                  }
                  Log.Seg[2]='\0';
                  j=0;
                  while(j<17)
                  {
                        caracter=fgetc(archivoTXT);
                        if((caracter>=97 && caracter<=122) || (carac-
ter>=65 && caracter<=90) || caracter=='.' || caracter==' ')
                              Log.Servidor[j]=caracter;
                        else
                        {
                              ServidorValido=0;
                              pos--;
                              break;
                        }
                        j++;
                  }
                  Log.Servidor[17]='\0';
                  caracter=fgetc(archivoTXT);
                  while(caracter!=':')
```

```
      caracter=fgetc(archivoTXT);
j=0;
while(j<5)
{
      caracter=fgetc(archivoTXT);
      j++;
}
j=0;
while(j<6)
{
      caracter=fgetc(archivoTXT);
      Log.ID[j]=caracter;
      j++;
}
Log.ID[6]='\0';
caracter=fgetc(archivoTXT);
while(caracter!=']')
      caracter=fgetc(archivoTXT);
caracter=fgetc(archivoTXT);
caracter=fgetc(archivoTXT);
if(caracter=='i')
      strcpy(Log.Accion, "imap serviceinit");
if(caracter=='A')
      strcpy(Log.Accion, "Authenticateduser");
if(caracter=='C')
      strcpy(Log.Accion, "Connectionreset by peer");
j=0;
if(caracter=='L')
{
      j=0;
      while(j<3)
      {
            caracter=fgetc(archivoTXT);
            j++;
      }
      if(caracter=='o')
      {
            strcpy(Log.Accion, "Logoutuser");
            j=-1;
      }
      else
      {
            strcpy(Log.Accion, "Loginuser");
            j=0;
      }
      while(j<6)
      {
            caracter=fgetc(archivoTXT);
            j++;
      }
      j=0;
      while(caracter!=' ')
      {
            caracter=fgetc(archivoTXT);
```

```
                              if((caracter>=97 && caracter<=122) ||
(caracter>=65 && caracter<=90) || caracter==' ')
                                    Log.User[j]=caracter;
                              else
                              {
                                    UserValido=0;
                                    break;
                              }
                              j++;
                        }
                        j--;
                  }
                  if(!UserValido)
                        j=0;
                  Log.User[j]='\0';
                  if((pos<400) && FechaValida && ServidorValido)
                        fecha = anx_lista(fecha, Log);
                  if(pos==117)
                        pos=117;
                  pos++;
                  FechaValida=1;
                  UserValido=1;
                  ServidorValido=1;
            }
      }
```

Cierra el archivoTXT.
Hace llamado al metdo consola_administracion.

```
      fclose(archivoTXT);
      consola_administracion();
}

void consola_administracion(void)
{
      char op='0', Login[20], Mes[10], Dia[8], Hora[8], Minutos[8], Se-
gundos[8];
      int n, i, Op;
      struct Log Log;

      clrscr();
```

Carga la pantalla de presentacion con las cinco opciones que existe en la consola de administracion de logs.

```
      while(op!='5')
      {
            clrscr();
            printf("\n\nCONSOLA DE ADMINISTRACION DE LOGS(Lista con las
fechas de los Logs)\n\n");
            printf("\n1. Numero de Logs.");
            printf("\n2. Listar los Logs.");
            printf("\n3. Mostrar si un usuario se ha logueado.");
            printf("\n4. Mostrar un Log en una fecha y hora especifi-
ca.");
            printf("\n5. Salir.");
```

```
            printf("\nDigite su opcion: ");
            op=getch();
            clrscr();
            n=long_lista(fecha);
            switch(op)
            {
                  case '1': {
```

Si selecciona la opcion uno se imprime la variable n que tiene el numero de logs

```
                              printf("El numero de Logs es = %d.",n);
                              printf("\n\n DIGITE UNA TECLA PARA TERMI-
NAR...");
                              getch();
                              }
                              break;
                  case '2': {
```

Si selecciona la opcion dos se llama al metodo info_lista que trae información de los logs y permite hacer la impresión del reporte. Mostrando el numero de log, el dia, mes, hora, seg, servidor, ID, accion y usuario que lo realizo.

```
                              i = 1;
                              while(i<n)
                              {
                                    Log=info_lista(fecha,i);

      if(((strcmp(Log.Accion,"Loginuser")==0)||(strcmp(Log.Accion,"Logout
user")==0))&&(strcmp(Log.User,"")!=0))
                                          printf("* Log en la Posicion
%d:\nFecha = %s %s %s:%s:%s.\nServidor = %s.\nID = %s.\nAccion =
%s.\nUser =
%s.\n",i,Log.Mes,Log.Dia,Log.Hora,Log.Min,Log.Seg,Log.Servidor,Log.ID,Log
.Accion,Log.User);
                                    else
                                          printf("* Log en la Posicion
%d:\nFecha = %s %s %s:%s:%s.\nServidor = %s.\nID = %s.\nAccion =
%s.\n",i,Log.Mes,Log.Dia,Log.Hora,Log.Min,Log.Seg,Log.Servidor,Log.ID,Log
.Accion);
                                    i++;
                              }
                              printf("\n\n DIGITE UNA TECLA PARA TERMI-
NAR...");
                              getch();
                              }
                              break;
                  case '3': {
```

Si selecciona la opcion tres, se solicita el login del usuario el cual se desea saber si se ha logueado. Despues de esto se llama al metodo info_lista que trae información de los logs y se compara los datos que recibe con el login del usuario, si existe uno registro se imprime.

