A. Contextualización

Internet está cambiando los ámbitos de negocio de muchas empresas. Conocer los intereses y los hábitos de navegación de los usuarios es ahora una de las claves para el éxito empresarial. Y, para ello, la mejor herramienta de la que disponemos es el análisis del tráfico de la red. Por ello, hemos recibido de una empresa de desarrollo de software el encargo de crear el motor de un analizador de accesos web.

Cualquier servidor web almacena en su disco duro un fichero –conocido como **log de accesos**– que describe las peticiones que recibe de los usuarios, junto con la valiosa información que necesitamos: fecha y hora de las peticiones, dirección IP del cliente, resultado, bytes enviados, etc. Nuestro objetivo es crear una herramienta para procesar estos ficheros, ofreciendo al usuario operaciones de consulta y análisis del tráfico. Para ello se deberá hacer un cuidadoso análisis del problema y un diseño adecuado de la organización y estructuración de la información manejada, con el objetivo de maximizar la eficiencia, tanto en consumo de memoria como en tiempos de ejecución.

B. Enunciado del problema

Un **servidor web** contiene un conjunto de ficheros (páginas web, imágenes, documentos, etc.) que, a su vez, pueden estar organizados en subdirectorios. El servidor recibe **peticiones** por parte de los **clientes** y responde en consecuencia, enviando el fichero al cliente o indicándole que ha habido algún problema. Además, almacena en su disco duro, en un fichero de texto, una nueva línea con la petición recibida y el resultado obtenido. Este fichero es llamado el **log (o bitácora) de accesos**.

Nuestra tarea es crear un motor para el análisis de ficheros de log, que incluya operaciones para cargar ficheros de log, seleccionar las peticiones según determinado criterio, y mostrar todas las peticiones o agrupadas de acuerdo a cierto criterio. En el diseño de la solución es importante tener en cuenta los grandes volúmenes de información que manejamos. Por ejemplo, al cabo de un día un servidor web típico puede recibir más de 20.000 peticiones, de unos 2.000 usuarios distintos, siendo transferido por la red más de 1 Gygabyte. En total, el log de accesos puede ocupar en torno a 5 Megabytes al día. A pesar de ello, necesitamos que la carga del fichero y la realización de consultas se realicen lo más rápidamente posible. Por ello, se debe afinar en la elección de las estructuras de datos y los algoritmos a utilizar.

La **carga de un fichero de log de accesos** es una operación mediante la cual se va leyendo el fichero y se crean en memoria las estructuras necesarias para las consultas posteriores. Esta carga también puede ser incremental, es decir, añadir un nuevo fichero sin eliminar las peticiones que estuvieran cargadas previamente.

Una vez cargado un fichero con sus peticiones, disponemos de una **operación de selección** para quedarnos con las peticiones que nos interesen; por ejemplo, seleccionar las peticiones a una página concreta, las que ha realizado cierto usuario, las de usuarios mejicanos, las que han provocado un error, etc. En cualquier momento de la ejecución habrá un subconjunto de peticiones seleccionadas del total de peticiones existentes (que podemos denominar **"la selección actual"**). Se pueden combinar selecciones con los operadores AND y OR; por ejemplo, podemos seleccionar las peticiones realizadas desde Rusia a las páginas de "`ginesgm`". La combinación AND/OR se hará siempre entre la selección actual y los resultados de una nueva operación de selección.

Sobre las peticiones seleccionadas disponemos de **operaciones de visualización**, para mostrar por pantalla los datos asociados a las mismas. La visualización puede ser completa o agrupada. En la primera, se listarán todas las peticiones con sus datos correspondientes, o bien un número limitado de las mismas (por ejemplo, de la 1 a la 100, o de la 101 a la 200). En la visualización agrupada elegimos un criterio de agrupamiento y se muestra el número de peticiones en cada grupo. Por ejemplo, si agrupamos por días deberá aparecer el número de peticiones al día; si agrupamos por usuario se mostrará el número de peticiones realizado por cada usuario distinto, etc.

