

**ESCUELA SUPERIOR
POLITECNICA DEL LITORAL**

FACULTAD DE INGENIERIA EN ELECTRICIDAD

**"DISEÑO E IMPLEMENTACION DE UN
SISTEMA DE INFORMACION PARA LA ESPOL
'SI-ESPOL', BASADO EN LA APLICACION
GOPHER DE LA UNIVERSIDAD DE
MINNESOTA"**

TESIS DE GRADO

Previa a la obtención del Título de:

INGENIERO EN COMPUTACION

Presentada por:

WILLIAM JARA M.

**GUAYAQUIL-ECUADOR
1994**

AGRADECIMIENTO

*Al ING. Jaime
Puente Director de Tesis,
por su ayuda y
colaboración para la
realización de este
trabajo:*

*DEDICATORIA**A mis padres*



ING. ARMANDO ALTAMIRANO
PRESIDENTE DEL TRIBUNAL



ING. JAIME PUENTE
DIRECTOR TESIS



ING. GUIDO CAICEDO
MIEMBRO TRIBUNAL



ING. SIXTO GARCIA
MIEMBRO TRIBUNAL

DECLARACION EXPRESA

"La responsabilidad por los hechos, ideas y doctrinas expuestos en esta tesis, me corresponden exclusivamente; y, el patrimonio intelectual de la misma, a la ESCUELA SUPERIOR POLITECNICA DEL LITORAL".

(Reglamento de Exámenes y Títulos profesionales de la ESPOL).

William Jara Morante

RESUMEN

El propósito del SI-ESPOL es el de dar a usuario una interfase amigable para que se pueda manejar información de una manera rápida y confiable. Con el SI-ESPOL podremos mantener una base de información propia de la ESPOL y que sea en un principio administrada por CESERCOMP y en un futuro cercano que cada unidad académica pueda administrar sus datos.

En pocas palabras la ESPOL contará con un sistema de información que sea de fácil y de rápido acceso desde cualquier red, PC o terminal conectado a INTERNET, a pesar de que estos usuarios esten en distintas plataformas (UNIX, NOVELL, MAC, DOS) además de darnos en muchos casos interfases gráficas.

Este sistema de información tendrá como base el software de la Universidad de Minnesota que sigue el modelo cliente-servidor.

Se implementarán las siguientes servicios:

- *Disponer de información propia tal como carreras ofrecidas, materias, reglamentos, etc.*
- *Poder poner a disposición de cualquier usuario en particular, información tal como notas, estado de materias aprobadas, etc. de este usuario en particular.*
- *Poner a disposición del gopher bases de datos tales como ORACLE por medio del lenguaje ORAPERL.*
- *Dar acceso a distintos computadores por medio de protocolos tales como FTP y TELNET.*
- *Facilitar muchas labores administrativas tales como biblioteca, pedido de listados, manejo de ayudantías, etc.*

Esto permite a un usuario novato acceder a estos recursos de una manera transparente.

INDICE GENERAL

RESUMEN	VI
INDICE GENERAL	VII
INDICE DE FIGURAS	IX
INTRODUCCION	10
I. EL GOPHER	
1.1 <i>Introducción</i>	11
1.1.1 <i>Definición del Gopher de Internet</i>	11
1.2 <i>Protocolo Gopher</i>	13
1.2.1 <i>Modelo del Gopher de Internet</i>	13
1.2.2 <i>Acceso a servicios.</i>	14
1.2.3 <i>Adición Modular de servicios</i>	16
1.2.4 <i>Ventajas</i>	17
1.3 <i>Requerimientos necesarios</i>	20
1.3.1 <i>Gopher servidor Unix</i>	20
1.3.2 <i>Gopher cliente Unix</i>	21
II. ANALISIS DE LAS NECESIDADES DE LA ESPOL A SER SATISFECHAS POR EL SI-ESPOL.	
2.1 <i>Necesidades de la ESPOL.</i>	23
2.2 <i>SI-ESPOL en el desarrollo institucional.</i>	26
III. DISEÑO GENERAL DEL SI-ESPOL	
3.1 <i>Criterio seguido en el desarrollo de las aplicaciones.</i>	28
3.2 <i>Normas recomendadas.</i>	34
3.3 <i>Formatos generales.</i>	35
IV. DISEÑO DETALLADO DE LAS APLICACIONES SOPORTADAS.	
4.1 <i>Manejo de las ayudantías académicas.</i>	36
4.2 <i>Manejo de pedidos de listados de contabilidad a Cesercomp.</i>	46
4.3 <i>Sistema de notas.</i>	52
4.4 <i>Conexión de las aplicaciones de biblioteca a INTERNET</i>	58
4.5 <i>Directorio telefónico de la ESPOL.</i>	64
4.6 <i>Información acerca de profesores, alumnos y trabajadores.</i>	69

V. GATEWAYS UTILIZADOS	
5.1 <i>Instalación del gopher SQL.</i>	73
5.2 <i>Instalación del WAIS.</i>	80
5.3 <i>Instalación del CSO.</i>	84
VI. INSTALACION Y CONFIGURACION DE LOS DISTINTOS CUENTES.	
6.1 <i>Cliente para DOS.</i>	89
6.2 <i>Cliente para Windows.</i>	99
6.3 <i>Cliente para MAC.</i>	109
CONCLUSIONES Y RECOMENDACIONES	111
APENDICES	
<i>APENDICE A: ARCHIVOS DE CONFIGURACION Y COMPILACION.</i>	
<i>APENDICE B: NFS.</i>	
<i>APENDICE C: FORMS ORACLE.</i>	
<i>APENDICE D: MANUAL PARA EL USO DEL USUARIO FINAL.</i>	
<i>APENDICE E: MANUAL PARA ENTRENAMIENTO DEL ADMINISTRADOR DEL GOPHER</i>	
BIBLIOGRAFIA.	

INDICE DE FIGURAS

<i>FIG. 1</i>	<i>CONSULTA DE AYUDANTIAS DISPONIBLES</i>	<i>39</i>
<i>FIG. 2</i>	<i>FORMA DE SOLICITUD DE AYUDANTIA</i>	<i>40</i>
<i>FIG. 3</i>	<i>FORMA DE PEDIDOS DE LISTADOS A CESERCOMP</i>	<i>50</i>
<i>FIG. 4</i>	<i>FORMA DE CONSULTA DE NOTAS</i>	<i>56</i>
<i>FIG. 5</i>	<i>FORMA DE CONSULTA DE BIBLIOTECA</i>	<i>63</i>
<i>FIG. 6</i>	<i>FORMA DE CONSULTA DE TELEFONOS</i>	<i>67</i>
<i>FIG. 7</i>	<i>FORMA DE CONSULTA DE PERSONAS</i>	<i>71</i>
<i>FIG. 8</i>	<i>EJECUCION DEL PHONE.EXE</i>	<i>88</i>
<i>FIG. 9</i>	<i>CONSULTA DEL MINUET EN EL CORREO ELECTRO.</i>	<i>89</i>
<i>FIG. 10</i>	<i>CONSULTA DEL GOPHER</i>	<i>95</i>
<i>FIG. 11</i>	<i>SETEO DE LOS SERVIDORES DEL MINUET</i>	<i>96</i>
<i>FIG. 12</i>	<i>SETEO DE LA DIRECCION IP</i>	<i>96</i>
<i>FIG. 13</i>	<i>CONSULTA DE LA BIBLIOTECA</i>	<i>97</i>
<i>FIG. 14</i>	<i>CONEXION SLIP POR EL TRUMPET WINSOCK</i>	<i>98</i>
<i>FIG. 15</i>	<i>SETEO DEL TRUMPET WINSOCK</i>	<i>101</i>
<i>FIG. 16</i>	<i>GOPHER CLIENTE BAJO WINDOWS</i>	<i>102</i>
<i>FIG. 17</i>	<i>SETEO DEL GOPHER EN EL CLIENTE BAJO WINDOWS</i>	<i>103</i>
<i>FIG. 18</i>	<i>SETEO DEL PESO DE LOS LENGUAJES</i>	<i>104</i>
<i>FIG. 19</i>	<i>CONFIGURACION DE LOS SERVIDORES</i>	<i>105</i>
<i>FIG. 20</i>	<i>SETEO DE LAS VISTAS</i>	<i>106</i>
<i>FIG. 21</i>	<i>CONSULTA DE NOTAS POR MEDIO DEL GOPHER</i>	<i>107</i>

CA

INTRODUCCION

El desarrollo de un sistema de información para la ESPOL se debe a que en los últimos años con la interconexión de la ESPOL a INTERNET se ha podido palpar que para un usuario novato es un poco difícil encontrar cierta información requerida debido a que INTERNET es un conjunto de bases de datos muy extenso con muchos recursos por lo que encontrar y captar estos, resulta un poco dificultoso.

Lo que el sistema de información para la ESPOL permitirá a un usuario promedio, es acceder a estos recursos de una manera rápida y fácil. El SI-ESPOL combinará las ventajas de los boletines electrónicos y las bases de datos existentes en nuestro caso particular ORACLE, permitiendo mostrar información de una manera jerarquizada, a la vez que permitirá soportar distintos tipos de información tales como: textos, gráficos, etc que podrán ser accedidos presionando una simple tecla o un click de mouse.

Además de proveer de una opción más para la distribución de datos que se encuentran en el 43XX a bases de datos de más fácil manejo, siendo una alternativa económica y de rápida implementación.

Los pasos a seguir son los siguientes:

- Instalación del distinto software necesario para las necesidades de la ESPOL.*
- Pruebas externas de estas herramientas tales como PERL, ORAPERL, GOPHER SQL, WAIS, CSO, entre otros.*
- Diseño de los distintos menus y de la apropiada distribución de la información.*
- Recolección de los datos de cada unidad y ponerlo en el formato requerido.*
- Diseño de las bases en ORACLE para la migración de información.*
- Migrar datos del 43XX a la base de datos ORACLE.*
- Instalación de los distintos clientes tales como UNIX, MAC, WINDOWS.*

EL GOPHER

LI INTRODUCCION

LI1 DEFINICION DEL GOPHER DE INTERNET

Este documento describe el protocolo, lista algunas de las implementaciones actualmente disponibles, y podremos ver algunas formas de como implementar nuevas aplicaciones cliente y servidor. Esta adaptado del documento "Protocolo Del GOPHER de Internet" editado en 1991.

El protocolo GOPHER Internet y su software sigue un modelo cliente/servidor. Este protocolo es uno del conjunto de protocolos TCP/IP. Los servidores GOPHER deberán escuchar en el puerto 70 (El puerto 70 es asignado por IANA para el GOPHER de Internet). Documentos que describen el protocolo residen en muchos de los servidores autónomos en Internet. Los usuarios corren el software cliente en sus computadores de escritorio, conectándose a un servidor y mandando al servidor un selector (una línea de texto, la cual puede estar vacía) via una conexión TCP en un puerto predefinido. El servidor responde con un bloque de texto terminado por un punto en la línea y cierra la conexión.

Los documentos y servicios residen en muchos servidores. El software cliente presenta a los usuarios una jerarquía de ítems y directorios - como un sistema de archivos -. La interface con el usuario del GOPHER está diseñada para asemejarse a un sistema de archivos, porque es un buen modelo para la organización de documentos y servicios; el usuario tiene a su disposición un gran sistema de información en red que contiene principalmente ítems de documentos, directorios, etc.

Los servidores retornan tanto las listas de directorios o documentos. Cada ítem en un directorio está identificado por :

- Un tipo (La clase de objeto que es)*
- Una cadena selectora (típicamente contiene un pathname usado por el computador destino para localizar el objeto deseado).*
- El nombre visible al usuario (usado para navegar a través del gopher)*
- Un nombre de computador (El cual indica cual computador debo contactar para obtener ese ítem).*

- Y un número de puerto IP (El puerto al cual el servidor escucha para la conexión)

El usuario solamente ve el nombre visible para el usuario. El software cliente puede localizar y retirar cualquier item por el trio: selector, hostname, puerto.

Para usar un item de búsqueda el cliente presenta una búsqueda a un tipo especial de servidor GOPHER: UN SERVIDOR DE BUSQUEDA. En este caso el cliente manda la cadena selectora y la lista de palabras a ser buscadas. La respuesta produce una "lista de directorios virtuales" que contienen items encontrados por el criterio de búsqueda.

Los GOPHERS clientes existen en todas las plataformas populares. Porque el protocolo es sencillo y simple, hacer servidores y clientes es rápido y fácil. El protocolo del **GOPHER** de Internet esta diseñado esencialmente para actuar como un sistema distribuido de documentos.

Este sistema se asemeja a un sistema de archivos por las siguientes razones:

(a) Un arreglo jerárquico de información es familiar para muchos usuarios. Directorios jerárquicos que contienen items (como los documentos, servidores, y subdirectorios) son ampliamente usados en boletines electrónicos y otros sistemas de información (campus-wide). Y además la gente que accesa a un servidor de información de un campus espera algún tipo de organización jerarquizada para la información presentada.

(b) Un sistema de archivos (file system) jerarquizado puede ser expresado en una sintaxis simple. La sintaxis usada para el protocolo del GOPHER de Internet es fácilmente entendible, y fue diseñado para hacer servidores y clientes fáciles. Se podrá usar Telnet para simular un requerimiento de un cliente del GOPHER y observará la respuesta del servidor. Herramientas de software especiales no son necesarias para esto pues la actividad común del usuario es hojear através de los directorios jerarquizados.

(c) Como el GOPHER fue diseñado para la Universidad, uno de las metas fue que los departamentos tengan la opción de publicar información desde sus pequeñas máquinas de escritorio, y a la vez que mucha de su información pueda ser publicada como simples archivos de texto organizados en directorios. Un protocolo modelado como un sistema de archivos tiene una utilidad inmediata, porque puede haber un mapeo directo del sistema de archivos en la máquina del usuario para los datos publicados via GOPHER.

(4) Una metáfora del sistema de archivos es extensible, dando el tipo de atributo de los items en el pseudo sistema de archivos, es posible poner otro tipo de items además del los documentos de texto. Complejos servicios de bases de datos pueden ser manejados por un tipo separado de items. Pero los sistemas de archivos no incluyen la búsqueda o criterios de búsqueda en la bases de datos para el acceso a documentos. Un servidor de búsqueda es también definido en este pseudo sistema de archivos. Cada servidor retorna "Directorios Virtuales" o lista de documentos con los resultados de los criterios de búsqueda.

1.2 PROTOCOLO GOPHER

1.2.1 EL MODELO GOPHER INTERNET

Una detallada interpretación de la sintaxis del GOPHER de Internet está disponible más adelante. Pero una lectura profunda no es necesaria para aprender el protocolo GOPHER de Internet.

En esencia, El Protocolo GOPHER consiste en un cliente conectándose a un servidor y mandando al servidor un selector (Una línea de texto, la cual puede estar vacía) via una conexión TCP. El server responde con un bloque de texto terminado con un punto en una línea, y cierra la conexión. Ningún estado es retenido por el servidor entre transacciones con el cliente. La natural simplicidad del protocolo viene de la necesidad de implementar rápidamente y eficientemente servidores y clientes para lentos y pequeñas computadoras (1 MB).

A continuación tendremos un ejemplo sencillo de una interacción cliente/servidor; interacciones más complicadas serán vistas más adelante. Asuma que un "bien conocido" servidor GOPHER escucha en un puerto para el campus. La única información sobre la configuración que el cliente mantiene es el nombre del servidor y el número del puerto (en este ejemplo tendremos `espol.edu.ec` y el puerto 70). En el ejemplo de abajo la *F* denota el caracter TAB.

CLIENTE : {Abre conexión para `espol.edu.ec` en el puerto 70}

SERVIDOR : {Acepta la conexión pero no dice nada}

CLIENTE : {Manda una línea vacía: Significando "liste que es lo tiene"}

SERVIDOR: {Manda una serie de líneas, cada una terminandado con CR LF}

!cerca del GOPHER de Internet!Stuff:About us!espol.edu.ec!70

!cerca de la ESPOLFZ,5692,AUMF!valdivia.espol.edu.ec!70

*Noticias de Computación*Fhuancavilca.espol.edu.ecF70

*Cursos, Calendarios*FFmatematicas.espo.edu.ecF70

*Departamento de Publicaciones*FStuff:DP:publicaciones.espol.edu.ecF70

{.....etc.....}

{Punto en una línea }

{Servidor cierra la conexión}

El primer caracter en cada línea dice cuando una línea describe un documento, directorio, o un servicio de búsqueda (Caracter '0','1','7';)

Hay grupo de caracteres usados que se los verá más adelante. Los siguientes Caracteres forman lo que será mostrado al usuario.

Los caracteres que siguen al TAB, forman la cadena selectora que el software cliente debe mandar; este nunca debería ser modificado por el gopher cliente. En la práctica, la cadena selectora es a menudo un pathname o otro selector de archivo usado por el servidor para localizar el ítem deseado. Los próximos dos delimitadores TAB, limitan los campos que denotan el nombre del host que tiene este documento (o directorio), y el puerto en el cual está conectado. Si hay todavía otro delimitador TAB, el cliente GOPHER debería ignorar estos. Un CR CF denota el final de el ítem.

En el ejemplo la línea 1 describe un documento que el usuario verá como "Acerca del GOPHER de Internet". Para obtener este documento, el software cliente deberá mandar la cadena selectora: "Stuff:About us" para el computador espol.edu.ec en el puerto 70. Si el cliente hace esto el servidor responde con el contenido de el documento, terminado por un punto en la línea. El cliente trata de presentar al usuario una lista de ítems como lo que sigue:

- 1- Acerca del GOPHER de Internet
- 2- Acerca de la Espol
- 3- Noticias de Computación
- 4- Cursos, Calendarios
- 5- Departamento de publicaciones

3.2.2 ACCESOS A SERVICIOS

Documentos (otros servicios que pueden ser vistos como documentos, ejemplo una guía de estudiantes profesores) son enlazados con las máquinas que dice su trió selector: string selector, nombre de la máquina, y el puerto IP. Se asume que habrá un servidor gopher principal -raíz- para la institución o el campus. La información

en este servidor gopher puede ser duplicada por uno o muchos más servidores para evitar un punto de falla y para repartir la carga por varios servidores. Los Departamentos que desean poner sus propios servidores departamentales necesitan registrar el nombre de la máquina y puerto con el administrador del servidor principal gopher, de la misma manera como se registra el nombre de una máquina con el domain-name server.

Un enlace el cual apunta al servidor departamental puede entonces ser hecha en el servidor gopher raíz. Esto asegura que usuarios puedan ser capaces de navegar en un filesystem jerárquico y virtual con una raíz conocida(servidor principal).

Note que no hay requerimientos entre una servidor secundario departamental y el servidor principal (raíz); ellos solamente ubican un enlace de un servidor gopher secundario al servidor principal departamental. Ellos pueden efectivamente ubicar los enlaces a cualquier servidor a que ellos decidan. Los enlaces también pueden apuntar de regreso al servidor principal. El filesystem (networked) virtual es estructura arbitraria y no necesariamente un árbol con raíces.

El modo superior es simplemente conveniente, por ser un punto de entrada. Un conjunto de servidores gopher enlazados de esta manera puede funcionar con un sistema de información (campus-wide).

Servidores pueden apuntar a otros que no sean servidores secundarios. En efecto los servidores pueden apuntar a otros servidores ofreciendo servicios muy útiles en cualquier parte de Internet.

1.2.2.1 PORTABILIDAD DE LOS SERVIDORES

Es recomendado que todo los servidores tengan nombres(alias) que sean usados por los gophers clientes para localizarlos. Enlaces a estos servidores deberían usar estos nombres (alias) en vez de los nombres primarios. Si la información necesita ser movida de una máquina a otra , un simple cambio del domain name system alias permite que esto ocurra sin ninguna reconfiguración de los clientes. No hay nada que prevenir en los servidores (gopher) secundarios o servicios que corren en otros servidores o otros puertos distinto que el 70.

1.2.2.2 CONTACTANDO ADMINISTRADORES DE SERVIDORES

Es recomendado que todo administrador de servidor tenga un documento llamado: "Alcance del Gopher de la Universidad" como el primer ítem de su menú principal. En este documento debería tener una descripción corta de que el servidor que mantiene, el porqué de su nombre, la dirección, el teléfono, y el e-mail de la persona que administra el servidor. Este provee una vía a los usuarios para contactar con el

administrador para corregir una información que se encuentre errada en el gopher o una opción que no este corriendo correctamente. Esto es también recomendado que los administradores pongan la fecha de la última actualización en los archivos.

12.3 LA ADICION MODULAR DE SERVICIOS

El primer caracter de cada linea en el directorio servidor de datos indica si el item es un archivo (caracter '0'), un directorio (caracter '1'), o una búsqueda (caracter '7'). Este es el conjunto base de items en el protocolo Gopher.

Es deseable para los gopher clientes el ser capaces de usar servicios diferentes y hablar distintos protocolos (simples como finger, otros como CSO servicios de guía, vmail, o X.500). Por ejemplo si un directorio servidor de datos escucha marcas de un item determinado ,por ejemplo '2', entonces esto significa que al usar este item , el cliente debería hablar el protocolo CSO.

Esto elimina la necesidad de ser capaces de anticipar todas las necesidades futuras y decirlos fijas en el protocolo Gopher; esto mantiene el protocolo extremadamente simple.

En contraposición a esta simplicidad, el esquema tiene la capacidad de expandirse y cambiar con el tiempo, arbitrariamente, añadiendo simplemente un caracter para un item nuevo esto trae la proliferación de servicios.

12.3.1 CONSTRUYENDO CLIENTES

Un cliente simplemente manda el string de pedido a un servidor, si este quiere retirar un documento o ver el contenido de un directorio. Por supuesto, cada host tiene sus apuntes a otros hosts, resultando en un "grafo" de hosts. El cliente software puede almacenar (o apilar) las localizaciones que este ha visitado en en la búsqueda de un documento. El usuario puede por lo tanto salir de la localizacion actual almacenando la anterior en la pila. Alternativamente, un cliente con una capacidad de múltiples ventanas debería ser capaz de mostrar más de un directorio o documento al mismo tiempo.

Un cliente inteligente podría almacenar el contenido de los directorios visitados, así evitando transacciones de la red, si la información ha sido previamente "cargada".

Si un cliente no entiende que es un item B, entonces puede simplemente ignorar el item en el directorio visitado; el usuario nunca verá este item, alternativamente el item podría ser visto como un tipo desconocido.

Los servidores principales para un campus son posiblemente los que cogan más tráfico que los servidores secundarios, y por esto es evidente que los servidores primarios no puedan tolerar ser dejados fuera de servicio por periodos muy largos. Esto hace más perceptible lo importante que es hacer que los servidores puedan conectarse primero a un servidor primario, moviéndose a otro si el principal falla (balanceando la carga).

1.2.3.2 CONSTRUYENDO SERVIDORES ORDINARIOS GOPHER

El string de recuperación (string selector) mandado al server podría tener un path a un archivo o directorio. Este podría ser el nombre de un script, una aplicación o ejecutar una búsqueda que genere un documento o un directorio. El servidor básico usa el string selector que llega, pero no incluye un CR-LF o TAB, si estos llegan primero.

Toda la inteligencia es manejada por el servidor gopher en lugar del protocolo. La implementación del protocolo puede desarrollarse como las necesidades los dicten y como el tiempo lo permita.

1.2.3.4 SERVIDORES DE PROPOSITOS GENERALES

Hay dos tipos especiales de servidores (más allá del servidor normal Gopher) que analizaremos a continuación.

1.- El directorio de un servidor gopher puede apuntar a un CSO nameserver (El servidor -gopher- retorna un tipo de caracter de '2') para permitir que haya un servicio de búsqueda de estudiantes y profesores. Si el item es seleccionado, el software cliente debe recurrir a un CSO nameserver cuando este se conecta al host apropiado.

2.- Un servidor gopher puede también apuntar a un servidor de búsqueda (retorna el primer caracter de 7). Los servidores más comunes ofrecen una búsqueda de índice de texto completo en el contenido de textos de documentos. Cada servidor "búsqueda

de texto completo" responde al cliente con una lista de todos los documentos conteniendo una o más palabras (El criterio de búsqueda). El cliente manda al servidor el string selector, un tab, y la búsqueda del string. Los espacios entre palabras son usualmente llenados con operadores AND.

La existencia del servidor CSO es por razones históricas: cuando se estaba diseñando, la guía de teléfonos del campus de la Universidad de Minnesota estaba usando el protocolo CSO y este mostraba simplicidad al equipo de desarrollo para usarlo.

8.2.3.5 CONSTRUYENDO SERVIDORES DE BUSQUEDA DE TEXTO COMPLETO

Un servidor de búsqueda de texto completo es un servidor de propósito general que conoce acerca de los procedimientos del gopher para el captar documentos. Estos servidores mantienen un índice de texto completo del contenido de los documentos de texto plano en servidores gopher en algún dominio especificado. Un servidor gopher de búsqueda de texto completo fue hecho utilizando estaciones Next por la facilidad para hacer máquinas de búsqueda. Un servidor de búsqueda para los sistemas Unix basado en la máquina de búsqueda WAIS. La cual esta disponible y corre como un software aparte que se lo adquiere por ftp en huembox@micro.umn.edu.

8.2.3.6 TIPOS DE ITEMS

El software cliente decide que items son disponibles y entre aquellos items que son representados por caracteres tenemos los siguientes:

0Item es un archivo

1Item es un directorio.

2Item es un servidor CSO.

3Error.

4Item es un archivo binario Macintosh.

5Item es un archivo binario DOS.

6El cliente debería leerlo hasta que la conexión TCP se cierre.

7Item es un archivo Unimix uuencoded.

7 Item es un servidor de búsqueda de texto completo.

8 Item apunta a una sesión telnet.

9 Item es un archivo binario.

Cliente debería leer hasta que la conexión TCP se cierre.

+ Item es un servidor redundante.

T Item apunta a una sesión texto tn3270.

g Item es un archivo gráfico del formato GIF.

I Item es alguna clase de archivo imagen. El cliente decide mostrarla.

Los caracteres O hasta Z son reservados. experimentos locales deberían usar otros caracteres. Las extensiones de la máquina especificada (usada) no son recomendables. Note que los tipos 5 hasta el tipo 9 el cliente debería ser preparado para que lea hasta que la conexión se cierre. No deberá haber punto al final del archivo; el contenido de estos archivos son binarios y el cliente deberá decidir que hacer con ellos basándose quizá en la extensión .xxx .

1.2.4 VENTAJAS

De la mejor manera se hará que los nuevos elementos en el gopher tales como nuevos protocolos sean manejados con nuevos tipos (Items). La filosofía del Gopher de internet es:

(a) Inteligencia es mantenida por el servidor. Los gopher clientes tienen la opción de ser capaces de acceder a nuevos tipos de documentos simplemente reconociendo el tipo de item. Cualquier otra "inteligencia" a ser soportada por el protocolo deberá ser mínima.

(b) El servidor deberá de mandar "texto". La respuesta del servidor gopher no debería tener caracteres especiales como TABS , LF. Los publicadores de documentos deberían tener filtros para eliminar todos estos caracteres especiales de los documentos que ellos deseen publicar.

El cliente debería hacer algo razonable con los caracteres especiales recibidos en el texto; como filtrarlos y eliminarlos.

1.3 REQUERIMIENTOS NECESARIOS

1.3.1 GOPHER SERVIDOR UNIX

En este punto veremos los requerimientos necesarios así como los parámetros necesarios para el funcionamiento del servidor UNIX.

Entre los puntos principales necesitamos una máquina Unix con una conexión TCP/IP para usar el servidor GOPHER.

También debemos tener una máquina que necesite una conexión a una máquina que esté corriendo una DOMAIN NAME SERVER(DNS).

A la vez que necesitamos el programa gopherd que es el resultado de la compilación de los distintos programas del paquete de la Universidad de Minnesota. El servidor gopher (gopherd) es un programa que acepta requerimientos de los gopher clientes y proporciona la información a ellos. Este utiliza un número de métodos para suministrar esta información. Mucha de esta información proviene de el sistema de archivos específicamente del directorio gopher-data.

Gopherd puede ser también un gateway para otros servicios de red como FTP y WAIS. El gopher cliente no necesita entender estos protocolos, el servidor gopher hace esto por ellos.

La primera cosa que se debería hacer es crear el directorio gopher-data. Este directorio contendrá toda la información que el gopher cliente observará.

El servidor gopher tomará esta información de los archivos y directorios en el gopher-data. Haciendo cambios en el árbol de directorios, se verán cambios para lo que el cliente verá.

Las transacciones que soportará son las siguientes.

*Los directorios dentro del directorio gopher-data son manejados como Directorios gopher.

*Los archivos de texto son representados como items textos.

*Archivos GIF son representados como archivos imagen GIF.

*Todos los archivos y directorios que comienzan con un punto (ejemplo .foo) o son de nombres etc,usr,bin,dev, o core son ignorados por el servidor.

*Los títulos de cada objeto presentado al cliente es el nombre del archivo que posee este objeto.

*Los archivos comprimidos con "compress" y con "gzip" son soportados. Ellos serán mandados a los clientes descomprimidos.

*Los Scripts ejecutables son representados como items de textos. La salida del script son mandados cuando los requerimientos del cliente necesita ver el item. El shell script debiera comenzar con los tres caracteres #!/.

*Archivos que terminan con .src son relacionados con archivos WAIS generados por el programa waisindex.

COMANDOS EN LINEA

Hay dos maneras de setear el gopherd el primero es editando el archivo gopherd.conf pero algunos parámetros se los pueden setear desde la linea de comandos a continuación se presentarán una lista de ellos.

```
gopherd [-C] [-o options] [-L load] [-l logfile] [-u user] gopher-data gopher-port
```

Options:

-L restringe la carga de conexiones al gopher server pero para esto el gopherd debe ser compilado con la opción LOADRESTRICT.

-C deshabilita el almacenamiento de los directorios recuperados.

-l logfile es para cada conexión tenga un registro para saber a que hora se conectó y que transacción que hizo.

-u es especificada cuando se corre el server desde el inetd.

-o especifica una alternativa para el archivo "gopherd.conf".

-C no usa las llamadas al sistema chroot la cual puede asegurar que solamente los archivos que son públicos pueden ser leídos desde el gopher

-u user corre como el usuario llamado user. El servidor esta corriendo con permisos reducidos (otros que el root) .

3.2 GOPHER CLIENTE UNIX

El gopher de Internet para Unix esta en código de lenguaje C. Se necesita compilar este código para convertirlo en ejecutables para el sistema.

Los clientes al acceder a los servidores los distintos tipos de items son distinguidos con caracteres especiales tales como un '/' para los directorios un ':' para los archivos de texto un '?' para distinguir los items de búsqueda.

A continuación se verá como el cliente puede acceder a servicios distintos de archivos y directorios:

Índices de Búsqueda

Items con <?> en el final del nombre son índices de búsqueda. Cuando se selecciona uno de estos items usted podrá ingresar un patrón a buscar. Estas palabras determinaran el los items que serán mostrados como un menú y su situación.

Servidores de Directorio Telefónico.

Estos items se distinguen de los demás porque tienen un simbolo <CSO> al final del nombre.

Al escoger este item se mostrará una forma que será llenada con los datos que se desee con los cuales se hará una búsqueda en los archivos y se mostrarán las personas que cumplan con estos datos.

2 ANALISIS DE LAS NECESIDADES DE LA ESPOL A SER SATISFECHAS POR SI-ESPOL

2.1 NECESIDADES DE LA ESPOL

En la actualidad con el desarrollo tan rápido de la computación y con una demanda de información que crece de una manera vertiginosa, la ESPOL se ve la necesidad de diseñar un sistema de información que sea de rápido acceso y que maneje volúmenes de información muy grandes y de tiempo de respuesta aceptable.

Por ser la ESPOL una institución que es una fuente de conocimientos y recursos humanos para el desarrollo nacional y regional se tiene que mostrar esta información de una manera organizada (jerarquizada). Otra cualidad que debería cumplir este sistema de información es que tiene que ser barato y que se adapte a los cambios que están por venir ya que en el mundo de los computadores no se pueden producir los cambios en 3 años.

Además que debe ser simple tanto en la forma de trabajar, como en la de organizar los datos.

La ESPOL cuenta además con una variedad de plataformas (Unix, DOS, MAC, etc) por lo tanto este SI debe de poder correr en cualquiera de esta plataformas y en las que estén por venir. Otra característica importante será la que la conexión sea barata y que muchos otros sistemas de información propios de cada facultad e instituto se puedan conectar a este, debido que a que en corto plazo en la universidad se proyecta un plan de interconectar muchas de las redes actualmente separadas.

Otra característica que este sistema de Información deberá cumplir es que no se quede estancado, sino que progrese día a día por lo que necesitará un especial interés de parte tanto administrativa como docente de la Espol, para darle impulso a este proyecto.

Una de las ventajas de un sistema de estas características es que se puede considerar como una parte para el desarrollo de este proyecto a los estudiantes.

Todas estas características mencionadas anteriormente se las pueden ver en el GOPHER que es la herramienta descrita en el capítulo primero. Por su costo nulo y por el gran éxito que ha tenido en Internet especialmente en Universidades que son centros de información, según investigaciones realizadas de un puesto 110 en el año 1991 ha pasado a un respetable octavo de las aplicaciones más utilizadas en

Internet. Actualmente se han y se están desarrollando software muy respetable en las distintas universidades -especialmente Estados Unidos y Europa- que tienden a solucionar problemas muy comunes en estos centros de estudios, además de que se están creando ambientes de trabajo para estudiantes.

Talvez en los actuales momentos no se tenga los recursos necesarios para tener lo último en tecnología pero si se tiene el recurso humano. Un plan estratégico que incluya a los estudiantes para que investigen en estas "Redes Internacionales" y a la vez de dar incentivos a estos estudiantes, servirán para que se impulse el diseño y el desarrollo de nuevas tendencias y tecnologías.

Entre todas estas necesidades tendremos que clasificarlas en dos grandes grupos.