```
                              clrscr();
                              printf("Escriba el Login del usuario: ");
```

```
                                    scanf("%s",Login);
                                    i = 1;
                                    while(i<n)
                                    {
                                          Log=info_lista(fecha,i);

      if((strcmp(Log.Accion,"Loginuser")==0)&&(strcmp(Log.User,Login)==0)
)
                                                printf("* Log en la Posicion
%d:\nFecha = %s %s %s:%s:%s.\nServidor = %s.\nID = %s.\nAccion =
%s.\nUser =
%s.\n",i,Log.Mes,Log.Dia,Log.Hora,Log.Min,Log.Seg,Log.Servidor,Log.ID,Log
.Accion,Log.User);
                                          i++;
                                    }
                                    printf("\n\n DIGITE UNA TECLA PARA TERMI-
NAR...");
                                    getch();
                                    }
                                    break;
                        case '4': {
Si selecciona la opcion cuatro, se solicita el mes, dia , hora, minuto y
segundo del registro que desea ser buscado. Despues de esto se llama al
metodo info_lista que trae información de los logs y se compara los datos
que se digitaron con los que estan registrado en los archivos, si existe
uno registro que coincida se imprime.

                                    clrscr();
                                    printf("Escriba la fecha y hora del
Log:\n");
                                    printf("Escriba el numero del mes (1-12):
");
                                    scanf("%d",&Op);
                                    if(Op==1)
                                          strcpy(Mes,"Jan");
                                    if(Op==2)
                                          strcpy(Mes,"Feb");
                                    if(Op==3)
                                          strcpy(Mes,"Mar");
                                    if(Op==4)
                                          strcpy(Mes,"Apr");
                                    if(Op==5)
                                          strcpy(Mes,"May");
                                    if(Op==6)
                                          strcpy(Mes,"Jun");
                                    if(Op==7)
                                          strcpy(Mes,"Jul");
                                    if(Op==8)
                                          strcpy(Mes,"Aug");
                                    if(Op==9)
                                          strcpy(Mes,"Sep");
                                    if(Op==10)
                                          strcpy(Mes,"Oct");
                                    if(Op==11)
```

```
                                    strcpy(Mes,"Nov");
                                if(Op==12)
                                    strcpy(Mes,"Dec");
                                printf("Escriba el dia (01-31): ");
                                scanf("%s",&Dia);
                                printf("Escriba la hora (00-23): ");
                                scanf("%s",&Hora);
                                printf("Escriba los minutos (00-59): ");
                                scanf("%s",&Minutos);
                                printf("Escriba los segundos (00-59): ");
                                scanf("%s",&Segundos);
                                i = 1;
                                while(i<n)
                                {
                                    Log=info_lista(fecha,i);

      if((strcmp(Log.Mes,Mes)==0)&&(strcmp(Log.Dia,Dia)==0)&&(strcmp(Log.
Ho-
ra,Hora)==0)&&(strcmp(Log.Min,Minutos)==0)&&(strcmp(Log.Seg,Segundos)==0)
)
                                        if(strcmp(Log.User,"")!=0)
                                            printf("* Log en la Posi-
cion %d:\nFecha = %s %s %s:%s:%s.\nServidor = %s.\nID = %s.\nAccion =
%s.\nUser =
%s.\n",i,Log.Mes,Log.Dia,Log.Hora,Log.Min,Log.Seg,Log.Servidor,Log.ID,Log
.Accion,Log.User);
                                        else
                                            printf("* Log en la Posi-
cion %d:\nFecha = %s %s %s:%s:%s.\nServidor = %s.\nID = %s.\nAccion =
%s.\nUser = Desconoci-
do.\n",i,Log.Mes,Log.Dia,Log.Hora,Log.Min,Log.Seg,Log.Servidor,Log.ID,Log
.Accion);
                                    i++;
                                }
                                printf("\n\n DIGITE UNA TECLA PARA TERMI-
NAR...");
                                getch();
                                }
                                break;
                    case '5': {
Si selecciona la opcion cinco, se llama al metodo salir.
                                salir();
                                }
                                break;
                    default: break;
            }
      }
}

void salir()
{

      clrscr();
      exit(0);
```

```
}
```

LISTA_AP.C

```
/*********************************************************************
*
* IMPLEMENTACION MODULO: lista_ap.c
*
* LUGAR Y FECHA DE CREACION: Santiago de Cali, Diciembre de 2002.
*
* Descripcion: TAD Lista: Implementado Sobre Apuntadores.
*
* Requerimientos: Se requiere del Algoritmo de Control, un tipo de datos
* llamado elemento, para la parte info del nodo.
*
*********************************************************************/
/*____________________________________________________________________*

       Archivos de interfaz de librerias estandar y otros modulos
*____________________________________________________________________*/
#include <dos.h>
/*____________________________________________________________________*

             Implementacion de Funciones Publicas
*____________________________________________________________________*/
lista inic_lista(void)
/* pre: true
 post: lst=< > */
{
      return(NULL);
}


lista ins_lista(lista lst, int pos, elemento elem)
/* pre: lst=<a1,a2,...,a(pos-1),a(pos),a(pos+1),...,an>; n>0; 1<=pos<=n;
elem
 post: lst=<a1,a2,...,a(pos-1),elem,a(pos),a(pos+1),...,a(n+1)> */
{
      lista nuevo, aux;
      int longitud, cont;
      longitud=long_lista(lst);
      nuevo=(lista)malloc(sizeof(struct nodo));
      if ( nuevo==NULL )
      {
            clrscr();
            printf("ERROR NO HAY MEMORIA DISPONIBLE.");
            sound(300);
```