C. Especificaciones del producto

El motor de análisis desarrollado debe ajustarse a las siguientes especificaciones. Estas especificaciones son de obligado cumplimiento, salvo las que se indiquen como opcionales. Se entiende que todo lo que no esté especificado se deja a la libre elección del implementador, en función de su criterio profesional. En cualquier caso, las decisiones a este respecto deberán estar justificadas adecuadamente.

C.1. Formato del log de accesos

El **log de accesos** es un fichero de texto ASCII en el que se describen las peticiones recibidas en el servidor y la acción llevada a cabo para cada una. Cada **petición** se escribe en una línea de texto. Las peticiones aparecen en el fichero por orden de llegada.

El formato de las líneas almacenadas en el fichero de log (las peticiones) es el siguiente:

ip - - fecha petic codres bytes proced agente

Todos los campos anteriores están separados por espacios en blanco. Los guiones (**- -**) aparecen literalmente. Se deberá ignorar cualquier línea que no contenga todos los campos mencionados. El significado y formato de los campos es el siguiente:

ip

Dirección IP del cliente que realiza la solicitud. Puede contener números, letras, puntos y guiones, pero no puede contener espacios en blanco. Por ejemplo, son direcciones IP válidas:

```
ginesgm.dif.um.es
155.54.12.197
dup-148-233-103-72.prodigy.net.mx
205sdl30m18.codetel.net.do
```

Las direcciones IP usan un formato de sufijos, que están delimitados por puntos. El primer sufijo es conocido como el **dominio** (por ejemplo: es, mx, com) y el segundo como **subdominio** (por ejemplo: um.es, net.mx, google.com).

fecha

Fecha y hora en la que se realiza la petición. Está delimitada por corchetes. El formato es el siguiente:

[*dd***/***mm***/***aaaa***:***hh***:***nn***:***ss* **+***xxxx***]**

Los símbolos y espacios aparecen de manera literal. Los campos en cursiva son:

dd → día, siempre con 2 dígitos

mm → mes: `Jan`, `Feb`, `Mar`, `Apr`, `May`, `Jun`, `Jul`, `Aug`, `Sep`, `Oct`, `Nov`, `Dec`

aaaa → año, con 4 dígitos

hh → hora, en formato de 0 a 23

nn → minutos, con 2 dígitos

ss → segundos, con 2 dígitos

xxxx → uso horario (no tenerlo en cuenta en esta práctica)

petic

Fichero solicitado por el cliente. Este campo está delimitado por comillas. Contiene varios subcampos separados por espacios, pero dentro de los subcampos no aparece ningún espacio. Será normalmente de la forma:

"GET *fichero* **HTTP/***xxx***"**

O bien:

"POST *fichero* **HTTP/***xxx***"**

fichero → fichero que se solicita, puede ser una página web, una imagen, un documento de texto, un vídeo, un directorio, etc.

xxx → versión del protocolo utilizada (no tenerlo en cuenta en esta práctica)

codres

Código del resultado. Será un número entero de 3 cifras que indica si la solicitud se ha realizado correctamente. Por ejemplo: 200 = correcto, 403 = acceso prohibido, 404 = página no encontrada, etc. Para más información se puede ver:
`http://www.helpwithpcs.com/courses/html/html_http_status_codes.htm`
aunque el programa no debe tener conocimiento del significado de los códigos.

bytes

Número de bytes enviados por la red en la respuesta al cliente. Será un número entero, desde 0 hasta varios miles de millones. En caso de ser 0 también puede aparecer un guión (**-**).

proced

Procedencia de la petición. Indica la página de la cual procede la petición del cliente. Por ejemplo, supongamos que el cliente pincha un enlace en la página A que le lleva hasta la página B. Entonces, al hacer la petición de B, el cliente indica que la procedencia es A. Este campo está delimitado por comillas y no contiene espacios en blanco. Si no hay procedencia aparecerá **"-"** (por ejemplo, cuando el usuario escribe directamente la dirección en la barra del navegador). Por ejemplo, para un usuario que accede a nuestro servidor desde Google, la procedencia podría ser:
`"http://www.google.es/search?hl=es&q=Gines+Garcia+Mateos&meta="`

agente

Información sobre la máquina desde la que el cliente realiza la petición. Normalmente describe el navegador y el sistema operativo del ordenador del cliente, aunque cada aplicación puede poner el texto que se le ocurra. Este campo está delimitado por comillas y puede contener espacios en blanco. Por ejemplo, podemos encontrar las siguientes cadenas:

```
"Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 4.0)"
"Mozilla/5.0 (X11; U; Linux i686; es-ES; rv:1.4.2) Gecko/20040308"
"Googlebot/2.1 (+http://www.google.com/bot.html)"
"FAST Enterprise Crawler/6 (www.fastsearch.com)"
```