-Necesidades Administrativas

-Necesidades Docentes

NECESIDADES ADMINISTRATIVAS

Entre las necesidades administrativas podemos mencionar que en la actualidad con sistemas hechos en un IBM 43XX , muchos de ellos hechos en cobol, nos muestran que aparte del enorme esfuerzo necesario para mantener estos sistemas, es muy grande el costo operativo de estos computadores especialmente para la ESPOL que no tiene recursos financieros muy grandes para la mantención de estos computadores. Aparte que el daño en algún dispositivo periférico en estos computadores son considerados en la actualidad como puntos críticos. Con el SI-ESPOL se propone al principio una migración muy pequeña de datos para que puedan ser evaluada y analizada el costo de esta migración y la facilidad que puede brindarnos la base de datos empleada, en este caso ORACLE. además de contar con lenguaje de programación en Unix que actualmente se esta poniendo a nivel mundial en un nivel muy alto como es el PERL que nos presenta muchas facilidades sobre los comunmente conocidos C-SHELL,BOURNE-SHELL,K-SHELL.

Además de tener acceso a bases de datos por medio de un lenguaje muy parecido al PERL ,ORAPERL , se nos presenta una serie de facilidades para el manejo de la base de datos Oracle de la ESPOL en la cual podemos poner información y manejarla de una manera muy consistente y confiable.

Todo este ambiente creado para el desarrollo del SI-ESPOL nos servirá como una opción más para el manejo de datos administrativos de la ESPOL dándonos más flexibilidad para la adaptación de las distintas plataformas.

Una pequeña parte para el manejo administrativo sería disminuir el flujo de papeles entre contabilidad y la parte operativa de cesercomp por medio de mail aunque lo óptimo sería que se pase la parte contable del sistema antiguo (IBM 43XX) a un sistema más flexible y que brinde más flexibilidad en el manejo de la información y que permita emitir informes a los propios usuarios.

De comprobarse la facilidad de migración entre el sistema 43XX a la base de datos utilizada para el manejo de la información en el gopher. Esta sería una solución informática mucho más barata y segura para el manejo de la información en la ESPOL.

NECESIDADES DOCENTES.

Talvez esta sea una de las partes más delicadas debido a que la ESPOL es por si un ente académico y en los últimos años debido en gran parte a la situación económica en que hemos vivido, la parte práctica de los cursos dados en la ESPOL corren en la mayoría de los casos en manos de los estudiantes y por esta circunstancia el estudiante no tiene un ambiente 'especializado' para su trabajo.

Como se dijo anteriormente por medio del sistema de información que se propone, El estudiante entraria a un mundo de información estructurada y organizada y con lo último en Desarrollo de software y teorías en el area que ellos deseen ya sea en Mecánica, Electrónica, Computación.

Desarrollando e investigando con estos sistemas de información que se pueden incrementar de manera rápida y a la vez que se pueden modularizar los gopher para que cada facultad administre su propia información y recursos.

La adaptación a otros sistemas que se estan desarrollando como WWW (hipertexto) entre otros es muy fácil, y aun más la base de información mantenida por el gopher puede ser utilizada por estos clientes.

2.2 SI-ESPOL EN EL DESARROLLO INSTITUCIONAL.

Como se conoce la ESPOL se encuentra en una era de cambios motivados unos por el cambio significativo en las teorías y métodos que experimenta el mundo y otros motivados por la situación económica en que vive.

Entre los cambios significativos que se presentan podemos encontrar el área de computación. Por el excesivo precio de los componentes y reparaciones de los sistemas actuales la ESPOL se ve en la necesidad de una migración que sea rápida y planificada. Esto a su vez genera muchas alternativas para este cambio por lo que el planificar estos cambios con mucha profundidad nos ayudará a eliminar la incertidumbre y mantener el control de estos proyectos.

Como se mencionó anteriormente el hecho de migrar los datos del IBM 43XX a una base de datos como lo es ORACLE nos ayudará a comprobar y tomar una decisión sobre esta migración.

Lo incierto del futuro y el cambio hacen de la planeación una necesidad. Así como el navegante no puede simplemente fijar una ruta y olvidarse de ella, la ESPOL tampoco puede establecer una meta y dejar las cosas así. Raras veces hay seguridad del futuro, y ella es tanto menor cuanto más lejos en el futuro haya que considerar las consecuencias de una decisión.

Tal vez la falla de un disco de un computador pueda afectar la situación presente pero es casi improbable que cambie las situaciones a corto plazo.

Sin embargo cuando se planifica a largo plazo, disminuye la certeza con respecto al ambiente interno y externo y se vuelve más incierta cualquier decisión. Cuando las tendencias no son fácilmente discernibles, la buena planeación puede volverse más difícil.

A causa de que toda la planeación se orienta hacia la consecución de los objetivos de la ESPOL, el simple acto de planear llama la atención sobre los objetivos. Los planes globales bien estudiados unifican las actividades interdepartamentales.

La mayoría de las veces los planes a corto plazo suelen hacerse sin referencia a los planes a largo plazo. Esto es a todas luces un grave error. Nunca se insistirá lo bastante en la importancia de integrar los dos y no se debería hacer ningún plan a corto plazo a menos que contribuyera a la realización del plan pertinente a largo plazo. Muchas de las pérdidas de la planeación surgen de decisiones sobre situaciones inmediatas que se hacen sin considerar su efecto sobre objetivos más remotos.

Algunas veces estas decisiones a corto plazo no solo no contribuyen al plan de largo plazo sino que con frecuencia impiden o provocan cambios en el plan a largo plazo.

Si los planes pueden cambiarse para afrontar situaciones que, o no fueron, o no podían ser previstas, el periodo de planeación puede ser más corto de lo que sería necesario de otra manera. A causa de la incertidumbre del futuro y el posible error aun en las más expertas predicciones, el ideal de la planeación es ser flexible. La habilidad de cambiar de dirección cuando por hechos inesperados haya que hacerlo sin costo demasiado elevado. Hay dos principios que se aplican para efectuar cambios en la dirección de la planeación, el principio de la flexibilidad y el principio del cambio de rumbo.

EL PRINCIPIO DE FLEXIBILIDAD

Cuanto mayor sea la flexibilidad que se pueda imprimir a los planes, menor será el peligro de pérdidas en las cuales se incurre por sucesos inesperados; pero el costo de la flexibilidad debe pesarse ante los riesgos que implican los compromisos futuros que han contraído.

EL PRINCIPIO DE CAMBIO DE RUMBO

Cuanto más se comprometen con el futuro las decisiones de la planeación, tanto más importante es para el encargado de la planeación revisar periódicamente los acontecimientos y las expectativas y rediseñar los planes en la forma necesaria para mantener el curso hacia las metas deseadas.

El trazado de plan para el cambio en la institución debe involucrar en el desarrollo de ella a los estudiantes como entes productivos, ya que este es el mayor recurso con que cuenta la ESPOL sobre cualquier otra institución o universidad ya que desde el inicio se hace una selección para su ingreso y por lo tanto cuenta con personas de gran capacidad. Y la flexibilidad que se aplique en este plan será importante por los constantes cambios a que se encuentra el mundo de la computación.

Además otra parte importante será el control que se haga sobre este plan para que se cumpla a cabalidad.

"PLANEAR es mirar adelante y controlar es mirar atrás"

Harold Koontz.

Algunas veces estas decisiones a corto plazo no solo no contribuyen al plan de largo plazo sino que con frecuencia impiden o provocan cambios en el plan a largo plazo.

Si los planes pueden cambiarse para afrontar situaciones que, o no fueron, o no podían ser previstas, el periodo de planeación puede ser más corto de lo que sería necesario de otra manera. A causa de la incertidumbre del futuro y el posible error aun en las más expertas predicciones, el ideal de la planeación es ser flexible. La habilidad de cambiar de dirección cuando por hechos inesperados haya que hacerlo sin costo demasiado elevado. Hay dos principios que se aplican para efectuar cambios en la dirección de la planeación, el principio de la flexibilidad y el principio del cambio de rumbo.

EL PRINCIPIO DE FLEXIBILIDAD

Cuanto mayor sea la flexibilidad que se pueda imprimir a los planes, menor será el peligro de pérdidas en las cuales se incurre por sucesos inesperados; pero el costo de la flexibilidad debe pesarse ante los riesgos que implican los compromisos futuros que han contraído.

EL PRINCIPIO DE CAMBIO DE RUMBO

Cuanto más se comprometen con el futuro las decisiones de la planeación, tanto más importante es para el encargado de la planeación revisar periódicamente los acontecimientos y las expectativas y rediseñar los planes en la forma necesaria para mantener el curso hacia las metas deseadas.

El trazado de plan para el cambio en la institución debe involucrar en el desarrollo de ella a los estudiantes como entes productivos, ya que este es el mayor recurso con que cuenta la ESPOL sobre cualquier otra institución o universidad ya que desde el inicio se hace una selección para su ingreso y por lo tanto cuenta con personas de gran capacidad. Y la flexibilidad que se aplique en este plan será importante por los constantes cambios a que se encuentra el mundo de la computación.

Además otra parte importante será el control que se haga sobre este plan para que se cumpla a cabalidad.

"PLANEAR es mirar adelante y controlar es mirar atrás"

Harold Koontz.

3.1 CRITERIO SEGUIDO EN EL DESARROLLO DE LAS APLICACIONES.

Los Criterios seguidos en el desarrollo de las aplicaciones se basan en criterios referentes a sistemas de información.

A continuación se harán ciertas definiciones referentes a sistemas de información:

SISTEMA: conjunto de elementos interrelacionados que tienen un fin común.

DATO: representación de un hecho, un objeto, etc.

El dato debe ser representado de alguna manera para ser entendido.

$$\boxed{\text{DATO}} + \boxed{\text{SIGNIFICADO}} = \boxed{\text{INFORMACION}}$$

La información es diferente dependiendo del significado o interpretación que se le dio al dato.

INFORMACION: Es el hecho de darle significado a un dato, este significado se aplica a través de los programas en términos de computación. En otro medio el significado está dado por los procedimientos.

Los datos deben estar en alguna parte para ser procesados, en un sistema de información se encuentran en la base de datos.

BASE DE DATOS: Es un conjunto de datos que es la base sobre la cual trabaja el sistema de información. Se debe tener alguna forma de manipular estos datos, esto es a través de software o cierta inteligencia que es el sistema manejador de Base de datos, esto es necesario para todo sistema de información. Se toma el dato se le aplica el significado y se le proporciona información.

Hablando apropiadamente un SISTEMA DE INFORMACION es todo sistema de procesamiento de datos ya sea manual o automatizado para apoyar funciones de administración, toma de decisiones y operación de un ente.

Si no existiese el recurso humano no habrían sistemas de información, el sistema de información debe tener sus elementos bien interrelacionados para obtener un fin.

PROCESO QUE SE EFECTUA CON LA INFORMACION

Entre los procesos que afectan a la información podemos diferenciar como sigue:

- 1.- Selección. Seleccionar aquellas informaciones que van a ser dirigidas a los usuarios. Se aplican las reglas definidas por el propio usuario, las que son útiles o no.
- 2.- Captación de Información. registrar lo que interesa, aquello que hemos definido como imprescindible o útil al sistema debe ser captado. El ser humano lo hace a través de los sentidos.

La información se capta de la siguiente manera:

- a.- Captación de nueva información: crear un concepto dentro del sistema, definir, dar forma a la información.
- b.- Cambio de soporte: Todo hecho debe ser factible de registrarse en un medio adecuado. La información debe estar en un medio para poder ser trasladada

Soporte: medio donde se encuentra la información.

ejemplo: Un profesor al calificar un examen lo hace en le papel, genera información al poner la nota, luego lleva esto a n medio magnético (cambio de soporte). Cuando el dato registrado no cumple con las reglas para ser interpretado no se encuentra en un medio adecuado. La información se genera una sola vez, lo que existe es un cambio continuo de soporte.

- 3.- Transmisión y almacenamiento de información.- La informaciones deben ser llevadas de un sitio al otro a través de canales (formales o informales).
- 4.- Transformación de la información: Se dispone de condiciones, que deben cumplirse y fórmulas o algoritmos. Las condiciones son requisitos para formular datos.

Segun el autor Shannong se presentan 3 problemas al comunicar información:

- 1.- Problemas a nivel técnico: implica precisión de la transmisión de un conjunto de símbolos desde el emisor al receptor.
- 2.- Problemas a nivel semántico: se refiere a la precisión en la que el receptor entiende el mensaje que ha mandado el emisor.

3.- *Problema a nivel de eficacia: esta involucrado en el éxito de la comunicación que esta medida en términos que a producido la acción deseada o el comportamiento deseado en el receptor.*

Un sistema de COMUNICACION esta compuesto de 4 elementos.

- 1.-Emisor
- 2.-Un canal
- 3.-Un mensaje
- 4.-Un receptor

FUENTES DE INFORMACION:

Las fuentes de información las podemos clasificar en Primarias y Secundarias:

PRIMARIA: No existe en ningun lugar accesible o conocido entonces debemos buscarlo directamente aquí existen 2 métodos:

-Observación: Se lo considera muy útil en cuanto a tener respuestas parciales a problemas particulares, observando acontecimientos relacionados con el caso.

-Experimentación: La persona que realiza el experimento tiene control sobre la fuente de las siguientes maneras:

- Definiendo el entorno
- Manipulando variables
- Encuestas: puede contar con un gran numero de fuentes de información implica una planeación extensa.
- Estimación subjetiva consiste en tener información de expertos.

SECUNDARIA: Nos va ha evitar tiempo y dinero , esta lo constituye información que esta almacenada en sitio accesible.

PROBLEMAS DE LA FUENTES DE INFORMACION

-IMPARCIALIDAD: La información no debe reflejar perjuicio alguno no debe contener desviaciones intencionales o puntos de vista que distorcionan la realidad. Quien puede alterar es la persona que recopila o procesa los datos sobre los cuales se basa la información. (0,0)

-**VALIDEZ:** esta determinada si es significativa y relevante para un objetivo propuesto. Para determinar esto podemos preguntarnos si responde a la pregunta planteada.

La información pierde validez si no lo utilizamos para lo que fue reunida o formulada.

-**CONFIABILIDAD:** Se refiere a la fidelidad de la imagen que la información intenta transmitir.

-**CONSISTENCIA:** Debe basarse en datos homogéneos, quiere decir que tipo y el número de unidades de información deben ser los mismos durante todo el proceso de investigación.

-**ANTIGUEDAD:** Es uno de los factores que influye mucho el valor dependiendo el momento que llega hay 1 regla : mientras haya mayor antigüedad en la información es mas cuestionable para el administrador.

-**RETRASO:** se alude como cualquier actividad o bien falta de actividad y que interpone tiempo entre el reconocimiento por parte del usuario de una necesidad de información y la recepción de esta. El retraso ocurre por el tiempo que requerimos para reunir y procesar los datos o lograr el acceso a la información.

(* Osorio)

CARACTERISTICAS DEL SISTEMA DE INFORMACION

-**Un standard de rendimiento:** es lo que se espera que de el sistema, lo que se espera es información expresada en términos de requerimientos del usuario.

-**Método de Medición:** En todo sistema de información debe existir alguna forma de medir el rendimiento.

-**Forma de comparar:** Lo producido con lo que se esperaba.

-**Retroalimentación:** Todo sistema debe tener una forma de retroalimentación , el medio externo cambia, lo que obliga al sistema a cambiar. Esta retroalimentación se produce por la comparación de lo producido en lo requerido.

Entre mayor interacción con el exterior mayor retroalimentación controles y métodos de corrección deben existir cuando se diseñan los sistemas al reducir la interferencia exterior se reduce lo que hay que corregir.



NIVELES DE INFORMACION

Entre los niveles de información tenemos los siguientes

1.- Información internacional: información que no solo interesa al medio local sino también a todas las partes.

2.- Información nacional: información que interesa a un determinado lugar.

3.- Información corporativa: interesa a la organización o corporación.

4.- Información departamental: interesa a un solo departamento.

5.- Información individual: interesa a cada individuo.

(objetivos)

OBJETIVOS DEL SISTEMA DE INFORMACION

1.- Proporcionar a los usuarios de herramientas que les permitan aprovechar los recursos de cómputo de manera fácil y oportuna para que puedan resolver sus propios problemas.

2.- Apoyar la tarea cortas de investigación que es uno de los objetivos de la ESPOL.

3.- Establecer una vía de comunicación adecuada entre los usuarios y el departamento de sistemas.

4.- Entrenar actualizar y dar soporte a los usuarios en los procedimientos para la utilización del software existente en INTERNET.

5.- Constituirse en una organización formal dentro del SI.

ORGANIZACION DEL SI-ESPOL

Es importante que desde el inicio el sistema de información cuente con una estructura organizacional apropiada donde las atribuciones funciones y responsabilidades de cada miembro esten claramente definidas

La posible conformación:

ADMINISTRADOR DEL S.I. o jefe del S.I.: esta persona tiene una posición importante en el área de sistemas básicamente se justifica en la habilidad de manejar un departamento y como el S.I. va a estar en contacto con muchos departamentos, es más difícil su manejo y entre sus características principales tenemos:

- persona que va a tomar decisiones sobre instalación de software modificaciones sustanciales del sistema. y es importante que esta persona se reporte directamente con el área de redes a la cual pondrá al tanto con los planes a realizar.

LOS INSTRUCTORES: son las personas que capacitan al usuario indicándoles como utilizar el software y a veces el hardware a fin de que puedan solucionar sus problemas evalúan soluciones alternativas y expresan sus necesidades.

ESPECIALISTAS EN PRODUCTOS: Estas personas dan apoyo a los usuarios en los productos en los que ellos son responsables y tienen que ser expertos en el uso de los mismos.

CONSULTORES: analizan la información que es presentada por los usuarios y la sintetizan en una solicitud de apoyo que identifique el mejor enfoque y paquete de software que ayudará a resolver el problema.

REQUERIMIENTOS DEL PERSONAL

Características del quienes conforman el SI.

- 1.-Conocimientos básicos de computación.
- 2.-Destreza para las comunicaciones interpersonales.
- 3.-Debe tener interés en la investigación.
- 4.-Capacidad para desarrollar su propia iniciativa.
- 5.-destreza para resolver problemas.
- 6.-Capacidad para la enseñanza.
- 7.-Paciencia y persistencia.

3.2 NORMAS RECOMENDADAS

PRINCIPIO GENERAL

(1) *Los principales servicios de SI-ESPOL están provistos para la ayuda a la investigación y la educación en y a través de NUESTRAS instituciones de investigación y educación. El uso para otros propósitos no es aceptable.*

USOS ESPECIFICAMENTE ACEPTABLES

(2) *Comunicación con investigadores y educadores extranjeros en conexión con la investigación o educación, cualquier red que el usuario extranjero emplee de tal manera que nos provea una comunicación que nos de un recíproco acceso a nuestros educadores e investigadores.*

(3) *Comunicación y intercambio para el desarrollo profesional, para mantenerse al tanto, o para el debate de temas en el campo o en el subcampo del conocimiento.*

(4) *Uso para sociedades-organizadas, asociaciones-universitarias, departamentos de gobierno, cualquier actividad involucrada con la investigación y educación del usuario.*

(5) *Uso en la aplicación o administración de contratos para la investigación o educación, pero no para otras relaciones de actividad pública.*

(6) *Cualquiera otra comunicación administrativa o actividades en apoyo de la investigación y educación.*

(7) *Anuncio de nuevos productos o servicios para uso en la investigación o educación, pero no para propaganda de ningún tipo.*

(8) *Comunicación poco frecuente de otro uso aceptable, excepto para el uso ilegal o para un uso específicamente inaceptable.*

USOS INACEPTABLES

- *Uso para actividades de lucro, a menos que sean cubiertas por el principio general o como un uso específicamente aceptable.*
- *El uso extensión para negocios personales o privados.*

3.3 FORMATOS GENERALES

Por el momento se soporto archivos texto tanto ASCII y para poder poner la información en el gopher debe seguir :

- *para los archivos textos deben ser las líneas de 70 caracteres de anchos para que puedan entrar en la pantalla.*
- *Los caracteres deben ser ascii 'Latin1'.*

También se soporta el formato word 2.0 . Se podría soportar el formato 6.0 pero la mayoría son del formato 6.0.

Se soporta también el formato de graficos gif se aconseja que las figuras no sean mayor que 60 Kbytes debido a que la velocidad actual de los modems.

También se soporta películas con formato mpg también se aconseja que no sean mayor de 60 Kbytes.

CAPITULO 4

4.- DISEÑO DETALLADO DE LAS APLICACIONES SOPORTADAS

A continuación se haría una explicación de las aplicaciones desarrolladas. El esquema a seguir para su explicación es el siguiente:

- 1.-Análisis
- 2.-Diseño
- 3.-Implementación
- 4.-Instructivo

Entre las aplicaciones desarrolladas tenemos:

- 1.-Manejo de ayudantías académicas.
- 2.-Manejo de pedidos de listados de contabilidad a Cesercomp.
- 3.-Consulta de Notas.
- 4.-Conexión de las aplicaciones de biblioteca a Internet.
- 5.-Directorio telefónico de la ESPOL.
- 6.-Información acerca de profesores, alumnos y trabajadores.

4.1 MANEJO DE AYUDANTÍAS ACADEMICAS.

ANÁLISIS

En la actualidad la manera en que se manejan los pedidos de ayudantías es el siguiente :

Las ayudantías que están disponibles se las pone en un cuadro informativo de la respectiva facultad o instituto, la persona o las personas interesadas se acercan a pedir información a la secretaria de dicha facultad o instituto, ella les indica los requisitos necesarios. Después de esto Los alumnos se encargan de llenar la solicitud y entregársela a la secretaria de la facultad para que ella se la entregue al profesor encargado y el seleccione a algún aspirante.

Lo que se propone es tener varias opciones en el gopher mediante las cuales tendremos información referente a las ayudantías:

- 1.- Información de los requisitos necesarios para ser ayudante en cualquier facultad (Información Constante).

2.- Las distintas ayudantías disponibles, las horas que necesitan ser dedicadas a esta ayudantía y el profesor que las necesita (Información Dinámica).

3.- La Solicitudes de ayudantías que deben ser llenadas por los estudiantes y mandadas automáticamente a los profesores interesados.

Estas tres opciones Cubren casi todos los aspectos de información que los estudiantes necesitan saber para ser ayudantes. Dándole al profesor titular una idea de quienes serían los alumnos interesados.

DISEÑO

El diseño de esta aplicación involucra tanto información que va a permanecer constante en el gopher, como información que va a ser dinámica y que va a estar almacenada en la base de datos ORACLE.

La información dinámica como se dijo va a ser manejada por medio de la base de datos ORACLE y el gopher va a acceder a ella por medio de ORAPERL, el lenguaje utilizado por el GOPHERSQL, para que se pueda manejar por medio de scripts los datos de la base.

En primer lugar tenemos que diseñar las tablas que van a mantener la información. A cada materia le corresponde una ayudantía.



¡Error! No se encuentra la fuente de la referencia.

Estas tablas creadas son: Personas, Materias y la descripción de cada una de ellas es la siguiente:

TABLA Personas

#ID_PERSONA	Varchar(7) not null,
CLASE	Varchar (1) not null,
APELLIDO_PATerno	Varchar (10) not null,
APELLIDO_MATERNO	Varchar(10) not null,
NOMBRE_PRIMERO	Varchar(10) not null,

NOMBRE_SEGUNDO Varchar(10),
DIRECCION_ELECTRONICA Varchar(25),
DIRECCION_DOMICILIARIA Varchar(40),
UNIDAD_PERTENECE Varchar(20),
FECHA_INGRESO Date,
STATUS Varchar(1).

TABLA Materia

ID_MATERIA Number(10),
NOMBRE_MATERIA Varchar (20),
UNIDAD Varchar (20),
HORAS_AYUDANTIA Number (2),
DISPONIBLE Varchar (1)

A continuación se hace una descripción detallada de los campos de cada tabla.

TABLA PERSONAS.

Esta tabla se va a usar adelante para mucha clase de información que va a referirse a los Profesores, Alumnos y Trabajadores de la ESPOL y los campos son:

ID_PERSONA es el número de roll del Profesor o Trabajador y el número de matrícula de los estudiantes.

CLASE es el tipo de persona puede haber tres posibilidades estas son P: Profesor, A: Alumnos, T: Trabajadores.

APELLIDO_PATERNO es el apellido paterno de la persona.

APELLIDO_MATERNO es el apellido materno de la persona.

NOMBRE_PRIMERO es el primer nombre de la persona.

NOMBRE_SEGUNDO es el segundo nombre de la persona.

DIRECCION_ELECTRONICA es la dirección electrónica de la persona.

DIRECCION_DOMICILIARIA es la dirección del domicilio.

UNIDAD_PERTENECE Es la unidad a la que pertenece la persona.

FECHA_INGRESO Es la fecha en la que ingreso la persona.

STATUS Es el status de la persona A:activa I:inactiva para alumnos, C:tiempo completo M:medio completo tanto para los profesores y trabajadores.

TABLA MATERIAS

Los campos de esta tabla son:

ID_MATERIA: es el código de la materia.

NOMBRE_MATERIA: es el nombre que tiene la materia.

UNIDAD: Unidad a la que pertenece la materia.

HORAS_AYUDANTIA: Las horas que tiene que disponer el ayudante.

DISPONIBLE: Si la ayudantia esta disponible S/N.

Toda la información referente a los profesores , materias deberán ser ingresadas por personas responsable por medio de FORMS que estarán disponible para estos usuarios.

IMPLEMENTACION

La implementación trae tras de sí la instalación de muchos productos adicionales tales como perl, oraperl.

El primer form afecta a la tabla personas, y el segundo form afecta a la tabla Materias. Todos estos forms estan descritos en el apéndice C-1.

En sí el ingreso de datos por medio de FORMS de oracle trae consigo la facilidad y la confiabilidad que el form presenta.

como se dijo tres son las opciones que estarán disponibles para los usuarios finales:

-La información referente a los requisitos de las ayudantias.

Consulta de Ayudantías Disponibles

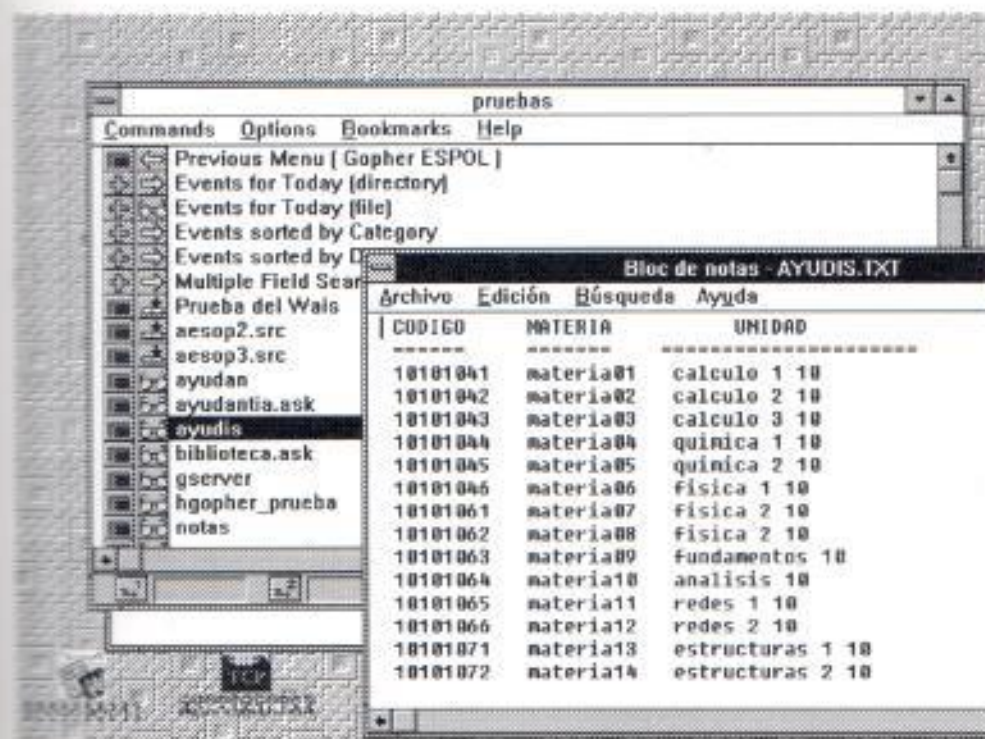


FIG No. 1

- Las ayudantías disponibles con el nombre de los profesores y las horas que se necesitan.

- Solicitud de ayudantía.

Solicitud de ayuda

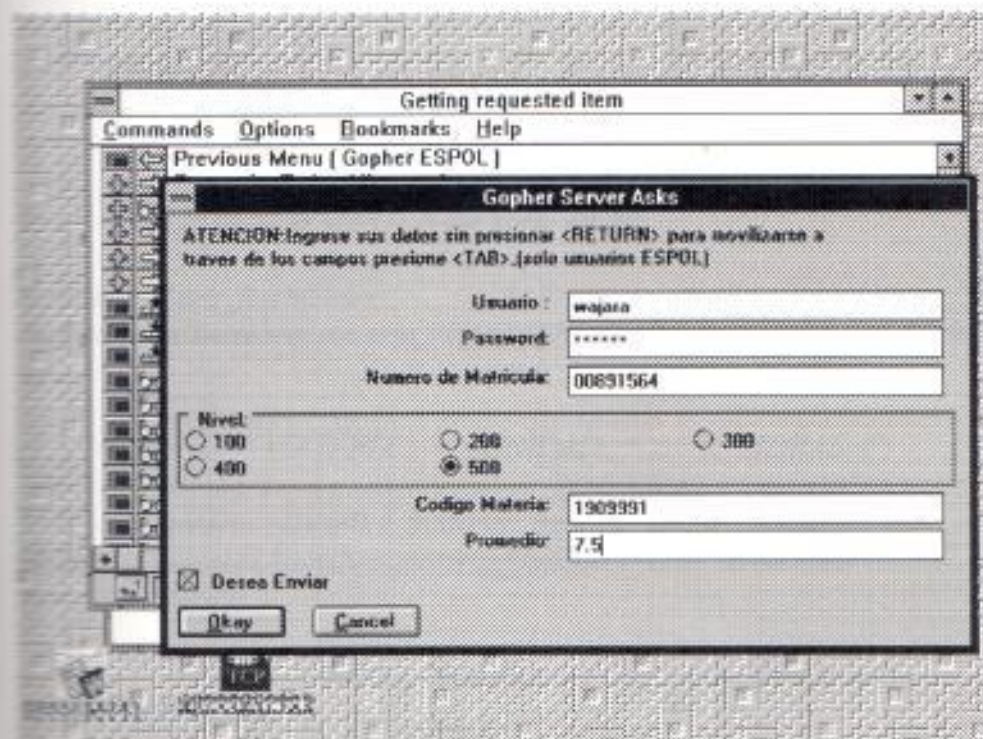


FIG. No 2

A continuación pongo los scripts desarrollados en *oraperl* para el acceso a esta información.

SCRIPT EN ORAPERL PARA LA CONSULTA DE AYUDANTIAS DISPONIBLES

```

#usr local/bin/oraperl
SO$user="gopher";
SO$pass="gopher";
SO$base="espol";

Sora_long = 1024;
Sora_cache = 20;
SEW{ORACLE_HOME} = "oracle/prog";
SEW{ORACLE_SID} = "espol";

```



```
Slda = &ora_login (SDBbase,SDBuser,SDBuser);
```

```
SLIST= " select id_materia,
         nombre_materia,
         unidad,
         horas_ayudantia
       from materias1
       where disponible='D'";
```

```
Scsr= &ora_open(Slda,SLIST) || die Sora_errstr;
```

```
Sofields = &ora_fetch(Scsr) || die Sora_errstr;
```

```
Sid_materia,Snombre_materia,Sunidad,Shoras_ayudantia) = &ora_fetch(Scsr);
```

```
print " CODIGO MATERIA UNIDAD HORAS\n";
```

```
print "-----\n";
```

```
print Sid_materia Snombre_materia Sunidad Shoras_ayudantia\n";
```

```
while((Sid_materia,Snombre_materia,Sunidad,Shoras_ayudantia) =
```

```
&ora_fetch(Scsr))
```

```
{
```

```
print Sid_materia Snombre_materia Sunidad
```

```
Shoras_ayudantia\n";
```

```
}
```

```
die Sora_errstr if (Sora_errno != 0);
```

```
&ora_close(Scsr) || die Sora_errstr;
```

```
&ora_logoff(Slda) || die Sora_errstr;
```

```
END
```

SCRIPT PARA LA SOLICITUD DE AYUDANTIA

SCRIPT PRINCIPAL

```
#!/usr/local/bin/oraperl
```

```
SDBuser="gopher";
```

```
SDBpass="gopher";
```

```
SDBbase="espol";
```

```

Susuario=<>;
Spasword=<>;
Smatricula=<>;
Snivel=<>;
Sid_materia=<>;
Spromedio=<>;
Sconfirmacion=<>;

```

```

chop(Susuari);
chop(Spasword);
chop(Smatricula);
chop(Snivel);
chop(Sid_materia);
chop(Spromedio);
chop(Sconfirmacion);

```

```

Sora_long = 1024;
Sora_cache = 20;

```

```

$ENV{ORACLE_HOME} = "oracle prog";
$ENV{ORACLE_SID} = "espol";
chop($resultado = `pcheck Susuario Spasword`);
if($resultado eq 'password correcto')

```

```

{
  $sda = &ora_login(SDBbase,SDBuser,SDBpass) | die "Sora_errstr" | n";

```

```

$LIST = "select apellido_paterno,
         apellido_materno,
         nombre_primero,
         unidad_pertenece,
         fecha_ingreso,
         status
        from personas
        where id_persona = $matricula";

```

```

$scsr = &ora_open($sda,$LIST) || die "Matricula inexistente",exit;
$sefields = &ora_fetch($scsr) || die "Sora_errstr",exit;

```

```

($sapellido_paterno,$sapellido_materno,$snombre_primero,$sente_per
tenece,$sfecha_ingreso,$sstatus) = &ora_fetch($scsr);

```

```

die "Sora_errstr" if($sora_errno != 0);
&ora_close($scsr) | die "Sora_errstr";

```

```

SLIST1="select nombre materia
      from materias1
      where id_materia=Sid_materia";

Scsr1= &ora_open(Slda,SLIST1) || die Sora_errstr;
Sfields = &ora_fetch(Scsr1)      || die Sora_errstr;

(Snombre_materia) = &ora_fetch(Scsr1);

die Sora_errstr if (Sora_erro != 0);
&ora_close(Scsr1) || die Sora_errstr;

push(@lines, "Solicitud de Ayudantia:\n\n");
push(@lines, "Yo Sapellido paterno      Sapellido materno,Snombre primero\n");
push(@lines, "solicito se me tome en consideracion para el concurso \n");
push(@lines, "de la AYUDANTIA ACADEMICA de la materia Snombre_materia\n");
push(@lines, "mis datos son los siguientes:\n\n");
push(@lines, "CODIGO MATERIA: Sid_materia\n");
push(@lines, "NOMBRE MATERIA: Snombre_materia\n");
push(@lines, "\n");
push(@lines, "APELLIDOS: Sapellido_materno Sapellido_paterno \n");
push(@lines, " NOMBRES : Snombre primero\n\n");
push(@lines, " FACULTAD-INSTITUTO: Sente_pertenece \n");
push(@lines, " NIVEL: Snivel \n");
push(@lines, " FECHA_INGRESO: Sfecha_ingreso\n");
push(@lines, " STATUS: Sstatus \n");
push(@lines, " \n\nUna copia de este mensaje sera mandado al profesor\n
encargado y otra al usuario que lo mando\n");
if (Sconfirmacion == '1')
{
  print @lines;
  open(emailit, "\usr/uch/mail -s \*** Solicitud de          ayudantia
academcia**" gopherda@espol.edu.ec      Susuario");
  print emailit @lines;
  close(emailit);
}
else
{
  print `cat .signature2`;
  print "\n";
  print "NO ha confirmado su solicitud ponga YES/NO\n";
}
&ora_logoff(Slda) || die Sora_errstr;

```

```

}
else
{
    print `cat .signature2`;
    print "\n";
    print "SU PASSWORD ES INCORRECTO O su usuario no
pertenece a la ESPOL\n";
    print "RECUERDE esta opcion esta disponible solo para usuarios de
la ESPOL\n";
}
END

```

EL SCRIPT PARA EL FORM DEL GOPHER

Note: ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a

Note: traves de los campos presione <TAB>, (solo usuarios ESPOL)

Note:

Ask: Usuario :

Askp: Password:

Ask: Numero de Matricula:

Choose: Nivel: 100 200 300 400 500

Ask:Codigo Materia:

Ask: Promedio:

Select: Desea Enviar: 0

INSTRUCTIVO

La información del ingreso de datos estan en el apéndice C-1, en este apéndice se ve los forms y los comandos desde el ambiente UNIX para ejecutar estos forms.