```
            sleep(1);
            nosound();
            sleep(3);
            clrscr();
            exit(0);
      }
      nuevo->info=elem;
      if ( pos==1 )
      {
            nuevo->sgte=lst;
            lst=nuevo;
      }
      else
      {
            aux=lst;
            cont=2;
            while ( cont!=pos )
            {
                  aux=aux->sgte;
                  cont++;
            }
            nuevo->sgte=aux->sgte;
            aux->sgte=nuevo;
      }
      if ( ( pos>longitud ) && ( pos!=1 ) )
      {
            clrscr();
            puts("ERROR POSICION INCORRECTA.");
            sound(300);
            sleep(1);
            nosound();
            sleep(2);
            clrscr();
      }
      return ( lst );
}

Anexar elemento a la lista
lista anx_lista(lista lst, elemento elem)
/* pre: lst=<a1,a2,...,an>; n>=0; elem
 post: lst=<a1,a2,...,an,elem> */
{
      lista nuevo, aux;
      En la variable nuevo se intenta reservar un espacio de memoria
      nuevo=(lista)malloc(sizeof(struct nodo));
      Si no se puede reservar se genera un error de memoria.
      if ( nuevo==NULL )
      {
            clrscr();
            printf("ERROR NO HAY MEMORIA DISPONIBLE.");
            sound(300);
            sleep(1);
            nosound();
            sleep(3);
```

```
            clrscr();
            exit(0);
      }
      De lo contrario se asigna al espacio en memoria el elmento y a la
      posición siguiente se inicializa en NULL
      nuevo->info=elem;
      nuevo->sgte=NULL;
      if ( lst==NULL )
      {
            lst=nuevo;
      }
      else
      {
            aux=lst;
            while ( aux->sgte!=NULL )
            {
                  aux=aux->sgte;
            }
            aux->sgte=nuevo;
      }
      return ( lst );
}
```

Eliminar elemento de la lista.

```
lista elim_lista(lista lst, int pos)
/* pre: lst=<a1,a2,...,a(pos),...,an>; n>0; 1<=pos<=n
 post: lst=<a1,a2,...,a(n-1)> */
{
      lista aux1, aux2;
      int cont;
      si la posición uno es igual a la que busca se librea el espacion en
      memoria
      if ( pos==1 )
      {
            aux2=lst;
            lst=lst->sgte;
            free(aux2);
      }
      else
      {
      Si no es asi se busca en cada una de las posiciones hasta que se
      encuentre y se libera.
            cont=2;
            aux2=lst;
            while ( ( cont!=pos ) && ( aux2->sgte!=NULL ) )
            {
                  aux2=aux2->sgte;
                  cont++;
            }
            if ( cont==pos )
            {
                  aux1=aux2->sgte;
                  aux2->sgte=aux1->sgte;
```

```
                free(aux1);
            }
            else
            {
                clrscr();
                puts("ERROR POSICION INCORRECTA.");
                sound(300);
                sleep(1);
                nosound();
                sleep(3);
                clrscr();
            }
        }
        return ( lst );
}

Obtener elemento de la lista.
elemento info_lista(lista lst, int pos)
/* pre: lst=<a1,a2,...,a(pos),...,an>; n>0; 1<=pos<=n
 post: a(pos) */
{
        lista aux;
        int cont;
        Se verifica si la lista esta vacia
        if ( vacia_lista(lst) )
        {
            clrscr();
            printf("Error: lista vacia.");
            sound(300);
            sleep(1);
            nosound();
            sleep(3);
            clrscr();
            exit(0);
        }
        aux=lst;
        cont=1;
        Si la lista no esta vacia se asigna los datos a la lista mientras
        el siguiente registro no sea NULL
        while ( ( cont!=pos ) && ( aux->sgte!=NULL ) )
        {
            aux=aux->sgte;
            cont++;
        }
        if ( cont!=pos )
        {
            clrscr();
            puts("ERROR POSICION INCORRECTA.");
            sound(300);
            sleep(1);
            nosound();
            sleep(3);
            clrscr();
        }
```

```
	return ( aux->info );
}
```

Metodo que retorna cuantos registros existen en la lista.

```
int long_lista(lista lst)
/* pre: lst=<a1,a2,...,an>
 post: n */
{
	lista aux;
	int cont;
	if ( lst==NULL )
	return ( 0 );
	else
	{
		aux=lst;
		cont=1;
		while ( aux->sgte!=NULL )
		{
			aux=aux->sgte;
			cont++;
		}
	}
	return ( cont );
}

/*_____________________________________________________________________*

		Implementacion de Funciones Privadas
*_____________________________________________________________________*/
lista inv_lista(lista lst)
/* pre: lst=<a1,a2,...,an>; n>0; 1<=pos<=n
 post: lst=<an,a(n-1),...,a1> */
{
	int cont, longitud;
	lista aux1, aux2;
	aux2=inic_lista();
	longitud=long_lista(lst);
	if ( !vacia_lista(lst) )
	{
		while ( longitud>0 )
		{
			cont=1;
			aux1=lst;
			while( cont<longitud )
			{
				aux1=aux1->sgte;
				cont++;
			}
			aux2=anx_lista(aux2, aux1->info);
			longitud--;
		}
	}
```