C.2. Sinónimos para los símbolos especiales

Hay que tener en cuenta que, tanto en el campo *fichero* (dentro de *petic*) como en *proced*, se pueden utilizar sinónimos para escribir los símbolos especiales, como "/", "~", "+", etc. La forma alternativa consiste en la secuencia: **%XX** siendo XX un número

en hexadecimal de 2 cifras, que corresponde al código ASCII del carácter. El programa debe ser capaz de detectar estas secuencias en la entrada, sustituyéndolas por los símbolos correspondientes. Por ejemplo, sustituir `%7E` (o `%7e`) por `~`; `%2F` por `/`, etc.

C.3. Interface del programa

Todas las interacciones entre el programa y el usuario se realizarán utilizando la entrada y la salida estándar. No se creará un interface complejo con el usuario de la aplicación. En este sentido, el programa debe poder admitir un uso *off-line*, esto es, redireccionando la entrada y la salida estándar a ficheros.

En particular, la interface del programa será una línea de comandos que deberá aceptar, por lo menos, las siguientes órdenes:

>> load *FICHERO_LOG*

Cargar el fichero con nombre *FICHERO_LOG*. Esta operación servirá para inicializar en memoria las estructuras de datos necesarias para el programa. Si ya hay peticiones cargadas de otros ficheros de log, se deberán acumular, es decir, no se borran las antiguas. Al acabar la lectura se deberá mostrar información resumida con, al menos: número de peticiones leídas, tiempo total de carga (en milisegundos) y memoria utilizada por el programa (en Kbytes). Opcionalmente se mostrará también el número de páginas, días y usuarios distintos, la cantidad total de datos enviados (es decir, la suma de los campos *bytes*) en Mbytes, y otra información estadística que se considere interesante. Tras esta operación, se supone que la "selección actual" son todas las peticiones.

>> clear

Elimina todas peticiones cargadas en operaciones `load` anteriores, liberando la memoria ocupada por el programa.

>> select *CRITERIO VALOR*

Selecciona, de entre todas las peticiones cargadas por operaciones `load`, aquellas que cumplan el criterio de selección dado por los campos *CRITERIO* y `VALOR`. Los posibles valores de `CRITERIO` y `VALOR` y su significado son los siguientes:

Criterio	Valor	Tipo	Significado	Ejemplo
cliente	DIRECCION	Obligatorio	Seleccionar las peticiones en las que la dirección IP (campo *ip*) coincida con la cadena DIRECCION.	`select cliente ginesgm.dif.um.es` `select cliente 155.54.1.292`

dominio	DOMINIO	Obligatorio	Peticiones en las que el sufijo de la dirección IP coincida con DOMINIO. Los sufijos están separados por puntos, de manera que DOMINIO comienza por el primer carácter después de un punto (ver ejemplos). Puede referirse a un dominio, subdominio, sub-subdominio, etc.	`select dominio dif.um.es`
				`select dominio um.es`
				`select dominio es`
				`select dominio 1.292`
				`select dominio dis.um.es`

fecha	INICIO FIN	Obligatorio	Peticiones en las que la fecha esté entre INICIO y FIN, ambas inclusive. Las fechas tendrán el formato dd/mm/aaaa, siendo los tres campos de tipo numérico.	`select fecha 01/01/2004`
				`  31/12/2004`
				`select fecha 20/10/2002`
				`  30/10/2002`

fichero	NOMBRE	Obligatorio	Peticiones en las que el fichero (dentro del campo *petic*) coincida con la cadena NOMBRE.	`select fichero /index.html`
				`select fichero /`
				`select fichero /~norberto/ad.html`
				`select fichero /grupos/parp/a.htm`

