



REPÚBLICA DEL ECUADOR

Escuela Politécnica Nacional

" E S C I E N T I A H O M I N I S S A L U S "

La versión digital de esta tesis está protegida por la Ley de Derechos de Autor del Ecuador.

Los derechos de autor han sido entregados a la "ESCUELA POLITÉCNICA NACIONAL" bajo el libre consentimiento del (los) autor(es).

Al consultar esta tesis deberá acatar con las disposiciones de la Ley y las siguientes condiciones de uso:

- Cualquier uso que haga de estos documentos o imágenes deben ser sólo para efectos de investigación o estudio académico, y usted no puede ponerlos a disposición de otra persona.
- Usted deberá reconocer el derecho del autor a ser identificado y citado como el autor de esta tesis.
- No se podrá obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de licencia que el trabajo original.

El Libre Acceso a la información, promueve el reconocimiento de la originalidad de las ideas de los demás, respetando las normas de presentación y de citación de autores con el fin de no incurrir en actos ilegítimos de copiar y hacer pasar como propias las creaciones de terceras personas.

Respeto hacia sí mismo y hacia los demás.

ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UN PROTOTIPO DE SOLUCIÓN INTEGRAL WEB PARA LA ADMINISTRACIÓN DE UNA RED TIPO SOHO (SMALL OFFICE HOME OFFICE) BAJO LA PLATAFORMA GNU/LINUX

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN
ELECTRÓNICA Y REDES DE INFORMACIÓN

BUELE GUALOTO CHRISTIAN DANIEL
christian_danny@hotmail.com

RUIZ CERÓN MARIANELA
maya_md22@hotmail.com

DIRECTOR: ING. XAVIER CALDERÓN HINOJOSA, MSC
xavier.calderon@epn.edu.ec

Quito, febrero 2011

DECLARACIÓN

Nosotros, Christian Daniel Buele Gualoto y Marianela Ruiz Cerón, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Christian Daniel Buele Gualoto

Marianela Ruiz Cerón

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Christian Daniel Buele Gualoto y Marianela Ruiz Cerón, bajo mi supervisión.

ING. XAVIER CALDERÓN, MSC.

DIRECTOR DEL PROYECTO

AGRADECIMIENTO

Para toda mi familia en especial a mis queridos padres **Inés Gualoto y Elías Buele**, por creer en mí y brindarme toda la ayuda posible cuando la necesité.

Para mis hermanas Paulina, Belén y mi hermano Andrés, por su apoyo incondicional y constituir un soporte en los momentos de alegría y tristeza. Gracias a Paulina y Belén por brindarme la ayuda necesaria para la culminación del presente proyecto.

Un sincero agradecimiento al Ing. Xavier Calderón por su predisposición y ayuda en el desarrollo y culminación del presente proyecto de titulación.

A todos los profesores que formaron parte de mi vida universitaria por inculcar en mí los dones de la responsabilidad, dedicación y perseverancia.

A mis compañeros con quienes compartí muchos momentos agradables, a mis amigos Carlos y Roberto por ser un apoyo y ayuda tanto en mi vida personal como en mi vida universitaria.

A Marianela Ruiz por ser una excelente compañera y amiga, en especial por brindarme el aliento necesario para seguir adelante en los momentos de adversidad.

Christian Daniel Buele Gualoto.

A Dios por ser mi guía e inspiración en éste arduo camino, por estar siempre conmigo en los momentos más difíciles, por su amor infinito.

A mis padres **Raúl Ruiz** y **Leonor Cerón** por su amor incondicional, por demostrarme con su ejemplo, que con dedicación y esfuerzo se puede llegar a alcanzar mis sueños.

A mi hermanita Caty que ha sido más que eso, mi mejor amiga, gracias por tus palabras de apoyo, por comprenderme siempre, porque con tus locuras has hecho que mi vida sea diferente.

A mis abuelitos Aníbal Cerón, Clemencia Flores, Luis Ruiz y Clementina Sarauz, que aunque ya no se encuentren a mi lado, con su ejemplo de vida han sembrado en mí la fortaleza necesaria para seguir adelante.

A mis Tíos y Tías por su cariño y apoyo.

A todos mis profesores, en especial al Ing. Xavier Calderón, por su tiempo, paciencia y guía en la elaboración del presente proyecto.

A Christian Buele por su cariño, apoyo y motivación, herramientas que me inspiraron en todo éste largo trayecto, por compartir conmigo los buenos momentos pero sobre todo aquellos que parecían difíciles de superar.

A mis amigas y amigos por estar siempre a mi lado, por sus acertados consejos, por celebrar mis alegrías y compartir mis tristezas, pero sobre todo por brindarme una amistad verdadera.

Marianela Ruiz Cerón.

DEDICATORIA

El presente proyecto de titulación está dedicado a mis Padres:

Inés Gualoto, aunque no alcanzaste a ver a tu querido hijo cumpliendo uno de sus sueños, quiero que sepas que este trabajo es dedicado para ti por ser una madre ejemplar y por quien nunca dejare de alcanzar mis sueños y ser quien soy, se que al recordar tu mirada mi corazón se alegra y toma un impulso enorme que me da la fuerza necesaria para seguir adelante... *te amo mucho madre....*

Elías Buele, por ser un gran hombre, brindarme todo el cariño y ayuda siempre que la necesite desde niño hasta la actualidad, este es un logro para ti por ser un ejemplo de vida, si mi corazón se encuentra lleno de felicidad el tuyo lo debe estar mil veces más... *te amo mucho padre.....*

Christian Daniel Buele Gualoto.

A mis padres, por todo el amor y dedicación con el que me educaron. A mi mami Leo por enseñarme que en mi diccionario no debe existir la palabra imposible, que todo se logra con mucho esfuerzo, por enseñarme a levantarme después de cada tropiezo, gracias por regalarme siempre una sonrisa un abrazo cálido, pero sobre todo por enseñarme a valorar las cosas más pequeñas y sencillas de la vida. A mi papi Raúl por estar siempre a mi lado, por arrancarme una sonrisa cuando he estado triste, por inculcarme valores que siempre llevaré conmigo. Para ustedes que son el pilar fundamental en mi vida y hacen que me sienta muy afortunada de ser su hija.

Marianela Ruiz Cerón.

CONTENIDO

CONTENIDO	I
ÍNDICE DE FIGURAS	IV
ÍNDICE DE TABLAS	IX
RESUMEN	XI
PRESENTACIÓN	XIII
CAPÍTULO 1	1
1 FUNDAMENTOS TEÓRICOS	1
1.1 INTRODUCCIÓN A LA ADMINISTRACIÓN DE SERVICIOS VÍA WEB	1
1.1.1 LA WEB	2
1.1.1.1 Hipertexto e Hipermedia	2
1.1.1.2 Visión general del uso de la Web	5
1.1.2 TIPOS BÁSICOS DE DOCUMENTOS WEB	11
1.2 EL WEBMASTER	12
1.2.1 FUNCIONES Y RESPONSABILIDADES DEL WEBMASTER	13
1.2.2 TIPOS DE WEBMASTER	14
1.2.3 SELECCIÓN DEL WEBMASTER	15
1.2.4 ALTERNATIVAS AL WEBMASTER	16
1.3 SERVIDOR WEB	16
1.3.1 FUNCIONAMIENTO DEL SERVIDOR WEB	17
1.3.2 ARQUITECTURA	18
1.3.3 TIPOS DE SERVIDORES WEB	20
1.3.3.1 Servidores Basados en Procesos	20
1.3.3.2 Servidores Basados en Hilos	20
1.3.3.3 Servidores Basados en Sockets dirigidos por Eventos	21
1.3.4 CÓDIGOS DE RESPUESTA DEL SERVIDOR WEB	21
1.3.5 HARDWARE Y SOFTWARE	22
1.3.5.1 Hardware	22
1.3.5.2 Software	23
1.4 HTML (HYPER TEXT MARKUP LANGUAGE O LENGUAJE DE MARCADO DE HIPERTEXTO)	24
1.4.1 CARACTERÍSTICAS BÁSICAS	25
1.4.1.1 HTML un Lenguaje de Etiquetado	25
1.4.1.2 Atributos	27
1.4.1.3 Sintaxis en HTML	30
1.4.1.4 Estructura básica HTML	31
1.4.2 TEXTO (párrafos, líneas, frases)	33
1.4.2.1 Párrafos y Líneas	33
1.4.2.2 Texto estructurado	38
1.4.2.3 Codificación de caracteres en HTML	40
1.4.3 LISTAS	41
1.4.4 ENLACES	43
1.4.4.1 URI (Uniform Resource Identifier o Identificador Uniforme de Recurso)	44
1.4.4.2 El elemento <code><a>...</code>	47
1.4.4.3 Otros elementos de enlace	50
1.4.5 IMÁGENES Y OBJETOS	52
1.4.5.1 Imagen <code></code>	52
1.4.6 TABLAS	54
1.4.7 HOJAS DE ESTILO	58
1.4.7.1 Añadir estilos a HTML	59
1.4.7.2 Hojas de estilo externa	65
1.5 TECNOLOGÍAS PARA PÁGINAS WEB DINÁMICAS DEL LADO DEL SERVIDOR	68
1.5.1 ASP (Active Server Pages o Páginas de Servidor Activo)	68
1.5.2 CGI (Common Gateway Interface o Interfaz de Acceso Común)	68
1.5.3 JSP (Java Server Pages o Página de Servidor Java)	69
1.5.4 WEB SERVICES	69
1.5.5 SELECCIÓN DE LA TECNOLOGÍA WEB DEL LADO DEL SERVIDOR	69

1.6	CGI COMO TECNOLOGÍA PARA PÁGINAS WEB DINÁMICAS DEL LADO DEL SERVIDOR.....	70
1.6.1	NORMA CGI.....	70
1.6.2	ESPECIFICACIONES DE LA INTERFAZ CGI.....	71
1.6.3	SALIDA DE UN PROGRAMA CGI.....	72
1.6.4	TIPOS DE CGI.....	75
1.6.5	FORMULARIOS HTML EN CGI.....	76
1.6.5.1	Funcionamiento de un formulario.....	76
1.6.5.2	Elementos de un formulario.....	77
1.6.5.3	Métodos de Envío de un Formulario.....	92
1.6.6	EJECUCIÓN DE UN PROGRAMA CGI.....	93
1.6.7	SEGURIDAD EN LOS PROGRAMAS CGI.....	95
1.7	LENGUAJES PARA LA PROGRAMACION DE CGI.....	96
1.7.1	SCRIPT DE BASH.....	97
1.7.2	APPLESCRIPT.....	98
1.7.3	C/C++.....	100
1.7.4	VISUAL BASIC.....	104
1.7.5	PERL.....	106
1.8	DEFINICIÓN DE UNA RED TIPO SOHO.....	108
1.8.1	REQUERIMIENTOS EN UNA RED TIPO SOHO.....	109
1.9	METODOLOGÍAS DE DESARROLLO.....	111
1.9.1	METODOLOGÍAS TRADICIONALES.....	113
1.9.2	METODOLOGÍAS ÁGILES.....	114
1.9.3	DIFERENCIA ENTRE METODOLOGÍAS.....	115
1.9.4	SELECCIÓN DEL GRUPO DE METODOLOGÍA.....	116
1.9.5	TIPOS DE METODOLOGÍAS ÁGILES.....	117
1.9.5.1	Scrum.....	117
1.9.5.1.1	Roles en Scrum.....	117
1.9.5.1.2	Artefactos o Documentos en Scrum.....	118
1.9.5.2	Crystal Clear.....	118
1.9.5.2.1	Roles en Crystal Clear.....	119
1.9.5.2.2	Propiedades o valores de Crystal Clear.....	120
1.9.5.3	XP (eXtreme Programming o Programación Extrema).....	120
1.9.5.3.1	Características o valores de la metodología XP.....	121
1.9.5.3.2	Fases en la metodología XP.....	122
1.9.5.4	ASD (Adaptive Software Development/Desarrollo de Software Adaptable).....	125
1.9.5.4.1	Ciclo de vida en ASD.....	125
1.9.6	COMPARACIÓN DE METODOLOGÍAS ÁGILES.....	127
1.9.7	SELECCIÓN DE LA METODOLOGÍA ÁGIL A EMPLEAR.....	128
CAPÍTULO 2	130
2	REQUERIMIENTOS HARDWARE Y SOFTWARE.....	130
2.1	ESPECIFICACIÓN DE REQUISITOS SEGÚN EL ESTÁNDAR IEEE 830-1998.....	130
2.1.1	INTRODUCCIÓN.....	130
2.1.1.1	Propósito.....	130
2.1.1.2	Ámbito del Sistema.....	131
2.1.1.3	Definiciones, Acrónimos y Abreviaturas.....	132
2.1.1.4	Referencias.....	132
2.1.1.5	Visión General de Documento.....	134
2.1.2	DESCRIPCIÓN GENERAL.....	134
2.1.2.1	Perspectiva del Producto.....	134
2.1.2.2	Funciones del Producto.....	145
2.1.2.2.1	Configuración del Servicio DHCP.....	145
2.1.2.2.2	Configuración del Servicio Proxy – Caché.....	145
2.1.2.2.3	Configuración del Servicio de Directorios Compartidos.....	145
2.1.2.2.4	Configuración del Servicio de Filtrado Web.....	146
2.1.2.2.5	Configuración del Servicio de Firewall.....	146
2.1.2.2.6	Configuración del Servicio de Estado del Sistema Operativo.....	146
2.1.2.2.7	Configuración del Servicio de Gráficas.....	147

2.1.2.2.8	Página Inicial	147
2.1.2.3	Características de los Usuarios	147
2.1.2.4	Restricciones.....	147
2.1.2.5	Suposiciones y Dependencias.....	148
2.1.2.6	Requisitos Futuros	148
2.1.3	REQUISITOS ESPECÍFICOS	148
2.1.3.1	Interfaces Externas	148
2.1.3.2	Funciones.....	148
2.1.3.3	Requisitos de Rendimiento	168
2.1.3.4	Restricciones de Diseño	169
2.1.3.5	Atributos del Sistema	169
CAPÍTULO 3	171
3	DESARROLLO E IMPLEMENTACIÓN DE LA INTERFAZ WEB	171
3.1	DESARROLLO E IMPLEMENTACIÓN	171
3.1.1	<i>Planificación del Proyecto</i>	171
3.1.1.1	Historia de Usuarios.....	171
3.1.1.2	Planificación de Entregables.....	171
3.1.1.3	Análisis de Iteraciones	174
3.1.2	<i>Velocidad del Proyecto y Programación en Pareja</i>	180
3.1.3	<i>Diseño</i>	180
3.1.3.1	Estructura de Diseño Web	181
3.1.3.1.1	Diseño de la Estructura	182
3.1.3.2	Interfaces	185
3.1.3.2.1	Página de Inicio	185
3.1.3.2.2	Módulo del Servicio de DHCP	186
3.1.3.2.3	Módulo del Servicio Proxy / Caché	188
3.1.3.2.4	Módulo del Servicio de Directorios Compartidos	190
3.1.3.2.5	Módulo del Servicio de Filtrado Web	193
3.1.3.2.6	Módulo del Servicio de Firewall	195
3.1.3.2.7	Módulo del Servicio de Estado del Sistema Operativo	198
3.1.3.2.8	Módulo del Servicio de Gráficas.....	202
3.1.4	<i>Codificación</i>	204
3.1.4.1	Módulo de Servicio DHCP	205
3.1.4.2	Codificación del Módulo de Servicio Proxy –Caché	210
3.1.4.3	Codificación del Módulo de Servicio de Directorios Compartidos	216
3.1.4.4	Codificación del Módulo de Servicio de Filtrado Web	220
3.1.4.5	Codificación del Módulo de Servicio de Firewall	223
3.1.4.6	Codificación del Módulo de Servicio de Estado del Sistema Operativo	229
3.1.4.7	Codificación del Módulo de Servicio de Gráficas	229
3.1.4.8	Codificación del Módulo de Página Inicial	233
CAPÍTULO 4	237
4	PRUEBAS Y EVALUACIÓN DE LA INTERFAZ WEB	237
4.1	IMPLEMENTACIÓN DE UNA RED LOCAL TIPO SOHO	237
4.2	PRUEBAS Y EVALUACIÓN.....	238
4.2.1	<i>PRUEBAS DE ACEPTACIÓN</i>	238
CAPÍTULO 5	274
5	CONCLUSIONES Y RECOMENDACIONES	274
5.1	CONCLUSIONES	274
5.2	RECOMENDACIONES.....	277
REFERENCIAS BIBLIOGRÁFICAS	279
A. ANEXO A	283
B. ANEXO B	300

ÍNDICE DE FIGURAS

FIGURA 1.1 HIPERTEXTO.....	3
FIGURA 1.2 FLUJO DE HIPERTEXTO.....	3
FIGURA 1.3 HIPERMEDIA.....	4
FIGURA 1.4 MODELO CLIENTE/SERVIDOR WEB.....	6
FIGURA 1.5 HTTP.....	7
FIGURA 1.6 EL WEBMASTER EN LA INTRANET.....	12
FIGURA 1.7 RELACIÓN CLIENTE – SERVIDOR.....	16
FIGURA 1.8 COMUNICACIÓN CLIENTE – SERVIDOR.....	17
FIGURA 1.9 FUNCIONAMIENTO DE UN SERVIDOR WEB.....	18
FIGURA 1.10 ARQUITECTURA DE UN SERVIDOR WEB.....	18
FIGURA 1.11 EJEMPLO DE UN ATRIBUTO.....	27
FIGURA 1.12 ELEMENTO HTML.....	29
FIGURA 1.13 VISUALIZACIÓN DE ELEMENTOS EN BLOQUE Y LÍNEA.....	30
FIGURA 1.14 CÓDIGO HTML DE ELEMENTOS EN BLOQUE Y LÍNEA.....	30
FIGURA 1.15 ELEMENTOS HTML BÁSICOS.....	33
FIGURA 1.16 PÁGINA WEB CON ELEMENTOS BÁSICOS.....	33
FIGURA 1.17 EL ELEMENTO PÁRRAFO.....	34
FIGURA 1.18 VISUALIZACIÓN DE UN PÁRRAFO.....	34
FIGURA 1.19 ENCABEZADOS O TÍTULOS.....	35
FIGURA 1.20 VISUALIZACIÓN DE TÍTULOS.....	35
FIGURA 1.21 AÑADIR UN SALTO DE LÍNEA.....	36
FIGURA 1.22 VISUALIZACIÓN DE UN SALTO DE LÍNEA.....	36
FIGURA 1.23 INSERCIÓN DE ESPACIOS EN BLANCO.....	37
FIGURA 1.24 VISUALIZACIÓN ESPACIOS EN BLANCO.....	37
FIGURA 1.25 AÑADIR TEXTO PREFORMATEADO.....	38
FIGURA 1.26 VISUALIZACIÓN DE TEXTO PRE FORMATEADO.....	38
FIGURA 1.27 ÉNFASIS EN EL TEXTO.....	39
FIGURA 1.28 VISUALIZACIÓN DE ÉNFASIS A MANERA DE LETRA CURSIVA.....	39
FIGURA 1.29 ÉNFASIS FUERTE EN EL TEXTO.....	39
FIGURA 1.30 VISUALIZACIÓN DE ÉNFASIS FUERTE COMO TEXTO EN NEGRITA.....	40
FIGURA 1.31 CODIFICACIÓN DE ESPACIOS EN BLANCO.....	41
FIGURA 1.32 GENERACIÓN DE LISTAS ORDENADAS.....	42
FIGURA 1.33 VISUALIZACIÓN DE LISTAS ORDENADAS.....	42
FIGURA 1.34 GENERACIÓN DE LISTAS NO ORDENADAS.....	43
FIGURA 1.35 VISUALIZACIÓN DE LISTAS NO ORDENADAS.....	43
FIGURA 1.36 URI (UNIFORM RESOURCE IDENTIFIER O IDENTIFICADOR UNIFORME DE RECURSO).....	44
FIGURA 1.37 CONSULTAS Y SECCIÓN.....	45
FIGURA 1.38 URI ABSOLUTO.....	45
FIGURA 1.39 URI RELATIVO.....	46
FIGURA 1.40 INSERCIÓN DE SECCIONES EN UN DOCUMENTO HTML.....	47
FIGURA 1.41 DESPLAZAMIENTO EN UNA PÁGINA WEB EMPLEANDO SECCIONES.....	48
FIGURA 1.42 INSERCIÓN DE SECCIONES EMPLEANDO EL ATRIBUTO ID.....	49
FIGURA 1.43 DESPLAZAMIENTO EN UNA PÁGINA WEB EMPLEANDO SECCIONES.....	50
FIGURA 1.44 EJEMPLO DE INSERCIÓN DE JAVASCRIPT EN UN DOCUMENTO HTML.....	51
FIGURA 1.45 EJEMPLO DE INSERCIÓN DE UNA HOJA DE ESTILO EN UN DOCUMENTO HTML.....	52
FIGURA 1.46 INGRESO DE UNA IMAGEN A UN DOCUMENTO HTML.....	54
FIGURA 1.47 VISUALIZACIÓN DE LA IMAGEN INSERTADA.....	54
FIGURA 1.48 VISUALIZACIÓN DEL TEXTO ALTERNATIVO A UNA IMAGEN.....	54
FIGURA 1.49 IMPLEMENTACIÓN DE UNA TABLA EN UN DOCUMENTO HTML.....	57
FIGURA 1.50 VISUALIZACIÓN DE LA TABLA CREADA.....	58
FIGURA 1.51 ESTILO PARA UNA TABLA EMPLEANDO EL ATRIBUTO STYLE.....	59
FIGURA 1.52 ESTILO APLICADO.....	60
FIGURA 1.53 ESTILO A UN ELEMENTO QUE APARECE VARIAS VECES.....	61
FIGURA 1.54 ESTILO APLICADO A TODAS LAS CELDAS.....	61
FIGURA 1.55 ESTILO EMPLEANDO EL ATRIBUTO CLASS.....	62
FIGURA 1.56 ESTILO APLICADO A LA SEGUNDA FILA.....	62

FIGURA 1.57 ESTILO EMPLEANDO EL ATRIBUTO ID.	63
FIGURA 1.58 ESTILO APLICADO A LA PRIMERA CELDA DE LA TERCERA FILA.	63
FIGURA 1.59 ESTILO PARA UN ELEMENTO EN LÍNEA.	64
FIGURA 1.60 ESTILO APLICADO AL TÍTULO DE LA TABLA.	64
FIGURA 1.61 ESTILO PARA UN ELEMENTO EN BLOQUE.	65
FIGURA 1.62 ESTILO EN BLOQUE APLICADO A DOS PÁRRAFOS.	65
FIGURA 1.63 ARCHIVO DE ESTILO EXTERNO CON EXTENSIÓN CSS.	67
FIGURA 1.64 REFERENCIA HACIA EL ARCHIVO CSS EXTERNO.	67
FIGURA 1.65 APLICACIÓN DEL ESTILO MEDIANTE UNA HOJA DE ESTILO EXTERNA.	67
FIGURA 1.66 CÓDIGO DE UN PROGRAMA CGI.	75
FIGURA 1.67 SALIDA DE UN PROGRAMA CGI.	75
FIGURA 1.68 VISUALIZACIÓN DE LA ETIQUETA <FORM>.	78
FIGURA 1.69 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <FORM>.	78
FIGURA 1.70 EJEMPLO DE ETIQUETA INPUT (TYPE="TEXT").	80
FIGURA 1.71 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "TEXT">.	80
FIGURA 1.72 EJEMPLO DE ETIQUETA INPUT (TYPE="PASSWORD").	81
FIGURA 1.73 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "PASSWORD">.	81
FIGURA 1.74 EJEMPLO DE ETIQUETA INPUT (TYPE="CHECKBOX").	82
FIGURA 1.75 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "CHECKBOX">.	82
FIGURA 1.76 EJEMPLO DE ETIQUETA INPUT (TYPE="RADIO").	83
FIGURA 1.77 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "RADIO">.	83
FIGURA 1.78 EJEMPLO DE ETIQUETA INPUT (TYPE="SUBMIT").	84
FIGURA 1.79 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "SUBMIT">.	84
FIGURA 1.80 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "HIDDEN">.	85
FIGURA 1.81 EJEMPLO DE ETIQUETA INPUT (TYPE="IMAGE").	85
FIGURA 1.82 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "IMAGE">.	86
FIGURA 1.83 EJEMPLO DE ETIQUETA INPUT (TYPE="BUTTON").	86
FIGURA 1.84 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <INPUT TYPE= "BUTTON">.	87
FIGURA 1.85 EJEMPLO DE ETIQUETA SELECT.	88
FIGURA 1.86 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <SELECT>.	88
FIGURA 1.87 EJEMPLO DE ETIQUETA OPTION Y SELECT.	89
FIGURA 1.88 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <OPTION>.	89
FIGURA 1.89 EJEMPLO DE ETIQUETAS <LABEL> <LEGEND> Y <FIELDSET>.	90
FIGURA 1.90 PROGRAMA CGI QUE INCLUYE LAS ETIQUETAS <LABEL> <LEGEND> Y <FIELDSET>.	91
FIGURA 1.91 EJEMPLO DE ETIQUETA TEXTAREA.	92
FIGURA 1.92 PROGRAMA CGI QUE INCLUYE LA ETIQUETA <TEXTAREA>.	92
FIGURA 1.93 PERMISOS DE UN ARCHIVO.	93
FIGURA 1.94 EJECUCIÓN DE UN PROGRAMA CGI.	95
FIGURA 1.95 CGI DESARROLLADO EN BASH.	97
FIGURA 1.96 VISUALIZACIÓN DE CGI DESARROLLADO EN BASH.	98
FIGURA 1.97 FORMULARIO DESARROLLADO EN APPLESCRIPT.	99
FIGURA 1.98 VISUALIZACIÓN DE UN CGI DESARROLLADO EN APPLESCRIPT.	100
FIGURA 1.99 CGI DESARROLLADO EN C.	102
FIGURA 1.100 ALMACENAMIENTO DEL ARCHIVO CCGI.C.	102
FIGURA 1.101 COMPILACIÓN DEL ARCHIVO CCGI.C.	102
FIGURA 1.102 ALMACENAMIENTO DEL EJECUTABLE CCGI.CGI.	103
FIGURA 1.103 VISUALIZACIÓN DE CGI DESARROLLADO EN C.	103
FIGURA 1.104 PARTE DEL CÓDIGO EMPLEADO PARA DECLARAR LAS VARIABLES DE ENTORNO CGI.	105
FIGURA 1.105 CÓDIGO HTML PARA EMPLEAR EL CGI DESARROLLADO EN VISUAL.	106
FIGURA 1.106 CGI DESARROLLADO EN PERL.	107
FIGURA 1.107 VISUALIZACIÓN DE CGI DESARROLLADO EN PERL.	108
FIGURA 1.108 USO DE UN SERVIDOR CENTRALIZADO.	110
FIGURA 1.109 CLIENTE UNIVERSAL.	111
FIGURA 3.1 ESTRUCTURA DE ÁRBOL O JERÁRQUICA.	181
FIGURA 3.2 ESTRUCTURA LINEAL.	182
FIGURA 3.3 ESTRUCTURA EN RED.	182
FIGURA 3.4 ESTRUCTURA DE DISEÑO DEL SISTEMA.	184
FIGURA 3.5 INTERFAZ-PÁGINA DE INICIO DEL SISTEMA DAMA.	185
FIGURA 3.6 INTERFAZ - CONFIGURACIÓN BÁSICA.	186

FIGURA 3.7 INTERFAZ - CONFIGURACIÓN DINÁMICA.....	187
FIGURA 3.8 INTERFAZ - CONFIGURACIÓN ESTÁTICA.....	188
FIGURA 3.9 INTERFAZ - DIRECCIONAMIENTO IP.	188
FIGURA 3.10 INTERFAZ - OPCIONES GENERALES.	189
FIGURA 3.11 INTERFAZ - LISTA DE CONTROL DE ACCESO.	189
FIGURA 3.12 INTERFAZ - POLÍTICAS DE ACCESO.	190
FIGURA 3.13 INTERFAZ - MANEJO DE USUARIOS.	190
FIGURA 3.14 INTERFAZ - VISUALIZACIÓN DE INFORMACIÓN GENERAL.	191
FIGURA 3.15 INTERFAZ - MANEJO DE USUARIOS.	192
FIGURA 3.16 INTERFAZ - AGREGAR UN DIRECTORIO.	192
FIGURA 3.17 INTERFAZ - MODIFICAR LAS CARACTERÍSTICAS DE LOS DIRECTORIOS COMPARTIDOS.	192
FIGURA 3.18 INTERFAZ - MODIFICACIÓN DE OPCIONES GENERALES.....	193
FIGURA 3.19 INTERFAZ - FILTRADO POR MIME.	194
FIGURA 3.20 MÓDULO SERVICIO DE FILTRADO WEB – FILTRADO POR EXTENSIÓN.	194
FIGURA 3.21 INTERFAZ - FILTRADO POR SITE.	195
FIGURA 3.22 INTERFAZ - FILTRADO POR FRASE.....	195
FIGURA 3.23 INTERFAZ - GENERACIÓN DE ZONAS.	196
FIGURA 3.24 INTERFAZ - GENERACIÓN DE RELACIÓN ENTRE UNA ZONA Y UNA INTERFAZ DE RED.	197
FIGURA 3.25 INTERFAZ - GENERACIÓN DE POLÍTICAS DE ACCESO DE ALTO NIVEL.	197
FIGURA 3.26 INTERFAZ - GENERACIÓN DE REGLAS DE ACCESO ESPECÍFICAS.	197
FIGURA 3.27 INTERFAZ - GENERACIÓN DE ENMASCARAMIENTO.....	198
FIGURA 3.28 INTERFAZ - ESTADÍSTICAS DE USO DEL DISCO DURO.	199
FIGURA 3.29 INTERFAZ - ESTADÍSTICA GENERAL DE MEMORIA.....	199
FIGURA 3.30 INTERFAZ - USO DE CPU.	200
FIGURA 3.31 INTERFAZ - INFORMACIÓN DE RED.	201
FIGURA 3.32 INTERFAZ - INFORMACIÓN ADICIONAL.....	202
FIGURA 3.33 INTERFAZ - RENDIMIENTO DE RED.	203
FIGURA 3.34 INTERFAZ - USO DE MEMORIA.....	203
FIGURA 3.35 INTERFAZ - CARGA DE CPU.	204
FIGURA 3.36 DIAGRAMA DE FLUJO – CONFIGURACIÓN BÁSICA DHCP.....	206
FIGURA 3.37 DIAGRAMA DE FLUJO – CONFIGURACIÓN DINÁMICA DHCP.	207
FIGURA 3.38 DIAGRAMA DE FLUJO – CONFIGURACIÓN ESTÁTICA DHCP – PARTE A.	209
FIGURA 3.39 DIAGRAMA DE FLUJO – CONFIGURACIÓN ESTÁTICA DHCP – PARTE B.	209
FIGURA 3.40 DIAGRAMA DE FLUJO – PROXY - MODIFICACIÓN DE OPCIONES GENERALES.....	211
FIGURA 3.41 DIAGRAMA DE FLUJO – PROXY – GENERACIÓN DE LISTAS DE CONTROL DE ACCESO.....	213
FIGURA 3.42 DIAGRAMA DE FLUJO – PROXY – GENERACIÓN DE POLÍTICAS DE ACCESO.....	214
FIGURA 3.43 DIAGRAMA DE FLUJO – PROXY – MANEJO DE USUARIOS.....	216
FIGURA 3.44 DIAGRAMA DE FLUJO – DIRECTORIOS COMPARTIDOS – AÑADIR UN DIRECTORIO.....	218
FIGURA 3.45 DIAGRAMA DE FLUJO – DIRECTORIOS COMPARTIDOS – MODIFICAR LOS CAMPOS DE UN DIRECTORIO COMPARTIDO.....	220
FIGURA 3.46 DIAGRAMA DE FLUJO – FILTRADO WEB – MODIFICACIÓN DE OPCIONES GENERALES.....	221
FIGURA 3.47 DIAGRAMA DE FLUJO – FILTRADO WEB – FILTRADO POR MIME.	222
FIGURA 3.48 DIAGRAMA DE FLUJO – FIREWALL – GENERACIÓN DE ZONAS.	224
FIGURA 3.49 DIAGRAMA DE FLUJO – FIREWALL – GENERACIÓN DE RELACIÓN ENTRE UNA ZONA Y UNA INTERFAZ DE RED.	225
FIGURA 3.50 DIAGRAMA DE FLUJO – FIREWALL – GENERACIÓN DE POLÍTICAS DE ALTO NIVEL.	226
FIGURA 3.51 DIAGRAMA DE FLUJO – FIREWALL – GENERACIÓN DE REGLAS DE ACCESO ESPECÍFICAS.	227
FIGURA 3.52 DIAGRAMA DE FLUJO – FIREWALL – GENERACIÓN DE ENMASCARAMIENTO.	228
FIGURA 3.53 DIAGRAMA DE FLUJO – ESTADO DEL SISTEMA OPERATIVO – ESTADO DEL DISCO DURO... ..	229
FIGURA 3.54 OBTENCIÓN DE INFORMACIÓN PARA LA GRÁFICA DE CARGA DE CPU.....	231
FIGURA 3.55 OBTENCIÓN DE INFORMACIÓN PARA LA GRÁFICA DE USO DE MEMORIA.....	231
FIGURA 3.56 OBTENCIÓN DE INFORMACIÓN PARA LA GRÁFICA DE RENDIMIENTO DE RED.....	232
FIGURA 3.57 CREACIÓN DE LA BASE DE DATOS RRD.	232
FIGURA 3.58 ALMACENAMIENTO DE LA INFORMACIÓN EN LA BASE DE DATOS RRD.	232
FIGURA 3.59 CREACIÓN DE GRÁFICAS MEDIANTE EL USO DE LA FUNCIÓN RRDs :: GRAPH.	233
FIGURA 3.60 EJEMPLO DE MANIPULACIÓN DE ARCHIVOS EN PERL.	234
FIGURA 3.61 INFORMACIÓN ALMACENADA EN LOS ARCHIVOS DE TEXTO.	235
FIGURA 3.62 NUEVA INFORMACIÓN ALMACENADA EN LOS ARCHIVOS DE TEXTO.	236

FIGURA 4.1 ESQUEMA DE LA RED LOCAL A IMPLEMENTAR EN LAS PRUEBAS DEL SISTEMA DE ADMINISTRACIÓN WEB DAMA.	237
FIGURA 4.2 PRUEBA DE ACEPTACIÓN - DHCP - DIRECCIONAMIENTO IP.	245
FIGURA 4.3 PRUEBA DE ACEPTACIÓN – PROXY-CACHÉ.	250
FIGURA 4.4 PROXY-CACHÉ – RESULTADO ESPERADO.	250
FIGURA 4.5 PRUEBA DE ACEPTACIÓN - FILTRADO WEB - INFORMACIÓN GENERAL.	251
FIGURA 4.6 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR MIME.	257
FIGURA 4.7 FILTROS POR MIME – RESULTADO ESPERADO.	257
FIGURA 4.8 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR EXTENSIÓN.	259
FIGURA 4.9 FILTROS POR EXTENSIÓN – RESULTADO ESPERADO.	259
FIGURA 4.10 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR SITE.	261
FIGURA 4.11 FILTROS POR SITE – RESULTADO ESPERADO.	261
FIGURA 4.12 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR FRASE.	263
FIGURA 4.13 FILTROS POR FRASE – RESULTADO ESPERADO.	263
FIGURA 4.14 PRUEBA DE ACEPTACIÓN – ESTADO DEL SISTEMA OPERATIVO – DISCO DURO.	270
FIGURA 4.15 PRUEBA DE ACEPTACIÓN – ESTADO DEL SISTEMA OPERATIVO – RAM Y CPU.	271
FIGURA 4.16 PRUEBA DE ACEPTACIÓN – ESTADO DEL SISTEMA OPERATIVO – INFORMACIÓN DE RED. ...	272
FIGURA 4.17 PRUEBA DE ACEPTACIÓN – ESTADO DEL SISTEMA OPERATIVO – INFORMACIÓN ADICIONAL.	272
FIGURA 4.18 PRUEBA DE ACEPTACIÓN – MÓDULO DEL SERVICIO DE GRÁFICAS.	273
FIGURA 4.19 PRUEBA DE ACEPTACIÓN - PÁGINA INICIAL – AUTENTICACIÓN DEL ADMINISTRADOR DE RED.	273
FIGURA B.1 INTERFAZ GRÁFICA PARA SERVICIOS EN MANDRIVA LINUX.	301
FIGURA B.2 INICIO DEL SERVIDOR WEB HTTPD (APACHE).	301
FIGURA B.3 SERVIDOR HTTPD (APACHE) DETENIDO.	301
FIGURA B.4 REINICIO DEL SERVIDOR WEB HTTPD (APACHE).	302
FIGURA B.5 ESTADO DEL SERVIDOR WEB HTTPD (APACHE).	302
FIGURA B.6 GESTOR DE SOFTWARE PARA MANDRIVA LINUX.	303
FIGURA B.7 BÚSQUEDA E INSTALACIÓN DEL SERVICIO DHCP.	304
FIGURA B.8 DEPENDENCIAS PARA DHCP.	304
FIGURA B.9 CONFIGURACIÓN DEL SERVICIO DHCP DINÁMICO.	308
FIGURA B.10 DETERMINACIÓN DE LA DIRECCIÓN FÍSICA EN WINDOWS.	309
FIGURA B.11 DETERMINACIÓN DE LA DIRECCIÓN FÍSICA EN LINUX.	309
FIGURA B.12 CONFIGURACIÓN DE UNA DIRECCIÓN IP ESTÁTICA.	310
FIGURA B.13 ARRENDAMIENTOS DE DIRECCIONES IP.	310
FIGURA B.14 SQUID EN EL GESTOR DE SOFTWARE MANDRIVA.	312
FIGURA B.15 CREACIÓN DE ACL'S.	314
FIGURA B.16 REGLAS DE ACCESO.	314
FIGURA B.17 ASIGNACIÓN DEL TAMAÑO PARA CACHÉ EN SQUID.	316
FIGURA B.18 TAMAÑOS MÁXIMO Y MÍNIMO DE DISCO.	317
FIGURA B.19 ARCHIVO ZONES.	319
FIGURA B.20 ARCHIVO INTERFACES.	321
FIGURA B.21 ARCHIVO POLICY.	322
FIGURA B.22 ARCHIVOS RULES Y RULES.DRAKX.	324
FIGURA B.23 ARCHIVO MASQ.	326
FIGURA B.24 INSTALACIÓN DEL SERVIDOR WEB APACHE EN MANDRIVA LINUX.	329
FIGURA B.25 VERIFICACIÓN DEL FUNCIONAMIENTO DEL SERVIDOR WEB.	330
FIGURA B.26 MÓDULO MPM PREFORK.	333
FIGURA B.27 MANEJO DE DIRECTIVAS PARA EL DIRECTORIO CGI-BIN.	335
FIGURA B.28 SERVIDOR SAMBA EN EL GESTOR DE SOFTWARE DE MANDRIVA.	340
FIGURA B.29 CONFIGURACIÓN Y VISUALIZACIÓN DE UN RECURSO COMPARTIDO.	344
FIGURA B.30 FLUJO DE INFORMACIÓN A TRAVÉS DE SQUID Y DANSGUARDIAN.	345
FIGURA B.31 INSTALACIÓN DE DANSGUARDIAN EN MANDRIVA.	346
FIGURA B.32 DANSGUARDIAN FILTRANDO CONTENIDO.	348
FIGURA B.33 NEGACIÓN DEL ACCESO AL DOMINIO YOUTUBE.COM.	351
FIGURA B.34 COMANDO FREE.	353
FIGURA B.35 COMANDO DF.	354
FIGURA B.36 INFORMACIÓN DHCP AL OBTENER UNA CONCESIÓN.	355
FIGURA B.37 IFCFG-LO, INFORMACIÓN DE LA INTERFAZ VIRTUAL LOOPBACK.	355

FIGURA B.38 COMANDO IFCONFIG.....	356
FIGURA B.39 COMANDO ROUTE.....	357
FIGURA B.40 ARCHIVO CPUINFO.....	358
FIGURA B.41 PARTE DE LA INFORMACIÓN DISPONIBLE EN /PROC/STAT.....	359
FIGURA B.42 COMANDO UNAME -A.....	361
FIGURA B.43 COMANDO UPTIME.....	362
FIGURA B.44 COMANDO W.....	362
FIGURA B.45 COMANDO WHO -B.....	363

ÍNDICE DE TABLAS

TABLA 1.1 ESTRUCTURA DE UNA PETICIÓN O RESPUESTA DE HTTP.....	7
TABLA 1.2 ENTIDADES PARA CARACTERES RESERVADOS EN HTML.....	40
TABLA 1.3 TIPOS DE DATOS QUE GENERA UN PROGRAMA CGI.....	73
TABLA 1.4 TIPOS DE ARCHIVO.....	93
TABLA 1.5 TIPOS DE PERMISOS.....	94
TABLA 1.6 IDENTIDADES.....	94
TABLA 1.7 ACCIONES SOBRE LOS ARCHIVOS.....	94
TABLA 1.8 DIFERENCIA ENTRE METODOLOGÍAS TRADICIONALES Y ÁGILES.....	115
TABLA 1.9 COMPARACIÓN DE METODOLOGÍAS ÁGILES.....	127
TABLA 2.1 SOFTWARE REQUERIDO.....	135
TABLA 2.2 REQUERIMIENTOS HARDWARE MÍNIMOS PARA DISTINTOS SERVICIOS.....	137
TABLA 2.3 ASPECTOS PARA EL CÁLCULO DE LA CAPACIDAD EN DISCO PARA SAMBA.....	142
TABLA 2.4 CAPACIDADES TOTALES PARA EL AMBIENTE DE DESARROLLO.....	144
TABLA 2.5 FORMATO DE HISTORIA DE USUARIO.....	149
TABLA 2.6 HISTORIA DE USUARIO-CONFIGURACIÓN BÁSICA DEL SERVIDOR DHCP.....	150
TABLA 2.7 HISTORIA DE USUARIO-CONFIGURACIÓN DINÁMICA DEL SERVICIO DHCP.....	151
TABLA 2.8 HISTORIA DE USUARIO-CONFIGURACIÓN ESTÁTICA DEL SERVICIO DHCP.....	151
TABLA 2.9 HISTORIA DE USUARIO-DIRECCIONAMIENTO IP.....	152
TABLA 2.10 HISTORIA DE USUARIO-MODIFICACIÓN DE OPCIONES GENERALES.....	152
TABLA 2.11 HISTORIA DE USUARIO-GENERACIÓN DE LISTAS DE CONTROL DE ACCESO.....	153
TABLA 2.12 HISTORIA DE USUARIO-GENERACIÓN DE POLÍTICAS DE ACCESO.....	153
TABLA 2.13 HISTORIA DE USUARIO-MANEJO DE USUARIOS PARA EL SERVICIO.....	154
TABLA 2.14 HISTORIA DE USUARIO-VISUALIZACIÓN DE INFORMACIÓN GENERAL.....	154
TABLA 2.15 HISTORIA DE USUARIO-MANEJO DE USUARIOS PARA EL SERVICIO.....	155
TABLA 2.16 HISTORIA DE USUARIO-ADICIÓN DE USUARIOS AL SERVICIO.....	155
TABLA 2.17 HISTORIA DE USUARIO-MODIFICACIÓN DE LAS CARACTERÍSTICAS DE LOS DIRECTORIOS COMPARTIDOS.....	156
TABLA 2.18 HISTORIA DE USUARIO-MODIFICACIÓN DE OPCIONES GENERALES.....	156
TABLA 2.19 HISTORIA DE USUARIO-FILTRADO POR MIME.....	157
TABLA 2.20 HISTORIA DE USUARIO-FILTRADO POR EXTENSIÓN.....	157
TABLA 2.21 HISTORIA DE USUARIO-FILTRADO POR SITE.....	158
TABLA 2.22 HISTORIA DE USUARIO-FILTRADO POR FRASE.....	158
TABLA 2.23 HISTORIA DE USUARIO-GENERACIÓN DE ZONAS.....	159
TABLA 2.24 HISTORIA DE USUARIO-GENERACIÓN DE RELACIÓN ENTRE UNA ZONA Y UNA INTERFAZ DE RED.	160
TABLA 2.25 HISTORIA DE USUARIO-GENERACIÓN DE POLÍTICAS DE ACCESO DE ALTO NIVEL.....	160
TABLA 2.26 HISTORIA DE USUARIO-GENERACIÓN DE REGLAS DE ACCESO ESPECÍFICAS.....	161
TABLA 2.27 HISTORIA DE USUARIO-GENERACIÓN DE ENMASCARAMIENTO.....	161
TABLA 2.28 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICAS DE USO DEL DISCO DURO.....	162
TABLA 2.29 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICAS DE DISCOS DUROS ADICIONALES Y PARTICIONES.....	162
TABLA 2.30 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICA GENERAL DE MEMORIA.....	163
TABLA 2.31 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICA DEL USO DE CPU.....	163
TABLA 2.32 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICA DE INTERFAZ CON SALIDA A INTERNET.	164
TABLA 2.33 HISTORIA DE USUARIO-GENERACIÓN DE TABLA DE ENRUTAMIENTO.....	164
TABLA 2.34 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICAS DE LA INTERFAZ DE RED LOCAL.....	165
TABLA 2.35 HISTORIA DE USUARIO-GENERACIÓN DE ESTADÍSTICAS DE USUARIOS Y TIEMPO ACTIVO DEL SISTEMA.....	165
TABLA 2.36 HISTORIA DE USUARIO-GENERACIÓN DE INFORMACIÓN DEL KERNEL.....	166
TABLA 2.37 HISTORIA DE USUARIO-GRÁFICAS DEL RENDIMIENTO DEL TRÁFICO DE RED.....	166
TABLA 2.38 HISTORIA DE USUARIO-GRÁFICAS DEL USO DE MEMORIA.....	167
TABLA 2.39 HISTORIA DE USUARIO-GRÁFICAS DE LA CARGA DEL CPU.....	167
TABLA 2.40 HISTORIA DE USUARIO-PÁGINA INICIAL.....	168
TABLA 3.1 FORMATO DE LA TABLA PARA LA PLANIFICACIÓN DE ENTREGABLES.....	171
TABLA 3.2 ESTIMACIÓN DE ENTREGABLES.....	173

TABLA 3.3 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN CONFIGURACIÓN BÁSICA DHCP.....	174
TABLA 3.4 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN CONFIGURACIÓN DINÁMICA DHCP.	175
TABLA 3.5 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN MANEJO DE USUARIOS PARA EL SERVICIO.	176
TABLA 3.6 ITERACIÓN-HISTORIA DE USUARIO-CAMBIO EN MODIFICACIÓN DE OPCIONES GENERALES. ...	177
TABLA 3.7 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN MÓDULO DEL SERVICIO FIREWALL.	178
TABLA 3.8 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN MÓDULO DE DISCO DURO.	179
TABLA 3.9 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN MÓDULO DE CPU Y MEMORIA.	179
TABLA 3.10 ITERACIÓN-HISTORIA DE USUARIO-MODIFICACIÓN EN MÓDULO DE GRÁFICAS.	180
TABLA 4.1 FORMATO DE CASO DE PRUEBA DE ACEPTACIÓN.	239
TABLA 4.2 PRUEBA DE ACEPTACIÓN – INGRESO CONFIGURACIÓN BÁSICA DHCP.	241
TABLA 4.3 PRUEBA DE ACEPTACIÓN – INGRESO CONFIGURACIÓN DINÁMICA DHCP.	242
TABLA 4.4 PRUEBA DE ACEPTACIÓN – INGRESO CONFIGURACIÓN ESTÁTICA DHCP (A).....	243
TABLA 4.5 PRUEBA DE ACEPTACIÓN – INGRESO CONFIGURACIÓN ESTÁTICA DHCP (B).....	244
TABLA 4.6 PRUEBA DE ACEPTACIÓN – INGRESO DE OPCIONES GENERALES.	246
TABLA 4.7 PRUEBA DE ACEPTACIÓN – INGRESO DE LISTAS DE CONTROL DE ACCESO.....	247
TABLA 4.8 PRUEBA DE ACEPTACIÓN – INGRESO DE POLÍTICAS DE ACCESO.	248
TABLA 4.9 PRUEBA DE ACEPTACIÓN – INGRESAR, MODIFICAR Y ELIMINAR USUARIOS DEL SERVICIO PROXY- CACHÉ.....	249
TABLA 4.10 PRUEBA DE ACEPTACIÓN – INGRESAR, MODIFICAR Y ELIMINAR USUARIOS DEL SERVICIO DE DIRECTORIOS COMPARTIDOS.	252
TABLA 4.11 PRUEBA DE ACEPTACIÓN – AÑADIR UN DIRECTORIO AL SERVICIO.	253
TABLA 4.12 PRUEBA DE ACEPTACIÓN – MODIFICAR LAS CARACTERÍSTICAS DE UN DIRECTORIO COMPARTIDO.....	254
TABLA 4.13 PRUEBA DE ACEPTACIÓN – MODIFICAR LAS OPCIONES GENERALES DEL SERVICIO DE FILTRADO WEB.	255
TABLA 4.14 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR MIME.	256
TABLA 4.15 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR EXTENSIÓN.	258
TABLA 4.16 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR SITE.....	260
TABLA 4.17 PRUEBA DE ACEPTACIÓN – INGRESAR FILTROS POR FRASE.....	262
TABLA 4.18 PRUEBA DE ACEPTACIÓN – CREACIÓN Y MODIFICACIÓN DE ZONAS.....	264
TABLA 4.19 PRUEBA DE ACEPTACIÓN – CREACIÓN DE RELACIONES ENTRE INTERFACES Y ZONAS.	265
TABLA 4.20 PRUEBA DE ACEPTACIÓN – CREACIÓN DE POLÍTICAS DE ALTO NIVEL.	266
TABLA 4.21 PRUEBA DE ACEPTACIÓN – CREACIÓN DE REGLAS DE ACCESO ESPECÍFICAS.....	267
TABLA 4.22 PRUEBA DE ACEPTACIÓN – CREACIÓN DE ENMASCARAMIENTO.....	268
TABLA 4.23 PRUEBA DE ACEPTACIÓN – VISUALIZAR ESTADÍSTICAS DE USO DEL DISCO DURO Y PARTICIONES.	269
TABLA 4.24 PRUEBA DE ACEPTACIÓN – VISUALIZAR ESTADÍSTICAS DE MEMORIA Y DE USO DE CPU.....	270
TABLA 4.25 PRUEBA DE ACEPTACIÓN – VISUALIZAR INFORMACIÓN DE RED.	271
TABLA A.1 MODOS DE APERTURA DE ARCHIVO.	288
TABLA A.2 OPERACIONES ARITMÉTICAS.	292
TABLA A.3 OPERACIONES LÓGICAS.	293
TABLA A.4 OPERACIONES PARA CARACTERES Y EXPRESIONES REGULARES.	294
TABLA A.5 CARACTERES ESPECIALES.....	297
TABLA A.6 OPERADORES PARA PATRONES EN UNA EXPRESIÓN REGULAR.	298
TABLA A.7 INSTRUCCIONES MÁS EMPLEADAS EN PERL.....	299
TABLA B.1 DECLARACIONES, PARÁMETROS Y OPCIONES EN DHCP.....	307
TABLA B.2 PROTOCOLOS SOPORTADOS POR SQUID.	312
TABLA B.3 OPCIONES DISPONIBLES EN UNA ACL.	316
TABLA B.4 ALGUNAS POLÍTICAS DENTRO DEL ARCHIVO <i>POLICY</i>	323
TABLA B.5 PRINCIPALES COLUMNAS DENTRO DEL ARCHIVO <i>RULES</i>	325
TABLA B.6 DIRECTORIOS Y ARCHIVOS PARA CONFIGURACIÓN DEL SERVICIO WEB APACHE.....	330
TABLA B.7 DIRECTIVAS PARA MANEJO DE TIEMPOS.	334
TABLA B.8 DIRECTIVAS PARA MANEJO DE TIEMPOS.....	343
TABLA B.9 ENTRADAS PARA CONFIGURAR DIRECCIONES IP Y PUERTOS.	347
TABLA B.10 LISTAS EMPLEADAS PARA PROCESOS DE FILTRADO.	351
TABLA B.11 ALGUNOS CAMPOS EN EL ARCHIVO CPUINFO.	359

RESUMEN

El presente Proyecto de Titulación describe el desarrollo de un prototipo de solución integral WEB para la administración de una red tipo SOHO (Small Office Home Office) bajo la plataforma GNU/LINUX, al cual se lo ha dividido en cinco capítulos, que detallan tanto la información necesaria, así como el proceso empleado para el desarrollo e implementación del prototipo.

En el primer capítulo se describen los conceptos fundamentales para conocer qué elementos deben intervenir en el desarrollo del proyecto, es así que se describe al lenguaje HTML como una parte fundamental del desarrollo, de la misma manera se describen las distintas tecnologías del lado del servidor para el desarrollo de páginas web dinámicas, para de ésta manera seleccionar la tecnología que cumpla con las características del proyecto. Además, se define qué es una red tipo SOHO (Small Office Home Office), así como sus características; como parte final del capítulo se describen las distintas metodologías de desarrollo para seleccionar aquella que se ajuste al ambiente de desarrollo.

En el segundo capítulo se emplea el estándar IEEE 830-1998, como herramienta para obtener los requerimientos tanto hardware y software necesarios para el desarrollo del proyecto, tomando en cuenta las características requeridas por el tipo de red en donde se implementa el prototipo.

En el tercer capítulo se emplea la metodología de desarrollo XP (Programación eXtrema) seleccionada en el primer capítulo, para el desarrollo e implementación de cada una de las funciones del proyecto: página de inicio, configuración de servicio DHCP, configuración del servicio Proxy-Caché, configuración del servicio de Directorios Compartidos, configuración del servicio de Filtrado Web, configuración del servicio de Firewall, configuración del servicio de Estado del Sistema Operativo y configuración del servicio de Gráficas, descritas en el segundo capítulo.

En el cuarto capítulo se describen las pruebas realizadas en una red tipo SOHO, empleando la sección correspondiente a pruebas de la metodología de desarrollo XP.

En el quinto capítulo se establecen las conclusiones y recomendaciones obtenidas en la elaboración del proyecto.

PRESENTACIÓN

En la actualidad existen sistemas de administración de servicios vía WEB integrados en un servidor propietario, gran parte de éstos están orientados a proveer servicios a grandes empresas, en una pequeña empresa u organización en la cual se cuenta con recursos limitados, es necesario buscar soluciones que impliquen bajos costos y se adapten a los requerimientos reales de la empresa.

La solución dada en el presente proyecto permite que la empresa pueda emplear recursos disponibles, como por ejemplo la utilización de equipos existentes para la implementación de un servidor, ya que en base a las características de una pequeña empresa el servidor demanda un mínimo de requerimientos hardware, añadiendo a esto el empleo de un sistema operativo como Linux, nos ofrece la posibilidad de poder utilizar herramientas para proveer servicios en una pequeña empresa, adaptarlas a sus necesidades y de ésta manera reducir costos.

Dado que este proyecto se encuentra enfocado a la administración de servicios en una red con un número limitado de usuarios, es ideal para ser implementado en laboratorios de Instituciones Educativas que requieran ser administrados.

La finalidad del desarrollo de una interfaz web que permita la administración de servicios, es facilitar el trabajo del webmaster o administrador de la red, el cual podrá configurar cada uno de los servicios disponibles en el servidor remotamente desde cualquier punto de acceso a la red, utilizando como única herramienta un navegador web, el cual permitirá invocar al sistema de administración.

En base a la información proporcionada en los párrafos anteriores se procede a desarrollar un prototipo de solución integral web que permita la administración de una red para una pequeña empresa tipo SOHO (Small Office Home Office), utilizando para ello la plataforma GNU/Linux, obteniendo como resultado una interfaz web funcional y amigable para el usuario.

Los Autores.

CAPÍTULO 1

1 FUNDAMENTOS TEÓRICOS

1.1 INTRODUCCIÓN A LA ADMINISTRACIÓN DE SERVICIOS VÍA WEB

La evolución en el desarrollo de páginas web ha permitido que estas pasen de ser una simple fuente de información (texto, imágenes, video) a páginas web funcionales que permiten la interacción con usuarios con el objetivo de cumplir distintas tareas, esta interacción se logra mediante la unión del lenguaje HTML y un lenguaje de programación que permita generar scripts¹ para manipular la información obtenida por medio de formularios dentro de una página web.

La administración de servicios a través de la web se enfoca a la utilización de una interfaz web funcional para la manipulación de contenidos dentro de un servidor, es decir a la modificación de archivos para que cumplan tareas específicas, por ejemplo, mediante la interfaz web se puede modificar aquellos archivos que permiten la ejecución de un servicio DHCP², para variar el rango de direccionamiento IP o definir direcciones IP estáticas. Cada servicio a ser utilizado dentro del servidor cuenta con un conjunto de archivos de configuración, la tarea de la interfaz web es permitir que el administrador de la red pueda modificar los distintos servicios sin necesidad de estar junto al servidor.

El administrador puede acceder al servidor desde cualquier estación de trabajo dentro de su intranet utilizando la dirección IP del servidor o su nombre de dominio, empleando un navegador web.

¹ **Scripts.**- Es un conjunto de instrucciones, generalmente almacenados en archivos de texto y son interpretados línea a línea en tiempo real para su ejecución.

² **Dynamic Host Configuration Protocol (DHCP).**- Protocolo de Configuración Dinámica de Host, es un protocolo que permite obtener la información de red necesaria para que un usuario pueda formar parte de una red.

1.1.1 LA WEB

La palabra web comúnmente es utilizada para referirse a internet pero en realidad se trata de la organización de información mediante una colección de documentos enlazados llamados hipertexto o hipermedia.

En el caso de una intranet esta colección de documentos se encontrará alojada en el servidor web local mientras que si se trata de internet esta colección de documentos puede estar alojada ya sea en un solo servidor o distintos servidores alrededor del planeta.

1.1.1.1 Hipertexto e Hipermedia

El texto tradicional aquel que encontramos comúnmente en cualquier libro o enciclopedia tiene una secuencia lineal, la página uno sigue a la dos y así sucesivamente el resto de páginas, este tipo de ordenamiento lineal permite obtener información de manera rápida para el caso de un documento pequeño, pero si contamos con una enciclopedia con cientos de páginas la búsqueda se vuelve un poco tediosa ya que se tendría que leer todo el documento para obtener la información necesaria, para este caso surgen los denominados índices y referencia que permiten acceder a la información de manera no lineal mejorando la exploración a través de un documento con cientos de páginas, de aquí viene la idea del concepto hipertexto e hipermedia.

- **Hipertexto**

Un documento de hipertexto es aquel documento electrónico que cuenta con varios enlaces que permiten acceder a fragmentos de información o documentos relacionados al tema que se trata, se caracteriza por contar únicamente con información textual. Por ejemplo, la figura 1.1 muestra una página web de wikipedia que proporciona información acerca de internet.



Figura 1.1 Hipertexto.

En la figura 1.1 se aprecia claramente el hipertexto, éste se muestra como texto de color azul y permite navegar hacia el resto de información. Por ejemplo, al momento que realizamos un clic sobre la palabra [Historia](#) la página web nos dirige al lugar donde se encuentra la información referida a historia. El flujo a través de la página web se muestra en la figura 1.2.



Figura 1.2 Flujo de hipertexto.

- **Hipermedia**

Hipermedia añade un nuevo concepto al hipertexto, permite agregar contenido multimedia, añadir gráficos o videos al contenido textual, la información toma un nuevo sentido para hipermedia ya que por medio de un video o imagen se proporciona información referente al tema que se trate en el documento web. Por ejemplo, la figura 1.3 muestra la página web de wikipedia acerca de internet con un enlace a información visual.