```
      return ( aux2 );
}

lista concatenar_lista(lista lst1, lista lst2)
/* pre: lst1=<a1,a2,...,an>; lst2=<b1,b2,...,bn>
 post: lst= <a1,a2,...,an,b1,b2,...,bn> */
{
      int pos, longitud;
      lista aux;
      longitud=long_lista(lst2);
      pos=1;
      aux=lst1;
      while ( pos<=longitud )
      {
            aux=anx_lista(aux, info_lista(lst2, pos));
            pos++;
      }
      return ( aux );
}

lista cambiar_lista(lista lst, int pos, elemento elem)
/* pre: lst=<a1,a2,...,a(pos-1),a(pos),a(pos+1),...,an>; n>0; 1<=pos<=n;
elem
 post: lst=<a1,a2,...,a(pos-1),elem,a(pos+1),...,a(n+1)> */
{
      lista nuevo, aux;
      int cont;
      nuevo=(lista)malloc(sizeof(struct nodo));
      if ( nuevo==NULL )
      {
            clrscr();
            printf("ERROR NO HAY MEMORIA DISPONIBLE.");
            sound(300);
            sleep(1);
            nosound();
            sleep(3);
            clrscr();
            exit(0);
      }
      nuevo->info=elem;
      nuevo->sgte=NULL;
      cont=1;
      aux=lst;
      while ( ( cont!=pos ) && ( aux->sgte!=NULL ) )
      {
            aux=aux->sgte;
            cont++;
      }
      if ( cont==pos )
      {
            aux->info=nuevo->info;
      }
      else
      {
```

```
            clrscr();
            puts("ERROR POSICION INCORRECTA.");
            sound(300);
            sleep(1);
            nosound();
            sleep(3);
            clrscr();
      }
      return ( lst );
}

lista intercambiar_lista(lista lst, int pospri, int posfin)
/* pre: lst=<a1,a2,...,a(pospri),...,a(posfin),...,an>; n>0
 post: lst=<a1,a2,...,a(posfin),...,a(pospri),...,an)> */
{
      lista pri, fin, aux;
      int cont=0;
      aux=(lista)malloc(sizeof(struct nodo));
      if ( aux==NULL )
      {
            clrscr();
            printf("ERROR NO HAY MEMORIA DISPONIBLE.");
            sound(300);
            sleep(1);
            nosound();
            sleep(3);
            clrscr();
            exit(0);
      }
      pri=lst;
      while ( cont<pospri )
      {
            pri=pri->sgte;
            cont++;
      }
      fin=pri;
      while ( cont<posfin )
      {
            fin=fin->sgte;
            cont++;
      }
      aux->info=fin->info;
      fin->info=pri->info;
      pri->info=aux->info;
      return ( lst );
}

lista dest_lista(lista lst)
/* pre: lst=<a1,a2,...,an>; n>0; 1<=pos<=n
 post: lst=< > */
{
      lista aux1, aux2;
      while ( !vacia_lista(lst) )
      {
```

```
            aux1=lst;
            while ( aux1->sgte->sgte!=NULL )
            {
                  aux1=aux1->sgte;
            }
            aux2=aux1;
            aux1=aux1->sgte;
            free(aux1);
            aux2->sgte=NULL;
            if ( lst->sgte==NULL )
            {
                  aux2=lst;
                  free(aux2);
                  lst=NULL;
            }
      }
      return ( lst );
}

elemento pre_lista(lista lst, int pos)
/* pre: lst=<a1,a2,...,a(pos-1),a(pos),...,an>; n>0; 1<=pos<=n
 post: a(pos-1) */
{
      int cont;
      lista aux;
      aux=lst;
      cont=2;
      while ( cont<pos )
      {
            aux=aux->sgte;
            cont++;
      }
      return ( aux->info );
}

elemento suc_lista(lista lst, int pos)
/* pre: lst=<a1,a2,...,a(pos),a(pos+1),...,an>; n>0; 1<=pos<=n
 post: a(pos+1) */
{
      int cont;
      lista aux;
      aux=lst;
      cont=2;
      while ( cont<=pos )
      {
            aux=aux->sgte;
            cont++;
      }
      aux=aux->sgte;
      return ( aux->info );
}

elemento prim_lista(lista lst)
/* pre: lst=<a1,a2,...,an>; n>0; 1<=pos<=n
```

```
 post: a1 */
{
      return ( lst->info );
}

elemento ult_lista(lista lst)
/* pre: lst=<a1,a2,...,an>; n>0; 1<=pos<=n
 post: an */
{
      lista aux;
      aux=lst;
      while ( aux->sgte!=NULL )
      {
            aux=aux->sgte;
      }
      return ( aux->info );
}

boolean vacia_lista(lista lst)
/* pre: lst TRUE
 post: si (lst==< >) => retorna (TRUE), sino => retorna (FALSE) */
{
      if ( lst==NULL )
      {
            return ( TRUE );
      }
      else
      {
            return ( FALSE );
      }
}
```

LISTA_AP.H

```
/*************************************************************************
                                    *****
                                      *
                        * INTERFAZ MODULO: lista_ap.h
*
* LUGAR Y FECHA DE CREACION: Santiago de Cali, Diciembre de 2002.
*
* Descripcion: TAD Lista: Implementado Sobre Apuntadores.
*
* Requerimientos: Se requiere del Algoritmo de Control, un tipo de datos
* llamado elemento, para la parte info del nodo.
*
*************************************************************************/