prefich	PREFIJO	Opcional	Peticiones en las que el fichero (dentro del campo *petic*) empiece con la cadena PREFIJO.	`select prefich /~domingo`
				`select prefich /~ginesgm`
				`select prefich /grupos/parp`

subfich	SUBCAD	Opcional	Peticiones en las que el fichero (dentro del campo *petic*) contenga como subcadena la cadena dada en SUBCAD.	`select subfich .html`
				`select subfich aed`
				`select subfich docencia`

codigo	VALOR	Obligatorio	Peticiones en las que el código de resultado (campo *codres*) coincida con el entero VALOR.	`select codigo 200` `select codigo 404`

bytes	MIN MAX	Opcional	Peticiones en las que el número de bytes enviados (campo *bytes*) esté entre los valores dados por los enteros MIN y MAX, ambos inclusive.	`select bytes 0 220`
				`select bytes 1242 16000`
				`select bytes 1000000 10000000`

proced	CADENA	Opcional	Peticiones en las que la procedencia (campo *proced*) contenga como subcadena el valor CADENA.	`select proced www.google.com` `select proced www.um.es`

maquina	CADENA	Opcional	Peticiones en las que la máquina (campo *agente*) contenga como subcadena el valor CADENA.	`select maquina Mozilla`
				`select maquina Linux`
				`select maquina Crawler`

Observar que algunos criterios de selección son de obligada implementación, mientras que otros se dejan como opcionales.

Al finalizar la operación se debe mostrar el número de peticiones que han sido seleccionadas, la suma de los campos *bytes* de esas peticiones, y el tiempo que se ha tardado en realizar la operación (medido en milisegundos), cada uno de ellos en una línea distinta. Por ejemplo, podríamos tener algo como lo siguiente:

```
>> select fichero /~ginesgm/museo.html
213 peticiones seleccionadas (2.75% del total)
342115 bytes enviados
0.142 milisegundos
```

>> selectand *CRITERIO VALOR*

La sintaxis de esta operación (es decir, los campos *CRITERIO* y *VALOR*), el significado y el funcionamiento son los mismos que para el comando `select`, pero en lugar de trabajar con todas las peticiones existentes se restringe a las seleccionadas actualmente. Esto es, sería un AND entre la selección actual y la selección dada por los campos *CRITERIO* y *VALOR*. Por ejemplo, para seleccionar las peticiones realizadas desde el dominio "`ru`" a las páginas de "`/~ginesgm`", deberíamos ejecutar los siguientes comandos:

```
>> select dominio ru
>> selectand prefich /~ginesgm
```

>> selector *CRITERIO VALOR*

Igual que el anterior pero con un OR. La sintaxis y el significado de los parámetros son los mismos que para el comando `select`; en este caso, el resultado sería la unión entre la selección actual y las peticiones que cumplan la consulta dada por los campos *CRITERIO* y *VALOR*. Por ejemplo, para seleccionar las peticiones realizadas a los directorios "`ginesgm`", "`domingo`" o "`nmarin`", deberíamos ejecutar los siguientes comandos:

```
>> select subfich ginesgm
>> selector subfich domingo
>> selector subfich nmarin
```

>> listall

Muestra por pantalla la lista de todas las peticiones seleccionadas actualmente, ordenadas por fecha, de menor a mayor. En primer lugar aparecerá una línea indicando el número de peticiones seleccionadas. A continuación vendrán las peticiones, cada una en una línea. Cada línea empezará por un número consecutivo (1, 2, 3, ...), y a continuación aparecerán los campos leídos del fichero de log, con el siguiente formato:

numero. ip fecha hora fichero proced agente

Finalmente aparecerá el tiempo tardado en ejecutar la operación, medido en milisegundos. Por ejemplo, el resultado de un listado podría ser el siguiente:

```
>> listall
Listando 21 peticiones seleccionadas
1. ginesgm.um.es 11/10/2002 12:34 /index.html "-" "Mozilla"
2. dif.ono.es 11/10/2002 12:35 /dos.html "www.um.es" "Linux 4"
...
21. halos.bda.com 11/10/2002 12:35 /us.html "um.es" "MS IE3.2"
2.432 milisegundos
```

Opcionalmente, a efectos de mejorar la visualización, se puede implementar algún parámetro o comando adicional para mostrar sólo algunos de los campos.