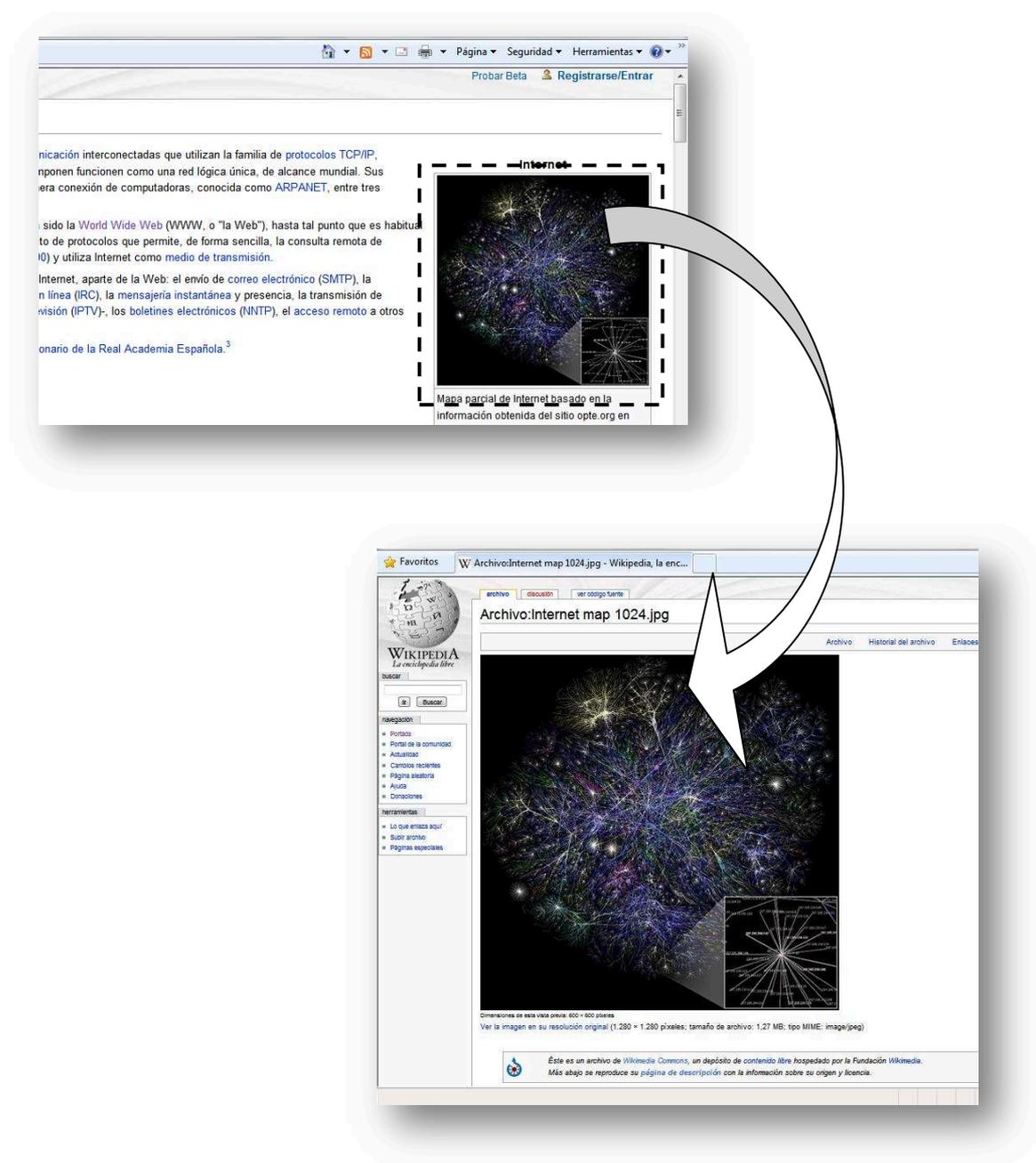


Figura 1.3 Hipermedia.

1.1.1.2 Visión general del uso de la Web

La web basa su funcionamiento en el modelo cliente/servidor, el cliente constituye aquel elemento del modelo que se encarga de generar solicitudes para obtener algún tipo de respuesta, mientras que el servidor es el elemento encargado de generar la respuesta a dicha solicitud.

En el ámbito de la web el servidor lo constituye un servidor Web, éste se encarga de generar la información solicitada por un navegador web (cliente). El flujo de información entre los elementos se realiza a través de una red, para el caso de una intranet el servidor web proporcionará información de manera local, mientras que si se trata de internet, el servidor web proporcionará información a todos los clientes que forman parte de él.

La interacción entre el explorador web y el servidor lo maneja el protocolo HTTP, mientras que la transmisión de información es manejada por TCP/IP. El explorador web, más conocido como browser, debe emplear algún tipo de esquema que permita identificar el lugar hacia donde se quiere comunicar para obtener información, para ello utiliza el URI (Uniform Resource Identifier o Identificador Uniforme de Recursos), el cual emplea una dirección numérica conocida como IP o su correspondiente alfanumérica conocida como nombre de dominio. Por ejemplo, <http://www.google.com/> o <http://74.125.65.106/>, gracias a este identificador el browser puede saber hacia dónde se tiene que dirigir para obtener la información necesaria para generar su correspondiente página web, esto de manera general ya que el flujo de información dentro de una red involucra varios elementos adicionales como routers, servidores de nombres de dominio entre otros.

La solicitud que envía el browser al servidor debe ser validada con la finalidad de determinar por ejemplo si se puede o no tener acceso al contenido solicitado, ya que éste puede estar implementado con algunas restricciones. Luego de este paso el servidor busca el archivo solicitado y añade información adicional para que sea interpretada por el browser, esta información le permite identificar los tipos de datos que va a recibir.

A esta información se la conoce como MIME³, para el caso de un documento con contenido HTML el tipo de MIME que se envía es text/html.

La figura 1.4 muestra como se produce el flujo de información entre el navegador web y el servidor.

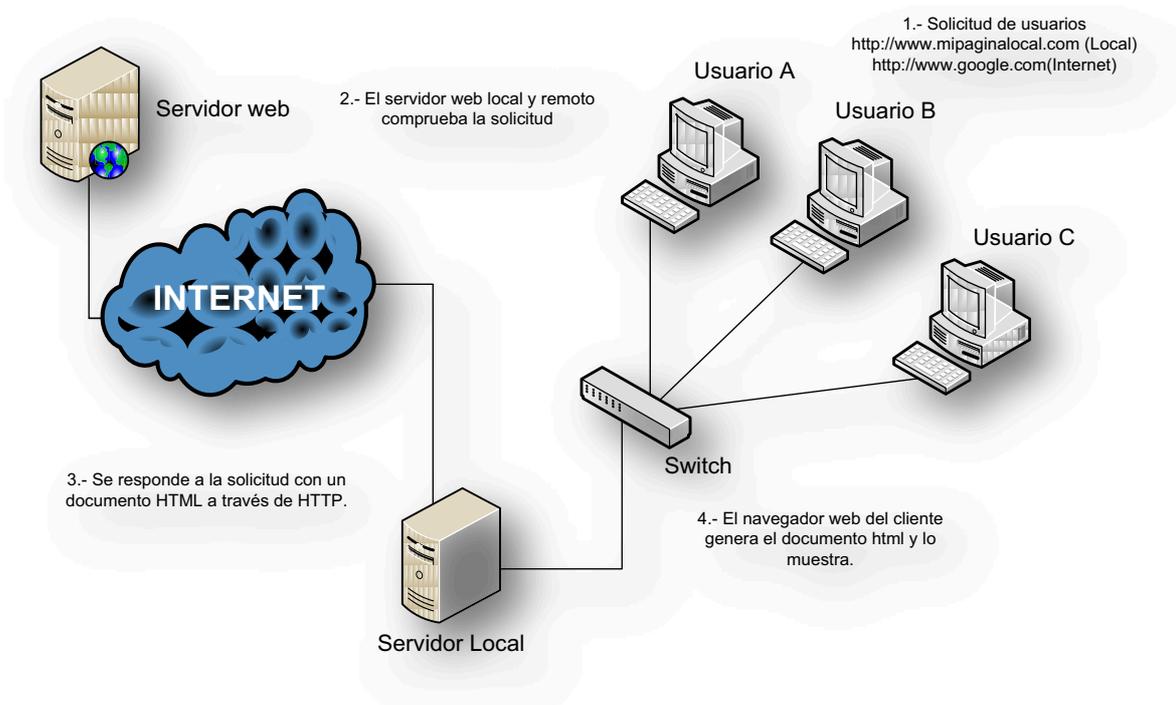


Figura 1.4 Modelo cliente/servidor Web.

- **Protocolo HTTP**

El Protocolo de Transferencia de Hipertexto más conocido como HTTP (HyperText Transfer Protocol) constituye un conjunto de reglas empleadas para la transferencia de datos en una comunicación Web (cliente/servidor), su finalidad es la transferencia de hipertexto. Añadiendo en la actualidad la transferencia de contenido multimedia (hipermedia) y la transferencia de archivos.

La comunicación en HTTP se realiza a través de TCP/IP en base a peticiones/respuestas, utiliza el puerto 80 (por defecto). La petición la realiza el

³ **Multipurpose Internet Mail Extensions (MIME).**- Extensiones Multipropósito de Correo de Internet, es un estándar de codificación para permitir enviar todo tipo de archivos a través de Internet.

navegador web y es recibida por el servidor web quien se encarga de aceptar o cerrar la conexión.

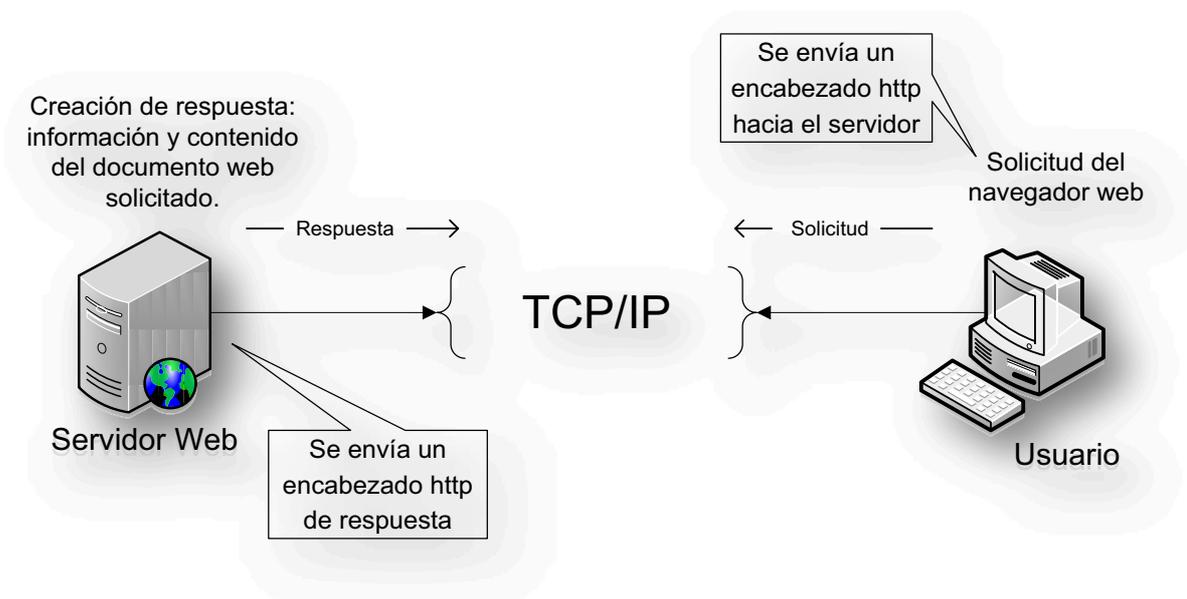


Figura 1.5 HTTP.

El formato de la petición y la respuesta HTTP consta de la siguiente estructura:

Línea de Petición o Respuesta
Encabezado-1: Valor-1
...
Encabezado-N: Valor-n
CUERPO

Tabla 1.1 Estructura de una petición o respuesta de HTTP.

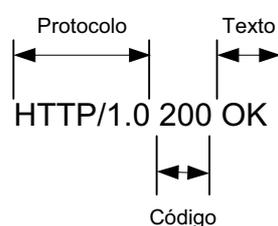
- **Petición**

Constituye la información que el navegador web envía al servidor, está formada por tres partes:

- **Respuesta**

Constituye la información que el servidor envía al navegador web, está formado por tres partes:

- **Línea de respuesta:** en esta línea se especifica la información correspondiente al protocolo empleado por el servidor, el estado de la solicitud mediante un código y un texto con información acerca del código. Cada uno separado por un espacio. Algunos mensajes se describen en el subcapítulo referente al servidor Web.



- **Encabezado:** estas líneas opcionales proporcionan información adicional acerca de la respuesta o el servidor. Está formado por: el tipo de encabezado seguido de dos puntos y su valor correspondiente.

Date: Sat, 15 Jan 2000 14:37:12

- **Cuerpo:** constituye el conjunto de líneas que forman el documento HTML solicitado.

<html> <body>.....</body> </html>

A continuación se presenta un ejemplo completo de respuesta HTTP:

Línea de Respuesta	➡	HTTP/1.1 200 OK
Encabezado	➡	{ Date: Sat, 15 Jan 2011 14:57:48 GMT Content-Type: text/html
Cuerpo	➡	<html><body>Contenido</body></html>

- **Protocolo HTTPS**

Hypertext Transfer Protocol Secure (ó HTTPS) constituye la combinación de HTTP y códigos de cifrado para realizar conexiones seguras entre el navegador web y el servidor, para ello emplea encriptación SSL⁵ o TLS⁶ la cual trabaja en una capa inferior (capa transporte) a http (capa aplicación) dentro de TCP/IP. HTTPS es utilizado en páginas cuyo manejo de información es muy sensible por ejemplo en bancos, transacciones de pago, en general en páginas donde se maneje información importante.

Para diferenciar una página web común de una segura hay que tomar en cuenta que todas las páginas seguras comienzan con HTTPS en su URI, mientras que dentro de la comunicación se emplea el puerto 443 por defecto a diferencia de HTTP que emplea el 80 por defecto.

- Negociación entre el servidor y el navegador web:
 - Para hacer esta negociación, el navegador web, envía al servidor las opciones de compresión, cifrado y versión SSL/TLS junto con algunos bytes aleatorios llamados Challenge de Cliente.
 - El servidor, escoge las opciones de compresión, cifrado y versión de SSL/TLS entre las que ha ofertado el cliente y le envía su decisión y su certificado.
 - Ambos negocian la clave secreta llamada master secret y usando esta clave, la Challenge de Cliente y las opciones negociadas se envía la información encriptada de tal manera que de ser interceptada no se puede descifrar.

⁵ **Secure Sockets Layer (SSL).**- Protocolo de Capa de Conexión Segura, es un protocolo desarrollado por Netscape para permitir la transferencia segura de datos mediante encriptación en Internet.

⁶ **Transport Layer Security (TLS).**- Seguridad de la Capa de Transporte, este protocolo es una versión estándar de la IETF del protocolo SSL, que abarca la capa transporte del modelo OSI. Proporciona comunicaciones seguras extremo-extremo en Internet.

Los certificados que utiliza un servidor pueden ser creados con herramientas como OpenSSL⁷ y firmados por una autoridad certificadora que garantiza que la comunicación es segura, o bien se puede autofirmar el certificado permitiendo el cifrado del canal de comunicación pero el cliente no puede garantizar que la comunicación se esté realizando con el servidor correcto.

1.1.2 TIPOS BÁSICOS DE DOCUMENTOS WEB

En sus inicios un documento web fue creado para mostrar información de una manera simple, donde se visualizaba el contenido como una hoja común llena de texto. La evolución ha permitido que se desarrollen nuevas formas de visualización con la posibilidad de interactuar con el usuario. Tomando en cuenta la interacción con el usuario y la forma de mostrar la información por parte de un servidor web los documentos se pueden agrupar en dos categorías.

- **Documento web estático**

Es aquel documento que se caracteriza por mostrar información que no cambia en el tiempo, la información puede ser texto, imágenes o un contenido multimedia que no interactúa con el usuario, su función es proporcionar información, la persona encargada para desarrollar este tipo de documento no requiere tener un conocimiento profundo sobre programación ya que puede utilizar herramientas disponibles que generen todo el contenido del documento web para luego ser almacenado dentro del servidor, la solicitud a este tipo de documento genera siempre el mismo resultado.

- **Documento web dinámico**

Un documento web dinámico es aquel cuyo contenido se genera a través de algún tipo de aplicación cuando el cliente realiza una solicitud al servidor. Se añade a la generación del contenido web la realización de procesos de cálculo y generación de respuestas, para esto requiere obtener información por parte del cliente por lo que el documento web empleará formularios para ingresar

⁷ **OpenSSL:** Es una implementación de código abierto del protocolo SSL (y del protocolo TLS) que consta de un paquete de herramientas de administración y librerías relacionadas con la criptografía. Proporciona comunicaciones seguras en Internet. Además permite crear certificados digitales que pueden aplicarse a un servidor.

información para ser enviada hacia el servidor. Por lo tanto, el contenido que muestra este documento cambia de manera frecuente ya que el usuario interactúa constantemente con el documento web. Para el desarrollo de este tipo de documento web se requiere tener un conocimiento amplio de programación y métodos de seguridad ya que se puede manipular información sensible entre el cliente y el servidor.

1.2 EL WEBMASTER

El webmaster o administrador web es aquella persona encargada de mantener el correcto funcionamiento de los servicios suministrados en la intranet. Dependiendo del tamaño de la empresa, el webmaster puede desempeñar diversas funciones, las cuales se detallarán posteriormente. Las principales responsabilidades del webmaster son la creación y mantenimiento de los servidores y servicios de la intranet. Es decir, es la persona responsable de todo el sitio web, ya que es quien cuenta con todos los permisos para realizar cambios en la estructura y el contenido del sitio. El webmaster trabaja conjuntamente con cada uno de los departamentos de la empresa u organización, para la creación y mantenimiento de contenidos y servicios dentro de la Intranet.

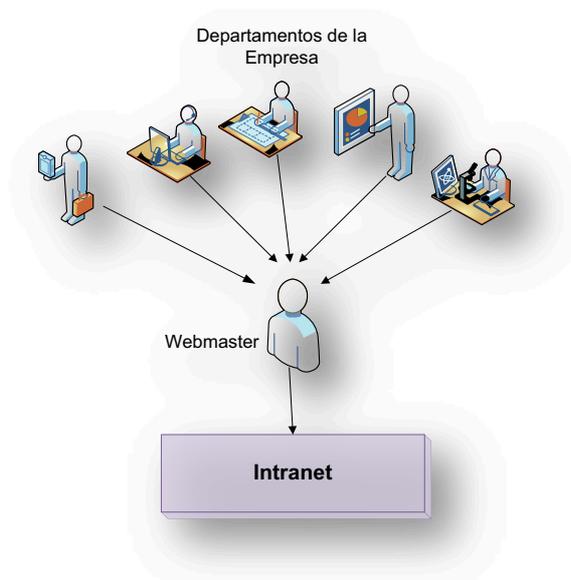


Figura 1.6 El Webmaster en la Intranet.

El webmaster debe contar con ciertas habilidades, las cuales le permitirán realizar un correcto mantenimiento del contenido web y una óptima administración de la intranet. Entre algunas de las habilidades con las cuales debe contar, están:

- Capacidad de análisis y resolución de problemas.
- Amplio conocimiento de Internet, lo cual incluye correo electrónico, ftp y telnet.
- Manejo de protocolos básicos de Internet.
- Dominar estándares como W3C.
- Conocimientos mínimos en algún lenguaje de programación como por ejemplo; Perl, Java, Java Script, PHP, entre otros.
- Habilidad para crear contenidos y manipulación digital de imágenes.
- Conocimiento acerca de teoría de redes y capas de red.
- Capacidad para la administración de seguridad.

Además el webmaster debe tener ciertos conocimientos para administrar y configurar servidores, independientemente de la plataforma en la que se encuentren.

1.2.1 FUNCIONES Y RESPONSABILIDADES DEL WEBMASTER

El Webmaster requiere de múltiples habilidades debido a las tareas que debe realizar dentro de la intranet y a la complejidad de las mismas, además debe estar en continuo aprendizaje debido a que la Web cambia continuamente. Es por esta razón que el perfil del webmaster va cambiando de acuerdo a las necesidades actuales de la empresa u organización. Un ejemplo de este cambio es la integración del webmaster en grupos de trabajo, asumiendo así cada vez más tareas de coordinación, ya que la tendencia actual es trabajar en grupos de varias personas, esto se aplica para grandes o medianas empresas, ya que en empresas pequeñas el webmaster se encargará de todas las tareas de diseño, mantenimiento y administración de los servicios de la intranet.

En el caso de grandes empresas, tareas como diseño y programación de páginas web dinámicas, pueden ser delegadas a cualquier miembro del grupo de trabajo, entonces el webmaster será quien tenga la visión general del proyecto y quién contemple los objetivos presentes y futuros de los servicios que se administran en

la Intranet. Debido a que el webmaster pertenece a un grupo de trabajo, debe contar con una amplia disponibilidad de trabajar conjuntamente con cada uno de los miembros de dicho grupo. Este grupo de trabajo debe estar conformado por representantes de cada una de las áreas de la organización con el fin de obtener mejores resultados.

El Webmaster debe estar familiarizado con el manejo de sistemas operativos y requisitos de hardware y software correspondientes a cada plataforma. Es decir el webmaster debe contar con un cierto dominio de la plataforma que se use en la intranet. Debido a que en la intranet continuamente se instalarán paquetes software o servicios, el webmaster debe estar en capacidad de instalarlos, realizar pruebas y darles el correspondiente mantenimiento y monitorización.

Las funciones del webmaster pueden dividirse en administración lógica y administración física. La administración lógica se refiere a la seguridad de la intranet y al mantenimiento de los servicios web. Además se debe contemplar dentro de este tipo de administración al plan de direccionamiento TCP/IP y a la administración de protocolos. En tanto la administración física se dedica a la parte hardware del servidor, es decir al mantenimiento de éste. Al mismo tiempo se refiere a la infraestructura en general de la intranet como por ejemplo dispositivos de acceso a Internet.

1.2.2 TIPOS DE WEBMASTER

La clasificación del webmaster se realiza en base al tipo de responsabilidades que adquiera y a las capacidades que posea. Tomando en cuenta estas características se presenta la siguiente clasificación:

- **Webmaster Técnico**

Se define como aquel webmaster que domina la programación en lenguajes para el desarrollo de páginas web dinámicas y es capaz de configurar servidores. Administra técnicamente sitios con un grado de complejidad. Las áreas en las que se desenvuelve son administración y programación.

- **Webmaster Funcional**

Se define como aquel webmaster que se dedica a la actualización, gestión y administración de contenidos. Este tipo de webmaster carece de conocimientos técnicos, por lo tanto el software requerido se le es entregado de antemano.

1.2.3 SELECCIÓN DEL WEBMASTER

El webmaster requiere de ciertos conocimientos dependiendo del tamaño de la empresa u organización en la que desempeñe su trabajo. En el caso de una pequeña empresa el webmaster generalmente cumple también con la función de administrador de la intranet. Esto no suele ser lo más conveniente, debido a que el webmaster tiene definidas sus propias responsabilidades, al ser además el administrador de la red, estas nuevas responsabilidades pueden hacer que el webmaster descuide sus obligaciones.

En medianas o grandes empresas generalmente se cuenta con departamentos de redes y sistemas, a los cuales se les otorga la responsabilidad de administrar y dar mantenimiento a los servicios web de la intranet. En este caso a pesar de que esta responsabilidad recae sobre un departamento en el que se cuenta con un mayor personal, se puede cometer el mismo error que en el caso anterior, debido a que cada persona tiene ya definidas ciertas responsabilidades inherentes a su función en la empresa, y al asignarle responsabilidades de webmaster se corre el riesgo de que el personal descuide sus verdaderas obligaciones. Es por esta razón que en este tipo de empresas se asigna el rol de webmaster a un grupo específico dentro de la intranet. De esta forma los servicios ofrecidos dentro de la intranet se desarrollarán eficientemente y en un tiempo óptimo. Este grupo será el encargado de definir las herramientas y los métodos necesarios para la administración de los servicios web de acuerdo a las necesidades de la empresa.

A pesar de contar con un grupo específico que desempeñará las tareas de webmaster, no es muy factible que se sobrecargue de trabajo al personal del que se dispone en dicho grupo. Debido a que esto dependerá de las necesidades y del tamaño de la intranet.

1.2.4 ALTERNATIVAS AL WEBMASTER

Una alternativa al webmaster son las compañías de apoyo que se dedican al préstamo de servicios Web. Estas compañías de apoyo se encargan de ayudar a todo tipo de empresas a codificar el formato HTML de las páginas, en cuanto a estándares de la Web y a seleccionar una plataforma adecuada de servidor. En la práctica puede resultar mucho más factible subcontratar este tipo de servicios, ya que la empresa puede no contar con los recursos necesarios ni con el personal lo suficientemente capacitado para abordar este tipo de proyectos. Esto ocurre principalmente en medianas y pequeñas empresas debido a la gran inversión que esto implica, por otro lado, se encuentra la necesidad de las mismas de contar con un sistema que permita administrar los servicios dentro de su intranet, más aún si esto se lo puede realizar vía Web. Es así que empresas que carezcan de un webmaster o de personal que se encargue de realizar esta tarea, es preciso la colaboración externa.

Contar con una buena compañía de apoyo que asista a la empresa en el desarrollo de este tipo de proyectos en la intranet, puede ser difícil de encontrar, debido a que se debe optar por una que cuente con la experiencia práctica necesaria en este campo. Además será conveniente que posea una amplia experiencia en la generación de páginas web dinámicas.

1.3 SERVIDOR WEB

Un servidor hace referencia tanto a la parte hardware como software. Para definir a un servidor se puede hacer referencia a la arquitectura cliente – servidor, el cliente es quien realiza las peticiones o requerimientos a través de una interfaz y el servidor responde con datos la petición solicitada, es decir que el servidor es el encargado de proveer los servicios requeridos a los usuarios que lo solicitan.

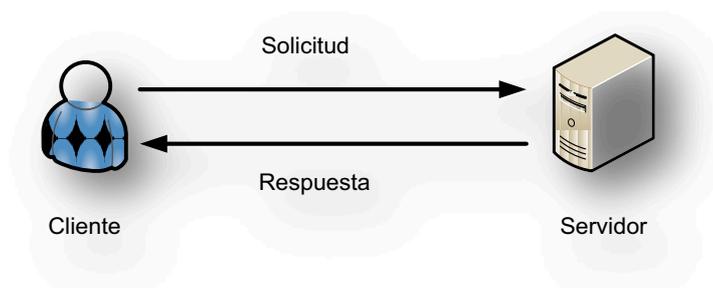


Figura 1.7 Relación Cliente – Servidor.

En el presente capítulo se hará énfasis en el servidor web, debido a que éste será utilizado en el desarrollo del prototipo.

Un Servidor Web provee datos en forma de páginas web con contenido de hipertexto o hipermedia a través de la red al navegador web del cliente que lo solicita.

Con el fin de aumentar la funcionalidad del servidor web más allá de que éste entregue contenido netamente estático y sea capaz de proveer contenido dinámico, se puede hacer uso de ciertas tecnologías como por ejemplo programas CGI, seguridad SSL, entre otras.

1.3.1 FUNCIONAMIENTO DEL SERVIDOR WEB

La comunicación entre el cliente y el servidor web, se detalla a continuación: al ingresar en el navegador una dirección (URI), por ejemplo:

<http://www.misitio.com.ec/>, inmediatamente éste realizará una petición HTTP al servidor asociado a dicha URI o dirección IP, según sea el caso. El servidor responde enviando el código HTML de la página, el navegador lo interpreta y lo muestra en pantalla. Es muy importante señalar que la función del servidor en este caso es únicamente la de transferir el código más no de interpretarlo, ya que dicha tarea le corresponde al navegador web del cliente.

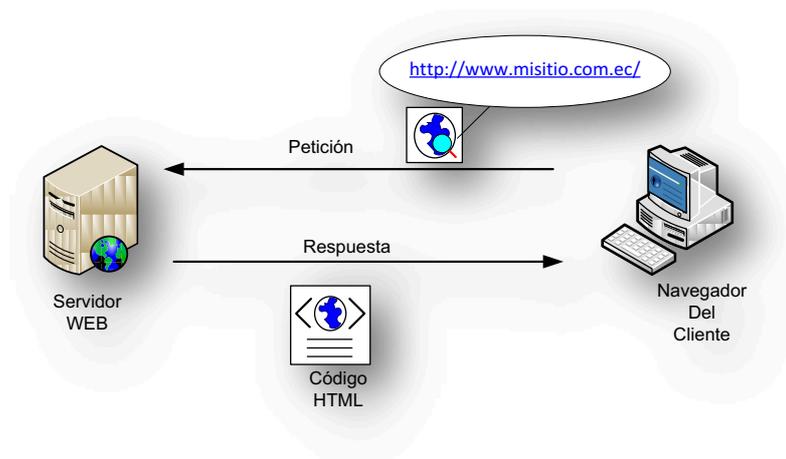


Figura 1.8 Comunicación Cliente – Servidor.

El servidor web es el encargado de proporcionar el acceso a los recursos solicitados (programas CGI/servlet, archivos, etc.) por el cliente y que se

encuentran bajo el control del sistema operativo. Dichos recursos pueden ser documentos estáticos como páginas HTML, archivos o documentos dinámicos como programas CGI, los cuales son ejecutados por el servidor.

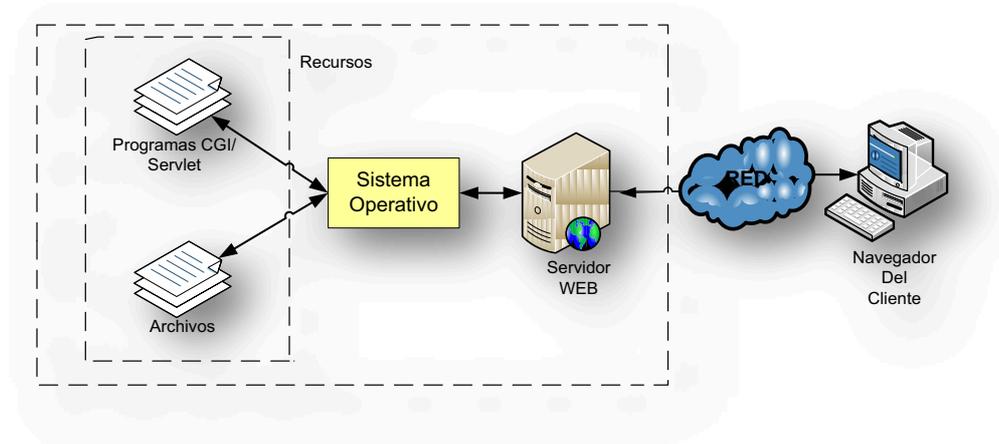


Figura 1.9 Funcionamiento de un Servidor Web.⁸

1.3.2 ARQUITECTURA⁹

En la figura 1.10 se muestra una arquitectura típica de un servidor web, en dos capas:

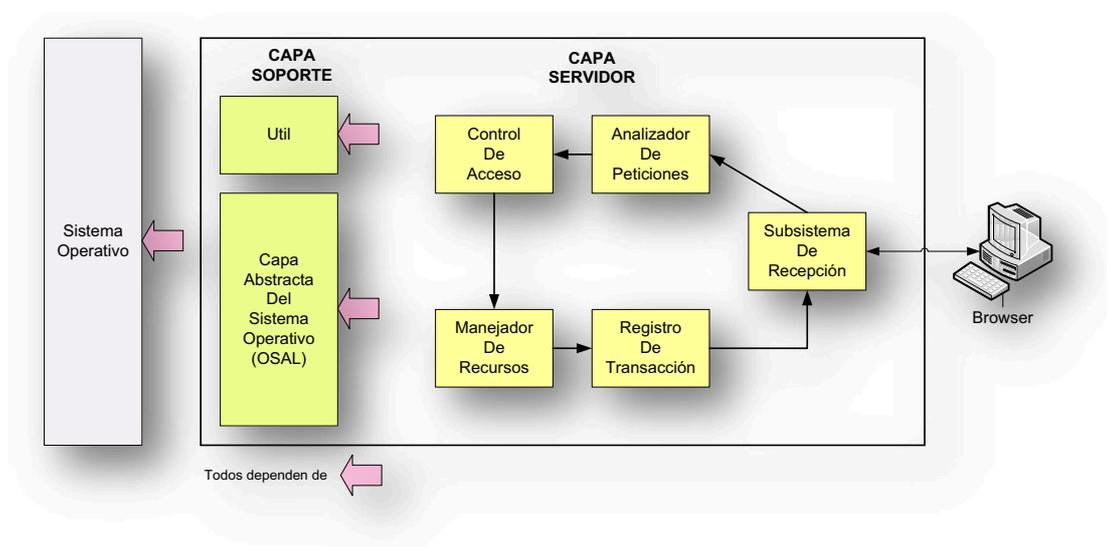


Figura 1.10 Arquitectura de un Servidor Web.

⁸ HASSAN E., Ahmed; HOLT C., Richard. **A Reference Architecture for Web Servers**. Tomado de: <http://www.bauhaus-stuttgart.de/dagstuhl/HassanHolt.pdf>

⁹ HASSAN E., Ahmed; HOLT C., Richard. **A Reference Architecture for Web Servers**. Tomado de: <http://www.bauhaus-stuttgart.de/dagstuhl/HassanHolt.pdf>

Este tipo de arquitectura está constituida por dos capas capa servidor y capa soporte.

- **CAPA SERVIDOR**

Esta capa está compuesta por cinco subsistemas, los cuales se encargan de implementar la funcionalidad de un servidor Web. Consta de los siguientes subsistemas:

Subsistema de Recepción: Su función es esperar por las peticiones HTTP de los clientes. Además analiza su contenido y determina las capacidades de los navegadores, es decir compatibilidad, tipo de navegador, etc. Este subsistema puede manejar múltiples peticiones.

Analizador de Peticiones: Este subsistema se encarga de traducir la localización del recurso de la red al nombre del archivo local. Por ejemplo, si se solicita el recurso `http://www.misitio.com.ec/index.html` se traduce al fichero local `/var/www/html/index.html`.

Control de Acceso: Se encarga de autenticar y permitir el acceso. Es decir que en caso de ser requerido, para permitir el acceso a los recursos solicitados requiere del ingreso de un nombre de usuario y contraseña.

Manejador de Recursos: Se encarga de determinar el tipo de recurso solicitado por el navegador web, lo ejecuta y genera la respuesta.

Registro de Transacción: Este subsistema se encarga de registrar todas las peticiones y su correspondiente resultado.

- **CAPA SOPORTE**

Esta capa actúa como interfaz entre el sistema operativo y los subsistemas de la capa servidor. Está constituido por los siguientes subsistemas:

Útil: Está compuesta por diversas funciones que son utilizadas por los demás subsistemas. Por ejemplo, contiene las funciones para la manipulación de cadenas o URI's.

OSAL (Operating System Abstraction Layer o Capa Abstracta del Sistema Operativo): Encapsula el funcionamiento específico del sistema operativo con la finalidad de facilitar la portabilidad del servidor Web a diferentes plataformas.

1.3.3 TIPOS DE SERVIDORES WEB¹⁰

Existen muchos tipos de servidores, pero su función en común es proporcionar el acceso a los servicios, archivos u otra información presente en el servidor a los hosts conectados a él. Existen varios tipos de servidores y entre los cuales podemos mencionar los siguientes:

1.3.3.1 Servidores Basados en Procesos¹¹

Se basa en la obtención de paralelismo mediante la duplicación del proceso de ejecución. Un diseño basado en procesos es aquel en el que el proceso principal espera por una nueva conexión, en ese momento se duplica creando una copia exacta que será la encargada de atender esa conexión.

Apache está basado en este diseño utilizando la técnica pre-fork, dicha técnica consiste en la creación previa de un grupo de procesos y su mantenimiento hasta que sea necesaria su utilización. Su simplicidad de implementación y su seguridad son una de las ventajas que ofrece este tipo de diseño.

La creación o eliminación de procesos implican tareas realmente pesadas para el sistema operativo y además consumen una gran cantidad de tiempo, es por esto que una gran desventaja de este diseño es su bajo rendimiento.

1.3.3.2 Servidores Basados en Hilos¹²

Los servidores basados en hilos o threads son un diseño mucho más común que el anterior. Este diseño comprende los conceptos básicos de funcionamiento del servidor basado en procesos. La principal ventaja de este diseño es que la creación de un hilo no es tan costosa como la creación de un proceso. Varios

¹⁰ alucard833: **Tipos de Servidores Web.**

Tomado de: <http://www.gratisblog.com/index.php?itemid=124960>

¹¹ **Procesos.-** Un proceso es un programa en ejecución. Un proceso es propietario de una serie de recursos como: ficheros, hilos, un espacio de direcciones de memoria, etc.

¹² **Hilos.-** Un hilo de ejecución se encarga de ejecutar el código contenido en el espacio de direcciones del proceso. Permite a una aplicación realizar varias tareas concurrentemente. Pueden compartir los mismos recursos. Un proceso puede contener varios hilos.

hilos de un mismo proceso pueden compartir datos entre ellos, debido a que comparten el mismo espacio de memoria. Pero el compartir el mismo espacio de memoria implica un riesgo de seguridad, que lo pone en desventaja frente al diseño anterior.

La simplicidad de diseño e implementación son unas de las características que hereda del servidor basado en procesos.

1.3.3.3 Servidores Basados en Sockets¹³ dirigidos por Eventos

El funcionamiento de este diseño se basa en la utilización de lecturas y escrituras asíncronas sobre sockets. Este tipo de servidores utilizan una llamada al sistema con el fin de examinar el estado de los sockets con los que trabajan.

El objetivo de estas funciones es verificar el estado de un grupo de sockets asociados a cada una de las conexiones. La velocidad es una de las ventajas de este diseño. En este diseño existe un solo proceso y un solo hilo, desde el cual se atienden todas las conexiones, es decir que se tiene una concurrencia aparente, la cual es su principal desventaja.

1.3.4 CÓDIGOS DE RESPUESTA DEL SERVIDOR WEB

Son generados por el servidor cuando el navegador web del cliente no puede mostrar la página web solicitada, éste código es mostrado en el navegador web del cliente e indica el motivo por el cual la solicitud no pudo ser realizada. Se encuentra formado por tres dígitos, el primero indica el estado, es decir: los códigos que inician con 2xx indican éxito, los códigos 3xx indican redirección, los códigos 4xx indica error debido al cliente y 5xx indican error debido al servidor, el segundo y tercer dígito corresponden al error específico.

A continuación se detallan los códigos de respuesta que el servidor devuelve con más frecuencia:

- **OK 200:** Este mensaje indica que todo se ha efectuado satisfactoriamente.
- **Moved 301:** Indica que los datos se han redireccionado a una nueva dirección.

¹³ **Sockets.-** Son puntos o mecanismos de comunicación entre procesos, estos permiten que un proceso se comunique con otro incluso estando en distintas máquinas.

- **Unauthorized 401:** Este mensaje indica que el cliente no tiene autorización para realizar operaciones.
- **Forbidden 403:** El script o programa tiene errores en los permisos.
- **Not found 404:** Este mensaje indica que la petición no ha sido encontrada.
- **Internal Error 500:** Indica que el error es del servidor.

1.3.5 HARDWARE Y SOFTWARE

Un servidor Web se compone básicamente de dos elementos:

- Hardware
- Software para Servicios Web

Estos elementos son necesarios para poder emitir las páginas web al cliente que lo solicite. La elección tanto de hardware como de software, dependerá del ambiente en el cual se desee implementar el servidor Web, así como el nivel de seguridad y nivel de servicio que se desee brindar.

1.3.5.1 Hardware

El servidor Web perfecto para una intranet, es un servidor dedicado. Generalmente los servidores Web suelen ser estaciones Linux, o puede también optarse por servidores Web Windows, Solaris, entre otros, los cuales permiten que máquinas basadas en procesadores básicos funcionen como servidores dedicados con un rendimiento aceptable.

Los requerimientos hardware¹⁴ para un servidor Web dependerán del nivel de servicio que se desee ofrecer, si se considera un servidor Web en Linux el cual brindará servicios a una pequeña empresa los requerimientos mínimos son los siguientes:

- Un procesador 250 MHz o superior.

¹⁴ Linux para Todos: **Servidor Web**,
Tomado de: <http://www.linuxparatodos.net/portal/staticpages/index.php?page=servidor-web>

- 256 MB de RAM.
- Una unidad de disco duro de 8 GB.

1.3.5.2 Software

El software para un servidor Web, comúnmente suele llamarse “Servidor Web”. Es un software que trabaja conjuntamente con el sistema operativo para suministrar las páginas web que los navegadores soliciten. Algunos de los servidores web que se pueden encontrar son: Apache, Cherokee, IIS, Nginx, Lighttpd, entre otros.

- **APACHE:** Es el servidor Web más utilizado, con una utilización aproximada del 65%. Es gratuito y de código abierto, lo cual permite su modificación de acuerdo a las necesidades del lugar en el que sea implementado. Es un servidor basado en procesos, utilizando las técnicas prefork o worker. Las principales metas de su diseño son: velocidad, simplicidad, multiplataforma y facilidad del desarrollo distribuido. Apache es usado principalmente para enviar páginas web estáticas y dinámicas en la web. Es también usado para tareas en las que se requiera de confiabilidad y seguridad de los contenidos. Apache utiliza una arquitectura con módulos de multiprocesamiento conocidos como MPM (Multi-Processing Modules)¹⁵.
- **CHEROKEE:** Es un servidor web gratuito, multiplataforma y de código abierto, el cual se destaca por su flexibilidad, velocidad y facilidad de uso.

Su diseño combina características de servidores basados en sockets no bloqueantes o dirigidos por eventos y servidores basados en hilos, con el fin de obtener los beneficios que los dos modelos brindan y reducir las desventajas. Este servidor procesa peticiones en cada uno de sus hilos, estos hilos se generan cuando arranca el servidor y permanecen hasta que termina su ejecución. Puede trabajar bajo la plataforma Linux o Windows.

IIS: Son las siglas de Internet Information Server, es un componente software que admite la creación, configuración y administración de sitios Web. Además

¹⁵ **Multi-Processing Modules (MPM).**- Módulos de Multiprocesamiento, son módulos que permiten al servidor Apache trabajar bajo diferentes sistemas operativos, así como realizar diferentes tareas de multiprocesamiento.

incluye otros servicios de internet para la plataforma Windows como: servicio de transferencia de ficheros por red (FTP), servicio para la transferencia de noticias (NNTP), servicio de transferencia de correo electrónico (SMTP). Es importante resaltar que IIS solo funciona sobre la plataforma Windows, si se desea emplearlo en otra plataforma se deberá utilizar una máquina virtual.

- **Ngnix:** Este es un servidor Web muy ligero y corre sobre las plataformas Windows y Linux. Se distribuye bajo la licencia BSD¹⁶. Se ha convertido en el cuarto servidor más utilizado.
- **Lighttp:** Este servidor Web fue diseñado para realizar tareas pesadas usando poca RAM y poco CPU. Este servidor es utilizado por páginas como por ejemplo youtube y otras que se caracterizan por soportar gran tráfico diariamente, es gratuito y se distribuye bajo licencia BSD.

1.4 HTML (Hyper Text Markup Language o Lenguaje de Mercado de Hipertexto)

HTML de manera simple no es más que un lenguaje empleado para la creación de páginas web, el cual permite la distribución y publicación de información de manera global a través de la WWW¹⁷, los creadores de páginas web emplean este tipo de lenguaje para el diseño de sus proyectos, ya que las herramientas de desarrollo web generan páginas mediante la creación de documentos HTML.

Al ser HTML un lenguaje de distribución global los navegadores web (clientes) deben estar en la capacidad de mostrar la información de manera similar independiente del tipo de navegador.

Para permitir la publicación global de la información, HTML debe contar con algún tipo de estandarización, la entidad encargada de definir el estándar y sus normas es la W3C World Wide Web Consortium que es un organismo sin fines de lucro

¹⁶ **Licencia BSD:** Es una licencia de distribución de software concedida principalmente para los sistemas BSD (Berkeley Software Distribution), la cual permite a los programadores utilizar, modificar y distribuir el código fuente y el código binario del programa original.

¹⁷ **World Wide Web (WWW).**- Constituye la colección de documentos de hipertexto o hipermedia enlazados y accesibles a un cliente a través de un navegador web, generalmente se denomina WEB.

creado para definir los diferentes mecanismos, normas y estándares dentro del campo de la WWW. Todos los estándares se encuentran en la página oficial de la W3C, por ejemplo, las normas oficiales para HTML 4.01 se pueden consultar en:

<http://www.w3.org/TR/html401/>

HTML provee al desarrollador de páginas web mecanismos para:

- Desarrollar y publicar documentos con encabezados, textos, tablas, listas, entre otras modificaciones sobre el texto.
- Obtener información a través de vínculos en hipertexto.
- Desarrollar formularios para establecer comunicación con servicios dentro de un servidor por ejemplo para buscar información, realizar consultas, solicitar información, etc.
- Incluir contenido multimedia como video, sonido, y otras aplicaciones dentro de la página web.
- Implementar hojas de estilo para el diseño visual de la página web. Lo cual permitirá diferenciar el código HTML empleado para estructurar el documento HTML y el código empleado para aplicar un diseño a la estructura del documento HTML.

1.4.1 CARACTERÍSTICAS BÁSICAS

Todo documento HTML cuenta con un conjunto de características comunes y básicas que deben ser utilizadas al momento de implementar una página web para cumplir con los estándares definidos por la W3C, un ejemplo de estas características es el manejo de la sintaxis dentro de un documento HTML y cuales elementos son indispensables para su funcionamiento. Sin estos conocimientos básicos sería difícil entender cómo funciona el lenguaje HTML.

1.4.1.1 HTML un Lenguaje de Etiquetado

Un reto para la informática en sus inicios constituyó la manera de almacenar información en un archivo digital, para ello se relacionó cada letra con un correspondiente valor numérico permitiendo almacenar la información como una cadena numérica, al momento de recibir este archivo digital se realizaba el

proceso inverso empleando tablas de conversión para obtener el contenido textual. A este tipo de conversión se la conoce como codificación de caracteres, una de las codificaciones más conocidas es ASCII y se la emplea en HTML para codificar caracteres especiales.

Ahora, si al texto almacenado mediante codificación en un archivo le queremos añadir formato como: negrita, cursiva, tamaño, etc. el almacenamiento cambia, para ello junto a la información se tendrá que añadir el formato. Es el caso del siguiente ejemplo:

`<parrafo>`

Texto informativo con `<importante>` palabras `</importante>` resaltadas

`</parrafo>`

A la información: *texto informativo con palabras resaltadas*, se ha añadido la información correspondiente a `</parrafo>...<parrafo>` que indica que la información está en forma de párrafo e `<importante>...</importante>` que especifica que la información **palabras** tiene importancia en relación al resto.

El proceso de indicar los distintos formatos para la información dentro del documento se denomina marcado (markup en inglés) mientras que las palabras para indicar el inicio y fin de un formato en el documento se denomina etiqueta.

Las etiquetas (aunque existen excepciones) se indican por pares, una etiqueta de apertura y una de cierre:

Apertura: `<nombre de etiqueta>`

Cierre: `</nombre de etiqueta>`

Tomando en cuenta estos aspectos se tiene claro de donde proviene HTML, HTML no es más que un lenguaje de etiquetado (markup language) que permite generar información con distintos formatos para que sean visualizados por medio de un navegador web, los documentos web emplean cientos de etiquetas para proporcionar información por esta razón las etiquetas en HTML emplean nombres cortos, lo cual permite disminuir el tamaño del documento final HTML.

1.4.1.2 Atributos

Como se mencionó en el párrafo anterior HTML emplea un lenguaje de etiquetado, cada una de estas etiquetas cumplen funciones específicas dentro del texto, por ejemplo, la etiqueta `<h1>...</h1>` proporciona un tamaño específico para el texto que se encuentra dentro de ésta. Algunas etiquetas no son tan simples pues requieren cierta información adicional para realizar una acción específica, a esta información se la conoce como atributo. Es el caso de la etiqueta `<a>` una etiqueta de enlace, por sí sola no realiza ninguna acción, al añadir información dentro de esta etiqueta se creará un vínculo hacia una URI específica. En el siguiente elemento:

```
<a href="http://www.miweb.com"> Este enlace apunta a mi página </a>
```

Se puede ver claramente que el atributo href permite crear un vínculo hacia la dirección www.miweb.com y será activada cuando se realice un clic sobre la frase: Este enlace apunta a mi página.



Figura 1.11 Ejemplo de un atributo.

Cada etiqueta cuenta con un conjunto de atributos que se pueden aplicar dependiendo de la acción que se quiera realizar, existen atributos comunes que pueden ser utilizados por la mayoría de etiquetas, por ejemplo para definir el idioma que se va a emplear dentro del documento. Cuando el documento HTML va a generar una página web dinámica, es decir que va a emplear formularios, se cuenta con atributos de eventos para manejar distintas acciones que se realicen

sobre el formulario, por ejemplo, al dar clic sobre un botón, se puede verificar que los campos dentro del formulario estén llenos.

Entre los principales atributos comunes se tiene:

- ***id="texto"*** especifica un identificador único para el elemento que emplee esta etiqueta.
- ***class="texto"*** indica el nombre de la clase para que CSS aplique un estilo al elemento HTML.
- ***style="texto"*** especifica un estilo CSS de manera directa sobre el elemento HTML.
- ***title="texto"*** permite establecer un título para el elemento HTML, por ejemplo, cuando se pasa el cursor sobre un enlace este indica un texto descriptivo del elemento.
- ***lang="código de idioma"*** especifica el idioma empleado por los valores de los atributos y la información contenida en un elemento HTML.
- ***tabindex = "numero"*** especifica el orden de tabulación de un elemento HTML dentro de una página web.

Para emplear páginas que utilicen el lenguaje JavaScript, se definen algunos atributos que permiten generar acciones cuando se realiza un evento, a éstos se los conoce como atributos de eventos, estos atributos son: onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup.

- ***Elemento HTML***

Se denomina elemento HTML al conjunto de partes que forman una marca dentro de un documento HTML. Por lo general está constituido por una etiqueta de inicio, cero o más atributos con sus valores, la información dentro de la etiqueta y una etiqueta de cierre. Algunos elementos no requieren de una etiqueta de cierre ni atributos pues han sido diseñados para realizar una tarea específica, es el caso del elemento `
` que realiza un salto de línea.

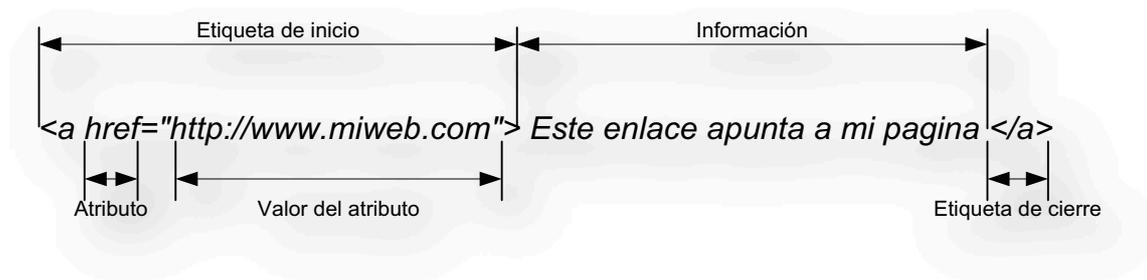


Figura 1.12 Elemento HTML.

La figura 1.12 muestra claramente como se distribuyen las partes dentro de un elemento HTML para el caso de una etiqueta de enlace `<a>` con el atributo `href` cuyo valor es <http://www.miweb.com>. La información de la etiqueta es *Este enlace apunta a mi pagina* y tiene como parte final la etiqueta de cierre ``.

La W3C clasifica a los elementos HTML en dos tipos:

- **En línea (inline):** son aquellos elementos que permiten visualizar un conjunto de caracteres, por ejemplo, una oración que tenga palabras en cursiva o negrita, un elemento en línea no emplea toda la longitud de la página solo emplea el espacio necesario para mostrar la información, de esta manera, si se escribe un conjunto de elementos en línea se mostrarán uno a continuación de otro.
- **En bloque (block):** son elementos que trabajan como bloques de texto, un conjunto de oraciones que se visualizan como párrafos o listas. Los elementos en bloque a diferencia de los en línea ocupan toda la longitud de la página, cuando se escribe un elemento en bloque a continuación de otro entre ellos se añade un espacio en blanco.

Los elementos en bloque pueden llevar dentro, tanto elementos en línea como otros elementos en bloque, mientras que los elementos en línea no pueden llevar ningún tipo de elemento de bloque.

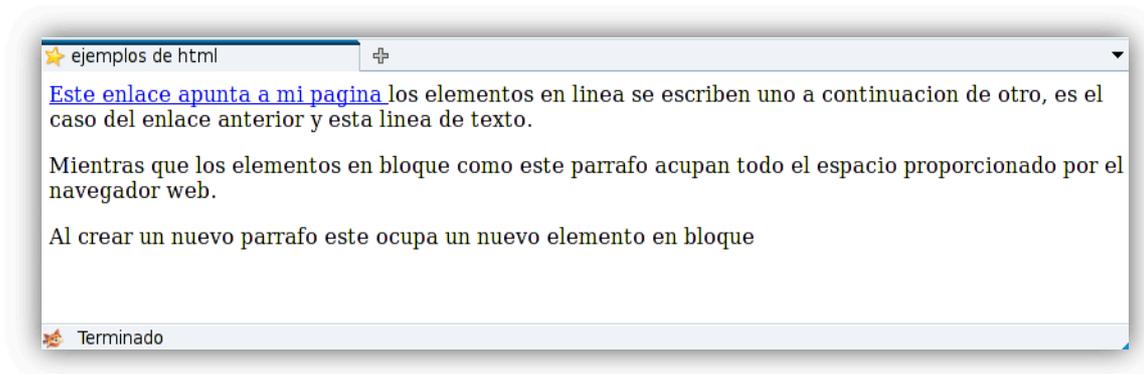


Figura 1.13 Visualización de elementos en bloque y línea.



Figura 1.14 Código HTML de elementos en bloque y línea.

1.4.1.3 Sintaxis en HTML

Cuando se crearon los primeros documentos HTML éstos podían ser escritos sin tomar en cuenta algunas normas que permitan el desarrollo correcto de un proyecto web, por esta razón la W3C publicó algunas normas que se deben tomar en cuenta al momento de escribir un documento HTML. Al cumplir con estas normas el navegador web va a interpretar correctamente el contenido del documento. Para ello:

- Al momento de utilizar una etiqueta ésta se debe cerrar de acuerdo a como se abre.

<p>mi texto esta fuerte en la pagina </p>

- Los nombres de las etiquetas y los atributos se escriben en minúsculas empleando su nombre asignado por la W3C.

```
< a href="http://mipagina.com">mi pagina web </a>
```

- Los valores en un atributo se escriben entre comillas.

```
< a href="http://mipagina.com">
```

- Como se mencionó en párrafos anteriores algunas etiquetas solo emplean una etiqueta de inicio y no requieren una etiqueta de cierre por ejemplo `
`.

También se debe tener presente la tabulación dentro del proceso de marcado pues permite un mejor entendimiento de cómo se dividen los distintos elementos dentro de la página web. HTML puede ser escrito en una sola línea o sin tabulaciones, pero esto no es profesional y no permite crear un documento ordenado para su fácil mantenimiento, ya que al emplear una tabulación ordenada se puede encontrar fácilmente las partes que requieran algún momento un cambio.

1.4.1.4 Estructura básica HTML

Un documento HTML básico se encuentra formado por:

- Una línea que contiene información acerca de la versión de HTML que utiliza el documento. HTML 4.01 especifica tres definiciones de tipos de documentos los cuales varían de acuerdo al tipo de elementos que se van a utilizar dentro de su contenido.
- **El DTD¹⁸ HTML 4.01 Estricto (Strict DTD)** incluye todos los elementos y atributos definidos en HTML 4.01. Su declaración es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

¹⁸ **Document Type Definition (DTD).**- Definición de Tipo de Documento, se encarga de definir qué elementos y atributos pueden ser empleados en el documento HTML.

- **El DTD HTML 4.01 Transicional (Transitional DTD)** incluye todo lo que incluye el DTD estricto más los elementos y atributos que son desaprobadados en HTML 4.01 (la mayoría de los cuales están relacionados con la presentación visual). Su declaración es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
"http://www.w3.org/TR/html4/loose.dtd">
```

- **El DTD HTML 4.01 para Documentos con Marcos (Frameset DTD)** incluye el DTD Transicional más los marcos. Su declaración es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"  
"http://www.w3.org/TR/html4/frameset.dtd">
```

- El elemento **<html>...</html>** que indica mediante sus etiquetas el inicio y el final del documento HTML. Ninguna etiqueta o contenido puede colocarse antes o después de la etiqueta **<html>** con excepción de la línea mencionada en el apartado anterior, En el interior de la etiqueta se definen la cabecera y el cuerpo del documento.
- El elemento **<head>...</head>** se encarga de delimitar la cabecera del documento y contiene la información a ser empleada por el documento HTML, por ejemplo, para definir hojas de estilo. La información correspondiente a la cabecera no es visualizada en el navegador con excepción del contenido de la etiqueta **<title>** que es utilizado por los navegadores como título de la ventana.
- El elemento **<body>...</body>** que delimita el cuerpo del documento HTML, contiene la información que el cliente puede visualizar en su navegador web.

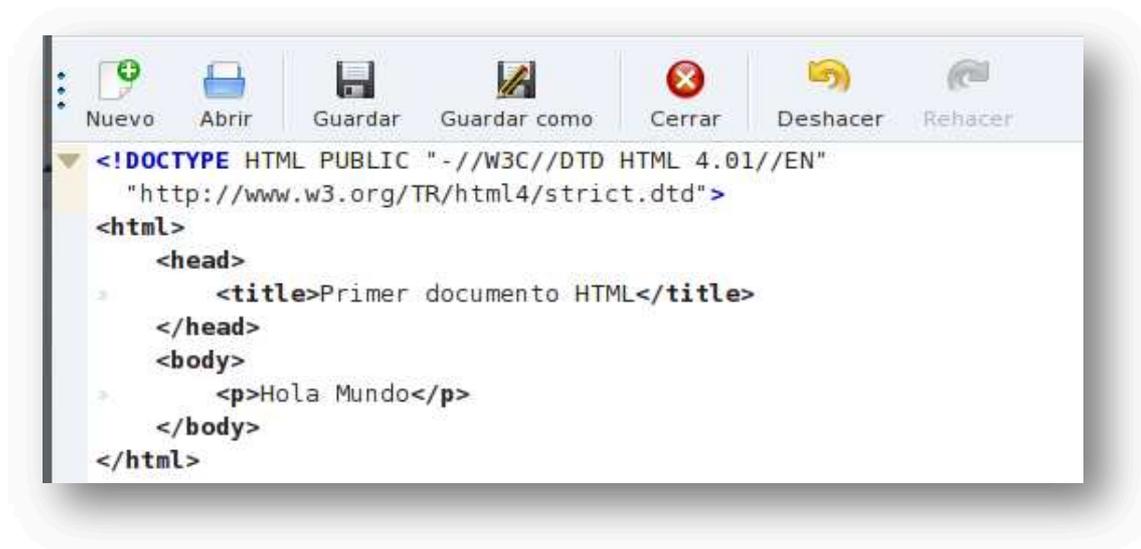


Figura 1.15 Elementos HTML básicos.

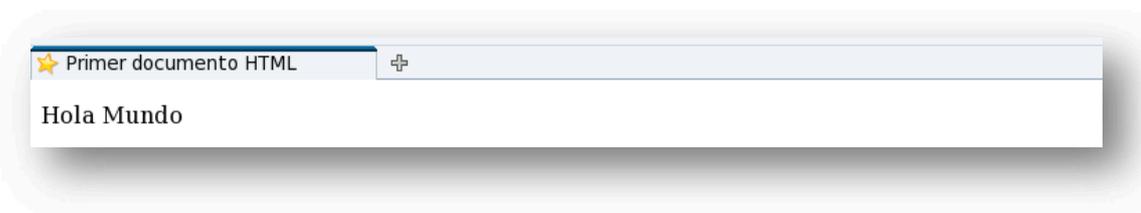


Figura 1.16 Página web con elementos básicos.