/*_______________________________________________________________________*

            Declaracion de Constantes y Macros, publicas
```

```
*_____________________________________________________________________*/
#define FALSE 0 /* Estado de verdad = falso */
#define TRUE 1 /* Estado de verdad = verdadero */

/*_____________________________________________________________________*

       Declaracion de Estructuras y Redefiniciones de Tipo, publicas
*_____________________________________________________________________*/
struct nodo
{
      elemento info;
      struct nodo *sgte;
};
typedef struct nodo *lista;

typedef int boolean;

/*_____________________________________________________________________*

             Prototipos de Funciones Publicas
*_____________________________________________________________________*/
/* Inicializadora */
lista inic_lista(void);
/* pre: true
 post: lst=< > */

/* Constructoras */
lista ins_lista(lista lst, int pos, elemento elem);
/* pre: lst=<a1,a2,...,a(pos-1),a(pos),a(pos+1),...,an>; n>0; 1<=pos<=n;
elem
 post: lst=<a1,a2,...,a(pos-1),elem,a(pos),a(pos+1),...,a(n+1)> */
lista anx_lista(lista lst, elemento elem);
/* pre: lst=<a1,a2,...,an>; elem
 post: lst=<a1,a2,...,an,elem> */

/* Destructora */
lista elim_lista(lista lst, int pos);
/* pre: lst=<a1,a2,...,a(pos),...,an>; n>0; 1<=pos<=n
 post: lst=<a1,a2,...,a(n-1)> */

/* Analizadoras */
elemento info_lista(lista lst, int pos);
/* pre: lst=<a1,a2,...,a(pos),...,an>; n>0; 1<=pos<=n
 post: a(pos) */
int long_lista(lista lst);
/* pre: lst=<a1,a2,...,an>
 post: n */

/*_____________________________________________________________________*

             Prototipos de Funciones Privadas
*_____________________________________________________________________*/
/* Constructoras */
lista inv_lista(lista lst);
```

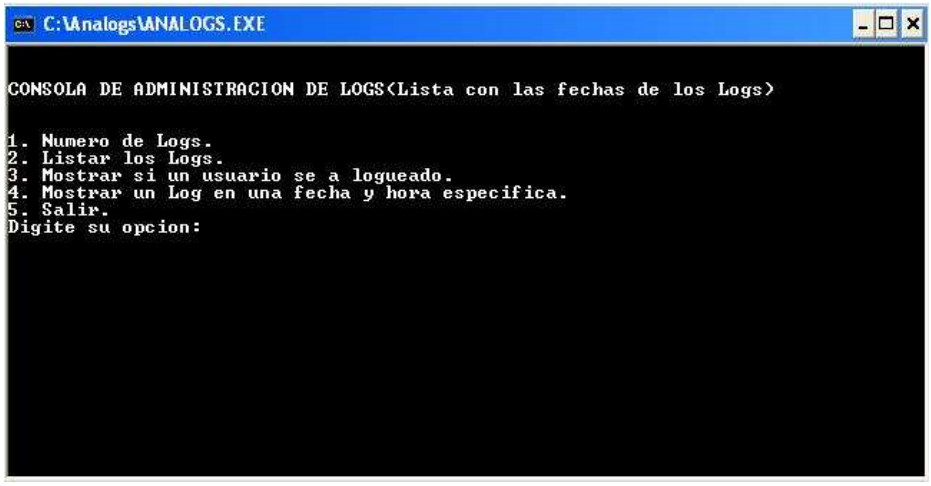
```
/* pre: lst=<a1,a2,...,an>
 post: lst=<an,a(n-1),...,a1> */
lista concatenar_lista(lista lst1, lista lst2);
/* pre: lst1=<a1,a2,...,an>; lst2=<b1,b2,...,bn>
 post: lst= <a1,a2,...,an,b1,b2,...,bn> */
lista cambiar_lista(lista lst, int pos, elemento elem);
/* pre: lst=<a1,a2,...,a(pos-1),a(pos),a(pos+1),...,an>; n>0; 1<=pos<=n;
elem
 post: lst=<a1,a2,...,a(pos-1),elem,a(pos+1),...,a(n+1)> */
lista intercambiar_lista(lista lst, int pospri, int posfin);
/* pre: lst=<a1,a2,...,a(pospri),...,a(posfin),...,an>; n>0
 post: lst=<a1,a2,...,a(posfin),...,a(pospri),...,an)> */

/* Destructora */
lista dest_lista(lista lst);
/* pre: lst=<a1,a2,...,an>
 post: lst=< > */

                            /* Analizadoras */
elemento pre_lista(lista lst, int pos);
/* pre: lst=<a1,a2,...,a(pos-1),a(pos),...,an>; n>0; 1<=pos<=n
 post: a(pos-1) */
elemento suc_lista(lista lst, int pos);
/* pre: lst=<a1,a2,...,a(pos),a(pos+1),...,an>; n>0; 1<=pos<=n
 post: a(pos+1) */
elemento prim_lista(lista lst);
                        /* pre: lst=<a1,a2,...,an>
 post: a1 */
elemento ult_lista(lista lst);
/* pre: lst=<a1,a2,...,an>
 post: an */
boolean vacia_lista(lista lst);
/* pre: lst TRUE
 post: si (lst==< >) => retorna (TRUE), sino => retorna (FALSE) */