Hay dos opciones en las cuales se utiliza el ORAPERL estas son:

- 1.-Para la consulta de ayudantias.*
- 2.-Solicitud para la ayudantía deseada.*

Bueno el primer script seria de ejecución sin ninguna interfase con el usuario por lo que no requiere datos de el, es un script de consulta.

El segundo script es la solicitud de ayudantia, en este tenemos un form del gopher con algunos campos que llenar. Entre los primeros campos tendríamos un filtro de seguridad que permitiría mandar información solo de personas que tengan usuario en la ESPOL.

Los siguientes campos serian para:

el ingreso del número de matricula del alumno que por el momento no se lo valida por que necesitaríamos mas datos en la base.

El campo del Código de la materia que se lo puede ver en la segunda opción.

En el Tercer campo se pide el apellido paterno y el apellido materno.

En el Cuarto campo se pide el Primer Nombre.

En el Quinto campo se pide el Nivel.

En el Sexto se pide el promedio.

El Séptimo una confirmación si se la envia o no.

4.2 MANEJO DE PEDIDOS DE LISTADOS DE CONTABILIDAD A CESERCOMP.

ANALISIS.

En la actualidad el flujo de papeles de Contabilidad a centro de cómputo es un muy alto entrando a trámites "burocráticos" que demoran algunas veces el desarrollo de las actividades tanto en Contabilidad y en Cesercomp.

Una de soluciones más óptimas para disminuir el flujo de papeles sería el de migrar las aplicaciones a un administrador de bases de datos en el cual sería más flexible administrar datos y sacar reportes en algunos casos por el mismo usuario.

Pero en la actualidad contamos un sistema desarrollado en una plataforma 370 IBM y que necesita la intervención de un operador para sacar un listado y de un analista para sacar la información.

Se ha considerado por el momento solamente la manera de perder los listados por que este proceso ocupa casi el 15% del tiempo, en cuya trayectoria intervienen muchas personas como son : el usuario , el mensajero y la recepción de CESERCOMP y de aquí al analista encargado.

El mecanismo recomendado será el de utilizar el gopher para por medio de un correo electrónico poder mandar los requerimientos del usuario hacia una analista, eliminando de esta forma al mensajero y la recepción de CESERCOMP.

A la vez que disminuimos papeles y tiempo perdido.

DISEÑO E IMPLEMENTACION.

En el diseño e implementación de esta opción tenemos los siguientes datos que son requeridos:

En primer lugar tenemos el filtro de seguridad que nos pide el usuario y el password del mismo. Esta opción solo podrá ser usada por los usuarios del sistema.

Los campos del Form son:

FECHA: es la fecha del pedido.

SOLICITADO POR: es la unidad que lo solicito.

ENTREGAR A: es la unidad a la cual hay que entregar el listado.

CODIGO DEL LISTADO: Se le antepone el código SFBP #### los últimos cuatro números son la secuencia asignada.

NOMBRE DEL LISTADO: Es el nombre del Listado.

CIA:

FECHA DE CIERRE: El cierre

MES INGRESO: El ingreso

Hay que llenar algunos campos que son

CUENTA

TERCEROS

NUMERO DE COPIAS: Es el número de copias.

PROCESO: que puede ser cuenta por cuenta o de tercero por tercero.

A continuación se presentan los script que sirvieron para el manejo de la información.

SCRIPT PRINCIPAL

```
#!/bin/sh
read user
read pass
read roll
```

```

read solicitado
read codigo
read nombre
read CIA
read fecha_cierre
read mes_ingreso
read OP
read comprobante
read cuentas1
read cuentas2
read cuentas3
read cuentas4
read cuentas5
read cuentas6
read cuentas7
read cuentas8
read cuentas9
read cuentasa
read cuentasb
read cuentasc
read cuentasd
read cuentase
read cuentasf
read cuentasg
read cuentash
read cuentasi
read copias
read Proceso
read s1

```

```

if ( test "Ss1" = "1") then
  if(.pcheck Suser Spass grep -s "password correcto") then
    /usr/ucb/mail -s "LISTADOS" gopherda@espol.edu.ec Suser@espol.edu.ec << EOF

```

ORDEN DE PROCESO

+ APLICACION: SHF FECHA: 'date'

SOLICITADO POR: Ssolicitado ENTREGAR A: Ssolicitado

DESCRIPCION DEL PROCESO:

CODIGO: SFBP Scodigo

NOMBRE: Snombre

CIA:SCIA FECHA DE CIERRE: Sfecha_cierre MES DE

INGRESO: Smes_ingreso

OP: SOP COMPROBANTE: Scomprobante

CUENTAS TERCEROS

Scuentas1

Scuentas2

Scuentas3

Scuentas4

Scuentas5

Scuentas6

Scuentas7

Scuentas8

Scuentas9

Scuentasa

Scuentash

Scuentasc

Scuentasd

Scuentase

Scuentasf

Scuentasg

Scuentash

Scuentasi

Numero de Copias: Scopias

PROCESO: SProceso

EOF

echo "Su SOLICITUD Para Listados a Cesercomp han sido mandado"

echo " a la persona encargada "

echo ""

echo "Usted recibira una copia de la informacion"

cat signature2

cat


```

else
  echo ""
  echo "SU PASSWORD ES INCORRECTO O su usuario no pertenece a la
ESPOL"
  echo "RECUERDE esta opcion esta disponible solo para usuarios de
la ESPOL"
  cat .signature2
fi

else
  echo "Su SOLICITUD Para Listados a Cesercomp NO ha sido mandada"
  echo ""
  echo "Revisar bien las instrucciones "
  echo "Recuerde con <TAB> se moviliza a traves de los campos "
  echo "Con <RETURN> manda los datos "
  cat .signature2
fi
exit

```

SCRIPT DE FORM DEL GOPHER

Note: ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a

Note: traves de los campos presione <TAB>, (solo usuarios ESPOL)

Note:

Ask: Usuario :

Ask: Password:

Ask: Roll Administrativo:

Choose: Solicitado por: Contabilidad

Ask: Codigo: SFBP

Ask: Nombre del Listado:

Note:----- DATOS REQUERIDOS -----

Ask: CIA:

Ask: Fecha de Cierre:

Ask: Mes Ingreso:

Ask: OP:

Ask: Comprobante:

Note: A continuacion ingrese en el siguiente formato

Note: *****CUENTAS---TERCEROS*****

Ask:

Ask:

Ask:

Ask:

Ask:

Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:
 Ask:

Ask: Numero de Copias:

Choose: Proceso: cuenta por cuenta tercero por tercero

Select: Desea enviar: 0

FORMA DE PEDIDOS DE LISTADOS A CERSERCOMP

Gopher Server Asks

ATENCIÓN: Ingrese sus datos sin presionar <RETURN> para movilizarse a través de los campos presione <TAB> (solo usuarios ESPOL)

Usuario:

Password:

Rol Administrativo:

Solicitado por:

Contabilidad

Codigo: SFBP

Nombre del Listado:

----- DATOS REQUERIDOS -----

CIA:

Fecha de Cierre:

Mes Ingreso:

DP:

Comprobante:

A continuación ingreso en el siguiente formato
 *****CUENTAS-----TERCEROS*****

Ok Cancel

FIG. No. 3

4.3 SISTEMA DE NOTAS.

ANÁLISIS

El sistema de notas que se propone es análogo al que tiene el 43XX en funcionamiento actualmente para la consulta de estas.

Esta opción de consulta de notas es tal vez una de las más importantes porque detrás de ella hay muchos procedimientos que han sido hechos. Tales como bajar información de 43XX a la base de datos ORACLE. Abriendo una posibilidad más para la migración de información del 43XX a plataformas más pequeñas pero con un potencial actualmente muy grande.

Como se dijo al principio este sistema de notas es análogo al del 43XX pero tiene la ventaja de poder ser accesado por medio de INTERNET desde cualquier parte del mundo por cualquier usuario.

DISEÑO

El diseño de esta aplicación al igual que la anterior involucra a la base de datos ORACLE y el gopher va a acceder a ella por medio del ORAPERL el lenguaje utilizado por el GOPHERSQL para que pueda manejar por medio de scripts los datos de la base.

Las tablas también involucran las tablas anteriormente mencionadas pero con la diferencia que ahora los registros que van a utilizarse va a ser el de los alumnos. y podemos observar que a un alumno puede tener muchas materias y una materia puede tener muchos alumnos.

Error! Marcador no definido.

Pero Normalizando esta tablas tenemos que nace un ente por la relación muchos a mucho por lo que tenemos:

Error! Marcador no definido.

Estas tablas *Personas*, *Notas*, *Materias* y la descripción de cada tabla es la siguiente:

TABLA *Personas*

#ID_PERSONA Varchar(7) not null,
CLASE Varchar (1) not null,
APELLIDO_PATERO Varchar (10) not null,
APELLIDO_MATERNO Varchar(10) not null,
NOMBRE_PRIMERO Varchar(10) not null,
NOMBRE_SEGUNDO Varchar(10) ,
DIRECCION_ELECTRONICA Varchar(25),
DIRECCION_DOMICILIARIA Varchar(40),
UNIDAD_PERTENECE Varchar(20),
FECHA_INGRESO Date,
STATUS Varchar(1).

TABLA *Notas*

#ID_MATERIA Varchar (10),
#ID_PERSONA Varchar (7),
PRIMER_PARCIAL Number(5,2),
SEGUNDO_PARCIAL Number (5,2),
NOTA_FINAL Number (5,2),
HORAS_MATERIA Number(2),
AÑO Number(4),
SEMESTRE Number(2),
APROBADO Varchar (1)

TABLA *Materia*

#ID_MATERIA Varchar(10),
NOMBRE_MATERIA Varchar (20),
UNIDAD Varchar (20),
HORAS_AYUDANTIA Number (2),
DISPONIBLE Varchar (1)

A continuación se hace una descripción detallada de los campos de cada tabla.

TABLA PERSONAS.

Esta tabla personas ya fue descrita anteriormente.

TABLA NOTAS

Esta tabla es una tabla de paso que se creo por la relación mucho a muchos por eso su llave son los campos ID_PERSONA y

ID MATERIA que pertenecen a otras tablas pero además tiene los campos:

PRIMER_PARCIAL: Es la primera nota del estudiante.

SEGUNDO_PARCIAL: Es la segunda nota del estudiante.

NOTA_FINAL: Es la nota final del semestre.

AÑO: es el año en que fueron dictadas.

SEMESTRE: es el semestre en que han sido dictadas.

APROBADO: Nos indica si aprobo o no el semestre.

TABLA MATERIAS

Esta tabla ya fue explicada anteriormente.

Toda la información referente a notas, personas y materias deberán ser ingresada por personas responsable por medio de FORMS que estarán disponible para estos usuarios.

IMPLEMENTACION

Para el ingreso de información se han desarrollado 3 forms el primero que afecta directamente a la tabla Notas, el segundo form afecta a la tabla personas, y el tercer form afecta a la tabla Materias. Todos estos forms estan descritos en el apendice C-1.

Los script de oraperl son los siguientes:

EL SCRIPT PRINCIPAL

```
#!/usr/local/bin/oraperl
```

```

chop(Susername= <>);
chop(Spassword= <>);
chop(Smatricula= <>);
chop(Snivel= <>);
chop(Ssemestre= <>);
chop(Sano= <>);
SDBuser="gopher";
SDBpass="gopher";
SDBbase="espol";

Sora_long = 1024;
Sora_cache = 20;

SENV{'ORACLE_HOME'}= "/oracle/prog";
SENV{'ORACLE_SID'}= "espol";
chop(Sresultado= '.pcheck Susername Spassword');
if(Sresultado eq 'password correcto')
{
  Swhere1=" A.semestre = Ssemestre and " if (Ssemestre);
  Swhere2=" A.ano = Sano and " if (Sano);
  Swhere= "Swhere1 Swhere2";
  Slda=&ora_login(SDBbase,SDBuser,SDBpass)||die"Sora_errstr \n";

  SLIST="select A.Apellido_paterno,
             A.apellido_materno,
             A.id_materia,
             B.nombre_materia,
             A.primera_parcial,
             A.segunda_parcial,
             A.nota_final,A.aprobado
from resultado A,materias1 B
where A.id_persona=Smatricula and
       Swhere
       B.id_materia = A.id_materia";

  Scsr= &ora_open(Slda,SLIST) || die Sora_errstr;
  Sfields = &ora_fetch(Scsr) || die Sora_errstr;

  (Sapellido_paterno,Sapellido_materno,Sid_materia,Snombre_mater
  ia,Sprimera_parcial,Ssegunda_parcial,Snota_final,Saprobado) = &ora_fetch(Scsr);

```

```

print"#####\n";
print"  ## # #### #### ####  #### \n";
print"  ## # # # ## # # ## \n";
print"  ## # # # ## ####  ## \n";
print"  # ## ####  ## # # #### \n";
print"#####\n";

print" APELLIDO PATERNO: Sapellido_paterno\n";
print" APELLIDO MATERNO: Sapellido_materno\n";
print" MATRICULA      : Smatricula\n";
print" AN-O           : Sano\n";
print" TERMINO No.    : Ssemestre\n";

print"CODIGO MATERIA 1er.PARCIAL 2do.PARCIAL NOTA-FINAL STATUS\n";
print" -----
----- \n";
print" Sid_materia  Snombre_materia      Sprimer_parcial      Ssegundo_parcial
Snota_final   Saprobadado\n";

while      ((Sapellido_paterno,Sapellido_materno,Sid_materia,Snombre_mate
ria,Sprimer_parcial,Ssegundo_parcial,Snota_final,Saprobadado) =
&ora_fetch(Scsr))
{
print" Sid_materia  Snombre_materia      Sprimer_parcial      Ssegundo_parcial
Snota_final   Saprobadado\n";
}
die Sora_errstr if (Sora_errno != 0);
do ora_close(Scsr) || die Sora_errstr;

do ora_logoff(Slda) || die Sora_errstr;
}
die
{
print `cat .signature2`;
print "\n";
print "SU PASSWORD ES INCORRECTO O su usuario no
pertenece a la ESPOL\n";
print "RECUERDE esta opcion esta disponible solo para      usuarios de
la ESPOL\n";
}
END

```

SCRIPT DEL FORM EN EL GOPHER

Note: ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a

Note: traves de los campos presione <TAB>, (solo usuarios ESPOL)

Note:

Ask: Usuario :

Ask: Password:

Ask: Numero de Matricula:

Choose: Nivel: 100 200 300 400 500

Ask: Semestre:

Ask: An-o:

INSTRUCTIVO.

En realidad el modo de funcionamiento es fácil al igual que el sistema que esta funcionando en el 43XX.

FORMA DE CONSULTA DE NOTAS

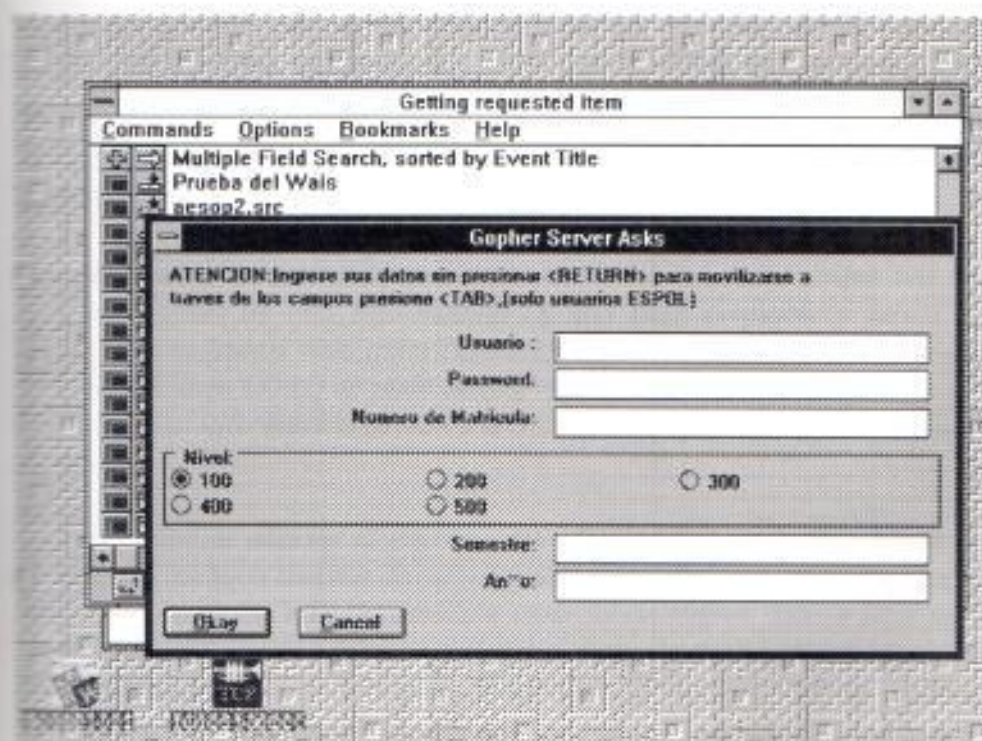


FIG. No 4

Al ingresar a la consulta se pide la identificación del usuario y el password que le corresponde, el usuario tendrá que ingresar los datos como son la matrícula, el semestre, el año, el código de la materia.

Si se deja en blanco cualquiera de estos campos se toma como default a todos los que hay en la base por ejemplo:

Si se llena la matrícula y el año trae todas las materias que tuvo el alumno en ese año.

Si se llena los datos de ese año y el semestre deseado trae las materias de ese semestre.

Si se llena los datos de ese año, el semestre y el código de la materia se trae los datos de esa materia.

4.4 CONEXION DE LAS APLICACIONES DE BIBLIOTECA A INTERNET.

ANALISIS.

En la actualidad la ESPOL cuenta con un sistema de biblioteca que tiene una parte de la información en el IBM 370 y otra parte de la información esta en IRIS que una base de datos que se encuentra en una PC.

La abundante información que existe en la biblioteca hace que sea necesario el almacenamiento de esta información en una base confiable y sólida y de rápido acceso. Como hemos visto anteriormente todas estas facilidades la brinda ORACLE. Con su utilitario LOAD se puede migrar información que se encuentra en ASCII por lo que migrar información de 43XX a Oracle o de una base que se encuentra en DOS a Oracle.

La facilidad de acceder al gopher y las ventajas que presenta a futuro hacen que el gopher sea un buen sistema de información. Hay distintas otras opciones para poder poner a disposición esta información como el de migrar todo a IRIS pero esta base no se presenta flexible ni tiene todas las ventajas de los manejadores de bases de datos conocidos.

DISEÑO

Como se dijo al principio se propone un sistema que pueda presentar los datos a los usuarios por medio del gopher y que este a su vez pueda acceder a ORACLE , para esto tendremos que utilizar tablas que puedan soportar esta información después de una entrevista con la personas encargadas de la biblioteca y al hacer un pequeño análisis llegamos a la conclusión que se pueden almacenar los datos en 1 tabla.

Error! Marcador no definido.

A continuación tendremos una descripción de esta tabla presente.

TABLA LIBRO

#DEWEY number (10) primary key not null,
TITULO Varchar(20) not null,
MATERIA Varchar (20) not null,
IDIOMA Varchar (10),
AUTOR Varchar (10),
INVENTARIO Varchar (10),
VALOR Varchar (10,2),
FECHA_INGRESO date,
SOLICITADO Varchar (10)

A continuación presentamos cada uno de los campos que tienen las tablas previas.

TABLA LIBRO

#DEWEY es el código asignado a este libro.
TITULO es el título del libro.
MATERIA La materia que trata el libro.
IDIOMA es el idioma que esta escrito el libro.
AUTOR es el nombre del autor.
INVENTARIO es el código del inventario.
VALOR es el valor que tiene el libro.
FECHA DE INGRESO es la fecha en que ingreso el libro.
SOLICITADO es la persona que solicito el libro o la facultad.

IMPLEMENTACION

La implementación de este sistema se la hizo con el lenguaje ORAPERL que nos permite el acceso a la base de datos con lo cual podemos utilizar sentencias SQL para que puedan hacer búsquedas y se puedan utilizar los índices que nos brinda ORACLE y de esta manera poder acceder de una manera más rápida.

Para el ingreso de información se han creado forms que relacionen estas dos tablas y se puedan ingresar o consultar la información desde el ambiente ORACLE.

A continuación ponemos los scripts que hicieron falta para poner esta información a disposición del gopher.

SCRIPT PRINCIPAL PARA BIBLIOTECA

```
#!/usr/local/bin/oraperl
chop(Susername = <>);
chop(Spassword = <>);
chop(Smateria = <>);
chop(Sautor = <>);
chop(Stitulo = <>);
chop(Sidioma = <>);

$DBuser = "gopher";
$DBpass = "gopher";
$DBbase = "espol";

Sora_long = 1024;
Sora_cache = 20;

$ENV{ORACLE_HOME} = "/oracle/prog";
$ENV{ORACLE_SID} = "espol";
chop(Sresultado = `pcheck Susername Spassword`);

if (Sresultado eq 'password correcto')
{
  $where1 = " (B.apellidos like '%Sautor%' or B.nombres like '%Sautor%')
and " if (Sautor);
  $where2 = " A.materia like '%Smateria%' and " if (Smateria);

  $where3 = " A.titulo like '%Stitulo%' and " if (Stitulo);
```

```

Swhere4=" A.idioma like "%Sidioma%" and " if (Sidioma);
Swhere - "Swhere1 Swhere2 Swhere3 Swhere4";
Slda=&ora_login(SDBbase,SDBuser,SDBpass)||die "Sora_errstr          \n";

SLIST="select A.titulo,
        A.materia,
        A.idioma,
        A.dewey,
        B.apellidos,
        B.nombres,
        B.cutter
from autor B,libro A
where Swhere B.cutter like A.autor";

Scsr= &ora_open(Slda,SLIST) || die Sora_errstr;
Sfields = &ora_fetch(Scsr)          || die Sora_errstr;

(Stitulo,Smateria,Sidioma,Sdewey,Sapellido,Snombre,Scutter) = &ora_fetch(Scsr);

print
===== \n";
print "TITULO: Stitulo SCODIGO:Sdewey\n";
print "AUTOR:Sapellido Snombre SCODIGO:Scutter \n";

while ((Stitulo,Smateria,Sidioma,Sdewey,Sapellido,Snombre,Scutter) =
&ora_fetch(Scsr))
{
print
===== \n";
print "TITULO: Stitulo SCODIGO:Sdewey\n";
print "AUTOR:Sapellido Snombre SCODIGO:Scutter \n";
}
die Sora_errstr if (Sora_erno != 0);
do ora_close(Scsr)|| die Sora_errstr;
do ora_logoff(Slda) || die Sora_errstr;

print
===== \n";

```



```

}
else
{
    print `cat .signature2`;
    print "n";
    print "SU PASSWORD ES INCORRECTO O su usuario no pertenece a la
    ESPOL\n";
    print "RECUERDE esta opcion esta disponible solo para usuarios de la
    ESPOL\n";
}
__END__

```

SCRIPT DEL FORM DEL GOPHER

Note: ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a

Note: traves de los campos presione <TAB>, (solo usuarios ESPOL)

Note:

Ask: Usuario :

Askp: Password:

Ask: Materia:

Ask: Autor:

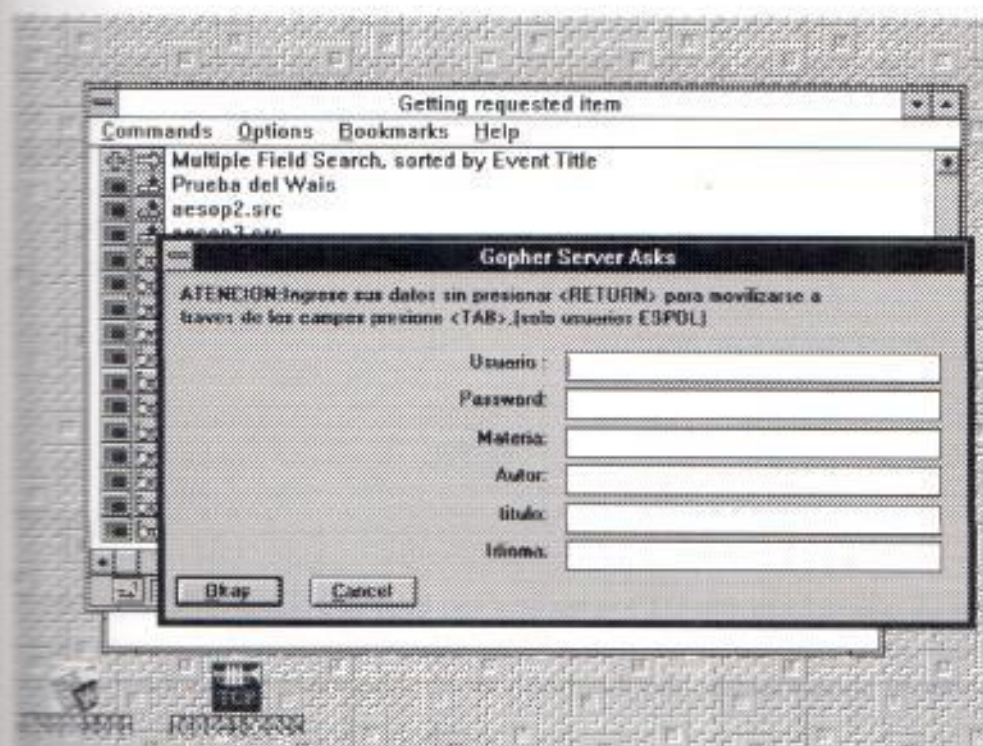
Ask: titulo:

Ask: Idioma:

INSTRUCTIVO.

El modo de usar esta información es muy fácil simplemente se pone el nombre el autor que quiere y trae la información que tiene con respecto al autor. Si se quiere especificar un poco mas se llena los datos de AUTOR y MATERIA y trae los datos de este autor y los libros de esta materia que se necesitan. y asi como uno desee. El idioma también es un pará de búsqueda que también se puede ingresar.

Los forms y el manejo para el ingreso de información en la bases se puede ver con más detalle en el apéndice C-1.

FORMA DE CONSULTA DE BIBLIOTECA*FIG. No 5*

4.5 DIRECTORIO TELEFONICO DE LA ESPOL

ANALISIS

El directorio telefónico de la ESPOL es talvez uno de los más consultados. Todos estos datos tienen que ser almacenados y podrían ser consultados tanto por nombre de usuarios, por ubicación, etc.

El resultado nos mostrará un archivo ascii con los datos que coincidieron con los patrones de búsqueda.

DISEÑO

En primera instancia que optó por utilizar el CSO que es el protocolo utilizado por algunas universidades para el manejo de información, al analizar el software se vio que era bueno pero tenía un pequeño problema este manejador de archivos ocupaba más menos unos 16 mega de espacio por lo que resultaba muy grande para el espacio en disco con que cuenta la ESPOL. Se hizo las consultas pertinentes y se decidió que la mejor opción era la de utilizar la base de datos ORACLE.

Se considero que tenían que haber una tabla la que contiene los datos de los teléfonos y su ubicación en la ESPOL.

Error! Marcador no definido.

Podemos ver que a un teléfono podemos asignar una persona. A continuación tenemos la tabla de Teléfonos que contiene la información de los teléfonos, la ubicación física, la persona responsable.

TABLA TELEFONO

NUMERO_TELEFONICO Number (6) not null,
NOMBRE_USUARIO Varchar(20),
ENIDAD Varchar (20) not null,

La descripción de los campos es la siguiente:

NUMERO_TELEFONICO Es el número telefónico propiamente dicho.

NOMBRE_USUARIO Es la persona que tiene responsabilidad sobre el teléfono.

UNIDAD Es la ubicación física del teléfono.

IMPLEMENTACION

La información es ingresada a la base en primera instancia por el utilitario LOAD y a continuación tendrá que ser ingresada por medio de forms de ORACLE que son descritos en el apéndice C-1.

A continuación se ponen los script utilizados para hacer la consulta de los teléfonos.

SCRIPT PRINCIPAL

```
#!/usr/local/bin/oraperl
chop(Snombre= <>);
chop(Sunidad= <>);
chop(Stelefono= <>);
SDBuser="gopher";
SDBpass="gopher";
SDBbase="espol";

Sora_long = 1024;
Sora_cache = 20;
$ENV{'ORACLE_HOME'}= "/oracle/prog";
$ENV{'ORACLE_SID'}= "espol";
$where4=" nombre_usuario like '%Snombre%' " if (Snombre);
$where5=" unidad like '%Sunidad%' " if (Sunidad);
$where6=" numero_telefonico = Stelefono " if (Stelefono);
$where= "$where4
$where5
$where6";
$lda=&ora_login(SDBbase,SDBuser,SDBpass) || die "Sora_errstr 'n' ";
$LIST="select nombre_usuario,unidad,numero_telefonico
from telefonos
where $where ";
$csr= &ora_open($lda,$LIST) || die Sora_errstr;
```


La descripción de los campos es la siguiente:

NUMERO TELEFONICO Es el número telefónico propiamente dicho.

NOMBRE_USUARIO Es la persona que tiene responsabilidad sobre el teléfono.

UNIDAD Es la ubicación física del teléfono.

IMPLEMENTACION

La información es ingresada a la base en primera instancia por el utilitario LOAD y a continuación tendrá que ser ingresada por medio de forms de ORACLE que son descritos en el apéndice C-1.

A continuación se ponen los script utilizados para hacer la consulta de los teléfonos.

SCRIPT PRINCIPAL

```
#!/usr/local/bin/oraperl
chop($nombre= <>);
chop($unidad= <>);
chop($telefono= <>);
$DBuser="gopher";
$DBpass="gopher";
$DBbase="espol";

$ora_long = 1024;
$ora_cache = 20;
$ENV{'ORACLE_HOME'}= "/oracle/prog";
$ENV{'ORACLE_SID'}= "espol";
$where4=" nombre_usuario like '%$nombre%' " if ($nombre);
$where5=" unidad like '%$unidad%' " if ($unidad);
$where6=" numero_telefonico = $telefono " if ($telefono);
$where= " $where4
$where5
$where6";
$lda=&ora_login($DBbase,$DBuser,$DBpass)||die "Sora_errstr 'n' ";
$LIST="select nombre_usuario,unidad,numero_telefonico
from telefonos
where $where ";
$csr=&ora_open($lda,$LIST) || die Sora_errstr;
```

```

Snfields = &ora_fetch(Scsr) || die Sora_errstr;

(Snombre_usuario,Sunidad,Snumero_telefonico) = &ora_fetch(Scsr);

print" INDICACIONES GENERALES \n";
print" Los Numeros que estan separados por un guion son numeros\n";
print" internos que pueden ser accesados desde el exterior de\n";
print" la ESPOL anteponiendo los digitos 269 XXX \n";
print" Los numeros que estan unidos como 456565 son telefonos\n";
print" directos que pueden ser accesados desde el exterior de\n";
print" la ESPOL marcandolos directamente\n\n";
print"
-----
===== \n";
print" PERSONA: Snombre_usuario\n";
print" UNIDAD: Sunidad\n";
print" TELEFONO: Snumero_telefonico\n";

while((Snombre_usuario,Sunidad,Snumero_telefonico) = &ora_fetch(Scsr))
{

print"
-----
===== \n";
print" PERSONA: Snombre_usuario \n";
print" UNIDAD: Sunidad\n";
print" TELEFONO: Snumero_telefonico\n";

}
die Sora_errstr if (Sora_errno != 0);
do ora_close(Scsr)|| die Sora_errstr;
&ora_logoff(Slda) || die Sora_errstr;

print"
----- \n";
__END__

```

SCRIPT DEL FORM DEL GOPHER

Note: ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a

Note: traves de los campos presione <TAB>, (solo usuarios ESPOL)

Note:

Ask: Nombre:

Ask: Unidad:

Ask: Telefono:

INSTRUCTIVO.