1.4.2 TEXTO (párrafos, líneas, frases)

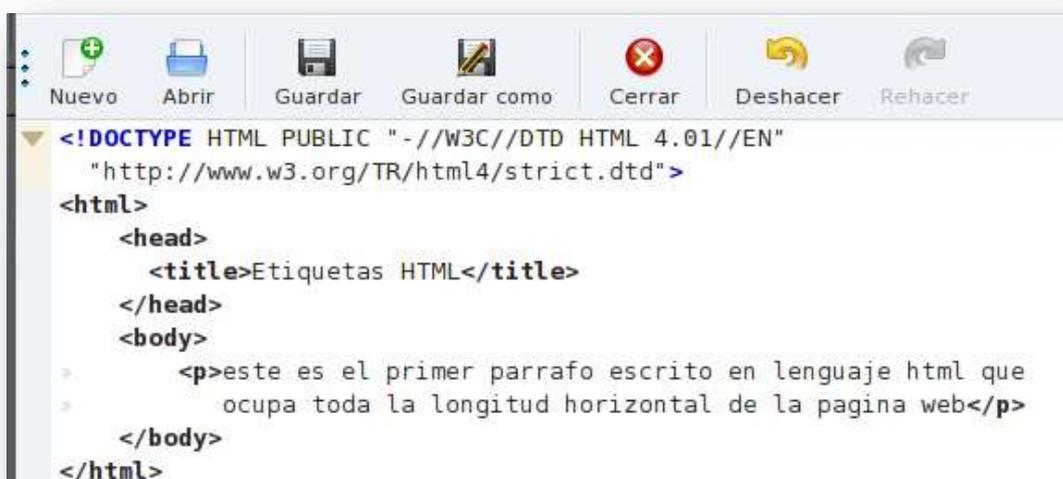
Una página web está diseñada para proporcionar información a manera de texto, imágenes, contenido multimedia, entre otros. El texto es el medio que proporciona la mayor cantidad de información, por esta razón requiere un tratamiento que permita distribuirlo de manera correcta dentro de una página web, para esto HTML define un conjunto de etiquetas que permiten estructurarlo de forma adecuada.

1.4.2.1 Párrafos y Líneas

En un documento HTML la información textual se maneja como conjuntos de texto ordenados de acuerdo al diseño del desarrollador, para el cliente en su navegador web la información se muestra como párrafos o secciones, HTML emplea las siguientes etiquetas para implementar párrafos y líneas:

- **Párrafos <p>...</p>**

La etiqueta párrafo es la más usada dentro del desarrollo de páginas web ya que permite separar la información textual como su nombre lo indica en forma de párrafos, la información dentro de esta etiqueta ocupa todo el espacio que el navegador puede proporcionar, con esto se puede apreciar que la etiqueta párrafo corresponde a un elemento de bloque. Puede emplear los atributos comunes de HTML para identificación, hojas de estilo, lenguaje. No tiene definidos atributos específicos.



```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Etiquetas HTML</title>
  </head>
  <body>
    <p>este es el primer parrafo escrito en lenguaje html que
    ocupa toda la longitud horizontal de la pagina web</p>
  </body>
</html>

```

Figura 1.17 El elemento párrafo.

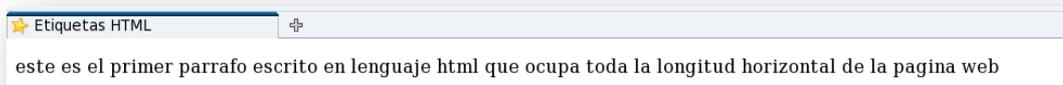


Figura 1.18 Visualización de un párrafo.

- **Encabezados <h1>...</h1>**

La etiqueta encabezado se utiliza para especificar títulos dentro de la información textual que se muestra en una página web, se definen 6 niveles de encabezado que van desde el nivel <h1> que es el más fuerte hasta el nivel <h6> que es el más débil o de menor importancia, de esta manera se definen 6 niveles jerárquicos dentro de un documento web donde pueden existir varios encabezados, los encabezados se los aplica de acuerdo a la importancia del texto

dentro del documento web. Al igual que la etiqueta párrafo constituye un elemento de bloque, no tiene atributos específicos y puede emplear atributos comunes.



Figura 1.19 Encabezados o títulos.

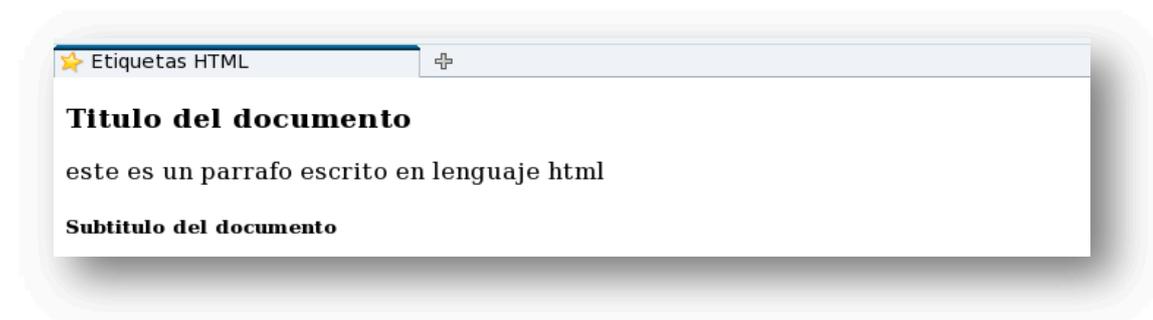


Figura 1.20 Visualización de títulos.

- **Nueva línea
**

Cuando se escribe algún texto por ejemplo, en el bloc de notas, las palabras son escritas en una sola línea, si se quiere emplear una nueva línea se tiene que pulsar la tecla *enter* y seguir con la escritura. La función de la etiqueta `
` es igual a la función que realiza la tecla *enter*, es decir si dentro de un párrafo colocamos esta etiqueta, la información que está luego de ella será mostrada en una nueva línea. La etiqueta `
` es una de las etiquetas HTML que no requiere información dentro de ella tomando el nombre de etiqueta vacía, ésta etiqueta se la define dentro del documento HTML de la siguiente manera `
`, no tiene definido atributos específicos y puede emplear atributos comunes.

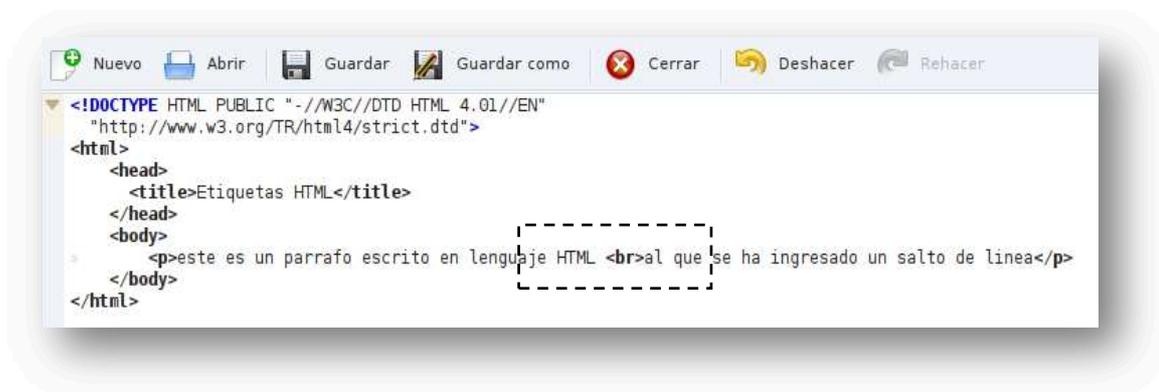


Figura 1.21 Añadir un salto de línea.

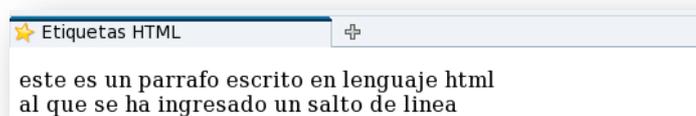


Figura 1.22 Visualización de un salto de línea.

- **Espacio en blanco**

Cuando se necesita ingresar dentro de un texto informativo en una página web espacios en blanco, no es tan simple como se realiza en la creación de una nueva línea, ya que se debe utilizar una codificación de carácter que se debe ingresar donde se quiera mostrar un espacio en blanco. Esta codificación es ** **; y se debe ingresar cada vez que se requiera un espacio en blanco dentro del texto.

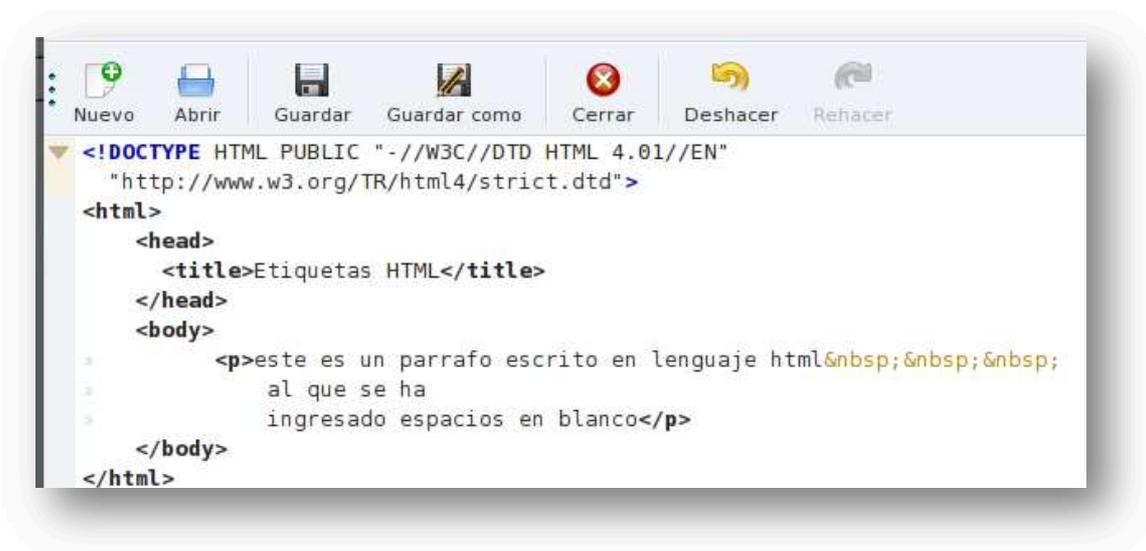


Figura 1.23 Inserción de espacios en blanco.

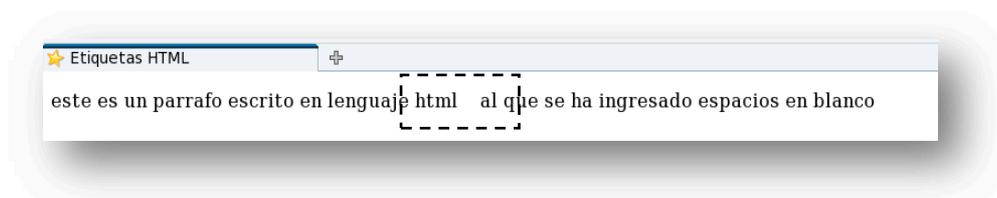


Figura 1.24 Visualización espacios en blanco.

- **Texto preformateado <pre>...</pre>**

La etiqueta `<pre>` es utilizada para mostrar texto que contenga espacios en blanco, nuevas líneas y tabulaciones pero de textos generados fuera del documento HTML. Es decir cuando se requiera mostrar información que no se haya escrito dentro del documento HTML sino que se haya obtenido como salida de algún programa.

Los navegadores tienen un tratamiento diferente para este tipo de texto, los muestran con un tipo de letra diferente al resto de la página respetando los espacios en blanco, nuevas líneas y tabulaciones.

Algo que hay que tomar en cuenta es que a diferencia de las etiquetas de bloque que respetan el espacio proporcionado por el navegador web, la etiqueta `<pre>` puede aumentar el espacio horizontal de la página si la información que muestra es de mayor longitud que la proporcionada por la página, activando el scroll horizontal para poder visualizar esta información. La etiqueta `<pre>` es un elemento de bloque puede emplear atributos comunes y no tiene atributos específicos.



Figura 1.27 Énfasis en el texto.

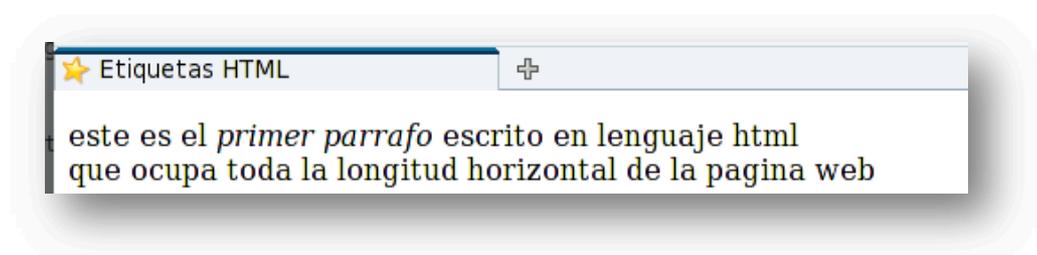


Figura 1.28 Visualización de énfasis a manera de letra cursiva.

- **Énfasis fuerte ...**

La etiqueta es utilizada para marcar las palabras de mayor importancia dentro del contenido textual, los navegadores web muestran el contenido de esta etiqueta como palabras en negrita. Al igual que la etiqueta es un elemento de línea puede utilizar atributos comunes y no define atributos específicos.

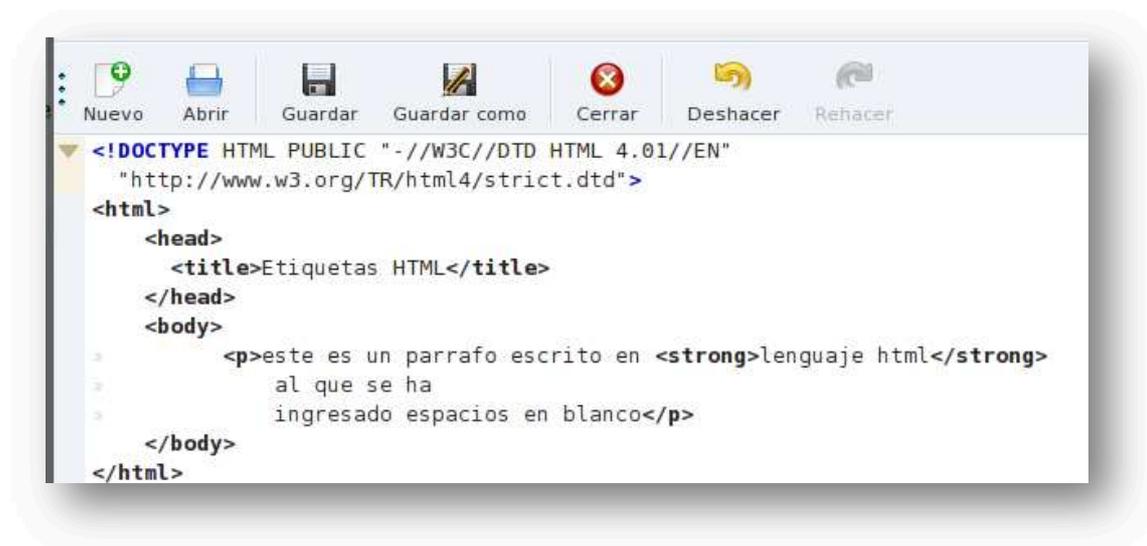


Figura 1.29 Énfasis fuerte en el texto.



Figura 1.30 Visualización de énfasis fuerte como texto en negrita.

1.4.2.3 Codificación de caracteres en HTML

Cuando se requiere emplear caracteres especiales distintos al idioma inglés como por ejemplo: á,é,ñ. o se requiere utilizar los signos empleados por las etiquetas `</>` o demás caracteres reservados para HTML, se emplea lo que se conoce como entidades HTML que no es más que un tipo de codificación que el navegador web tiene que interpretar para luego mostrar su caracter respectivo. Para el caso de caracteres distintos al idioma inglés se puede definir dentro del documento HTML la utilización de codificación UTF-8 para emplear cualquier caracter textual dentro del documento. Este tipo de codificación depende de varios factores externos al desarrollo de la página ya que depende por ejemplo del tipo de navegador que tenga el cliente o de los distintos procesos que tiene que realizar el documento para llegar al cliente, con la codificación UTF-8 no se asegura que se puedan observar los caracteres de manera correcta, lo que se soluciona empleando las denominadas entidades HTML.

Nombre	Caracter	Descripción
&amp;	&	Ampersand
&quot;	“	Comillas
&lt;	<	Menor que
&gt;	>	Mayor que
&nbsp;		Espacio en blanco
&apos;	‘	Apóstrofe

Tabla 1.2 Entidades para caracteres reservados en HTML.

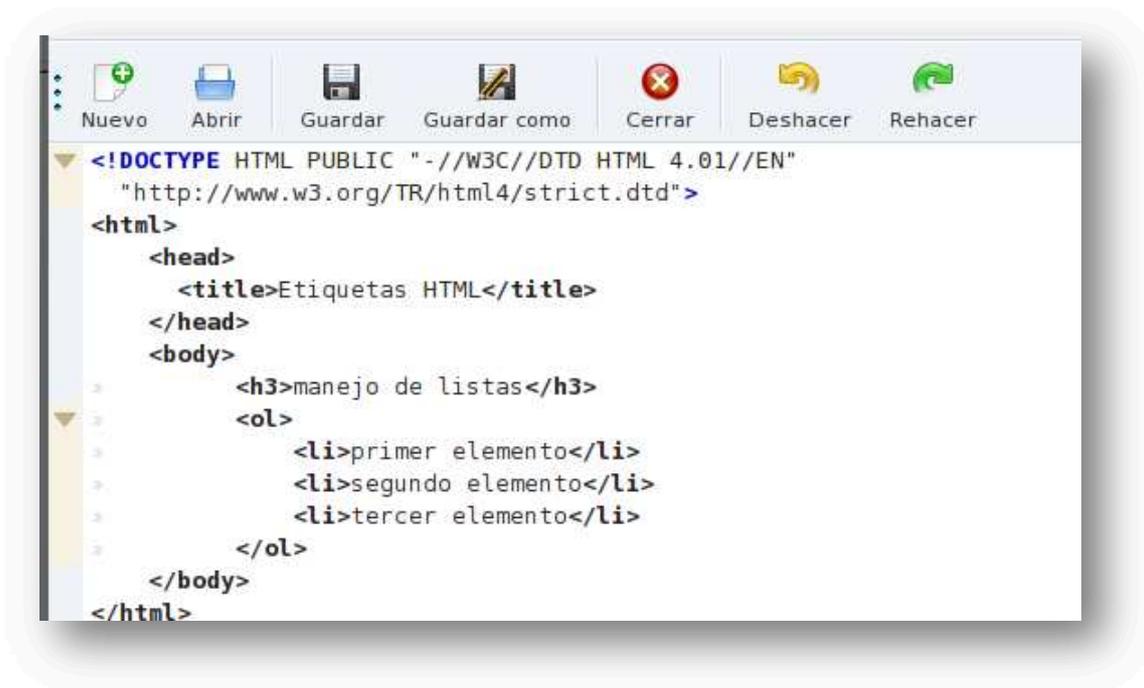


Figura 1.32 Generación de listas ordenadas.

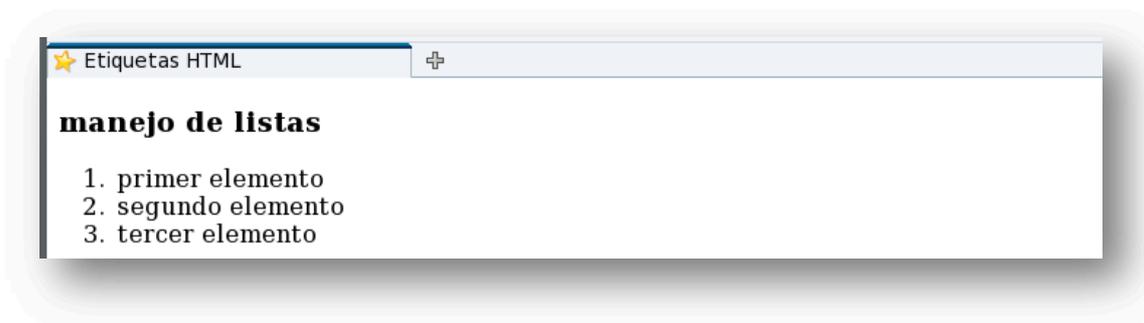


Figura 1.33 Visualización de listas ordenadas.

- **Listas no ordenadas** `...`

A diferencia de las listas ordenadas permite crear listas cuyos elementos no tiene una secuencia específica. Los elementos pueden ser escritos en cualquier lugar dentro de la lista. La etiqueta `` puede emplear atributos comunes, no define atributos específicos y pertenece a los elementos en bloque.

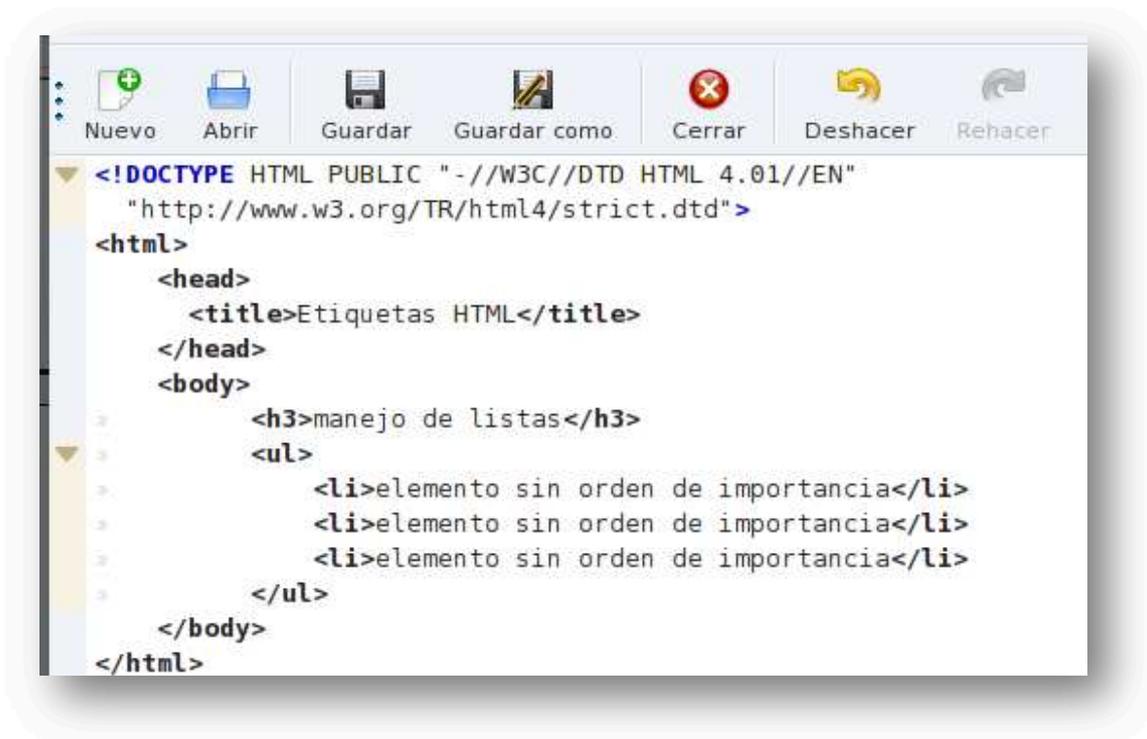


Figura 1.34 Generación de listas no ordenadas.

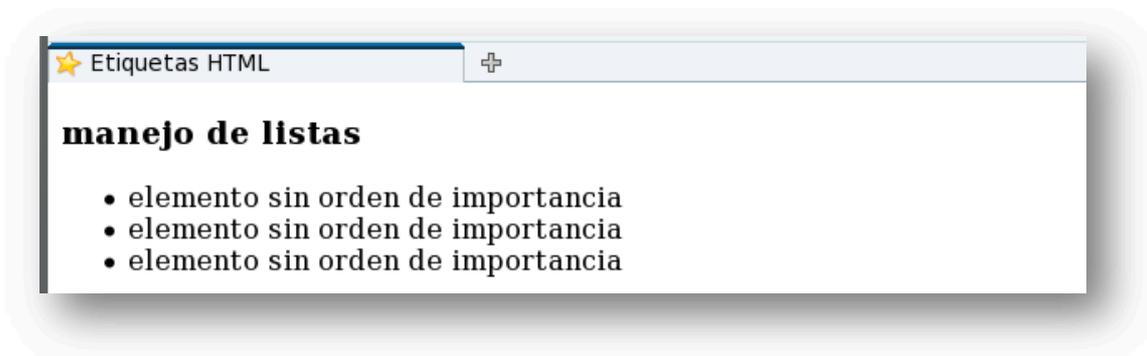


Figura 1.35 Visualización de listas no ordenadas.

1.4.4 ENLACES

La principal utilidad de HTML y la que ha permitido que tenga un desarrollo notable, es la posibilidad de manejo de hipertexto a través de vínculos o enlaces, para lograr crear documentos interactivos con la posibilidad de enlazar distintos recursos web. Un vínculo está formado por dos extremos denominados anclas (anchors) y una dirección (URI). La solicitud parte desde el extremo origen y apunta hacia el extremo destino. El destino proporciona diferentes recursos web como imágenes, documentos HTML, sonidos, etc. con la ayuda del URI.

1.4.4.1 URI (Uniform Resource Identifier o Identificador Uniforme de Recurso)

Todos los recursos dentro de la web ya sean imágenes, videos, audio, etc. tienen una identificación que permite diferenciarlos del resto de recurso, a este identificador se lo conoce como URI o identificador uniforme de recurso, constituye el elemento más utilizado al momento de implementar enlaces dentro de un documento HTML por lo que se hace indispensable su estudio.

Los URI se componen de tres partes:

- El protocolo de comunicación (http,ftp,https).
- El nombre del servidor (máquina que aloja el recurso web).
- Ruta (lugar donde se encuentra el recurso web dentro del servidor).

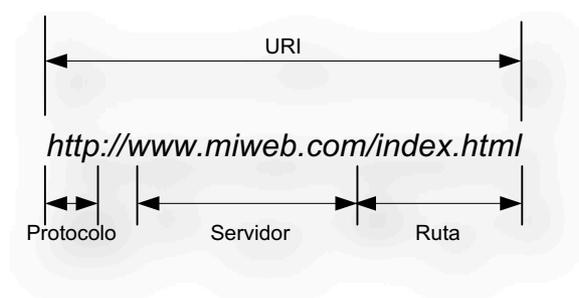


Figura 1.36 URI (Uniform Resource Identifier o Identificador Uniforme de Recurso).

La figura 1.36 muestra un URI básico. Cuando se requiere realizar una consulta a una página web por ejemplo a una base de datos o para dirigirse a una sección dentro de la página, al URI se agregan ciertos caracteres en la parte destinada a la ruta. En el caso de una consulta siempre comienza con el carácter ? y contiene una sucesión de palabras separadas por = y &. Para el caso de dirigirse a una sección de la página web se utiliza el carácter #.

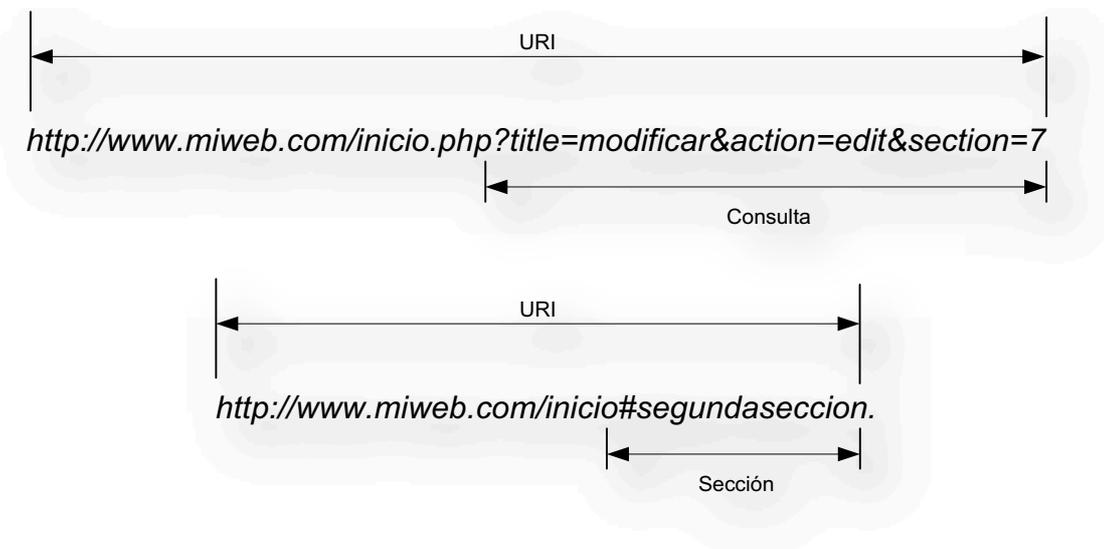


Figura 1.37 Consultas y sección.

- **URI absoluto y relativo**

Una página web puede tener varios enlaces dentro de su contenido, estos enlaces pueden conducir a páginas web que se encuentran ya sea en el propio servidor o en otros servidores en internet. Cuando se accede al contenido del propio servidor, estos enlaces toman el nombre de internos, de manera similar cuando el enlace conduce a recursos web fuera del servidor toman el nombre de enlaces externos.

Al momento de implementar enlaces dentro de un documento HTML hay que tener presente que tipo de recurso se va a mostrar cuando se activa el enlace, este contenido puede ser interno o externo.

Cuando se maneja contenido interno se definen dos tipos de URI's adicionales:

- **URI absoluto:** está formado por todos los elementos que componen el URI.



Figura 1.38 URI absoluto.

- **URI relativo:** está formado por la ruta del lugar donde se encuentra el documento, omitiendo algunas partes de la URI para hacerla más breve.

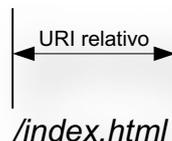


Figura 1.39 URI relativo.

La principal utilidad de URI's relativas es evitar que se tenga que indicar dentro del enlace toda la ruta para acceder a algún recurso web dentro del servidor. Por ejemplo, si dentro del servidor se han definido varios directorios y dentro de estos varias páginas web, resulta tedioso tener que ingresar toda la ruta de cada página, gracias a la ayuda de las URI's relativas este trabajo disminuye ya que solo se tiene que indicar las distintas rutas dentro del servidor y el navegador web se encargará de añadir el resto de elementos que componen el URI para dirigirse hacia el recurso solicitado.

Para determinar la ruta a ingresar en un enlace relativo se debe tener en cuenta como se hallan distribuidos los directorios dentro del servidor. Si una página se encuentra en el mismo directorio que la página de inicio solamente se debe ingresar el nombre del documento HTML. Por ejemplo, si se tiene la página de inicio *http://www.miweb.com/index.html* y se necesita mostrar la página *ejemplo1.html* que se encuentra en el mismo directorio (raíz /) solo se debe ingresar la URI relativa *ejemplo1.html*. Cuando se necesita ingresar a un nivel inferior es decir a un directorio dentro del raíz, solo es necesario ingresar la ruta correspondiente partiendo del directorio raíz. Por ejemplo, */masejemplos/ejemplo2.html*.

Para regresar desde un nivel inferior a un superior se emplea la secuencia *../* por ejemplo, si el documento HTML actual se encuentra en */masejemplos/mejores/ejemplo.html* y se quiere regresar al nivel superior */masejemplos/ejemplo2.html* la URI relativa correspondiente sería *../ejemplo2.html*. Se puede desplazar como máximo tres niveles ya que al aplicar

más se perdería el sentido de URI's relativas el cual busca reducir la escritura de la ruta completa. Para regresar desde un nivel muy lejano al raíz solo es necesario utilizar el símbolo / de esta manera se cargará la página inicial.

1.4.4.2 El elemento a <a>...

La etiqueta <a> permite crear enlaces dentro de una página web, para ello se debe utilizar un conjunto de atributos específicos que añaden a la etiqueta la funcionalidad que requiere. Los atributos específicos más importantes son:

- **name="texto"** permite nombrar al enlace para que pueda ser llamado por otro enlace (secciones de página).
- **href="URI"** es el más importante ya que es empleado para determinar el recurso hacia donde apunta el enlace, dentro de esta etiqueta pueden estar URI's relativas, absolutas, internas o externas.

Cuando una página es demasiado extensa puede resultar útil el poder desplazarse a distintas secciones de la página a través de un enlace, esto se logra con la ayuda del atributo específico *name* que permite crear enlaces vacíos. Estos enlaces se colocan en el lugar hacia donde se va a dirigir la página cuando un enlace de este tipo sea activado. Por ejemplo:

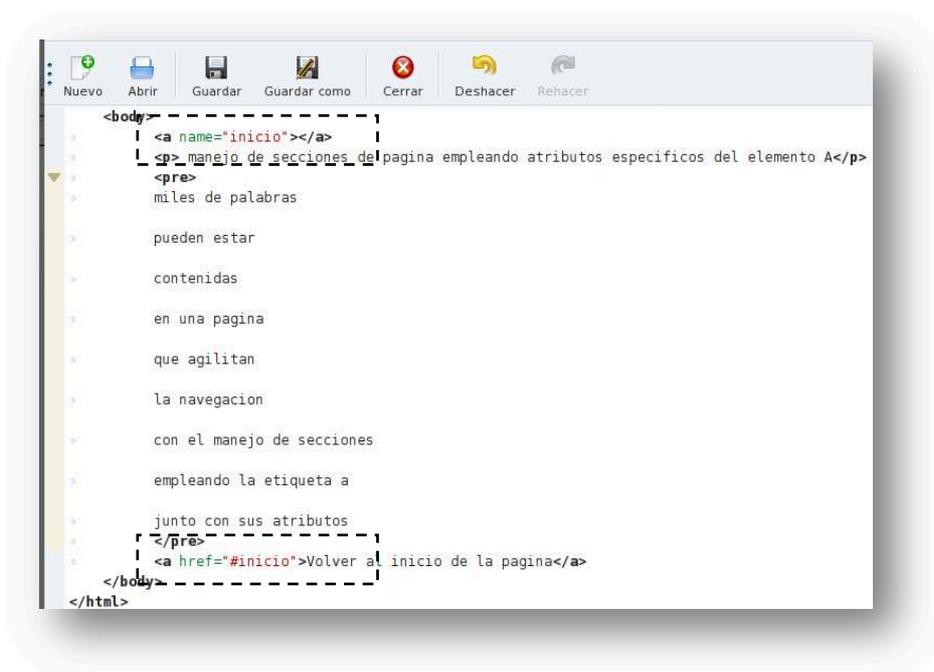
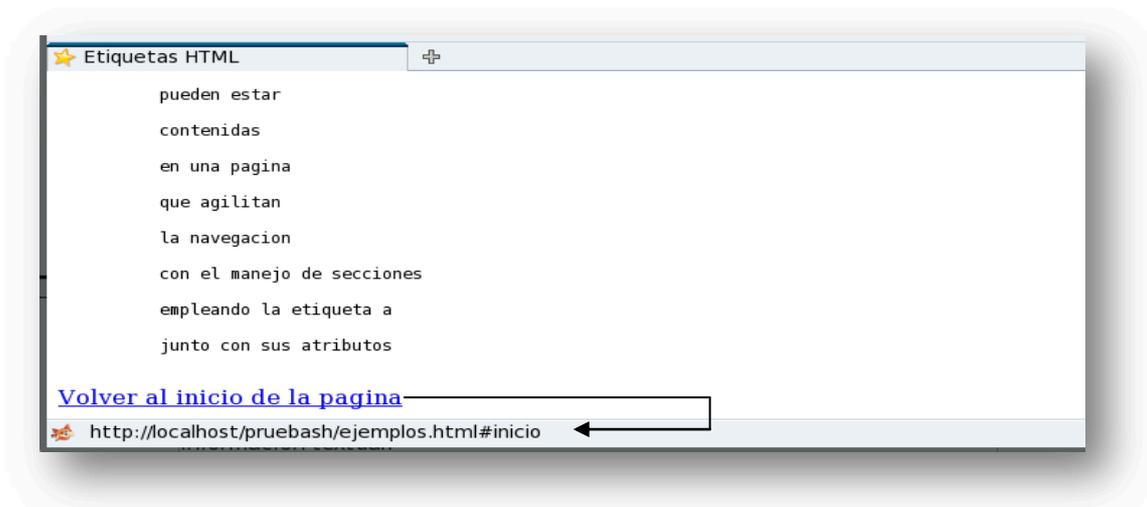
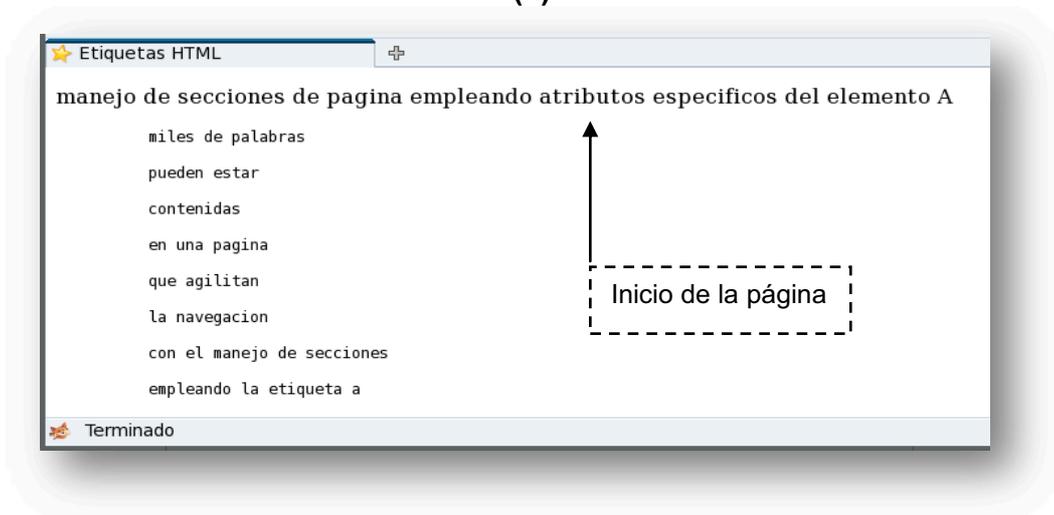


Figura 1.40 Inserción de secciones en un documento HTML.

La etiqueta `` corresponde al enlace vacío y determina el lugar de la página donde se va a dirigir el navegador cuando se active el enlace `Volver al inicio de la página`, hay que tomar en cuenta un elemento importante dentro de la etiqueta anterior, el signo #, este signo permite al enlace saber hacia dónde tiene que dirigirse al momento de ser activado. De esta manera se puede manejar distintas secciones dentro de una página que contenga gran cantidad de información textual.



(a)

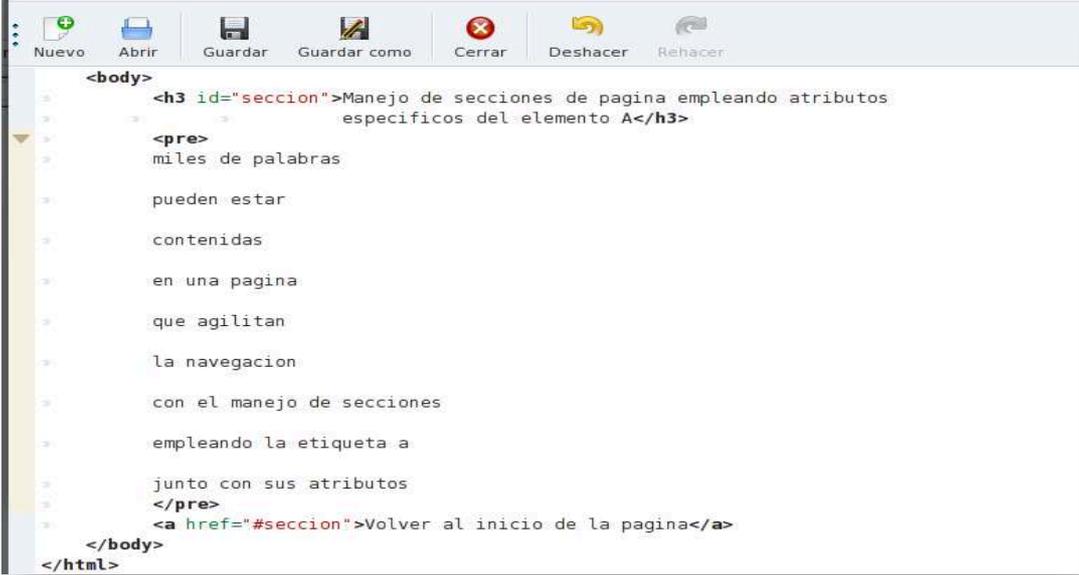


(b)

Figura 1.41 Desplazamiento en una página web empleando secciones.

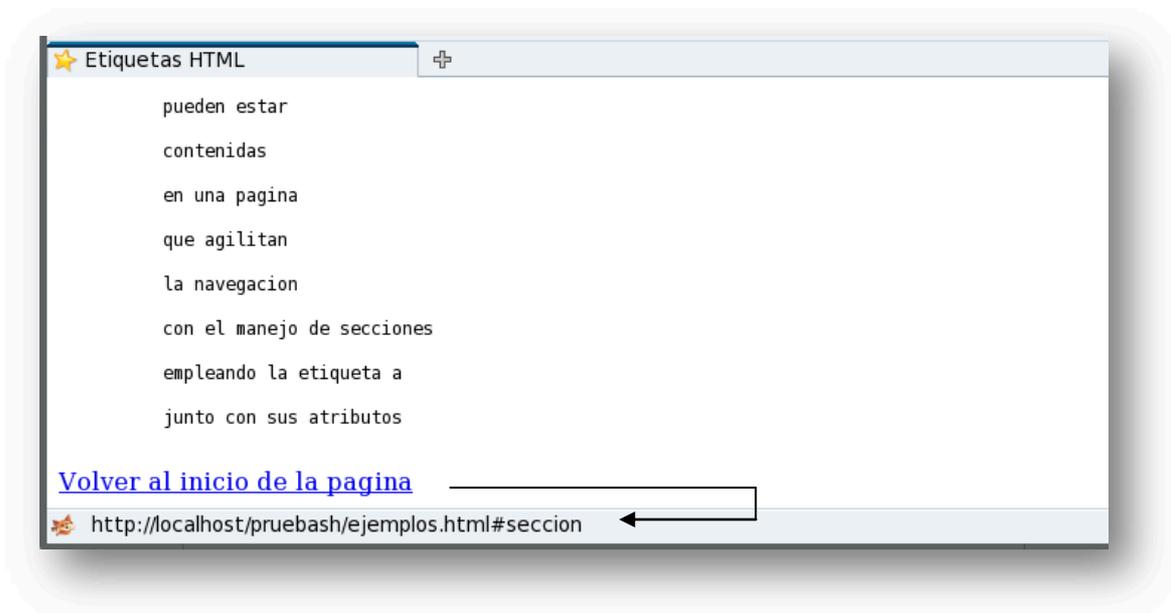
Si el puntero pasa sobre el enlace que se encuentra en la figura 1.41 (a), éste apunta hacia la sección correspondiente en la página web, al dar clic sobre el enlace el navegador web se dirige hacia el inicio de la página como se observa en la figura 1.41 (b).

También se puede emplear un atributo común para manejar secciones dentro de un documento web, se trata del atributo `<id>` y realiza la misma función que el atributo `name`. Por ejemplo:



```
<body>
  <h3 id="seccion">Manejo de secciones de pagina empleando atributos
    especificos del elemento A</h3>
  <pre>
    miles de palabras
    pueden estar
    contenidas
    en una pagina
    que agilitan
    la navegacion
    con el manejo de secciones
    empleando la etiqueta a
    junto con sus atributos
  </pre>
  <a href="#seccion">Volver al inicio de la pagina</a>
</body>
</html>
```

Figura 1.42 Inserción de secciones empleando el atributo ID.



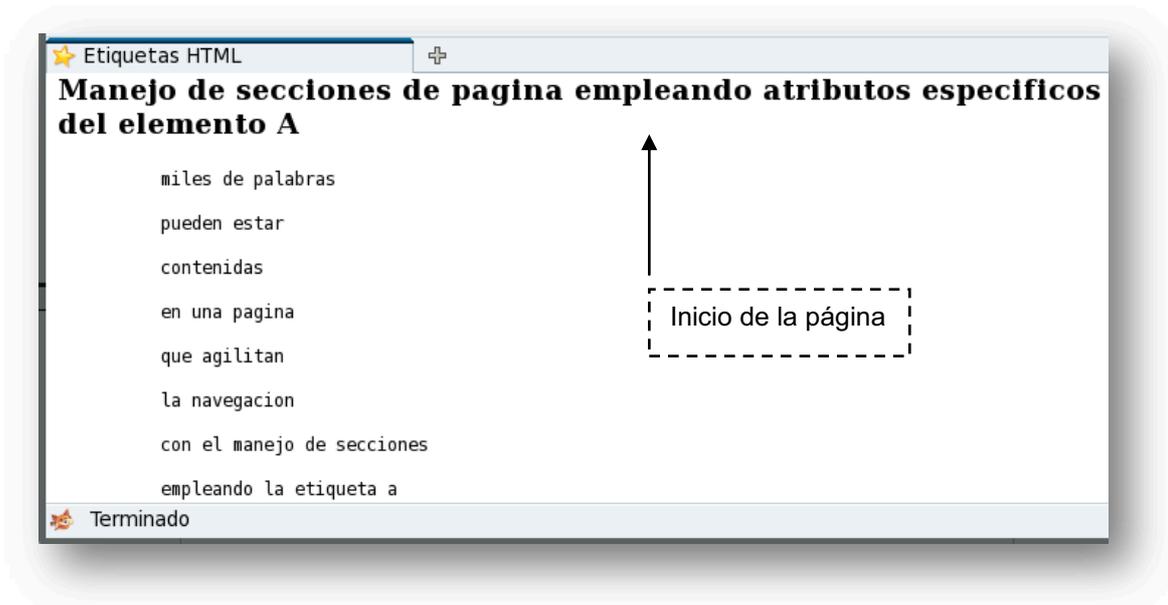


Figura 1.43 Desplazamiento en una página web empleando secciones.

Si se han definido distintas etiquetas con el atributo `<id>` dentro de un documento HTML y si estas pueden ser utilizadas para definir secciones dentro del documento, se aconseja que éstas sean utilizadas en lugar de crear enlaces vacíos con *name*.

1.4.4.3 Otros elementos de enlace

Cuando empleamos la etiqueta `<a>` para crear un enlace dentro de una página web, esta es identificada en la página, pero requiere la interacción con el cliente, ya que se debe realizar una acción que active el enlace. Si se quiere emplear hojas de estilo o emplear lenguajes como JavaScript estos deben ser llamados sin la necesidad de que el cliente los active. Para estos casos HTML define dos tipos de etiquetas:

- **Código ejecutable `<script>...</script>`**

La etiqueta `<script>` es empleada para ingresar ya sea un bloque de código o para invocar un archivo externo que contengan bloques de código, HTML mediante esta etiqueta puede ingresar varios lenguajes siendo el más utilizado JavaScript. No utiliza atributos comunes pero define atributos específicos, entre ellos están:

- **`src="url"`** se encarga de indicar la ruta donde se encuentra localizado el archivo externo que contiene el código a ser utilizado.

- **Type="tipo de contenido"** especifica el tipo de código que se va a incluir dentro de la etiqueta `<script>`, como se mencionó generalmente se emplea JavaScript.

Por ejemplo, las siguientes líneas muestran la manera de utilizar la etiqueta `<script>` para invocar un documento externo. Esta etiqueta puede aparecer en cualquier parte de un documento HTML, generalmente se la ingresa dentro de la cabecera del documento.



Figura 1.44 Ejemplo de inserción de JavaScript en un documento HTML.

- **Enlace de recursos etiqueta `<link>`**

Como su nombre lo indica la etiqueta `<link>` permite enlazar a la página recursos externos. Generalmente es empleada para invocar hojas de estilo CSS que requiere el documento HTML para aplicar un estilo a la página web. Puede utilizar atributos tanto comunes como específicos, dentro de éstos últimos se destacan:

- **rel="tipo de relación"** especifica la relación de la página web actual con el recurso solicitado.
- **type="tipo de contenido"** permite indicar al navegador del cliente el tipo de contenido que tendrá el recurso solicitado.
- **media="tipo de medio"** determina el tipo de medio al que se aplicará la relación entre el recurso y la página web.

La siguiente línea muestra la manera de emplear la etiqueta `<link>` para utilizar hojas de estilo CSS dentro de una página web. Se puede apreciar además que

constituye un elemento vacío pues no requiere información entre sus etiquetas, es similar a la etiqueta `
` que solo emplea una etiqueta tanto para apertura como cierre. A diferencia del elemento `<script>` esta solo puede ser escrita dentro de la cabecera del documento HTML.



Figura 1.45 Ejemplo de inserción de una hoja de estilo en un documento HTML.

1.4.5 IMÁGENES Y OBJETOS

La inserción de contenido multimedia dentro de una página web ha permitido que esta evolucione de tal manera que pase de ser una simple página de contenido textual, a ser una página con gran contenido informativo a través de medios como el sonido, el video, las animaciones, etc. que proporcionan un gran ambiente interactivo al usuario. Es por esta razón que HTML emplea un conjunto de etiquetas que permite la introducción de este tipo de contenido dentro de un documento web dinámico.

1.4.5.1 Imagen ``

La etiqueta `` es una de las más usadas dentro de un documento HTML ya que permite la inserción de imágenes en su contenido. Constituye una etiqueta vacía por lo que no contiene información en su interior. Al igual que la mayoría de etiquetas puede emplear cualquier atributo común y define varios atributos específicos:

- **`src="URI"`** especifica el URI donde se encuentra la imagen que se quiere visualizar.
- **`alt="texto"`** proporciona una descripción textual acerca de la imagen.
- **`name="texto"`** da un nombre al elemento imagen.

- ***height="medida"*** especifica la altura que empleará la imagen para mostrarse en una página web.
- ***width="medida"*** especifica la anchura que empleará la imagen para mostrarse en una página web.

Los atributos *src* y *alt* son indispensables dentro de la etiqueta ``. El primer atributo se encarga de determinar el lugar donde se encuentra almacenada la imagen mediante su URI, al igual que las etiquetas de enlaces puede emplear URI's relativos o absolutos. El segundo atributo proporciona información textual acerca de la imagen, su tamaño se halla restringido a un número menor a 1024 caracteres.

El tipo de imágenes que puede soportar HTML es variado ya que no se describe un tipo estricto de formato. Se recomienda que sea un formato común o aquel de mayor utilización pues de esta manera se asegura que los navegadores puedan interpretar correctamente la imagen. Los formatos PNG, JPG y GIF, son soportados por todos los navegadores web actuales.

Cuando se requiere modificar una imagen en relación a su tamaño original se emplean los atributos *height* y *width*. El primer atributo define la altura, mientras que el segundo define su ancho. El ingreso de la unidad de medida para estos atributos se la realiza de dos maneras; mediante números enteros, para lo cual el navegador interpreta que su medida está dada en pixeles o mediante porcentajes, cuando se emplea porcentajes el navegador lo interpreta haciendo referencia al elemento que contiene la imagen. Si la imagen no está contenida en algún elemento la relación de porcentajes la realiza respecto a la altura y el ancho de la página web.



Figura 1.46 Ingreso de una imagen a un documento HTML.

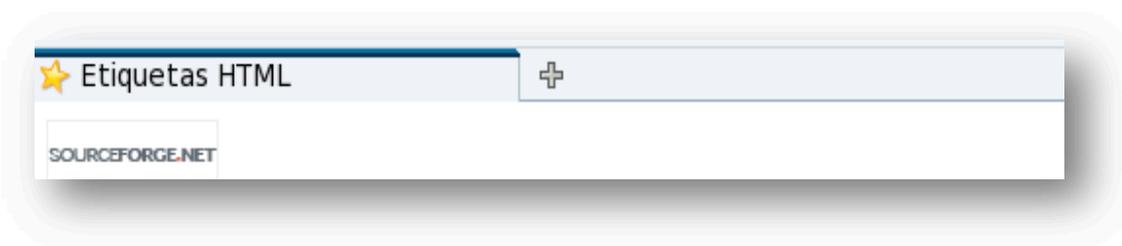


Figura 1.47 Visualización de la imagen insertada.

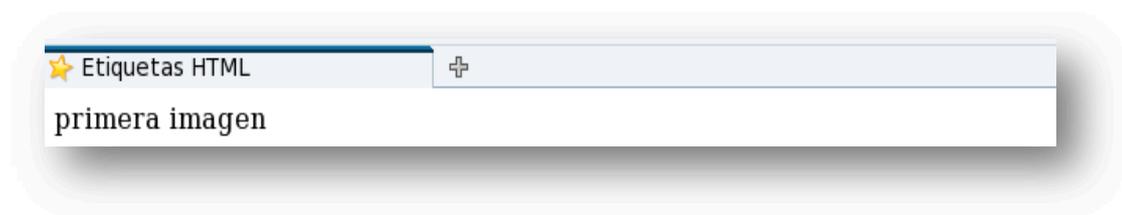


Figura 1.48 Visualización del texto alternativo a una imagen.

1.4.6 TABLAS

Un elemento que al igual que las imágenes provee un método para proporcionar información al lenguaje HTML lo constituyen las tablas. Las tablas en HTML son muy flexibles y completas, permiten añadir títulos, generar columnas y filas para formar celdas. La información dentro de una celda puede ser textual, visual o por medio de enlaces a otras páginas, no existe restricción en cuanto al contenido de las celdas pero éstas deben ser exclusivamente usadas para ingresar información ya que en años anteriores su utilización también fue enfocada a la decoración de páginas web.

- **Generación de tablas <table>...</table>**

El elemento que contendrá todas las celdas necesarias para proporcionar información lo constituye una tabla. Una tabla permitirá la generación de filas y columnas de acuerdo a la distribución que se quiera desarrollar. Esta etiqueta puede emplear tanto atributos comunes como específicos, de entre éstos últimos se destacan:

- **summary="texto"** define un resumen acerca de la información contenida en la tabla, es utilizado por navegadores que proporcionan una salida por voz.
- **border="pixels"** define el borde que rodea a la tabla.
- **cellspacing="longitud"** define el espacio entre las celdas.
- **cellpadding="longitud"** define el espacio dentro de la celda.

- **Título <caption>...</caption>**

Por lo general las tablas requieren de un título que mencione acerca de que información se va a proporcionar en la tabla, para ello se emplea la etiqueta <caption>, esta etiqueta debe ser colocada después de la etiqueta inicial de <table> y solo puede ser ingresada una por cada tabla que se genere. Puede emplear cualquier atributo común y no define atributos específicos.

- **Creación de filas <tr>...</tr>**

La forma en que HTML genera tablas en un documento web sigue un orden específico, primero se define la tabla, luego se deben definir las filas y dentro de estas se definen las distintas celdas que contendrán la información. Para crear una fila dentro de una tabla se emplea la etiqueta <tr> la cual ocupará todo el espacio dentro de la tabla ya que constituye un elemento de bloque. Al igual que el resto de etiquetas puede emplear atributos comunes y no define atributos específicos.

- **Creación de celdas <td>...</td>**

Si ya se crearon las filas es el momento de crear las columnas. La unión de estos dos elementos permiten crear una celda la cual se encargará de contener la

información, para generar una celda se emplea la etiqueta `<td>` que puede emplear atributos comunes y define varios atributos específicos, entre ellos:

- **`abbr="texto"`** se emplea para definir información abreviada acerca del contenido de la celda.
- **`colspan="numero"`** se utiliza para determinar el número de columnas que ocupará la celda.
- **`rowspan="numero"`** es utilizada para definir el número de filas que ocupará la celda.

- **Encabezados `<th>...</th>`**

Cuando se necesita generar encabezados ya sea en una fila o en una columna se emplea la etiqueta `<th>` que se encarga de dar énfasis a la información que se encuentra dentro de esta etiqueta, transformando a la celda en una cabecera ya sea de una columna o de una fila, de manera más clara da un aspecto de negrita al texto que se encuentra dentro de la etiqueta. Define los mismos atributos que la etiqueta `<td>` tanto para los atributos comunes como para los específicos haciendo uso de un atributo en especial:

- **`Scope="valor"`** se encarga de especificar el conjunto de celdas para las cuales la celda de encabezado actual será su cabecera. Los valores que puede utilizar este atributo son: `row`(fila), `col`(columna), `rowgroup`(grupo de filas), `colgroup`(grupo de columnas).

```

<html>
  <head>
    <title>Etiquetas HTML</title>
    <style type="text/css">
      td,th{
        text-align:center;border-color: #000000; border-style:solid
      }
    </style>
  </head>
  <body>
    <table style="border-color: #ffccff; border="10">
      <caption>Creaci&ocirc;n de tablas en HTML</caption>
      <tr>
        <th>Primera columna</th>
        <th>Segunda columna</th>
        <th>Tercera columna</th>
      </tr>
      <tr>
        <td>celda 1.1</td>
        <td>celda 1.2</td>
        <td>celda 1.3</td>
      </tr>
      <tr>
        <td>celda 2.1</td>
        <td>celda 2.2</td>
        <td>celda 2.3</td>
      </tr>
    </table>
  </body>
</html>

```



Generaciôn de celdas como encabezados o títulos.



Generaciôn de celdas de datos.



Generaciôn de filas.



Título de tabla.

Figura 1.49 Implementaciôn de una tabla en un documento HTML.

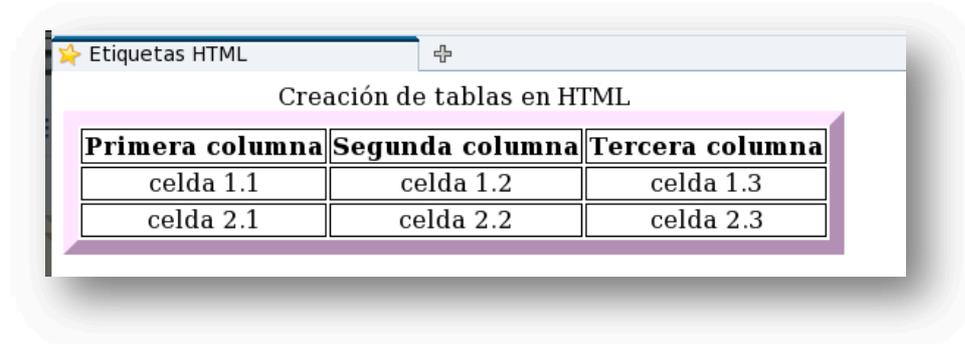


Figura 1.50 Visualización de la tabla creada.

Existe un número adicional de etiquetas para el manejo de columnas que permiten por ejemplo, generar un tipo de formato para un grupo de columnas o simplemente para una columna. Estas etiquetas son `<col/>` para el caso de afectar a una sola columna o `<colgroup>` para afectar a un conjunto de columnas. Estas etiquetas definen tanto atributos comunes como el resto de etiquetas, así como atributos específicos para dar un formato al contenido de las celdas ya sea en el caso de una columna o en un grupo de columnas. Estas etiquetas son empleadas para el desarrollo de tablas complejas que requieren una norma de visualización, por ejemplo, para generar tablas en el área de contabilidad, es por esta razón que este tipo de etiquetas está fuera del alcance del presente proyecto por lo que no se realiza un estudio profundo acerca de este tipo de etiquetas.

1.4.7 HOJAS DE ESTILO

El objetivo inicial de HTML se enfocó en proporcionar información a sus clientes mediante un navegador web, dejando a un lado el aspecto visual con el que se mostraría la página web, al transcurrir el tiempo el espectro de alcance de internet abarcó cientos de personas de distintas procedencias, lo que permitió que todo el mundo tenga acceso mediante su navegador web a páginas con amplio contenido informativo pero carentes de contenido visual que provoque una atracción al cliente. Es por esta razón que se comenzó a dar diseño a las páginas web, primero se utilizó los atributos disponibles en las etiquetas HTML para proporcionar por ejemplo, color al texto, color de fondo a las tablas, color de fondo al cuerpo del documento o emplear tablas para estructurar el contenido de la página web. Al emplear estos atributos y etiquetas se aumentó la complejidad del desarrollo de documentos HTML por esta y más razones se desarrolló lo que se

conoce como hojas de estilo o CSS¹⁹ que permiten separar el contenido informativo de la parte gráfica en una página web.

1.4.7.1 Añadir estilos a HTML

Existen varias maneras de implementar un estilo dentro de un documento HTML, puede emplearse el atributo *style* dentro de una etiqueta para dar un formato a su contenido, o puede ser declarado dentro de la cabecera del documento HTML empleando el elemento `<style>` para formar una hoja de estilo que luego puede ser empleada en cualquier lugar del documento html. Por ejemplo:



Figura 1.51 Estilo para una tabla empleando el atributo style.