#include "c:\analogs\lista_ap.c"
```

ANEXO 2
MANUAL DE USUARIO ANALOGS

Página inicial

Esta es la página de acceso a la aplicación, en la que se describe el nombre, la versión del programa y una breve introducción, para ingresar al menú de la aplicación es necesario presionar cualquier tecla.

Página de menú



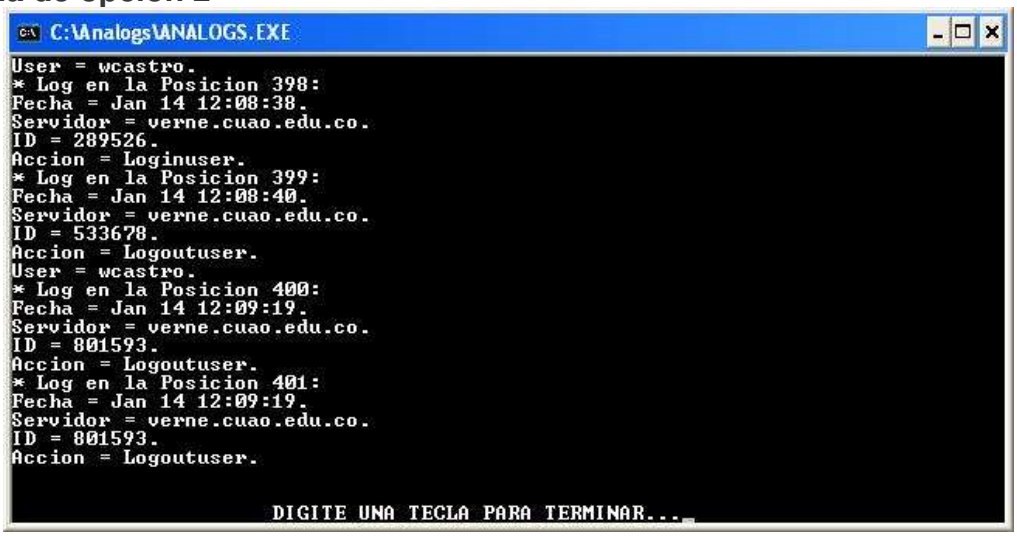
Esta es la página que muestra el menú principal de la aplicación, la cual tiene las siguientes opciones:

1. Número de Logs
2. Listar los Logs
3. Mostrar si un usuario se ha logueado.
4. Mostrar un Log en una fecha y hora especifica
5. Salir.

Página de opción 1

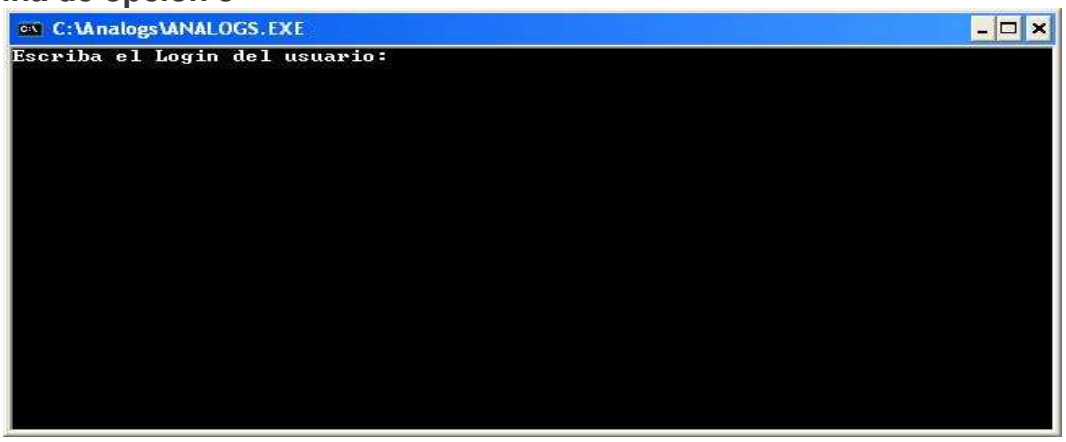
Esta página muestra la cantidad de registros que existen en el archivo de logs. Para terminar y regresar al menú principal solo es necesario presionar una tecla.

Página de opción 2

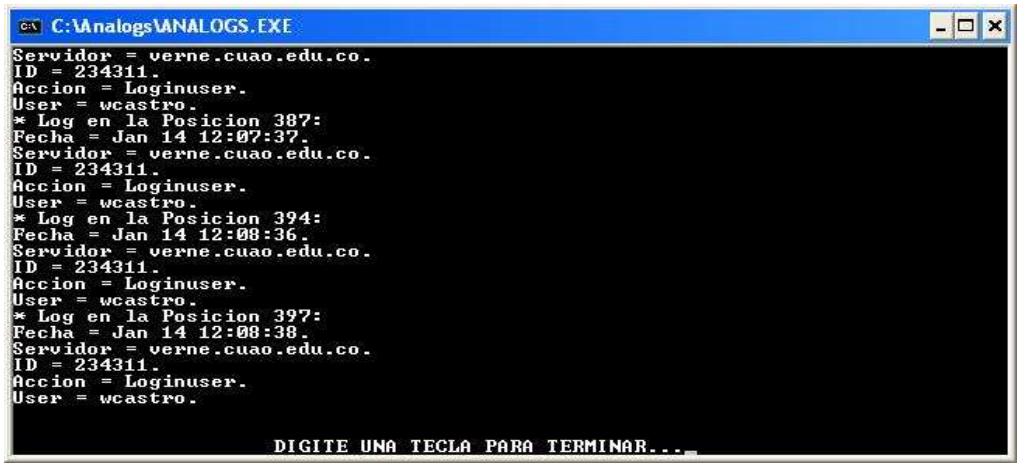


Esta página muestra todos los registros que existen en el archivo de logs, detallando el usuario, la posición en que fue encontrada el mismo, la fecha y hora exacta, el servidor y el ID. Para terminar y regresar al menú principal solo es necesario presionar una tecla.

Página de opción 3



Esta página permite mostrar los registros de un usuario con solo digitar su login. Una vez digitado el login se presiona la tecla ENTER y el sistema muestra todos los registros que existen en el archivo de logs para ese usuario, detallando la fecha y hora exacta, la posición, el servidor y el ID como se muestra en la siguiente imagen. Para terminar y regresar al menú principal solo es necesario presionar una tecla.



Página de opción 4

Esta página permite mostrar los registros que fueron efectuados en una fecha y hora exacta.

El usuario debe digitar el mes, el día, la hora, el minuto y segundo del que quiere ver el registro, una vez hecho esto se presiona la tecla ENTER y el sistema muestra todos los registros que existen en el archivo de logs para esa descripción de campos, detallando la fecha y hora exacta, la posición, el servidor, el ID y el usuario como se muestra en la siguiente imagen. Para terminar y regresar al menú principal solo es necesario presionar una tecla.

ANEXO 3
COMPLEJIDAD COMPUTACIONAL[24]