Al igual que los otras opciones el manejo es muy sencillo hay que ingresar los patrones de búsqueda que uno desea encontrar.

Tenemos en el form del gopher los siguientes campos:

PERSONA es la persona responsable

UNIDAD es la unidad en que se encuentra el teléfono.

TELEFONO es el número telefónico

Uno puede meter cualquiera de estos parámetros y el oracle se encargará de encontrar patrones que se asemejen a estos.

Ejemplo si ingresamos el apellido de una persona nos traerá los teléfonos que tienen como responsables a personas con este apellido.

Si lo hacemos con la unidad nos traerá los teléfonos de esta unidad.

Si ingresamos el teléfono nos traerá los datos del responsable de este teléfono.

FORMA DE CONSULTA DE TELEFONOS

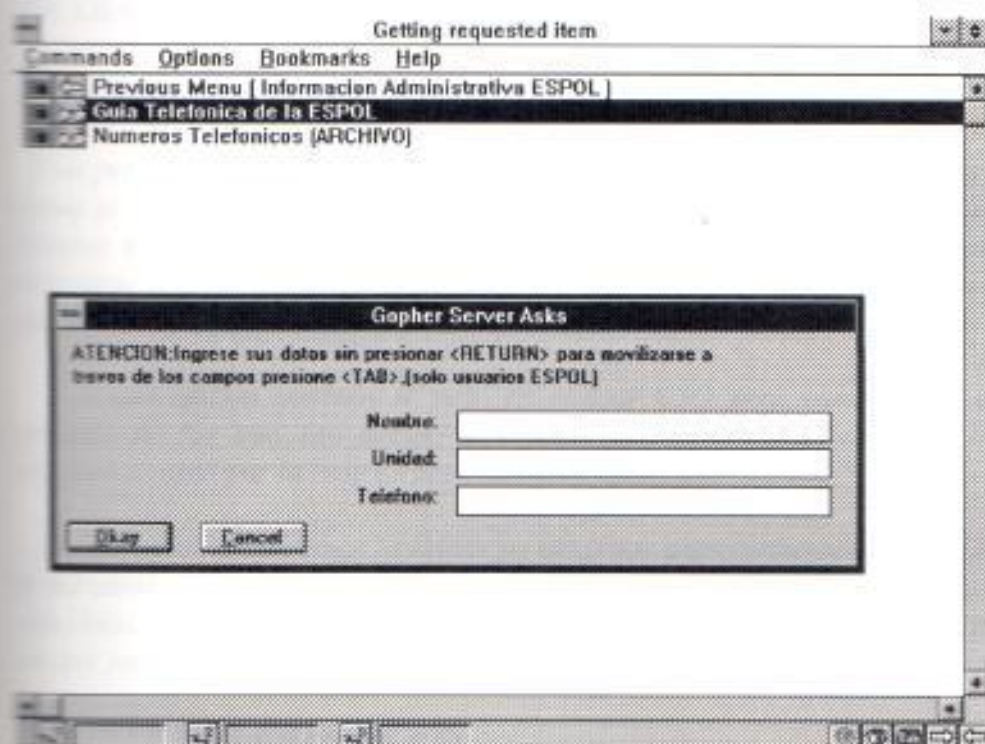


FIG. No 6

4.6 INFORMACION ACERCA DE PROFESORES, ALUMNOS Y TRABAJADORES.

ANALISIS

Una persona extraña a la ESPOL que busque datos de alguna persona en particular como su dirección electrónica, unidad a la que pertenece, no tendría el apoyo de un sistema que le permita buscar a el mismo dicha información, por lo que se hace necesario que pongamos a disposición del público datos de personas en particular que están en la ESPOL.

En esta opción también se pesó en utilizar CSO pero como vimos en el caso anterior no fue muy útil debido al espacio que ocupaba en el disco. Por lo que también se optó por la base Oracle.

En este caso utilizaremos la tablas ya descritas anteriormente como PERSONAS y pondremos a disposición del público datos como:

Apellidos, Nombres, Dirección electronica, si es un profesor, estudiante o trabajador, en que facultad trabaja, que materias da, Y el status.

DISEÑO E IMPLEMENTACION

Bueno en el diseño no se tiene que explicar mucho sobre la tabla de personas pues ya fue explicada anteriormente.

El script que se implemento para hacer la consulta lo ponemos a continuación.

SCRIPT PRINCIPAL

```
#!/usr/local/bin/oraperl
chop($apellido_paterno=<>);
chop($apellido_materno=<>);
chop($nombre=<>);
chop($stipo=<>);
chop($sumidad=<>);
$DBuser="gopher";
$DBpass="gopher";
$DBbase="espol";

$ora_long = 1024;
$ora_cache = 20;
```

```

SENV{'ORACLE_HOME'}= "/oracle/prog";
SENV{'ORACLE_SID'}= "espol";
if ( $Stipo eq 'PROFESOR' )
  { $clase = 'P'; }
if ( $Stipo eq 'ALUMNO' )
  { $clase = 'A'; }
if ( $Stipo eq 'TRABAJADOR' )
  { $clase = 'T'; }
$wh2=" apellido_paterno like '%$apellido_paterno%' and "
($apellido_paterno);
$wh3=" apellido_materno like '%$apellido_materno%' and "
if ($apellido_materno);
$wh4=" nombre_primero like '%$nombre%' and "
if ($nombre_primero);
$wh5=" unidad_pertenece like '%$unidad%' and "
if ($unidad);
$wh6=" clase = '$clase' and" if ($Stipo);
$wh = " $wh2
      $wh3
      $wh4
      $wh5
      $wh6";

$lda=&ora_login(SDBbase,SDBuser,SDBpass)||die "Sora_errstr" "n";
$LIST="select apellido_paterno,
             apellido_materno,
             nombre_primero,
             nombres_segundo,
             direccion_electronica,
             status,
             unidad_pertenece
       from personas
       where $wh id_persona=id_persona ";
$csr=&ora_open($lda,$LIST) || die Sora_errstr;
$fields = &ora_fetch($csr) || die Sora_errstr;

($apellido_paterno,$apellido_materno,$nombre_primero,$nombre_s
egundo,$direccion_electronica,$status,$unidad_pertenece) = &ora_fetch($csr);

print
=====
~n";
print " APELLIDOS: $apellido_paterno $apellido_materno\n";

```

```

print" NOMBRES: Snombre primero Snombres segundo\n";
print" DIRECCION ELECTRONICA: Sdireccion_electronica\n";
print" STATUS: Sstatus\n";
print" UNIDAD: Sunidad_pertenece\n";
while((Sapellido_paterno,Sapellido_materno,Snombre_primero,Sno
mbre_segundo,Sdireccion_electronica,Sstatus,Sunidad_pertenece)
&ora_fetch($csr))
{
print"
-----
\n";
print" APELLIDOS: Sapellido_paterno Sapellido_materno\n";
print" NOMBRES: Snombre_primero Snombres_segundo\n";
print" DIRECCION ELECTRONICA: Sdireccion_electronica\n";
print" STATUS: Sstatus\n";
print" UNIDAD: Sunidad_pertenece\n";
}
die Sora_errstr if (Sora_errno != 0);
do ora_close($csr)|| die Sora_errstr;
&ora_logoff($lda) || die Sora_errstr;

print"
-----
\n";

__END__

```

SCRIPT DEL FORM DEL GOPHER

Note: ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a

Note: traves de los campos presione <TAB>, (solo usuarios ESPOL)

Note:

Ask: Apellido Paterno:

Ask: Apellido Materno:

Ask: Nombre:

Choose: CARGO: ALUMNO PROFESOR TRABAJADOR

Ask: Unidad:

INSTRUCTIVO

Por ser un form de consulta estará libre para cualquier persona que quiera acceder a esta información por lo que no se pondrán filtros de seguridad en su ingreso.

Su funcionamiento sera simple: solo habrá que ingresar los patrones de búsqueda que se desea encontrar en los campos del form del gopher.

En estos campos que son

APELLIDO PATERNO:

APELLIDO MATERNO:

NOMBRE:

FACULTAD:

TIPO PERSONA:

Con estos datos cualquiera que sea el criterio tendremos a la persona que calza con esos patrones por ejemplo si ponemos todos los de apellido paterno JIMENEZ me traerá todos los alumnos, profesores, y trabajadores que cumplen con ese criterio. y de esa manera se podrán hacer filtros muy específicos.

FORMA DE CONSULTA DE PERSONAS

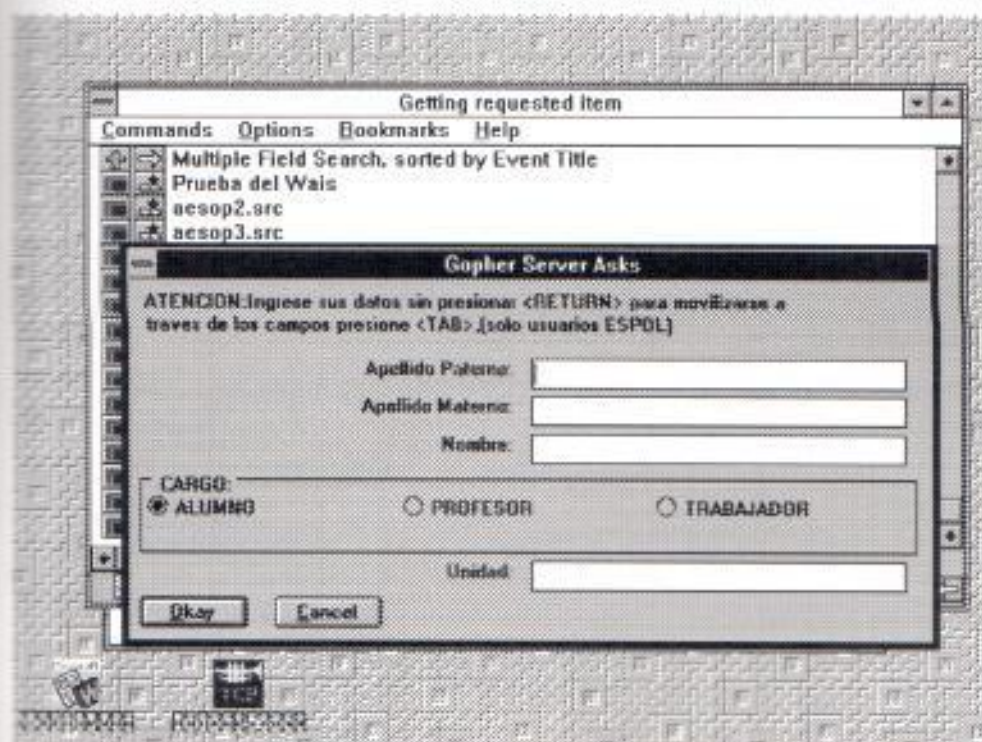


FIG. No 7

CAPITULO 5

5.1 INSTALACION DEL GOPHERSQL

Podemos definir al *gophersql* como un programa el cual acepta requerimientos del *gopher* y traslada estos a requerimientos SQL. Este puede hablar con bases de datos tales como Sybase y Oracle, pero para que el script pueda hablar tenemos que tener instalado en el computador tanto PERL como ORAPERL. Antes de describir el GOPHERSQL describiremos a PERL y a ORAPERL.

PERL

PERL significa " Practical Extraction and Report Language". El PERL es un interpretador de lenguaje optimizado para la búsqueda arbitraria en archivos textos, extrayendo información de estos archivos, e imprimiendo reportes basados en esta información. Este también es un buen lenguaje para muchas tareas administrativas. El lenguaje está hecho para que sea práctico (fácil de usar, eficiente, completo). Este combina algunas de las ventajas del C, Sed, Awk y Sh, haciendo a las personas que están familiarizadas con estos lenguajes muy fácil de aprenderlo. A diferencia de muchos utilitarios en Unix, perl no limita el volumen de los datos. La recursión es ilimitada. Y las tablas de Hash usadas por los arreglos asociativos crecen cuando es necesario previniendo la degradación en el performance. Perl usa sofisticadas técnicas para la búsqueda de patrones y cuando se lo hace en volúmenes de información muy grande lo hace de una manera muy rápida. Los scripts en Perl son tan rápidos como los programas en C.

ORAPERL

Oraperl es una versión de Perl el cual se ha sido extendido (gracias a las características *usersubs*) para permitir el acceso a bases de datos ORACLE. Las funciones que hacen posible este acceso se las describe a continuación.

Las principales funciones para el acceso a la base de datos son *&ora_login()*, *&ora_open()*, *&ora_bind()*, *&ora_fetch()*, *&ora_close()*, *&ora_do()* y *ora_logoff()*.

Sida = &ora_login (Snombre_base, Susuario, Spassword)

Para poder acceder a la información mantenida dentro de la base de datos Oracle, un programa tiene primero que logonearse dentro de esta llamando a la función *&ora_login ()*. Esta función es llamada con tres parametros que son el nombre de la base a usar, el usuario de la base, y el tercero es el password de este usuario. El valor retornado es el identificador del login, y en el ejemplo lo identificamos como *Sida*.

Scsr = &ora_open (Slda, Sstatement [, cache])

Para especificar una sentencia SQL a ser ejecutada, el programador debería llamar a la función &ora_open(). Esta función tiene al menos dos parámetros: un identificador del login (obtenido por la función &ora_login()) y la sentencia SQL a ser ejecutada. Un tercer parámetro es opcional especifica el número de registros a ser traídos por la función. Esta función retorna el identificador de la sentencia SQL (esta función es equivalente a abrir un cursor en Oracle PL/SQL) y en el ejemplo lo tenemos almacenado en Scsr.

Otros ejemplos de esta función son:

```
Scsr = &ora_open (Slda, 'select ename, sal from emp order
                by ename ',10);
```

```
Scsr = &ora_open (Slda, insert into dept values                (:1,:2,:3)');
```

La última sentencia es usada para la sustitución de variables.

&ora_bind(Scsr, Svar,...)

Si una sentencia SQL contiene variables, &ora_bind es usado para asignar valores a estas. Esta función toma un identificador de sentencia (obtenido por la función &ora_open()) como su primer parámetro, seguido por cuantos parámetros sean requeridos por esta sentencia.

Ejemplo:

```
&ora_bind (Scsr,50 , 'management', 'Paris');
```

%fields = &ora_fetch (Scsr)

@data = &ora_fetch(Scsr)

La función &ora_fetch() es usada en conjunción con una sentencia SELECT para obtener información de la base de datos. Esta función toma un parámetro mandatorio, un identificador de sentencia obtenido de &ora_open ().

Usado en un contexto escalar, la función retorna el número de campos retornados por la búsqueda pero la información no es aun utilizada. Este puede ser de mucha utilidad en un programa el cual permite al usuario entrar una sentencia interactivamente.

Usado en un contexto de arreglo, el valor retornado es un arreglo conteniendo la información, un elemento por campo. Si cualquier campos fuera NULL (desde el punto de vista ORACLE) el dato correspondiente sería indefinido.

&ora_close (Scsr)

si una sentencia SQL no es requerida, entonces el identificador de la sentencia debe ser liberado. Este es hecho por la llamada a la función &ora_close() con el identificador de la sentencia como su único parámetro.

&ora_do (Slda, Ssentencia)

No todas las sentencia SQL retornan datos o contienen sustitución de variables. En estos casos la función &ora_do() puede ser usada como una alternativa a &ora_open y &ora_close(). Esta función toma dos parámetros, un identificador de login y la sentencia ejecutada.

Esta función es más o menos equivalente a &ora_close(&ora_open(Slda,Ssentencia)).

&ora_do() retorna un valor indefinido si un error ocurre. De otra manera, este retorna el número de registros afectados por el comando o el string 'OK' si el comando fue exitoso pero no modificó registros.

&ora_logoff (Slda)

Cuando el programa no necesita acceder más a la base de datos, el identificador de login debería ser liberado usando la función &ora_logoff(Slda).

Funciones adicionales estas disponibles y son : &ora_titles(), &ora_lengths(), &ora_types(), &ora_autocommit(), &ora_commit(), &ora_rollback() y ora_version().

@titles= &ora_titles (Scsr)

Un programa puede determinar los títulos de los campos de una búsqueda llamando a &ora_titles(). Esta función toma un parámetro mandatorio, un identificador de sentencia indicando la búsqueda para la cual los títulos son requeridos. Los títulos son retornados como un arreglo de strings, uno por cada columna.

@lengths = &ora_lengths (Scsr)

Un programa puede determinar la longitud de cada uno de los campos retornados por una búsqueda llamando a `&ora_lengths()`. Esta función toma un parámetro que es el identificador de la sentencia SQL. Las longitudes son retornadas en un arreglo de enteros, uno por cada campo.

`@types = @ora_types($csr)`

Un programa puede determinar el tipo de cada uno de los campos retornados por una búsqueda llamando a la función `&ora_type()`. Esta función toma un parámetro que es el identificador de la sentencia. Los tipos son retornados como un arreglo, uno por cada campo.

`@ora_autocommit($lda, $on_or_off)`

El modo de Autocommit en el cual cada transacción es 'commiteada' (confirmada) inmediatamente, sin esperar un commit explícito puede ser habilitado o deshabilitado con la función `&ora_autocommit`. Esta función toma dos parámetros, un identificador de login y un valor verdadero o falso indicando cuando un autocommit va a ser habilitado.

Hay que aclarar que el autocommit puede ser seteado por login no por sentencia. Si se necesita un autocommit por sentencia, se debería hacer múltiples llamadas a `&ora_login` y usar un identificador separado por cada sentencia.

`&ora_commit($lda)`

`&ora_rollback($lda)`

Modificaciones a la base pueden hacerse commit o rollback usando `&ora_commit` y `&ora_rollback`. Estas funciones toman un parámetro que es el identificador del login.

`&ora_version()`

La función `&ora_version()` imprime el número de la versión y la información concerniente a Oraperl.

VARIABLES

Seis variables especiales manejan el ORAPERL que son `$ora_cache`, `$ora_long`, `$ora_trunc`, `$ora_errno`, `$ora_errstr` y `$ora_verno`.

Estas variables son usadas para manejar el comportamiento del ORAPERL bajo ciertas circunstancias.

Sora_cache

La variable *Sora_cache* determina el medida del cache usado por la función *&ora_open()* para las sentencias *SELECT* si una medida de cache no es dada.

Esto es inicializado con el valor reportado por el *&ora_version()* pero puede ser seteado dentro de un programa para a todas las llamadas subsecuentes a *&ora_open()*. Cursores los cuales están todavía abiertos no son afectados.

Sora_long

Normalmente, *Oraperl* interroga a la base para determinar la longitud de cada campo y setea el espacio del buffer. Cuando *&ora_open* determina que un campo es de un tipo *LONG*, esta da el espacio indicado por la variable *Sora_long*.

Sora_trunc

Por lo que *Oraperl* no puede determinar exactamente la longitud máxima de un campo tipo *LONG*, es posible que el ancho indicado por *Sora_long* no es suficiente para almacenar la información apuntada. En cuyo caso, el segundo parámetro de *&ora_fetch()* indica cuando la truncación debería permitirse o debería provocar un error. Si este segundo parámetro no es especificado entonces *Sora_trunc* es utilizado como un default.

VARIABLES DE STATUS.

Estas variables reportan información acerca de condiciones de error o acerca del *Oraperl*. Ellas pueden solamente ser leídas; un error fatal ocurre si un programa intenta cambiarlas.

Sora_errno

Sora_errno contiene el código de error Oracle provocado por la última llamada a alguna función.

Hay dos casos de particular interés concerniente a *&ora_fetch()*. Si un campo tipo *LONG* o *LONGRAW* es truncado (y la truncación es permitida) entonces *&ora_fetch()* se completará exitosamente pero *Sora_errno* será seteada en 1406 para indicar la truncación. Cuando *&ora_fetch()* falla, *Sora_errstr* deberá setearse a cero.

Sora_errstr

La variable *Sora_errstr* contiene el mensaje de error de oracle correspondiente al actual valor de *Sora_errno*.

Sora_verno

La variable Sora_verno contiene el número de versión de Oraperl en la forma v.ppp.

GOPHERSQL

En la Universidad de Minnesota se ha desarrollado el software que permite al usuario Gopher acceder a datos en una base de datos SQL. Esta pieza de software es llamada un gateway. En general un gateway translada las operaciones y datos de un sistema en operaciones soportadas por un sistema diferente e incompatible.

El gateway translada operación Gopher en sentencias SQL y los resultados son trasladados en datos para el gopher. El método gateway de acceso a una base de datos simplifica las operaciones.

La arquitectura cliente-servidor usada por el Protocolo Gopher de Internet. Ha sido descrito como "brutalmente" simple. Y se debe en gran parte al arreglo de la información que se mantiene en sistemas de archivos jerárquicos pero gran parte de la información almacenada no se mantiene en un sistema de archivos, gran parte se almacena en bases de datos especialmente relacionales. Las bases de datos son mucho mejores que los archivos por un gran número de razones entre ellas la consistencia de los datos y sus índices. Para manejar este tipo de datos se ha desarrollado un gateway que traslada los requerimientos del Gopher en sentencias SQL y los resultados SQL en datos para el Gopher. Esta gateway permite al usuario del Gopher mirar los datos dentro de las tablas SQL usando búsquedas. Este simplifica las operaciones en la base.

CARACTERISTICAS DEL GATEWAY SQL

El gateway SQL permite a personas usar un Cliente Gopher para acceder a los datos contenidos en una base de datos SQL sin tener que conocer SQL. Los clientes pueden ser usados con el gateway, y no se necesitan modificaciones en ellos.

El SQL gateway acepta requerimientos y translada estos en sentencias SQL que son pasadas via TCP a bases de datos tanto Sybase o Oracle.

El gateway SQL permite al Cliente Gopher :

- * Ver las tablas de una base de datos desde alguna opción del gopher.*
- * Ver las columnas de una tablas determinada.*
- * Ver el contenido de una columna.*

- * Ver como muchos registros son resultado de una búsqueda antes viendo estos registros.
- * Ver registros con formato texto.
- * Ver / importar registros con valores de tabulación.
- * Añadir registros a una tabla.
- * Buscar en una tabla llenando una form del Gopher.

El administrador tiene el control sobre la configuración.

SEMANTICA DE UN GATEWAY , O COMO ESTE TRABAJA

Un gateway es un cosa muy simple, Este translada comandos y datos de un formato a otro. El gateway SQL translada Operaciones Gopher en sentencias SQL, y esto nos ayuda a no tener que implementar todas las facilidades de una base de datos que resultaría muy difícil (Esto podría ser posible con un directorio muy grande).

El protocolo gopher es una herramienta para obtener información que esta basada en el modelo cliente-servidor. Este usa tres transacciones básicas.

- Listado de Directorios.
- Extracción de Archivos
- Búsqueda y retorno de una lista de directorios.

Estas directivas son muy poderosas, sobre 1500 lugares en el mundo ahora usan el protocolo gopher.

SQL es un lenguaje muy standard para las bases de datos . Algunas de las operaciones más comunes soportadas por SQL son:

- * "SELECT" seleccionar registros de una tabla.
- * "INSERT" insertar registros a una tabla.
- * Búsqueda en múltiples tablas para información.
- * Ordenamiento de resultados.
- * Agrupamiento de resultados.
- * Creación borrado de tablas o vistas.
- * Calculos en la información.

Muchas de las bases de datos soportan el concepto de el Diccionario de datos. El diccionario de datos es una base que describe el contenido de otras bases, tablas y columnas. Diferentes propietarios tienen diferentes formatos de diccionario de datos, Esto causa problemas.

Los SQL gateway entienden un limitado número de comandos. Estos comandos actúan entre el Gopher y las sentencias SQL. El gateway entiende los siguientes comandos:

- Tomar un listado de todas las tablas.
- Tomar un listado de columnas en tablas específicas.
- Tomar un listado de distintos valores en una columna específica.
- Muestra registros dados en una búsqueda.
- Inserta un nuevo registro.

Es muy dificultoso setear el Gopher para una base de datos SQL. Para correr el gophersql se necesita Perl, Oraperl (como lo vimos anteriormente) además de utilizar rutinas especiales implementar características específicas.

5.2 INSTALACION DEL WAIS

El WAIS (Wide Area Information Server) se tomó en cuenta para que funcione en conjunto con el gopher debido a la gran aceptación que ha tenido en la comunidad Internet, debido a su portabilidad y su fácil acceso. Podemos ver que el WAIS es uno de los sistemas que a muy poco tiempo revolucionará la manera de transmitir el conocimiento. Este proyecto (WAIS) intenta hacer un "back-bone" para la distribución de la información.

EL WAIS es un conjunto de productos suministrados por muchos proveedores para ayudar a los usuarios finales a encontrar y obtener información sobre redes de computadoras. Thinking Machines, Apple Computer y Dow Jones inicialmente implementaron un sistema para sus ejecutivos de negocios. Estos productos se volvieron ampliamente disponibles para y de varias compañías por lo que se volvieron de fácil acceso y por ello el relativo éxito que han tenido hasta hoy.

La persona que es nueva en el WAIS se hará las siguientes preguntas:

QUE HACE EL WAIS?

Usuarios en diferentes plataformas pueden acceder a información de personal, compañías y marketing por medio una interface final. La información puede ser cualquier cosa: texto, gráficos, voz y documentos formateados. Desde que se usa un protocolo, la información puede ser almacenada en cualquier parte y en cualquier tipo de máquina. Cualquiera puede utilizar este sistema ya que se hacen preguntas de una forma natural para encontrar documentos relevantes. Los Documentos relevantes pueden ser devueltos al servidor para mejorar la búsqueda. Esto evita complicados lenguajes de búsqueda. Búsquedas exitosas pueden ser automáticamente diseñadas para alertar cuando nueva información se encuentra disponible.

COMO EL WAIS TRABAJA?

El servidor toma una pregunta del usuarios y hace lo mejor para encontrar documentos relevantes. Los servidores no entienden la pregunta en lenguaje natural, preferiblemente ellos tratan de encontrar documentos que contienen estas palabras y frases. La interface del usuario (CLIENTES) hablan con los servidores usando un a extensión del protocolo Z39.50. Usando un standard público permite a los proveedores competir uno con otro , mientras no tienen que preocuparse por el protocolo.

Los problemas que están siendo manejados en el diseño de este sistema incluyen la interface con el usuario, la unión de información de muchas fuentes, encontrar fuentes de información lo suficientemente buenas, y formar un armazón para la rápida proliferación de los servidores de información.

Un protocolo abierto para conectar las interfaces de usuario en las estaciones y servidores es crítico para la expansión de los servidores de información. el éxito de este sistema radica en la "masa crítica" de usuarios y servidores.

La idea principal para el WAIS es que los servicios de información deberían ser fácilmente y libremente distribuidos.

Para analizar mas a fondo este proyecto del WAIS tenemos que ver que tres son los elementos claves para este tipo de sistema de información:

- 1.- Las estaciones de trabajo.
- 2.- Los servidores de información.
- 3.- El protocolo a usarse.

1.- Hablemos sobre el rol de las estaciones de trabajo en el WAIS.

En la actualidad las estaciones de trabajo se han desarrollado tanto para ser computadores sofisticados que pueden almacenar cientos de libros de información, multiprocesos y comunicarse sobre una variedad de redes. las avanzadas capacidades de la estaciones de trabajo son usadas para encontrar la información apropiada para el usuario contactando, escudriñando y negociando con los servidores de información. La explosión de la información disponible puede cambiar la manera en que nosotros utilizamos la computadoras, y puede resultar dificultoso la manera de manejar esta información. El WAIS propone encontrar información con un mecanismos llamado "Navegación por contenido".

A.- Acceso a documentos con Navegacion por Contenido.

Actualmente, la manera común de encontrar un documento es utilizando un utilitario. Estos árboles estructurados (Sistema de Archivos) requieren que el usuario recuerde donde ha puesto cada archivo. Este método trabaja cuando el usuario está familiarizado con la organización de archivos. Actualmente el número de potenciales archivos se incrementa a su vez que el espacio de disco se vuelve más barato y las redes permiten accesos remotos. En un punto el número de archivos se vuelve muy grande, será casi imposible que el usuario pueda recordar la ubicación de los archivos.

Otra técnica utilizada actualmente es la de hacer documentos con enlaces de hipertexto, que ayuda a los usuarios a moverse a través de gran cantidad de información. Los sistemas de hipertexto permite al autor proveer de caminos a través de los documentos. Los enlaces de hipertexto dan al autor otra herramienta para guiar al usuario y aumenta la capacidades del sistema de archivos.

Una técnica diferente que podría permitir el acceso a una larga colección de documentos basados en el contenido de documentos puede ser llamada "Navegación por contenido". Con esta herramienta, documentos son accedidos por medio de una pregunta en Inglés. Una línea o un encabezado, podría describir posibles documentos que contesten esta pregunta. Estos documentos podrían ser vistos o usados, además que cada documento podría ser calificado de que tan bien contesta a la pregunta y los documentos que mas altamente calificados se encuentren entonces estos serían presentados al usuario. Pero actualmente el lenguaje natural en un cien por ciento es imposible, por lo que un método para calificar al documento sería el contar el número de palabras que coinciden entre la pregunta y el documento. De lo que podemos concluir que los documentos buscados pueden ser encontrados por el "navegador" usando un conjunto de palabras que sean patrones de búsqueda y que como resultado devuelve los documentos ordenado por el número de palabras que coincidieron con la pregunta.

3.- Carpetas Dinámicas encuentran información para el usuario.

La navegación por contenido toma una pregunta y retorna una lista ordenada de los documentos posibles. El resultado de una pregunta pueden no contener una copia del documento, solamente una referencia o puntero a un documento. Esta pregunta y respuesta puede ser grabada como una carpeta de archivos. Esta capacidad de almacenar la búsqueda se convierte en importante cuando algunas de las preguntas toman tiempo de contestar porque los datos pueden ser difíciles de acceder. Estas carpetas dinámicas tienen la característica de tener una pregunta asociada.

La interface de usuario debería proveer la pregunta asociada a la carpeta dinámica.

2.- Usando servidores de información

Los servidores de información son los que contestan las preguntas. Un servidor puede ser local o remoto, tiene una base de datos que pueda buscar y retirar la información. Estos servidores pueden ser accedidos fácilmente por las estaciones de trabajos sobre una red con un protocolo estandar usando una herramienta de Navegación por contenido para ejecutar búsquedas y las carpetas dinámicas para mantener y coordinar las respuestas.

3.- Hablemos un poco del protocolo.

La implementación inicial proveya un protocolo para la base de datos de DOWQUEST un servicio para la obtención de información provisto por DOW JONES NEWS. La interfaces de la estaciones de trabajo fueron implementadas en Macintosh como parte del proyecto WAIS (Wide Area Information Server). La intención es de proveer de sofisticadas y expandibles interfaces computador - computador para futuras bases de datos.

Este protocolo es basado en el Z39.50 - 1988 ("El estandar) Information Retrieval Service Definitions and Protocol Specification for Library Applications.

El estandar especifica una definición de capa de aplicación y la especificación de protocolo para Obtención de Información. El protocolo para la obtención de información permite que una aplicación de una computadora pueda hacer búsquedas en la base de otra computadora. El protocolo especifica los procedimientos y estructuras para el envío de requerimientos (incluyendo la sintá de la búsqueda) , los requerimientos para la transmisión de registros de la base de datos localizado por la búsqueda , las respuestas al requerimientos, el control de acceso.

5.3 INSTALACION DEL CCSO.

El nombre completo es CCSO Nameserver es una especie de "Libro telefónico". Este puede mantener una relativamente pequeña cantidad de información acerca de un gran número de personas o cosas y provee un rápido acceso para esta información en INTERNET. Nació en la Universidad de Illinois y en este se mantiene la información de la "PAGINAS BLANCAS" de los estudiante y profesores.

A diferencia de un directorio con archivos la información en el CCSO Nameserver es dinámica. Esta puede ser actualizada en cualquier tiempo, de cualquier computadora en INTERNET capaz de correr el programa cliente PH.

A continuación se examinará tres aspectos de el CCSO :

- Capacidades.*
- Implementación*
- Ventajas.*

LAS CAPACIDADES: La Base de datos.

El CCSO Nameserver maneja una base de datos que consiste de muchas entradas individuales. Cada Entrada contiene una o más campos, cada campo consiste de uno o mas caracteres ascii (incluyen tab y un CR). Cada campo esta asociada con una descripción de campo particular. Una descripción del campo incluye un nombre, una longitud máxima para los campos que este describe, y ciertas propiedades que determina como el campo es usado.

No hay esencialmente limites intrinsecos en la medida de la base , en el número de entradas , número de descripciones del campo, número de campos por entrada o medida de campos.

Ciertos campos en la base son indexados. Palabras de estos campos pueden ser usadas como llaves para entradas seleccionadas en la base. Palabras de cualquier campo pueden ser usados para pulir la selección hecha por las llaves de los campos. El esquema de indexamiento usado es el de "doble-hashing", que resultan en búsquedas muy rápidas. La tabla de Hash es también indexada para facilitar el encuentro de patrones en ella.