Como se observa en la figura 1.51 el atributo style permite generar una tabla con un borde de tamaño 10 cuyo color es #ffccff (valor hexadecimal del color violeta).

¹⁹ **Cascading Style Sheets (CSS).**- Hojas de Estilo en Cascada, lenguaje empleado para la creación de hojas de estilo con el fin de separar el aspecto visual del contenido presente en una página web.

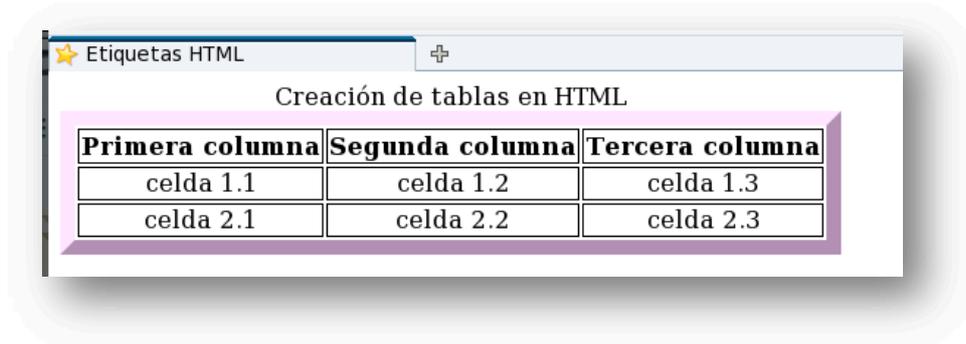


Figura 1.52 Estilo aplicado.

Los valores para el atributo `style` se definen como *“nombre:valor”* si existiera más de un valor para el atributo, cada uno se colocaría a continuación separado por un punto y coma *“border-width:1px ; border:solid”*.

- **El elemento `<style>...</style>`**

El elemento `<style>` es empleado para dar características de estilo a varias etiquetas dentro de un documento HTML, para ello se genera un conjunto de reglas de estilo dentro del elemento `<style>` y se lo coloca dentro de la cabecera del documento. Define los siguientes atributos específicos:

- **`type="tipo de contenido"`** especifica el lenguaje de las hojas de estilo a ser empleado para el documento, por ejemplo (*“text/css”*).
- **`media="tipo de medio"`** especifica el medio donde se va a aplicar el estilo.
- **`title="texto"`** proporciona información consultiva acerca del elemento `<style>`.

Cuando se emplea el elemento `<style>` dentro de la cabecera del documento HTML, se pueden generar reglas de estilo para los siguientes casos:

- Para dar estilo a un elemento que aparece varias veces en el documento HTML. por ejemplo:

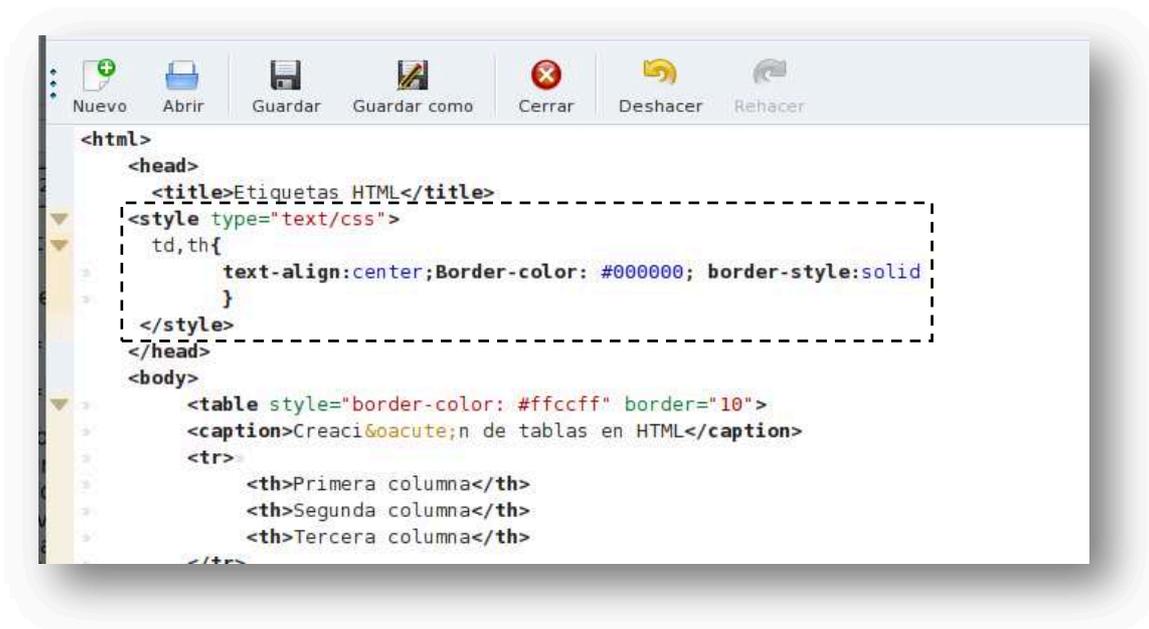


Figura 1.53 Estilo a un elemento que aparece varias veces.

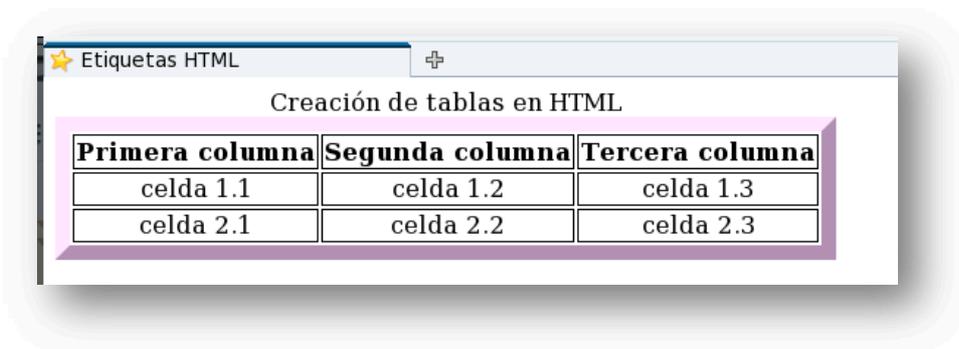


Figura 1.54 Estilo aplicado a todas las celdas.

- Para dar estilo a elementos que pertenezcan a una clase específica, para ello el elemento empleará el atributo class como referencia para la aplicación del estilo. La declaración de una clase dentro de `<style>` se la realiza indicando el elemento al que se aplicará el estilo seguido de un punto y a continuación el valor del atributo class.

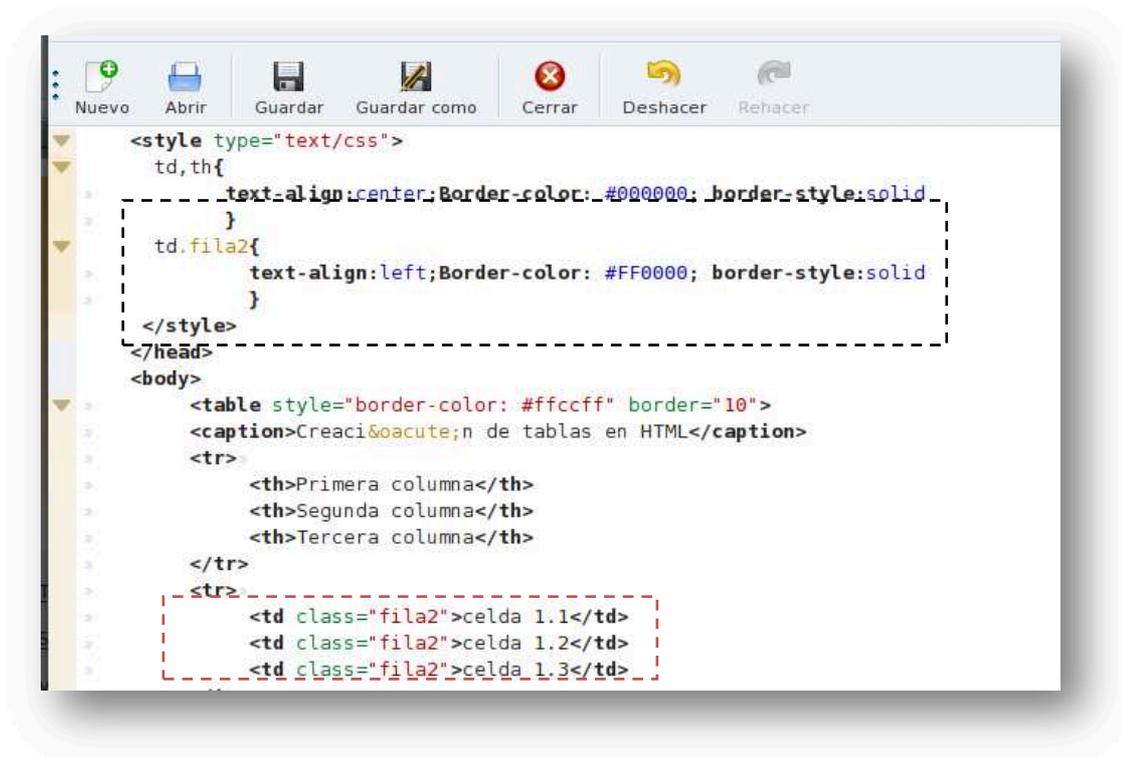


Figura 1.55 Estilo empleando el atributo class.

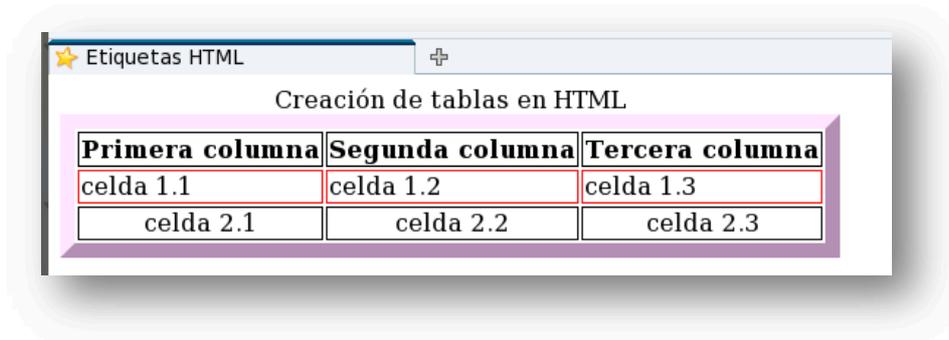


Figura 1.56 Estilo aplicado a la segunda fila.

- Para dar estilo a un elemento que emplea el atributo id, con ello se aplicará el estilo para una sola aparición del elemento. La declaración dentro del elemento `<style>` se la realiza empleando el signo numeral seguido del valor del atributo id.

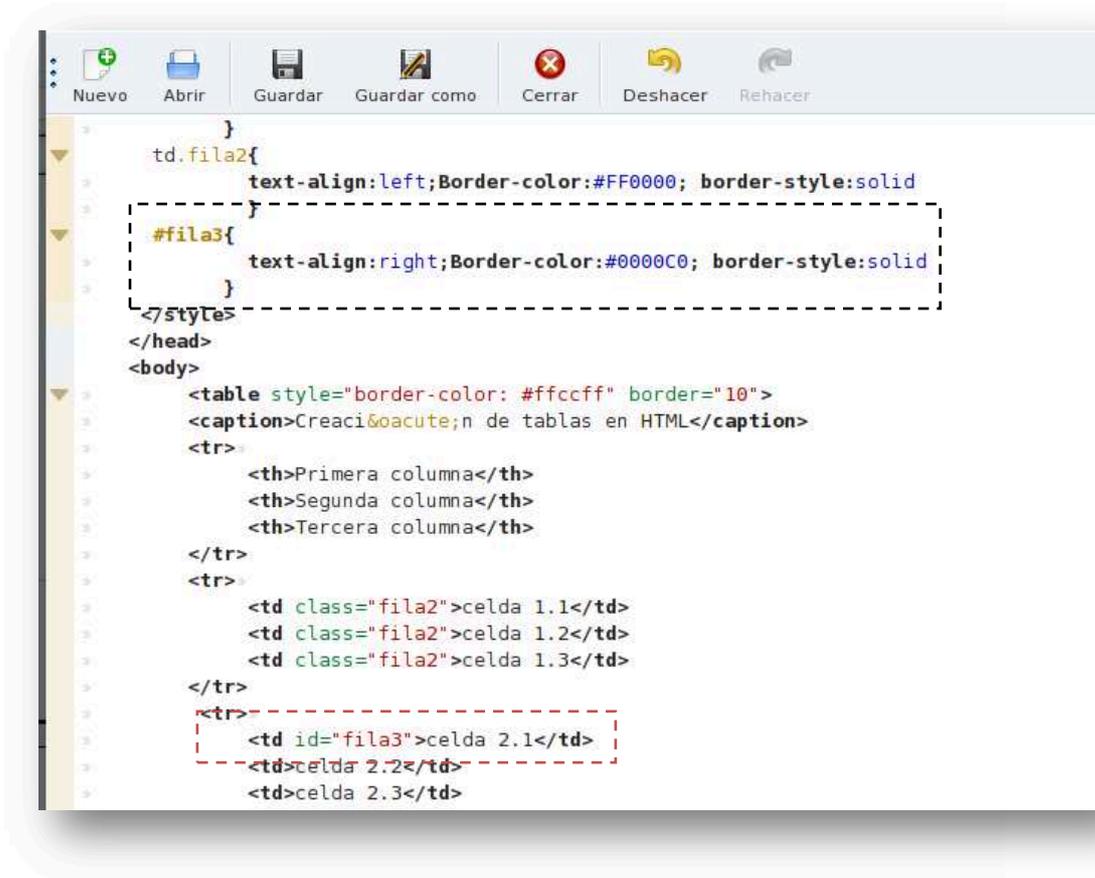


Figura 1.57 Estilo empleando el atributo id.

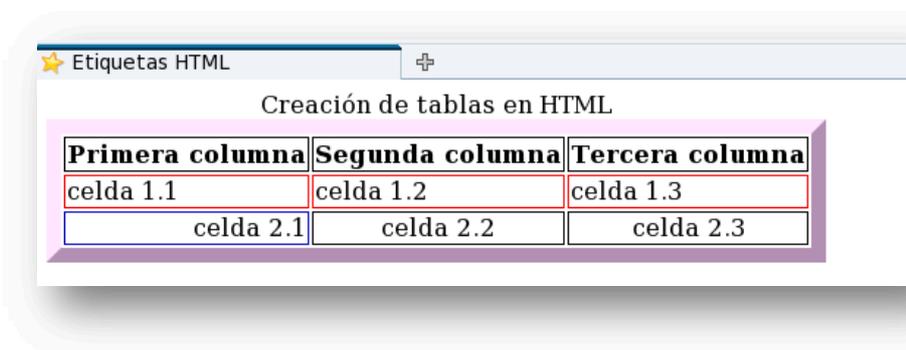


Figura 1.58 Estilo aplicado a la primera celda de la tercera fila.

Existen dos etiquetas adicionales que permiten agregar estilo a elementos en línea o elementos en bloque. Estas etiquetas son útiles cuando se requiere aplicar un estilo a un contenido específico para el caso de elementos en línea o a un conjunto de elementos para el caso de bloque. Para elementos en línea se emplea el atributo `...` que al igual que al resto de etiquetas se le puede aplicar estilos mediante los métodos que se mencionó en párrafos

anteriores. Para el caso de elementos en bloque se emplea el atributo `<div>...</div>` y al igual que `span` puede aplicar diferentes estilos. Por ejemplo:

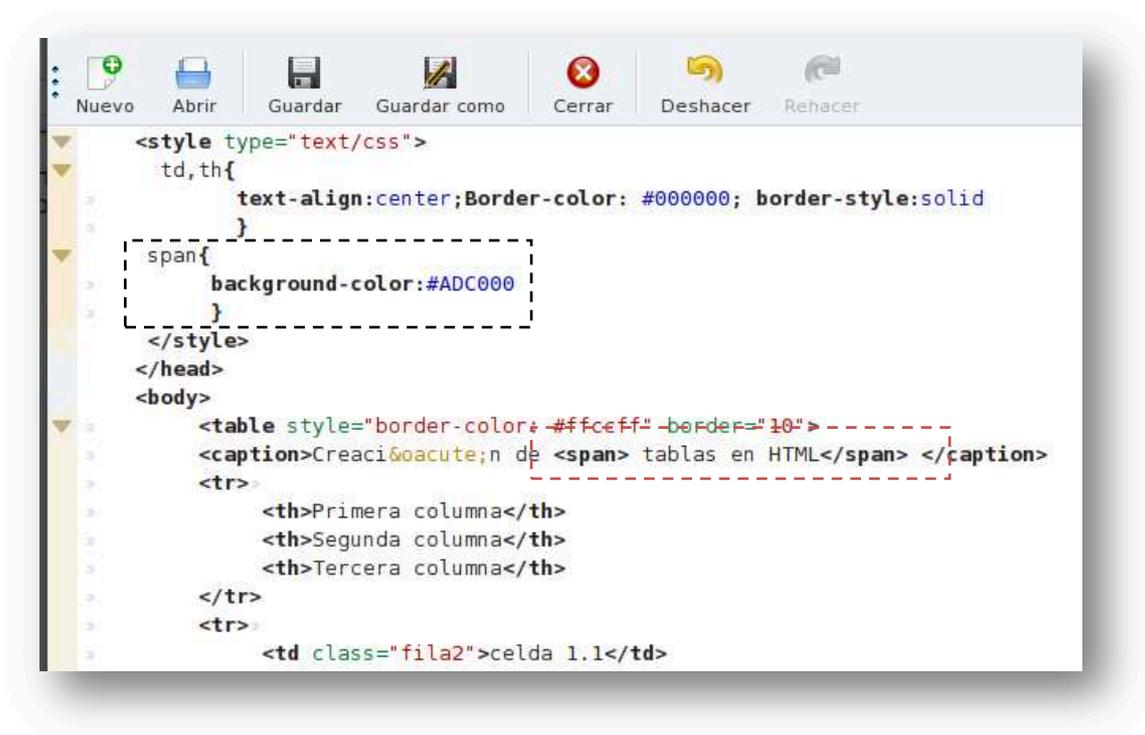


Figura 1.59 Estilo para un elemento en línea.

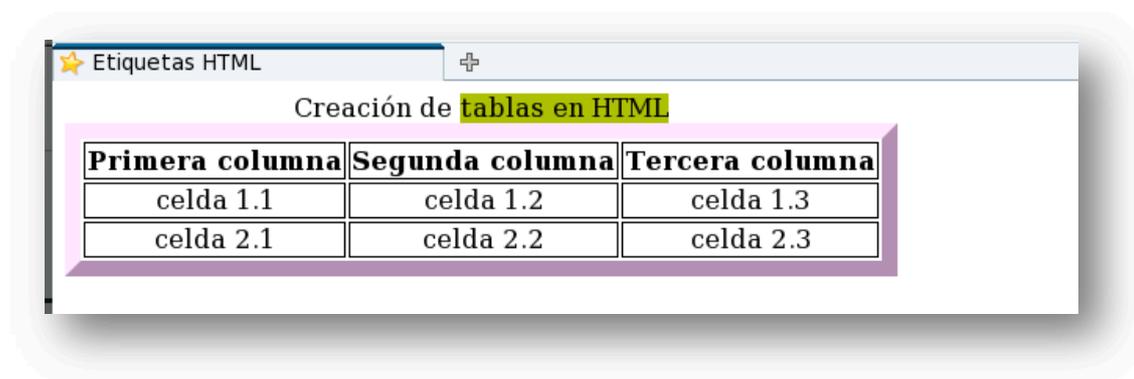


Figura 1.60 Estilo aplicado al título de la tabla.

Como se aprecia el objetivo del elemento `span` es permitir aplicar un estilo a una parte del texto que se encuentra dentro del elemento `<caption>`. Para el caso de elementos en bloque el elemento `div` permite aplicar un estilo a varios elementos:

```

td,th{
    text-align:center;Border-color: #000000; border-style:solid
}
div.par{
    text-align: justify
}
</style>
</head>
<body>
<div align="center">
<table style="border-color: #ffccff" border="10">
<caption>Creaci&oacute;n de <span> tablas en HTML</span> </caption>
<tr>
<th>Primera columna</th>
<th>Segunda columna</th>
<th>Tercera columna</th>
</tr>
<tr>
<td>celda 1.1</td>
<td>celda 1.2</td>
<td>celda 1.3</td>
</tr>
</table>
</div>
<div class="par">
<p>este es un ejemplo de aplicaci&oacute;n de estilos para elementos en bloque
<p>el atributo div y span emplean las mismas reglas para aplicar estilos. Emple
</div>

```

Figura 1.61 Estilo para un elemento en bloque.

Creación de tablas en HTML

Primera columna	Segunda columna	Tercera columna
celda 1.1	celda 1.2	celda 1.3

este es un ejemplo de aplicación de estilos para elementos en bloque empleando la etiqueta div al igual que las etiquetas anteriores permite agregar estilos al documento.

el atributo div y span emplean las mismas reglas para aplicar estilos. Empleando ya sea: clases, el atributo id o mediante la declaración de estilos mediante el atributo style.

Figura 1.62 Estilo en bloque aplicado a dos párrafos.

1.4.7.2 Hojas de estilo externa

HTML permite generar hojas de estilo externas que pueden ser invocadas por un documento HTML, gracias a esto se puede separar el código HTML del código de

diseño y se facilita la modificación de manera independiente de cada uno de estos códigos.

Para crear una hoja de estilo externa solo se requiere un editor de texto simple donde se ingresan las diferentes reglas de estilo en base a los criterios que se mencionaron en párrafos anteriores, este documento de texto se lo almacena con la extensión *css* y estará listo para ser solicitado por un documento HTML.

Ahora en el documento HTML se debe ingresar un enlace que le permita obtener el estilo a ser aplicado, para esto se emplea el elemento **<link>** que se coloca dentro de la cabecera y cuyos atributos tendrán los siguientes valores:

- **href** → indicará la localización de la hoja de estilo mediante el URI.
- **type** → especificará el tipo de lenguaje para la hoja de estilo.
- **rel** → indicará si la hoja de estilo es única o existen alternativas.

Por ejemplo:

```
<LINK href="estilo.css" rel="stylesheet" type="text/css">
```

El URI para esta hoja de estilo corresponde a *estilo.css*, el atributo *rel* indica que se trata de una sola hoja de estilo *stylesheet* y el atributo *type* indica que el tipo de lenguaje es *text/css*. Para el caso que se emplee una hoja de estilo alternativa el atributo *rel* tendrá el valor *alternate stylesheet*.

En un documento HTML se puede combinar el uso de estilos ya sea con estilos en línea mediante el atributo *style*, hojas de estilo externas mediante *link* o con el elemento **<style>** dentro de la cabecera del documento. La decisión acerca de que estilos se van a emplear depende del desarrollador de la página web. A continuación se muestra un ejemplo sencillo acerca de la utilización de hojas de estilo externas.



Figura 1.63 Archivo de estilo externo con extensión CSS.



Figura 1.64 Referencia hacia el archivo CSS externo.

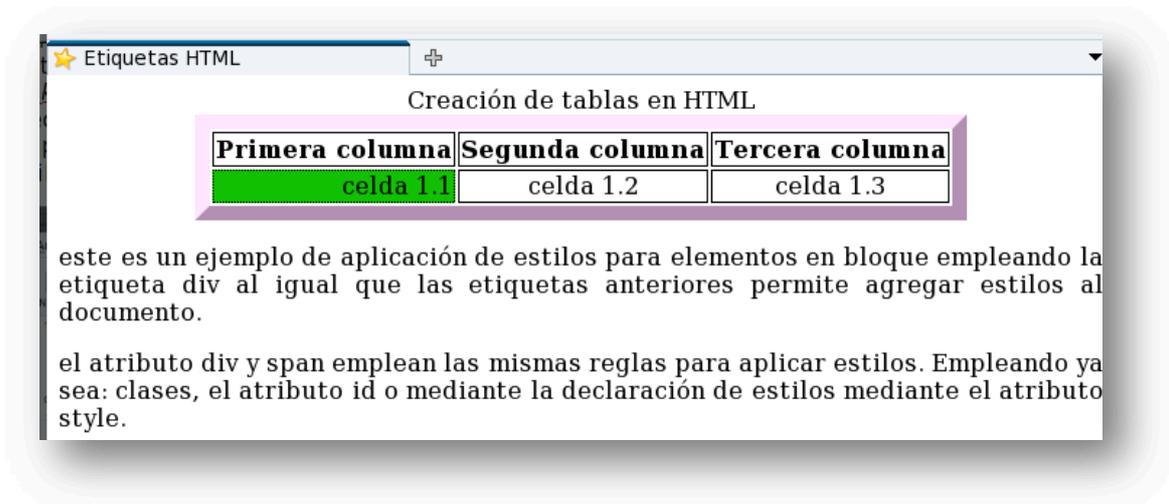


Figura 1.65 Aplicación del estilo mediante una hoja de estilo externa.

1.5 TECNOLOGÍAS PARA PÁGINAS WEB DINÁMICAS DEL LADO DEL SERVIDOR

Como su nombre lo indica son programas o scripts que se ejecutan en el lado del servidor y que facilitan la comunicación entre una aplicación que reside en el servidor y el navegador web del cliente. El procedimiento de interacción que se realiza con el cliente es el siguiente:

- El usuario envía los datos o requerimientos usando para ello un navegador web.
- El servidor, mediante el uso de un programa desarrollado en cualquier tipo de tecnología, ejecuta las instrucciones enviadas por el cliente.
- El servidor procesa dicha información y la envía. Los resultados arrojados por el programa podrán ser visualizados por el usuario a través de una página web en su navegador.

Actualmente existen algunas tecnologías web que se ejecutan en el lado del servidor, para el efecto se han considerado las más utilizadas:

1.5.1 ASP (Active Server Pages o Páginas de Servidor Activo)

Esta tecnología es propia de Microsoft. Son programas que permiten generar en el servidor (Internet Information Server (IIS)), los resultados de la petición generada por el cliente a través de un navegador web. Emplea Jscript, VBScript o PerlScript como lenguajes de programación. Una de sus aplicaciones es la gestión de bases de datos. Una de las desventajas de utilizar ASP, es que se requiere escribir varias líneas de código para realizar tareas sencillas, como por ejemplo crear o borrar archivos, además es una tecnología propietaria. La extensión de estos archivos es *.asp*.

1.5.2 CGI (Common Gateway Interface o Interfaz de Acceso Común)

Conocido también como pasarelas o CGI scripts, son programas que contienen una serie de instrucciones, generalmente escritas en lenguajes de programación como Perl o C, que permiten procesar en el servidor web, las peticiones

realizadas por un cliente mediante un navegador web. Dentro de las aplicaciones de los CGI's están motores de búsqueda, permiten el envío de datos de formularios, mapas de imágenes, chat, juegos en línea, gestión de bases de datos. Una desventaja de utilizar CGI es que éste crea tantos procesos como peticiones de clientes recibidas, lo que hace que por cada petición se consuma tiempo y memoria. La extensión de estos archivos es *.cgi*.

1.5.3 JSP (Java Server Pages o Página de Servidor Java)

Es una tecnología de Sun Microsystems, son programas que se ejecutan en el servidor web y cuyo resultado puede ser visualizado por el cliente a través de una navegador web. Emplea Java como lenguaje de programación, aunque las nuevas versiones hacen uso de Java Beans. Una de las aplicaciones más sobresalientes de JSP es la gestión de bases de datos. Una de las desventajas que presenta JSP es la complejidad de aprendizaje. La extensión de estos archivos es *.jsp*.

1.5.4 WEB SERVICES

Es una tecnología diseñada para permitir la comunicación entre aplicaciones en una red, mediante protocolos como HTTP, independientemente del lenguaje de programación o el sistema operativo que se utilice. La comunicación entre el servidor y el cliente, se basa en el empleo de mensajes XML, de ésta manera, la interfaz de los web services estará definida en función de los mensajes que se envía y recibe.

1.5.5 SELECCIÓN DE LA TECNOLOGÍA WEB DEL LADO DEL SERVIDOR

Para el presente proyecto se ha optado por el uso de CGI como tecnología web del lado del servidor, debido a que ésta tecnología nos permitirá generar directamente contenido web dinámico, ya que tiene como característica integrar un lenguaje de etiquetado como HTML y un lenguaje de programación como por ejemplo Perl o C en un mismo archivo, de ésta manera se facilitará el acceso a una serie de servicios y aplicaciones del lado del servidor y permitirá al cliente visualizar dicha información a través de cualquier navegador web.

Si bien actualmente existen tecnologías como JSP, ASP o Web Services, cuyo fuerte es la gestión de bases de datos, el uso de CGI's sigue siendo la forma más

práctica y sencilla de realizar tareas de administración en el servidor mediante el envío y recepción de datos entre el navegador web y programas del lado del servidor, así como también ejecutar comandos propios del sistema operativo Linux, que son las tareas que comúnmente se realizarán en éste proyecto. Cabe aclarar que si bien se definió en párrafos anteriores que el nivel de concurrencia es una desventaja del uso de CGI's, éste no será un aspecto crítico para el desarrollo del presente proyecto, debido a que está orientado a un único usuario que es el webmaster o administrador de la red, por lo que las únicas peticiones que el servidor tendrá que atender serán de éste usuario.

1.6 CGI COMO TECNOLOGÍA PARA PÁGINAS WEB DINÁMICAS DEL LADO DEL SERVIDOR

CGI son las siglas de Common Gateway Interface, o Interfaz de Acceso Común. Es una tecnología que permite la interconexión de aplicaciones externas con servidores de información, como por ejemplo los servidores Web. Es decir que va a realizar la función de interfaz o pasarela entre el servidor Web y los programas, denominados programas CGI, mediante el uso del protocolo HTTP y el lenguaje HTML.

Un ejemplo claro del uso de CGI's, puede ser el acceso por Internet a una base de datos de una empresa u organización desde la Web. En este caso es preciso crear un programa CGI, el cual se ejecute en un servidor web, que permita transmitir información al motor de la base de datos, recibir los resultados y presentarlos al cliente. Se debe tener en cuenta que un programa CGI no debe invertir demasiado tiempo en sus procesos, ya que no permitirá que el navegador web consiga concluir su tarea eficientemente.

1.6.1 NORMA CGI

La Interfaz de Acceso Común (CGI) es un estándar ampliamente utilizado en la construcción de documentos Web dinámicos. Fue desarrollada por el Centro Nacional de Supercómputo (NCSA) para emplearse en su servidor Web. La norma CGI se refiere a la interacción entre un servidor y un programa CGI. La elección del lenguaje de programación dependerá del tipo de documento y si permite o no la generación de CGI's, por ejemplo para documentos con intenso

cálculo aritmético se puede optar por un lenguaje como C o C++ y para documentos que solo requieran un poco de formato se puede elegir un lenguaje interpretado como por ejemplo Perl.

1.6.2 ESPECIFICACIONES DE LA INTERFAZ CGI

Los programas CGI son ejecutables, usarlos es equivalente a dejar a los clientes ejecutar un programa en el servidor, lo cual implica un problema de seguridad. Por lo tanto se deben establecer ciertas medidas de seguridad cuando se desee implementarlos. Los programas CGI necesitan residir en un directorio especial, generalmente es el directorio cgi-bin, dependerá del sistema operativo en el que se encuentre el servidor web para establecer su ubicación, de ésta manera el servidor Web sabrá que este directorio contiene programas que deberán ser ejecutados y su salida deberá ser enviada al navegador web del cliente, dicho directorio está controlado por el webmaster o administrado de la red, de esta manera se impide que un usuario común pueda crear programas CGI.

Los programas CGI pueden escribirse en cualquier lenguaje de programación que sepa manejar tanto la entrada estándar²⁰ como la salida estándar²¹, además que pueda ser ejecutado en el sistema, como:

- Visual Basic.
- AppleScript.
- PERL.
- C/C++.
- Un Shell de mandatos de UNIX.

La selección del lenguaje de programación dependerá un poco de sobre qué sistema operativo está el servidor, del tipo de CGI o API soportados por el mismo

²⁰ **Entrada estándar.**- Se refiere a esta entrada como el archivo STDIN, es el lugar desde el cual un programa pretende leer su información de entrada, por defecto los procesos emplean como STDIN el teclado.

²¹ **Salida estándar.**- Se refiere a esta salida como el archivo STDOUT, es el lugar donde un programa pretende escribir su salida, por defecto se emplea la pantalla o una ventana del terminal como STDOUT.

y del tipo de aplicación que se desee programar, es decir puede tratarse de una aplicación web o una local.

Si seleccionamos un lenguaje de programación como C se tendrá que compilarlo y poner el ejecutable en el directorio destinado a los CGI's. Por otro lado, si seleccionamos un lenguaje interpretado como Perl, o el shell de UNIX, no se tendrá necesidad de ésto, ya que simplemente se requerirá que en el directorio cgi-bin residan secuencias de tipo script, a las que no se asocian ningún código fuente. Los programas CGI son más fáciles de depurar, modificar y mantener que un programa compilado típico. Lo más recomendable es usar un lenguaje lo más portable posible, ya que el cambio de sistema operativo afectaría mínimamente al programa hecho y no se tendrán que realizar muchos cambios.

1.6.3 SALIDA DE UN PROGRAMA CGI

La forma en la que se invoca a un CGI es mediante un URI que apunte al fichero ejecutable, el cual se encuentra generalmente en el directorio cgi-bin. El servidor HTTP al recibir una petición dentro de este directorio, comprenderá que se trata de un programa y procederá a ejecutarlo. Al ejecutarse un CGI su salida estándar queda inmediatamente conectada con el servidor, que le hará llegar al cliente la respuesta a la petición realizada.

La salida de un programa CGI no se restringe al HTML, ya que la norma permite que las aplicaciones CGI generen tipos arbitrarios de documentos, como por ejemplo texto normal, imágenes, animaciones, etc. Para distinguir entre los distintos documentos, la norma define una serie de cabeceras que permite a los programas CGI describir el tipo de documento. Aunque la salida de un programa CGI es un flujo de datos, se puede considerar dividida en cabecera y cuerpo.

- **Cabecera**

Las cabeceras generadas por los programas CGI, emplean un formato similar a las cabeceras empleadas por un servidor Web cuando envía un documento a un navegador web. La cabecera contiene información que el servidor utiliza para construir la cabecera HTTP de su respuesta al cliente. Una cabecera consta de una o más líneas de texto más una línea en blanco que se deja entre las

cabeceras CGI y el contenido de la información, si se omite esta línea el sistema no funcionará y el cliente no podrá acceder a la información. Cada una de estas líneas especifica información del documento y la representación de los datos. Generalmente pueden aparecer tres líneas: Content-type, Location y Status.

- **Content-type:** Indica el tipo de datos que generará el programa CGI. La sintaxis empleada para indicar el tipo de datos es la sintaxis MIME:

Content-type:type/subtype

En la siguiente tabla se describen los diferentes tipos de datos que se usan en la programación CGI:

type/subtype	Descripción
audio/basic	Este tipo abarca todos los formatos de sonido .au y .snd.
audio/x-wav	Archivo de sonido en formato WAV.
audio/midi	Archivo de sonido en formato MIDI, que tendrá la extensión .mid.
application/postscript	Indica al cliente que los datos están en formato postscript.
application/zip	Archivo comprimido en formato ZIP.
image/gif image/jpeg image/x-xbitmap	Son los tres tipos de imágenes soportados en general por los clientes WWW sin llamar a un viewer externo.
text/HTML	Es el tipo de dato más utilizado, indica al cliente que los datos deben interpretarse como mandatos HTML.
text/plain	Indica al cliente que los datos son de texto plano, que no debe interpretarse en ningún sentido.
video/mpeg	Archivo de vídeo en formato MPEG, que tendrá una de las siguientes extensiones: .mpeg, .mpg, .mpe.
video/xmvideo	Archivo de vídeo en formato AVI.

Tabla 1.3 Tipos de Datos que genera un Programa CGI.

- **Location:** No solo es posible devolver un cierto tipo de información mediante un CGI, sino también un puntero a la información deseada. Dicho puntero será un URI. Cabe mencionar que esta línea es opcional.

Para dicho efecto se define otra cabecera de CGI denominada Location, cuya sintaxis es la siguiente:

Location :<URI>

Por ejemplo, la salida de un CGI podría ser la siguiente:

Content-type: text/html

Location: http://misitio.com.ec:8000

- **Status:** Esta línea generalmente no es muy utilizada. Status permite indicar el código de retorno que el servidor debe enviar al cliente en el diálogo HTTP. Su sintaxis es la siguiente:

Status: código mensaje

Cuando esta línea no es incluida en las cabeceras CGI, el código de retorno es “200 OK”, lo que se traduce en el diálogo HTTP con el cliente, en la línea “HTTP 200 OK”.

- **Cuerpo**

El cuerpo contiene los datos del documento, los cuales deben estar acorde a lo especificado en la cabecera Content-type, antes mencionada. Por ejemplo, en la figura 1.66 se puede observar que en el cuerpo de un programa CGI, se ha ingresado código HTML, ya que la cabecera del documento es del tipo *text/html*.

```
#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<END;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 T
<html>
<head>
  <title>CGI</title>
</head>
<body>
  <div align="center">
    <h1>Salida de un CGI</h1>
    <form name="form1" action="http://www.yahoo.es" target="_blank" method="post">
      <input type="text" name="prueba"><br><br>
      <input type="submit" value="enviar" class="boton">
    </form>
  </div>
</body>
</html>

END
;
```

Figura 1.66 Código de un Programa CGI.



Figura 1.67 Salida de un programa CGI.

1.6.4 TIPOS DE CGI

Un programa CGI puede ser desarrollado para cumplir diversas tareas, de las más simples a las más complejas, es así que se pueden crear aplicaciones para llenar formularios simples hasta aplicaciones que permitan administrar remotamente una red. A continuación se describen algunas de las tareas frecuentemente realizadas, mediante un programa CGI:

- **Administración remota del servidor:** Este tipo de CGI permite interactuar con el servidor desde la web. Lo que hace es invocar los programas que controlan o modifican el comportamiento del servidor.

- **Buscador:** Permite localizar páginas en base a términos especificados, para esto usa una tabla que enumera las palabras y por cada una especifica las páginas dónde se encuentra.
- **Contador de accesos:** Este tipo de aplicación es muy común, permite contar el número de veces que se ha solicitado una página. Este valor se guarda en un fichero.
- **Contribuciones:** Permite añadir enlaces o anotaciones a una página, además indica el origen de los mismos.
- **Correo:** Obtiene información estructurada del usuario.
- **Estadísticas de uso:** Este tipo de CGI permite mantener un registro o log de las actividades producidas en el servidor.

1.6.5 FORMULARIOS HTML EN CGI

La aplicación más común de los programas CGI es la creación y tratamiento de formularios, que no son más que documentos HTML, con determinados elementos que permiten enviar información al servidor a través de una tecnología como CGI.

Un CGI es invocado mediante un mandato POST o GET, el cual se transmite desde el documento HTML. Una vez que el servidor recibe el documento, se procede a analizar y procesar la información que se introdujo en el formulario.

1.6.5.1 Funcionamiento de un formulario

En un formulario se describirán mediante elementos HTML las diferentes secciones en las que el cliente podrá ingresar información. Cada una de estas secciones contendrá varios elementos HTML los cuales se identifican mediante un nombre simbólico, cuando el formulario es enviado hacia el servidor, este recibe el conjunto de elementos identificándolos a través de su nombre junto al valor introducido por el cliente.

El programa encargado del tratamiento del formulario se ejecuta mediante el uso de un programa CGI en el lado del servidor, este programa puede escribirse en

cualquiera de los lenguajes de programación que soporte el desarrollo de este tipo de tecnología.

No es necesario que todos los campos de un formulario contengan información, podrán existir campos obligatorios y campos que no lo son, todo dependerá del tipo de información que se quiera obtener del cliente.

El fin de la edición de un formulario termina cuando se pulsa un botón, solo en ese momento es cuando el programa recibe la información del formulario. La transacción se efectúa una sola vez, es decir que si se ha ingresado un dato erróneo, corresponde al programa de tratamiento enviar un mensaje de error al cliente y nuevamente mostrar la pantalla de edición.

1.6.5.2 Elementos de un formulario

Los elementos en un formulario permiten realizar una interfaz de edición. La inclusión de elementos para creación de formularios no difiere con los procedimientos utilizados en la creación de documentos HTML estáticos.

- **Generación de un formulario** `<form>...</form>`

Esta etiqueta encierra todos los contenidos del formulario (botones, cuadros de texto, etc.) e indica la presencia de un formulario en un documento HTML. Emplea atributos comunes y define los siguientes atributos específicos:

- **`method="get o post"`** Especifica la forma cómo se envían los datos del formulario al servidor. El método que se emplea depende del modo de funcionamiento del servidor.
- **`action="URI"`** Indica el URI de la aplicación del servidor, el cual se encarga de procesar los datos del formulario.

La sintaxis general de apertura de un formulario es:

`<form method= "tipo de método" action="URI del script">`

La etiqueta `<form>` puede contener cualquier etiqueta a excepción de otro `<form>`. Cuando se requiera diferenciar los formularios unos de otros en un documento HTML, se puede hacer uso de una etiqueta de encabezado como por ejemplo: `<h1>`.

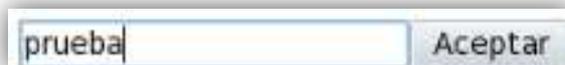


Figura 1.68 Visualización de la etiqueta `<form>`.

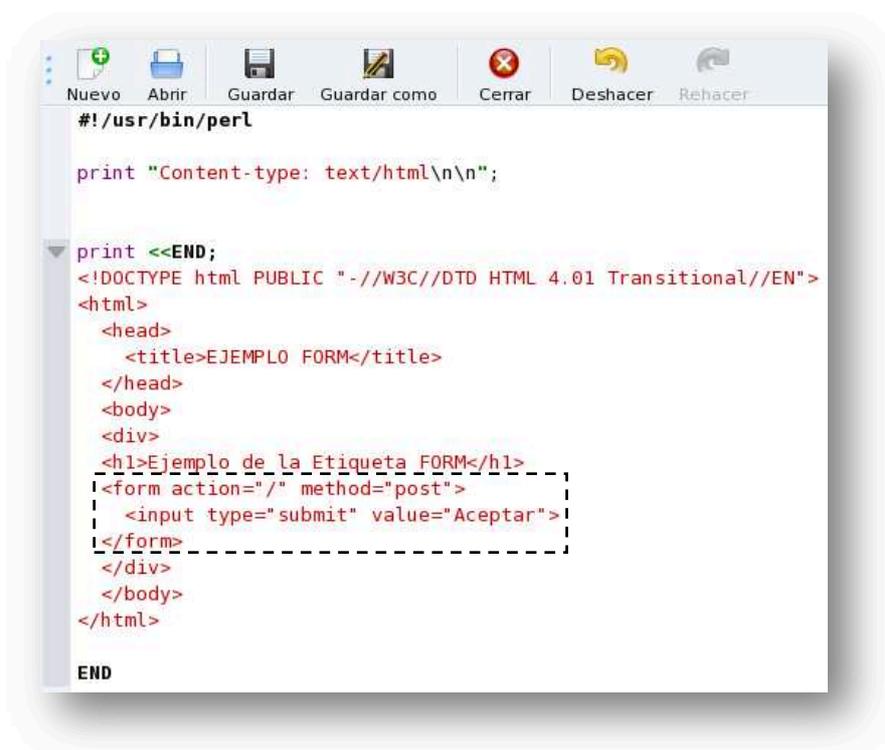


Figura 1.69 Programa CGI que incluye la etiqueta `<form>`.

- **La etiqueta `<input>`**

La etiqueta INPUT se utiliza para indicar un elemento de entrada simple dentro de un `<form>`. Esta etiqueta es autónoma, ya que no es terminal ni está rodeada obligatoriamente de cierta clase de código. Emplea atributos comunes y entre los atributos específicos se tiene:

- **`type= "tipo de entrada"`** Especifica el tipo de elementos que se incluirá en el formulario. Los valores que `type` puede tener son los siguientes:

text, password, checkbox, radio, submit, reset, file, hidden, image, button.

- ***name="nombre de la etiqueta"*** Asigna un nombre al elemento input, de esta manera el servidor puede procesar el formulario.
- ***value="valor del elemento"*** Es el valor inicial del elemento input.
- ***size="tamaño"*** Se refiere al tamaño inicial del elemento, cuando se trata del campo texto y password, se refiere al número de caracteres, para los demás controles hace referencia al tamaño en pixeles.
- ***maxlength="tamaño"*** Número máximo de caracteres para el elemento input text y password.
- ***checked="checked"*** Permite indicar la opción preseleccionada. Es utilizado en los elementos radiobutton y checkbox.
- ***disabled="disabled"*** Se usa para deshabilitar el elemento, de esta manera su valor no se envía al servidor junto con el resto de datos.
- ***readonly="readonly"*** Indica que el contenido del elemento es solo de lectura.
- ***src="URI"*** Este atributo es usado por el elemento para crear botones con imágenes, e indica el URI de la imagen que se usará como botón de formulario.
- ***alt="text"*** Se usa para descripción alterna al uso de una imagen.

- ***Tipos de controles creados con input***

El tipo de elemento definido por INPUT depende del atributo *type*.

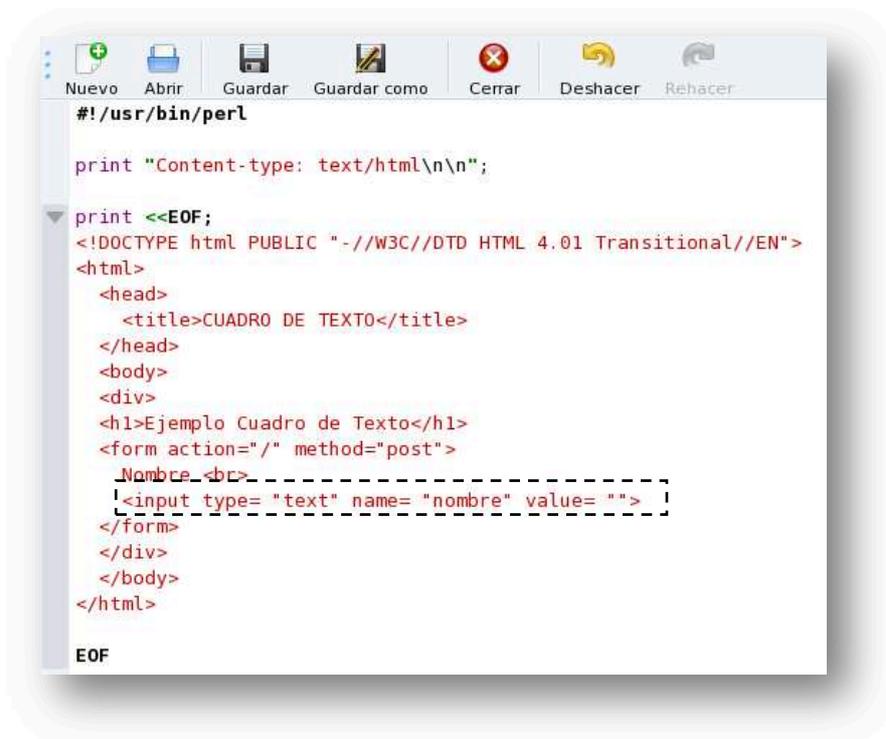
Cuadro de Texto

Es uno de los elementos más utilizados en los formularios. Este muestra un cuadro de texto vacío, en el cual se puede introducir texto.



Figura 1.70 Ejemplo de etiqueta input (type="text").

A continuación se muestra el programa CGI correspondiente a la figura 1.70:



```

#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>CUADRO DE TEXTO</title>
  </head>
  <body>
    <div>
      <h1>Ejemplo Cuadro de Texto</h1>
      <form action="/" method="post">
        Nombre <br>
        <input type="text" name="nombre" value="" >
      </form>
    </div>
  </body>
</html>

EOF

```

Figura 1.71 Programa CGI que incluye la etiqueta `<input type="text">`.

El atributo *type* es el que permite diferenciar los elementos que se pueden crear con la etiqueta `<input>`, como se puede ver en la figura 1.71 el valor que toma *type* es *text*, ya que se desea crear un cuadro de texto. En los campos de los formularios el atributo *name* es el más importante, si un campo no lo incluye, sus datos no serán enviados al servidor, ya que el valor indicado en este atributo, que en este caso es *nombre*, es utilizado por la aplicación del servidor para obtener el valor de este campo del formulario. Si se crea un formulario para insertar datos, los cuadros de textos deben estar vacíos, es decir que no se debe incluir el atributo *value* o como en el ejemplo anterior se incluye este atributo con un valor vacío *value=""*. Si se desea crear un formulario para modificar datos este atributo contendrá el valor que se desea mostrar.

Cuadro de Contraseña

Es similar al anterior, la diferencia radica en que el texto introducido se presenta de tal modo que se oculten los caracteres, usualmente mediante una serie de asteriscos. Es muy útil para escribir passwords e información sensible.

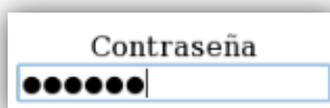


Figura 1.72 Ejemplo de etiqueta input (type="password").

```

Nuevo  Abrir  Guardar  Guardar como  Cerrar  Deshacer  Rehacer
#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>CUADRO DE CONTRASE&ntilde;a</title>
  </head>
  <body>
    <div>
      <h1>Ejemplo Cuadro de Contrase&ntilde;a</h1>
      <form action="/" method="post">
        <input type="password" name="contraseña" value="">
      </form>
    </div>
  </body>
</html>

EOF

```

Figura 1.73 Programa CGI que incluye la etiqueta `<input type="password">`.

Checkbox

También denominadas casillas de verificación, son elementos que permiten seleccionar y deseleccionar opciones de forma individual.

Música Preferida: Rock Pop Romántica

Figura 1.74 Ejemplo de etiqueta input (type="checkbox").

```
#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>CHECKBOX</title>
  </head>
  <body>
    <div>
      <h1>Ejemplo Checkbox</h1>
      <form action="/" method="post">
        Música Preferida:
        <input type="checkbox" name="musica" value="rock">Rock
        <input type="checkbox" name="musica" value="pop">Pop
        <input type="checkbox" name="musica" value="romantica">Romántica
      </form>
    </div>
  </body>
</html>

EOF
```

Figura 1.75 Programa CGI que incluye la etiqueta `<input type="checkbox">`.

En este caso el valor que toma *type* es *checkbox*. El texto que se encuentra a continuación de cada checkbox como se puede ver en la figura 1.75, no se ha creado mediante ningún atributo, debido a que se lo ha creado manualmente fuera del control del formulario. Si se desea mostrar un checkbox seleccionado por defecto, se usa el atributo *checked*.

```
<input type="checkbox" ... checked="checked" />
```

Radiobutton

También conocido como casillas de selección, son elementos mutuamente excluyentes, es decir que su contenido activo o inactivo condiciona al de los restantes botones de su clase agrupados en un mismo bloque. Cada vez que se selecciona una opción, inmediatamente se deselecciona la otra opción.

Género:
 Femenino
 Masculino

Figura 1.76 Ejemplo de etiqueta input (type="radio").

```
#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>RADIOBUTTON</title>
  </head>
  <body>
    <div>
      <h1>Ejemplo Radiobutton</h1>
      <form action="/" method="post">
        Géneros:
        <input type= "radio" name= "genero" value= "femenino">Femenino<br>
        <input type= "radio" name= "genero" value= "masculino">Masculino
      </form>
    </div>
  </body>
</html>

EOF
```

Figura 1.77 Programa CGI que incluye la etiqueta `<input type= "radio">`.

Para crear un radiobutton el valor de *type* es *radio*. Para este tipo de elemento el atributo *name* es utilizado para indicar los radiobutton que pertenecen a un mismo bloque. Como se puede observar en la figura 1.77 el atributo *name* de los dos radiobutton tienen el mismo nombre, lo cual indica que están relacionados.

Botón de Envío de Formulario

El valor del atributo para este elemento es *submit*. Es un botón que permite enviar al servidor los datos introducidos por el usuario.



Figura 1.78 Ejemplo de etiqueta input (type="submit").

```

#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>BOTON ENVIO DE FORMULARIO</title>
  </head>
  <body>
    <div>
      <form action="/" method="post">
        <input type="submit" name="buscar" value="Buscar">
      </form>
    </div>
  </body>
</html>

EOF

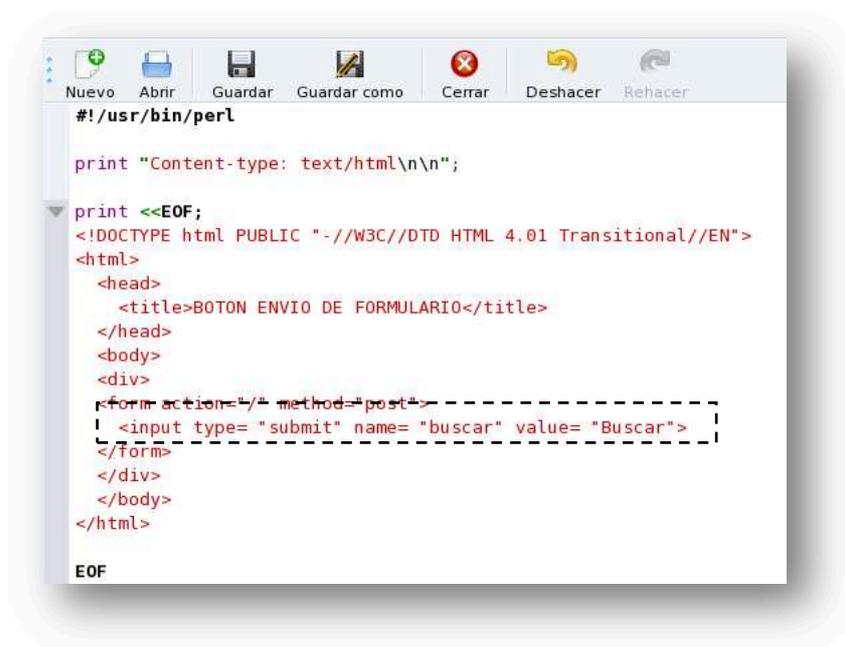
```

Figura 1.79 Programa CGI que incluye la etiqueta `<input type="submit">`.

Si no se define el atributo *value* el navegador empleado por el cliente muestra un texto predefinido respectivo.

Campos Ocultos

Se utiliza para añadir información oculta en el formulario. El valor de *type* utilizado para este elemento es *hidden*. Los campos ocultos no se muestran en pantalla.



```

#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <title>BOTON ENVIO DE FORMULARIO</title>
</head>
<body>
<div>
  <form action="/" method="post">
    <input type="submit" name="buscar" value="Buscar">
  </form>
</div>
</body>
</html>

EOF

```

Figura 1.80 Programa CGI que incluye la etiqueta `<input type="hidden">`.

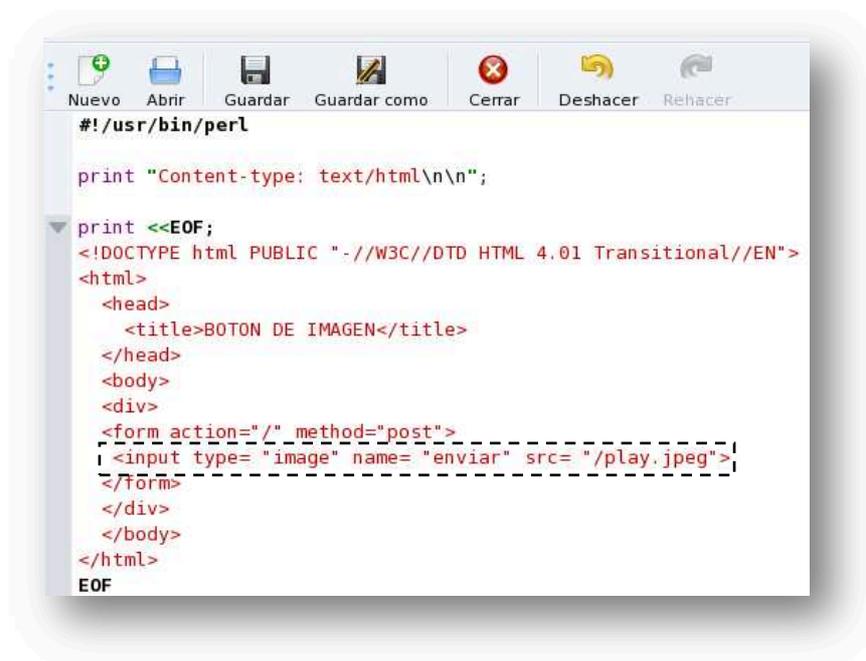
Este elemento es utilizado cuando se requiere incluir información necesaria para el servidor, pero que no es preciso mostrar al usuario.

Botón de Imagen

Permite personalizar el aspecto de los botones en el formulario. El valor del atributo `type` usado para este elemento del formulario es `image`. El atributo `src` indica el URI de la imagen que debe mostrar el navegador en lugar del botón normal. Esta imagen debe residir en el directorio asignado por el Webmaster para el almacenamiento de imágenes, por ejemplo el servidor web Apache designa por defecto el directorio `/var/www/html`.



Figura 1.81 Ejemplo de etiqueta `input (type="image")`.



```

#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>BOTON DE IMAGEN</title>
  </head>
  <body>
    <div>
      <form action="/" method="post">
        <input type="image" name="enviar" src="/play.jpeg">
      </form>
    </div>
  </body>
</html>
EOF

```

Figura 1.82 Programa CGI que incluye la etiqueta `<input type="image">`.

Botón

Permite realizar actividades complejas dentro del formulario, como por ejemplo guardar cambios. Este tipo de control por sí solo no realiza ninguna acción específica, siempre requiere de un lenguaje de programación, de esta manera el botón se programará para realizar una tarea cuando se pulse sobre este. El valor del atributo `type` utilizado para este elemento del formulario es `button`.



Figura 1.83 Ejemplo de etiqueta `input (type="button")`.

```

#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>BOTON</title>
  </head>
  <body>
    <div>
      <form action="/" method="post">
        <input type="button" name="guardar" value="Guardar Cambios">
      </form>
    </div>
  </body>
</html>

EOF

```

Figura 1.84 Programa CGI que incluye la etiqueta `<input type= "button">`.

Si no se define el atributo *value*, el navegador del cliente muestra el texto predefinido respectivo.

- **Elemento select** `<select>...</select>`

Puede incluirse dentro de una estructura `<form>...</form>` conjuntamente con texto y otros elementos HTML. Las etiquetas `<select>` se visualizan como listas desplegables y menú de opciones.

Existen etiquetas `<select>` de apertura y cierre, a diferencia de la etiqueta `<input>`. Dentro de una etiqueta `<select>`, solo se permite etiquetas `<option>`, cuyo contenido será únicamente texto, sin marcadores HTML.

```

<select name= "Sistemas Operativos">
  <option> Windows</option>
  <option> Linux</option>
  <option> Otro</option>
</select>

```

Los atributos más utilizados son los siguientes:

- ***name="nombre"*** Es el nombre del elemento `<select>`, es un campo obligatorio.
- ***size="número"*** Cuando este valor es menor que 1, `<select>` se mostrará como un menú de opciones. Si `size` es mayor o igual a 2, `<select>` se mostrará como una lista desplegable, es decir que en este caso representa el número de filas que se muestran en la lista, por default se muestra solo una.
- ***multiple="multiple"*** Permite seleccionar más de un elemento.

Otros atributos de `<select>` son: `disable`, `onchange`, `onfocus`, `onblur`, etc.



Figura 1.85 Ejemplo de etiqueta `select`.

```

Nuevo  Abrir  Guardar  Guardar como  Cerrar  Deshacer  Rehacer
#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>EJEMPLO SELECT</title>
  </head>
  <body>
    <div>
      <h1>Ejemplo de la Etiqueta SELECT</h1>
      <form action="/" method="post">
        <select name= "Sistemas Operativos" size="3" multiple="multiple">
          <option>Windows</option>
          <option>Mac</option>
          <option>Linux</option>
          <option>Otro</option>
        </select>
      </form>
    </div>
  </body>
</html>
EOF

```

Figura 1.86 Programa CGI que incluye la etiqueta `<select>`.

- **El elemento <option>...</option>**

Este elemento es empleado para definir cada dato dentro de un elemento <select>. A la etiqueta <option> se asocian los siguientes atributos:

- **selected="selected"** Especifica que opción se elige por defecto. Si en el atributo <select> se admite múltiples selecciones mediante el atributo <multiple>, es posible señalar varias opciones <selected>.
- **value="text"** Es el valor que se envía al servidor cuando se selecciona una opción.