La Teoría de la complejidad computacional es la parte de la teoría de la computación que estudia los recursos requeridos durante el cálculo para resolver un problema. Los recursos comúnmente estudiados son el *tiempo* (número de pasos de ejecución de un algoritmo para resolver un problema) y el *espacio* (cantidad de memoria utilizada para resolver un problema). Se pueden estudiar igualmente otros parámetros, tales como el número de procesadores necesarios para resolver el problema en paralelo. La teoría de la complejidad defiere de la teoría de la computabilidad en que esta última se ocupa de la factibilidad de expresar problemas como algoritmos efectivos sin tomar en cuenta los recursos necesarios para ello.

Hoy en día las máquinas resuelven problemas mediante algoritmos que tienen como máximo una complejidad o coste computacional polinómico, es decir, la relación entre el tamaño del problema y su tiempo de ejecución es polinómica.

La complejidad en tiempo de un problema es el número de pasos que toma resolver una instancia de un problema, a partir del tamaño de la entrada utilizando el algoritmo más eficiente a disposición. Intuitivamente, si se toma una instancia con entrada de longitud n que puede resolverse en n^2 pasos, se dice que ese problema tiene una complejidad en tiempo de n^2. Por supuesto, el número exacto de pasos depende de la máquina en la que se implementa, del lenguaje utilizado y de otros factores.

24. http://es.wikipedia.org/wiki/Complejidad_computacional

Para no tener que hablar del costo exacto de un cálculo se utiliza la notación O. Cuando un problema tiene costo en tiempo $O(n^2)$ en una configuración de computador y lenguaje dado este costo sera el mismo en la mayoría de los computadores, de manera que esta notación generaliza la noción de coste independientemente del equipo utilizado

La complejidad en el tiempo fué la utilizada en nuestro cálculo

Tiempo de ejecución para el peor de los casos, en función de n, donde n es el número de caracteres del archivo LOGS.txt

INSTRUCCIÓN	T(n)
void anali- sis_logs(void)	
{	
struct Log Log;	0
struct Log2 Log2;	0
FILE *archivoTXT;	0
int num, j, pos=0, FechaValida=1, ServidorValido=1, UserValido;	3
char caracter;	0
if ((archivoTXT=fopen("d:\\tc\\tmp\\logs.txt","rt")) == NULL)	1
{	
printf("\n\n\nERROR!!! EL ARCHIVO NO PUDO SER ABIER- TO\n");	
printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");	
getch();	
fclose(archivoTXT);	
salir();	
return;	
}	
fseek(archivoTXT,0L,SEEK_END);	1
if (ftell(archivoTXT)==0)	1
{	
printf("\n\n\nERROR! EL ARCHIVO ESTA VACIO\n");	
printf("\nDIGITE UNA TECLA PARA TERMINAR...\n");	
getch();	
fclose(archivoTXT);	
salir();	
}	
rewind(archivoTXT);	1
while (!feof(archivoTXT))	n+1