CAPACIDADES: El servidor

La base de datos reside enteramente en una computadora y es manejada por un programa servidor , qi (query interpreter). Múltiples instancias de qi pueden estar ejecutándose en forma simultánea y el acceso a la base de datos es controlada por

locks. Cualquier número de procesos pueden leer la base. Pero si un proceso este escribiendo en la base , en cuyo caso todos procesos deberán esperar para que proceso complete su trabajo.

Qi usa un esquema de réplica de comandos como el usado por el FTP. Este acepta comandos de su entrada estandard, y escribe la réplica en el estandard output. Ambos comandos son replicados y captados en "netascii"; líneas consisten de caracteres impresos terminados con un caracter de newline (ASCII 10) o un CR y un newline (ASCII 13 ASCII10). adicionalmente el backslash "\" es usado como escape ciertos caracteres, como en el lenguaje de programación C.

Comandos consisten de palabras seguidos a veces de uno o más argumentos o palabras. Los comandos incluyen: **query** para búsqueda en la base de datos; **change** para cambiar los campos en las entradas; **add** para aumentar nuevas entradas. Las contestaciones consisten de códigos numéricos que están de -599 hasta 599, y texto adicional. Los códigos numéricos pueden indicar una operación en proceso (100-199), éxito (200-299), un requerimiento (300-399), etc.

El comportamiento del *qi* puede ser modificado para usar ciertas opciones acceso por el comando *set*. El número de opciones disponible es pequeña; las opciones mas importantes son **echo**, la cual causa que *qi* imprima comando en su output antes de ejecutar a ellos, y **limit**, el cual permite al usuario el especificar un máximo número de entradas a las cuales un commando puede aplicar.

Qi opera en tres modos diferentes: *anonymous*, *login*, y *hero*. Cada modo es mas liberal en la operación y consecuentemente mas dificultoso para acceder.

El modo *anonymous* es usado para hacer búsqueda de información publica y para otros pocos propósitos. En modo *anonymous*, hay un máximo número de entradas que pueden ser vistas con un comando; el propósito de esta limitación es el de desalentar el uso de el *qi* para la preparación de listas de discusión. El modo *Anonymous* es usado para mas búsquedas del Nameserver.

Para entrar en el modo *login*, un usuario debería identificarse a el mismo como el dueño de un Nameserver particular dando un alias (*login name*) y un password. En adición a las capacidades del modo *anonymous* el modo *login* permite al usuario logonearse para cambiar campos en el Nameserver.

El modo *Hero* es accesado tambien ingresando al modo *login* como un Nameserver "hero" (*superuser*) o corriendo el *qi* directamente de un terminal, preferiblemente que sobre una red. En este modo , todos los limites artificiales son movidos; el *hero* puede cambiar cualquier campo en cualquier entrada en la base de datos, el modo *Hero* es usado para propósitos administrativos.

CAPACIDADES : *Busquedas.*

Una de la principales funciones del Nameserver es el de contestar queries. Un nameserver query consiste de cinco elementos: El comando query, valores para uno o mas campos indexados, valores para cero o mas campos no indexados, opcionalmente el comando retornado, opcionalmente una lista de campos a imprimir de la entradas seleccionadas. Un par de ejemplos clarificara esto.

Primero , una busqueda; los argumentos son interpretados como requerimientos para palabras de el nombre o sobrenombre, ambos son campos indexados.

```
qi> query steven dorner
```

```
-200:1:    alias:gopherda
-200:1:    name:jara william
-200:1:    email:wajara@espol.edu.ec
-200:1:    phone:(563-4) 269-247
-200:1:    address: Garzota 2da etapa
-200:1:           : Mz. 51 villa 10
-200:1:    title: Ing. Computacion
-200:1:    nickname:william
-200:1:    hours:8-4 fines semana
200:ok.
```

Aqui es un ejemplo que usa todos los cinco elementos. El campo departamento no es indexado.

CAPACIDADES: *El cliente*

Usualmente el Nameserver es accesado via el programa cliente ph. Este programa hace una conexion con la computadora que mantiene la base de datos. Este provee asistencia al usuario de el Nameserver; este formula queries, formatea al respuestas del Nameserver, y provee otras características utiles para el usuario.

El ph opera en dos modos: linea de comando e interactivo. En el modo de comando de linea, ph forma un query para el nameserver de los argumentos datos a el, manda este al qi , y muestra el resultado y sale. En el modo interactivo, el ph lee comando del usuario , manda estos al qi, y imprime las respuestas del qi. respuestas son automaticamente a un programa que las ordena y mantiene las respuestas.. Algunos de los comandos dados al ph son expandidos con mas comandos del qi. Por ejemplo el comando edit del ph primero pregunta al qi por el valor del campo deseado pune el valor en un editor donde el usuario edita el campo, y entonces utiliza el comando change para cambiar el campo con el valor deseado.

IMPLEMENTACION: La fuente.

El nameserver esta escrito en C (una pequeña parte esta escrita en lex Lexical Analyzer Generator), y corre en sistemas Unix. El cliente ph puede ser corrido en sistema 4.[23]BSD . Una version de ph existe para VMS, DOS, MAC y una version limitada existe para VM/CMS.

IMPLEMENTACION: La base de datos.

La base de datos es mantenida en seis archivos con la extension .dir, .dov, .idx, .iov, .seq, y .bdb. las extensiones .dir y .dov contiene los datos. Los .idx y .ipv contienen la tabla de hash, con punteros a los archivos de datos. El archivo .seq contiene todas las palabras de la tabla hash; esta es usada para el maching de los queries. El archivo .bdb acelera la busqueda de el archivo .seq.

El archivo .dir consiste de un encabezdo y una registro de longitud fija por cada entrada de la base. Si hay muchos datos para un registro, el restante es ubicado en el archivo .dov. El archivo .dov tambien consiste de archivos fijos y si algunos no es suficiente el restante puede ser ubicado en mas registros .dov. Mas una entrada es realmente una lista enlazada de registros de longitud fija y no limite en la longitud.

Cada entrada comienza con alguna informacion de longitud fija, seguida por los campos que hacen esta informacion. Cada campo tiene un string nulo tipo ASCII. Un campo comienza con un string ASCII que tiene el ID de la descripcion del campo y dos puntos (:).El campo de datos seguido y entonces el terminador nulo (ASCII 0).

El archivo .idx esta hecho de un numero fijo de registros de longitud fija. Cada registro que esta en uso contiene una palabra de un campos indexado, y un conjunto de punteros para los registros .dir que contiene la palabra en un campos indexado. Cualquier Overflow en el archivo .idx es manejado como un overflow en el archivo .dir; el exceso de punteros son puestos en uno o más registros de longitud fija en el archivo .iov. Las palabras son indexadas por una función de hash, si la localidad seleccionada no esta vacia pero no contiene la palabra deseada, la función de hash es iterada, hasta un limite y al llegar a este limite el indice falla.

El archivo .seq usa registros de longitud fija para mantener una lista ordenada de todas las palabras de la tabla hash (.idx y .iov) . Cada hoja contiene más de cuatro palabras y un puntero a la siguiente hoja en un orden alfabético. Con cada palabra esta almacenado un puntero dentro de la tabla hash donde la palabra es encontrada.

El archivo .bdx tiene registro (llamados nodos) el cual contiene llave de cuatro bytes, y dos punteros; uno para el nodo previo en orden alfabético y uno al siguiente nodo en orden alfabético.

IMPLEMENTACION: Búsquedas.

Una búsqueda es primero separado en sus partes componentes. Entonces, los argumentos seleccionados de una búsqueda son chequeados contra los argumentos indexados. El argumento más largo es buscado uno por uno en el tabla hash, coincidiendo caracteres, una búsqueda es hecha a través de los archivos .bdx y .seq.

IMPLEMENTACION: Descripciones de campos

Las descripciones de campo son mantenidos en un archivo que qi lee cada vez que es corrido. Este archivo consiste de líneas describiendo cada campo, en ASCII, con separaciones (:) los elementos de un línea. Primero viene el id number del campo, entonces el nombre del campo y su longitud máxima. Finalmente, hay una lista separada por (:) de las propiedades de los campos.

Desde que el archivo es leído cada vez que el qi comienza, las líneas pueden ser añadidas para definir nuevos campos. Todas las subsecuentes invocaciones de qi serán capaces de reconocer y usar los campos.

VENTAJAS: Velocidad

Para chequear la velocidad se hicieron pruebas con 300 palabras de diferentes partes del índice; y busco cada una usando qi. Qi encontró 396 entradas en 78 segundos; que es casi 1/4 segundo por cada uno. Usando llaves de 4 letras y wildcard el resto, qi encontró 9213 entradas en 460 segundos, cerca de 11/2 por cada uno.

En el uso en la red, La respuesta es lenta, por que el cliente tiene que establecer una conexión con el host que tiene la base de datos. Buscando 100 palabras indexadas invocaciones distintas de ph tomo 109 segundo 1 segundo por cada uno;

CAPITULO 6

EL CLIENTE DOS

REQUERIMIENTOS

El cliente gopher que se va a utilizar es el que pertenece al paquete denominado **MENUNET** que contiene tanto POP mail, FTP, TELNET, NEWS y GOPHER. Este paquete corre en versiones de DOS mayores que el 3.3. Este paquete corre en gran rango de IBM PC y compatibles. Incluyendo los que contienen monitor monocromático.

Un mouse es de mucha ayuda pero es opcional. además necesitamos una conexión **SLIP** que a continuación detallaremos.

SLIP (SERIAL LINE INTERNET PROTOCOL), es una manera de establecer una conexión de red através de un modem. Esto significa que programas como POPMail, Gopher y NCSA telnet operarán sobre una línea telefónica como si estuviesen conectado a la red por una tarjeta de red.

EJECUCION DEL PHONE.EXE

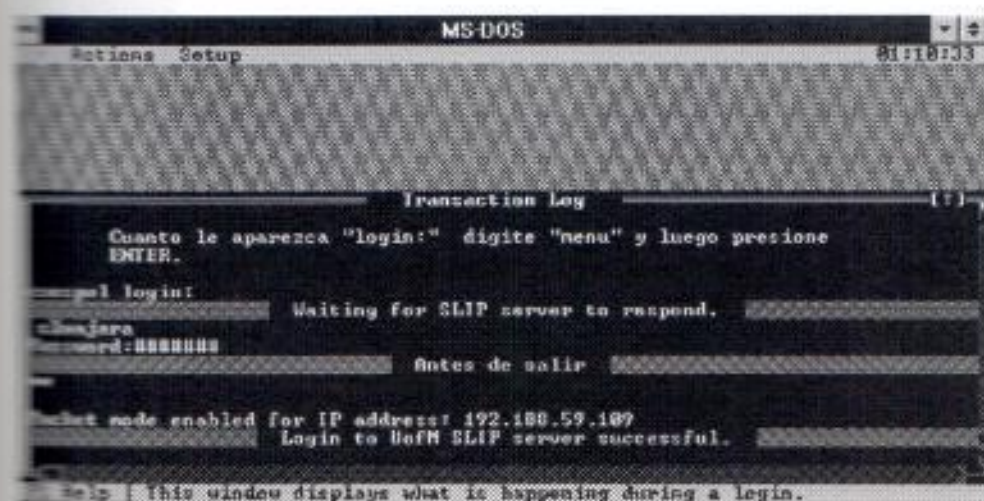


FIG. No 8

Desarrolladores de aplicaciones de red a menudo diseñan su software para usar una Especificación de packet Driver. Un packet driver es una pequeña pieza de software que maneja comunicaciones entre tu computadora y el mundo exterior. Un packet driver comunica con una tarjeta ethernet, una tarjeta Token Ring, un puerto serial, o alguna otra pieza de hardware que esta conectada a la red. Este provee de una "gama" que permite a la aplicación de red comunicarse apropiadamente con la red. Aplicaciones escritas para la especificación de packet driver no necesitan conocer

como comunicarse con todo adaptador posible de red; Ellos solo necesitan conocer como direccionar el packet driver.

El packet driver que utilizamos se llama UMSLIP esta escrito especialmente para comunicaciones SLIP. y usa el puerto serial en vez de una tarjeta de red para hacer la conexión con la red.

Las aplicaciones de red como POPMail y GOPHER hablan solamente al la interfaces del packet driver para coger su información y mandan la información por el.

Para correr el gopher cliente tuvimos que traer dos paquetes el sliparc.exe

CONSULTA DEL MINUET EN EL CORREO ELECTRONICO

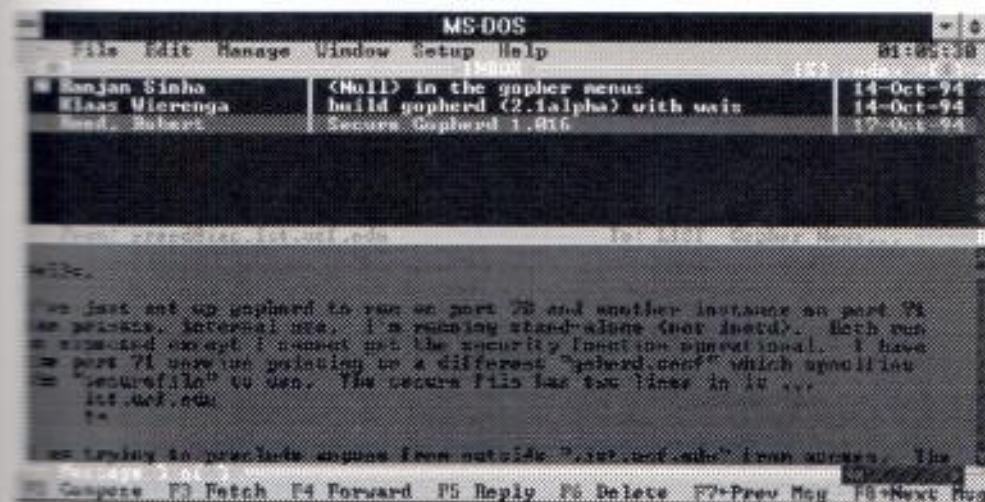


FIG. No 9

y el minuarc.exe que contienen el primero los archivos para poder establecer una conexión slip y el segundo que contiene el paquete integrado de la aplicación tanto con gopher, pop mail, ftp, telnet y News.

El archivo sliparc.exe es un executable que desempaqueta los archivos slip.bat, umslip.com, termin.com, phone.exe.

slip.bat un batch que facilita la tarea para comunicarse con la red.

UMSLIP.COM es el packet driver el cual transforma señales de la red en señales para una línea analoga.

TERMIN.COM el programa *TERMIN.COM* para remover *UMSLIP* de memoria.

PHONE.EXE El programa el cual marca el número telefónico y establece una conexión *SLIP*.

Para hacer la conexión *slip* se tuvo que modificar los parámetros predefinidos en el archivo *phone.cmd* que a continuación mostramos:

```
#####
#
# Modem and Host Scripts -- U of M version 1.99
#
# U of M basic scripts
#
#####
#
# Modem scripts
#
# For each modem type xxx, supply script:
#
# Modem.xxx.Dial      Dials the phone, waits for connection
# Modem.xxx.Hangup    Gets the modem to hang up the phone
# Modem.xxx.Status    Returns the modem's status (-1,0,1-?,no cd,cd)
#
#####
#
#
# Procedure Modem.ADI.Dial
# Message 'Waking up the ADI-100'
# DTR
# TimeOut 5      'Your ADI-100 did not respond.'
# Expect 'IRECTORY'
# Expect '>'
# Message "Changing terminal type to 19200 baud"
# Send 'T<'
# Expect 'TYPE'   'Did not get "TERMINAL TYPE" prompt from ADI-100'
# Expect '>'
# Send '19<'
# Expect 'CHANGE(T)' 'Did not get "CHANGE" prompt from ADI-100'
# Message "Dialing the SLIP server"
# Send 'D<'
# Expect 'NUMBER'  'Did not get "NUMBER" prompt from ADI-100'
```

```

Expect '>'
Send '%n<'
TimeOut 30
Reject 'BUSY' 'ADI-100 said server is busy.'
Reject 'TIMEOUT' 'ADI-100 said connection timed out.'
#
Expect 'INITIATED' 'ADI-100 did not say "DATA CALL INITIATED".'
Serial 19200 8 None
Message "Waking up terminal server"
repeat < "User Access" 10 2
EndProcedure Modem.ADI.Dial
#
#
Procedure Modem.ADI.HangUp
DTR Off
EndProcedure Modem.ADI.HangUp
#
#
#####
#
#
procedure Modem.US-Robotics-Sport.Dial
Message 'Initializing Sportster'
Send 'AT &B1 &H1<'
Reject 'ERROR' 'Modem did not like initialization string.'
Expect 'OK' 'Modem did not respond with OK.'
jump Modem.Default.Dial
EndProcedure Modem.US-Robotics-Sport.Dial
#####
#
#
#####
#
#
procedure Modem.Comstation3.Dial
Message 'Initializing Comstation3 modem'
Send 'AT V1 X4 &K3 &W1<'
Reject 'ERROR' 'Modem did not like initialization string.'
Expect 'OK' 'Modem did not respond with OK.'
jump Modem.Default.Dial
EndProcedure Modem.Comstation3.Dial
#####
#
#
#####

```

```

procedure Modem.Zoom.Dial
Message 'Initializing Zoom modem'
Send 'AT V1 X4 S37=9 S38=0 N0 &D2 &W0<'
Reject 'ERROR' 'Modem did not like initialization string.'
Expect 'OK' 'Modem did not respond with OK.'
join Modem.Default.Dial
EndProcedure Modem.Zoom.Dial
+
+
#####
+
+
procedure Modem.MultiModem-v32.Dial
Message 'MultiModem v.32'
Send 'AT V1 X4 &C1 &E4 &E1 S13=0 SRI<'
Reject 'ERROR' 'Modem did not like initialization commands.'
Expect "OK" 'Modem did not say OK.'
join Modem.Default.Dial
EndProcedure Modem.MultiModem-v32.Dial
+
+
#####
+
+
procedure Modem.PowerUser-v1.5.Dial
Message 'PowerUser modem'
join Modem.Default.Dial
EndProcedure PowerUser-v1.5.Dial
+
+
+
procedure Modem.Hayes.Dial
Message 'Hayes newer modem'
Send 'AT V1 X4 &C1 &D2 &S0<'
TimeOut 5 'Your modem is not responding. try turning it off, then on.'
Reject "ERROR" 'Your modem said "Error" to the initialization string.'
Expect 'OK' 'Your modem did not say "OK".'
join Modem.Default.Dial
EndProcedure Modem.Hayes.Dial
+
+
procedure Modem.Hayes-Optima.Dial
Message 'Hayes Optima modem'
Send 'AT N1 V1 X4 &C1 &D2 &S0<'
TimeOut 5 'Your modem is not responding. try turning it off, then on.'
Reject "ERROR" 'Your modem said "Error" to the initialization string.'

```

```

Expect 'OK'      'Your modem did not say "OK".'
join Modem.Default.Dial
EndProcedure Modem.Hayes.Dial
#
#
#####
#
procedure Modem.Hayes-1200.Dial
Message 'Hayes 1200 baud modem'
join Modem.Default.Dial
EndProcedure Modem.Hayes-1200.Dial
#
#
#####
#
# Below are the default procedures that are done if there is
# no modem-specific procedure for these cases.
#
#
procedure Modem.Default.Dial
Message 'Waking up your modem...'
Send 'ATE1 H0 M1 Q0 V1 %i<'
Send 'AT&FE1LIV1&C1&D2S0=0S7=90<'

Timeout 5      'Your modem is not responding, try turning it off, then on.'
Reject "ERROR"  'Your modem said "Error" to the initialization string.'
Expect 'OK^M'   'Your modem did not say "OK".'
Wait 1
Message 'Dialing up the SLIP server...'
SendPT 'AT DP %n^M' 'AT DT %n^M'
Timeout 60     'No connection after 1 minute.'
Reject 'NO DIALTONE' 'Your modem can't get a dial tone, check phone line to
modem.'
Reject 'BUSY'   'The server is fully occupied.'
Reject 'NO CARRIER' 'The server modem is not responding. Try again.'
Expect 'CONNECT' 'Did not get the CONNECT response.'
Message 'SLIP server reached'
EndProcedure Modem.Default.Dial
#
#
#
Procedure Modem.Default.HangUp
Flush
Message "Hanging up phone."

```



```

TimeOut 5  "Your modem may be already hung up"
Wait 2
Send '+++ '
Wait 2
Send '<'
Send 'AT H0<'
Expect 'OK'
Message    'Hangup complete.'
EndProcedure Modem.Default.HangUp
=
=
#####
=
=
=   Host Scripts
=
=   For each host hhh, supply two scripts:
=
=   Host.hhh.Login      Logs in and goes into SLIP mode
=
=   Host.hhh.LogOut    Does any logout chores (often empty)
=
=
Procedure  Host.UofM.Login
Expect 'cspol login:'
TimeOut 60  'The U of M SLIP server is not responding.'
Message    'Waiting for SLIP server to respond.'
#Quiet ON
#TimeOut 5
#Message    'SLIP server is responding.'
#Message    'Sending your user name and password.'
#Quiet OFF
#Expect 'Username:'
#Expect 'cspol login:'
Send '%u<'
Expect 'Password:'
Private
Send '%p<'
#Wait 1
Reject 'Access denied' 'Your user name or password was not accepted.'
TimeOut 30  'SLIP server did not respond to your validation request.'
#Quiet ON

```

```

=iTimeout 10 'SLIP server did not respond to SLIP command.'
Grab MYIP
Message 'Login to UofM SLIP server successful.'
EndProcedure Host.UofM.Login
=
=
=
Procedure Host.UofM.LogOut
EndProcedure Host.UofM.LogOut
=
= End of Script file
=

```

despues de la conexión slip ya podemos desempaquetar el archivo minuarc.exe que contiene los archivos para la ejecución del MINUET.

Para setear el minuet tenemos que tener en cuenta la dirección internet dada por el slip y esta tenemos que añadirla a la configuración del sistema en configuración network.

CONSULTA DEL GOPHER POR MEDIO DEL MINUET

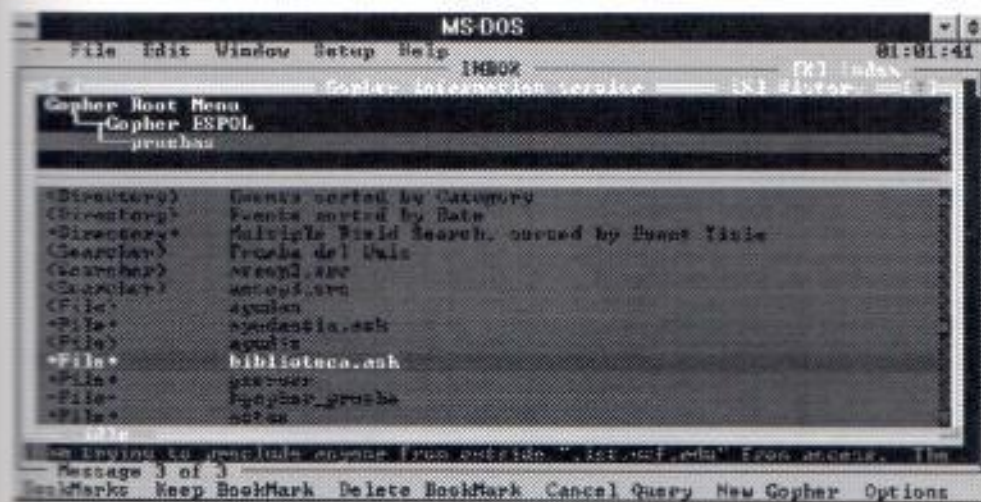


FIG. No 10

Con este seteo también tenemos que poner los servidores tanto gopher como pop mail y News.

SETEO DE LOS SERVIDORES DEL MINUET



FIG. No 11

SETEO DE LA DIRECCION IP EN MINUET

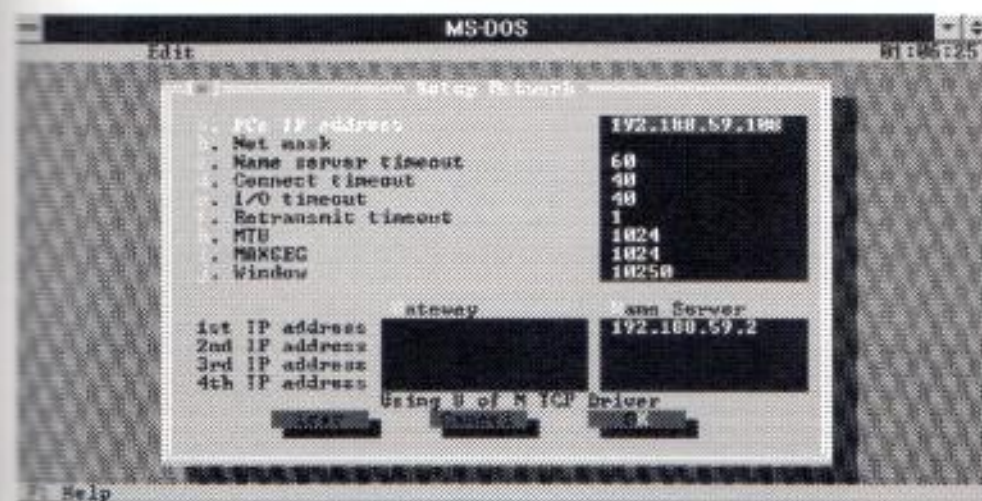
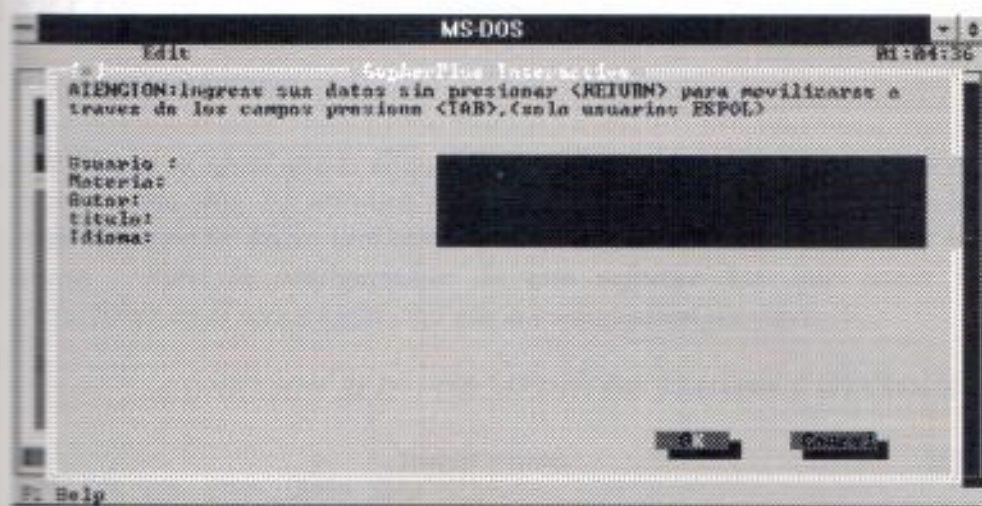


FIG. No 12

El gopher muestra sus archivos y los items textos con una interface muy parecida al unix, pero con la ventaja que hay dos ventanas una que nos muestra el item y otra que nos muestra la ubicacion en el árbol de la informacion.

CONSULTA DE BIBLIOTECA POR MEDIO DEL MINUET*FIG. No 13*

6.2 CLIENTE WINDOWS

REQUERIMIENTOS

Este cliente al igual que el cliente DOS necesita un software (packet driver) para la conexión via slip. El software utilizado es el WINSOCK que da muchas ventajas como la de poder hacer una conexión slip de manera muy sencilla por medio de los iconos . Para la configuración de este software hay que hacer las debidas correcciones en el script login.cmd que a continuación ponemos.

CONEXION SLIP POR MEDIO DE TRUMPET WINSOCK

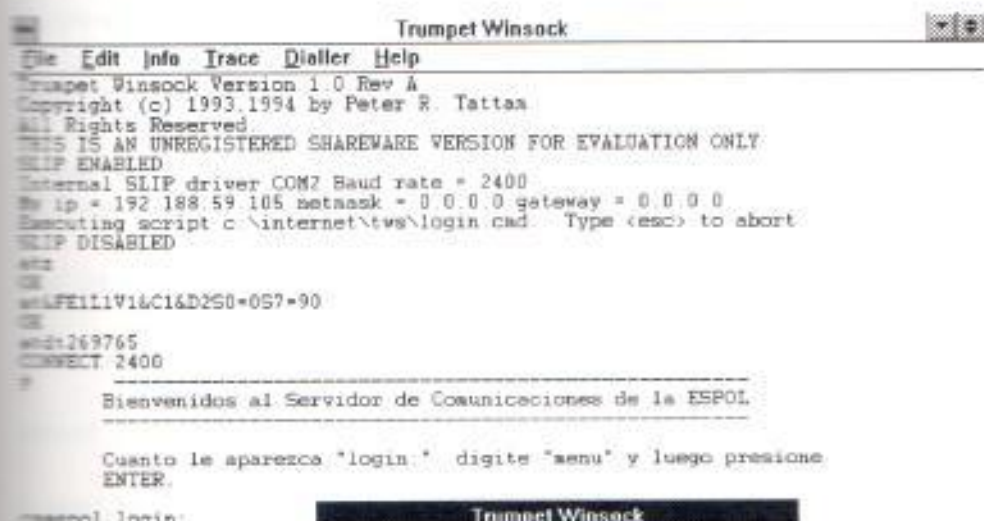


FIG. No 14

LOGIN.CMD

```

# initialize modem
#
output atz\13
input 10 OK^n
output at&FE11V1&C1&D2S0=0S7=90\13
input 10 OK^n

```

```
=
# set modem to indicate DCD
#
# output at&c1
# input 10 OK\n
#
# send phone number
#
output atdt269765\13
#
# my other number
#
#output atdt241644\13
#
# now we are connected.
#
input 30 CONNECT
#
# wait till it's safe to send because some modem's hang up
# if you transmit during the connection phase
#
wait 30 dcd
#
# now prod the terminal server
#
output \13
#
# wait for the username prompt
#
input 30 csespol login:
username Enter your username
#output \u\13
output slwajara\13
#
# and the password
#
input 30 Password:
password Enter your password
#output \p\13
output pM145a\13
#
# we are now logged in
#
# input 30 >
```

```
=
= see who on for informational reasons.
=
= output who\13
= input 30 >
=
= jump into slip mode
=
= output slip\13
=
= wait for the address string
=
input 30 Packet mode enabled for IP address:
=
= parse address
=
address 30
= input 30 \n
=
= we are now connected, logged in and in slip mode.
=
= display \n
= display Connected. Your IP address is \i.\n
=
= ping a well known host locally... our slip server won't work
= for a while
=
= exec pingw 192.188.59.4
=
= now we are finished.
=
```

GOPHER CLIENTE BAJO WINDOWS

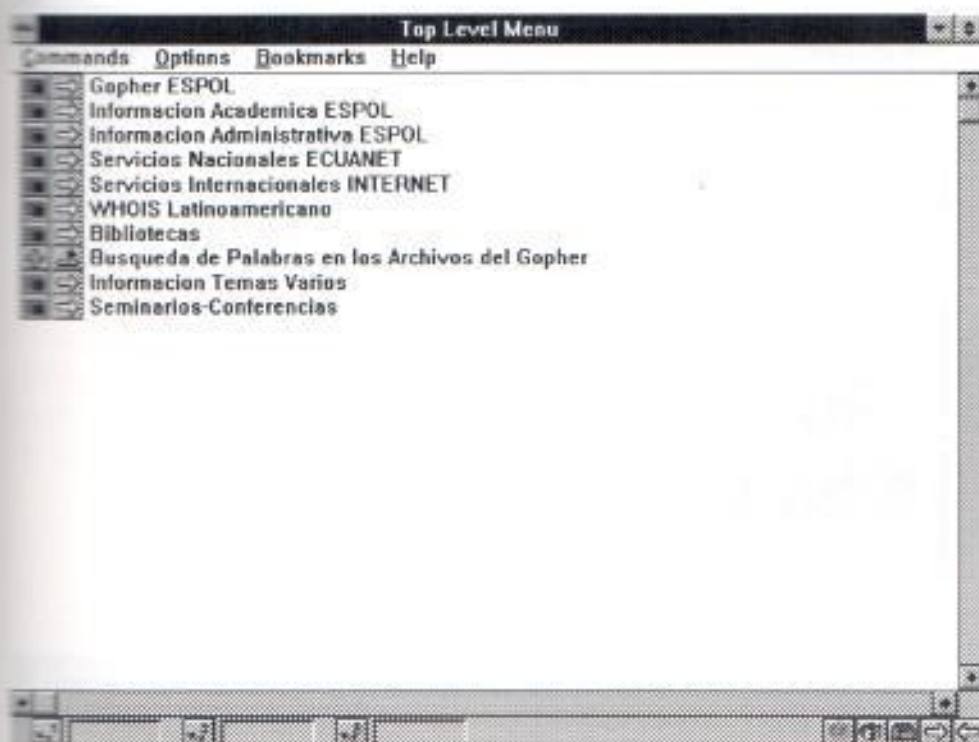


FIG. No. 16

VISTAS

El más útil atributo *gopher+* es el de las vistas. Con documentos normales de *gopher* (algunas veces llamadas vistas) el servidor *gopher* dice "Aquí está un documentos este es texto o una imagen". Pero no se conoce el tipo de texto o imagen que es, lo que nos provoca un gran problema ya que no sabemos que clase es. Con un ítem de *Gopher+* podemos decir exactamente que clase de ítem es, por ejemplo si este es un archivo Postscript o es un archivo ASCII. Si es una imagen puede ser gif o jpeg o algo más.