Otros atributos son: label, disabled, etc.



Figura 1.87 Ejemplo de etiqueta option y select.

```

#!/usr/bin/perl

print "Content-type: text/html\n\n";

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>OPTION</title>
  </head>
  <body>
    <div>
      <form action="/" method="post">
        <label>Sistema Operativo</label> <br>
        <select id="so" name="so">
          <option value="" selected="selected">- selecciona -</option>
          <option value="windows">Windows</option>
          <option value="linux">Linux</option>
          <option value="mac">Mac</option>
          <option value="otro">Otro</option>
        </select>
      </form>
    </div>
  </body>
</html>
EOF

```

Figura 1.88 Programa CGI que incluye la etiqueta <option>.

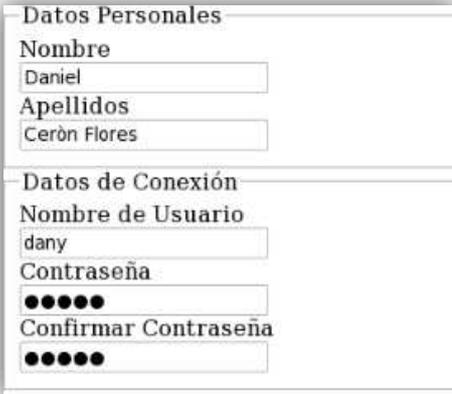
- **Elementos `<fieldset>..</fieldset>`, `<legend>..</legend>` y `<label>..</label>`**

Estos elementos son utilizados para mejorar la estructura de los formularios creados.

- **`<fieldset>...</fieldset>`:** Permite agrupar de forma lógica varios campos de un formulario. De esta manera se facilita la navegación por el formulario.
- **`<legend>...</legend>`:** Define el título o leyenda de un conjunto de campos de formulario agrupados con la etiqueta *fieldset*.

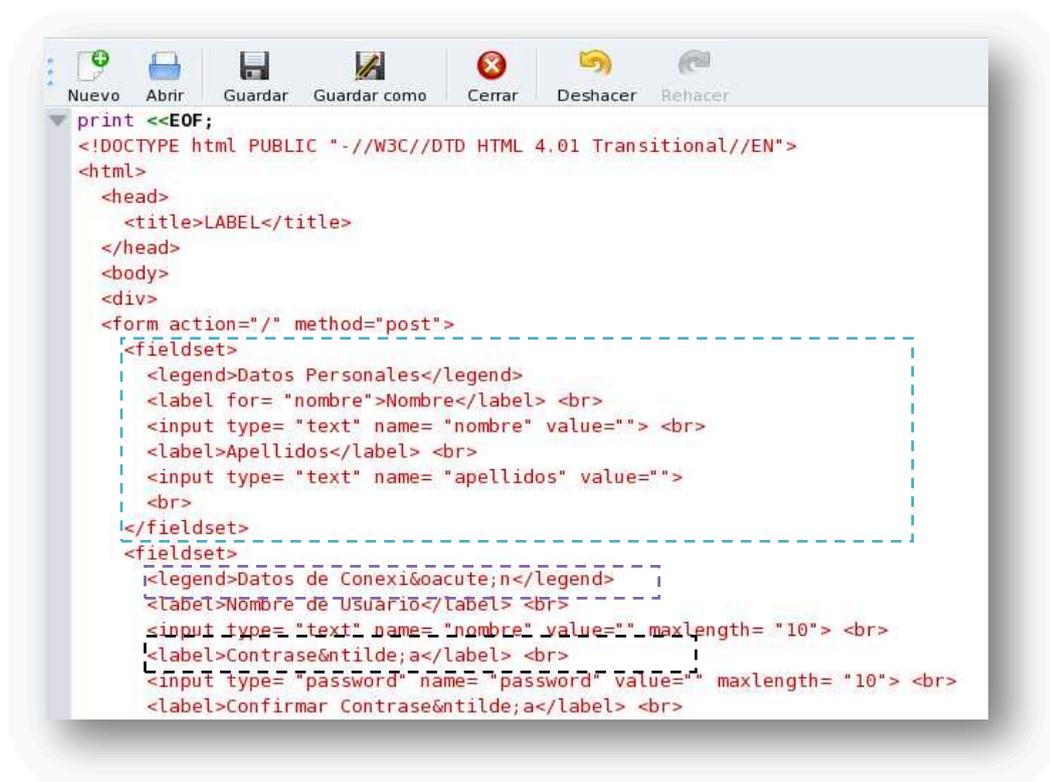
Como se puede ver en la figura 1.89 el navegador muestra un borde por cada agrupación de elementos. La etiqueta *legend* se encuentra dentro de cada etiqueta *fieldset* y presenta un título por cada agrupación de elementos.

- **`<label>...</label>`:** Define el título o leyenda de los campos establecidos en el formulario. Esta etiqueta es muy útil para el uso con elementos como *radiobutton* y *checkbox*. El único atributo que generalmente se utiliza con esta etiqueta es:
 - **`for= "id de elemento"`** Especifica el identificador del campo de formulario para el que esta etiqueta hace de título.



The image shows a web form with two distinct sections, each enclosed in a box-like container. The first section is titled "Datos Personales" and contains two text input fields. The first field is labeled "Nombre" and contains the text "Daniel". The second field is labeled "Apellidos" and contains the text "Cerón Flores". The second section is titled "Datos de Conexión" and contains three text input fields. The first field is labeled "Nombre de Usuario" and contains the text "dany". The second field is labeled "Contraseña" and contains five dots. The third field is labeled "Confirmar Contraseña" and contains five dots.

Figura 1.89 Ejemplo de etiquetas `<label>` `<legend>` y `<fieldset>`.



```

print <<EOF;
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>LABEL</title>
  </head>
  <body>
    <div>
      <form action="/" method="post">
        <fieldset>
          <legend>Datos Personales</legend>
          <label for="nombre">Nombre</label> <br>
          <input type="text" name="nombre" value="" <br>
          <label>Apellidos</label> <br>
          <input type="text" name="apellidos" value="" <br>
          <br>
        </fieldset>
        <fieldset>
          <legend>Datos de Conexi&oacute;n</legend>
          <label>Nombre de Usuario</label> <br>
          <input type="text" name="nombre" value="" maxlength="10"> <br>
          <label>Contrase&ntilde;a</label> <br>
          <input type="password" name="password" value="" maxlength="10"> <br>
          <label>Confirmar Contrase&ntilde;a</label> <br>
        </fieldset>
      </form>
    </div>
  </body>
</html>

```

Figura 1.90 Programa CGI que incluye las etiquetas `<label>` `<legend>` y `<fieldset>`.

- **El elemento `<textarea>`...`</textarea>`**

Permite crear una zona de texto, cuyo tamaño está definido por los siguientes atributos:

- **`rows="número"`** Define el número de filas del campo de texto, es decir representa la altura vertical en caracteres.
- **`cols="número"`** Define el número de columnas del campo texto, es decir representa la anchura horizontal en caracteres.

Otros atributos son: `name`, `disable`, `readonly`, `onselect`, `onchange`, `onfocu`, `onblur`, etc.

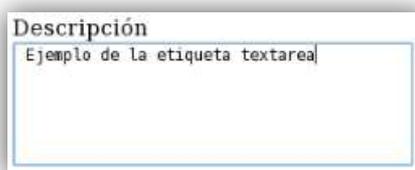


Figura 1.91 Ejemplo de etiqueta textarea.



Figura 1.92 Programa CGI que incluye la etiqueta `<textarea>`.

Esta etiqueta presenta una desventaja, y es que no define un atributo que limite el máximo número de caracteres que se puede introducir. Para poder limitar su tamaño se requiere el uso de algún lenguaje de programación.

1.6.5.3 Métodos de Envío de un Formulario

El atributo *method* de la etiqueta *form*, anteriormente descritos, especifica el método HTTP usado para enviar formularios al servidor. Este atributo puede tener dos valores:

- **GET:** El método por omisión, que incluye el contenido del formulario al URI como si se tratara de una consulta normal. Una gran desventaja de este método es no permitir el envío de archivos adjuntos conjuntamente con los formularios. *Get* es utilizado en formularios de búsqueda, es decir en aquellos formularios que no modifican la información.

- **POST:** Este método envía el contenido del formulario al servidor en forma de bloque de datos y no como parte del URI. Esta característica permite que este método sea el más utilizado, *post* permite el envío de mucha más información en comparación al anterior, es utilizado cuando el formulario modifica la información original, es decir que pueden insertar, borrar o modificar información.

1.6.6 EJECUCIÓN DE UN PROGRAMA CGI

Para que un programa CGI pueda ejecutarse, éste debe contar con los permisos correspondientes. Se debe tomar en cuenta que los permisos en un archivo, permiten realizar sobre él tareas como lectura, escritura y ejecución. Los permisos de un archivo generalmente se presentan de la siguiente forma:

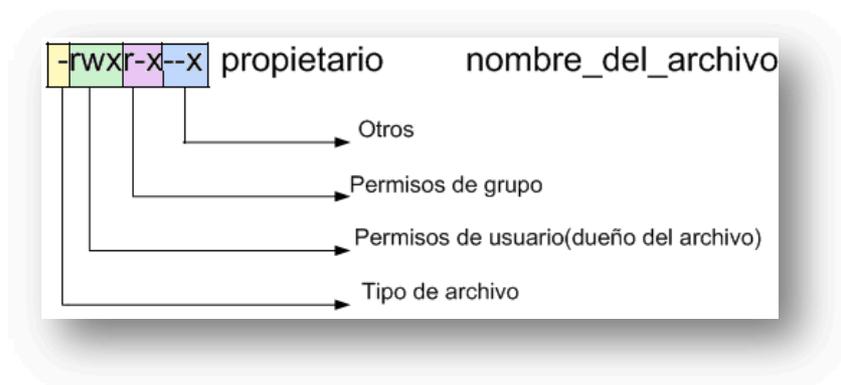


Figura 1.93 Permisos de un archivo.

La siguiente tabla contiene los posibles tipos de archivos que se pueden tener en el primer campo:

Símbolo	Significado
-	Archivo común
d	Directorio
c	Dispositivo de caracteres (ej: impresora)
b	Dispositivo de bloque (ej: CD-ROM)
l	Enlace simbólico
s	Socket
p	Pipe

Tabla 1.4 Tipos de archivo.

Los siguientes campos se toman en grupos de tres, cada grupo pertenece a una clase de permisos, dichos permisos pueden ser los siguientes:

Permiso	Valor Numérico	Significado
r	4	Permiso de Lectura
w	2	Permiso de Escritura
x	1	Permiso de Ejecución

Tabla 1.5 Tipos de Permisos.

Además de los tipos de permisos, se define al propietario del archivo y las acciones que se realiza sobre el mismo.

Identidades	Significado
u	Usuario propietario del archivo
g	Grupo al que pertenece el usuario
o	Otros (ni el propietario ni su grupo)
a	Todos (u, g y o)

Tabla 1.6 Identidades.

Acciones	Significado
+	Añadir los permisos
-	Eliminar los permisos
=	Hace que sea el único permiso

Tabla 1.7 Acciones sobre los archivos.

De acuerdo a lo antes mencionado, si se observa la figura 1.93 se puede ver que el primer campo (-) corresponde a un archivo común. El segundo campo (rwx) indica que el propietario de ese archivo tiene permisos de lectura, escritura y ejecución. El tercer campo (r-x) indica que los usuarios del grupo tienen permisos de lectura y ejecución, pero no tiene permisos de escritura. El siguiente campo (--x) indica que los demás usuarios solo pueden ejecutar el archivo.

Normalmente los programas CGI tienen permisos -rwxr--r--.

Una vez que el programa CGI cuente con los permisos necesarios, se procede a ejecutarlo, para esto en el navegador web se debe digitar el nombre del servidor o su dirección IP, seguido del directorio que contiene los programas CGI (cgi-bin) y por último el nombre del programa cgi.



Figura 1.94 Ejecución de un programa CGI.

Cuando el cliente ha enviado el URI que contiene el CGI, como se puede visualizar en la figura 1.95, el servidor recibe dicha petición y comprueba si la invocación se trata de un CGI. A continuación el servidor prepara el entorno para ejecutar la aplicación, lo ejecuta, capturando su salida estándar. Posteriormente la aplicación realiza su función, mientras se va generando un objeto MIME que la aplicación escribe en su correspondiente salida estándar. Cuando la aplicación finaliza, el servidor envía conjuntamente la información generada con la propia, al cliente, el cual se encuentra en estado de espera. La aplicación debe especificar el tipo de objeto MIME que se genera (por ejemplo Content-type: text/html), mientras que el servidor se encargará de calcular el tamaño del objeto producido.

1.6.7 SEGURIDAD EN LOS PROGRAMAS CGI

Los programas CGI son frecuentemente utilizados para realizar procesos de administración dentro de un servidor, esto implica el envío de información confidencial, en el caso de que un usuario mal intencionado la obtuviese, podría utilizarla para varios fines, por ejemplo cambiar algunas de las configuraciones realizadas en el servidor. Muchos de estos riesgos se deben a la forma en cómo se escriben los CGI's y a la configuración del servidor en cuanto a seguridad.

Cuando se invoca un programa CGI o se envía un formulario, de una u otra forma el servidor está permitiendo al usuario ejecutar un programa que reside en dicho servidor. Es decir, que prácticamente el usuario tiene la oportunidad de controlar la ejecución del CGI, ya sea desde la línea de URI o después de llenar un formulario, de manera que un usuario malicioso podría intentar introducir una serie de parámetros que permitan que el CGI funcione de manera inadecuada.

Cuando se programa un CGI, sobre todo si va a ser ejecutado en un servidor, se deben tener en cuenta las siguientes consideraciones:

- La posibilidad de que el CGI sea violentado por la entrada del usuario para que se ejecuten ciertos comando, los cuales pueden causar graves daños en el servidor.
- La posibilidad de revelar información sensitiva del servidor, ya que permitirá al usuario malicioso conocer mejor la configuración del sistema y así buscar posibles agujeros por los cuales pueda atacar.
- Lo más recomendable es designar un único directorio en donde se almacenen todos los programas CGI, que típicamente es el directorio cgi-bin, dicho directorio deberá tener permisos limitados, de modo que los usuarios no puedan modificar, eliminar o instalar programas.
- Los métodos empleados para enviar los datos de los formularios desde el navegador al servidor Web son: GET y POST, los cuales se especifican en el atributo METHOD. La diferencia entre estos métodos es que el CGI recibirá los datos enviados con POST leyendo la entrada estándar, mientras que los enviados con GET se recibirán por líneas de comandos y la variable de entorno QUERY_STRING. Generalmente se suele emplear el método POST, pero desde el punto de vista de seguridad ninguno es mejor que otro, si bien las peticiones enviadas mediante el método GET son registradas en los ficheros de registro o logs, en el caso de enviar información confidencial sin cifrar, dicha información quedaría expuesta para cualquier usuario malicioso.

1.7 LENGUAJES PARA LA PROGRAMACION DE CGI

Los CGI's pueden ser desarrollados en varios lenguajes de programación, sin embargo hay que tener en cuenta el tipo de sistema operativo sobre el cual se van a desarrollar los CGI's, de manera que se pueda saber el tipo de lenguaje de programación que el servidor puede interpretar, la mayoría de servidores web donde se implementan los CGI's son desarrollados bajo el sistema operativo LINUX, aunque también pueden ser implementados bajo otro tipo de sistema.

Debido a que los CGI's reciben sus parámetros a manera de texto, resulta útil emplear un lenguaje que permita la manipulación sencilla de este tipo de parámetros junto con la manipulación de ficheros, ya que por lo general se requiere manipular archivos dentro de ficheros en el servidor. Dado que CGI es una interfaz que lleva varios años siendo utilizada se han determinado lenguajes donde el desarrollo de CGI's es más avanzado en relación a otros, por lo que la cantidad de información de uno respecto a otro es escasa, en los siguientes párrafos se trata de explicar los distintos lenguajes, donde se implementaron o se implementan actualmente este tipo de interfaces.

1.7.1 SCRIPT DE BASH

En sistemas operativos basados en UNIX el principal intérprete de lenguajes de instrucción lo constituye el bourne again Shell más conocido como bash, el cual a través del Shell del sistema operativo permite ingresar instrucciones para llevar a cabo una tarea, es el caso del desarrollo de un simple CGI, por ejemplo:



```
#!/bin/bash

echo "Content-type: text/html"
echo ""
echo '<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
>
  "http://www.w3.org/TR/html4/loose.dtd">'
echo '<html>'
echo '<head>'
echo '<title>CGI desarrollado en bash</title>'
echo '</head>'
echo '<body>'
echo 'Hola Mundo'
echo '</body>'
echo '</html>'
```

Figura 1.95 CGI desarrollado en Bash.

Con este ejemplo se puede apreciar la manera de generar una página web que proporciona el mensaje *"Hola Mundo"*, mostrando en el título del navegador web el mensaje: *CGI desarrollado en bash*. Ahora cada una de las líneas del script realiza una tarea:

- **#!/bin/sh:** esta línea se encarga de buscar el intérprete para la información que viene en las siguientes líneas.
- **echo "Content-type: text/html":** la instrucción echo es interpretada por bash como una instrucción print, es decir que muestre la información en pantalla, para este caso se especifica que la información es del tipo HTML, es decir que va a ser visualizada en un navegador web.
- **echo " ":** es una línea en blanco empleada por el intérprete para separar la definición del tipo de contenido, del contenido real del documento.
- **echo<etiquetas>:** estas líneas proporcionan el código HTML para la generación de una página web en la cual se visualizará el mensaje *Hola Mundo*.

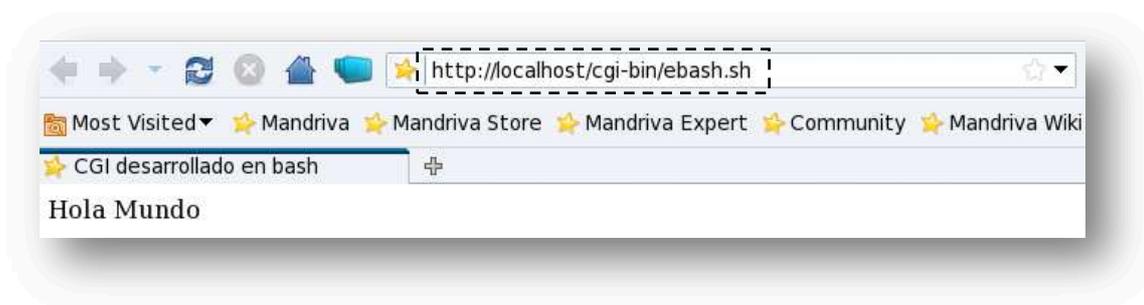


Figura 1.96 Visualización de CGI desarrollado en Bash.

Aunque el bash constituye un potente medio de desarrollo no proporciona todas las herramientas necesarias para desarrollar CGI's que cumplan tareas más complejas, para este caso existen otros lenguajes que permiten desarrollar CGI's con un grado mayor de complejidad.

1.7.2 APPLESCRIPT

Applescript es un lenguaje de programación desarrollado por Apple para su sistema operativo Mac OS, constituye un lenguaje de scripting que permite desarrollar tareas paralelas al desarrollo de aplicaciones que utilizan interfaces gráficas GUI²².

²² **Graphical User Interface (GUI).**- Interfaz Gráfica de Usuario, constituye un conjunto de objetos gráficos tales como iconos, ventanas, cursores, etc. que actúan como interfaz para que el usuario

El desarrollo de CGI's mediante applescript se realiza separando la parte correspondiente al documento HTML, de la programación realizada para el tratamiento de la información proporcionada por un formulario. Es el caso del siguiente ejemplo tomado de un documento HTML disponible en internet, el cual desarrolla un CGI en base a los siguientes pasos:

- Se crea un documento en texto plano donde se ingresa las etiquetas para el desarrollo de un documento HTML con formularios. Este documento se lo almacena según el autor del documento en la carpeta de páginas web del sistema operativo con la extensión HTML.
- Se crea un applescript con el código correspondiente para el manejo de la información que proporciona el formulario. Y se lo almacena en la misma carpeta que el documento anterior pero con la extensión ACGI y se procede a demostrar su funcionamiento.

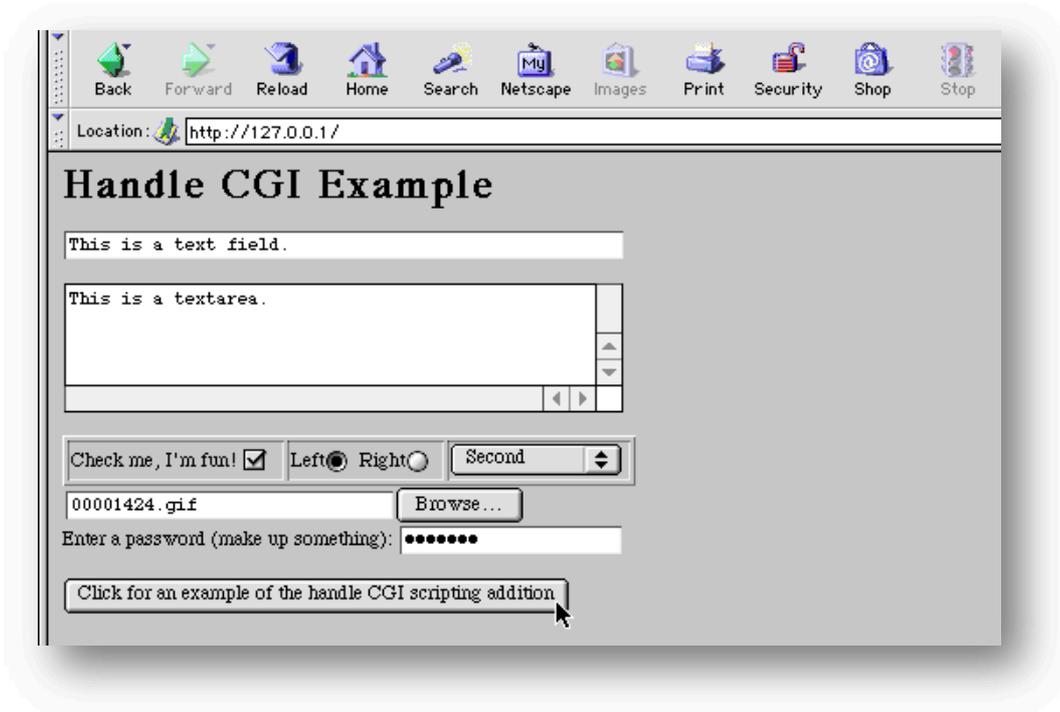


Figura 1.97 Formulario desarrollado en Applescript.²³

pueda visualizar y observar las acciones que se puede realizar sobre la información proporcionada por la interfaz.

²³ WEIDERSPAN, Jon: **Writing CGIs**.

Tomado de: <http://www.linkedresources.com/teach/applescript/writingCGIs.shtml>



Figura 1.98 Visualización de un CGI desarrollado en Applescript.²⁴

No se encuentra información actualizada acerca de cómo se realizan CGI's con applescript, lo que se trata de indicar con los párrafos anteriores es la facilidad que tienen las interfaces CGI para ser implementadas dentro de otros sistemas operativos como es el caso de MAC OS.

1.7.3 C/C++

Para implementar un CGI mediante el lenguaje de programación C/C++ se requiere que este sea convertido en un ejecutable binario. Por lo general muchos de los desarrolladores emplean herramientas disponibles en el sistema operativo Windows para compilar y obtener un ejecutable binario, pero contrarresta con el sistema operativo de mayor utilización dentro de los servidores web ya que estos emplean el sistema operativo Linux, por lo que se hace complicada la ejecución de un programa desarrollado en una arquitectura respecto a la otra. También se debe tener presente que hay que permitir la ejecución del programa agregando

²⁴ WEIDERSPAN, Jon: **Writing CGIs**.

Tomado de: <http://www.linkedresources.com/teach/applescript/writingCGIs.shtml>

permisos dentro del servidor, es decir se debe permitir la manipulación de ficheros dentro del servidor.

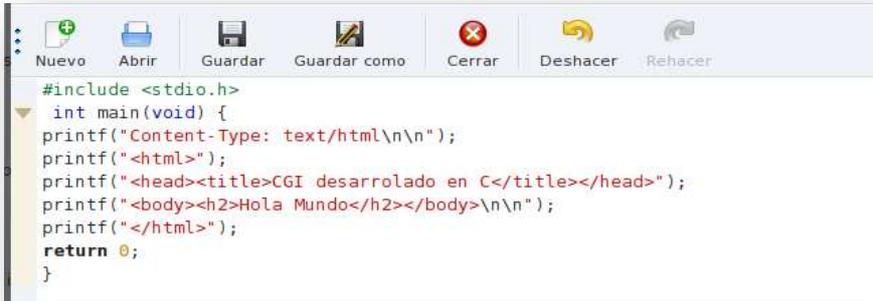
Para asegurar la solución a los problemas planteados se puede emplear un compilador de C dentro del servidor web, que permita generar el ejecutable binario, así como permitir el inicio de sesión para un usuario que requiera realizar modificaciones sobre ficheros. Normalmente se sigue un conjunto de pasos para la implementación de un CGI empleando el lenguaje de programación C/C++, estos pasos son:

- Compilar y ejecutar el programa desarrollado en C/C++, para el caso de un servidor Linux que es el más empleado en la actualidad, se cuenta con el compilador gcc o cc que permite obtener el ejecutable binario.
- Realizar los cambios que pudieran ser necesarios para emplearlo como un script CGI, para ello se debe determinar que el ejecutable lea la información de acuerdo a como es enviada por el formulario a través del método GET y la variable de entorno QUERY_STRING. También se debe determinar el método de salida de la información empleando la salida estándar (stdout) ya que los CGI's emplean tanto la entrada como salida estándar para comunicarse con los clientes. Por esta razón hay que generar una salida que contenga una cabecera HTTP para ser interpretada por el cliente, toda esta información de salida está en formato HTML.
- Se ejecuta una nueva compilación con los cambios realizados y se ejecutan pruebas para verificar el funcionamiento del mismo.
- Luego de realizar las pruebas respectivas se procede a realizar la última compilación para obtener el archivo cgi definitivo, por ejemplo, mediante la siguiente línea de comando: `gcc -o prueba.cgi prueba.c` se obtendría el ejecutable prueba.cgi mediante la compilación del archivo prueba.c.
- El programa prueba.cgi debe ser colocado en el directorio correspondiente a CGI's, dentro de servidores Linux el fichero se denomina cgi-bin, y corresponde al directorio que tiene permisos de ejecución por defecto dentro de estos servidores.

- Finalmente se crea un documento HTML con formularios para invocar el archivo CGI y verificar el correcto funcionamiento del ejecutable. Con estos pasos se debería realizar de manera correcta un ejecutable de C/C++ para ser utilizado como un archivo CGI.

Un ejemplo sencillo para verificar el funcionamiento de CGI's desarrollados en C/C++ constituye la generación del mensaje hola mundo.

El código correspondiente para el archivo a ser compilado es:



```

#include <stdio.h>
int main(void) {
printf("Content-Type: text/html\n\n");
printf("<html>");
printf("<head><title>CGI desarrollado en C</title></head>");
printf("<body><h2>Hola Mundo</h2></body>\n\n");
printf("</html>");
return 0;
}

```

Figura 1.99 CGI desarrollado en C.

Se procede a almacenar los archivos en el directorio correspondiente:



Figura 1.100 Almacenamiento del archivo ccgi.c.

Se realiza la compilación:

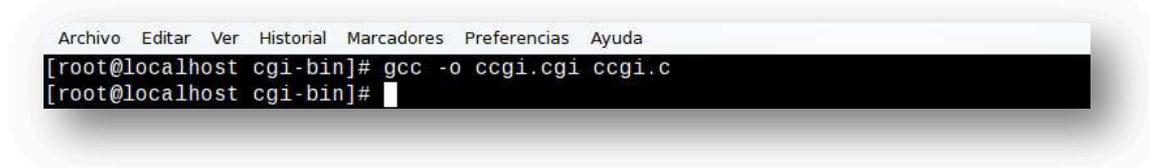


Figura 1.101 Compilación del archivo ccgi.c.

Se obtiene el archivo ejecutable:



Figura 1.102 Almacenamiento del ejecutable ccgi.cgi.

Luego de compilar el archivo y almacenarlo dentro del fichero correspondiente se debería verificar en un navegador web el mensaje *Hola Mundo* ingresando el URI correspondiente al servidor donde se encuentre el archivo CGI.

Una manera fácil de evitar copiar el archivo en la carpeta correspondiente es crear el archivo .c en la carpeta cgi-bin directamente, para compilarlo solo es necesario dirigirse mediante la consola de comandos hasta la carpeta cgi-bin y realizar la compilación correspondiente, de esta manera el ejecutable aparece directamente en esta carpeta y no es necesario copiarla a la carpeta cgi-bin. Con ello solo queda visualizar el CGI a través del navegador.

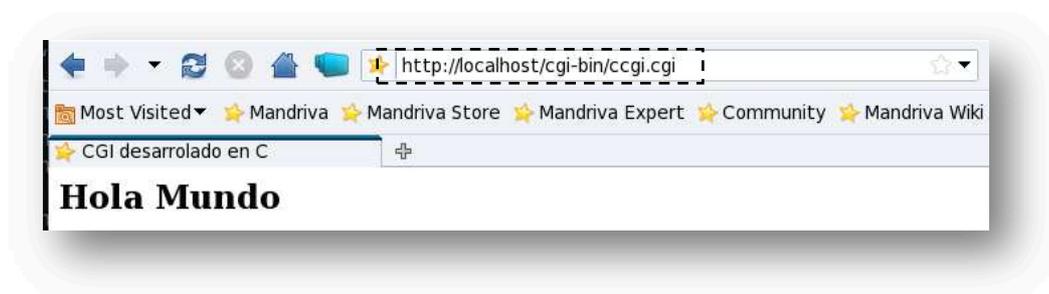


Figura 1.103 Visualización de CGI desarrollado en C.

Para el manejo de secuencias de texto enviadas desde el cliente a través de formularios se debe realizar un tratamiento especial ingresando dentro del código del archivo ejecutable funciones que permitan leer e interpretar la secuencia de texto que llega desde el cliente y darle el tratamiento correspondiente para obtener una salida HTML como respuesta al cliente.

La implementación de CGI utilizando C/C++ implica realizar pasos adicionales a lenguajes que interpretan directamente la información proporcionada por el cliente. Es por esta razón que resulta más complicado implementar CGI en este tipo de lenguaje pues complica aún más el desarrollo de los mismos.

1.7.4 VISUAL BASIC

El desarrollo de CGI's en sus inicios estuvo ligado a lenguajes de consola como es el caso del bash de UNIX o C/C++, mientras que visual basic fue desarrollado como lenguaje para trabajar en entornos visuales para sistemas operativos Windows, por esta razón la elaboración de CGI's no se la realizaba de manera directa, además CGI emplea la entrada (stdin) y salida estándar (stdout) tanto para manejar la información que proviene del cliente así como para enviar el resultado, mientras que Visual Basic no soportaba el manejo de las entradas estándar por lo que se necesitaba emplear métodos que permitan el manejo de este tipo de entradas.

Para permitir el manejo de la entrada y salida estándar se tuvo que utilizar la API²⁵ de Windows o emplear programas intermediarios que permitan la comunicación con la entrada y salida estándar. Hay que tener en cuenta que el servidor web en donde se va a implementar el CGI corresponde a un servidor Windows denominado IIS (Internet Information Server) obviamente bajo el sistema operativo Windows.

Algunas técnicas que permiten superar las dificultades presentadas para la lectura y envío de información se las menciona a continuación:

- Visual Basic puede emplear rutinas externas (DLL²⁶) para que la API del sistema operativo pueda leer y escribir en la entrada y salida estándar, y permitir el desarrollo de un CGI.
- Se puede emplear una aplicación intermedia que interactúe con Visual Basic y la entrada y salida estándar, es el caso de la aplicación WinCGI que permite realizar esta interacción.
- Una alternativa a la implementación de CGI constituye la utilización de una interfaz desarrollada para la generación de programas que se

²⁵ **Application Programming Interface (API).**- Interfaz de Programación de Aplicaciones, Constituye un conjunto de funciones y procedimientos que ofrecen las bibliotecas de un sistema operativo al programador, con la finalidad de facilitar la programación.

²⁶ **Dinamic-Link Library (DLL).**- Biblioteca de Enlace Dinámico, Es una biblioteca que contiene código y datos que pueden ser llamados por más de una aplicación al mismo tiempo o trabajar con otras DLL's para realizar algún tipo de tarea dentro del sistema operativo.

ejecuten sobre un servidor Windows, es el caso de ISAPI's (Internet Server Application Programming Interface o Interfaz de Programación de Aplicaciones para Servidores de Internet).

A manera de ejemplo se tomará la información proporcionada por la página web de Microsoft para demostrar cómo se realizaba la programación de CGI's en Visual Basic, dado que constituye un lenguaje de programación antiguo donde se implementó CGI, en la actualidad el desarrollo de páginas dinámicas en Windows es realizado a través del lenguaje ASP²⁷.

```
Option Explicit
Public Const STD_INPUT_HANDLE = -10&
Public Const STD_OUTPUT_HANDLE = -11&
Public Const CGI_AUTH_TYPE      As String = "AUTH_TYPE"
Public Const CGI_CONTENT_LENGTH As String = "CONTENT_LENGTH"
Public Const CGI_CONTENT_TYPE  As String = "CONTENT_TYPE"
Public Const CGI_GATEWAY_INTERFACE As String = "GATEWAY_INTERFACE"
Public Const CGI_HTTP_ACCEPT   As String = "HTTP_ACCEPT"
Public Const CGI_HTTP_REFERER  As String = "HTTP_REFERER"
Public Const CGI_QUERY_STRING  As String = "QUERY_STRING"
Public Const CGI_REMOTE_ADDR   As String = "REMOTE_ADDR"
Public Const CGI_REMOTE_HOST   As String = "REMOTE_HOST"
Public Const CGI_REMOTE_USER   As String = "REMOTE_USER"
Public Const CGI_REQUEST_METHOD As String = "REQUEST_METHOD"
Public Const CGI_SCRIPT_NAME   As String = "SCRIPT_NAME"
Public Const CGI_SERVER_NAME   As String = "SERVER_NAME"
Public Const CGI_SERVER_PROTOCOL As String = "SERVER_PROTOCOL"
```

Figura 1.104 Parte del código empleado para declarar las variables de entorno CGI.²⁸

²⁷ **Active Server Pages (ASP).**- Páginas de Servidor Activo, constituye un lenguaje de scripting desarrollado por Microsoft para generar en conjunto con el lenguaje HTML páginas web dinámicas sobre un servidor web IIS.

²⁸ Soporte de Microsoft: **Cómo escribir aplicaciones CGI en Visual Basic.** Tomado de: <http://support.microsoft.com/kb/239588/es>

Al igual que en los lenguajes anteriores se tiene que crear un documento HTML con un formulario que permita la ejecución del programa desarrollado en Visual Basic.

```
<HTML>
<HEAD>
<TITLE>Testing VB CGI</TITLE>
</HEAD>
<BODY>
<FORM action="/cgi-bin/hello.exe" method="POST">
<INPUT TYPE="TEXT" NAME="Name"> Name<BR>
<INPUT TYPE="SUBMIT">
</FORM>
</BODY>
</HTML>
```

Figura 1.105 Código HTML para emplear el CGI desarrollado en visual.²⁹

1.7.5 PERL

Perl es el lenguaje de programación más utilizado para desarrollar aplicaciones CGI en la web, constituye un lenguaje interpretado, es decir que su ejecución se la realiza a través de un intérprete y no necesita pasar por un proceso de compilación, a diferencia del lenguaje C/C++ que requiere ser compilado para obtener un archivo ejecutable. El intérprete puede ser instalado en cualquier plataforma para su ejecución, mientras que un lenguaje de compilación tiene que pasar por el proceso de compilado para cada una de las plataformas sobre las que se quiera obtener su ejecutable.

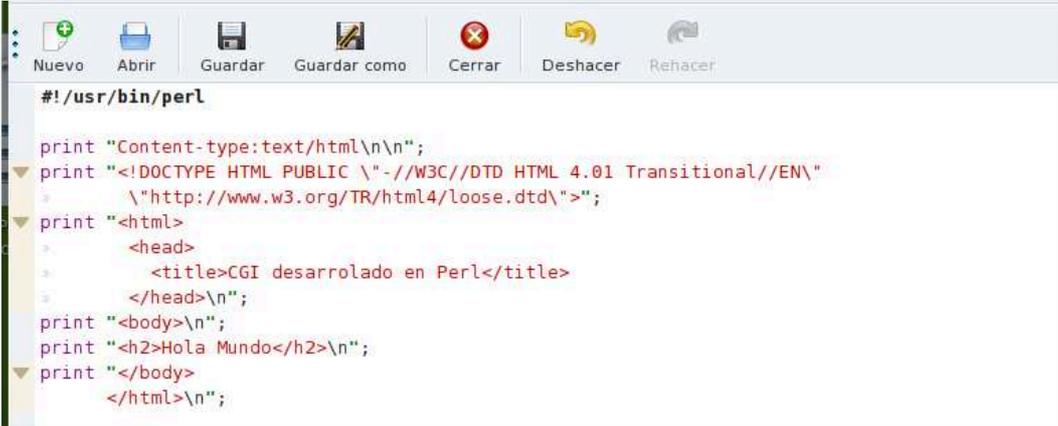
Perl permite manipular fácilmente el texto que proviene de la entrada estándar (stdin) ya que no requiere de un intermediario para el flujo de información como en el caso de Visual Basic, además posee un poderoso manejo de expresiones regulares que permiten manipular fácilmente el texto que obtiene de su entrada. Para el desarrollo de aplicaciones web, Perl permite que el lenguaje HTML sea

²⁹ Soporte de Microsoft: **Cómo escribir aplicaciones CGI en Visual Basic**. Tomado de: <http://support.microsoft.com/kb/239588/es>

embebido en su lenguaje, es decir que en un solo documento se puede trabajar con los dos lenguajes.

Perl trabaja principalmente sobre sistemas operativos Linux en conjunto con el servidor web Apache, debido a que este servidor es ampliamente utilizado en internet se han desarrollado un conjunto de aplicaciones web con Perl teniendo una evolución importante desde sus inicios, permitiendo en la actualidad realizar labores de administración sobre el sistema operativo con la ayuda de un CGI.

El desarrollo de un CGI a manera de ejemplo es muy simple, a diferencia de los lenguajes mencionados en párrafos anteriores, un CGI básico se halla estructurado de la siguiente manera:

A screenshot of a text editor window showing a Perl CGI script. The window has a menu bar with options: Nuevo, Abrir, Guardar, Guardar como, Cerrar, Deshacer, and Rehacer. The script content is as follows:

```
#!/usr/bin/perl
print "Content-type:text/html\n\n";
print "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\"
      \"http://www.w3.org/TR/html4/loose.dtd\">";
print "<html>
      <head>
      <title>CGI desarrollado en Perl</title>
      </head>\n";
print "<body>\n";
print "<h2>Hola Mundo</h2>\n";
print "</body>
      </html>\n";
```

Figura 1.106 CGI desarrollado en Perl.

- La primera línea se encarga de determinar el lugar donde se encuentra el intérprete Perl que ejecutará el script que viene en las siguientes líneas.
- La segunda línea se encarga de especificar el tipo de datos que va a recibir el navegador web, para este caso se trata de una página HTML, aunque también puede tratarse de otro tipo de datos como texto, imágenes, etc.
- Las siguientes líneas se encargan de desarrollar el contenido de la página web empleando todos los elementos necesarios.

Hay que tomar en cuenta que Perl emplea la función print para mostrar la información en pantalla y debe ser empleada para cada elemento HTML que se quiera mostrar dentro de una página web dinámica (CGI).

Ahora si se necesita emplear varias líneas de código HTML, el ingreso de la función print en cada línea HTML resulta molesto, para esto Perl cuenta con funciones adicionales que permiten eliminar la inserción de la función print en cada línea HTML.



Figura 1.107 Visualización de CGI desarrollado en Perl.

A diferencia de los lenguajes de programación mencionados, Perl se ha mantenido como el lenguaje principal para el desarrollo de páginas web dinámicas empleando CGI, en la web se puede encontrar cientos de ejemplos y proyectos desarrollados en Perl para aplicaciones web, pero no es el único lenguaje ya que existen un conjunto de lenguajes que han sido desarrollados para generar páginas web dinámicas.

1.8 DEFINICIÓN DE UNA RED TIPO SOHO

Las intranets no solo son útiles para grandes empresas, las pequeñas empresas como por ejemplo una red tipo SOHO (Small Office-Home Office o Pequeña Oficina, Oficina en Casa) puede también utilizar una intranet, de modo que sus usuarios se beneficien de los servicios disponibles dentro de ésta. Una red tipo SOHO está constituida por un número de entre 2 a 20 usuarios.

Encuestadores, consultores, abogados y asesores profesionales son ejemplos de empresas de este tipo. Este concepto también puede adaptarse por ejemplo al ámbito educativo, específicamente a laboratorios en los que se cuente con la cantidad necesaria de máquinas para crear una red de este tipo.

1.8.1 REQUERIMIENTOS EN UNA RED TIPO SOHO

Debido a que las redes tipo SOHO cuentan con un número limitado de usuarios y recursos, demandarán el uso de ciertos servicios que se ajusten a sus necesidades, por lo que a continuación se describen algunos de los factores que se deben tomar en cuenta al momento de implementar tanto la red como los servicios que éste tipo de red requiere:

- **Disponibilidad y costo de los equipos:** En muchos casos el presupuesto para conectar un conjunto de equipos en una red tipo SOHO, que permita compartir un conjunto de servicios disponibles para los usuarios de la red es limitado.
- **Disponibilidad de acceso a Internet:** La configuración de la red puede limitarse debido a las opciones disponibles para conectarse a internet. Una red tipo SOHO generalmente cuenta con un solo enlace de acceso a Internet a través de un gateway residencial, por lo que se hace necesario el uso de un servicio que permita proveer el acceso a Internet a todos los usuarios que forman parte de la red. Esto se puede lograr a través de un servicio denominado proxy.
- **Simplicidad de la configuración:** La mayoría de redes de empresas pequeñas, no son administradas por un departamento específico. La configuración elegida para la red debe coincidir con los recursos disponibles para su instalación y mantenimiento. La instalación de un servidor que provea los servicios a todos los usuarios de la red facilitará el trabajo de la persona que se encargará de la administración y mantenimiento de la misma.

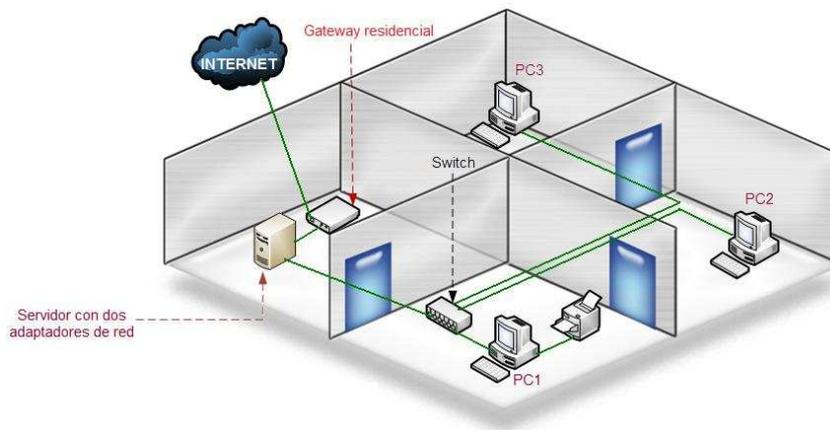


Figura 1.108 Uso de un servidor centralizado.³⁰

- **Acceso de los usuarios a los servicios:** Se deberá elegir un mecanismo que facilite la comunicación entre las estaciones de trabajo de la red con los servicios disponibles a través de un servidor, el uso de un servicio como dhcp facilitará éste proceso, ya que proveerá la información necesaria para que las estaciones de trabajo puedan formar parte de la red y de ésta forma acceder a los servicios proporcionados dentro de la misma.
- **Políticas de acceso a Internet:** Al contar una red SOHO con un solo enlace a Internet con un ancho de banda limitado, el acceso a contenidos web que provoquen distracción en el personal de la empresa, consumirá ancho de banda innecesario, por lo que se requerirá la aplicación de políticas que limiten el acceso a Internet, obteniendo como resultado un acceso a Internet eficiente y un mejor desempeño por parte de los empleados de la empresa.
- **Seguridad para la conexión a Internet:** La red SOHO debe estar protegida, ya que puede estar expuesta a posibles ataques debido al uso inadecuado de internet. Esto se lo puede realizar, mediante el uso de un firewall, que puede residir en el servidor de la red.
- **Servicios que permiten crear, compartir y administrar la información:** Una red debe facilitar la búsqueda y administración de documentos que los usuarios crean para ser compartidos dentro de ella, de ésta manera los

³⁰ Microsoft TechNet: **Home and Small Office Network Topologies.**
Tomado de: <http://technet.microsoft.com/en-us/library/bb457037.aspx>

usuarios pueden tener acceso a información actualizada de cualquier documento compartido. Para facilitar este procedimiento se puede utilizar un servicio que permita crear directorios en los cuales los usuarios podrán publicar y administrar de forma centralizada sus documentos.

- **Cliente universal:** Un concepto importante dentro de una red es el concepto de cliente universal, que es aquel que puede acceder a todos los servicios de una red, independiente del sistema operativo y del tipo de máquina que utilice.

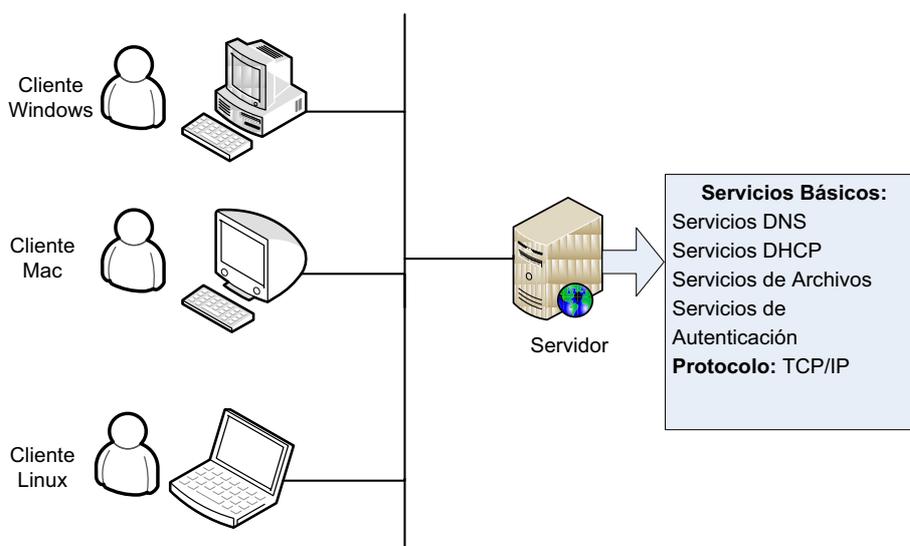


Figura 1.109 Cliente Universal.

En base a los factores analizados anteriormente la perspectiva de una red tipo SOHO a administrar quedaría estructurada por un servidor centralizado en donde se instalarán los servicios mencionados.

1.9 METODOLOGÍAS DE DESARROLLO

Existe un amplio conjunto de metodologías para el desarrollo de software algunas de estas han sido analizadas, aplicadas y posteriormente adaptadas universalmente, para el caso del desarrollo de aplicaciones web la realidad es distinta, si bien se han propuesto varias metodologías tales como MIDAS³¹,

³¹ **Metodología Interactiva de Desarrollo de Aplicaciones Software (MIDAS).**- Permite el desarrollo de sistemas Web.

EORM³², RMM³³, OOHDM³⁴ entre otras, su enfoque no está destinado a todas las posibilidades de desarrollo sobre un documento web, como se mencionó en la sección 1.1.2 existen dos tipos de documentos web destinados a realizar tareas específicas.

Tanto un documento estático, como un dinámico proporcionan información visual al cliente, la diferencia radica al momento de crear una interacción entre la aplicación web y el cliente (documentos dinámicos), los documentos web dinámicos utilizan esta interacción para realizar distintos procesos, dichos procesos van a ser ejecutados a través de distintas funciones creadas por el desarrollador sumando a esto la parte visual del documento web, es en esta parte medular donde las metodologías mencionadas anteriormente no prestan especial atención, es decir, ponen énfasis en la navegabilidad del usuario y la apariencia visual de la aplicación, mas no en los procesos para generación de funciones que realicen tareas específicas.

Para el presente proyecto la parte medular constituye la generación de varias funciones que permitan la interacción entre el cliente y la aplicación web para modificar los archivos de configuración de los servicios presentes en el servidor, mas no para acceder a algún tipo de base de datos, por lo que las metodologías mencionadas no encajan con el enfoque dado al proyecto, lo que hace necesario la utilización de una metodología de desarrollo de software que se adapte al proyecto a desarrollar, sin dejar de lado los procesos de diseño y navegabilidad con los que debe contar la aplicación web para la posterior interacción con el usuario.

A continuación se van a definir dos grupos principales de metodologías de desarrollo de software: las tradicionales y las ágiles, posterior a ello se analizará

³² **Enhanced Object Relationship Methodology (EORM).**- Metodología de Relación entre Objeto, permite el modelado orientado a objetos mediante las relaciones entre objetos.

³³ **Relationship Management Methodology (RMM).**- Metodología de Administración de Relaciones, es un proceso que permite el análisis, diseño y desarrollo de aplicaciones hipermedia.

³⁴ **Object-Oriented Hypermedia Design Method (OOHDM).**- Método de Diseño de Desarrollo en Hipermedia Orientado a Objetos, ésta metodología permite crear aplicaciones multimedia de forma simple y eficaz.

que grupo encaja con el prototipo, para dentro del grupo de metodologías escogido seleccionar la metodología final a emplear para el desarrollo del mismo, cabe recalcar que la metodología a ser seleccionada será adaptada al proyecto en las distintas fases que esta pueda contener.

1.9.1 METODOLOGÍAS TRADICIONALES

Las metodologías tradicionales han sido desarrolladas como herramientas para trabajar sobre proyectos que van a pasar por un proceso amplio de planificación, en esta planificación se detallarán todos los aspectos a tomar en cuenta para posterior a ello realizar el proceso de desarrollo de software, los tiempos de ejecución y costo al aplicar metodologías tradicionales es elevado ya que todo ha sido calculado previamente, al haberse realizado una planificación exhaustiva este tipo de metodologías ponen resistencia a cambios en el proceso de desarrollo ya que se asume que todos los detalles han sido tomados en cuenta. Otro aspecto importante en esta metodología es el establecimiento de controles para los distintos procesos a través de herramientas, notaciones, artefactos³⁵, actividades, documentación detallada con el fin de obtener un software más eficiente.

El enfoque de las metodologías tradicionales esta dado por una planificación exhaustiva, así como un control detallado de los distintos procesos para posterior a ello iniciar el desarrollo, este tipo de metodologías son empleadas para grandes proyectos de desarrollo donde se espera obtener un software que explique al detalle todos los procesos realizados al cliente. Entre los diferentes tipos de metodologías tradicionales se tiene:

- RUP(Rational Unified Process o Proceso Unificado Racional)
- MSF(Microsoft Solutions Framework)
- FLIP(Fusebox Lifecycle Process)

³⁵ **Artefacto.-** Constituye un entregable o producto obtenido luego de realizar el proceso de desarrollo.

1.9.2 METODOLOGÍAS ÁGILES

En la actualidad el desarrollo de software ha tomado una velocidad incremental que requiere respuestas inmediatas a un problema planteado por un cliente en tiempos cortos y a bajo costo, que satisfaga las necesidades que ha planteado el cliente, es decir se necesita que el cliente pase a formar parte del grupo de desarrollo de software con la finalidad de agilizar los procesos y obtener la solución más aproximada a lo deseado, de la misma manera se requiere que el software al estar en las distintas etapas de desarrollo se adapte a las nuevas problemáticas que el cliente pueda encontrar mientras observa el desarrollo del mismo; al cliente le interesa obtener artefactos que le proporcionen una idea o avances acerca del software, es decir el grupo de desarrollo proporciona al cliente avances del software para que pueda verificar el funcionamiento y proporcione inquietudes e inconvenientes con el avance proporcionado, lo que permite facilitar el desarrollo ya que los desarrolladores se enfocan en resolver los problemas en cada avance, a estos avances en las metodologías ágiles se los denomina iteraciones, cada iteración permitirá al desarrollador obtener una visión clara de los problemas planteados por el cliente y en base a estos proporcionar la respectiva solución, esta es una de las diferencias claras con las metodologías tradicionales donde la planificación se la realiza al inicio previo al desarrollo y no se admiten fácilmente cambios sobre la planificación realizada. El enfoque principal de las metodologías ágiles es la satisfacción del cliente al obtener un software muy aproximado a la solución del problema planteado, en tiempos cortos, bajo costo, sin necesidad de obtener una documentación detallada del proceso. A continuación se mencionan las metodologías ágiles más empleadas:

- SCRUM
- Crystal Clear
- XP (Xtreme Programming)
- ASD(Adaptive Software Development)

1.9.3 DIFERENCIA ENTRE METODOLOGÍAS

En la tabla 1.8 se proporciona una comparativa entre metodologías, con la finalidad de obtener una visión más clara de los principales aspectos que diferencian a una metodología de la otra, para de esta manera seleccionar aquel grupo que se aproxima al ambiente de desarrollo del presente proyecto.

Metodologías tradicionales	Metodologías ágiles
Se basa en estándares y normas	Se basa en la experiencia e inventiva obtenida en desarrollos previos.
Existe un exhaustivo control en todos los procesos involucrados en el desarrollo.	Existe un menor control de procesos ya que se busca obtener una solución que satisfaga al cliente.
El cliente interactúa con los desarrolladores mediante reuniones	El cliente pasa a formar parte del equipo de desarrollo.
Grupos de trabajo de desarrollo grandes por lo general distribuidos en distintas áreas.	Grupos de trabajo pequeños.
Existe prioridad sobre los procesos y documentación.	Existe prioridad sobre la interacción con las personas involucradas, se proporciona documentación básica.
Arquitectura de software esencial obtenida en el proceso de planificación inicial.	Menor énfasis en la arquitectura se adapta a los requerimientos durante el proceso de desarrollo.
Sigue estrictamente las fases establecidas en la planificación inicial.	Se permiten cambios en el proceso de desarrollo con la finalidad de mejorar la calidad de la solución final.
Más artefactos ya que se tiene que manipular todos los procesos que han sido previamente planificados.	Pocos artefactos, debido a que se limitan a desarrollar las soluciones a las problemáticas continuas generadas por el cliente.

Tabla 1.8 Diferencia entre metodologías tradicionales y ágiles.

1.9.4 SELECCIÓN DEL GRUPO DE METODOLOGÍA

Un punto básico al momento de desarrollar una aplicación web a través de una metodología es la necesidad de una verificación continua del desarrollo, mediante la creación de artefactos previos a la obtención final de la aplicación, con la finalidad de obtener un producto que se adapte a la visión de solución que tuvo el cliente, es decir, se requiere que el cliente forme parte del desarrollo para verificar continuamente los artefactos generados y dar una aprobación a la solución o verificar si existen puntos de falla que van a ser posteriormente resueltos por el grupo de desarrollo. El tiempo de desarrollo para una aplicación web debe ser corto, ya que se necesita que el producto se encuentre a disponibilidad a corto plazo y con la posibilidad que el cliente pueda contar con módulos previos en funcionamiento que verifiquen el avance del trabajo en desarrollo, dado que el tiempo empleado para el presente proyecto es corto y su tamaño relativamente pequeño, la adaptabilidad de este a una metodología ágil es clara, además, el número de personas involucradas en este proceso es de 2, es decir no se cuenta con un grupo amplio, lo cual lo direcciona hacia metodologías ágiles; al desarrollar una aplicación cambiante en el tiempo los requisitos en cuanto a la arquitectura de software variarán, debido a que se puede requerir en el transcurso del desarrollo diferentes lenguajes o herramientas para obtener una solución al problema, por esta razón la definición de una arquitectura inicial que permanezca sin modificaciones no es aplicable para el presente proyecto, este es otro punto adicional que enfoca el proyecto hacia las metodologías ágiles. Otro punto importante constituye el proceso de documentación, el tiempo de desarrollo como se mencionó para el proyecto es corto y lo que se busca es obtener la solución final sin necesidad de generar extensos documentos acerca del desarrollo, dado el tiempo disponible, lo que se busca es generar la solución con documentación básica y concisa acerca del desarrollo. Con todas las observaciones mencionadas se puede apreciar claramente que el presente proyecto se enfoca hacia las metodologías ágiles pues cumple con la mayoría de requerimientos planteados en la tabla 1.8. A continuación luego de haber determinado el grupo de metodologías al cual se direcciona el proyecto, se procederá a determinar, dentro del grupo de metodologías, la que se adapta de manera específica al proyecto.

1.9.5 TIPOS DE METODOLOGÍAS ÁGILES

En esta sección se va a realizar un desarrollo de las distintas metodologías ágiles mencionadas en la sección 1.9.2 para en base a ello determinar la metodología ágil a ser aplicada en el presente proyecto.

1.9.5.1 Scrum

Scrum es una metodología incremental e iterativa, es óptima para trabajar con un equipo compuesto por 7 (± 2) personas, cada persona debe contar con todas las habilidades necesarias para desarrollar el producto (programador, diseñador, etc). Se enfoca en el trabajo organizacional para obtener un producto con óptima calidad, la documentación no es necesaria para poner en ejecución el proyecto, esta puede ser generada al transcurso del desarrollo, lo que lo diferencia de otras metodologías donde la documentación es imprescindible. Otra idea principal de scrum es poner al proyecto en ejecución desde el primer instante, para obtener de manera rápida resultados que puedan ser apreciados por el cliente, con la finalidad de verificar el avance del proyecto y satisfacer al cliente con lo que está haciendo. Al formar parte del equipo de desarrollo el cliente puede plantear sus dudas para que los desarrolladores puedan obtener de manera inmediata una respuesta al problema planteado, el interés se enfoca en la pronta respuesta al cliente.

1.9.5.1.1 Roles en Scrum

- **ProductOwner (propietario del producto):** Es la persona interesada en el resultado del producto final.
- **Scrum Master (Facilitador):** Es el líder del equipo de desarrollo, se encarga de verificar que se cumplan todas las reglas, normas y procesos de scrum, generando un ambiente de trabajo óptimo para todo el equipo durante el transcurso del proyecto.
- **Team (Equipo):** Conjunto de personas (7 ± 2) que se encargan del desarrollo del producto. Con un número menor a 5 la falta de algún miembro compromete el desarrollo normal del proyecto. Con un número mayor a 9 se tiende a perder comunicación y colaboración entre personas.

1.9.5.1.2 Artefactos o Documentos en Scrum

- **Product Backlog (Lista de requisitos priorizada):** Representa las expectativas que tiene el cliente respecto a los objetivos y entregables del proyecto. Es realizado por el cliente con la ayuda del resto de roles.
- **Sprint Backlog (Lista de tareas de la iteración):** En base al Product Backlog se genera la lista de tareas que el equipo realizará durante la iteración para alcanzar los objetivos y obtener los entregables para mostrar al cliente al final de la iteración.
- **Burndown charts (Gráficos de trabajo pendiente):** Es un gráfico que permite determinar la velocidad a la cual se está progresando en el proceso de alcanzar objetivos y obtener entregables, para saber si se completará todas las tareas en el tiempo estimado.

La metodología Scrum cuenta con un conjunto de actividades que se realizan en el transcurso del desarrollo del proyecto, la principal actividad la constituye **sprint**, esta actividad no es más que la ejecución de la iteración, una iteración es un espacio de tiempo corto y fijo para la ejecución de las tareas asignadas en sprint backlog generalmente de un mes o hasta 2 semanas que permitan cumplir los objetivos y obtener un producto entregable para el usuario.