>> list *PRIMERA CANTIDAD* (comando opcional)

PRIMERA y *CANTIDAD* son dos números naturales. El resultado es igual que el comando `listall`, pero en lugar de listar todas las peticiones, lista *CANTIDAD* peticiones empezando por *PRIMERA*. Por ejemplo, el comando:

```
>> list 100 20
```

mostraría las peticiones desde la 100 hasta la 119. Se entiende que las peticiones están ordenadas por fecha, siendo la primera (la más antigua) la número 1. Igual que antes, se debe mostrar el tiempo tardado en ejecutar la operación.

>> group *CAMPO*

De entre todas las peticiones seleccionadas actualmente, muestra los distintos valores de *CAMPO* existentes, y cuántas peticiones tienen ese valor. Por ejemplo, si el campo es **dia** se deberá mostrar una lista de los días existentes en las peticiones seleccionadas y el número de peticiones que hay en cada uno. Finalmente aparecerá el tiempo tardado. Por ejemplo, una ejecución podría ser:

```
>> group dia
4 días distintos
1. 11/10/2004 67
2. 12/10/2004 17
3. 13/10/2004 98
4. 14/10/2004 72
0.218 milisegundos
```

Los valores de *CAMPO* y el significado son los siguientes:

Campo	Significado
dia	Agrupar por día
mes	Agrupar por mes
cliente	Agrupar por el valor de IP
dominio	Agrupar por dominio de dirección IP, es decir, considerando sólo un elemento de sufijo (por ejemplo, "es", "com", "net", etc.)
subdominio	Agrupar por subdominio de dirección IP, es decir, considerando dos elementos de sufijo (por ejemplo, "um.es", "ono.com", "google.es", etc.)
fichero	Agrupar por nombre de fichero solicitado
directorio	Agrupar por directorio del fichero solicitado. Se supone que el directorio es igual

	al nombre pero eliminando lo que aparezca después del último "/". Por ejemplo, el directorio de "`/~ginesgm/files/uno.html`" sería "`/~ginesgm/files/`"
`codigo`	Agrupar por el código de resultado de la petición

Es obligatorio implementar al menos uno de los campos de agrupamiento, a elegir por los alumnos. Los demás quedan como opcionales. Asimismo, se puede añadir alguno más que se considere de interés.

`>> help`

Mostrar información sobre todos los comandos disponibles. Documentar especialmente los comandos opcionales y los nuevos, en caso de que se haya introducido alguno.

`>> quit`

Salir del programa, liberando la memoria que se haya utilizado.

C.4. Requisito adicional

Adicionalmente, se impone el siguiente requisito en referencia a las estructuras de datos utilizadas en el programa. Sea **P** la suma de los números de DNI de los miembros del grupo de prácticas. La práctica deberá implementar y utilizar al menos la siguiente estructura de datos (aunque se valorará el uso de otras), en función del valor de **D** obtenido con la fórmula: $\mathbf{D = \lfloor P / 37 \rfloor \bmod 4}$.

$\mathbf{D = \lfloor P / 37 \rfloor \bmod 4}$	**Estructura de datos**
0	Árboles trie
1	Árboles AVL
2	Dispersión abierta
3	Dispersión cerrada

D. Memoria de la práctica

La memoria de la práctica deberá contener **obligatoriamente** los siguientes apartados.

D.1. Portada

Nombre de los alumnos, dirección de e-mail y número de prácticas de cada uno y el valor **D** descrito en el apartado **C.4**. En la cara opuesta a la portada deberá aparecer el login y el password de la cuenta en la que están los ficheros de la práctica.

D.2. Análisis del problema

Por análisis se entiende el estudio preliminar del problema. Incluye:

- Estudio de los requisitos del problema tratado. Describir en palabras propias el problema a tratar, centrándose en los aspectos más relevantes. Abstraer. Definir las partes de la especificación que sean ambiguas o incompletas. Decidir las partes opcionales que se realizan.
- Estudio de los tipos abstractos necesarios para resolver el problema y las operaciones sobre los mismos. Encontrar los tipos abstractos que aparecen en el problema. Analizar las diferentes alternativas que se presentan para la implementación de estos tipos.