SETEO DEL GOPHER EN EL CLIENTE BAJO WINDOWS

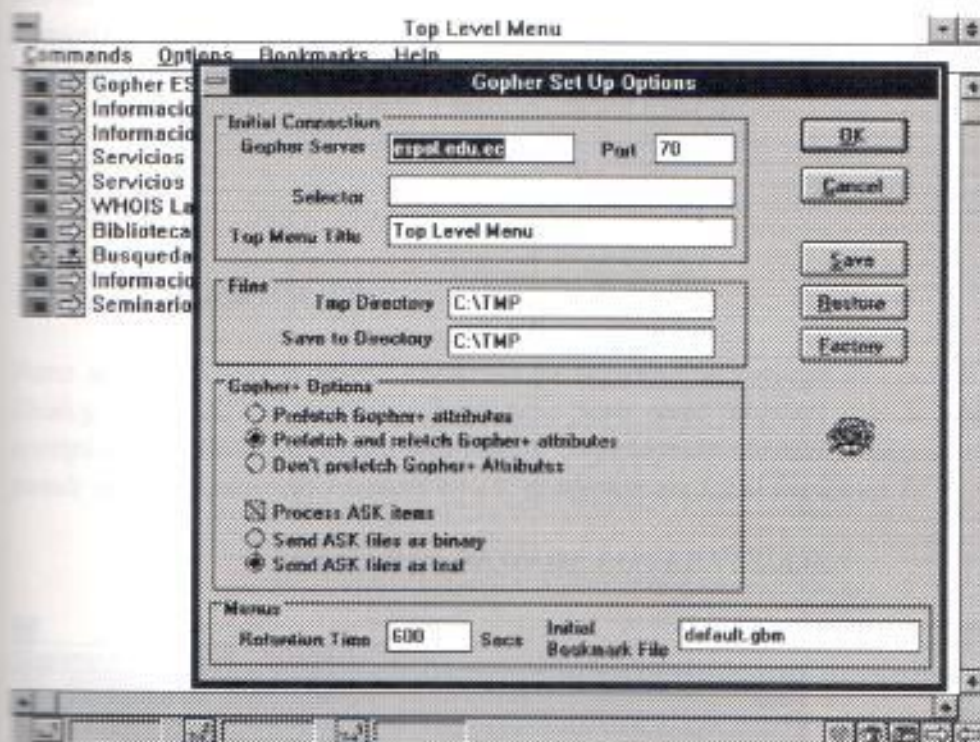



FIG. No. 17

También un ítem de gopher puede tener más de una vista por ejemplo este podría estar disponible en ASCII o Postscript y uno podría escoger el que quiere ver. Es más se podría ver información administrativa como quien escribió el documento, cuando fue la última actualización.

Cuando un ítem gopher+ tiene uno o más tipos de vistas asociados con él, se puede ver el símbolo  y se puede seleccionar cualquiera de los tipos de vista que uno quiera y también se puede ubicar sobre el ítem y dar doble click y se traerá la vista preferida por default.

El Hgopher selecciona cual vista se quiere ver usando el sistema de pesos, con este sistema se puede decir y escoger ítems que cumplan con:

- Siempre traiga Postscript si esta disponible.
- Siempre traer el postscript del host que esta comunicado con tarjeta de area local pero para otros computadores traer ASCII.
- Si hay que escoger entre JPEG y un GIF tomar un GIF aunque este sea más grande que el JPEG por más de 200 Kbytes.

-Si el documento está en FRANCÉS, traerlo en vez del que está en INGLÉS.

Varias cosas pueden tener un peso asociados a estos. Este peso es generalmente un número entero. Cuando el Hgoher tiene que traer una de muchas vistas. Este pesa cada tipo de vista y la que tenga el mayor peso es apuntada.

PESOS CON LENGUAJES.

Para seleccionar el lenguaje preferido se necesita de invocar la opción Language Dialog. se puede entonces setear un peso para cada lenguaje que se prefiera. Por ejemplo si se quiere ver textos en Francés, seguidos de Alemán y luego en Inglés, se puede setear el peso del Francés en 10, el alemán en 5 y el Inglés en 1..

SETEO DEL PESO DE LOS LENGUAJES

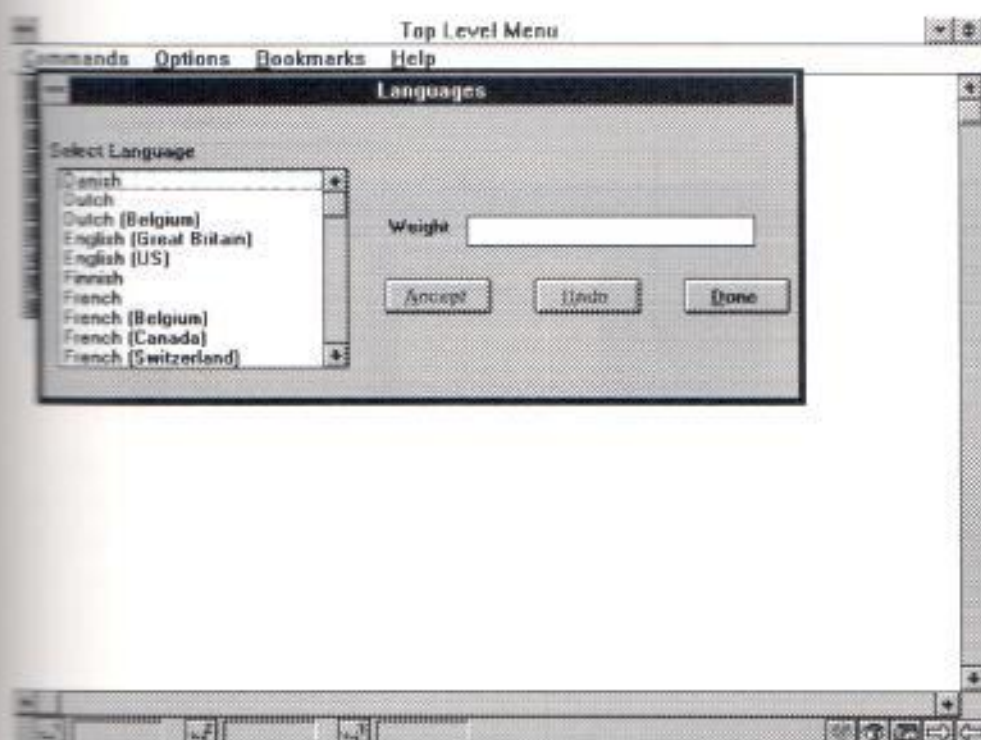


FIG. No. 18

PESOS EN TIPOS DE VISTA

Al entrar a la opción *Viewer Dialog* se puede setear dos pesos para una vista particular. Una para computadoras definidas como rápidas y otras definidas como lentas. Si solo se especifica un peso, este es usado para ambos tipos de computadoras.

Un ejemplo de setear el peso de los tipos de vista es por ejemplo si se setea la vista *image/jpeg* a 1000 y *image/gif* a 900 siempre se tomará jpeg sobre la gif si esta una disponible.

Se puede también poner en categorías los hosts en el mundo, estas categorías son rápidos y lentos. Usando la opción *viewer Dialog* se puede dar pesos a cada uno de ellos. Note que rápido/lento no se refiere a la velocidad de la máquina esto se refiere a la rapidez de acceso a la red. Un CRAY por ejemplo puede ser la máquina más rápida en el mundo pero si esta al final de una línea de 2400 bps cae en la categoría lenta.

CONFIGURACION DE LOS SERVIDORES

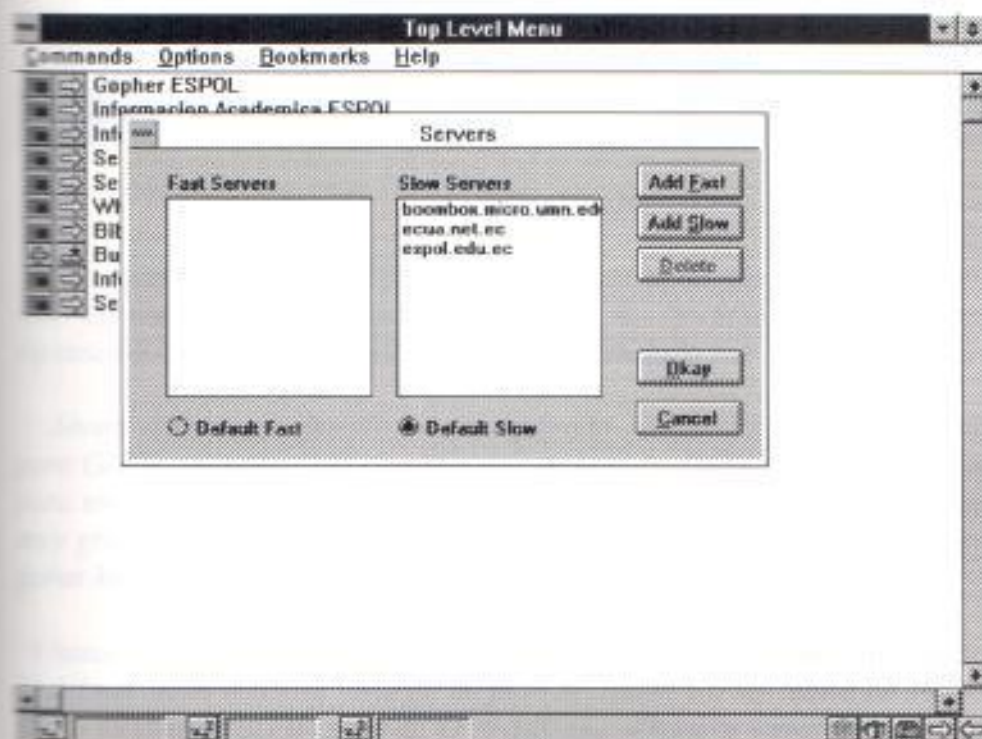


FIG. No 19

SETEO DE LAS VISTAS

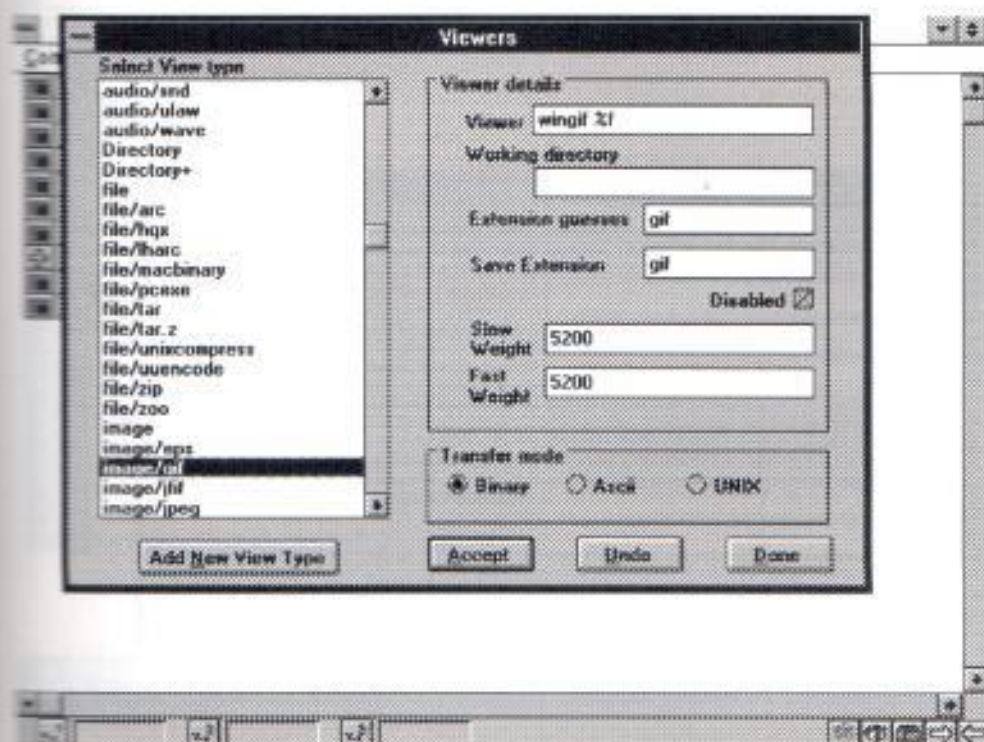
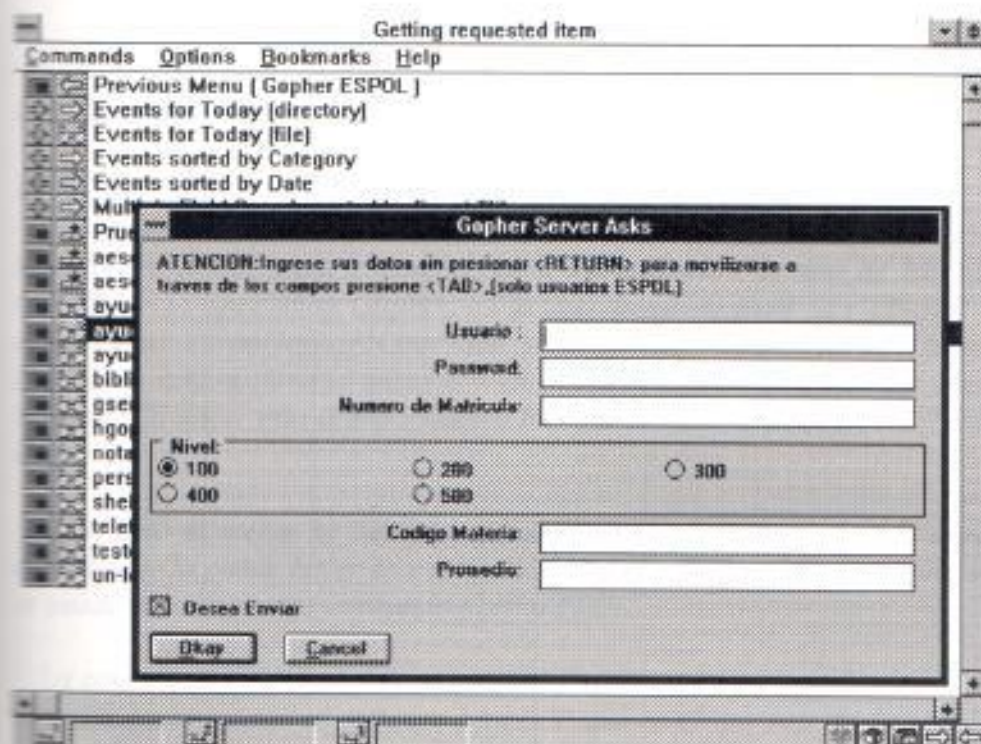


FIG. No. 20

Una cosa más acerca del peso, generalmente es un número entero pero puede ser una expresión conteniendo los símbolos +, - y S(SIZE). el símbolo S(SIZE) representa la medida estimada en Kilobytes y es dado por el gopher servidor.

Ahora veamos un ejemplo más complicado. Supongamos que tenemos una vista para GIF y una para JPEG y sabemos que la vista JPEG toma tres veces más tiempo para mostrar imágenes que GIF. entonces se preferiría traer un GIF a menos que sea muy grande y tome mucho tiempo en transferirlo. Se tiene un gopher local llamado *gopher.here.edu* que directamente esta conectado pro FDDI.

Usando la opción SERVER DIALOG al *gopher.here.edu* como un servidor rápido y el default como lento. la vista gif un peso de 10000 y a los lentos de 10000- S(SIZE). Dar a la vista JPEG un peso de 9700-S(SIZE). Ahora se tomará siempre un gif sobre un JPEG de una máquina definida rápida. y se tomará un gif de todas las otras computadoras a menos que la diferencia entre las medidas sea mayor que 200 Kbytes.

CONSULTA DE NOTAS POR MEDIO DEL GOPHER BAJO WINDOWS*FIG. No 21*

6.3 CLIENTE MACINTOSH

REQUERIMIENTOS

El software cliente para Macintosh es el TurboGopher es uno de los clientes más rápidos de la aplicación Gopher, en primer lugar se tiene que obtener el archivo comprimido TurboGopherxx.hqx que se lo puede obtener de la Universidad de Minesotta, este archivo se encuentra en forma hexadecimal y hay que combertilo en binario. Al hacer esta transformación hay que instalar el paquete en el grupo que se desee.

En este momento se tiene la aplicación pero hay que hacer la conexión esta conexión se la puede hacer en cualquier protocolo TCP serial , esto quiere decir que se puede conectar tanto utilizando SLIP o PPP.

La conexión escogida es la conexión PPP que emula una tarjeta de red y una capa superior encontramos al software MacTCP que es el que nos da la interface con el protocolo TCP/IP.

Configuramos el software para que al conectarse haga requerimientos directamente al gopher de la ESPOL.

En lo que se refiere a requerimientos de hardware el TurboGopher requiere una Macintosh corriendo un sistema 6.0.7 o superior . Es muy recomendable usar un sistema 7 o mayor . Se debe tener al menos un mega de memoria.

VISTAS

Items que son especificados como DOS son vistos como documentos con una pequena PC, similarmente items especificando en UNIX son mostrados con UX en ellos. Se puede grabar cualquiera de este tipo de archivos en el disco del MAC, pero al tratar de visualizarlos no se podra hacerlo para esto se necesita alguna herramienta especial para usar estos items.

Al igual que el gopher cliente para windows el gopher para Mac tiene la opcion de ver las vistas alternativas pero con la diferencia que hay que accesar al menu del gopher para invocarlas.

BOOKMARKS

Si se hace una búsqueda o se encuentra una opción que se desea regresar más tarde. Se puede setear un bookmark en el item del menu. El setear un bookmark hace que se grave una referencia a una carpeta, archivo, una session , etc en realidad cualquier item del gopher.

Los bookmark son colocados en una ventana especial. Para usar un bookmark se utiliza la opción 'Show bookmarks' .

Cuando el TurboGopher comienza este busca por el archivo de seteo. con el cual puede apuntar a cualquier gopher o item que encuentre especificado en este archivo. Si no encuentra este archivo este crea uno nuevo con los defaults.

CONCLUSIONES
Y
RECOMENDACIONES

CONCLUSIONES Y RECOMENDACIONES

- 1.- La migración hecha de los datos del NSSX a la base de datos ORACLE ha resultado exitosa y de consumo de recursos casi nulo, por lo que ha quedado demostrado que es posible esa migración de datos, que resulta conveniente por lo económico y las facilidades que presta a futuro.
- 2.- Las transacciones hechas por el servidor y el cliente del gopher que son cuatro en el peor de los casos hacen que sea un software que decongestiona mucho el tráfico de la red lo que hace que sea un candidato muy bueno para el futuro.
- 3.- En los actuales momento que casi todos los recursos la politécnica en lo referente a Internet se encuentran concentrados en el SUN del centro de cómputo, hace que el Gopher sea uno de los mejores servidores de información debido a que es muy liviano en recursos que consume en comparación a otros servidores de información.
- 4.- Los recursos consumidos por la base de datos Oracle tal vez resulten muchos pero a futuro si se opta por un sistema que este basado en ORACLE que sería uno de los propuestos para una migración, estos recursos ya no resultarán muchos.
- 5.- Las bases diseñadas para estas aplicaciones serán útiles para aplicaciones que se hagan en un futuro.
- 6.- El servidor WAIS que se ha instalado es un complemento para el sistema de información debido a que organiza índices con los archivos indicados y que pueden ser accedidos desde cualquier parte por un servidor WAIS o un gopher con capacidad de de acceder a servidores WAIS.
- 7.- EL PERL (Practical Extraction and Report Language) que es un interpretador de lenguaje optimizado para la búsqueda arbitraria de archivos de textos, extrayendo información de estos archivos e imprimiendo reportes basados en esta información servirá de mucho para tareas en Unix y para script de cualquier tipo ya que es muy fácil de manejar.
- 8.- El ORAPEL y el Gophersql. El Oraperl es un lenguaje que tiene la misma filosofía del Perl y el Gophersql es un script que corre en Oraperl. La principal diferencia entre el Perl y el Oraperl es que el Oraperl puede hacer sentencias SQL y manejar una serie de parámetros propios de las bases de datos. En realidad no todos los scripts se hicieron utilizando el gophersql sino que se hicieron con el Oraperl, por la mayor flexibilidad que ofrecía.

9.- Por la mayor flexibilidad que ofrece el Oracle que ya esta instalado y por los muchos recursos que consume el CSO que son 16M en discon duro sin los datos , se optó por la instalación del cliente ademas de todas las facilidades de manejar datos por parte de Oracle.

10.- Todo lo realizado hasta el momento en el desarrollo del SI-ESPOL no servirá de nada si no se da un apoyo por parte de las autoridades académicas y administrativas ya que no solo el gopher es lo que está instalado sino que detrás de él hay una serie de servidores y utilitarios que servirán para dar la funcionalidad. con el impulso que se da a este proyecto no solamente se estará dando impulso a un sistema de información sino que se creará nuevos proyectos y de esa manera este sera un incentivo más para la investigación.

11.- La administración del gopher es muy importante debido a que la información puesta en el será un incentivo o un desaliento para quienes ingresen a él, la información tendrá que estar siendo reemplazada por información nueva y actualizada.

RECOMENDACIONES.

-El administrador del gopher tendrá que ser un estudiante por preferencia estudiante del nivel 400.

-No deberá desalentarse nuevos servidores de información y crear polémica entre uno y otro tipo de servidores de información más bien utilizar lo ya hecho y de esta manera avanzar más.

-Mejorar los recursos ya que con el tiempo resultarán muy limitados.

-Incentivar a la población estudiantil al uso de estos servidores de información ya que des esta manera se incentiva la investigación.

- Se propone mejorar el sistema de notas ya implementado para el CRECE y que el Gopher pueda acceder en línea a los datos. Esto se podría obtenerse ya sea con los datos en línea en el host o con los datos en la base de datos Oracle que lo que se recomienda.

- Los registro de materias se lo podría hacer con este sistema implementarlos sería un complemento del punto anterior.

- Escoger un servidor de hipertexto sería lo recomendable pero al hacer esto sería necesario subir la velocidad de los modem que se utilizan actualmente.

-Para un futuro próximo y si se ha hecho la migración de datos a una base de datos relacional, como se ha dicho anteriormente Oracle, se Podría impulsar implementaciones clientes-servidor para distintas facultades o unidades.

APENDICE A
ARCHIVOS DE CONFIGURACION

GOPHER

MAKEFILE.CONFIG

```
#-----  
# This is where most of the configuration parameters are defined  
# also see conf.h  
#  
  
#*****  
*  
# lindner  
# 3.13  
# 1994/05/27 04:56:29  
# /home/mudhoney/GopherSrc/CVS/gopher+/Makefile.config.dist,v  
# Exp  
#  
# Paul Lindner, University of Minnesota CIS.  
#  
# Copyright 1991, 1992 by the Regents of the University of Minnesota  
# see the file "Copyright" in the distribution for conditions of use.  
#*****  
*  
# MODULE: Makefile.config.dist  
# Parameters for the gopher distribution  
#*****  
**  
# Revision History:  
# Makefile.config.dist,v  
# Revision 3.13 1994/05/27 04:56:29 lindner  
# Use -O  
#  
# Revision 3.12 1994/03/08 15:54:06 lindner  
# Add item for VFORK  
#  
# Revision 3.11 1994/01/20 06:43:07 lindner  
# support for systems that have flock() instead of fcntl()-locking  
#  
# Revision 3.10 1994/01/12 22:23:46 lindner
```


Fixes for Data General

Revision 3.9 1994/01/06 06:58:40 lindner
Additions for client logging

Revision 3.8 1993/11/03 15:02:26 lindner
pl10

Revision 3.7 1993/09/22 04:32:32 lindner
Doc fixes

Revision 3.6 1993/09/03 03:26:12 lindner
Make sun shared libraries optional

Revision 3.5 1993/07/30 14:21:42 lindner
A/UX mods, and Mitra autoexit mods

Revision 3.4 1993/07/27 05:33:51 lindner
Mitra mondo debug overhaul

Revision 3.3 1993/04/15 22:02:32 lindner
CLIENTOPTS added

Revision 3.2 1993/03/19 19:41:46 lindner
updates for sco

Revision 3.1.1.1 1993/02/11 18:02:49 lindner
Gopher+1.2beta release

Revision 1.7 1993/02/09 21:34:03 lindner
New MAN5DIR line, changed install to install -c

Revision 1.6 1993/01/19 04:52:22 lindner
Renamed Makefile.config to Makefile.config.dist

Revision 1.5 1993/01/13 16:19:58 lindner
Changes for SVR4 (add -lnsl to libs line..)

Revision 1.4 1992/12/28 21:45:48 lindner
Removed trailing slash on CLIENTLIB

Revision 1.3 1992/12/21 20:38:05 lindner
Added warning about -DBIO (from dgg)
#

```
# Revision 1.2 1992/12/13 06:10:26 lindner
# Fixed SVR4LIBS line (didn't need bsd stuff anymore) also removed HPLIBS
#
# Revision 1.1 1992/12/10 22:57:05 lindner
# Initial revision
#
#
#*****
*/
```

```
#-----
# Your favorite C compiler
#
# Note that sun international users should use /usr/5bin/cc instead of cc
#
# Sco's cc compiler gives lots of problems that gcc will fix, and gcc
# is now reasonably easy to get running under SCO. Using this removes
# the need for -UM_XENIX -DSCO_UNIX as used for gopher1.1
# note that if you use gcc, you'll also need -lintl in SCOLIBS
#
```

CC = cc

```
#-----
# Optimization level.
#
```

OPT=-g
#OPT=-O

```
#-----
# System Selection, note that you won't have to edit
# unless you have compilation problems.
#
# Add -DUSG for System V
# -DBSD for BSD
# -DNO_WAITPID if you have wait3 instead of waitpid()
# -DUSE_FLOCK if you have flock instead of fcntl() locking
```

GSYSTYPE=

```
#-----  
# The ranlib command on your system. A/UX (and probably other sysv's  
# should change this to "touch"  
# Known to need changing on: A/UX, SCO3.2.4, IRIX
```

```
RANLIB = ranlib
```

```
#-----  
# The install command on your system. OSF/1 should change this  
# to installbsd  
#  
# SCO ODT systems should change this to bsdinst  
# other SCO systems can duplicate this functionality easily  
# dont use SCO's "install" program it behaves totally differently  
# IBM AIX systems should change this to /usr/ucb/install  
#
```

```
INSTALL = install -c
```

```
#-----  
# Where shall we install stuff?  
#  
PREFIX      = /home/user/gopher  
CLIENTDIR   = $(PREFIX)/bin  
CLIENTLIB   = $(PREFIX)/lib  
SERVERDIR   = $(PREFIX)/etc  
# On SCO manuals are in /usr/man but its easiest to do a  
# symbolic link from /usr/local/man to /usr/man for this and other packages  
MAN1DIR      = $(PREFIX)/man/man1  
MAN5DIR      = $(PREFIX)/man/man5  
MAN8DIR      = $(PREFIX)/man/man8
```

```
#-----  
# DEBUGGING control...  
#  
# Comment this to make a slimmer executable...
```

```
DEBUGGING = -DDEBUGGING
```

```
#-----  
# Optional server features.  
#
```

```

# Add -DADD_DATE_AND_TIME to add dates and times to the gopher titles
#
# Add -DLOADRESTRICT if you want to restrict access based on load avg.
# (Note you'll need to add -lkvm in SERVERLIBS) Note also that this has
# only been tested under SunOS 4.1.1
#
# Add -DBIO if you're using the biology portion of Don Gilbert's modified
# wais8b5 that supports boolean and phrase searching.
# PLEASE NOTE: the -DBIO option is NOT needed nor recommended for use
# of the boolean and phrase searching portion of this modified wais,
# just the symbol searching. But, you must compile wais and gopher with
# the same option setting (-DBIO or NOT).
# The source can be gotten from:
#
# ftp.bio.indiana.edu
#
# Add -DDL and define DLPATH and DLOBS if you want to include support
# for Tim Cook's 'dl' databases You will also have to have a
# working copy the program with source code in the DLPATH
# directory. The files getdesc.o and enddesc.o must be there.
# Source for dl can be gotten from:
#
# admin.viccol.edu.au
# pub/dl/describe-1.8.tar.Z or higher...
#
# Add -DUMNDES if you'd like to try out the Admit1 protocol extension
#
# Add -DCAPFILES if you want compatibility with the older .cap directory
#
# Add -DSETPROCTITLE if you want to set the process title displayed
# by the 'ps' command (only works on bsdish systems...)
#
SERVEROPTS = -DSETPROCTITLE -DCAPFILES #-DUMNDES -DBIO -
DDL -DLOADRESTRICT
#DLPATH = /home/mudhoney/lindner/src/describe
#DLOBS = $(DLPATH)/getdesc.o $(DLPATH)/enddesc.o

#-----
# Optional client features.
#
# Add -DNOMAIL if you don't want remote users mailing documents
#

```


Add -DAUTOEXITONU if you want to treat q and u as the same, and automatically
exit from the top menu - usefull if Gopher called from another app.

CLIENTOPTS = #-DNOMAIL -DREMOTEUSER -DCLIENT_LOGGER

#-----
Libraries for clients and servers
Ultrix needs -lcursesX instead of -lcurses
#

#-----
Libraries... Uncomment out SEQLIBS if compiling on sequent Dynix,
" " PTXLIBS if compiling on sequent Dynix/ptx,
" " UMAXLIBS if compiling under UMAX,
" " SCOLIBS if compiling under SCO Unix.
" " AUXLIBS if compiling under A/UX
" " INTERACTIVELIBS if compiling under Interactive
#

Note: SCOLIBS needs -lintl if using gcc to compile in order to find strftime
#

#UMAXLIBS = -lresolv
#LOADLIBS = -lkvm
#SEQLIBS = -lseq
#PTXLIBS = -lseq -lsocket -linet -lnsl
#SCOLIBS = -lsocket -lintl
#SVR4LIBS = -lsocket -lnsl
#AUXLIBS = -lmalloc
#INTERACTIVELIBS= -linet
#DGUXLIBS = -lnsl

OTHERLIBS = \$(UMAXLIBS) \$(SEQLIBS) \$(PTXLIBS) \$(SCOLIBS) \
\$(SVR4LIBS) \$(AUXLIBS) \$(DGUXLIBS)

CLIENTLIBS = -lcurses -ltermcap -lgopher \$(OTHERLIBS) -ldl
SERVERLIBS = -lm -lgopher \$(OTHERLIBS) \$(LOADLIBS) -ldl

Uncomment out this line to use shared libraries on Sun systems
#

#SHAREDLIB = sun

#-----

*# If your hostname command returns the Fully Qualified Domain Name
(i.e. it looks like foo.bar.edu and not just foo) then make
the domain name a null string. Otherwise put in the rest of
your domain name that `hostname` doesn't return.
Set to Null on SCO3.2.4*

DOMAIN =

*#-----
SERVERDIR is the default directory for gopherd. It can be
overridden on the command line

SERVERPORT is the default port for gopherd. It too can be
overridden on the command line.
#*

*SERVERDATA = /home/apoyo/gopherda/gopher-data
SERVERPORT = 70*

*#-----
Compatibility defines

If you don't have the strstr() function call then add -DNOSTRSTR

Most of these are automatically defined via the built in compiler
definitions. Don't worry about them unless you have problems
#*

*COMPAT = # -DNOSTRSTR # -DNO_STRDUP # -DNO_BZERO # -
DNO_TMPNAM # -DNO_VFORK*

*#-----
Stuff that follows shouldn't be changed
#*

*OBJINCLUDES = -I../object
OTHERINCLUDES = -I. -I../ir -I../ui
INCLUDES = \$(OBJINCLUDES) \$(OTHERINCLUDES)*

LDFLAGS
SHELL

= *-L../object*
= */bin/sh*

□

CONF.H

More configuration parameters.