```
{
        caracter=fgetc(archivoTXT);                                     n
        if(caracter=='\n')                                              n
        {
                caracter=fgetc(archivoTXT);                             n
                if(caracter>=65 && caracter<=90)                        n
                {
                        j=0;                                            n
                        while(j<3)                                      4n
                        {
                                Log.Mes[j]=caracter;                    3n
                                caracter=fgetc(archivoTXT);             3n
                                j++;                                    3n
                        }
                        Log.Mes[3]='\0';                                n
                        j=0;                                            n
                        if(caracter>=65 && caracter<=90)                n
                        {
                                FechaValida=0;                          n
                                pos--;                                  n
                        }
                        while(j<2)                                      3n
                        {
                                Log.Dia[j]=caracter;                    2n
                                caracter=fgetc(archivoTXT);             2n
                                j++;                                    2n
                        }
                        Log.Dia[2]='\0';                                n
                        caracter=fgetc(archivoTXT);                     n
                        j=0;                                            n
                        while(j<2)                                      3n
                        {
                                Log.Hora[j]=caracter;                   2n
                                caracter=fgetc(archivoTXT);             2n
```

```
                    j++;                                                    2n
                }
                Log.Hora[2]='\0';                                           n^2
                caracter=fgetc(archivoTXT);                                 n^2
                j=0;                                                        n^2
                while(j<2)                                                  3n
                {
                    Log.Min[j]=caracter;                                    2n
                    caracter=fgetc(archivoTXT);                             2n
                    j++;                                                    2n
                }
                Log.Min[2]='\0';                                            n^2
                caracter=fgetc(archivoTXT);                                 n^2
                j=0;                                                        n^2
                while(j<2)                                                  3n
                {
                    Log.Seg[j]=caracter;                                    2n
                    caracter=fgetc(archivoTXT);                             2n
                    j++;                                                    2n
                }
}
Log.Seg[2]='\0';                                                            n
j=0;                                                                        n
while(j<17)
{                                                                           18n
                caracter=fgetc(archivoTXT);                                 17n
                if((caracter>=97 && caracter<=122) || (caracter>=65 && carac-
                ter<=90) || caracter=='.' || caracter==' ')                 17n
                    Log.Servidor[j]=caracter;                               17n
                else
                {
                    ServidorValido=0;
                    pos--;
```

	break;	
	}	
	j++;	17n
}		
Log.Servidor[17]='\0';		n
caracter=fgetc(archivoTXT);		n
while(caracter!=':')		n(n+1)= n^2 + n
	caracter=fgetc(archivoTXT);	n^2
j=0;		n
while(j<5)		5n
{		
	caracter=fgetc(archivoTXT);	4n
	j++;	4n
}		
j=0;		n
while(j<6)		6n
{		
	caracter=fgetc(archivoTXT);	5n
	Log.ID[j]=caracter;	5n
	j++;	5n
}		
Log.ID[6]='\0';		n
caracter=fgetc(archivoTXT);		n
while(caracter!=']')		n(n+1)= n^2 + n
	caracter=fgetc(archivoTXT);	n^2
caracter=fgetc(archivoTXT);		n
caracter=fgetc(archivoTXT);		n
if(caracter=='i')		
	strcpy(Log.Accion, "imap serviceinit");	
if(caracter=='A')		
	strcpy(Log.Accion, "Authenticateduser");	

el peor de los casos es que los : esten al final

el peor de los casos es que los] esten al final

```
if(caracter=='C')
                    strcpy(Log.Accion, "Connectionreset by peer");
j=0;                                                                    n
if(caracter=='L')                                                       n
{
                    j=0;                                                n
                    while(j<3)                                          3n
                    {
                                caracter=fgetc(archivoTXT);             2n
                                j++;                                    2n
                    }
                    if(caracter=='o')                                   n
                    {
                                strcpy(Log.Accion, "Logoutuser");
                                j=-1;
                    }
                    else
                    {
                                strcpy(Log.Accion, "Loginuser");        n
                                j=0;                                    n
                    }
                    while(j<6)                                          6n
                    {
                                caracter=fgetc(archivoTXT);             5n
                                j++;                                    5n
                    }
                    j=0;                                                n
                    while(caracter!=' ')                                n(n+1)= n^2 + n
                    {
                                caracter=fgetc(archivoTXT);             n^2
                                if((caracter>=97 && caracter<=122) || (carac-
                                ter>=65 && caracter<=90) || caracter==' ')   n^2
                                            Log.User[j]=caracter;       n^2
                                else
```

el peor de los casos es que el caracter esté al final

```
                                        {
                                                UserValido=0;
                                                break;
                                        }
                                        j++;                                    n^2
                                }
                                j--;                                            n
                        }
                        if(!UserValido)                                         n
                                j=0;                                            n
                        Log.User[j]='\0';                                       n
                        if((pos<400) && FechaValida && ServidorValido)          n
                                fecha = anx_lista(fecha, Log);                  n(T(anx_lista))   Si no dejarlo como orden constante = 0
                        if(pos==117)                                            n
                                pos=117;                                        n
                        pos++;                                                  n
                        FechaValida=1;                                          n
                        UserValido=1;                                           n
                        ServidorValido=1;                                       n
                }
        }
        Fclose(archivoTXT);                                                     1
        consola_administracion();                                               T(consola_administracion)   Si no dejarlo como orden constante = 0
}
```

Evaluar la funcion T(n)

La suma de la columna T(n) nos da el orden de la complejidad computacional

$O(n) = n^2$

ANEXO 4
CARTA DE LA UNIVERSIDAD

----- Original Message -----
From: Jorge Ernesto Peñaloza A.
To: Carlos Alberto Murillo Murillo; hfpanesso@cuao.edu.co
Sent: Monday, August 23, 2004 3:24 PM
Subject: Informe Estadísticas Web

Dr. Carlos Alberto Murillo
Dr. Héctor Fabio Panesso

Cordial Saludo.

Por la presente me permito agradecer a ustedes por el envío del documento "ESTADÍSTICAS DEL SERVIDOR WEB UNIVERSIDAD AUTÓNOMA DE OCCIDENTE", el cual considero un gran aporte de su parte para el mejoramiento y fortalecimiento de los servicios de red que presta la Universidad.

El documento en mención ha sido envíado para su análisis a los funcionarios del Departamento de Redes, quienes una vez analizado están de acuerdo con sus recomendaciones y desean profundizar con ustedes el alcance del proyecto y de las herramientas por ustedes empleadas.

Cordialmente,

Jorge Ernesto Peñaloza
División de Tecnologías
Universidad Autónoma de Occidente
E-mail: jorgep@cuao.edu.co