- Especificación de los tipos abstractos. Especificar los principales tipos abstractos de datos encontrados. Se puede usar un formato propio de especificación.

D.3. Diseño de la aplicación

El diseño es la toma de decisiones respecto a la estructura del programa a implementar. Incluye:

- Estructura de datos para cada tipo de datos. Decidir qué estructura de datos se va a utilizar para cada tipo, justificando la decisión. Particularizar las características generales de la estructura para el problema concreto, indicando posibles modificaciones y adaptaciones. Mostrar un esquema gráfico global de la estructura de tipos de datos existentes.
- Algoritmos sobre los tipos de datos. Descripción abstracta de los algoritmos utilizados, correspondientes a cada operación del analizador.
- Descomposición modular del programa. Decidir cómo se organiza la implementación del programa. Qué módulos existen. Cuál es la responsabilidad de cada uno. Qué relación de uso hay entre los módulos.
- Documentar cualquier otra decisión de diseño que pueda resultar de interés.

D.4. Listado del código

Entregar un listado del código, comentado, incluyendo ficheros de cabecera, de implementación y el fichero `makefile` necesario para compilar el programa.

D.5. Informe de desarrollo

El informe de desarrollo es un documento sobre el proceso personal y del grupo para la resolución de la práctica. Debe contener lo siguiente:

- Organización temporal de las fases de resolución (análisis, diseño, implementación, pruebas), no de forma genérica sino para esta práctica en concreto.
- Cómo ha sido la coordinación y el reparto del trabajo entre los miembros del grupo. Concretar personas y tareas realizadas.
- Tablas de dedicación personal en las distintas fases del trabajo. Se utilizarán tablas como las explicadas en las páginas 37 y 350 del texto guía, rellenadas con el mayor rigor posible.
- Conclusiones y valoraciones personales.

E. Evaluación de la práctica

E.1. Obligatorio

Para aprobar la práctica se requiere que:

- El programa se pueda compilar sin errores en las máquinas del laboratorio de prácticas, en la fecha y hora en la que se realice la entrevista con los alumnos.
- El programa debe funcionar correctamente, sin colgarse y produciendo resultados correctos para el conjunto de pruebas que determine el profesor.
- La interface del programa debe ajustarse, como mínimo, a la especificación del apartado C.
- La memoria de la práctica debe contener todos los puntos indicados en el apartado D. La memoria debe ser entregada en el plazo que se establezca.

E.2. Criterios de valoración

La práctica se puntuará de acuerdo a los siguientes criterios de calidad del software:

- Análisis y diseño. Se valorará la calidad y adecuación del diseño y el análisis realizados, y la dedicación a estas fases previas a la implementación.

- Modularidad. La funcionalidad debe estar bien repartida entre los módulos. Debe estar claro el sentido y la responsabilidad de cada módulo. Se debe respetar el principio de ocultación de la implementación.
- Abstracciones. Se deben encontrar los tipos abstractos que aparecen, e implementarlos usando clases, de forma adecuada.
- Implementación. Se valorará positivamente la implementación de más de una estructura de datos, siempre que resulte conveniente.
- Uso del lenguaje. El código deber ser claro, legible, robusto y eficiente. No crear procedimientos muy largos y complejos. Se valorará el uso de clases genéricas (plantillas) y precondiciones / postcondiciones (asertos).

Los comandos obligatorios contarán un 75% sobre la nota total de la práctica. Para superar la práctica deben estar implementados todos ellos. Los comandos y criterios opcionales contarán el 25% restante. Se entiende que para aspirar al máximo posible es suficiente con realizar en torno a 1/3 de los comandos y criterios opcionales.

E.3. Otras cuestiones

La práctica se deberá realizar en grupos de dos alumnos. Se permite también la realización en grupos de tres alumnos. En este caso, deberán implementarse necesariamente 1/3 de los comandos y criterios opcionales.

Para realizar las pruebas, el profesor dejará en la página web de la asignatura (http://dis.um.es/~ginesgm/aed.html) ficheros de prueba suficientemente grandes.

Se establece como fecha tope de entrega de esta práctica el viernes 25 de febrero de 2005. Las fechas de las entrevistas se fijarán con posterioridad.