* *Revision History:*

* *conf.h,v*

* *Revision 3.18 1994/06/09 04:06:28 lindner*

* *F.Macrides 27-May-1994 Added option to allow 'd'ete only for*

* *bookmarks via a DELETE_BOOKMARKS_ONLY compilation symbol.*

* *Added option to not read maps from the user rc file (i.e., only from*
* *the system rc file) in SecureMode or NoShellMode, via the compilation*
* *symbol SECURE_MAPS_GLOBALRC_ONLY.*

* *Added info about the NOMAIL compilation symbol in the VMS section.*

* *Put back -force_html %s for the lynx command in the VMS section*
* *(really *is* needed with the current text html code when foo.html*
* *files are supplied by a gopher+ server from it's own data tree; other*
* *Web browsers don't have that switch, and won't work right with gopher+*
* *servers until they can use the gopher+ extra stuff to determine the*
* *mime type.*

* *Revision 3.17 1994/05/11 02:48:16 lindner*

* *fix for VMS gopherprint defines*

* *Revision 3.16 1994/04/14 17:03:02 lindner*

* *fix for html command*

* *Revision 3.15 1994/01/20 06:43:51 lindner*

* *text/html viewer support for lynx 2.1 & CERN's www-linemode client*

* *Revision 3.14 1994/01/14 16:24:22 lindner*

* *Added anonymous ftp type 'f' option*

* *Revision 3.13 1993/11/03 03:36:35 lindner*

* *Mod for variable records*

* *Revision 3.12 1993/10/13 16:46:51 lindner*

* *Updates for %s on defaults, vms mods*

* *Revision 3.11 1993/09/22 04:30:31 lindner*

* *Add option to conf.h for Max WAIS documents*

- *
* *Revision 3.10 1993/09/11 07:08:50 lindner*
* *Mucho stuff for VMS, callable HTML stuff*
*
- * *Revision 3.9 1993/08/28 04:59:03 lindner*
* *Moved GLOBALRC definition to conf.h for VMS*
*
- * *Revision 3.8 1993/08/19 20:32:59 lindner*
* *add default remoterc, change read timeout to 1 minute*
*
- * *Revision 3.7 1993/08/12 06:35:08 lindner*
* *Don't override CONF_FILE definition, use mail instead of /bin/mail for VMS*
*
- * *Revision 3.6 1993/08/04 22:07:42 lindner*
* *Use /bin/mail instead of ucmail*
*
- * *Revision 3.5 1993/07/27 05:35:30 lindner*
* *reading material for VMS, dead code removal*
*
- * *Revision 3.4 1993/04/15 22:08:51 lindner*
* *Remote user mods (Mitra)*
*
- * *Revision 3.3 1993/03/18 23:11:16 lindner*
* *1.2b3 release*
*
- * *Revision 3.2 1993/02/19 21:25:03 lindner*
* *Updated pager command for gopher+ stuff.*
*
- * *Revision 3.1.1.1 1993/02/11 18:02:49 lindner*
* *Gopher+1.2beta release*
*
- * *Revision 1.7 1993/02/09 22:49:34 lindner*
* *Fixes for new mapping thing*
*
- * *Revision 1.6 1993/01/08 23:04:48 lindner*
* *Changed TN3270_COMMAND for Multinet*
*
- * *Revision 1.5 1992/12/31 05:32:43 lindner*
* *Added mods for VMS*
*
- * *Revision 1.4 1992/12/22 21:45:26 lindner*
* *Fixed bug with that zcat code I just added...*
*
- * *Revision 1.3 1992/12/21 20:27:25 lindner*

** Added #ifdef to make zcat changable..*

** Revision 1.2 1992/12/13 05:56:32 lindner*
** Added options for connection time-out code in the server (mtm)*

** Revision 1.1 1992/12/11 19:01:58 lindner*
** Gopher 1.1 Release*

*/**
** Defaults for the client program*
** On startup the client will contact either the gopher server*
** CLIENT1_HOST or CLIENT2_HOST randomly.*

** Set CLIENT2_PORT to 0 if you only want one root machine*
**/*

#define CLIENT1_HOST "espol.edu.ec"
#define CLIENT2_HOST "espol.edu.ec"

#define CLIENT1_PORT 70
#define CLIENT2_PORT 0

*/**
** Defaults for the client's Gopher server aFTP gateway.*

** On command 'f' the client will prompt for an aFTP Host*
** and create a directory for it via the gateway.*
** You can optionally enter a selector for a particular*
** directory on the aFTP Host, and the gateway will return*
** that instead of the Host's root directory.*

** Set AFTP_HOST to your Gopher server with the gateway implemented*

** Set AFTP_PORT to its port number*
**/*

#define AFTP_HOST "espol.edu.ec"

#define AFTP_PORT 70

```

/*
 * Define this if you want the 'delete' command restricted to bookmarks
 */
/* #define DELETE_BOOKMARKS_ONLY /* */

/*
 * Define this if you want only the system rc file read for maps when
 * the client is invoked in SecureMode or NoShellMode (bookmarks in the
 * user account's gopherrc will still be read.
 */
/* #define SECURE_MAPS_GLOBALRC_ONLY /* */

/*
 * Override some defaults for various platforms
 */

#if defined(sun)
#define PLAY_COMMAND "play -v 40 -"
#endif

#if defined(NeXT)
#define NO_VPRINTF
#define PLAY_COMMAND "play -v 40 -"
#endif

#if defined(_SEQUENT_)
#define PRINTER_COMMAND "lp"
#endif

#if defined(VMS)
/*
 * VMS systems use VARIABLE length records for text files and
 * FIXED 512 records for binary files that are saved or cached.
 * Comment this out if you want to use stream_LF format instead.
 * The Bookmark (sys$login:gopherrc.) and configuration
 * (GopherP_Dir:gopher.rc and GopherP_Dir:gopherremote.rc, see
 * below) files are stream_LF regardless of how this program

```

```

* logical is set.
*/
#define VMSRecords /* */

/*
* The "builtin" pager is the default VMS utility for displaying text.
* Alternatively, define "TPU" for invoking callable TPU. Its /READ_ONLY
* and /NOJOURNAL qualifiers are added internally by the software.
* A still better alternative is to acquire MOST, optimized for Gopher and
* C SWING, via anonymous FTP from narnia.memst.edu. Define it with the -n
* and +s switches.
*/
#define PAGER_COMMAND "builtin" /* */
/* #define PAGER_COMMAND "TPU/NOINI/COM=GopherP_Dir:GOPHER.TPU
%s" /* */
/* #define PAGER_COMMAND "most -n +s %s" /* */

/*
* MAIL_COMMAND is the program logical for the mail verb.
* MAIL_ADRS is the argument for an sprintf() command that can add
* "prefix%" "ADDRESS" to the Internet mail address given by the
* user. It is structured for PMDF's IN% "INTERNET_ADDRESS"
* scheme. The %s is replaced with the address given by the
* user. No conversion will be done if a DECNET or simple
* VMS MAIL address is given. The default definition of "%s" for
* MAIL_ADRS does not provide translation. If you want to use
* PMDF's prefix of IN%, Message Exchange's (MX's) prefix of MX%,
* MultiNet's prefix of SMTP%, or Wollongong's prefix of WINS%,
* comment out the default definition of MAIL_ADRS and uncomment
* the appropriate line below to your selected system.
*/
#define MAIL_COMMAND "mail"
#define MAIL_ADRS "%s" /* */
/* #define MAIL_ADRS "\"IN%%\" \"%s\" \"\" \"\" /* */
/* #define MAIL_ADRS "\"MX%%\" \"%s\" \"\" \"\" /* */
/* #define MAIL_ADRS "\"WINS%%\" \"%s\" \"\" \"\" /* */
/* #define MAIL_ADRS "\"SMTP%%\" \"%s\" \"\" \"\" /* */

/*
* Use MultiNet's command verb for telnet and tn3270, so that it can
* co-exist with other TCPIP transports on VMS systems.
*/
#if defined(MULTINET)
# define TELNET_COMMAND "multinet telnet"

```



```

# define TN3270_COMMAND "multinet telnet/tn3270"
#else
# define TELNET_COMMAND "telnet"
# define TN3270_COMMAND "tn3270"
#endif

/*
* The Printer Command may be setup to use a command procedure to save the
* file before printing it. This avoids the problem of Gopher removing the
* temporary file before it can be printed. For example set up a command
* procedure in a public place (e.g., GopherP_Dir:GOPHERPRINT.COM)
containing:
*   S file="sys$scratch:gopher_" + f$extract(21,2,f$time()) + ".tmp"
*   S copy 'p1' 'file'
*   S print 'f$trnlhm("GOPHERQUEUE")' /delete/noidentify 'file'
* (Note that the logical GOPHERQUEUE can be used to set options like
* default queue name or form type eg DEFINE/JOB GOPHERQUEUE
*/queue=que1")
* and then define the Printer Command appropriately
*/
#define PRINTER_COMMAND "print %s" /* */
/* #define PRINTER_COMMAND "@GopherP_Dir:GOPHERPRINT %s" /* */

/*
* Sounds are not implemented on VMS.
* They can only be 's'aved or 'D'ownloaded.
*/
#define PLAY_COMMAND "- none -"

/*
* If IMAGE_COMMAND is defined as "- none -"
* a print command is not added to its default map,
* and images can only be 's'aved or 'D'ownloaded.
*/
#define IMAGE_COMMAND "xv %s" /* */
/* #define IMAGE_COMMAND "- none -" /* */

/*
* The builtin HTML browser is not yet functional.
*
* Lynx is a Curses-based HTML browser, available from ftp2.cc.ukans.edu,
* which was initially designed to use Gopher as its server and still works
* well with this software (on both VMS and Unix platforms).

```

** The WWW Line-Mode browser is available from info.cern.ch*

**/*

/#define HTML_COMMAND "- none -" */*

#define HTML_COMMAND "lynx -force_html %s" / lynx 2.2 or greater */*

/ #define HTML_COMMAND "www" /* WWW Line-Mode client */*

*/**

** Point these to the default configuration files for view command maps:*

** S define/system "GopherP_Dir" "device:[directory]"*

** Make sure the files have commands mapped appropriately for VMS,*

** and have any maps which don't apply commented out.*

**/*

#define GLOBALRC "GopherP_Dir:gopher.rc"

#define REMOTERC "GopherP_Dir:gopherremote.rc"

*/**

** Point this to the on-line Gopher+ help file.*

**/*

#define GOPHERHELP "GopherP_Dir:gopher.hlp"

*/**

** Define this if you don't want remote users mailing or downloading documents.*

** (printing and saving to disk are always disabled for remote users).*

**/*

/ #define NOMAIL /* */*

#endif / VMS (Have you noticed how verbose VMSers tend to be?!?!?) */*

*/**

** Now set the parameters, only if not set above...*

**/*

#ifndef PAGER_COMMAND

#define PAGER_COMMAND "builtin"

#endif

#ifndef MAIL_COMMAND

#define MAIL_COMMAND "/bin/mail"

#endif

```
#ifndef TELNET_COMMAND
#define TELNET_COMMAND "telnet"
#endif
```

```
#ifndef TN3270_COMMAND
#define TN3270_COMMAND "tn3270"
#endif
```

```
#ifndef PRINTER_COMMAND
#define PRINTER_COMMAND "lpr"
#endif
```

```
#ifndef PLAY_COMMAND
#define PLAY_COMMAND "/bin/false"
#endif
```

```
#ifndef MIME_COMMAND
#define MIME_COMMAND "metamail -P"
#endif
```

```
#ifndef IMAGE_COMMAND
#define IMAGE_COMMAND "xloadimage -fork %s"
#endif
```

```
#ifndef HTML_COMMAND
#define HTML_COMMAND "lynx -force_html %s"
#endif
```

```
#ifndef REMOTERC
#define REMOTERC "/home/user/gopher/lib/gopherrc.remote"
#endif
```

```
/* ***** gopherd configuration ***** */
```

```
/*
 * The maximum number of hits to return from a query to a
 * WAIS index.
 */
```

```
#define WAISMAXHITS 40
```

```
/*
 * The load average at which to restrict connections
 */
```

```
#define MAXLOAD 10.0
```

```
/*  
 * Return type for signal()  
 */
```

```
#define SIGRETTYPE void
```

```
/*  
 * Timeout for network reads (1 minute)  
 */
```

```
#define READTIMEOUT (1 * 60)
```

```
/* We need to define this since inetd.conf can only have a few  
 arguments, and we need lots of them */
```

```
#if !defined(CONF_FILE)  
# define CONF_FILE      "/home/user/gopher/etc/gopherd.conf"  
#endif
```

```
□
```


ARCHIVO DE CONFIGURACION DEL PERL

```
# : Makefile.SH,v 16579Revision: 4.0.1.4 16579Date: 92/06/08 11:40:43 $
#
# SLog:      Makefile.SH,v $
# Revision 4.0.1.4 92/06/08 11:40:43 lwall
# patch20:  cray didn't give enough memory to /bin/sh
# patch20:  various and sundry fixes
#
# Revision 4.0.1.3 91/11/05 15:48:11 lwall
# patch11:  saberized perl
# patch11:  added support for dbz
#
# Revision 4.0.1.2 91/06/07 10:14:43 lwall
# patch4:   cflags now emits entire cc command except for the filename
# patch4:   alternate make programs are now semi-supported
# patch4:   uperl.o no longer tries to link in libraries prematurely
# patch4:   installperl now installs x2p stuff too
#
# Revision 4.0.1.1 91/04/11 17:30:39 lwall
# patch1:   C flags are now settable on a per-file basis
#
# Revision 4.0 91/03/20 00:58:54 lwall
# 4.0 baseline.
#
#
```

```
CC = cc
YACC = /usr/bin/yacc
bin = /usr/local/bin
scriptdir = /usr/local/bin
privlib = /usr/local/lib/perl
mansrc = /usr/man/man1
manext = 1
LDFLAGS =
CLDFLAGS =
SMALL =
LARGE =
mallosrc = malloc.c
mallocobj = malloc.o
SLN = ln
RMS = rm -f
```

libs = -ldl -lnsl -ldbm -lm -lposix

public = perl taintperl suidperl

shellflags =

*# To use an alternate make, set in config.sh.
MAKE = make*

CCCMD = `sh \$(shellflags) cflags \$@`

private =

scripts = h2ph

manpages = perl.man h2ph.man

util =

sh = Makefile.SH makedepend.SH h2ph.SH

h1 = EXTERN.h INTERN.h arg.h array.h cmd.h config.h form.h handy.h

h2 = hash.h perl.h regcomp.h regexp.h spat.h stab.h str.h util.h

h = \$(h1) \$(h2)

c1 = array.c cmd.c cons.c consarg.c doarg.c doio.c dolist.c dump.c

c2 = eval.c form.c hash.c \$(mallocsrc) perl.c regcomp.c regex.c

c3 = stab.c str.c token.c util.c usersub.c

c = \$(c1) \$(c2) \$(c3)

s1 = array.c cmd.c cons.c consarg.c doarg.c doio.c dolist.c dump.c

s2 = eval.c form.c hash.c perl.c regcomp.c regex.c

s3 = stab.c str.c token.c util.c usersub.c perl.c

saber = \$(s1) \$(s2) \$(s3)

obj1 = array.o cmd.o cons.o consarg.o doarg.o doio.o dolist.o dump.o

obj2 = eval.o form.o \$(mallocobj) perl.o regcomp.o regex.o

obj3 = stab.o str.o token.o util.o

obj = \$(obj1) \$(obj2) \$(obj3)

tobj1 = tarray.o tcmd.o tcons.o tconsarg.o tdoarg.o tdoio.o tdolist.o tdump.o
tobj2 = teval.o tform.o thash.o \$(mallocobj) tregcomp.o tregexec.o
tobj3 = tstab.o tstr.o ttoke.o tutil.o

tobj = \$(tobj1) \$(tobj2) \$(tobj3)

lintflags = -hbvxac

addedbyconf = Makefile.old bsd eunice filexp loc pdp11 usg v7

grrr

SHELL = /bin/sh

.c.o:

\$(CCCMD) \$*.c

all: \$(public) \$(private) \$(util) uperl.o \$(scripts)
cd x2p; \$(MAKE) all
touch all

This is the standard version that contains no "taint" checks and is
used for all scripts that aren't set-id or running under something set-id.
The \$& notation is tells Sequent machines that it can do a parallel make,
and is harmless otherwise.

perl: \$& perly.o \$(obj) hash.o usersub.o
\$(CC) \$(LARGE) \$(CLDFLAGS) \$(obj) hash.o perly.o usersub.o \$(libs) -o
perl

This command assumes that /usr/include/dbz.h and /usr/lib/dbz.o exist.

dbzperl: \$& perly.o \$(obj) zhash.o usersub.o
\$(CC) \$(LARGE) \$(CLDFLAGS) \$(obj) zhash.o /usr/lib/dbz.o perly.o
usersub.o \$(libs) -o dbzperl

zhash.o: hash.c \$(h)
\$(RMS) zhash.c
\$(SLN) hash.c zhash.c
\$(CCCMD) -DWANT_DBZ zhash.c
\$(RMS) zhash.c

uperl.o: \$& perly.o \$(obj) hash.o

-ld \$(LARGE) \$(LDFLAGS) -r \$(obj) hash.o perly.o -o uperl.o

saber: \$(saber)

load \$(saber)

load /lib/libm.a

*# This version, if specified in Configure, does ONLY those scripts which need
set-id emulation. Suidperl must be setuid root. It contains the "taint"
checks as well as the special code to validate that the script in question
has been invoked correctly.*

suidperl: \$(CC) tperly.o sperl.o \$(tobj) usersub.o

*\$(LARGE) \$(CLDFLAGS) sperl.o \$(tobj) tperly.o usersub.o \$(libs) \
-o suidperl*

*# This version interprets scripts that are already set-id either via a wrapper
or through the kernel allowing set-id scripts (bad idea). Taintperl must
NOT be setuid to root or anything else. The only difference between it
and normal perl is the presence of the "taint" checks.*

taintperl: \$(CC) tperly.o tperl.o \$(tobj) usersub.o

*\$(LARGE) \$(CLDFLAGS) tperl.o \$(tobj) tperly.o usersub.o \$(libs) \
-o taintperl*

Replicating all this junk is yucky, but I don't see a portable way to fix it.

tperly.o: perly.c perly.h \$(h)

\$(RMS) tperly.c

\$(SLN) perly.c tperly.c

\$(CCCMD) -DTAINT tperly.c

\$(RMS) tperly.c

tperl.o: perl.c perly.h patchlevel.h perl.h \$(h)

\$(RMS) tperl.c

\$(SLN) perl.c tperl.c

\$(CCCMD) -DTAINT tperl.c

\$(RMS) tperl.c

sperl.o: perl.c perly.h patchlevel.h \$(h)

\$(RMS) sperl.c

\$(SLN) perl.c sperl.c

\$(CCCMD) -DTAINT -DIAMSUID sperl.c

\$(RMS) sperl.c

tarray.o: array.c \$(h)
\$(RMS) tarray.c
\$(SLN) array.c tarray.c
\$(CCCMD) -DTAINT tarray.c
\$(RMS) tarray.c

tcmd.o: cmd.c \$(h)
\$(RMS) tcmd.c
\$(SLN) cmd.c tcmd.c
\$(CCCMD) -DTAINT tcmd.c
\$(RMS) tcmd.c

tcons.o: cons.c \$(h) perly.h
\$(RMS) tcons.c
\$(SLN) cons.c tcons.c
\$(CCCMD) -DTAINT tcons.c
\$(RMS) tcons.c

tconsarg.o: consarg.c \$(h)
\$(RMS) tconsarg.c
\$(SLN) consarg.c tconsarg.c
\$(CCCMD) -DTAINT tconsarg.c
\$(RMS) tconsarg.c

tdoarg.o: doarg.c \$(h)
\$(RMS) tdoarg.c
\$(SLN) doarg.c tdoarg.c
\$(CCCMD) -DTAINT tdoarg.c
\$(RMS) tdoarg.c

tdoio.o: doio.c \$(h)
\$(RMS) tdoio.c
\$(SLN) doio.c tdoio.c
\$(CCCMD) -DTAINT tdoio.c
\$(RMS) tdoio.c

tdolist.o: dolist.c \$(h)
\$(RMS) tdolist.c
\$(SLN) dolist.c tdolist.c
\$(CCCMD) -DTAINT tdolist.c
\$(RMS) tdolist.c

tdump.o: dump.c \$(h)
\$(RMS) tdump.c

\$(SLN) dump.c tdump.c
\$(CCCMD) -DTAINT tdump.c
\$(RMS) tdump.c

teval.o: eval.c \$(h)
\$(RMS) teval.c
\$(SLN) eval.c teval.c
\$(CCCMD) -DTAINT teval.c
\$(RMS) teval.c

tform.o: form.c \$(h)
\$(RMS) tform.c
\$(SLN) form.c tform.c
\$(CCCMD) -DTAINT tform.c
\$(RMS) tform.c

thash.o: hash.c \$(h)
\$(RMS) thash.c
\$(SLN) hash.c thash.c
\$(CCCMD) -DTAINT thash.c
\$(RMS) thash.c

tregcomp.o: regcomp.c \$(h)
\$(RMS) tregcomp.c
\$(SLN) regcomp.c tregcomp.c
\$(CCCMD) -DTAINT tregcomp.c
\$(RMS) tregcomp.c

tregexec.o: regex.c \$(h)
\$(RMS) tregexec.c
\$(SLN) regex.c tregexec.c
\$(CCCMD) -DTAINT tregexec.c
\$(RMS) tregexec.c

tstab.o: stab.c \$(h)
\$(RMS) tstab.c
\$(SLN) stab.c tstab.c
\$(CCCMD) -DTAINT tstab.c
\$(RMS) tstab.c

tstr.o: str.c \$(h) perly.h
\$(RMS) tstr.c
\$(SLN) str.c tstr.c
\$(CCCMD) -DTAINT tstr.c

\$(RMS) tstr.c

*ttoke.o: toke.c \$(h) perly.h
\$(RMS) ttoke.c
\$(SLN) toke.c ttoke.c
\$(CCCMD) -DTAINT ttoke.c
\$(RMS) ttoke.c*

*tutil.o: util.c \$(h)
\$(RMS) tutil.c
\$(SLN) util.c tutil.c
\$(CCCMD) -DTAINT tutil.c
\$(RMS) tutil.c*

*perly.h: perly.c
@ echo Dummy dependency for dumb parallel make
touch perly.h*

*perly.c: perly.y perly.fixer
@ \
case "\$(YACC)" in \
bison) echo 'Expect' 25 shift/reduce and 59 reduce/reduce conflicts;; \
*) echo 'Expect' 27 shift/reduce and 57 reduce/reduce conflicts;; \
esac
\$(YACC) -d perly.y
sh \$(shellflags) ./perly.fixer y.tab.c perly.c
mv y.tab.h perly.h
echo 'extern YYSTYPE yylval;' >>perly.h*

*perly.o: perly.c perly.h \$(h)
\$(CCCMD) perly.c*

*install: all
./perl installperl*

*clean:
rm -f *.o all perl taintperl suidperl perly.c
cd x2p; \$(MAKE) clean*

*realclean: clean
cd x2p; \$(MAKE) realclean
rm -f *.orig */*.orig *~ */*~ core \$(addedbyconf) h2ph h2ph.man
rm -f perly.c perly.h t/perl Makefile config.h makedepend makedir
rm -f makefile x2p/Makefile x2p/makefile cflags x2p/cflags*

```
rm -f c2ph pstruct
```

```
# The following lint has practically everything turned on. Unfortunately,  
# you have to wade through a lot of mumbo jumbo that can't be suppressed.  
# If the source file has a /*NOSTRICT*/ somewhere, ignore the lint message  
# for that spot.
```

```
lint: perly.c $(c)  
    lint $(lintflags) $(defs) perly.c $(c) - perl.fuzz
```

```
depend: makedepend  
    - test -f perly.h || cp /dev/null perly.h  
    ./makedepend  
    - test -s perly.h || /bin/rm -f perly.h  
    cd x2p; $(MAKE) depend
```

```
test: perl  
    - cd t && chmod +x TEST */*.t  
    - cd t && (rm -f perl; $(SLN) ../perl perl) && ./perl TEST </dev/tty
```

```
clist:  
    echo $(c) | tr ' ' '\012' >.clist
```

```
hlist:  
    echo $(h) | tr ' ' '\012' >.hlist
```

```
shlist:  
    echo $(sh) | tr ' ' '\012' >.shlist
```

```
# AUTOMATICALLY GENERATED MAKE DEPENDENCIES--PUT NOTHING  
BELOW THIS LINE
```

```
$(obj) hash.o:  
    @ echo "You haven't done a ""make depend" yet!"; exit 1
```

```
makedepend: makedepend.SH  
    /bin/sh $(shellflags) makedepend.SH
```

```
□
```


ARCHIVO DE CONFIGURACION DEL ORAPERL

```
# Makefile for Oraperl and Coraperl
# Change these to your ORACLE installation directory and Perl source directory
#
ORACLE_HOME    = /oracle/prog
SRC             = /home/user/gopher/perl-4.036
# Oracle Definitions, copied from $(ORACLE_HOME)/c/demo/proc.mk
# ALL_ORA_LIBS is the only entry that the Makefile actually uses;
# change it to whatever you need to link Pro*C programs
#
OTHERLIBS      = `cat $(ORACLE_HOME)/rdbms/lib/sysliblist`
CLIBS          = $(OTHERLIBS)
OCILIB         = $(ORACLE_HOME)/rdbms/lib/libocic.a
NETLIBS        = $(ORACLE_HOME)/rdbms/lib/osntab.o \
                $(ORACLE_HOME)/rdbms/lib/libsqlnet.a
ORALIBS        = $(ORACLE_HOME)/rdbms/lib/libora.a

ALL_ORA_LIBS   = $(CLIBS) $(OCILIB) $(NETLIBS) $(ORALIBS)

# Perl Definitions, taken from $SRC/usub/Makefile
# Don't include the curses libraries here - they go in CURSELIB
#
GLOBINCS      =
LOCINCS       =
LIBS           = `.$(SRC)/config.sh; echo $$libs`

# Oraperl Definitions

# Set DEBUG to -DDEBUGGING, -DPERL_DEBUGGING or leave blank (see
orafns.h)
# If it is not blank, uncomment the definition of DEBUG_O
#
DEBUG         = -DDEBUGGING#-DPERL_DEBUGGING
DEBUG_O       = debug/debug.o

# Curses libraries, only required if you want to build Coraperl
# You may also need -ltermcap
#
CURSELIB      = -lcurses

# Row cache size for SELECT statements.
# If you want to change the default, uncomment this and set the value you want
#
```

```

#CACHE          = -DCACHE_SIZE=3

# Round variable padding
# (if you want binding an empty string to provoke an error, uncomment this line)
#
#BIND          = -DNO_BIND_PADDING

# Some system-specific things
#
# If your system library does not include strtoul, uncomment the next line
STRTOUL       = strtoul.o
#
# If your malloc() returns anything other than a char *, set the appropriate
# type here (don't include the *)
# MALLOC_PTR_TYPE=void
#
# If you are using Perl v3 instead of v4, uncomment the next line
# STR_2MORTAL   = -Dstr_2mortal=str_2static

# Database, username and password to use for testing
#
#TESTDATA     = t scott tiger
TESTDATA      = espol gopher gopher

#
# Leave these blank lines so that patches to what is below
# won't be upset by your changes to the setups above.
#

# From here on, you shouldn't need to change anything. If you do, let me know.

SRCS          = oracle.mus orafns.c getcursor.c colons.c usersub.c \
              debug.c strtoul.c
OBJS          = oracle.o orafns.o getcursor.o colons.o debug.o $(STRTOUL)
OOBJS        = $(OBJS) usersub.o $(DEBUG_O)
COBJS        = $(OBJS) cusersub.o $(DEBUG_O)
HDRS          = patchlevel.h orafns.h
DEFS          = $(STRTOUL) $(PUTENV) $(STR_2MORTAL) $(DEBUG) $(CACHE)
$(BIND)

CFLAGS        = -ldbug -I$(SRC) $(GLOBINCS) $(LOCINCS) $(DEFS)

oraperl: $(SRC)uperl.o $(OOBJS)

```

```

$(CC) -o oraperl $(SRC)/uperl.o $(OOBJS)
      -lm $(ALL_ORA_LIBS) $(LIBS) -ldl

coraperl: $(SRC)/uperl.o $(COBJS) $(SRC)/usub/curses.o
$(CC) -o coraperl $(SRC)/uperl.o $(COBJS) $(SRC)/usub/curses.o
      -lm $(ALL_ORA_LIBS) $(LIBS) $(CURSELIB)

all:   oraperl coraperl

test:  oraperl
      @oraperl -e '&ora_version'
      @(cd testdir ;
      rm -f My-Results ;
      echo "Testing oraperl, please wait ..." ;
      for i in *.pl ; do /usr/local/bin/oraperl $$i $(TESTDATA) ; done > My-
Results ; \
      if cmp -s Standard-Results My-Results ;
      then echo "Test successful" ;
      else echo "Test failed - compare My-Results with Standard-Results" ; \
      fi; echo)

# We use oraperl although perl would suffice, because we know where it is!

install: oraperl install.pl
      @./oraperl ./install.pl ${SRC}

cusersub.c:  usersub.c
      @rm -f cusersub.c
      ln usersub.c cusersub.c

cusersub.o:  cusersub.c
      $(CC) -c $(CFLAGS) -DCURSES cusersub.c

oracle.c: $(SRC)/usub/mus oracle.mus
      $(SRC)/usub/mus oracle.mus > oracle.c

$(OOBJS) $(COBJS):      $(HDRS)

dbug/dbug.o:
      (cd dbug ; $(MAKE) dbug.o)
      @echo "      (back to main directory)"

clean:
      (cd dbug ; $(MAKE) clean)

```

```
@echo "      (back to main directory)"  
rm -f nohup.out *.o oracle.c cusersub.c  
rm -f testdir/My-Results listing tags core
```

```
realclean clobber: clean  
  (cd debug ; $(MAKE) clobber)  
  (cd doc ; $(MAKE) clobber)  
  @echo "      (back to main directory)"  
  rm -f oraperl coraperl
```

```
listing:  
  pr -fn Makefile $(HDRS) $(SRCS) >listing
```

```
docs:  
  (cd doc ; $(MAKE) docs)  
  @echo "      (back to main directory)"
```

```
shar: clean  
  shar -n oraperl-v2 -a -s kstock@encore.com -F -o :Part -l 64 \\  
  Readme [C-Q]* Row* [S-z]*
```

□

ARCHIVO DE CONFIGURACION DE WAIS

```
#  
# Top level make of the WAIS system  
# brewster 2/91  
# jonathan 6/91  
  
# $Log:      Makefile-release,v $  
# Revision 1.8 92/05/07 15:54:08 jonathan  
# Updated for release.  
#  
# Revision 1.7 92/03/26 18:29:59 jonathan  
# Fixed some broken make lines.  
#  
# Revision 1.6 92/03/07 19:45:01 jonathan  
# Added recommendation for IBM.  
#  
# Revision 1.5 92/02/27 10:07:24 jonathan  
# got rid of automatic setting of TOP. Used Simon's approach instead.  
  
# Revision 1.4 92/02/13 12:27:53 jonathan  
# Removed references to seeker.  
#  
# Revision 1.3 92/02/13 12:05:17 jonathan  
# Removed release targets.  
#  
# Revision 1.2 92/02/13 11:57:56 jonathan  
# Added $Log for RCS.  
#  
#  
# common customizations:  
# see the CFLAGS variable for pointers.  
#  
# to do:  
# create the scripts, install the pointer to this version if it is the  
# newest.  
#  
# SGIs want this uncommented  
# SHELL=/bin/sh  
  
RELEASE = freeWAIS-0.3
```

```
RM = /bin/rm -f
AR = ar
```

```
# on SGIs set this to true
# RANLIB = true
RANLIB = ranlib
```

```
# on IBM RS6000 this should be c89.
CC = cc
#CC = gcc
# set this for your site. This syntax only works in SunOS
# for other UNIX-like OS's set this to this directory.
#TOP:sh = pwd
# or fill in the blank for other OS's
#comment-me:
# @echo You must set "\$(TOP)" to point to the freeWAIS src directory
TOP = /home/user/gopher/freeWAIS-0.3
```

```
SUPDIR = $(TOP)/ir
```

```
# compiler specific stuff
#
# for old BSD add -DBSD
# for newer BSD that needs to use <sys/dir.h>, add -DBSD43
# for System V add -DSYSV
# for XENIX add -M3e -Zi
# USG for Unix Dired in lib
# for SGIs running IRIX 4.0.1, add -cckr
# for NeXTSTEP add -DNeXT and -posix
# for Linux add -DLINUX
#
# For a little better security in the server, add -DSECURE_SERVER
# this sets the server user id to -u argument after startup.
# for relevance feedback in the search engine, add -DRELEVANCE_FEEDBACK
#
# dgg additions
# LITERAL == waisserver, search for "literal strings"
# BOOLEANS == waisserver, search with boolean AND, NOT operators
# PARTIALWORD == waisserver, search for partial words, hum* matches human,
# hummingbird, ...
# BIO == waisindex, waisserver changes including symbol indexing & search
# & bio data formats
#
```

```

# -DTELL_USER lets the server know who you are at connect time
# -DUSE_SYSLOG if you want logging to be done with syslog rather than
#   fprintf
# -DNEED_VSYSLOG if your C library does not have a vsyslog() function
#   in it (and you defined USE_SYSLOG)
# -DDUMPCORE will force the waisserver to dump the core when aborting
#   otherwise the core will not be dumped
# -DEND_MERGE if you want to merge the index files at the end of an
#   index process otherwise they are merged as we go along
# -DSTEM_WORDS to stem words during indexing and queries
#
# Note - the default Porter Stemmer removes trailing e's from words -
# variable becomes variabl - this can impact the use of literals in
# searches!!!!!!!!!!!!
#
# -DLIST_STEMS to show stemmed words in server log and indexer output
# -DSOLARIS for SunOS 5.2 (Solaris 2.2) machines.
#
# Use this version of CFLAGS for SGIs with gcc
# CFLAGS = -IS(SUPDIR) -DTELL_USER -DSECURE_SERVER -
# DRELEVANCE_FEEDBACK -DUSG -DBOOLEANS -DPARTIALWORD -
# DLITERAL -DSOUND -DBIBDB
#
# Use this version of CFLAGS for DECstation with gcc
# CFLAGS = -ansi -IS(SUPDIR) -DTELL_USER -DUSG -DSECURE_SERVER -
# DRELEVANCE_FEEDBACK -DBOOLEANS -DPARTIALWORD -DLITERAL -
# DSOUND -DBIBDB -DULTRIX
#
# Use this version of CFLAGS for DECstation with cc
# CFLAGS = -IS(SUPDIR) -DTELL_USER -DUSG -DSECURE_SERVER -
# DRELEVANCE_FEEDBACK -DBOOLEANS -DPARTIALWORD -DLITERAL -
# DSOUND -DBIBDB -DULTRIX
#
# Use this version of CFLAGS for DEC Alpha with gcc
# CFLAGS = -ansi -IS(SUPDIR) -DTELL_USER -DUSG -DSECURE_SERVER -
# DRELEVANCE_FEEDBACK -DBOOLEANS -DPARTIALWORD -DLITERAL -
# DSOUND -DBIBDB
#
# Use this version of CFLAGS on Sun with gcc (not Solaris)
# CFLAGS = -ansi -IS(SUPDIR) -DTELL_USER -DUSG -DSECURE_SERVER -
# DRELEVANCE_FEEDBACK -DBOOLEANS -DPARTIALWORD -DLITERAL -
# DSOUND -DBIBDB
#
# Use this version of CFLAGS for Linux with gcc

```



```
# CFLAGS = -O2 -ansi -m486 -fwritable-strings -I$(SUPDIR) -DTELL_USER -
DUSG -DSECURE_SERVER -DRELEVANCE_FEEDBACK -DBOOLEANS -
DPARTIALWORD -DLITERAL -DSOUND -DBIBDB -DLINUX
```

```
CFLAGS = -g -I$(SUPDIR) -DTELL_USER -DUSG -DSECURE_SERVER -
DRELEVANCE_FEEDBACK -DBOOLEANS -DPARTIALWORD -DLITERAL -
DSOUND -DBIBDB
```

```
# Solaris should use
#LIB = -lsocket -lnsl
# SGIs should use this
#LIB = -lmalloc
LIB =
```

```
# There are different versions of curses which could be used, so set the
# library here
CURSESLIB = -lcurses
```

```
#Solaris machines don't use -k
MFLAGS = -k
```

```
MAKE = make $(MFLAGS)
```

```
#default: config.h lib ir ui bin doc x tags
default: config.h lib ir ui bin check
    @echo "Welcome to WAIS"
```

```
config.h: config
    ./config > config.h
```

```
config: config.c
    $(CC) $(CFLAGS) -o config config.c -ldl
```

```
lib::
    cd lib; $(MAKE) CC=$(CC) CFLAGS="$(CFLAGS)" RANLIB=$(RANLIB) \
TOP=$(TOP)
```

```
ir::
    cd ir; $(MAKE) CC=$(CC) CFLAGS="$(CFLAGS)" RANLIB=$(RANLIB) \
TOP=$(TOP) LIB=$(LIB)
```

```
waisindex::
    cd ir; $(MAKE) waisindex CC=$(CC) CFLAGS="$(CFLAGS)" \
RANLIB=$(RANLIB) TOP=$(TOP) LIB=$(LIB)
```


waisserver::

```
cd ir; $(MAKE) weisserver CC=$(CC) CFLAGS="$$(CFLAGS)" \
RANLIB=$(RANLIB) TOP=$(TOP) LIB=$(LIB)
```

ui::

```
cd ui; $(MAKE) CC=$(CC) CFLAGS="$$(CFLAGS)" TOP=$(TOP) \
CURSESLIB=$(CURSESLIB) LIB=$(LIB)
```

*# NeXT and ULTRIX don't have an env command, so this doesn't work.
try going to the x directory and just doing a make -k
you may have to edit the CFLAGS in the Makefile yourself.*

x::

```
cd x; xmkmf; make depend; make  
# cd x; (env TOP=$(TOP) CC=$(CC) CFLAGS="$$(CFLAGS)"  
MAKE="$$(MAKE)" ./makex.sh)
```

bin::

```
cd bin; $(MAKE) CC=$(CC) CFLAGS="$$(CFLAGS)" TOP=$(TOP)
```

test::

```
@echo $(MAKE) CC=$(CC) CFLAGS="$$(CFLAGS)" TOP=$(TOP)
```

check::

```
cd wais-test; $(MAKE)
```

to make the emacs tags table for meta-:

tags:

```
etags -f TAGS ir/*.ch  
etags -af TAGS ui/*.ch  
etags -af TAGS x/*.ch
```

Remove objects and library.

clean:

```
$(RM) *%  
$(RM) *~  
$(RM) \#*\#  
$(RM) core  
$(RM) TAGS  
$(RM) -r SearchLog  
$(RM) wais-sources/wais-docs*  
$(RM) config config.h  
cd lib; make $@
```

cd ir; make \$@
cd ui; make \$@
cd bin; make \$@
cd wais-test; make \$@
cd x; make \$@

rlocks:

\$@
cd lib; \$@
cd ir; \$@
cd ui; \$@
cd bin; \$@
cd doc; \$@
cd x; \$@

□

APENDICE B

NFS

NETWORK FILE SYSTEM

NFS permite a los directorios y archivos ser compartidos a través de la red. Fue originalmente desarrollado por SUN Microsystem, pero ahora es soportado por casi todas las implementaciones de UNIS, y muchos otros sistemas operativos. Con NFS, usuarios y programas pueden acceder a archivos localizados en sistemas remotos como si fueran archivos locales. En un ambiente NFS perfecto un usuario nunca conoce ni se preocupa donde los archivos estas guardados.