1.9.5.2 Crystal Clear

Crystal Clear corresponde a una metodología dentro de un grupo denominado *Crystal*, las metodologías crystal se clasifican de acuerdo a dos características importantes que tienen los proyectos: tamaño y complejidad. De esta manera se tiene:

- Clear para equipos de hasta 8 personas.
- Amarillo para equipos entre 10 a 20 personas.
- Naranja para equipos entre 20 a 50 personas.
- Roja para equipos entre 50 a 100 personas.
- Azul para equipos entre 100 a 200 personas.

El presente proyecto será desarrollado por 2 personas por lo que el tipo de metodología que se describirá a continuación corresponde a *crystal clear*.

Crystal clear tiene un enfoque al trabajo en equipo, sin exhaustiva documentación y con orientación al desarrollo de más entregables que puedan comprobar el funcionamiento del producto en desarrollo. La documentación es opcional, mientras que el desarrollo es un hecho. Otro factor importante dentro de esta metodología es la comunicación, ya que de esta manera se permite obtener interrogantes y resolver dudas de manera rápida y si a esto se suma la inclusión del cliente dentro del desarrollo el resultado puede ser más óptimo.

1.9.5.2.1 Roles en Crystal Clear

La metodología *Crystal Clear* cuenta con 8 roles destinados a tareas específicas:

- **Patrocinador:** Consigue los recursos y define la totalidad del proyecto.
- **Usuario Experto:** Se encarga de generar la Lista de Actores/Objetivos, así como también el documento de Casos de Uso y Requerimientos. Trabaja en conjunto con el experto en negocio.
- **Diseñador Principal:** Se encarga de generar la arquitectura, es una persona con amplio conocimiento en lo referente a las distintas áreas donde se va a enfocar el proyecto. Diseño, programación, etc.
- **Diseñador Programador:** Trabaja en conjunto con el diseñador principal para crear los distintos diseños de interfaz, diagramas de flujo, códigos de desarrollo, las pruebas y el sistema final. Un diseñador que no sepa programar no es aceptado en esta metodología.
- **Experto en Negocios:** Como se mencionó trabaja en conjunto con el usuario experto, debe contar con un conocimiento acerca de las reglas y políticas del negocio.
- **Coordinador:** Se encarga de desarrollar los distintos artefactos del proyecto por ejemplo: el Plan de Entrega, el Plan y Estado de Iteración.

- **Verificador:** Se encarga de verificar los errores encontrados en el proceso de desarrollo para informar al equipo.
- **Equipo:** Es el encargado del proceso de desarrollo, resolver los problemas planteados y alcanzar los objetivos.

1.9.5.2.2 *Propiedades o valores de Crystal Clear*

- **Entregas frecuentes:** Como su nombre lo indica se tiene que realizar una entrega frecuente de la evolución en el desarrollo del proyecto al cliente, esta puede ser diaria, semanal o mensual depende del tamaño del proyecto.
- **Comunicación:** La comunicación es un elemento importante en el desarrollo del proyecto, esta se debe priorizar entre los diferentes roles involucrados en el proyecto.
- **Mejoras reflexivas:** Se deben realizar charlas periódicas durante las distintas iteraciones para compartir experiencias y dificultades durante el desarrollo.
- **Seguridad personal:** Discutir con los miembros del equipo cuando algún problema ocurre dentro de este.
- **Concentración:** Tener claro lo que se está desarrollando, en un tiempo y ambiente adecuado.
- **Fácil acceso a usuarios claves:** Las personas dentro del equipo deben tener cotidiana comunicación con expertos y deben permitir que el usuario forme parte del equipo para encontrar errores tempranamente.

1.9.5.3 **XP (eXtreme Programming o Programación Extrema)**

XP es una metodología enfocada principalmente a la satisfacción del cliente, es decir se espera generar un proyecto en el menor tiempo posible y a bajo costo, para ello *xp* genera al igual que las metodologías mencionadas anteriormente avances del proyecto periódicamente, para que el cliente pueda verificar el funcionamiento del mismo, esto se logra al integrar en el equipo de desarrollo al

cliente con la finalidad de obtener problemas y generar soluciones de manera inmediata. *Xp* acepta cambios en los requisitos en el transcurso del desarrollo del proyecto por lo que una comunicación clara entre el personal de desarrollo y el cliente constituye otro punto clave en la metodología pues permitirá desarrollar un óptimo resultado. La metodología ha sido diseñada para trabajar con un grupo pequeño de desarrolladores con la finalidad de evitar al máximo perder la comunicación entre ellos y distorsionar la búsqueda de soluciones al problema que es el objetivo principal. Se prefiere en la metodología obtener un software que funcione a una exhaustiva documentación, se creará documentación solo cuando se necesite tomar una decisión importante.

1.9.5.3.1 Características o valores de la metodología XP

- **Comunicación:** La comunicación cumple un papel importante en la metodología ya que permite obtener ideas, plantear problemas y obtener soluciones de manera rápida al proyecto en desarrollo.
- **Simplicidad:** El diseño y codificación del software debe ser simple y claro, centrado únicamente en los requerimientos planteados por el cliente sin enfocarse en cosas innecesarias.
- **Realimentación (feedback):** Se muestra al cliente continuos avances en el desarrollo del proyecto para conseguir un software acorde a sus necesidades, de esta manera el equipo de desarrollo puede realizar modificaciones continuas o retroceder en el caso necesario cuando el resultado no es aceptado por el cliente.
- **Coraje:** Esta propiedad describe la valentía que debe tener cada individuo dentro del equipo de desarrollo para cumplir con los 3 puntos anteriores, valentía para despejar todas las dudas con el cliente, crear diseños simples, y confiar que la realimentación es efectiva.
- **Respeto:** Debe existir un ambiente de respeto entre todos los miembros del equipo de desarrollo, el cual se manifiesta de varias formas, por ejemplo, cada miembro respeta el trabajo del resto, sin hacer de menos a

otros, más bien se debe apoyar al resto ya que el trabajo final constituye la unión del trabajo realizado por cada miembro.

1.9.5.3.2 Fases en la metodología XP

La metodología XP (programación extrema) define 4 fases en el proceso de desarrollo:

1. Planificación.
2. Diseño.
3. Codificación.
4. Pruebas.

Primera fase: planificación del proyecto

- ***Historia de usuarios:*** Tiene la misma finalidad que los casos de uso. La historia de usuarios es desarrollada con el cliente. Emplea lenguaje natural no técnico, en ésta historia se definen las funcionalidades que va a tener cada módulo del proyecto, es decir las tareas que se van a realizar para generar un avance en el proyecto.
- ***Planificación de entregables:*** En base a la información definida en la historia de usuarios, este plan determinará: el número de historias de usuario que se crearán para cada versión de software entregable así como su respectiva fecha de publicación. El tiempo de desarrollo de una historia está entre 1 y 3 semanas.
- ***Iteraciones:*** Una iteración tienen una duración de aproximadamente 3 semanas y consiste en desarrollar las tareas mencionadas en las historias de usuario, que han sido seleccionadas en la etapa de planificación de entregables por el cliente para su implementación. Si no se obtuvieron resultados favorables en una iteración previa se tendrá entre 1 a 3 días para resolver estos inconvenientes.
- ***Velocidad del proyecto:*** corresponde a la rapidez con la que se está desarrollando el proyecto, para obtener un valor estimativo basta con

contabilizar el número de historias de usuarios que han sido implementadas en una iteración, de esta manera se puede realizar un control para determinar si se está cumpliendo con las tareas durante el tiempo estimado o se necesita replantear un nuevo plan de entregables.

- **Programación en pareja:** Un punto importante dentro de la metodología XP es la programación en pareja, con ello se busca obtener un óptimo diseño y calidad en el producto en desarrollo, ya que mientras un programador genera código, el otro puede proporcionar ideas que simplifiquen el proceso de manera similar en el caso del diseño, de esta manera el producto final será de calidad.
- **Reuniones diarias:** Es conveniente realizar reuniones con todas las personas involucradas en el proceso de desarrollo con la finalidad de exponer problemas e inquietudes para buscar de manera conjunta soluciones rápidas, todas las personas en la reunión pueden dar su punto de vista acerca de la discusión planteada.

Segunda fase: Diseño

- **Diseños simples:** La metodología sugiere que los diseños sean lo más simples y sencillos posibles con la finalidad de disminuir el tiempo y esfuerzo al desarrollar la solución.
- **Glosario de términos:** Se debe realizar una adecuada descripción de los métodos que se emplearán para el diseño así como un adecuado glosario de términos con el fin de facilitar el desarrollo.
- **Riesgos:** Si surgen problemas durante el proceso de diseño, los desarrolladores asignados deberán investigar y reducir al máximo el riesgo encontrado.
- **Funcionalidad extra:** No se debe crear nada más ni nada menos que la solución a los problemas planteados por el cliente. El añadir funcionalidades extras aumenta el trabajo y pérdida de tiempo.

- **Refactorizar:** Se puede revisar el código implementado en búsqueda de posibles mejoras que faciliten la obtención de un resultado sin alterar su funcionalidad.
- **Tarjetas C.R.C (Clase, Responsabilidad, Colaboración):** Son tarjetas empleadas para la programación orientada a objetos. Cada tarjeta representa un objeto. Una tarjeta se halla estructurada de la siguiente manera: la clase del objeto se la escribe en la parte superior. Se crean dos columnas, en la columna de la izquierda se indican las responsabilidades y objetivos del objeto, en la derecha las clases que colaboran con el objeto.

Tercera fase: Codificación

Para que el proyecto cumpla con los objetivos planteados, la codificación debe realizarse siguiendo distintos estándares y normas que rigen sobre el lenguaje de desarrollo que se vaya a emplear. Al utilizar normas y estándares el código puede ser fácilmente entendible y adaptable a cambios en un intervalo de tiempo corto. La metodología sugiere crear un test previo antes de proceder a la programación, con la finalidad que el desarrollador se enfoque en la resolución del test y de esta manera resolver el problema planteado en la historia de usuario, una vez implementado el código este pasará sin problema por los distintos test presentados por el cliente ya que previamente fueron resueltos. La metodología también sugiere contar con un repositorio común donde cada pareja de desarrolladores almacenará su código junto al test respectivo de manera que el equipo de desarrollo se encuentre al día con las distintas modificaciones que se realicen sobre el proyecto, si alguien desea modificar el código lo puede hacer siempre y cuando su código cumpla con el test respectivo. El desarrollo colectivo también es permitido, de manera que todos los desarrolladores deben tener conocimiento acerca de las distintas áreas del proyecto. La optimización se la debe realizar al finalizar pues el objetivo de la codificación es obtener un código que funcione y sea correcto. La resolución de distintos test's siempre ayudará al desarrollador a encaminar su trabajo hacia una óptima respuesta al cliente.

Cuarta fase: Pruebas

Como se mencionó en la tercera fase de la metodología el uso de un test cumple un papel importante dentro del desarrollo. Si un código es almacenado en el repositorio sin su respectivo test este no podrá ser tomado en cuenta. Al almacenar un código junto con el test se asegura la propiedad de refactorización pues un desarrollador puede modificar el contenido pero no podrá variar su funcionalidad ya que debe cumplir con el test planteado. Existen dos tipos de pruebas: las de unidad y las de aceptación.

- **Pruebas de unidad:** Son aquellas pruebas creadas para verificar el funcionamiento de cada tarea dentro de una historia de usuario. Por ejemplo, verificar que el texto ingresado en un área de texto sea el adecuado.
- **Pruebas de aceptación:** Son pruebas definidas por el cliente para verificar el funcionamiento general de cada historia de usuario. El cliente determinará si se cumplió o no con la resolución de los problemas planteados en la historia. Por ejemplo, mediante un formulario se obtiene la información necesaria y esta es almacenada en una base de datos exitosamente.

1.9.5.4 ASD (Adaptive Software Development/Desarrollo de Software Adaptable)

La metodología *ASD* al igual que el resto de metodologías mencionadas en esta sección tiene como principal objetivo adaptarse al cambio continuo en el proceso de desarrollo. *ASD* se diferencia del resto de metodologías al no definir fases de planificación, diseño y desarrollo del software. *ASD* emplea sus denominados ciclos de vida, de esta manera se tienen 3 ciclos: especular, colaborar y aprender. La metodología también emplea procesos iterativos una característica de las metodologías ágiles, donde en cada iteración se producirán cambios y se generarán errores, la finalidad es adaptarse al cambio continuo en el proceso de desarrollo.

1.9.5.4.1 Ciclo de vida en *ASD*

- **Especular:** Corresponde al primer ciclo en la metodología, aquí se definen los distintos objetivos y metas del proyecto, en base a este análisis se definirán las distintas iteraciones necesarias para alcanzar los objetivos. Se

debe poner mayor énfasis en aquellas características que buscan resolver problemáticas planteadas exclusivamente por el cliente. Cada iteración puede ser repetida hasta que el cliente y el equipo de desarrollo se encuentren satisfechos con el resultado. En este ciclo también se define el tiempo necesario para alcanzar los objetivos, con la finalidad de que el equipo de desarrollo tenga claro el tiempo estimado y en base a este determinar las prioridades sobre las tareas a desarrollar.

- **Colaborar:** En este ciclo se procede a desarrollar cada una de las tareas planificadas en la etapa de especulación, cada grupo de desarrollo debe proporcionar al resto de desarrolladores las nuevas experiencias adquiridas en el proceso, para evitar de esta manera que se investigue sobre algo que ya ha sido solucionado, lo que permite ahorrar tiempo para utilizarlo en resolver nuevas problemáticas. Por esta razón la comunicación entre el equipo cumple un papel primordial dentro del ciclo.
- **Aprender:** El último ciclo dentro de la metodología tiene como finalidad capturar todo lo que se ha aprendido en el transcurso del desarrollo, sea esto bueno o malo, para obtener información que contribuya al éxito en la obtención final del producto. Se definen 4 tipos de aprendizaje:
 - **Calidad del producto desde el punto de vista del cliente:** Determina si se ha tenido éxito en el desarrollo mediante la satisfacción del cliente con el producto obtenido.
 - **Calidad del producto desde el punto de vista de los desarrolladores:** Se realiza un proceso de evaluación al producto con un enfoque técnico para determinar si se ha cumplido con los estándares y normas establecidas para el proceso de desarrollo.
 - **La gestión del rendimiento:** Es una evaluación empleada para determinar lo que se ha aprendido en el proceso, es decir, evaluar la información que se ha generado y ha servido de ayuda a otros grupos, así como la que se ha utilizado para resolver un problema que ya existía.

- **Situación del proyecto:** Determina las características iniciales para la siguiente iteración en base a la información obtenida en la iteración anterior.

1.9.6 COMPARACIÓN DE METODOLOGÍAS ÁGILES

	SCRUM	XP	ASD	Crystal Clear
Número de desarrolladores.	7 ± 2 personas.	2 personas.	10 personas.	8 personas
Iteraciones cortas.	Si	Si	Si	Si
El cliente forma parte del equipo.	Si/No	Si	No	Si/No
Documentación.	Opcional	No	No	Opcional
Simplicidad en el diseño.	Si	Si	Si	Si
Reuniones diarias.	Si	No	No	Si
Etapas de Planificación, diseño, desarrollo.	Si	Si	No	Si

Tabla 1.9 Comparación de metodologías ágiles.

Como se puede observar en la tabla 1.9 la diferencia entre las metodologías ágiles es mínima, la diferencia más clara radica en el número de desarrolladores involucrados en el proyecto. Este aspecto será de vital importancia al momento de determinar el tipo de metodología a implementar para el desarrollo de un proyecto. En la siguiente sección se describirá el porqué de la selección de una metodología para el desarrollo del presente proyecto, enfocándose en ciertos aspectos claves que encajan a la metodología seleccionada con el ambiente en el que se desarrollará el presente proyecto.

1.9.7 SELECCIÓN DE LA METODOLOGÍA ÁGIL A EMPLEAR

Se ha seleccionado una metodología ágil dado que el desarrollo de una interfaz web siempre estará sujeto a continuos cambios en el transcurso del proyecto. El cliente como parte del equipo de desarrollo es fundamental pues tanto el diseño como la funcionalidad de la interfaz web deben satisfacer sus expectativas, la generación de versiones previas que puedan ser revisadas por el cliente es otro aspecto que no puede ser excluido, esto permitirá al desarrollador obtener un producto final óptimo resolviendo todas las dudas e inquietudes realizadas por el cliente en cada versión proporcionada, la generación de cada versión se la debe realizar en cortos periodos de tiempo, se necesita planificar iteraciones cortas que permiten obtener rápidamente versiones del producto en desarrollo pues una interfaz web requiere estar en funcionamiento en el menor tiempo posible. La documentación para el presente proyecto no es exhaustiva pues lo que se pretende es obtener un producto funcional en un lapso de tiempo corto que satisfaga al cliente. El punto de vital importancia en la selección de la metodología es el número de desarrolladores involucrados en el proyecto pues permitirá enfocarse hacia una metodología en particular. El presente proyecto es desarrollado entre dos personas, de allí que la metodología que encaja con este aspecto es *XP*, el desarrollo en parejas permitirá mejorar el código y obtener soluciones simples que no afecten la funcionalidad que el cliente desea obtener. En lo referente al diseño se tendrá que desarrollar solo lo que el cliente solicita ni más ni menos, con la finalidad de no desperdiciar el tiempo en cosas que al cliente no le interesan. De esta manera se ha mencionado claramente el ambiente en el cual se desarrollará el presente proyecto.

Como conclusión final se puede apreciar que la metodología que encaja con el ambiente de desarrollo para el presente proyecto es *XP*, por lo que la metodología seleccionada es la mencionada, algo que hay que tomar en cuenta es que la metodología será adaptada al desarrollo de aplicaciones web, por lo que en algunas fases de la metodología no se seguirá estrictamente con lo mencionado en la sección 1.8.5.3.2.

Para la selección de la metodología se han tomado en cuenta todos los aspectos mencionados en la tabla 1.9 es decir:

- Uso de Iteraciones cortas.
- El cliente forma parte del equipo.
- El número de desarrolladores.
- Documentación.
- Diseño y funcionalidad.

De esta manera se concluye con la selección de la metodología de desarrollo.

CAPÍTULO 2

2 REQUERIMIENTOS HARDWARE Y SOFTWARE

2.1 ESPECIFICACIÓN DE REQUISITOS SEGÚN EL ESTÁNDAR IEEE 830-1998

Si bien el desarrollo software es una parte fundamental del presente proyecto, la especificación y análisis de los requerimientos lo son aún más, por éste motivo se ha tomado como referencia el estándar IEEE 830-1998 con el fin de obtener los requisitos necesarios para el desarrollo e implementación del presente proyecto, éste estándar presenta una serie de pasos a seguir, los cuales permiten detallar de forma clara uno a uno los requerimientos que facilitarán tanto el desarrollo como la implementación del proyecto, un aspecto importante de éste estándar, es que permite integrar los requerimientos del sistema desde un enfoque global en el que se incluye tanto al cliente, desarrollador y usuario.

2.1.1 INTRODUCCIÓN

En ésta sección se proporcionará de forma general la información necesaria acerca del ambiente en el cual se desarrollará el proyecto, éste estándar presenta un esquema que consta de varias subsecciones las cuales se detallan a continuación:

2.1.1.1 Propósito

La realización de éste documento tiene como finalidad especificar los requerimientos necesarios a seguir para el desarrollo de una interfaz Web que permita administrar servicios que se encuentran instalados en el sistema operativo Linux, cuya administración se la realizará de manera remota mediante un explorador Web, como proyecto previo para la obtención del título de Ingeniero en Electrónica y Redes de Información.

Este documento está dirigido al Director del presente proyecto de titulación, los miembros que formarán parte del tribunal y para aquellas personas que se encuentren interesadas para que obtengan información acerca del desarrollo e implementación del proyecto.

2.1.1.2 Ámbito del Sistema

El proyecto tiene por nombre DAMA, debido a que representa a las iniciales de los nombres de los desarrolladores del proyecto.

La interfaz Web permitirá administrar remotamente desde un navegador web los servicios de firewall, filtrado de contenidos, DHCP, compartir directorios, verificar el estado del sistema, visualizar gráficas que muestran el consumo de CPU y memoria del servidor, así como el tráfico de red tanto de entrada como de salida. El desarrollo de la interfaz Web está orientado a empresas pequeñas, las cuales cuentan con un máximo de 20 usuarios, por éste motivo las características de los servicios se han limitado a éste ambiente, es decir se han tomado en cuenta solo aquellos requerimientos necesarios en cada servicio para satisfacer las necesidades de éste tipo de empresas.

Los beneficios, objetivos y metas que se obtendrán con el desarrollo de éste sistema son:

Beneficios:

- El webmaster o administrador de la red podrá administrar los servicios instalados en el servidor de manera remota mediante una Interfaz Web.
- Se facilitará la administración de los servicios instalados en un servidor Linux.
- Obtener un control adecuado sobre los usuarios y los contenidos Web a los que éstos puedan acceder, además se facilitará la creación de directorios compartidos para el almacenamiento de información.

Objetivos y Metas:

- Facilitar la administración de servicios en una red tipo SOHO.
- Crear una interfaz Web amigable para el administrador de la red de éste tipo de empresas.
- Satisfacer las necesidades de administración rápida y sencilla requeridas por el cliente.

2.1.1.3 Definiciones, Acrónimos y Abreviaturas

La mayoría de palabras a las cuales se hará referencia en éste documento, ya han sido definidas tanto en el capítulo 1 como en el capítulo 2 en sus respectivas secciones, en ese caso únicamente se indicará la sección en la que se encuentran definidas.

Definiciones

- **Site.-** En éste documento al hablar de site, se hace referencia a un sitio web que está compuesto por un conjunto de páginas web que se encuentran organizadas de forma jerárquica.
- **Dirección IP.-** Es una serie de dígitos que permiten identificar de manera lógica a un dispositivo dentro de una red que haga uso del protocolo IP (Internet Protocol o Protocolo de Internet) correspondiente a la capa 3 del modelo de referencia OSI, ésta dirección dentro de la red podrá ser asignada al dispositivo de forma dinámica o estática.
- **Firewall.-** Conocido también con el nombre de cortafuegos, es un software que permite analizar los paquetes de datos tanto del tráfico de entrada como de salida, permitiendo o denegando el acceso a las diferentes aplicaciones que intercambien paquetes a través de una red como por ejemplo Internet.
- **MIME.-** Se encuentra definido en el capítulo 1 sección 1.1.1.2.

Acrónimos

- **IEEE.-** Institute of Electrical and Electronics Engineers o Instituto de Ingenieros Eléctricos y Electrónicos.

Abreviaturas

2.1.1.4 Referencias

[1] Especificación de Requisitos según el estándar de IEEE 830-1998.

<http://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf>

[2] MONFERRER A., Raúl: Especificación de Requisitos según el estándar de IEEE 830-1998.

<http://www.eqaula.org/eva/file.php/1074/ERS.pdf>

[3] Estándar 830-1998 IEEE Especificaciones de Requerimientos del Sistema.

[eqaula.org/eva/file.php/1074/Estandar_830-IEEE.ppt](http://www.eqaula.org/eva/file.php/1074/Estandar_830-IEEE.ppt)

[4] MEDINA, Jorge Armando - SÁNCHEZ M., Alejandro: Requerimientos hardware Squid

http://www.tuxjm.net/docs/Manual_de_Instalacion_de_Servidor_Proxy_Web_con_Ubuntu_Server_y_Squid/html-onechunk/

[5] Linux para todos: Requerimientos hardware Squid

<http://www.linuxparatodos.net/portal/staticpages/index.php?page=servidor-proxy>

[6] H., John: Requerimientos hardware Samba

<http://www.samba.org/samba/docs/man/Samba-Guide/secure.html#ch4bsc>

[7] Requerimientos hardware Dansguardian

<http://dansguardian.org/?page=requirements>

[8] Linux para todos: Requerimientos hardware DHCPD

<http://www.linuxparatodos.net/portal/staticpages/index.php?page=servidor-dhcp>

[9] Canaima: Requerimientos hardware DHCPD

http://wiki.canaima.softwarelibre.gob.ve/wiki/index.php/Servidor_DHCP#.Requerimientos_m.C3.ADnimo_de_hardware_y_software

[10] Solución empresarial: Requerimientos hardware Shorewall

http://www.solucionempresarial.com.ar/inicio/index.php?option=com_content&view=article&id=51:servidorshorewall&catid=43:solucionescorporativas&Itemid=74

2.1.1.5 Visión General de Documento

El documento se ha dividido según lo especificado en el estándar IEEE 830, en las siguientes secciones: *introducción*, en ésta sección se detalla de manera general las características del proyecto a desarrollarse, *descripción general* en ésta sección se describirán todos los factores que afectan al sistema tales como: perspectiva del producto, las funciones del proyecto, características de las personas que manipularán el proyecto, las restricciones que presenta el sistema, dependencias y los requisitos futuros, *requisitos específicos* en ésta sección se detallarán todos los requisitos suficientes, para que los desarrolladores puedan planificar y generar las pruebas necesarias para verificar que el sistema funciona adecuadamente según las necesidades expuestas por el usuario.

2.1.2 DESCRIPCIÓN GENERAL

2.1.2.1 Perspectiva del Producto

El sistema DAMA es un proyecto independiente, es decir que no trabaja conjuntamente con otros sistemas.

En ésta sección se realizará un listado del software, así como la descripción de las características del hardware requerido, en lo referente al software la información será detallada ampliamente en anexos, por lo que únicamente se indicarán las secciones en las cuales se pueden encontrar las respectivas definiciones citadas en el listado.

DAMA es una aplicación Web que para su desarrollo, implementación y funcionamiento requiere del siguiente software:

	Servicio	Software	Referencia
DESARROLLO	---	Perl	Anexo A
	---	HTML	Capítulo1, sección 1.4
	---	Hojas de Estilo	Capítulo1, sección 1.4.7
	---	JavaScript	[a]
	---	Kwrite	[b]
IMPLEMENTACIÓN	Sistema Operativo	Mandriva Linux	[c]
	Servidor Web	Apache	Anexo B – Apache
FUNCIONAMIENTO	DHCP	Dhcpd	Anexo B - DHCPD
	Proxy-Caché	Squid	Anexo B - SQUID
	Firewall	Shorewall	Anexo B- SHOREWALL
	Compartir Directorios	Samba	Anexo B - SAMBA
	Filtrado de Contenidos	Dansguardian	Anexo B - DANSGUARDIAN

Tabla 2.1 Software requerido.

[a] JavaScript.- Es un lenguaje de scripting del lado del cliente que permite manipular los objetos que forman parte de un documento Web con el fin de crear algún tipo de acción o evento sobre éstos, obteniendo así páginas Web dinámicas, por ejemplo, permite crear un mensaje de alerta cuando se ha realizado alguna acción sobre un elemento, siendo éste un textarea, input, etc. La interpretación de javascript es realizada por el navegador Web.

[b] Kwrite.- Es un sencillo editor de texto que además sirve como herramienta de programación para los desarrolladores, ya que una de sus características más relevantes es que detecta el tipo de lenguaje que se está escribiendo; al detectar el tipo de lenguaje proporciona un tipo de marcación que facilita al programador el desarrollo de código, además permite la fácil identificación de su estructura, entre

los lenguajes que pueden ser detectados por Kwrite están Perl, Java, PHP, Hojas de Estilo (CSS), JavaScript, entre otros.

[c] Mandriva Linux.- Es una distribución del sistema operativo GNU/Linux, esta distribución fue desarrollada tanto para usuarios comunes como para empresas, es decir que es un sistema operativo que está orientado a satisfacer cualquier tipo de necesidades como: computadores personales, servidores, firewalls, etc. Anteriormente se le conocía con el nombre de Mandrake. Para el caso específico de un servidor, Mandriva ofrece algunas herramientas que permiten diferenciarlo de las demás distribuciones, como por ejemplo, cuenta con un sistema de archivos cifrado para garantizar la integridad de los datos, facilita la instalación y configuración de servicios como DHCP, SAMBA, PROXY, etc., una de las características más relevantes es que cuenta con una interfaz gráfica muy parecida a Windows, un ejemplo de esto es que cuenta con un centro de control en el que se pueden realizar múltiples tareas, entre las más destacadas tenemos: administrar los usuarios del sistema, compartir recursos con usuarios Windows, instalar paquetes software de manera muy sencilla, permite configurar los dispositivos de red así como compartir la conexión con otros usuarios, habilitar y deshabilitar los servicios del sistema, permite actualizar el sistema, administrar el hardware, configurar un contrafirewall personal, configurar el sistema de arranque, entre otras y todo esto se lo realiza de forma gráfica, lo cual hace su uso mucho más amigable en comparación a otras distribuciones que en muchos casos tienen que recurrir al Shell de comandos para realizar algunas de éstas tareas.

La versión que se instalará en el servidor será Mandriva Linux One 2009.1 Spring, ésta versión cuenta con KDE4 como entorno de escritorio predeterminado, la versión del kernel Linux es la 2.6.29.1 y soporte para sistema de archivos ext4.

En cuanto al **hardware requerido** para la implementación y ejecución tanto de la interfaz Web como para los diferentes servicios que se ejecutarán en el servidor, se emplea la siguiente información:

Cada servicio a ser ejecutado en un servidor ha sido diseñado para trabajar con determinados valores de recursos hardware tanto en memoria, disco duro y CPU.

Si bien esta configuración pondrá en ejecución cada servicio, no se los estará enfocando al ambiente real donde se ejecutarán, por ejemplo, para el caso de servicios como Proxy o filtrado web, se manejarán solicitudes para un número determinado de usuarios, lo que provoca que la necesidad de hardware dependa de éste número para ciertos servicios, ahora, ningún servicio implementado en el presente proyecto instala por defecto algún tipo de interfaz que permita su configuración, tan solo se cuenta con archivos de texto destinados a realizar esta tarea, por lo que los requerimientos hardware en lo referente a herramientas de configuración es nulo. En la tabla 2.2 se muestran las características hardware comunes para cada servicio, la información proporcionada se ha obtenido de las páginas web oficiales y relacionadas con el tema, ^{[4][5][6][7][8][9][10]} algunas características hardware dependen del número de usuarios que se manejan por lo que para el presente proyecto dichos valores posteriormente pueden disminuir debido al número reducido de usuarios.

Descripción		Requerimientos Hardware		
Servicio	Software	Memoria	Disco Duro	CPU
Proxy	Squid	32 MB	4 GB	250 MHz
WEB	Apache	32 MB	2 GB	150 MHz
Filtrado WEB	Dansguardian	32MB	2 GB	150 MHz
DHCP	DHCPD	16 MB	No requerido	250 MHz
Firewall	Shorewall	128 MB	10 GB	250 MHz
Compartir directorios	Samba	128 MB	> 10 GB	250 MHz

Tabla 2.2 Requerimientos hardware mínimos para distintos servicios.

En la tabla 2.2 se han señalado algunas filas de color para indicar que estos servicios, que si bien muestran sus requerimientos mínimos, necesitan realizar ciertos cálculos estimativos en base a la cantidad de usuarios que se van a manejar a través de cada servicio, para en base a esto obtener el hardware requerido.

Si bien el servicio web está diseñado para manejar varias solicitudes de usuarios, el enfoque dado en el presente proyecto se limita al manejo de una sola conexión

realizada por el administrador de la red o webmaster, por tal motivo el manejo de varios usuarios para el servicio web es relevante y su uso en hardware se limita a las características básicas requeridas.

Para el caso del servicio DHCP los requisitos mencionados en la tabla 2.2 se aplicarán para el cálculo del requerimiento hardware total. El servicio DHCP se limita a escuchar peticiones de los clientes que forman parte de la red, realizar la negociación y proveer la información de red necesaria. No almacenan información ni realizan procesos adicionales que afecten el grado de rendimiento del hardware del servidor, el servicio necesita un tamaño básico de memoria dado que la información de intercambio entre el servicio y los usuarios se maneja a través de ésta, puesto que no se requiere manejar información del o hacia el disco duro no se especifica un tamaño, la capacidad de utilización de memoria debe ser suficientemente amplia para evitar realizar procesos de intercambio o swap sobre el disco duro.

En lo referente al firewall se procederá de manera similar al servicio DHCP, se emplearán los datos proporcionados por la tabla 2.2 para realizar los cálculos y en conjunto con el resto de servicios obtener el hardware requerido. A continuación se desarrollará una explicación más amplia de aquellos servicios que manejan solicitudes de varios usuarios, el número de usuarios servirá como punto importante al momento de determinar la capacidad de hardware requerido.

- **PROXY (SQUID)**

El servicio proxy emplea dos elementos importantes dentro del hardware disponible en un servidor, la memoria y el disco duro. Estos elementos son empleados para realizar procesos de almacenamiento y obtención de información acerca de las páginas web que son solicitadas por un cliente (proceso de cache), el uso de CPU no es un factor crítico dentro del proxy ya que solo realiza un uso intenso de CPU al momento de iniciar los procesos que ponen en marcha al servicio.

El enfoque dado al presente proyecto limita el número de usuarios, la capacidad máxima de usuarios que manejará el servidor, para el caso en el cual todos los

usuarios accedan al servicio es de 20, por lo que las exigencias de hardware no demandarán grandes capacidades tanto en disco duro, memoria y CPU.

SQUID define en su archivos de configuración una línea destinada a determinar el valor del tamaño de memoria que empleará para el proceso de cache, a través de la entrada *cache_mem*, éste valor de memoria es empleado por squid para almacenar los objetos que son solicitados con mayor frecuencia por el cliente para accederlos inmediatamente y no realizar procesos de búsqueda en el disco duro. El tamaño por defecto es de 8MB. A esto hay que agregarle un valor adicional que emplea squid para mantener una tabla de índices para los objetos almacenados en cache, por cada GB empleado para almacenar objetos en el disco duro se empleará 6MB en memoria RAM aproximadamente. El cálculo de éste valor se desprende de aquellos empleados por squid para manejar los objetos; por cada objeto, squid usa en promedio 75 bytes de RAM. El tamaño de un objeto que solicita un cliente es de un valor promedio de 14KB. Ahora si empleamos 1GB en disco duro la cantidad de objetos que se podrían almacenar es de aproximadamente 75,000. De esta manera la cantidad de memoria RAM requerida en base a la cantidad de objetos sería de aproximadamente 5.5MB.

$$\#de\ objetos = \frac{1GB}{14KB} \times \frac{2^{20}KB}{1GB} = 74898$$

$$cantidad\ de\ RAM = 75000 \times 75B \times \frac{1MB}{2^{20}B} = 5,36MB$$

Para ver el tamaño promedio de los objetos en cache se emplea el comando:

```
# squidclient -h "dirección_IP_squid" mgr:info | grep "Mean"
```

Que obtiene como respuesta: *Mean Object Size: 14.10 KB*

Como se puede observar en cálculos anteriores existe un valor que permite calcular la cantidad de memoria RAM requerida en base al número de objetos, éste valor constituye el espacio en disco empleado para almacenar los objetos, de esta manera el siguiente valor a calcular corresponde al tamaño en disco que se va a emplear para almacenar los objetos. Y en base a éste valor se determinará el tamaño total de memoria RAM requerido para este servicio.

El cálculo del espacio en disco duro requerido por squid, va a depender de algunos factores, por ejemplo, el ancho de banda, la cantidad de datos que va a manejar el servicio, así como el tiempo que se proveerá el servicio a los clientes. Para determinar la cantidad promedio de datos que maneja el servicio se puede hacer referencia al ancho de banda disponible, para ello se va a asumir un ancho de banda de 1Mbps, dado que al ser el proyecto un prototipo se desarrollará adoptándolo a la capacidad de ancho de banda con la que cuentan los desarrolladores. La cantidad de bytes por segundo que se pueden transmitir en el ancho de banda mencionado es de alrededor de 125KB/s.

Ahora se tiene que asumir la peor condición de funcionamiento, ésta ocurre cuando los usuarios emplean todo el ancho de banda disponible, por ejemplo durante una hora y todos los objetos son almacenados en el disco duro (cache), de esta manera la cantidad de bytes que se han obtenido de internet en una hora corresponden a:

$$\frac{125KB}{1s} \times \frac{3600s}{1h} \times \frac{1MB}{2^{10}KB} = 439MB/h$$

Tomando en cuenta que el servicio proxy va a ser utilizado durante 8 horas al día, la cantidad de espacio requerido para la peor condición durante un día será de aproximadamente 3,5GB.

Luego de haber obtenido el valor aproximado en disco necesario para un día en la peor condición, se puede proceder a calcular la cantidad de memoria RAM requerida:

$$3,5GB \times \frac{5,36MB}{1GB} \cong 19MB$$

La capacidad de CPU en lo referente al servicio proxy no afecta directamente a su funcionamiento, pero se debe emplear un buen procesador que facilite los procesos de puesta en marcha de servicios y distintos procesos que ejecute el sistema operativo, los puntos clave en el servicio proxy como se puede observar lo constituyen tanto la capacidad de disco y memoria.

- **FILTRADO WEB (DANSGUARDIAN)**

En la sección 2.2.6 se mencionó que dansguardian trabaja en conjunto con el servicio proxy proporcionado por squid, por ésta razón los requerimientos hardware en la página oficial de dansguardian hacen referencia al tamaño de memoria enfocado al proceso de cache del proxy, indicando que el requerimiento de memoria por parte de dansguardian es mínimo. La capacidad de memoria y CPU necesarias para poner en marcha el servicio proporcionado por dansguardian se ha dividido de acuerdo al número de usuarios que el servicio va a manejar, se tienen dos posibilidades:

- Para un número de 50 usuarios la capacidad de CPU requerida es de 150MHz con un tamaño de memoria de 32MB.
- Para un número mayor a 50 usuarios los mínimos requerimientos son 150MHz y un tamaño de memoria de 64MB.

La variación del tamaño de memoria en las dos posibilidades se enfoca al servicio proporcionado por squid. El presente proyecto maneja un número máximo de 20 usuarios, por lo que se empleará la primera opción manteniendo el valor requerido de CPU dado que se manejarán distintos procesos y servicios, mientras que la capacidad de memoria puede variar, la opción seleccionada menciona que se manejará como máximo 50 usuarios con un tamaño de memoria de 32MB; si lo enfocamos a 20 usuarios, la capacidad disminuirá al orden de los 13MB teniendo una aproximación con el valor que se calculó previamente en squid. De esta manera se puede tener claro que los requerimientos de memoria para éste servicio están relacionados con los requerimientos que squid necesita para su funcionamiento.

- **COMPARTIR DIRECTORIOS (SAMBA)**

El servicio proporcionado por Samba permite a los usuarios emplear directorios dentro del servidor para almacenar información, el requerimiento importante para éste servicio constituye la capacidad de almacenamiento que se va a proporcionar a cada usuario, en lo referente a la capacidad de memoria se debe tomar en cuenta que cada usuario consumirá un porcentaje de memoria de

aproximadamente 4MB, para la peor condición es decir que todos los usuarios accedan al servicio la cantidad de memoria empleada es aproximadamente (4MB x 20 usuarios) 80MB.

Hay que tomar en cuenta varios aspectos al momento de determinar la capacidad de disco duro: el tipo de documentos que se van a almacenar, que cantidad de espacio se espera proporcionar, la creación de directorios será similar para todos, existirán directorios compartidos a todos los usuarios. En la tabla 2.3 se determina estos aspectos junto a una breve descripción para cada uno de ellos.

Aspecto	Descripción
Tipo de documentos	Al tratarse de un proyecto enfocado a empresas SOHO. La mayor cantidad de documentos que se emplearán son del tipo ofimático. Limitando el contenido multimedia.
Espacio estimado a proporcionar	Tomando en cuenta el tipo de datos que se almacenará, un espacio de 700MB será utilizado por cada usuario en un año en la peor condición, dado que los archivos ofimáticos no poseen gran tamaño.
La creación y acceso de directorios es similar para todos los usuarios.	Tanto el acceso como la creación de directorios estará a cargo del administrador del sistema, lo recomendable es crear directorios exclusivamente para los usuarios que lo requieran con la finalidad de evitar el mal uso del espacio disponible en disco.
Existirán directorios comunes a todos los usuarios.	La finalidad de contar con un directorio común, es permitir que todos los usuarios puedan acceder y modificar los distintos documentos dentro del directorio, por lo que al momento de obtener el valor de espacio total se tendrá que asignar un espacio adicional que tome en cuenta este aspecto.

Tabla 2.3 Aspectos para el cálculo de la capacidad en disco para samba.

Comúnmente dentro de una empresa la mayor cantidad de documentos ofimáticos que se utilizan son del tipo texto, y hojas de cálculo, los documentos para generar presentaciones son empleados pocas veces por la mayoría de personal dentro de una empresa, por lo que, para el cálculo del espacio en disco necesario para el servicio proporcionado por samba se lo realizará enfocado en los dos tipos de archivos más empleados.

Los tamaños de documentos ofimáticos utilizados para calcular el espacio necesario en disco serán enfocados a documentos con contenidos de texto, imágenes, estilos, que provocan que un documento tenga un tamaño razonable el cual va a ser tomado como peor condición.

Un documento de Microsoft Word con 10 imágenes, estilos sobre el texto, y un número de 10 hojas tiene un tamaño promedio de 600KB; para un documento de Microsoft Excel con texto, 10 imágenes, estilos, en una sola hoja de cálculo el tamaño promedio es de 500KB.

Diariamente se crean alrededor de 15 documentos de los cuales se almacenan localmente en la estación de trabajo 8, en el servidor se almacenan aquellos documentos con un grado de importancia generalmente 5, el resto de documentos son temporales y empleados para procedimientos de impresión.

Al contar con un número de 20 usuarios, cada uno de ellos almacenará un total de 5 documentos sea de texto u hojas de cálculo en el espacio disponible en el disco del servidor, la peor condición corresponderá a la situación en la cual todos almacenen diariamente 5 documentos durante todo un año, el promedio de tamaño del documento sea de texto u hoja de cálculo será de 550 KB, para esto la cantidad de espacio requerido corresponderá a:

$$550\text{KB} \times \frac{5 \text{ documentos}}{1 \text{ día}} \times \frac{5 \text{ días}}{1 \text{ semana}} \times \frac{52 \text{ semana}}{1 \text{ año}} \times \frac{1 \text{ MB}}{2^{10}\text{KB}} \cong 700 \text{ MB}$$

$$700 \text{ MB} \times 20 \text{ usuarios} = 14000 \text{ MB}$$

$$14000 \text{ MB} \times \frac{1 \text{ GB}}{2^{10}\text{MB}} = 13,67 \text{ GB}$$

directorios comunes \cong 2GB más.

total necesario \cong 15 GB

Luego de haber determinado los requerimientos necesarios para el servicio proporcionado por samba, se puede concluir que la cantidad de memoria destinada a éste servicio es aproximadamente 80MB para la peor condición, de la misma manera, la cantidad de espacio en disco requerida para la peor condición es de 15GB.

- **ESTIMACION FINAL**

La determinación del valor total necesario para disco duro, memoria y CPU se la realizará en base a la suma de los valores de cada uno de los servicios mencionados en párrafos anteriores, de esta manera se tiene:

Servicio	Memoria	Disco Duro	CPU
Proxy	19 MB	4 GB	250 MHz
WEB	32 MB	2 GB	150 MHz
Filtrado WEB	13 MB	2 GB	150 MHz
DHCP	16 MB	No requerido	250 MHz
Firewall	128 MB	10 GB	250 MHz
Compartir directorios	80 MB	15 GB	250 MHz
TOTAL	288 MB	33 GB	1300 MHz

Tabla 2.4 capacidades totales para el ambiente de desarrollo.

Los valores obtenidos corresponden a la peor condición de funcionamiento, en la cual todos los usuarios se encuentran interactuando con el servidor. Como se puede observar en la tabla 2.4 la capacidad de memoria se aproxima a un valor estándar 256 MB, para el disco duro la capacidad obtenida se aproxima a un valor estándar de 40GB, mientras que el requerimiento de CPU corresponde a un valor estándar de aproximadamente 1.3 GHz. Hay que tomar en cuenta que se tiene que añadir a estos valores los correspondientes al uso del sistema operativo y demás procesos que se ejecutan en el servidor.

Los valores totales en la tabla 2.4 son suficientes para el ambiente en el cual se desarrolla el presente proyecto, dado que se está desarrollando un prototipo y no será puesto en funcionamiento en un ambiente real a largo plazo, estos valores permitirán demostrar el funcionamiento tanto de la interfaz de administración web así como también los distintos servicios que dicha interfaz manejará.

Como información adicional, se indica que el servidor debe estar activo para que los equipos de la red local puedan acceder a los servicios, además éste contará con dos adaptadores de red, uno para conectarse con la red local y otro para conectarse a Internet. Debido al número de usuarios que comprende una red tipo SOHO y al ancho de banda disponible para el acceso a Internet, los adaptadores de red a ser utilizados serán del tipo Fast Ethernet (100Mbps).

2.1.2.2 Funciones del Producto

En ésta sección se define de forma general y clara, la funcionalidad de cada módulo que comprende DAMA:

2.1.2.2.1 Configuración del Servicio DHCP

- Configuración Básica del Servicio DHCP.
- Configuración Dinámica del Servicio DHCP.
- Configuración Estática del Servicio DHCP.
- Direccionamiento IP.

2.1.2.2.2 Configuración del Servicio Proxy – Caché

- Modificación de Opciones Generales.
- Generación de Listas de Control de Acceso.
- Generación de Políticas de Acceso.
- Manejo de Usuarios para el servicio.

2.1.2.2.3 Configuración del Servicio de Directorios Compartidos

- Visualización de Información General.

- Manejo de Usuario para el servicio.
- Adición de Directorios al servicio.
- Modificación de las características de los Directorios Compartidos.

2.1.2.2.4 Configuración del Servicio de Filtrado Web

- Modificación de Opciones Generales.
- Filtrado por MIME.
- Filtrado por Extensión.
- Filtrado por Site.
- Filtrado por Frase.

2.1.2.2.5 Configuración del Servicio de Firewall

- Generación de Zonas.
- Generación de relación entre una Zona y una Interfaz de Red.
- Generación de Políticas de Acceso de alto nivel.
- Generación de Reglas de acceso específicas.
- Generación de Enmascaramiento.

2.1.2.2.6 Configuración del Servicio de Estado del Sistema Operativo

- Disco Duro:
 - Estadísticas de uso del Disco Duro.
 - Estadísticas de Discos Duros Adicionales y Particiones.
- RAM y CPU:
 - Estadística General de Memoria.
 - Uso de CPU.
- Información de Red:

- Interfaz con salida a Internet.
- Tabla de Enrutamiento.
- Interfaz de Red Local.
- Información Adicional:
 - Usuarios y Tiempo activo del Sistema.
 - Información del Kernel.

2.1.2.2.7 *Configuración del Servicio de Gráficas*

- Gráficas del Rendimiento del Tráfico de Red.
- Gráficas del Uso de Memoria.
- Gráficas de la Carga del CPU.

2.1.2.2.8 *Página Inicial*

2.1.2.3 **Características de los Usuarios**

DAMA está enfocada a usuarios con conocimientos en administración de redes, servicios que se ejecutan en Linux, conocimientos básicos en el manejo de interfaces Web, es decir está enfocada hacia el webmaster o administrador de una red local.

2.1.2.4 **Restricciones**

- El producto DAMA administrará un número de hasta 20 usuarios, debido al tipo de empresa al cual está orientado.
- Los servicios que corren sobre el servidor Linux emplearán las características de acuerdo al tipo de empresa en el cual se van a ejecutar.
- DAMA se implementará y ejecutará sobre el servidor Web instalado en el sistema operativo Linux.
- El direccionamiento empleado en la red local será el protocolo IPv4.

2.1.2.5 Suposiciones y Dependencias

El producto DAMA no sufrirá ninguna modificación hasta la fase final, para su desarrollo empleará todos los requisitos que se han mencionado sin ningún tipo de alteración.

Para su ejecución e implementación DAMA requiere: sistema operativo Linux, conjunto de servicios a ser administrados, servidor Web, navegador Web.

2.1.2.6 Requisitos Futuros

El desarrollo del producto DAMA tiene como objetivo la obtención del título de Ingeniero en Electrónica y Redes de Información por parte de los desarrolladores, razón por la cual al finalizar el producto, éste se dará por concluido, por lo que no tendrá ningún tipo de requisitos futuro.

2.1.3 REQUISITOS ESPECÍFICOS

2.1.3.1 Interfaces Externas

DAMA presenta una interfaz Web amigable para el usuario, el acceso a cada módulo se encuentra distribuido a manera de menú en la parte superior central de la interfaz Web donde se observará el nombre de cada módulo y se accederá a él a través de un enlace de forma que el usuario pueda identificarlos e ingresar a cada uno de ellos fácilmente, dentro de un módulo cada funcionalidad se presenta de manera organizada facilitando las tareas que el usuario desee realizar en cada uno de ellos, el acceso a una funcionalidad se la realiza a través de un enlace que dirige al navegador hacia la sección en la página Web donde se encuentra dicha funcionalidad en el módulo respectivo. La interfaz proporcionará al usuario mensajes de alerta, confirmación y obtención de ayuda en cada uno de los módulos, de ésta forma se guiará al usuario en el uso y navegabilidad de DAMA.

2.1.3.2 Funciones

Dado que para el desarrollo del proyecto se ha seleccionado la metodología XP (Xtreme Programming o Programación Extrema), se emplearán historias de usuarios propias de ésta metodología para definir las diferentes funciones.

El formato de la historia de usuario a emplear es la siguiente:

Historia de Usuario	
Número:	Nombre:
Modificación de Historia de Usuario(Nro. y Nombre):	
Usuario:	Iteración Asignada:
Prioridad (Alta/Media/Baja):	
Riesgo en Desarrollo (Alta/Media/Baja):	
Descripción:	
Observaciones:	

Tabla 2.5 Formato de Historia de Usuario.

En donde:

- *Número*: corresponde al número de historia de usuario.
- *Nombre*: es el nombre de la historia.
- *Usuario*: es el nombre del usuario del producto.
- *Modificación de Historia de Usuario*: es empleado cuando se ha solicitado una modificación sobre la función definida en la historia de usuario.
- *Iteración Asignada*: es un número empleado para indicar la iteración de la función dentro del proceso de su desarrollo.
- *Prioridad*: se define como alta, media o baja, e indica el nivel de prioridad que comprende el desarrollo.
- *Riesgo en Desarrollo*: se define como alta, media o baja, e indica posibles riesgos que podrían presentarse en el proceso de desarrollo, por ejemplo el uso de tecnología o herramientas no conocidas.
- *Descripción*: en ésta sección se describe la funcionalidad solicitada por el cliente.
- *Observaciones*: se definen diferentes aspectos a tomar en cuenta al momento del desarrollo.

Para el número de historia de usuario se empleará una abreviatura, la cual se definirá al inicio de cada módulo, por ejemplo para el módulo de DHCP se

empleará la abreviatura **HUD (Nro.)**, donde **HU** hace referencia a Historia de Usuario, **D** al servicio DHCP y **(Nro.)** corresponderá al número de historia.

MÓDULO DEL SERVICIO DHCP

La abreviatura empleada para éste módulo será: **HUD (Nro.)**. En la tabla 2.6 se presenta la historia de usuario correspondiente a la configuración básica del servidor DHCP.

Historia de Usuario	
Número: HUD01	Nombre: Configuración Básica del Servicio DHCP
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá ingresar la dirección IP para la interfaz que proveerá el servicio, dirección de red a la cual pertenece la interfaz, máscara de red y dirección broadcast de red, una vez ingresados los parámetros se contará con un botón que permitirá guardar los cambios.	
Observaciones: Si los parámetros no han sido ingresados correctamente, DAMA le indicará al usuario mediante mensajes de alerta, que el ingreso de parámetros ha sido incorrecto, por lo que tendrá que ingresarlos nuevamente hasta que concuerde con el formato establecido. Cada vez que se realice una configuración se requiere del reinicio del servicio para que los cambios se hagan efectivos.	

Tabla 2.6 Historia de Usuario-Configuración Básica del Servidor DHCP.

La tabla 2.7 muestra la historia de usuario correspondiente a la configuración dinámica del servicio DHCP:

Historia de Usuario	
Número: HUD02	Nombre: Configuración Dinámica del Servicio DHCP
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá ingresar la dirección IP máxima, dirección IP mínima, que corresponden al rango de direcciones IP que se asignará, tiempo mínimo de alquiler de la dirección IP en segundos y tiempo máximo de alquiler de la dirección IP en segundos. Una vez ingresados los parámetros, se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: En caso de que los parámetros sean ingresados incorrectamente, DAMA le informará al usuario mediante mensajes de alerta, que el parámetro ingresado no corresponde al formato establecido. Cada vez que se realice una configuración se requiere del reinicio del servicio para que los cambios se hagan efectivos.	

Tabla 2.7 Historia de Usuario-Configuración Dinámica del Servicio DHCP.

La tabla 2.8 muestra la historia de usuario correspondiente a la configuración estática del servicio DHCP:

Historia de Usuario	
Número: HUD03	Nombre: Configuración Estática del Servicio DHCP
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se contará con dos formas de asignar una dirección IP estática, en la parte superior se podrá ingresar la dirección IP, dirección MAC y nombre del host, una vez ingresados los parámetros se contará con un botón que permitirá guardar los cambios, posterior a esto se podrá seleccionar un host que ya ha utilizado el servicio DHCP, aquí se podrá añadir un host seleccionándolo de una lista, asimismo se podrá visualizar los hosts con dirección IP estática que están utilizando el servicio, por último se contará con un botón que permitirá eliminar un host de ésta configuración.	
Observaciones: En el caso de ingresar los parámetros, DAMA le indicará al usuario mediante mensajes de alerta si los valores ingresados no son suficientes o están incorrectos. Cada vez que se realice una configuración se requiere del reinicio del servicio para que los cambios se hagan efectivos.	

Tabla 2.8 Historia de Usuario-Configuración Estática del Servicio DHCP.

La tabla 2.9 muestra la historia de usuario correspondiente al direccionamiento IP:

Historia de Usuario	
Número: HUD04	Nombre: Direccionamiento IP
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Aquí se mostrará una lista de las direcciones IP asignadas dinámicamente, en ésta lista se podrá visualizar la dirección IP, dirección MAC, el nombre del host, y los detalles acerca de la concesión.	
Observaciones:	

Tabla 2.9 Historia de Usuario-Direccionamiento IP.

MÓDULO DEL SERVICIO PROXY - CACHÉ

La abreviatura empleada para éste módulo será: *HUP* (Nro.). La tabla 2.10 muestra la historia de usuario correspondiente a la modificación de opciones generales:

Historia de Usuario	
Número: HUP01	Nombre: Modificación de Opciones Generales
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se mostrarán los parámetros que se encuentren configurados en ese momento, además algunos de éstos se podrán modificar como por ejemplo el idioma, el puerto por el cual se conecta el servicio, el tipo de proxy, el espacio empleado por el servicio para almacenar la información de las páginas visitadas y el nombre que permitirá identificar al servicio. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: En el caso de ingresar nuevos valores, DAMA le indicará al usuario mediante mensajes de alerta si éstos son incorrectos. Cada vez que se realice cambios en la configuración, es necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.10 Historia de Usuario-Modificación de Opciones Generales.

La tabla 2.11 muestra la historia de usuario correspondiente a la generación de listas de control de acceso:

Historia de Usuario	
Número: HUP02	Nombre: Generación de Listas de Control de Acceso
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se permitirá la creación de diferentes tipos de listas de control de acceso, además se contará con la opción de almacenar o no el contenido de las páginas web en el disco duro. Se contará con un botón que permitirá agregar la nueva lista de control de acceso a un área de texto en donde se podrá visualizar las listas de control de acceso creadas, éstas se encontrarán en modo de lectura, se contará con una opción que permitirá modificar su contenido. Una vez realizada la configuración se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado ingresar algún parámetro obligatorio o si éstos han sido ingresados incorrectamente. Cada vez que realice cambios en la configuración, es necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.11 Historia de Usuario-Generación de Listas de Control de Acceso.

La tabla 2.12 muestra la historia de usuario que corresponde a la generación de políticas de acceso:

Historia de Usuario	
Número: HUP03	Nombre: Generación de Políticas de Acceso
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Por cada lista de control de acceso creada se requerirá de la creación de una política de acceso, se podrá permitir o denegar el acceso a esa lista de control, se tendrá la opción de solicitar autenticación para el acceso, se contará con un botón que permitirá agregar la nueva política de acceso a un área de texto que en un principio será solo de lectura, en donde se podrá visualizar las políticas existentes, aquí se tendrá una opción que permitirá modificar su contenido. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: Cada vez que se realicen cambios en la configuración será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.12 Historia de Usuario-Generación de Políticas de Acceso.

La tabla 2.13 muestra la lista de usuario correspondiente al manejo de usuarios para el servicio:

Historia de Usuario	
Número: HUP04	Nombre: Manejo de Usuarios para el Servicio
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se permitirá agregar usuarios mediante el ingreso del nombre del usuario así como el ingreso de una contraseña y su correspondiente confirmación, se contará con un botón que permitirá agregar al nuevo usuario, se podrá eliminar un usuario del servicio seleccionándolo de entre una lista de selección, automáticamente se mostrará un mensaje de confirmación. Además se podrá cambiar la contraseña del usuario, seleccionando su nombre de entre una lista de selección y a continuación se ingresará la nueva contraseña con su correspondiente confirmación, se contará con un botón que permitirá guardar la nueva contraseña.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado ingresar algún parámetro o si los ha ingresado incorrectamente.	

Tabla 2.13 Historia de Usuario-Manejo de Usuarios para el Servicio.

MÓDULO DEL SERVICIO DE DIRECTORIOS COMPARTIDOS

La abreviatura empleada para éste módulo será: *HUS* (Nro.).La tabla 2.14 muestra la historia de usuario que corresponde a la visualización de información general:

Historia de Usuario	
Número: HUS01	Nombre: Visualización de Información General
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar tanto el grupo de trabajo como los hosts dentro de la red local que tendrán acceso al servicio.	
Observaciones: Todos los usuarios Windows deben encontrarse en el mismo grupo de trabajo para poder acceder al servicio.	

Tabla 2.14 Historia de Usuario-Visualización de Información General.

La tabla 2.15 muestra la historia de usuario correspondiente al manejo de usuarios para el servicio:

Historia de Usuario	
Número: HUS02	Nombre: Manejo de Usuarios para el Servicio
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se permitirá agregar usuarios mediante el ingreso de su nombre, contraseña y su correspondiente confirmación, se contará con un botón que permitirá agregar el nuevo usuario al servicio. Se podrá cambiar la contraseña del usuario seleccionando su nombre de entre una lista de selección e ingresando su nueva contraseña y su correspondiente confirmación, se contará con un botón que permitirá guardar la nueva contraseña. Además permitirá eliminar un usuario del servicio, seleccionando el nombre del usuario de entre una lista de selección, automáticamente se desplegará un mensaje de confirmación.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado ingresar un parámetro o si ha ingresado alguno de forma incorrecta.	

Tabla 2.15 Historia de Usuario-Manejo de Usuarios para el Servicio.

La tabla 2.16 muestra la historia de usuario correspondiente a la adición de directorios al servicio:

Historia de Usuario	
Número: HUS03	Nombre: Adición de Directorios al Servicio
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá agregar un nuevo directorio a ser compartido, mediante el ingreso de opciones que son generales como nombre y un comentario, así como aspectos de seguridad en el que se definirá si el directorio será solo de lectura o escritura, si permitirá el acceso como invitado o si restringirá el acceso a ciertos usuarios. Se contará con un botón que permitirá agregar el directorio a ser compartido.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado ingresar algún parámetro obligatorio o si ha ingresado alguno de forma incorrecta. Cada vez que se realicen cambios en la configuración es necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.16 Historia de Usuario-Adición de Usuarios al Servicio.

La tabla 2.17 muestra la historia de usuario correspondiente a la modificación de las características de los directorios compartidos:

Historia de Usuario	
Número: HUS04	Nombre: Modificación de las Características de los Directorios Compartidos
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá modificar tanto el comentario como los aspectos de seguridad del directorio compartido, con solo seleccionar su nombre de entre una lista de selección. Una vez hechas las modificaciones, se contará con un botón que permitirá guardar los cambios realizados. Además al seleccionar un nombre de directorio, se contará con un botón que permitirá borrar el directorio del servicio.	
Observaciones: DAMA le indicará al cliente mediante mensajes de alerta si ha olvidado ingresar parámetros obligatorios o si los valores ingresados son incorrectos. Cada vez que se realicen cambios en la configuración, es necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.17 Historia de Usuario-Modificación de las Características de los Directorios Compartidos.