NFS tiene algunos beneficios:

- *Este reduce los requerimientos de almacenamiento de disco por que en la red se puede mantener una sola copia de un directorio, mientras que el directorio puede ser accesible para cualquiera en el red.*
- *NFS simplifica las tareas de soporte centralizadas, porque los archivos pueden ser actualizadas centralmente.*
- *NFS permite a los usuarios el uso de comandos familiares UNIX para manipular archivos remoto en vez de aprender nuevos comandos.*

Hay dos lados en el NFS un lados cliente y otro lado servidor. El cliente es el sistema que usa el directorio remoto como si ellos fueran parte de sus sistema de archivos locales. El servidor es el sistema que hace los directorios disponibles para el uso. El unir un directorio remote a un sistema de archivos locales (Una funcion del cliente) es llamado montaje del directorio. El ofrecer un directorio para un acceso remoto (una funcion del servidor) es llamada exportar un directorio. Frecuentemente un sistema puede correr ambos el cliente y el servidor NFS. a continuacion veremos como configurar un sistema para exportar y montar directorios.

Hay que tener en cuenta que el NFS no es el único sistema de compartición de archivos hay otros dos sistemas que son AT&T's RFS (Remote file sharing), y el Andrew Filesystem (AFS). RFS ha estado disponible bajo el sistema V por algunos años, pero no es ampliamente usado. y AFS esta en ambiente de investigacion y solamente unos cuantos cientos de lugares poseen AFS mientras NFS esta en cientos de miles.

DEMONIOS NFS.

El NFS se corre con la ayuda de algunos demonios, algunos ejecutando funciones de cliente y otros ejecutando funciones de servidor. A continuación se presentan los demonios de NFS y sus respectivas funciones:

nfsd [#servidores] *El demonio nfsd de NFS corre en los servidores NFS. Este demonio sirve los requerimientos NFS del cliente. la opción #servidores especifica cuantos demonios deberían haber comenzado. Ocho son los comunmente usados.*

biod [#servidores] *El demonio de bloque I/O, biod, corre en los cliente NFS. Este demonio maneja el lado cliente de el NFS I/O. #servidores especifica el numero de demonios a ser corridos, y ocho es lo común.*

rpc.lockd *El demonio lock (Bloqueos), rpc.lockd, maneja los requerimeintos de bloqueo. ambos el cliente y el servidor corren este demonio. Los cliente hacen requerimientos de bloqueo de archivos y los servidores permiten esto.*

rpc.statd *El demonio que monitoria el status de la red. es requerido por rpc.lockd para proveer los servicios de monitoreo. En particular, este permite bloqueos a ser reseteados apropiadamente despues de un crash. ambos cliente y servidores corren rpc.statd.*

rpc.mountd *El demonio de montaje procesa los requerimientos de montaje de los clientes. Los servidores NFS corren el demonio rpc.mountd.*

Los demonios necesarios para comenzar NFS se inicializan desde los archivos de arranque. A continuación el ejemplo muestra el tipo de código que esta incluido en el archivo de arranque del cliente. El codigo chequea la existencia de biod, rpc.statd, y rpc.lockd y si ellos están presentes comienza ocho copias de biod y una copia de rpc.statd y rpc.lockd.

```
if [ -f /usr/etc/biod -a -f /usr/etc/rpc.statd  
-a -f /usr/etc/rpc.lockd ]; then  
biod 8;      echo -n `biod`  
rpc.statd & echo -n `statd`  
rpc.lockd & echo -n `lockd`  
fi
```


El servidor NFS corre todos los demonios mostrados mas el nfsd y el rpc.mountd. El tipo de codigo que comienza los demonios adicionales necesarios para un servidor NFS son:

```
if [-f /etc/export ]; then  
    > /etc/xtab  
    exportfs -a  
    nfsd 8 &          echo -n `nfsd`  
    rpc.mountd  
fi
```

Este ejemplo de codigo primero chequea por la existencia de /etc/exports, el cual es el archivo que contiene informacion acerca de directorios que el servidor exporta a sus clientes NFS. Si /etc/exports es encontrado, el código vacía /etc/xtab y corre exportfs. exportfs, usa la informacion a exportar de los directorios especificados, y lista informacion acerca de los directorios exportados en el archivo /etc/xtab. (la opción -a dice a exportfs que exporte todos los directorios que estan en /etc/exports).

Después , el codigo comienza ocho copias de nfsd y una copia de rpc.mountd. El demonio de montaje determina cual directorio debería ser montado para el respectivo requerimiento leyendo el archivo /etc/xtab creado por exportfs.

APENDICE C FORMAS DE ORACLE

A continuación se presentan las distintas formas que fueron hechos para la administración de los datos en la base de datos Oracle. Todos estos fueron hechos en la herramienta FORMS3.0 y nos da una fácil interface para el administrador de los datos y la bases de datos.

A continuación se presenta la herramineta utilizada FORMS3.0 , se accesa a ella por medio del emulador REFLECTION 2 para windows que nos permite emula muy bien el juego de teclas que necesitamos para crear las formas.

FORMA DE ORACLE PARA EL MANEJO DE DATOS DE PERSONAS

The screenshot shows a window titled "Reflection 2 - SETTINGS.R2W" with a menu bar containing "File", "Edit", "Terminal", "Connection", "Options", "Window", and "Help". The main form area is titled "INGRESO DE PERSONAS" and is divided into three sections: "GOPHER", "INGRESO DE PERSONAS", and "ESPOL". The "INGRESO DE PERSONAS" section contains the following fields:

- ID. PERSONA : [Redacted]
- APELLIDOS : [Redacted]
- NOMBRES : [Redacted]
- UNIDAD : [Redacted] CLASE [Redacted]
- DIRECCION ELEC: [Redacted]
- DOMICILIO : [Redacted]
- FECHA INGRESO : [Redacted] STATUS [Redacted]

At the bottom of the form, there is a "Count: #0" label and a "<Repl ace>" button. The emulator's status bar at the bottom shows "221,22" and "Compose Num Caps Hold".

MANTENCION DE NOTAS DE ESTUDIANTES.

En la mantencion de las notas de los estudiante tenemos un Form que nos liga tanto al estuante como a las materias que este posee de esta manera podemos modificar a las notas que esten erradas , podremos ingresas nuevas materias, y ademas que nos permite una busqueda por estudiante. Este forms lo tendra la persona encargada de los datos.

FORMA PARA LA MANTENCION DE NOTAS DE ESTUDIANTES

Reflection 2 - SETTINGS.R2W
File Edit Terminal Connection Options Window Help

GOPHER MANTENCION DE NOTAS DE LOS ESTUDIANTES ESPOL

ID PERSONA: [REDACTED]

PERSONA : [REDACTED] [REDACTED] [REDACTED]

UNIDAD : [REDACTED] STATUS [REDACTED]

NOTAS DEL ESTUDIANTE

ID MATERIA	NOMBRE MATERIA	1er.P.	2do.P.	NotaF	An°o	Semes.	AP.
[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]

Count: *0 <Replace>

915.19 ? Compose Num Caps Hold

MANTENCION DE MATERIAS

Las materias seran mantenidas al igual que el caso anterior por medio de Forms en este form se mantendran el codigo de la materia , el nombre de la materia , la unidad las horas que seran necesarias para una ayudantia y si esta disponible la ayudantia.

FORMA ORACLE PARA LA MANTENCION DE MATERIAS

The image shows a terminal window titled "Reflection 2 - SETTINGS.R2W". The menu bar includes "File", "Edit", "Terminal", "Connection", "Options", "Window", and "Help". The main window is titled "GOPHER" and "MANTENCION DE LAS MATERIAS". It displays a table with four columns: "CODIGO", "NOMBRE", "UNIDAD", and "HORAS DIS.". The table is currently empty. Below the table, it shows "Count: *0" and a "<Replece>" button. At the bottom of the terminal, there is a status bar with "373.5" and a help icon, and a keyboard shortcut bar with "?", "Compose", "Num", "Copy", and "Hold".

CODIGO	NOMBRE	UNIDAD	HORAS DIS.
--------	--------	--------	------------

Count: *0 <Replece>

MANTENCION DE APLICACION DE BIBLIOTECA

Esta aplicación sirve para la modificación y la actualización de los datos de los libro y sus autores en la biblioteca.

El DEWEY es el código único del libro.

Con estos codigos podemos localizar a libro y a un autor de una manera muy rápida por medio de las formas.

FORMA ORACLE PARA LA MANTENCION DE LA APLICACION DE BIBLIOTECA

Reflection 2 - SETTINGS.R2W
File Edit Terminal Connection Options Window Help

GOPHER MANTENCION DE LA APLICACION DE BIBLIOTECA ESPOL

DEMEY [REDACTED]

TITULO : [REDACTED]

MATERIA: [REDACTED]

AUTOR : [REDACTED]

IDIOMA : [REDACTED] VALOR: [REDACTED]

INGRESO: [REDACTED] INVENTARIO: [REDACTED]

Count: #0 <Replace>

911.16 ? Compose Num Caps Hold

MANTENCION DE TELEFONOS ESPOL

Este form al igual que los anteriores nos permite el acceso a los datos de la base que mantiene a los telefonos de la ESPOL. las facilidades son las mismas que los forms anteriores.

FORMA ORACLE PARA LA MANTENCION DE TELEFONOS

Reflection 2 - SETTINGS.R2W
File Edit Terminal Connection Options Window Help

GOPHER GUIA DE TELEFONOS ESPOL

TELEFONO : [REDACTED]

USUARIO : [REDACTED]

UNIDAD : [REDACTED]

TELEFONO : [REDACTED]

USUARIO : [REDACTED]

UNIDAD : [REDACTED]

Count: #0 <Replace>

029, 21 ? Compose Num Caps Hold

APENDICE D

MANUAL PARA EL USO DEL USUARIO FINAL

El uso de aplicaciones que a continuación se presentan es muy sencillo debido a que el gopher ha sido diseñado para personas que tengan poca experiencia en computación

AYUDANTIAS ACADEMICAS

El form que a continuación se presenta es el utilizado para la solicitud de ayudantía académica.

SOLICITUD DE AYUDANTIA

The image shows a screenshot of a Gopher client window titled "Getting requested Item". The window has a menu bar with "Commands", "Options", "Bookmarks", and "Help". Below the menu bar is a "Previous Menu [Gopher ESPOL]" field. The main content area is titled "Gopher Server Asks" and contains the following text: "ATENCION: Ingrese sus datos sin presionar <RETURN> para movilizarse a través de los campos: presione <TAB> (solo usuarios ESPOL)". The form includes several input fields: "Usuario:" with the value "wajara", "Password:" with "*****", "Numero de Matricula:" with "00891564", "Codigo Materia:" with "1909991", and "Promedio:" with "7.5". There are also radio buttons for "Nivel:" with options 100, 200, 300, 400, and 500 (selected). A checkbox labeled "Desea Enviar" is checked. At the bottom are "Okay" and "Cancel" buttons.

Esta solicitud es llenada primero

Usuario: es el usuario del sistema operativo que se valida para que lo puedan utilizar solo personas que pertenezcan a la ESPOL.

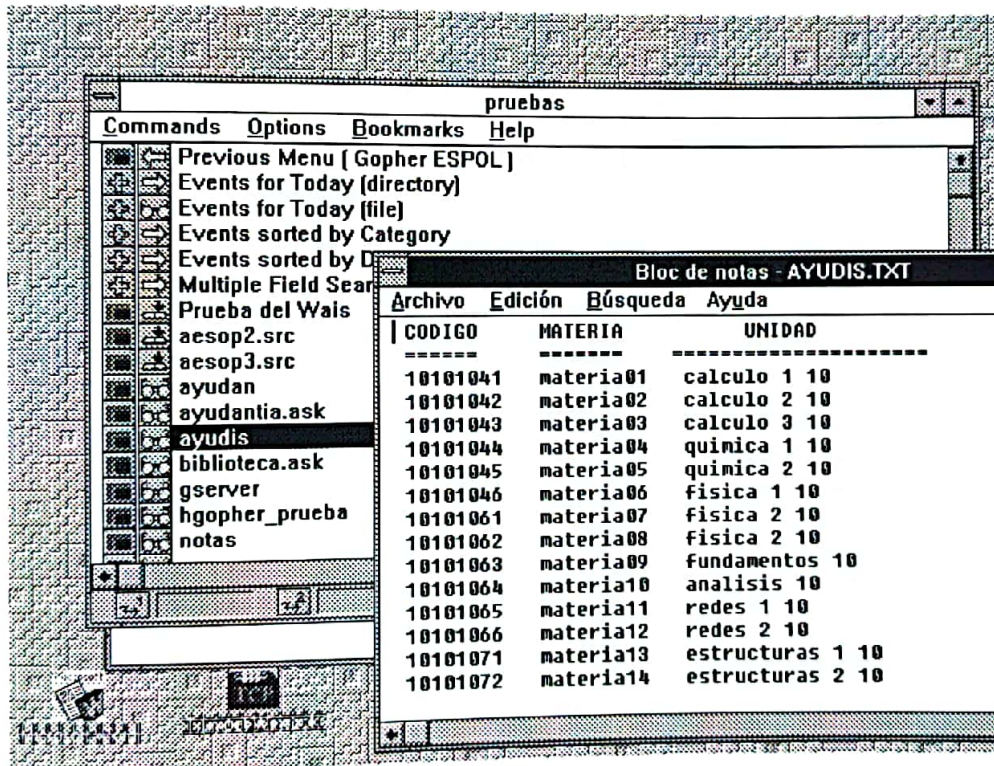
Password: es el password del usuario del sistema operativo.

Numero de Matricula: es el numero de matricula del estudiante.

Nivel: es nivel se lo tiene que ingresar (es informativo).

Código de materia: Es el código de materia que se desea ser ayudante, si no se sabe el código de la materia entonces se lo puede consultar en el forms de consulta de materias.

CONSULTA DE MATERIAS



Promedio: es indicando cual es el promedio del estudiante.

Una vez llenado el forms se procede a enviar la solicitud que va a ser enviada por correo electrónico a la persona encargada de receptor estas solicitudes.

LISTADOS DE CONTABILIDAD A CESERCOMP.

Esta opción tiene como objetivo disminuir el tiempo invertido en el pedido de listados de contabilidad a Cesercomp a continuación se presentan los campos utilizados:

SOLICITUD DE LISTADOS A CESERCOMP

The image shows a screenshot of a Gopher Server dialog box titled "Gopher Server Asks". The dialog contains the following fields and instructions:

- ATENCIÓN:** Ingrese sus datos sin presionar <RETURN> para movilizarse a través de los campos presione <TAB> (solo usuarios ESPOL)
- Usuario:** [Text input field]
- Password:** [Text input field]
- Roll Administrativo:** [Text input field]
- Solicitado por:** [List box with "Contabilidad" selected]
- Código SFBP:** [Text input field]
- Nombre del Listado:** [Text input field]
- DATOS REQUERIDOS:**
 - CIA:** [Text input field]
 - Fecha de Cierre:** [Text input field]
 - Mes Ingreso:** [Text input field]
 - DP:** [Text input field]
 - Compañía:** [Text input field]
- A continuación ingrese en el siguiente formato:**
 - CUENTAS-----** [Text input field]
 - TERCEROS-----** [Text input field]
- Next** and **Cancel** buttons.

Usuario: es usado para validar la entrada de cualquier usuario a esta opción dicho usuario tiene que pertenecer a la ESPOL.

Password: es el password del usuario arriba mencionado.

Roll Administrativo: las personas que usarán este form pertenecerán al area administrativa por lo que poseerán roles administrativos.

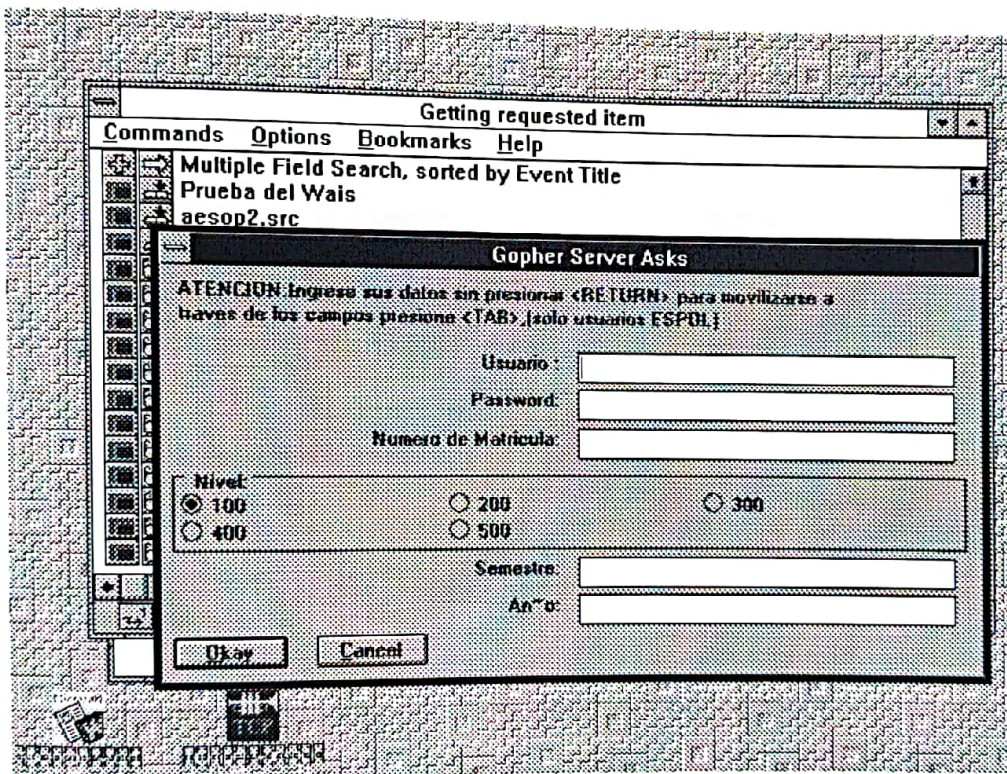
Código SFBP: Es utilizado para la clasificación del listado.

Nombre del listado: igual que el anterior sirve la clasificación del listado.

CONSULTA DE NOTAS

Esta opción nos permite consultar la nota de cualquier estudiante de cualquier facultad, dicha información esta almacenada en la base de datos ORACLE del servidor de Internet.

FORM DE CONSULTA DE NOTAS



The image shows a screenshot of a Gopher client window titled "Getting requested item". The window has a menu bar with "Commands", "Options", "Bookmarks", and "Help". Below the menu bar, there is a search bar containing "Multiple Field Search, sorted by Event Title" and "Prueba del Wais". The current directory path is "aesop2.src". A dialog box titled "Gopher Server Asks" is overlaid on the window. The dialog box contains the following text: "ATENCIÓN: Ingrese sus datos sin presionar <RETURN> para moverse a través de los campos presione <TAB> (solo usuarios ESPOL)". Below this text are several input fields: "Usuario:" with a text box, "Password:" with a text box, "Número de Matrícula:" with a text box, "Nivel:" with radio buttons for 100, 200, 300, 400, and 500, "Semestre:" with a text box, and "Año:" with a text box. At the bottom of the dialog box are "Ok" and "Cancel" buttons.

Los campos requeridos son:

Usuario: es el usuario del sistema operativo (es un filtro de seguridad para que cualquier persona no pueda ingresar);

Password: es el password del usuario del sistema operativo

Número de matrícula: es el número de matrícula de la persona que solicita información.

Semestre: es el semestre del cual se quiere pedir las notas.

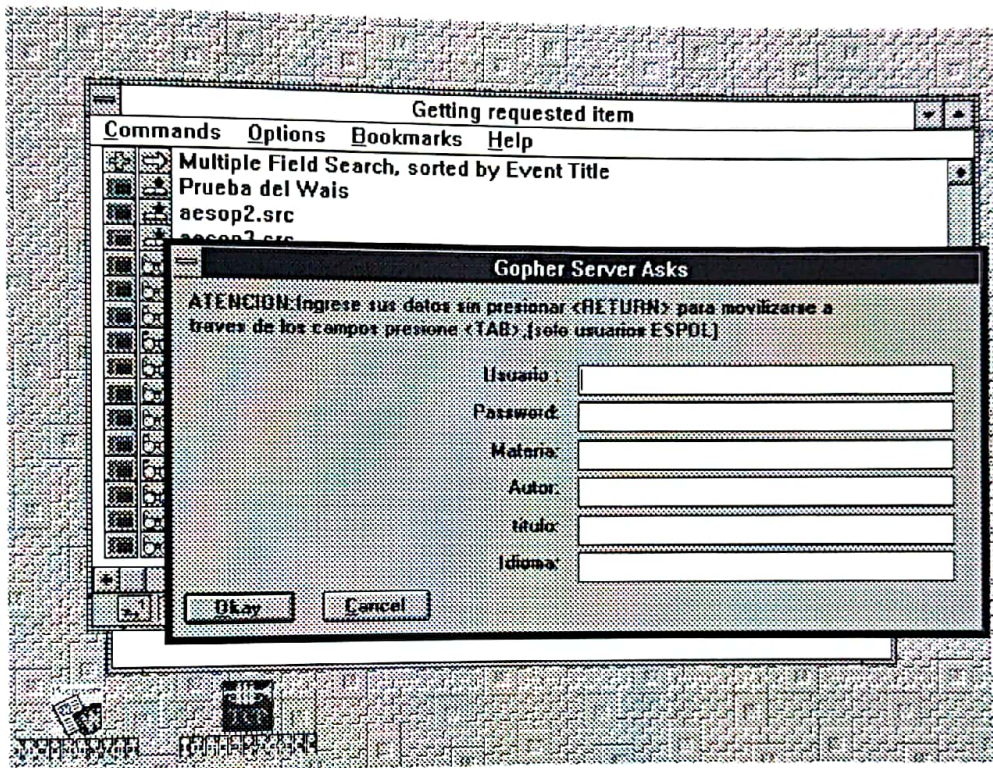
Año: es el año del cual se solicita la información.

En cualquiera de estos casos del semestre y del año al poner nulo trae por default todos los datos del estudiante.

CONSULTA BIBLIOTECA

La consulta de biblioteca nos da la facilidad poder acceder con patrones de búsqueda que van a ser recogidos por Oracle y los resultados serán devueltos al usuario.

FORMS PARA BUSQUEDA DE LIBROS



Los campos requeridos son

Usuario: es el usuario del sistema operativo (es un filtro de seguridad para que cualquier persona no pueda ingresar);

Password: es el password del usuario del sistema operativo

Materia: al ingresar por ejemplos solo FIS el sistema traerá todos los libros de las materias que comiencen con FIS por ejemplos : FISICA, FISICA NUCLEAR, etc.

Autor: Lo mismo pasará con autor al ingresar solo el nombre KENNEDY traerá todos los libros que tengan autor KENNEDY.

Título: se refiere a los títulos de los libro y se comporta de igual manera que los campos anteriores.

Cualquiera de estos campos pueden ser combinados para traer el resultado de varios criterios de seleccion por ejemplos al poner en materia FIS y en autor KENNEDY se traerán todos los libros que tengan autor Kennedy y que la Materia a la cual pertenecen comience con FIS.

DIRECTORIO TELEFONICO

Sirve para dar la información sobre la unidad y el número telefónico de un empleado o profesor de la ESPOL.

FORM PARA SOLICITAR INFORMACION SOBRE TELEFONOS

The screenshot shows a Gopher client window titled "Getting requested item". The window has a menu bar with "Commands", "Options", "Bookmarks", and "Help". Below the menu bar, there are three menu items: "Previous Menu [Informacion Administrativa ESPOL]", "Guia Telefonica de la ESPOL", and "Numeros Telefonicos [ARCHIVO]". The main content area displays a dialog box titled "Gopher Server Asks". The dialog box contains the following text: "ATENCION: Ingrese sus datos sin presionar <RETURN> para moverse a traves de los campos presione <TAB> (solo usuarios ESPOL)". Below this text are three input fields labeled "Nombre:", "Unidad:", and "Telefono:". At the bottom of the dialog box are two buttons: "Okay" and "Cancel". The window also has a status bar at the bottom with several icons.

Los campos requeridos son:

Nombre: se ingresa el nombre o apellido de la persona de la cual se quiere saber el número telefónico.

Unidad: si uno quiere saber el número de la persona en una unidad determinada.

Telefono: si uno sabe el telefono pero quiere saber a quien pertenece.

INFORMACION PERSONAS

Con esta opción uno podrá saber información sobre alguien en particular como sería su dirección electrónica, unidad a la que pertenece, etc.

FORM PARA SOLICITAR INFORMACION DE UNA PERSONA

Getting requested item

Commands Options Bookmarks Help

Multiple Field Search, sorted by Event Title

Prueba del Wais

acsop2.src

acsop3.src

Gopher Server Asks

ATENCIÓN: Ingrese sus datos sin presionar <RETURN> para movilizarse a través de los campos presione <TAB> (solo usuarios ESPOL)

Apellido Paterno:

Apellido Materno:

Nombre:

CARGO:

ALUMNO PROFESOR TRABAJADOR

Unidad:

Okay Cancel

Los campos que se requieren son:

Apellido Paterno: Uno puede buscar a la persona por el apellido paterno.

Apellido Materno: Tiene la misma función que el anterior.

Nombre: Podremos buscar a las personas por su nombre.

Cargo: podremos traer información de todos los profesores, alumnos o trabajadores.

Unidad: se podrán traer información sobre cualquier unidad y las personas que pertenecen a esa unidad.

APENDICE E

MANUAL PARA ENTRENAMIENTO DEL ADMINISTRADOR DEL GOPHER

Este manual se lo va a dividir en dos secciones:

- La primera se referira a la instalacion del paquete gopher*
- La segunda tratará el mantenimiento del gopher.*

INSTALACION.

*Todos los paquetes de instalación del gopher y todos sus accesorios fueron traídos mediante el **FILE TRANSFER PROTOCOL (FTP)** de la universidad de Minnesota cuya dirección electrónica es boombox.micro.umn.edu.*

*La herramienta gopher como casi todas las herramientas en INTERNET son primero empaquetadas con el comando **tar** y este a su vez es comprimido por medio del **compress** del UNIX. Una vez desempaquetado se procede a la compilación del paquete. En esta parte tenemos que setear algunos parámetros en los archivos a los cuales se va a tener acceso durante la compilación estos archivos son el **conf.h** y el **makefile.config**. A continuación pondremos cada una de las variables principales que se necesitan:*

MAKEFILE.CONFIG

***PREFIX** = /home/user/gopher*

Este es el directorio raíz donde van a ser instalados los ejecutables ya compilados.

***CLIENTDIR** = \$(PREFIX)/bin*

Este directorio es donde van a ser instalados los ejecutables del cliente gopher.

***CLIENTLIB** = \$(PREFIX)/lib*

Es donde estarán todas las librerías

***SERVERDIR** = \$(PREFIX)/etc*

Aquí se mantendrán los archivos de importancia del servidor gopher

```
MAN1DIR = $(PREFIX)/man/man1  
MAN5DIR = $(PREFIX)/man/man5  
MAN8DIR = $(PREFIX)/man/man8
```

Es donde están los archivos de ayuda del gopher servidor y cliente.

```
DOMAIN =
```

si al poner el comando hostname' este retorna la dirección del servidor entonces lo dejamos como nulo.

```
SERVERDATA = /home/apoyo/gopherda/gopher-data
```

Se refiere al directorio donde va a quedar almacenada la información del gopher.

```
SERVERPORT = 70
```

Esta variable setea el puerto TCP/IP al cual el cliente tendrá que conectarse para acceder al servidor.

```
CONF.H
```

```
CLIENT1_HOST "espol.edu.ec"  
CLIENT1_PORT 70
```

Estas variables son las que van a mantener el gopher default al cual tienen que conectarse los clientes.

```
CLIENT1_HOST "espol.edu.ec"  
CLIENT1_PORT 0
```

Estas variables son las que van a mantener el nombre y el puerto del segundo gopher al cual pueden conectarse los clientes, si no hay otro gopher al cual puedan conectarse se le pone el puerto 0.

además en este archivo hay muchos otros variables que se la definen para que el cliente ejecute los distintos comandos.

com por ejemplo:


```
#define HTML_COMMAND "lynx -force_html %s"
```

que el la sentencia para invocar al lynx que es un editor de hipertexto para modo ascii.

Después de haber estos seteos o los que uno desee se ejecuta el comando
make all

Este comando lo que hará es la ejecución de los distintos pasos para la compilación, luego enlaza los programas objetos y los instala en los respectivos directorios.

MANTENCION

La mantención de los datos del gopher es muy sencilla ya que se almacena la información en un sistema de archivos jerárquico, que son los que se usan en DOS y UNIX con sus respectivos directorios, subdirectorios y archivos por lo que cada item en los menus equivalen a un archivo en el directorio.

El gopher soporta distintos tipos de datos que más adelante se detallan. En cada menu (directorio) tenemos un subdirectorio llamado .cap en el cual se ponen otros archivos que tiene los mismos nombres de los archivos de datos. Cada uno de estos archivos tienen parámetros que se utilizan ya sea para poner un nombre más descriptivo o para ubicarlo en el menú.

Los parámetros son:

Name =

Aqui se puede poner el nombre que aparecerá en el menú principal.

Type =

Es el tipo de dato que tiene el archivo esto es:

0 Item es un archivo

1 Item es un directorio

2 Item es un servidor CSO

3 Error

4 Item es un archivo binario Macintosh

- 5 Item es un archivo binario DOS*
- 6 Item es un archivo Unix uuencoded*
- 7 Item es un servidor de búsqueda de texto completo*
- 8 Item apunta a una sesión telnet*
- 9 Item es un archivo binario*
- T Item apunta a una sesión de texto tn3270*
- g Item es un archivo gráfico del formato GIF*
- I Item es alguna clase de archivo imagen.*

Numb=

Nos indica la posición en la cual va ubicada en el menú.

Hosts=

Es el host al cual va a comunicarse.

Port=

Es el puerto al cual va a comunicarse

Como se puede ver la mantención de este archivo nos daría la interfase con el usuario . El gopher es muy fácil de usar y tiene un mantenimiento también muy fácil . Hay que tomar en cuenta que todos los archivos que se colocan son copias imágenes de los originales.

BIBLIOGRAFIA

1. **LINDNER PAUL**, *Protocolo Gopher* , Universidad de Minnesota, 1991.
2. **KAHLE BREWSTER**, *Overview WAIS, Thinking Machine*, 1992.
3. **DORNER STEVEN**, *Description CSO, Universidad de Carolina del Norte*, 1990
4. **WALL LARRY**, *Perl Kit*, 1991.
5. **STOCK KEVIN**, *Oraperl*, 1993