MÓDULO DEL SERVICIO DE FILTRADO WEB

La abreviatura empleada para éste módulo será: *HUW (Nro.)*. La tabla 2.18 muestra la historia de usuario correspondiente a la modificación de opciones generales:

Historia de Usuario	
Número: HUW01	Nombre: Modificación de Opciones Generales
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Permitirá visualizar opciones generales como dirección IP y puerto por donde escucha el servicio, el idioma seleccionado y la dirección IP y puerto del servicio Proxy, dado que trabaja en conjunto con éste servicio. A su vez se permitirá modificar las opciones de idioma y el puerto por donde escucha el servicio. Una vez modificadas estas opciones, se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: Cada vez que se modifique la configuración, es necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.18 Historia de Usuario-Modificación de Opciones Generales.

La tabla 2.19 muestra la historia de usuario correspondiente al filtrado por MIME:

Historia de Usuario	
Número: HUW02	Nombre: Filtrado por MIME
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Permitirá filtrar el contenido web mediante el tipo de MIME seleccionado. Se presentará dos formas de filtrar, la primera es seleccionando un tipo MIME de entre una lista de selección preestablecida y la segunda es ingresando un nuevo tipo MIME, una vez ingresado o seleccionado el tipo MIME, se contará con un botón que permita ingresar el nuevo tipo a un área de texto que en un principio será solo de lectura, en dicha lista se podrá visualizar todos los filtros por MIME agregados, ésta contará con una opción que permitirá modificar su contenido. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si los filtros por MIME seleccionados o ingresados ya existen en la lista, o si el filtro ingresado no corresponde al formato indicado. Cada vez que se modifique la configuración, es necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.19 Historia de Usuario-Filtrado por MIME.

La tabla 2.20 muestra la historia de usuario que corresponde al filtrado por extensión:

Historia de Usuario	
Número: HUW03	Nombre: Filtrado por Extensión
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Permitirá filtrar el contenido web mediante extensiones de archivos no permitidos, para esto se ingresará un nuevo filtro por extensión, con la ayuda de un botón se permitirá agregar el nuevo filtro a un área de texto que permitirá visualizar los filtros por extensión existentes, ésta lista en un principio será solo de lectura, ésta contará con una opción que permitirá modificar el contenido de la misma. Una vez realizadas las modificaciones se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si el filtro por extensión ingresado ya existe dentro de la lista o si el valor ingresado no corresponde con el formato indicado. Cada vez que se modifique la configuración será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.20 Historia de Usuario-Filtrado por Extensión.

La tabla 2.21 muestra la historia de usuario correspondiente al filtrado por site:

Historia de Usuario	
Número: HUW04	Nombre: Filtrado por Site
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Permitirá bloquear el acceso a todo el contenido de un sitio web mediante la selección o ingreso de nombres de dominios, se podrá incluir nuevos dominios seleccionándolo de entre una lista con nombres predefinidos o ingresando un nuevo nombre de dominio, una vez ingresado o seleccionado un valor, se contará con un botón que permitirá ingresar el nuevo nombre a un área de texto en el cual se visualizará los filtros por site existentes, ésta lista en un principio será solo de lectura, además contará con una opción que permitirá modificar el contenido de la misma. Una vez realizadas las modificaciones, se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si los valores seleccionados o ingresados ya existen en la lista, o si el valor insertado no corresponde al formato. Cada vez que se modifique la configuración, será necesario reiniciar el servicio, para que los cambios se hagan efectivos.	

Tabla 2.21 Historia de Usuario-Filtrado por Site.

La tabla 2.22 muestra la historia de usuario correspondiente al filtrado por frase:

Historia de Usuario	
Número: HUW05	Nombre: Filtrado por Frase
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Permitirá filtrar el contenido de una página web por la palabra o frase seleccionada o ingresada, se podrá añadir un nuevo filtro mediante la selección de un palabra o frase de entre una lista de selección preestablecida o mediante el ingreso de una nueva palabra o frase, se contará con un botón que permitirá agregar el nuevo filtro a un área de texto en donde se visualizarán los filtros por palabra o frase existentes, ésta lista en un principio será solo de lectura, ésta contará con una opción que permitirá modificar el contenido de la misma. Una vez realizadas las modificaciones, se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al cliente mediante mensajes de alerta si los filtros seleccionados o ingresados ya existen en la lista, o si el valor ingresado no corresponde al formato indicado. Cada vez que se modifique la configuración, el servicio deberá ser reiniciado, para que los cambios se hagan efectivos.	

Tabla 2.22 Historia de Usuario-Filtrado por Frase.

MÓDULO DEL SERVICIO DE FIREWALL

La abreviatura empleada para éste módulo será: *HUF* (Nro.). La tabla 2.23 muestra la historia de usuario correspondiente a la generación de zonas:

Historia de Usuario	
Número: HUF01	Nombre: Generación de Zonas
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se ingresará el nombre de la zona a ser administrada por el firewall, se contará con un botón que permitirá agregar la nueva zona a un área de texto en donde se podrá visualizar las zonas definidas en el firewall, en un principio la lista será solo de lectura, ésta contará con una opción que permitirá modificar su contenido. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si el nombre de zona no ha sido ingresado o si ésta se ha ingresado incorrectamente. Cada vez que se modifique la configuración, será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.23 Historia de Usuario-Generación de Zonas.

La tabla 2.24 muestra la historia de usuario que corresponde a la generación de relación entre una zona y una interfaz:

Historia de Usuario	
Número: HUF02	Nombre: Generación de relación entre una Zona y una Interfaz de Red
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se realizará la creación de relaciones entre una zona y una interfaz de red, se seleccionará un nombre de zona anteriormente creada, de entre una lista de selección, asimismo se seleccionará una interfaz con la cual se desee relacionar a la zona, con la ayuda de un botón se agregarán éstos valores a un área de texto en donde se podrá visualizar las relaciones creadas, en un principio ésta lista será solo de lectura, ésta contará con una opción que permitirá modificar su contenido. Se contará con un botón que permita guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado seleccionar algún valor o si la relación seleccionada ya ha sido creada. Cada vez que se modifique la configuración, será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.24 Historia de Usuario-Generación de relación entre una Zona y una Interfaz de Red.

La tabla 2.25 muestra la historia de usuario que corresponde a la generación de políticas de acceso de alto nivel:

Historia de Usuario	
Número: HUF03	Nombre: Generación de Políticas de Acceso de alto nivel
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se permitirá crear políticas de acceso de alto nivel entre una zona origen y una zona destino, tanto las zonas como las políticas serán seleccionadas de entre una lista de selección que se establecerá para cada uno de éstos valores. Con la ayuda de un botón se podrán agregar éstos valores a un área de texto que permitirá visualizar las políticas aplicadas, en un principio esta lista será solo de lectura, ésta contará con una opción que permitirá modificar su contenido. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado seleccionar algún valor o si la política creada ya existe. Cada vez que se modifique la configuración, será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.25 Historia de Usuario-Generación de Políticas de Acceso de alto nivel.

La tabla 2.26 muestra la historia de usuario correspondiente a la generación de reglas de acceso específicas:

Historia de Usuario	
Número: HUF04	Nombre: Generación de Reglas de acceso específicas
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Permitirá crear reglas de acceso específicas para las solicitudes de conexión, seleccionando una acción de entre una lista de selección preestablecida, para su posterior configuración, mediante un botón se agregará ésta acción a un área de texto que permitirá visualizar las reglas creadas, en un principio esta lista será solo de lectura, ésta contará con una opción que permitirá modificar su contenido. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: DAMA le indicará al usuario mediante mensajes de alerta si ha olvidado seleccionar la acción. Cada vez que se modifique la configuración será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.26 Historia de Usuario-Generación de Reglas de acceso específicas.

La tabla 2.27 muestra la historia de usuario correspondiente a la generación de enmascaramiento (NAT):

Historia de Usuario	
Número: HUF05	Nombre: Generación de Enmascaramiento
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Con la ayuda de un botón se permitirá añadir la interfaz con conexión a internet, para su posterior configuración, éste valor se agregará a un área de texto en donde se podrá visualizar el enmascaramiento actual, en un principio esta lista será solo de lectura, ésta contará con una opción que permitirá modificar su contenido, en éste caso relacionar la interfaz agregada con la red para el proceso de enmascaramiento. Se contará con un botón que permitirá guardar los cambios realizados.	
Observaciones: Cada vez que se modifique la configuración será necesario reiniciar el servicio para que los cambios se hagan efectivos.	

Tabla 2.27 Historia de Usuario-Generación de Enmascaramiento.

MÓDULO DEL SERVICIO ESTADO DEL SISTEMA OPERATIVO

La abreviatura empleada para éste módulo será: *HUE (Nro.)*.

- *Disco Duro*

En la tabla 2.28 se muestra la historia de usuario correspondiente a la generación de estadísticas de uso del disco duro:

Historia de Usuario	
Número: HUE01	Nombre: Generación de Estadísticas de uso del Disco Duro
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar información del disco duro como porcentajes de utilización, tamaño total y disponible tanto para el sistema operativo como para los directorios personales.	
Observaciones: Para el caso del sistema operativo Linux los directorios personales se conocen como /home.	

Tabla 2.28 Historia de Usuario-Generación de Estadísticas de uso del Disco Duro.

En la tabla 2.29 se muestra la historia de usuario correspondiente a la generación de estadísticas de discos duros adicionales y particiones.

Historia de Usuario	
Número: HUE02	Nombre: Generación de Estadísticas de Discos Duros Adicionales y Particiones
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: En el caso de existir particiones o discos duros adicionales, se podrá visualizar información como porcentajes de utilización, tamaño total y disponible, tanto para la partición como para el disco duro adicional.	
Observaciones: Esta información solo se podrá visualizar en caso de existir una partición o disco duro adicional.	

Tabla 2.29 Historia de Usuario-Generación de Estadísticas de Discos Duros Adicionales y Particiones.

- RAM y CPU

La tabla 2.30 muestra información acerca de la historia de usuario correspondiente a la generación de estadística general de memoria:

Historia de Usuario	
Número: HUE03	Nombre: Generación de Estadística General de Memoria
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar información detallada acerca de la memoria como porcentaje de utilización, tamaño total y disponible tanto para la memoria física asignada al sistema operativo, la memoria utilizada por el kernel en ese momento, la memoria física asignada para intercambio como para la memoria física total.	
Observaciones: Se dará mayor importancia a la estadística relacionada con la memoria utilizada por el kernel en ese momento.	

Tabla 2.30 Historia de Usuario-Generación de Estadística General de Memoria.

La tabla 2.31 muestra la historia de usuario correspondiente a la generación de estadística del uso de CPU:

Historia de Usuario	
Número: HUE04	Nombre: Generación de Estadística del Uso de CPU
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar información general acerca del estado del CPU como porcentaje de uso, el vendedor, modelo, velocidad, tamaño de caché, número de núcleos y la temperatura actual del CPU, el cual contará con una imagen que cambiará según la temperatura indicada. También se visualizará información correspondiente a cada núcleo en el caso de existir más de uno, la información que se detallará será: porcentaje de uso, vendedor, modelo, velocidad, tamaño de caché e identificador de núcleo.	
Observaciones:	

Tabla 2.31 Historia de Usuario-Generación de Estadística del Uso de CPU.

- *Información de Red*

La tabla 2.32 muestra información que corresponde a la generación de estadística de interfaz con salida a Internet:

Historia de Usuario	
Número: HUE05	Nombre: Generación de Estadística de Interfaz con salida a Internet
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Baja	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar información detallada acerca de la interfaz con salida a Internet, la información que se mostrará será: nombre de la interfaz y el protocolo que emplea, nombre del dispositivo al cual se conecta, información obtenida al negociar con el dispositivo, tiempos y fechas asignadas a la interfaz e información referente al estado de la interfaz en ese momento.	
Observaciones:	

Tabla 2.32 Historia de Usuario-Generación de Estadística de Interfaz con salida a Internet.

La tabla 2.33 muestra la historia de usuario que corresponde a la generación de tabla de enrutamiento:

Historia de Usuario	
Número: HUE06	Nombre: Generación de Tabla de Enrutamiento
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Baja	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar información acerca de las rutas empleadas por el sistema para enviar los paquetes ya sea dentro de la red local o hacia Internet.	
Observaciones:	

Tabla 2.33 Historia de Usuario-Generación de Tabla de Enrutamiento.

La tabla 2.34 muestra la historia de usuario correspondiente a la generación de estadística de la interfaz de red local:

Historia de Usuario	
Número: HUE07	Nombre: Generación de Estadística de la Interfaz de Red Local
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Baja	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar información detallada acerca de la interfaz de red local, aquí se podrá encontrar información como: nombre de la interfaz, información general de red asignada al dispositivo.	
Observaciones:	

Tabla 2.34 Historia de Usuario-Generación de Estadísticas de la Interfaz de Red Local.

- *Información Adicional*

La tabla 2.35 muestra la historia de usuario correspondiente a la generación de estadística de usuarios y tiempo activo del sistema:

Historia de Usuario	
Número: HUE08	Nombre: Generación de Estadística de Usuarios y Tiempo activo del Sistema
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Baja	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se proporcionará información general del servidor.	
Observaciones:	

Tabla 2.35 Historia de Usuario-Generación de Estadísticas de Usuarios y Tiempo activo del Sistema.

La tabla 2.36 muestra la historia de usuario correspondiente a la generación de información del kernel:

Historia de Usuario	
Número: HUE09	Nombre: Generación de Información del Kernel
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Baja	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se proporcionará información acerca del kernel del sistema operativo.	
Observaciones:	

Tabla 2.36 Historia de Usuario-Generación de Información del Kernel.

Tanto en éste módulo como en el módulo que corresponde al servicio de gráficas, se contará con una opción que permitirá reiniciar y apagar el servidor, ésta funcionalidad se encontrará visible en cada una de las funcionalidades que conforman éstos módulos.

MÓDULO DEL SERVICIO GRÁFICAS

La abreviatura empleada para éste módulo será: *HUG* (Nro.). La tabla 2.37 muestra la historia de usuario correspondiente a las gráficas del rendimiento del tráfico de red:

Historia de Usuario	
Número: HUG01	Nombre: Gráficas del Rendimiento del Tráfico de Red
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar el rendimiento tanto del tráfico de red entrante como saliente, de cada interfaz de red del servidor, éstas se podrán identificar fácilmente ya que cada tipo de tráfico se mostrará con un color diferente. En cada caso se mostrará el rendimiento de red por día. Además, se incluirá el valor máximo, mínimo, promedio y el último valor, correspondiente a cada tipo de tráfico. En cada eje se incluirán las unidades correspondientes.	
Observaciones: Los datos se actualizarán automáticamente.	

Tabla 2.37 Historia de Usuario-Gráficas del Rendimiento del Tráfico de Red.

La tabla 2.38 muestra la historia de usuario que corresponde a las gráficas del uso de memoria:

Historia de Usuario	
Número: HUG02	Nombre: Gráficas del Uso de Memoria
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar el uso de memoria total y usada en el servidor, cada una de éstas se mostrarán en las gráficas con un color diferente. Se presentará una gráfica que muestre el uso de memoria del servidor por día. Se incluirá el valor máximo, mínimo, promedio y el último valor tanto para el uso de memoria total como para la memoria usada. En cada eje se incluirán las unidades correspondientes.	
Observaciones: Los datos se actualizarán automáticamente.	

Tabla 2.38 Historia de Usuario-Gráficas del Uso de Memoria.

La tabla 2.39 muestra la historia de usuario correspondiente a las gráficas de la carga del CPU:

Historia de Usuario	
Número: HUG03	Nombre: Gráficas de la Carga del CPU
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se podrá visualizar la carga del CPU del servidor. Se presentará una gráfica que muestre la carga del CPU del servidor por día. Se incluirá el valor máximo, mínimo, promedio y el último valor de la carga de CPU. En cada eje se incluirán las unidades correspondientes.	
Observaciones: Los datos se actualizarán automáticamente.	

Tabla 2.39 Historia de Usuario-Gráficas de la Carga del CPU.

En cada uno de los módulos anteriores se contará con una opción que proporcionará ayuda acerca del manejo de la interfaz Web.

MÓDULO DE PÁGINA INICAL

La abreviatura empleada para éste módulo será: *HUB (Nro.)*. En la tabla 2.40 Se muestra la historia de usuario que corresponde a la página inicial o de bienvenida:

Historia de Usuario	
Número: HUB01	Nombre: Página Inicial
Modificación de Historia de Usuario(Nro. y Nombre): --	
Usuario: Administrador	Iteración Asignada: 1
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se proporcionará información acerca del estado del servidor. Este módulo podrá ser visualizado por todos los usuarios. Para poder ingresar a los demás módulos se requerirá de un nombre de usuario y una contraseña.	
Observaciones: Únicamente el administrador conocerá el nombre de usuario y contraseña.	

Tabla 2.40 Historia de Usuario-Página Inicial.

2.1.3.3 Requisitos de Rendimiento

En ésta sección se presentan los requerimientos tanto estáticos como dinámicos incluidos en el sistema.

Requisitos Estáticos:

- Se contará con un navegador web, el cual permitirá el acceso al sistema DAMA.
- Se contará con un sistema de autenticación por lo que solo el Administrador de la red tendrá acceso al sistema.
- El volumen de información a manejar es relativamente pequeño, debido a que se manejarán archivos de texto plano que no demandan gran cantidad de requerimientos.

Requisitos Dinámicos:

- En cada módulo se realizará una transacción por funcionalidad.
- La cantidad de datos a procesar por transacción es mínima debido a que se permitirá el acceso a un solo usuario, que en éste caso será el administrador de la red.

2.1.3.4 Restricciones de Diseño

- La interfaz web funcionará para una resolución mayor a 1024 por 768 píxeles.
- Dependiendo del navegador que se esté utilizando, la interfaz sufrirá pequeñas variaciones en su aspecto, por ejemplo el diseño de bordes, efectos de sombreado, etc., debido a que cada navegador interpreta de manera diferente al sistema, esto no afectará de ninguna manera la funcionalidad de cada módulo y del sistema DAMA en sí.
- El webmaster o administrador de la red deberá tener conocimientos básicos acerca de los servicios que se van a manejar a través de la interfaz Web, ya que si bien el sistema DAMA proporciona ayuda acerca de las funcionalidades de cada módulo, un modificación inapropiada en la configuración de los servicios podría poner en riesgo el buen funcionamiento del servidor.

2.1.3.5 Atributos del Sistema

A continuación se detallarán las propiedades referentes con la calidad del sistema:

Disponibilidad.- Debido a que el sistema DAMA se implementará en un servidor que estará activo permanentemente y que además permitirá realizar tareas que son comúnmente utilizadas en una red local, ésta aplicación se encontrará disponible las 24 horas los 7 días de la semana.

Seguridad.- El único usuario autorizado para acceder y hacer uso de las funcionalidades del sistema DAMA, será el webmaster o administrador de la red, debido a que éste proyecto está enfocado a éste tipo de usuario, para esto se

implementará un sistema de autenticación mediante el ingreso de un login y una contraseña, que serán solo de conocimiento del administrador, de ésta manera se evitará que otro tipo de usuario pueda acceder a la información del servidor o realizar cambios en la configuración de los servicios.

El webmaster o administrador de la red sólo tendrá acceso a los servicios disponibles a través de la interfaz Web, de ésta manera se evitará que se puedan realizar modificaciones sobre otro tipo de aspectos dentro del servidor que afecten su funcionamiento.

Portabilidad.- El sistema DAMA podrá visualizarse tanto en navegadores propios de Linux como de Windows, cabe recalcar que el sistema será implementado bajo el sistema operativo Linux.

El sistema podrá ser implementado en cualquier servidor que cumpla con las características de hardware y software analizadas en el presente capítulo.

CAPÍTULO 3

3 DESARROLLO E IMPLEMENTACIÓN DE LA INTERFAZ WEB

3.1 DESARROLLO E IMPLEMENTACIÓN

A continuación se realizará el desarrollo de la interfaz Web en base a la metodología de desarrollo XP, seleccionada en el capítulo 1 sección 1.8.7 y los requerimientos tanto hardware como software establecidos en el capítulo 2.

3.1.1 Planificación del Proyecto

3.1.1.1 Historia de Usuarios

Las historias de usuarios han sido detalladas en el capítulo 2 sección 2.1.3.2, ya que forman parte de la especificación de requisitos según el estándar IEEE 830-1998.

3.1.1.2 Planificación de Entregables

La tabla 3.1 muestra el formato acogido para la definición del plan de ejecución de las iteraciones, para la generación de un módulo (entregable).

Módulo	Nro.	Historia de Usuario	Tiempo Estimado		
			Fecha Inicio	Fecha Final	Horas Estimadas

Tabla 3.1 Formato de la tabla para la planificación de entregables.

En donde:

- **Módulo:** define el nombre del módulo.
- **Nro. :** corresponde al número de historia de usuario dentro del módulo correspondiente.
- **Fecha Inicio:** es la fecha en la cual empieza la iteración.
- **Fecha Final:** es la fecha en la cual termina la iteración.
- **Horas Estimadas:** es el número de horas empleadas en el desarrollo de cada iteración.

La tabla 3.2 muestra los entregables correspondientes a cada módulo:

Módulo	Nro.	Historia de Usuario	Tiempo Estimado		
			Fecha Inicio	Fecha Final	Horas Estimadas
Servicio DHCP	HUD01	Configuración Básica del Servicio DHCP	2010-05-03	2010-05-05	15
	HUD02	Configuración Dinámica del Servicio DHCP	2010-05-05	2010-05-07	15
	HUD03	Configuración Estática del Servicio DHCP	2010-05-10	2010-05-14	30
	HUD04	Direccionamiento IP	2010-05-17	2010-05-21	30
		<i>Posibles Nuevas Iteraciones</i>	2010-05-25	2010-05-25	10
Servicio Proxy - Caché	HUP01	Modificación de Opciones Generales	2010-05-25	2010-05-28	25
	HUP02	Generación de Listas de Control de Acceso	2010-05-31	2010-06-04	25
	HUP03	Generación de Políticas de Acceso	2010-06-07	2010-06-10	20
	HUP04	Manejo de Usuarios para el Servicio	2010-06-11	2010-06-16	20
		<i>Posibles Nuevas Iteraciones</i>	2010-06-17	2010-06-17	10
Servicio de Directorios Compartidos	HUS01	Visualización de Información General	2010-06-17	2010-06-21	15
	HUS02	Manejo de Usuarios para el Servicio	2010-06-22	2010-06-25	20
	HUS03	Adición de Directorios al Servicio	2010-06-28	2010-06-30	15
	HUS04	Modificación de las Características de los Directorios Compartidos	2010-07-01	2010-07-02	10
		<i>Posibles Nuevas Iteraciones</i>	2010-07-05	2010-07-05	10
Servicio de Filtrado Web	HUW01	Modificación de Opciones Generales	2010-07-05	2010-07-09	25
	HUW02	Filtrado por MIME	2010-07-12	2010-07-15	15
	HUW03	Filtrado por Extensión	2010-07-16	2010-07-20	15
	HUW04	Filtrado por Site	2010-07-21	2010-07-26	15
	HUW05	Filtrado por Frase	2010-07-27	2010-07-30	15
		<i>Posibles Nuevas Iteraciones</i>	2010-08-02	2010-08-02	10
Servicio de Firewall	HUF01	Generación de Zonas	2010-08-02	2010-08-04	25
	HUF02	Gen. Relación entre una Zona y una Interfaz de Red	2010-08-05	2010-08-09	15
	HUF03	Generación de Políticas de Acceso de alto nivel	2010-08-10	2010-08-12	15

	HUF04	Generación de Reglas de Acceso Específicas	2010-08-13	2010-08-17	15
	HUF05	Generación de Enmascaramiento	2010-08-18	2010-08-20	15
	<i>Posibles Nuevas Iteraciones</i>		2010-08-23	2010-08-23	10
Estado del Sistema Operativo	HUE01	Generación de Estadísticas de uso del Disco Duro	2010-08-23	2010-08-27	25
	HUE02	Generación de Estadísticas de Discos Duros Adicionales y Particiones	2010-08-30	2010-09-01	15
	HUE03	Generación de Estadísticas General de Memoria	2010-09-02	2010-09-06	15
	HUE04	Generación de Estadística de uso de CPU	2010-09-07	2010-09-10	20
	HUE05	Generación de Estadística de Interfaz con salida a Internet	2010-09-13	2010-09-15	15
	HUE06	Generación de Tabla de Enrutamiento	2010-09-16	2010-09-20	15
	HUE07	Generación de Estadística de la Interfaz de Red Local	2010-09-21	2010-09-22	10
	HUE08	Generación de Estadística de Usuarios y Tiempo activo del Sistema	2010-09-23	2010-09-24	10
	HUE09	Generación de Información del Kernel	2010-09-27	2010-09-28	10
		<i>Posibles Nuevas Iteraciones</i>		2010-09-29	2010-09-29
Gráficas	HUG01	Gráficas del Rendimiento del Tráfico de Red	2010-09-29	2010-10-05	25
	HUG02	Gráficas del Uso de Memoria	2010-10-06	2010-10-08	15
	HUG03	Gráficas de la Carga del CPU	2010-10-11	2010-10-13	15
	<i>Posibles Nuevas Iteraciones</i>		2010-10-14	2010-10-14	10
Página Inicial	HUB01	Página de Bienvenida al Sistema de Administración DAMA	2010-10-15	2010-10-15	5
Tiempo Total Estimado			121 días		690 horas

Tabla 3.2 Estimación de Entregables.

Las fechas de posibles iteraciones serán empleadas siempre y cuando el cliente solicite algún tipo de cambio sobre el entregable, caso contrario éste tiempo será utilizado en el desarrollo de la siguiente iteración.

3.1.1.3 Análisis de Iteraciones

A continuación se realizará el análisis de las iteraciones correspondientes a cada módulo del sistema.

- **Módulo del Servicio DHCP**

- **Primera Iteración.-** El cliente se encuentra satisfecho con la mayoría de las funcionalidades desarrolladas, pero solicita que en las configuraciones básica y dinámica se muestre información acerca de la configuración actual del servicio, por lo que se requiere de una nueva iteración. En la tabla 3.3 y 3.4 se muestran las nuevas historias de usuario para la modificación solicitada, el tiempo a emplear en la iteración estará dentro del rango establecido en la tabla 3.2.

La abreviatura empleada para estas historias de usuario será *HUDM* (Nro.), en donde *M* representa modificación.

Historia de Usuario	
Número: HUDM01	Nombre: Modificación en Configuración Básica DHCP
Modificación de Historia de Usuario(Nro. y Nombre): HUD01 - Configuración Básica del Servicio DHCP	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se solicita mostrar la configuración actual para el servicio. Se cargará en las diferentes entradas de texto la información actual.	
Observaciones: Se añadirá un mensaje informativo indicándole al cliente que la información que se muestra es la configurada en ese momento.	

Tabla 3.3 Iteración-Historia de Usuario-Modificación en Configuración Básica DHCP.

Historia de Usuario	
Número: HUDM02	Nombre: Modificación en Configuración Dinámica DHCP
Modificación de Historia de Usuario(Nro. y Nombre): HUD02 - Configuración Dinámica del Servidor DHCP	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se solicita mostrar la configuración actual para el servicio. Se cargará en las diferentes entradas de texto la información actual.	
Observaciones: Se añadirá un mensaje informativo indicándole al cliente que la información que se muestra es la configurada en ese momento.	

Tabla 3.4 Iteración-Historia de Usuario-Modificación en Configuración Dinámica DHCP.

- **Segunda Iteración.-** Al mostrar la solución al cliente, éste quedó satisfecho con los cambios realizados, por lo que se da por finalizado el desarrollo de éste módulo.
- **Módulo del Servicio Proxy – Caché**
 - **Primera Iteración.-** Al mostrar la solución al cliente, éste se mostró satisfecho, por lo que se da por finalizado el desarrollo de éste módulo, el tiempo estimado para la posible nueva iteración será empleado en el desarrollo inicial del siguiente módulo.
- **Módulo del Servicio de Directorios Compartidos**
 - **Primera Iteración:** El cliente se mostró satisfecho con la mayoría de las funcionalidades realizadas, pero se solicitó un cambio en el alineamiento de los elementos de ingreso de datos en Manejo de Usuarios. La tabla 3.5 muestra las nuevas historias de usuario para la modificación solicitada, el tiempo a emplear en la iteración estará dentro del rango establecido en la tabla 3.2.

La abreviatura empleada para ésta historia de usuario será *HUSM* (Nro.), en donde *M* representa modificación.

Historia de Usuario	
Número: HUSM01	Nombre: Modificación en Manejo de Usuarios para el Servicio
Modificación de Historia de Usuario(Nro. y Nombre): HUS02 - Manejo de Usuarios para el Servicio	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se solicita cambios en el alineamiento de los elementos de ingreso de datos.	
Observaciones:	

Tabla 3.5 Iteración-Historia de Usuario-Modificación en Manejo de Usuarios para el Servicio.

- **Segunda Iteración.-** Al mostrar la solución al cliente, quedó satisfecho con los cambios realizados, por lo que se da por concluido el desarrollo de éste módulo.
- **Módulo del Servicio de Filtrado Web**
 - **Primera Iteración.-** Al mostrar la solución al cliente, éste se mostró satisfecho con la mayoría de las funcionalidades realizadas, pero se solicitó un cambio en la descripción de opciones generales en donde existía ambigüedad en la información mostrada. La tabla 3.6 muestra la historia de usuario para la modificación solicitada, el tiempo a emplear en la iteración estará dentro del rango establecido en la tabla 3.2.

La abreviatura empleada para ésta historia de usuario será *HUWM* (Nro.), en donde *M* representa modificación.

Historia de Usuario	
Número: HUWM01	Nombre: Cambio en Modificación de Opciones Generales
Modificación de Historia de Usuario(Nro. y Nombre): HUW01 - Modificación de Opciones Generales	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se solicitó un cambio en la descripción de opciones generales. Se describirá cada una de las opciones de forma que sea entendible para el usuario.	
Observaciones:	

Tabla 3.6 Iteración-Historia de Usuario-Cambio en Modificación de Opciones Generales.

- **Segunda Iteración.-** El cliente se mostró satisfecho con los cambios realizados, por lo que se da por concluido el desarrollo de éste módulo.
- **Módulo del Servicio de Firewall**
 - **Primera Iteración.-** Al mostrar la solución al cliente, éste se mostró satisfecho con la mayoría de las funcionalidades realizadas, pero solicitó que se indicara qué líneas no se pueden remover de las áreas de texto. La tabla 3.7 muestra la historia de usuario para la modificación solicitada, el tiempo estimado para ésta iteración está dentro del rango establecido en la tabla 3.2.

La abreviatura empleada para ésta historia de usuario será *HUFM* (Nro.), en donde *M* representa modificación.

Historia de Usuario	
Número: HUFM01	Nombre: Modificación en Módulo del Servicio Firewall
Modificación de Historia de Usuario(Nro. y Nombre): HUF01, HUF02, HUF03, HUF04, HUF05	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se solicitó una información clara acerca de qué líneas no se pueden remover de las áreas de texto. Se especificará en la interfaz mensajes claros acerca de la información que no debe ser removida.	
Observaciones:	

Tabla 3.7 Iteración-Historia de Usuario-Modificación en Módulo del Servicio Firewall.

- **Segunda Iteración:** El cliente se mostró satisfecho con los cambios realizados, por lo que se da por concluido el desarrollo de éste módulo.
- **Módulo del Servicio de Estado del Sistema Operativo**
 - **Primera Iteración.-** Al mostrar la solución al cliente, éste solicitó que los módulos de disco duro, CPU y memoria cuenten con barras estadísticas, de forma que se pueda visualizar rápidamente los datos proporcionados en cada módulo. El tiempo estimado para el desarrollo de ésta iteración se encuentra establecido dentro del rango especificado en la tabla 3.2. En las tablas 3.8 y 3.9 se muestran las historias de usuario para la modificación solicitada.

La abreviatura empleada para estas historias de usuario será *HUEM* (Nro.), en donde *M* representa modificación.

Historia de Usuario	
Número: HUEM01	Nombre: Modificación en Módulo de Disco Duro
Modificación de Historia de Usuario(Nro. y Nombre): HUE01 - HUE02	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se contará con un gráfico por cada información proporcionada en éste módulo.	
Observaciones:	

Tabla 3.8 Iteración-Historia de Usuario-Modificación en Módulo de Disco Duro.

Historia de Usuario	
Número: HUEM02	Nombre: Modificación en Módulo de CPU y Memoria
Modificación de Historia de Usuario(Nro. y Nombre): HUE03 - HUE04	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Media	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se contará con un gráfico por cada información proporcionada en éste módulo.	
Observaciones:	

Tabla 3.9 Iteración-Historia de Usuario-Modificación en Módulo de CPU y Memoria.

- **Segunda Iteración.-** El cliente se mostró satisfecho con los cambios realizados, por lo que se da por concluido el desarrollo de éste módulo.
- **Módulo del Servicio de Gráficas**
 - **Primera Iteración.-** Al mostrar la solución al cliente, éste solicitó que tanto para la gráfica de rendimiento del tráfico de red, uso de memoria como para la de carga del CPU, se puedan visualizar también gráficas por hora, semana, mes y año, con un indicativo en cada una para poder identificar de qué gráfica se trata. En la tabla 3.10 se muestra la historia de usuario para la modificación solicitada.

La abreviatura empleada para ésta historia de usuario será *HUGM* (Nro.), en donde *M* representa modificación.

Historia de Usuario	
Número: HUGM01	Nombre: Modificación en Módulo de Gráficas
Modificación de Historia de Usuario(Nro. y Nombre): HUG01 - HUG02- HUG03	
Usuario: Administrador	Iteración Asignada: 2
Prioridad (Alta/Media/Baja): Alta	
Riesgo en Desarrollo (Alta/Media/Baja): Baja	
Descripción: Se presentará cinco tipos de gráficas tanto para rendimiento del tráfico de red, uso de memoria como para carga del CPU, éstas serán: gráfica por hora, por día, por semana, por mes y por año. Para cada gráfica se aplicará las mismas características descritas en las historias de usuario HUG01, HUG02 y HUG03.	
Observaciones: Los datos se actualizarán automáticamente.	

Tabla 3.10 Iteración-Historia de Usuario-Modificación en Módulo de Gráficas.

- **Segunda Iteración.-** El usuario se mostró satisfecho con las modificaciones realizadas, por lo que se da por terminado el desarrollo de éste módulo.

3.1.2 Velocidad del Proyecto y Programación en Pareja

Los aspectos relacionados con la velocidad del proyecto ya han sido detallados en la tabla 3.2 como en la sección 3.1.1.3 del presente capítulo, en cuanto al punto relacionado con la programación en pareja, a continuación se definen los nombres de quienes son los encargados de desarrollar todas las historias de usuario con el fin de obtener un sistema óptimo y de calidad.

Programadores Responsables:

- Buele Gualoto Christian Daniel.
- Ruiz Cerón Marianela.

3.1.3 Diseño

Para la fase de diseño se ha generado una hoja de estilo CSS común para todos los elementos que conformarán cada módulo dentro de la interfaz Web, dado que facilita el diseño y permite cumplir con una de las sugerencias de la metodología que busca simplicidad y eficiencia en el diseño. Las hojas de estilo CSS ya han sido definidas en el capítulo1 sección 1.4.7.

En ésta fase de desarrollo del proyecto no se emplearán las herramientas que sugiere la metodología, dado que su enfoque está orientado al desarrollo de software y no al diseño de páginas Web dinámicas, por lo que en ésta fase se emplearán mecanismos de diseño orientados al desarrollo de páginas Web dinámicas.

3.1.3.1 Estructura de Diseño Web

La elección de una correcta estructura de diseño Web es un aspecto fundamental en el desarrollo del presente proyecto, debido a que una correcta elección permitirá que el webmaster o administrador de la red pueda visualizar cada una de las funcionalidades en cada módulo de manera fácil y clara. Tanto para la planificación como para la elección de la estructura, se tomará en cuenta las historias de usuario desarrolladas en el capítulo 2 sección 2.1.3.2 y 3.1.1.3. A continuación se definen brevemente los diferentes tipos de estructuras:

- **Estructura de Árbol o Jerárquica.-** Como su nombre lo indica es una estructura que se encuentra conformada por niveles o jerarquías. A partir de una página raíz o principal se enlaza a páginas de niveles inferiores hasta llegar a las páginas del último nivel.

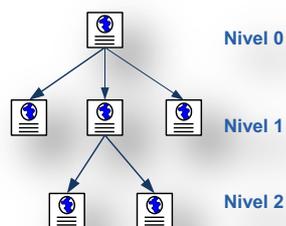


Figura 3.1 Estructura de Árbol o Jerárquica.

- **Estructura Lineal.-** En éste tipo de estructura las páginas se encuentran dispuestas en un mismo nivel, de modo que a partir de una página específica se disponen las demás páginas una tras otra, pudiendo ir a la siguiente página o a la anterior.



Figura 3.2 Estructura Lineal.

- **Estructura en Red.-** Este tipo de estructura no presenta ningún nivel de jerarquía, ya que todas las páginas se enlazan unas con otras como en una red, de ésta forma desde cada página podemos enlazarnos con cualquier otra del sistema. En ésta estructura es recomendable informar al usuario en qué página se encuentra, para evitar que éste se pierda en la navegación.

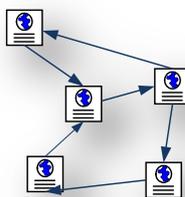


Figura 3.3 Estructura en Red.

Dentro de los diferentes tipos de estructuras, la que más se ajusta a las características del sistema DAMA, es la estructura de red, debido a que en el sistema se contará con un menú desde el cual el usuario puede dirigirse a cada uno de los módulos independientemente del módulo en el que se encuentre en ése momento. Se ha tomado en cuenta la recomendación de éste tipo de estructura, por lo que en cada módulo se encontrará la información necesaria que le indique al usuario en qué módulo se encuentra y de ésta forma no se pierda en la navegación.

3.1.3.1.1 *Diseño de la Estructura*

En ésta instancia se organizará la información que se va a presentar en el sistema DAMA, para esto se realizará un inventario de toda la información que se podrá visualizar, como primer paso se organizará la información por temas y a continuación se construirá gráficamente la estructura de la interfaz Web.

Este procedimiento se lo realiza con el fin de probar si el diseño realizado resulta fácilmente navegable para el usuario, así también se podrá comprobar si los contenidos y los enlaces mostrados son claros, fáciles de comprender y manejar.

Cabe mencionar que cuando se habla de *tema* se hace referencia a una página web, el cual representa a un módulo dentro del desarrollo, cada una de las funcionalidades que forman parte de un módulo corresponderán al contenido de cada tema.

- ***Organización por Temas:***

La organización de los temas se describe en el capítulo 2 sección 2.1.2.2.

- ***Diseño de la Estructura de la Interfaz Web:***

Una vez organizados los temas en cada módulo, se procede a realizar la estructura de diseño del sistema:

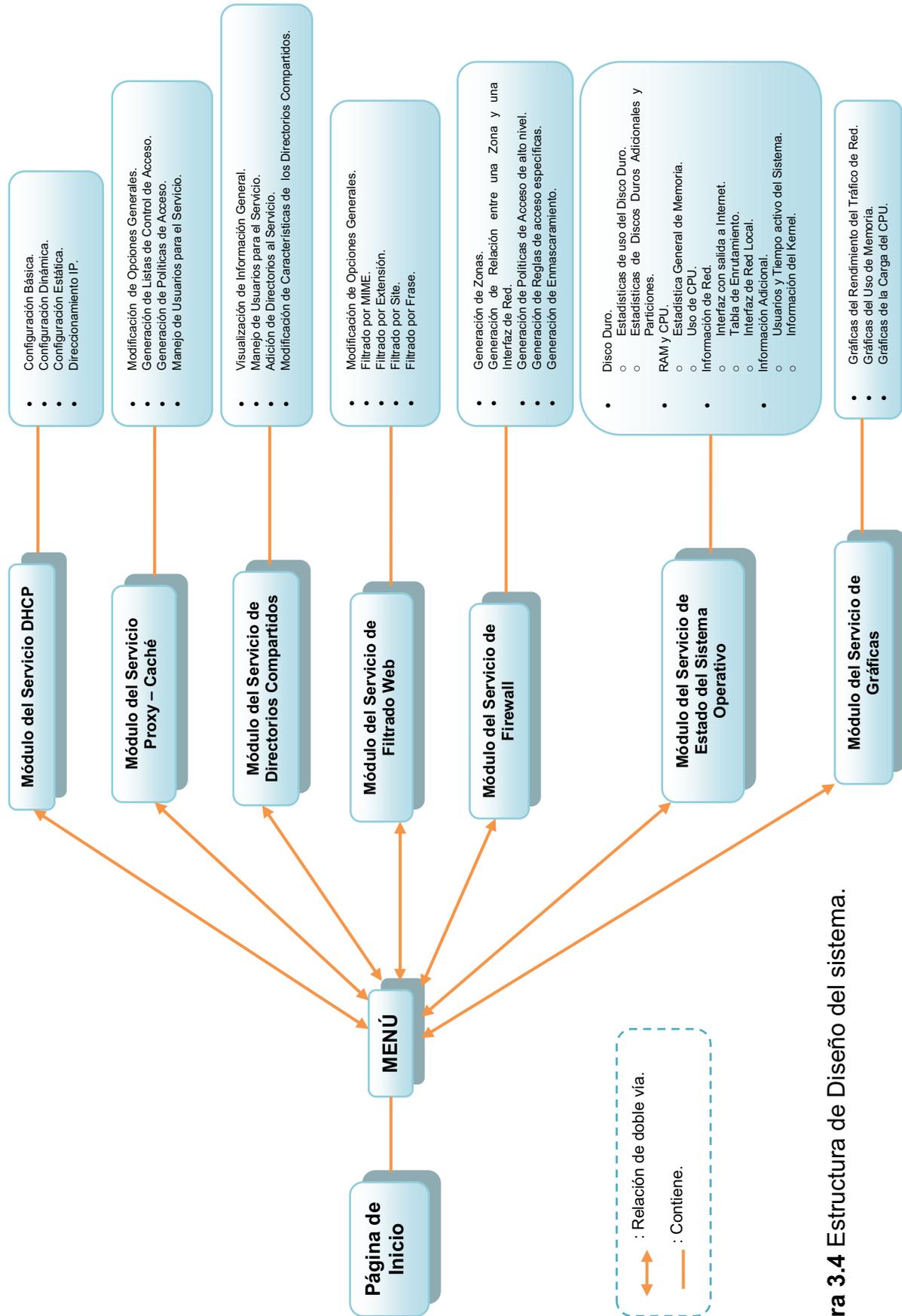


Figura 3.4 Estructura de Diseño del sistema.

⇄ : Relación de doble vía.
 — : Contiene.

La estructura de diseño del sistema muestra una navegabilidad clara, se tendrá inicialmente una página de bienvenida (página de inicio), ésta página contará con un menú desde el cual se podrá acceder a cualquier módulo de servicio disponible en el sistema. El menú estará siempre presente en cada uno de los módulos del sistema, con el fin de poder contar con un acceso inmediato a cualquier módulo.

3.1.3.2 Interfaces

A continuación se detallará cada una de las interfaces Web creadas en base a la organización de los temas descritos en la sección 3.1.3.1.1 del presente capítulo.

3.1.3.2.1 Página de Inicio

En la figura 3.5 se puede visualizar la página de bienvenida (página de inicio) que presenta el sistema DAMA:

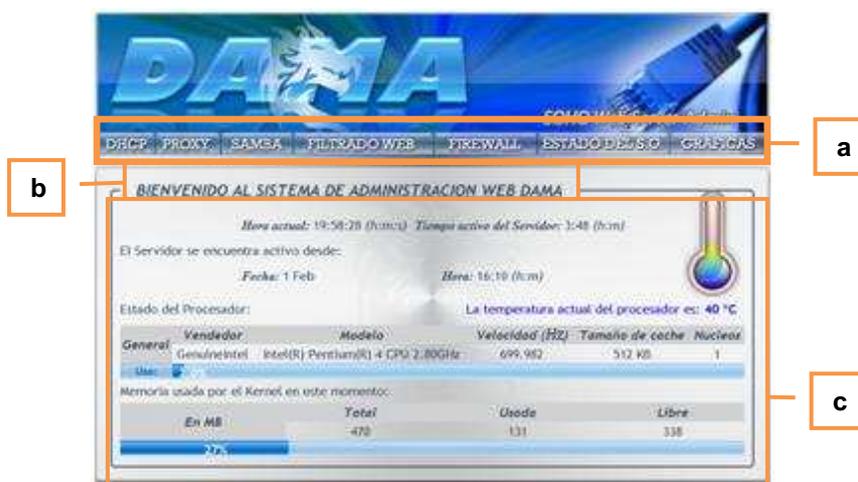


Figura 3.5 Interfaz-Página de Inicio del sistema DAMA.

En donde:

(a): En ésta sección se muestra el menú desde el cual se puede ingresar a cada uno de los módulos de servicios dentro del sistema.

(b): Se presenta un mensaje de bienvenida al sistema, de ésta forma se le indicará al usuario que se encuentra en la página inicial.

(c): En ésta sección de la página se podrá visualizar un resumen de la información acerca del estado actual del servidor.

3.1.3.2.2 Módulo del Servicio de DHCP

En la figura 3.6 se muestra parte de la interfaz que presenta el módulo del servicio de DHCP, en éste caso se puede visualizar la parte correspondiente a la configuración básica del servicio.



Figura 3.6 Interfaz - Configuración Básica.

En donde:

(a): Corresponde al menú principal desde el cual se puede acceder a los diferentes módulos del sistema.

(b): Es un acceso que permite obtener ayuda acerca del módulo actual, el acceso variará dependiendo del módulo en el que el usuario se encuentre, éste se encontrará presente en todas las secciones del módulo.

(c): Es un submenú que permite reiniciar los servicios disponibles en el sistema. Cabe mencionar que éste submenú se encontrará presente siempre en el módulo ya que es una funcionalidad importante dentro del sistema.

(d): Es un submenú para la navegabilidad hacia secciones dentro del módulo.

(e): Corresponde a la configuración básica del servicio DHCP según las especificaciones presentes en la historia de usuario de la tabla 2.6 del capítulo 2.

La figura 3.7 muestra la sección que forma parte del presente módulo y que corresponde a la configuración dinámica del servicio DHCP.

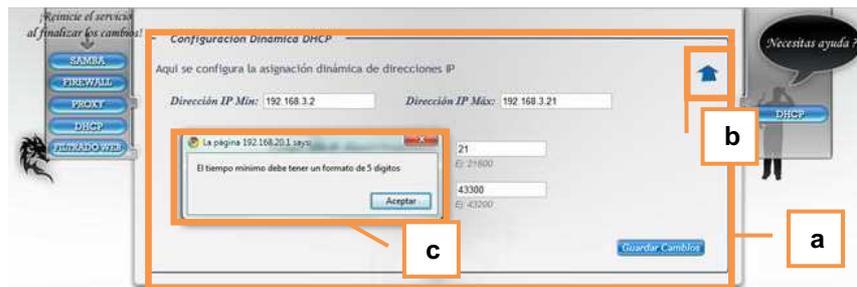


Figura 3.7 Interfaz - Configuración Dinámica.

En donde:

(a): Corresponde a la configuración dinámica del servicio DHCP según las especificaciones presentes en la historia de usuario de la tabla 2.7 del capítulo 2.

(b): Es un enlace que permite dirigir al usuario de la sección actual hacia el menú principal, para que pueda acceder a otros módulos o a su vez mediante el submenú a otras secciones dentro del módulo. Éste enlace se encontrará presente en todas las secciones del módulo.

(c): Corresponde a uno de los mensajes de alerta generados en base a la descripción de las observaciones detalladas en cada una de las historias de usuario del módulo del servicio DHCP.

En la figura 3.8 y 3.9 se muestran las secciones que corresponden a la configuración estática y al direccionamiento IP dentro del módulo del servicio de DHCP.

En donde:

(a) y (b): Corresponden a la configuración estática y direccionamiento IP del servicio DHCP según las especificaciones presentes en las historias de usuario de las tablas 2.8 y 2.9 respectivamente, detalladas en el capítulo 2.



Figura 3.8 Interfaz - Configuración Estática.

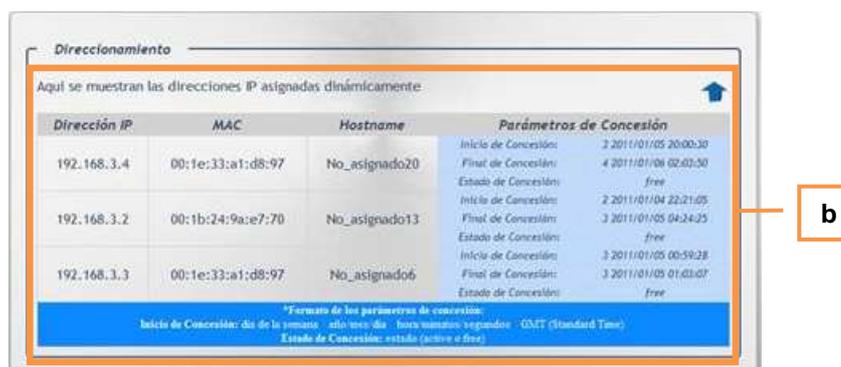


Figura 3.9 Interfaz - Direccionamiento IP.

3.1.3.2.3 Módulo del Servicio Proxy / Caché

La figura 3.10 muestra parte de la interfaz que presenta el servicio Proxy – Caché, en éste caso se visualiza la sección correspondiente a la modificación de opciones generales.

En donde

(a), (b) y (c): Como se puede observar en la figura 3.10, tanto el menú principal como los submenús que corresponden al reinicio de los servicios y a la ayuda, se encuentran presentes en todos los módulos.

(d): Es un submenú que permite la navegabilidad hacia las diferentes secciones del presente módulo.

(e): Corresponde a la visualización y modificación de opciones generales, según las especificaciones descritas en la historia de usuario de la tabla 2.10 del capítulo 2.

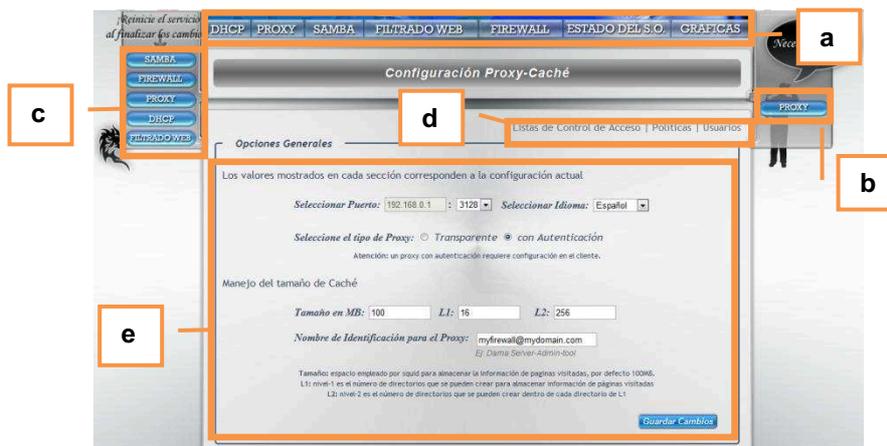


Figura 3.10 Interfaz - Opciones Generales.

En la figura 3.11 se muestra la sección que corresponde a la generación de listas de control de acceso:



Figura 3.11 Interfaz - Lista de Control de Acceso.

En donde:

(a): Corresponde a las opciones de generación de listas de control de acceso detalladas en la historia de usuario de la tabla 2.11 del capítulo 2.

(b): Es un enlace por medio del cual el usuario se puede dirigir desde la sección actual hacia el menú principal, para que pueda acceder a otros módulos o mediante el submenú a otras secciones. Éste enlace se encontrará presente en todas las secciones del presente módulo.

(c): Corresponde a uno de los mensajes de alerta generados en base a la descripción de las observaciones detalladas en cada una de las historias de usuario del módulo del servicio Proxy - Caché.

Las figuras 3.12 y 3.13 muestran las secciones que corresponden a la generación de políticas de acceso y al manejo de usuarios. En donde (a) y (b), corresponden a cada una de las opciones detalladas en las tablas 2.12 y 2.13 del capítulo 2, para la generación de políticas de acceso y manejo de usuarios respectivamente.



Figura 3.12 Interfaz - Políticas de Acceso.



Figura 3.13 Interfaz - Manejo de Usuarios.

3.1.3.2.4 Módulo del Servicio de Directorios Compartidos

La figura 3.14 muestra la sección que corresponde a la visualización de información general del servicio de directorios compartidos:



Figura 3.14 Interfaz - Visualización de Información General.

En donde:

(a), (b) y (c): Como se puede observar tanto el menú principal como los submenús que corresponden a la ayuda y reinicio de los servicios se encuentran presentes en todos los módulos.

(d): Es un submenú que permite la navegabilidad hacia las diferentes secciones dentro del módulo.

(e): Corresponde a las opciones de visualización de información general según las especificaciones descritas en la historia de usuario de la tabla 2.14 del capítulo 2.

En la figura 3.15 se muestra la sección correspondiente al manejo de usuarios del servicio de directorios compartidos. En donde:

(a): Corresponde a las opciones de manejo de usuarios conforme a las especificaciones descritas en la historia de usuario de la tabla 2.15 del capítulo 2.

(b): Es uno de los mensajes de alerta generados en base a la descripción de las observaciones detalladas en cada una de las historias de usuario del módulo del servicio de Directorios Compartidos.

(c): Es un enlace que permite al usuario dirigirse desde la sección actual hacia el menú principal para acceder a otros módulos o mediante el submenú a otras secciones. Éste enlace se encontrará presente en todas las secciones de éste módulo.

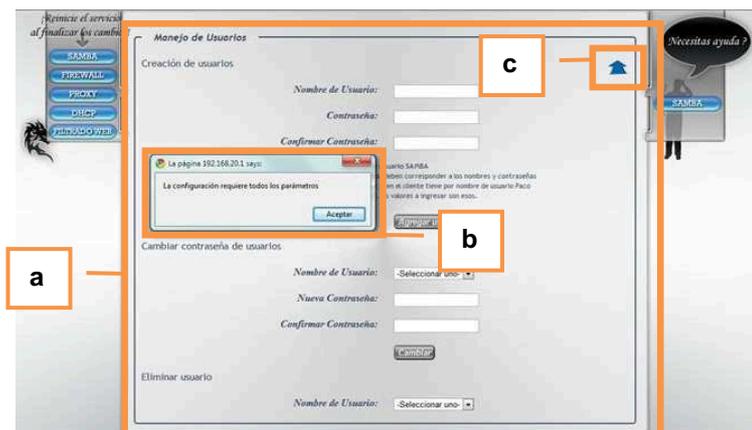


Figura 3.15 Interfaz - Manejo de Usuarios.

En las figuras 3.16 y 3.17 se muestran las secciones que permiten agregar y modificar directorios compartidos. En donde (a) y (b) corresponden a las opciones descritas en las historias de usuario detalladas en las tablas 2.16 y 2.17 del capítulo 2, para añadir directorios y modificarlos respectivamente.

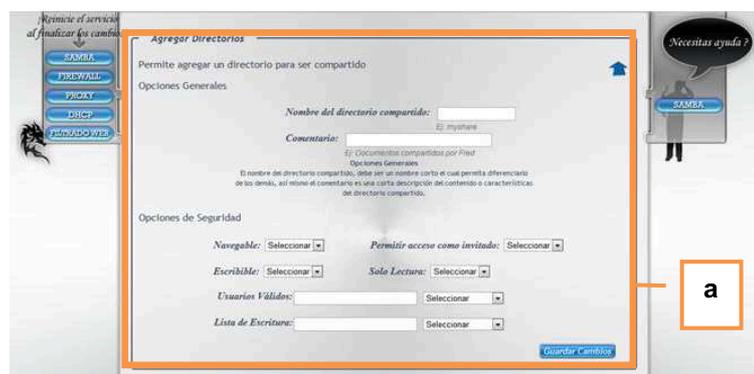


Figura 3.16 Interfaz - Agregar un Directorio.

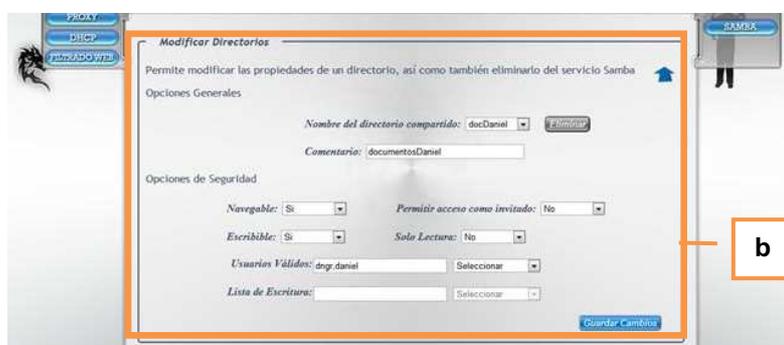


Figura 3.17 Interfaz - Modificar las Características de los Directorios Compartidos.

3.1.3.2.5 Módulo del Servicio de Filtrado Web

La figura 3.18 muestra la sección que corresponde a la visualización y modificación de opciones generales. En donde:

(a), (b) y (c): Aquí se puede visualizar tanto el menú principal como los submenús que corresponden a la ayuda y reinicio de servicios. Estas opciones se encontrarán presentes en cada uno de los módulos.

(d): Es un submenú que le permite al usuario dirigirse de la sección actual hacia cualquier sección dentro del módulo.

(e): Corresponde a cada una de las opciones que permiten la visualización y modificación de las opciones generales del servicio de filtrado web, según las especificaciones descritas en la historia de usuario de la tabla 2.18 del capítulo 2.



Figura 3.18 Interfaz - Modificación de Opciones Generales.

La figura 3.19 muestra la sección que permite realizar filtrado web por MIME.

En donde:

(a): Corresponde a la sección que permite realizar el filtrado web por MIME, según las especificaciones descritas en la historia de usuario de la tabla 2.19 del capítulo 2.

(b): Es uno de los mensajes de alerta generados en base a la descripción de las observaciones que se detallaron en cada una de las historias de usuario del módulo del servicio de Filtrado Web.

(c): Es un enlace que le permite al usuario dirigirse desde la sección actual hacia el menú principal para acceder a otros módulos o mediante el submenú hacia otras secciones del presente módulo. Este enlace se encuentra presente en cada una de las secciones de éste módulo.



Figura 3.19 Interfaz - Filtrado por MIME.

En las figuras 3.20, 3.21 y 3.22 se muestran las secciones que corresponden a:

(a): Filtrado web por extensión, desarrollado en base a las especificaciones descritas en la historia de usuario en la tabla 2.20 del capítulo 2.

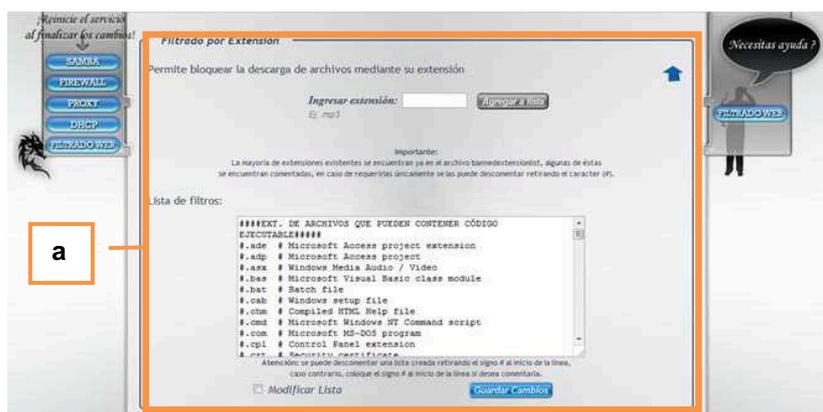


Figura 3.20 Módulo Servicio de Filtrado Web – Filtrado por Extensión.

(b): Filtrado por Site, según las especificaciones descritas en la historia de usuario en la tabla 2.21 del capítulo 2.

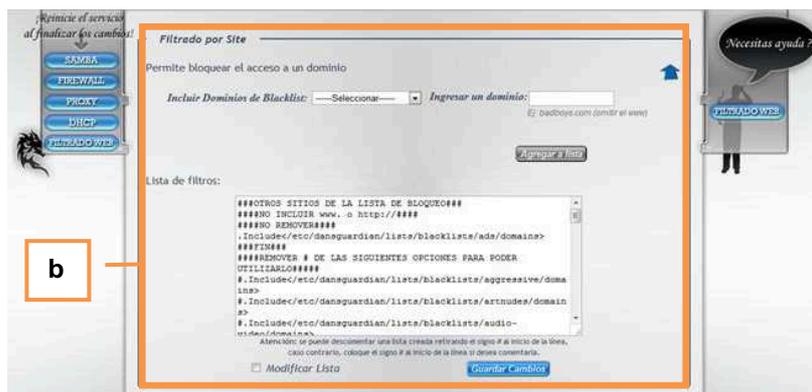


Figura 3.21 Interfaz - Filtrado por Site.

(c): Filtrado por Frase, desarrollado en base a las especificaciones descritas en la historia de usuario en la tabla 2.22 del capítulo 2.

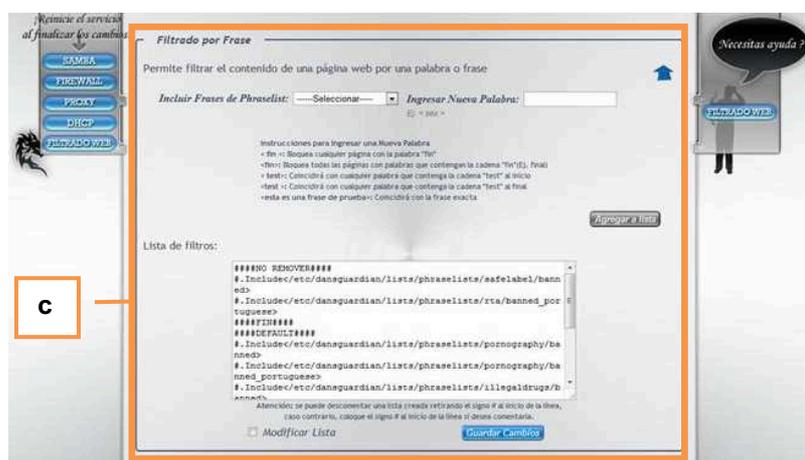


Figura 3.22 Interfaz - Filtrado por Frase.

3.1.3.2.6 Módulo del Servicio de Firewall

La figura 3.23 muestra la sección que corresponde a la generación de zonas del servicio de Firewall. En donde:

(a), (b) y (c): Aquí se puede visualizar tanto el menú principal como los submenús de ayuda y reinicio de servicios. Como se puede observar éstas opciones se encuentran presente en todos los módulos del sistema.

(d): Es un submenú que le permite al usuario dirigirse desde la sección actual hacia las diferentes secciones del módulo actual.

(e): Son cada una de las opciones que permiten la generación de zonas, según las especificaciones descritas en la historia de usuario en la tabla 2.23 del capítulo 2.

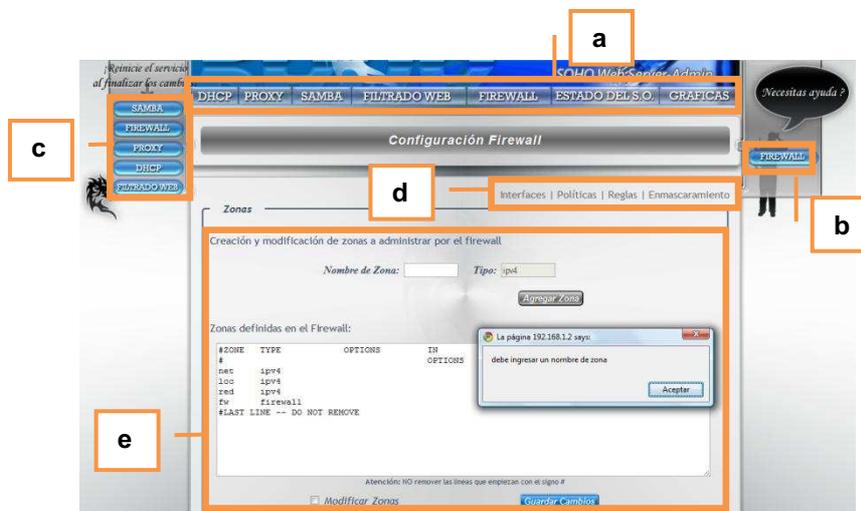


Figura 3.23 Interfaz - Generación de Zonas.

En la figura 3.24 se muestra la sección que permite la creación de relaciones entre una zona y una interfaz de red. En donde:

(a): Corresponde a cada una de las opciones que permiten la creación de relaciones entre una zona y una interfaz de red, según las especificaciones descritas en la historia de usuario en la tabla 2.24 del capítulo 2.

(b): Es uno de los mensajes de alerta generados según la descripción de las observaciones que se detallaron en cada una de las historias de usuario del módulo del servicio de Firewall.

(c): Es un enlace que le permite al usuario dirigirse desde la sección actual hacia el menú principal para acceder a otros módulos o mediante el submenú a otras secciones del módulo actual.

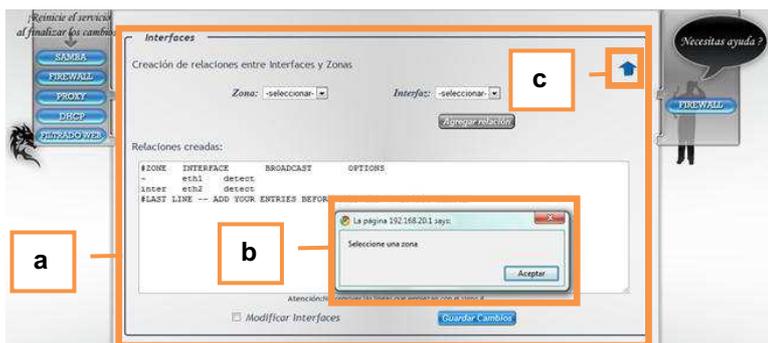


Figura 3.24 Interfaz - Generación de Relación entre una Zona y una Interfaz de Red.

En las figuras 3.25, 3.26 y 3.27, se pueden visualizar las secciones que corresponden a:

(a): Generación de políticas de alto nivel, desarrollado en base a las especificaciones descritas en la historia de usuario en la tabla 2.25 del capítulo 2.



Figura 3.25 Interfaz - Generación de Políticas de Acceso de alto nivel.

(b): Generación de reglas de acceso, desarrollado según las especificaciones descritas en la historia de usuario en la tabla 2.26 del capítulo 2.



Figura 3.26 Interfaz - Generación de Reglas de Acceso específicas.

(c): Generación de enmascaramiento, desarrollado en base a las especificaciones descritas en la historia de usuario en la tabla 2.27 del capítulo 2.

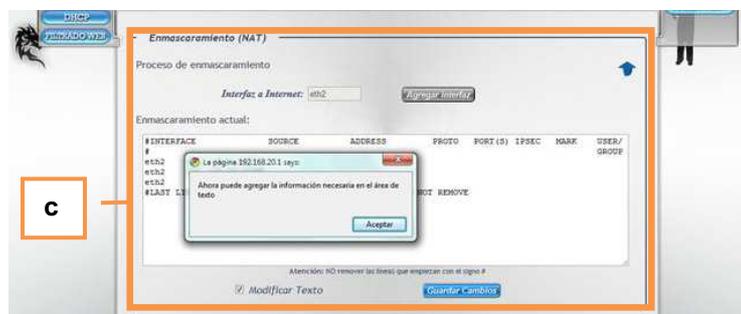


Figura 3.27 Interfaz - Generación de Enmascaramiento.

3.1.3.2.7 Módulo del Servicio de Estado del Sistema Operativo

A continuación se muestra cada una de las interfaces desarrolladas para el módulo correspondiente al servicio de Estado del Sistema Operativo.

- **Disco Duro**

En la figura 3.28 se muestra la interfaz desarrollada para la sección de estado del Disco Duro. En donde:

(a) y (b): Corresponden tanto al menú principal como al submenú que permite reiniciar los servicios. Estas opciones se encuentran siempre presentes en todos los módulos del sistema.

(c): Es el submenú que permite reiniciar y apagar el servidor. Este se encontrará presente tanto en el módulo de servicio de estado del sistema operativo como en el módulo de servicio de gráficas.

(d): Es un submenú que le permite al usuario dirigirse desde la sección actual hacia la segunda sección de éste módulo, en caso de existir particiones o discos duros adicionales.

(e): Corresponde a la sección en donde se puede visualizar estadísticas de uso del Disco Duro, en base a las especificaciones descritas en la historia de usuario en la tabla 2.28 del capítulo 2.

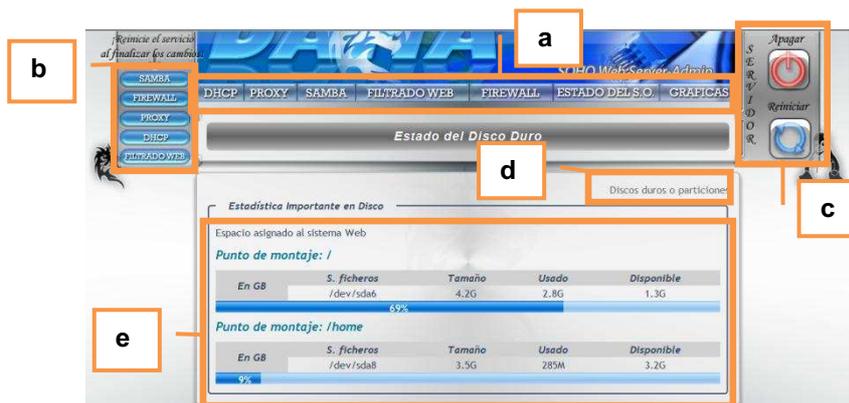


Figura 3.28 Interfaz - Estadísticas de uso del Disco Duro.

La interfaz que corresponde a discos duros adicionales o particiones, mantiene el mismo formato que la figura 3.28, por ésta razón no se la ha incluido.

- **RAM y CPU**

Las figuras 3.29 y 3.30 muestran las interfaces desarrolladas tanto para visualizar estadísticas de memoria como para uso de CPU. En donde:

(a): Corresponde a la sección que permite visualizar estadísticas del estado de la memoria RAM, según las especificaciones descritas en la historia de usuario en la tabla 2.30 del capítulo 2.

(b): Es un submenú que le permite al usuario trasladarse desde la sección actual, hacia la segunda sección del módulo actual.

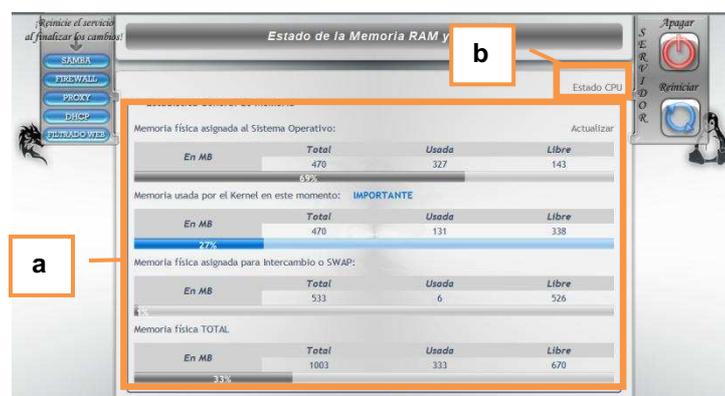


Figura 3.29 Interfaz - Estadística General de Memoria.

(c): Corresponde a la sección que permite visualizar las estadísticas acerca del estado del CPU, en base a las especificaciones detalladas en la historia de usuario en la tabla 2.31 del capítulo 2.

(d): Es un enlace que le permite al usuario dirigirse de la sección actual hacia la sección superior del módulo actual.



Figura 3.30 Interfaz - Uso de CPU.

- **Información de Red**

La figura 3.31 muestra la interfaz desarrollada que permite visualizar la información de red. En donde:

(a), (b) y (c): Como se puede observar tanto el menú principal como los submenús que permiten reiniciar los servicios y apagar o reiniciar el servidor, se encuentran presentes siempre tanto en el módulo de estado del sistema operativo como en el módulo de servicio de gráficas.

(d): Es un submenú que le permite al usuario dirigirse de la sección actual hacia la sección de tabla de enrutamiento o a la de interfaz local.

(e): Corresponde a la sección que permite visualizar información acerca de la interfaz de red con salida a Internet, según las especificaciones descritas en la historia de usuario en la tabla 2.32 del capítulo 2.



Figura 3.31 Interfaz - Información de Red.

Las interfaces que corresponden tanto a tabla de enrutamiento como Interfaz local, mantienen el mismo formato, por ésta razón no se ha incluido las figuras que corresponden a sus interfaces.

- **Información Adicional**

En la figura 3.32 se muestra la interfaz desarrollada que permite visualizar información adicional acerca del estado del sistema operativo. En donde:

(a), (b) y (c): Como se puede observar tanto el menú principal como los submenús que permiten reiniciar los servicios y apagar o reiniciar el servidor, se encuentran siempre presentes tanto en el módulo del servicio de estado del sistema operativo como en el módulo de servicio de gráficas.

(d): Es un submenú que le permite al usuario dirigirse desde la sección actual hacia la sección del módulo actual que proporciona información del kernel.

(e): Corresponde a la sección del módulo actual, en donde se puede visualizar información acerca de los usuarios y tiempo activo del sistema, según las especificaciones detalladas en la historia de usuario en la tabla 2.35 del capítulo 2.

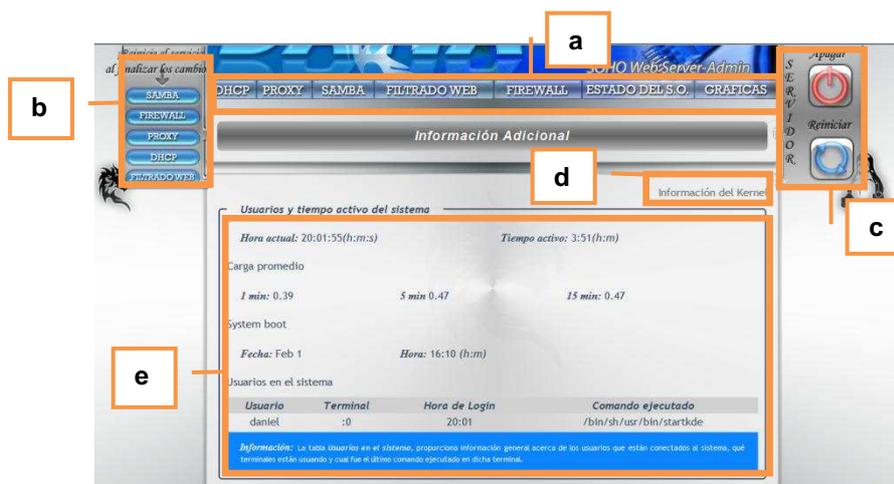


Figura 3.32 Interfaz - Información Adicional.

La sección que corresponde a información del Kernel, tiene el mismo formato que la figura 3.32, razón por la cual no se ha incluido la gráfica correspondiente a su interfaz.

3.1.3.2.8 Módulo del Servicio de Gráficas

A continuación se presenta las diferentes interfaces desarrolladas para el módulo de servicio de gráficas.

- **Rendimiento de Red**

En la figura 3.33 se muestra la interfaz desarrollada, que permite visualizar las gráficas del rendimiento del tráfico de red. En donde:

(a), (b) y (c): Como se puede observar tanto el menú principal como los submenús que permiten reiniciar los servicios y apagar o reiniciar el servidor, se encuentran presentes siempre en los módulos de estado del sistema operativo y servicio de gráficas.

(d): En ésta sección se puede visualizar las gráficas del rendimiento del tráfico de red, según las especificaciones descritas en la historia de usuario en la tabla 2.37 del capítulo 2.

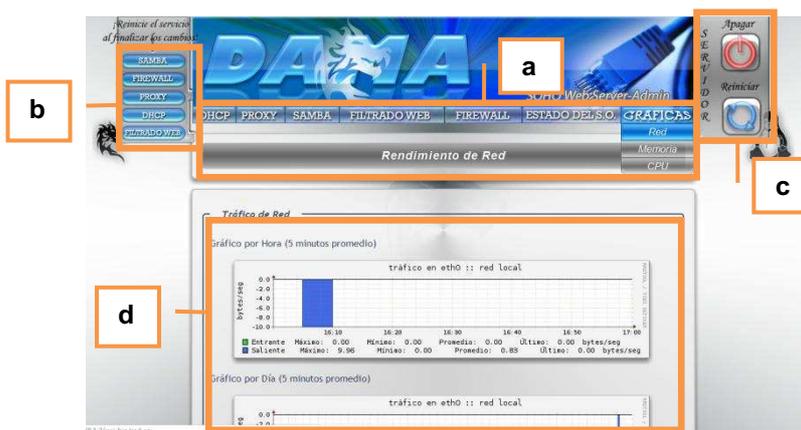


Figura 3.33 Interfaz - Rendimiento de Red.

- **Uso de Memoria**

En la figura 3.34 se muestra la interfaz desarrollada, que permite visualizar las gráficas del uso de memoria del servidor. En donde (a) corresponde a la sección en donde se muestran la gráficas, según las especificaciones detalladas en la historia de usuario en la tabla 2.38 del capítulo 2.



Figura 3.34 Interfaz - Uso de Memoria.

- **Carga de CPU**

En la figura 3.35 se muestra la interfaz desarrollada, que permite visualizar la carga del CPU del servidor. En donde (a) corresponde a la sección en donde se muestran las gráficas según las especificaciones descritas en la historia de usuario en la tabla 2.39 del capítulo 2.

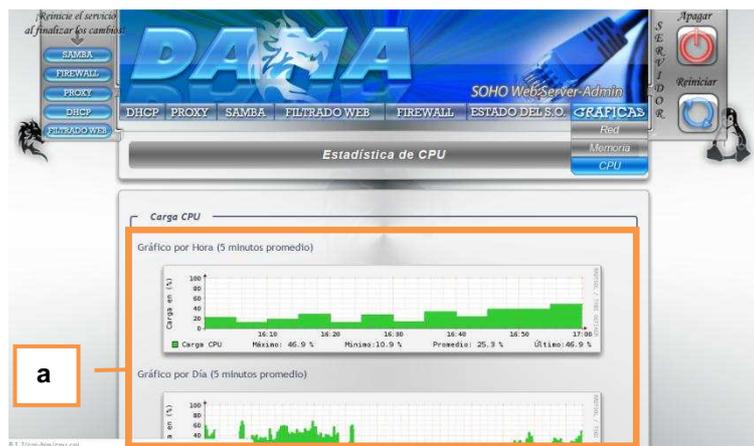


Figura 3.35 Interfaz - Carga de CPU.

3.1.4 Codificación

Para el desarrollo de las funcionalidades presentes en cada uno de los módulos que conforman el sistema DAMA, se ha empleado el lenguaje de programación Perl. En esta sección se explicará la lógica empleada en cada una de las funcionalidades implementadas en los módulos mediante el uso de diagramas de flujo, es decir la forma cómo el sistema se comporta frente a las actividades que realiza el usuario en cada una de las secciones dentro de un módulo. Además se describirá parte del código que se considera relevante dentro del proceso de desarrollo.

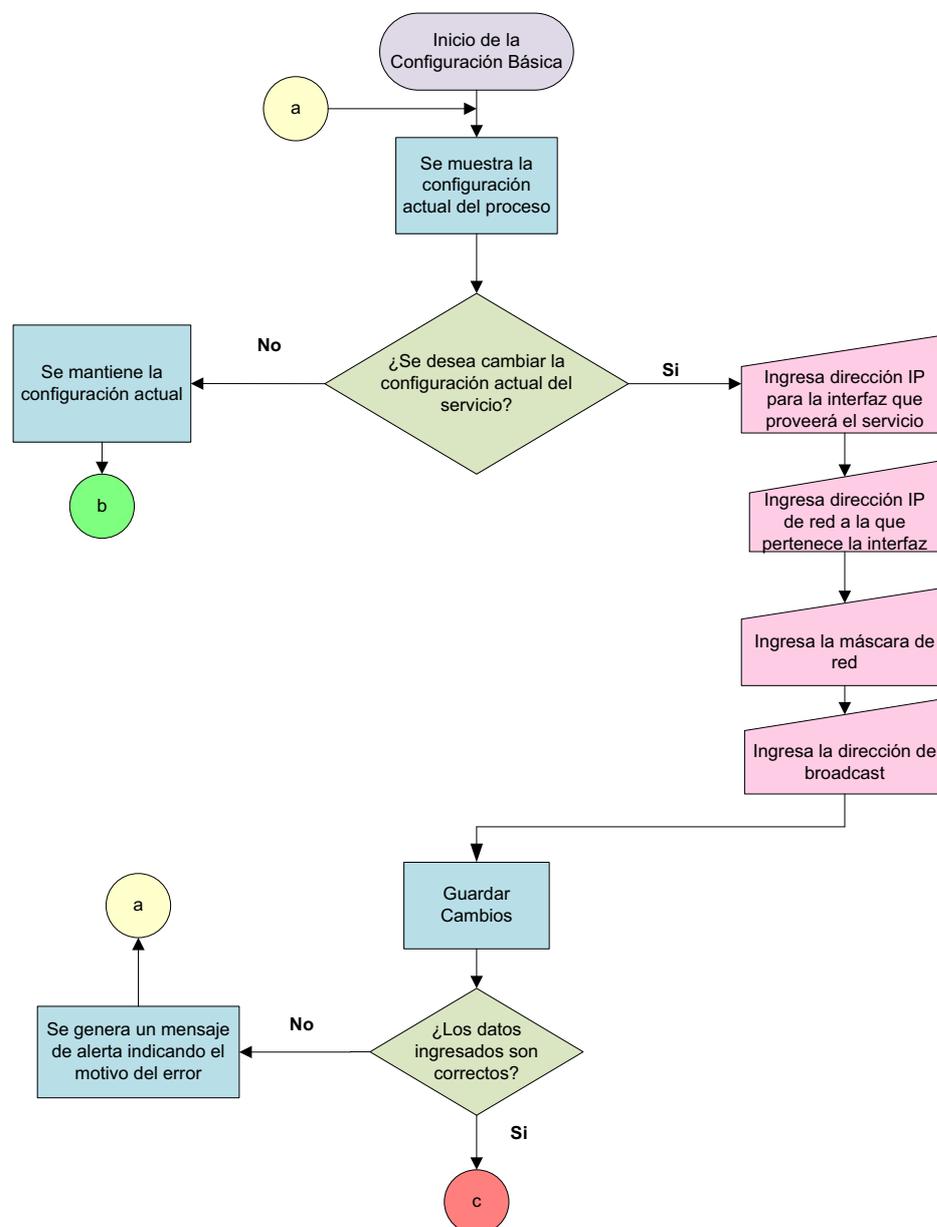
El sistema de administración web DAMA, se encuentra compuesto por los siguientes módulos:

- Módulo de Servicio DHCP.
- Módulo de Servicio Proxy –Caché.
- Módulo de Servicio de Directorios Compartidos
- Módulo de Servicio de Filtrado Web.
- Módulo de Servicio de Firewall.
- Módulo de Servicio de Estado del Sistema Operativo.
- Módulo de Servicio de Gráficas.
- Módulo de Página de Inicio.

3.1.4.1 Módulo de Servicio DHCP

Este módulo consta de las siguientes secciones:

- Configuración Básica DHCP:** Esta sección permite configurar los parámetros básicos del servicio. Dentro de los parámetros que se permite configurar en ésta sección están: dirección IP asignada a la interfaz que proveerá el servicio, la dirección IP de la red a la cual pertenece la interfaz, la máscara de red y por último la dirección broadcast de la red. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



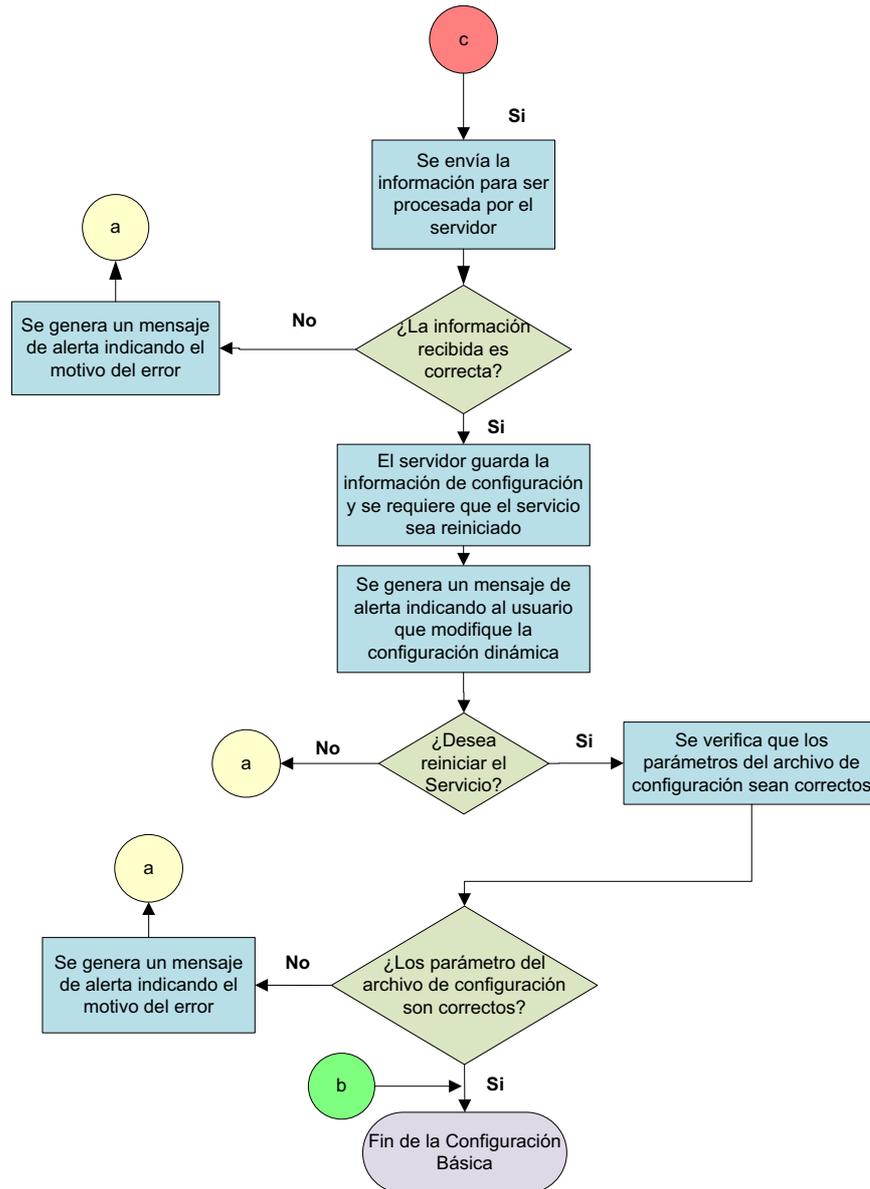


Figura 3.36 Diagrama de Flujo – Configuración Básica DHCP.

- Configuración Dinámica DHCP:** Permite configurar la asignación dinámica de direcciones IP. Dentro de los parámetros que permite configurar están: el rango de direcciones IP máxima y mínima, el tiempo máximo y mínimo de alquiler de la dirección IP, cabe mencionar que éste valor está dado en segundos. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.

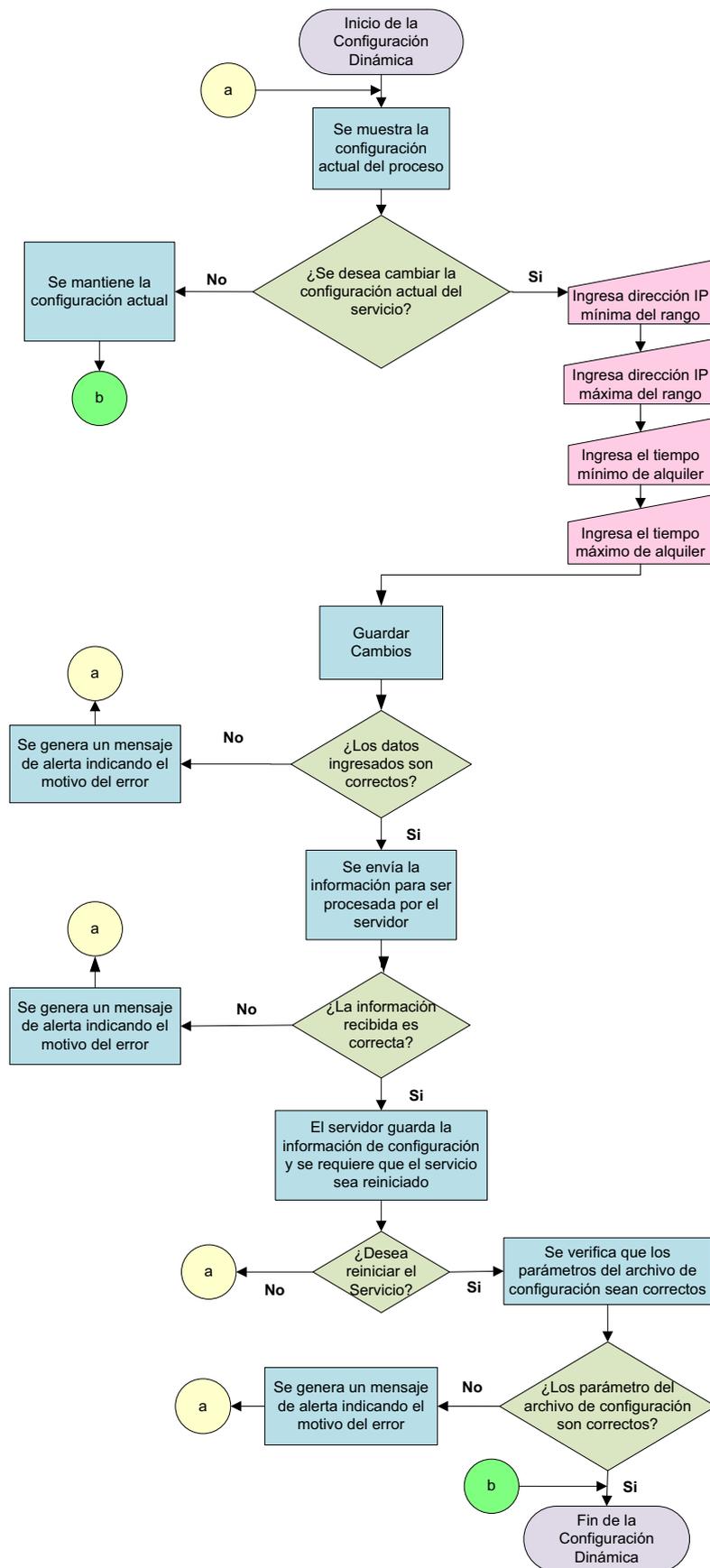
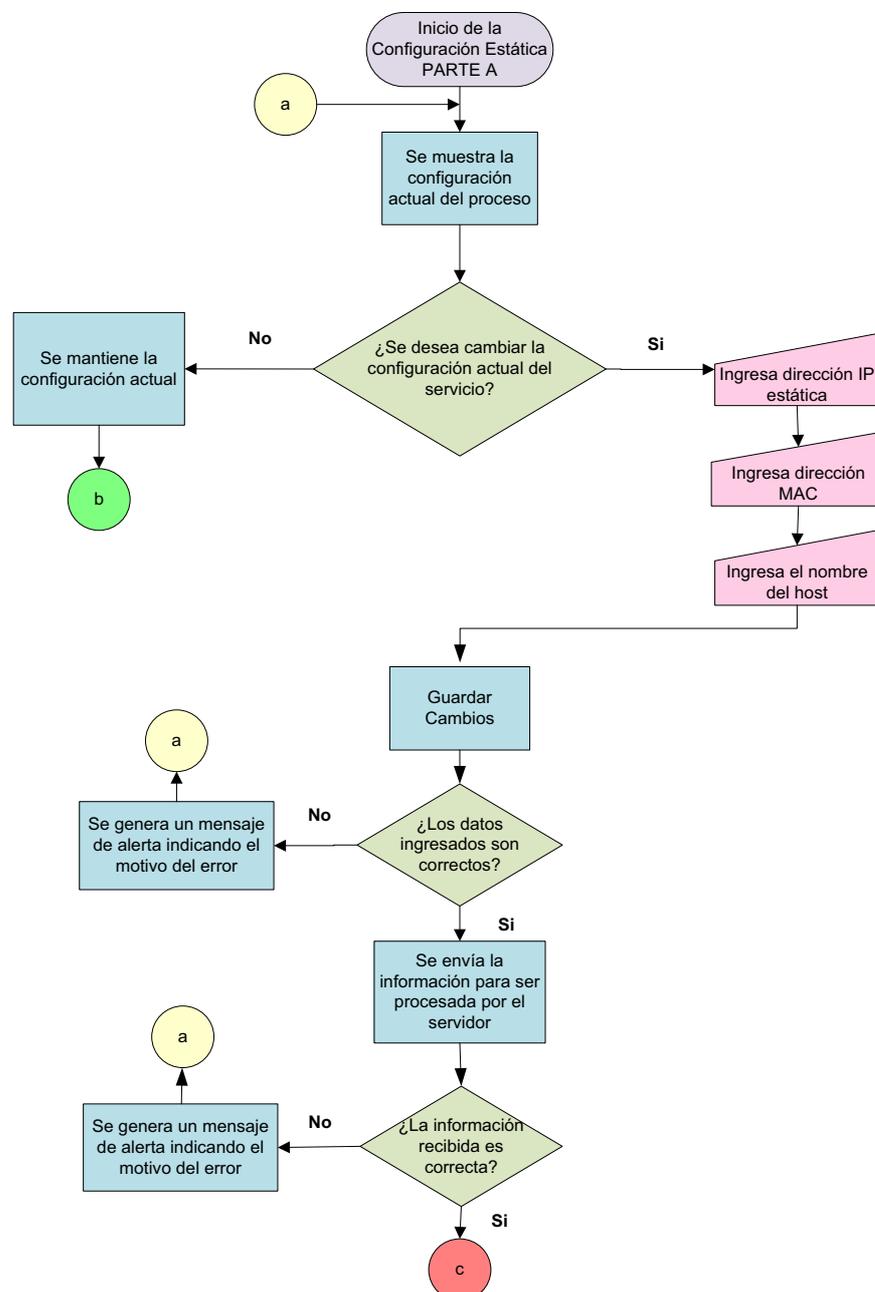


Figura 3.37 Diagrama de Flujo – Configuración Dinámica DHCP.

- Configuración Estática DHCP:** Permite configurar los parámetros para asignar una dirección IP fija a un host conocido. Con el fin de realizar de forma clara los diagramas de flujo, se ha dividido ésta sección en dos partes A y B, dentro de los parámetros que se permite configurar en la parte A están: dirección IP, dirección MAC y nombre del host. En la parte B se mostrarán aquellos host que ya cuentan con una asignación estática, además se podrá seleccionar a un host que ya cuente con una dirección IP asignada por el servicio DHCP para asignarle esa dirección de manera permanente, teniendo la opción de eliminar las asignaciones estáticas existentes.



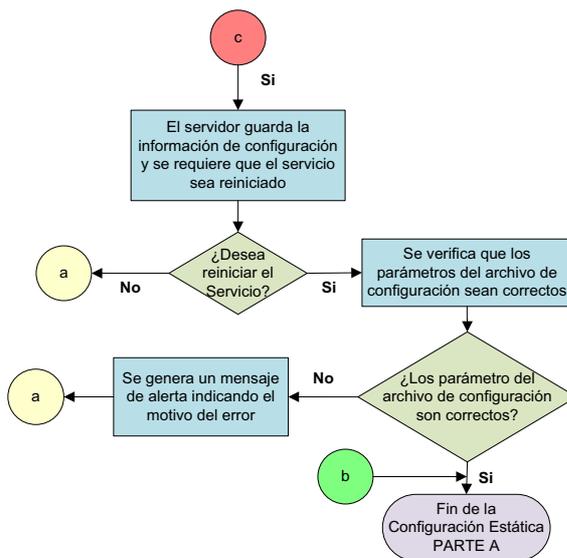


Figura 3.38 Diagrama de Flujo – Configuración Estática DHCP – Parte A.

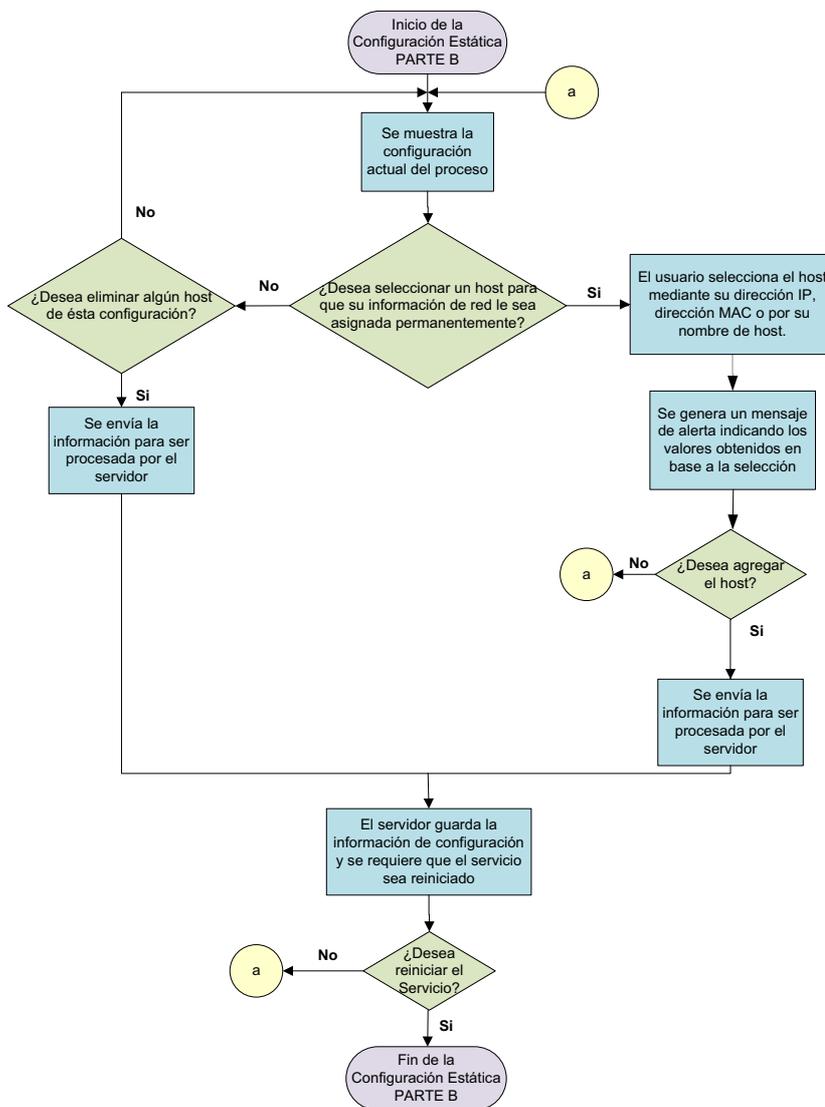


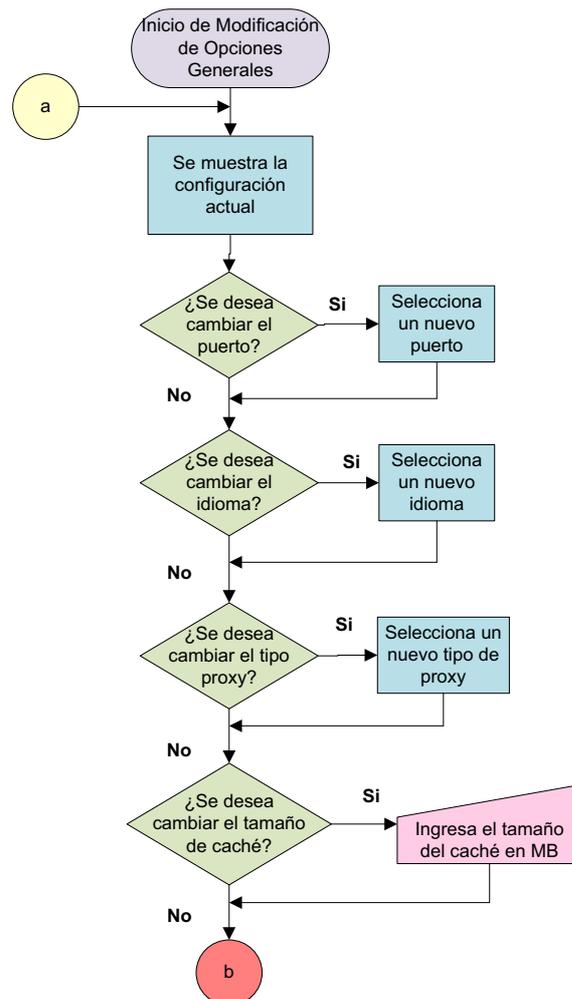
Figura 3.39 Diagrama de Flujo – Configuración Estática DHCP – Parte B.

- **Direccionamiento IP DHCP:** En ésta sección no se realiza un diagrama de flujo, debido a que el usuario sólo podrá visualizar una tabla con información acerca de las IPs asignadas dinámicamente. Si se observa en la figura 3.9 correspondiente a la sección 3.1.3.2, la información que se proporciona es la requerida en la historia de usuario en la tabla 2.9 del capítulo 2.

3.1.4.2 Codificación del Módulo de Servicio Proxy –Caché

En éste módulo se encuentran las siguientes secciones:

- **Modificación de Opciones Generales:** Esta sección permite modificar ciertos parámetros del servicio Proxy-Caché, como por ejemplo: el puerto, el idioma, el tipo de proxy, el tamaño de caché y el nombre de identificación para el proxy. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



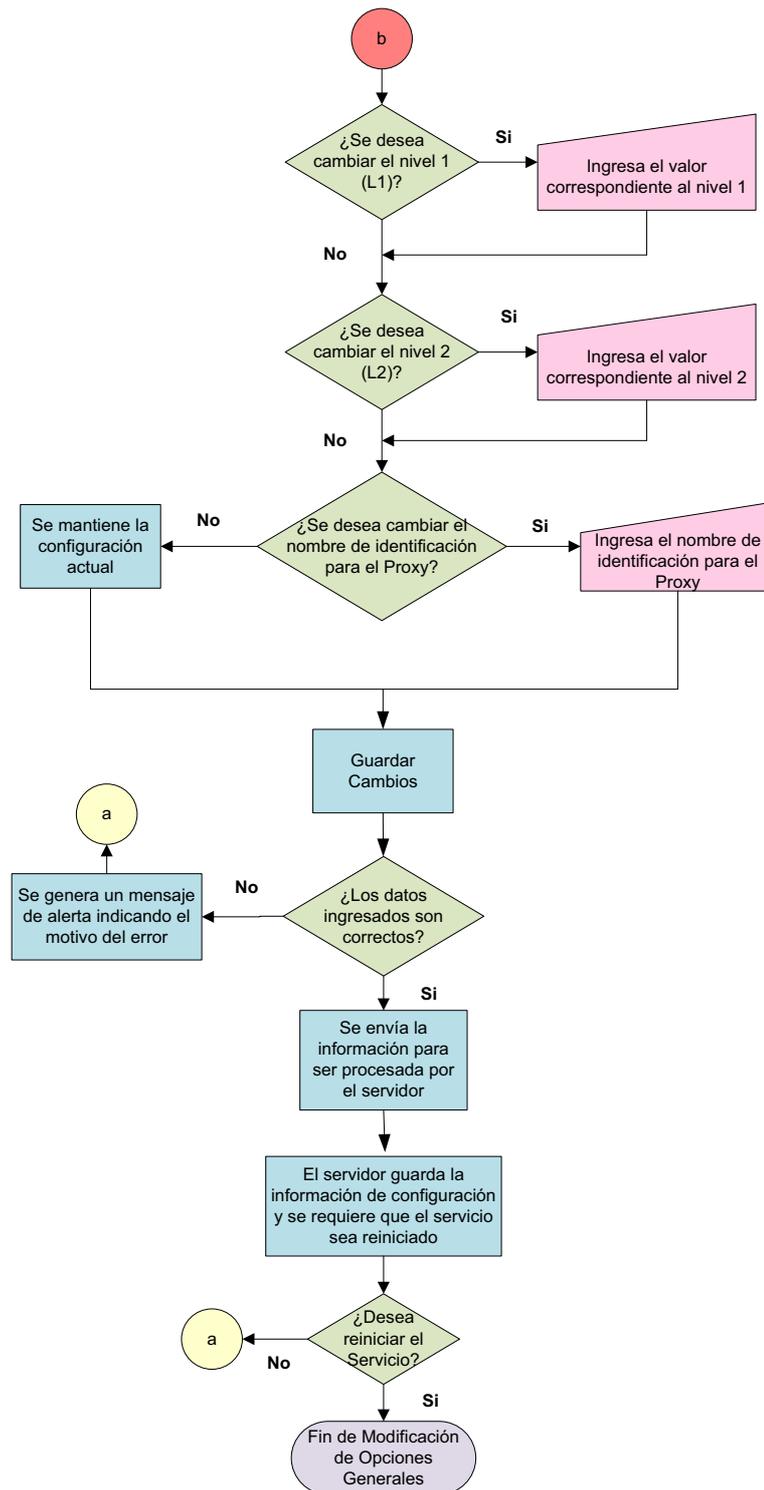
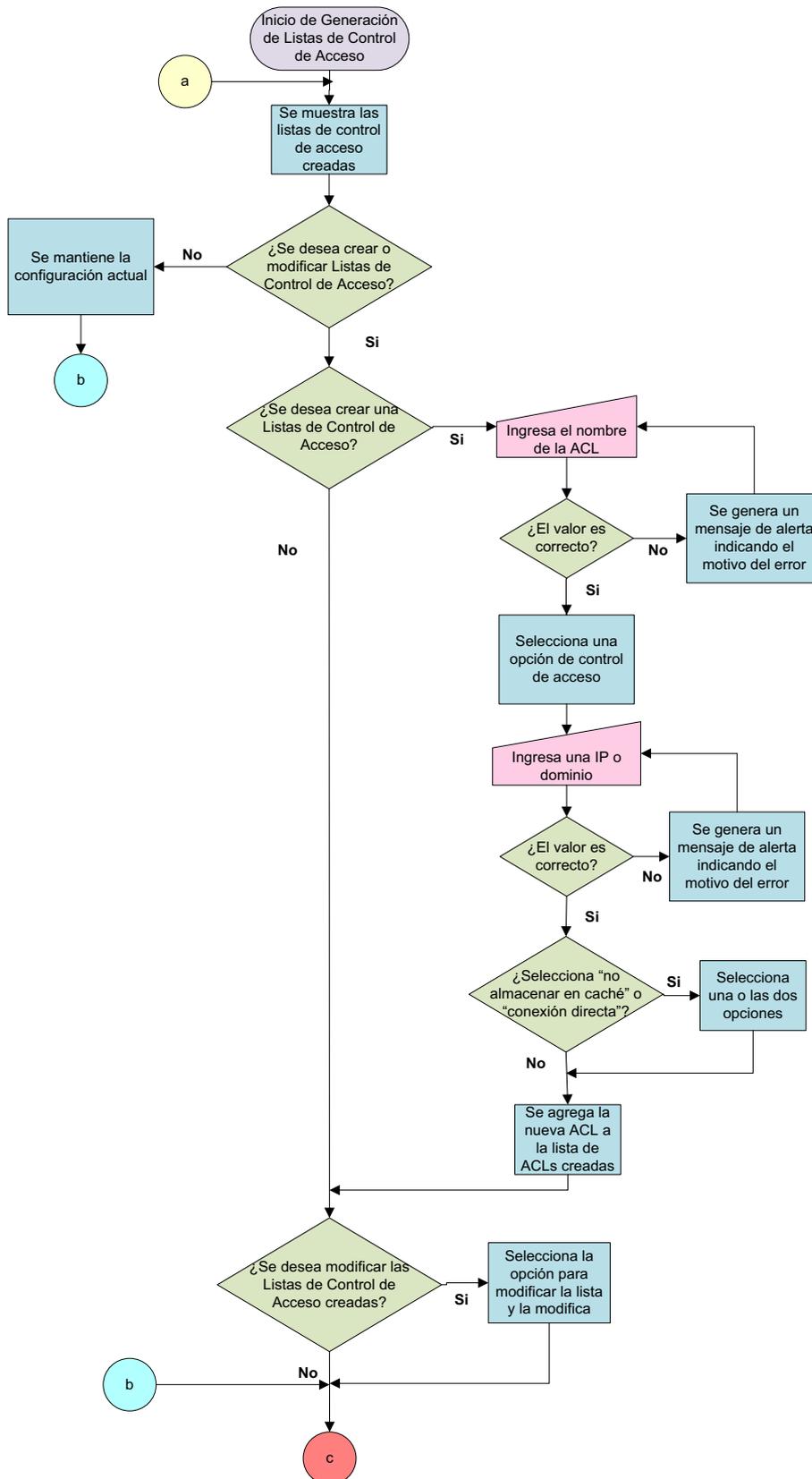


Figura 3.40 Diagrama de Flujo – Proxy - Modificación de Opciones Generales.

- **Generación de Listas de Control de Acceso:** Permite crear y modificar listas de control de acceso, para esto la interfaz solicita que el usuario ingrese cierta información: el nombre de la ACL, seleccione una opción para la ACL, una

dirección IP o un dominio. A continuación se muestra el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



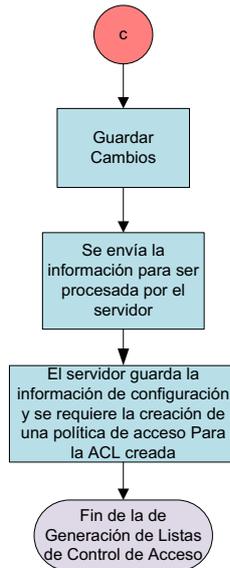
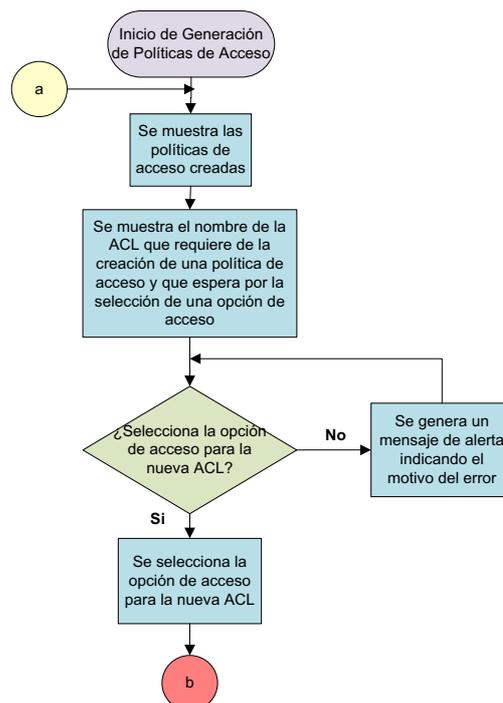


Figura 3.41 Diagrama de Flujo – Proxy – Generación de Listas de Control de Acceso.

- Generación de Políticas de Acceso:** Esta sección permite crear una política de acceso para la ACL creada en la sección anterior, es decir que por cada ACL se requiere de la creación de una política de acceso, lo único que se requiere es seleccionar una opción para permitir o denegar el acceso al servicio Proxy. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



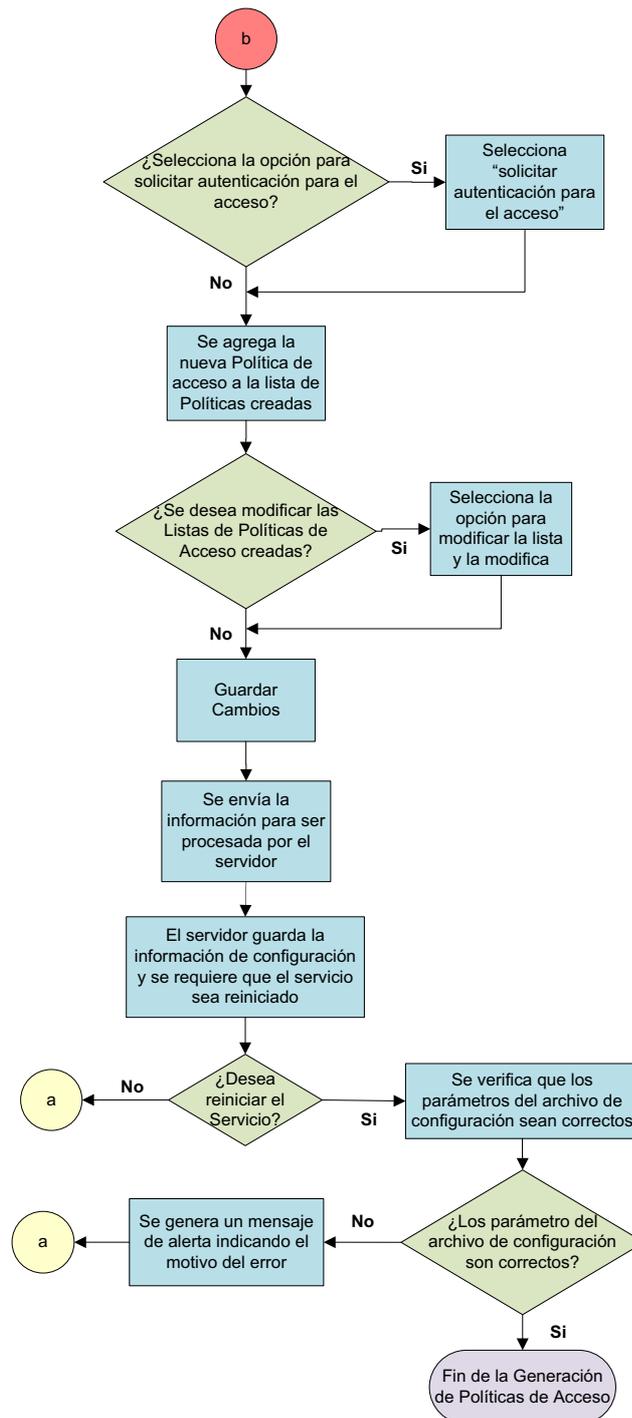
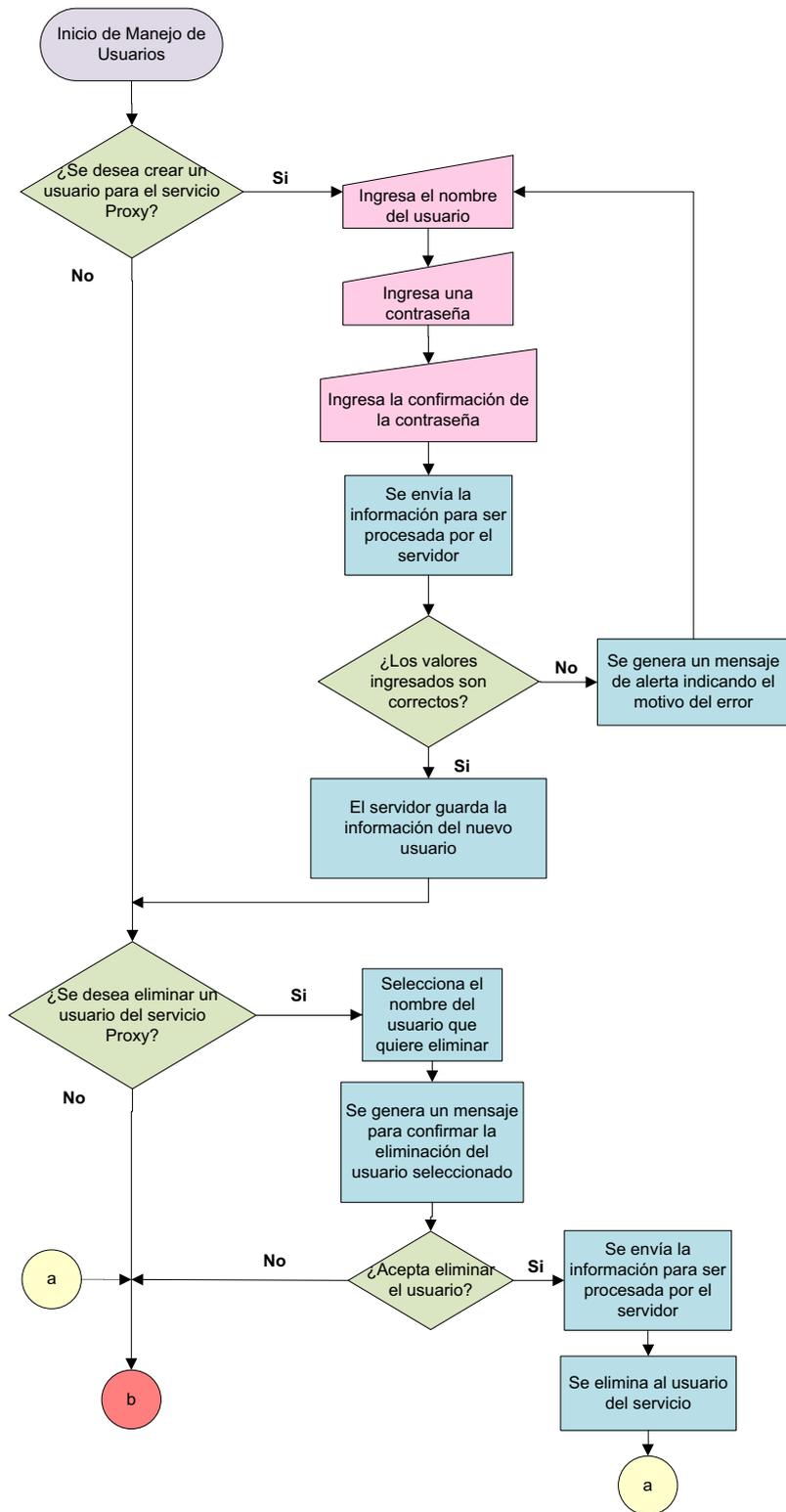


Figura 3.42 Diagrama de Flujo – Proxy – Generación de Políticas de Acceso.

- **Manejo de Usuarios para el Servicio:** Permite crear, eliminar y modificar usuarios para el servicio Proxy – Caché, los parámetros que se requieren son los siguientes: el nombre de usuario y una contraseña. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



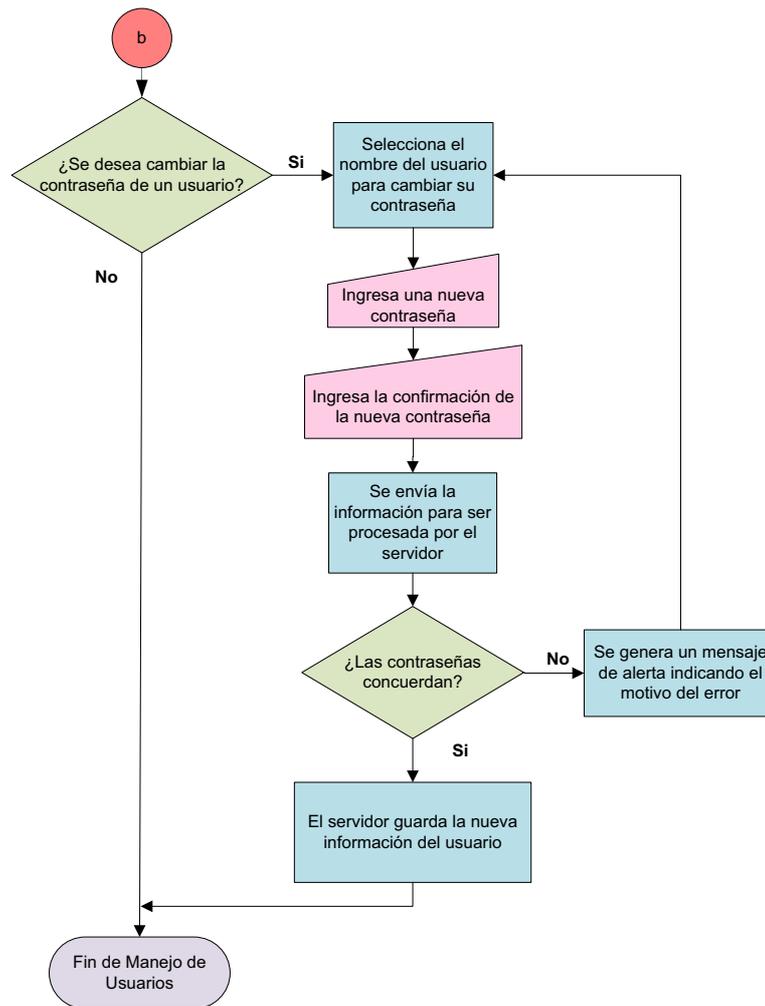


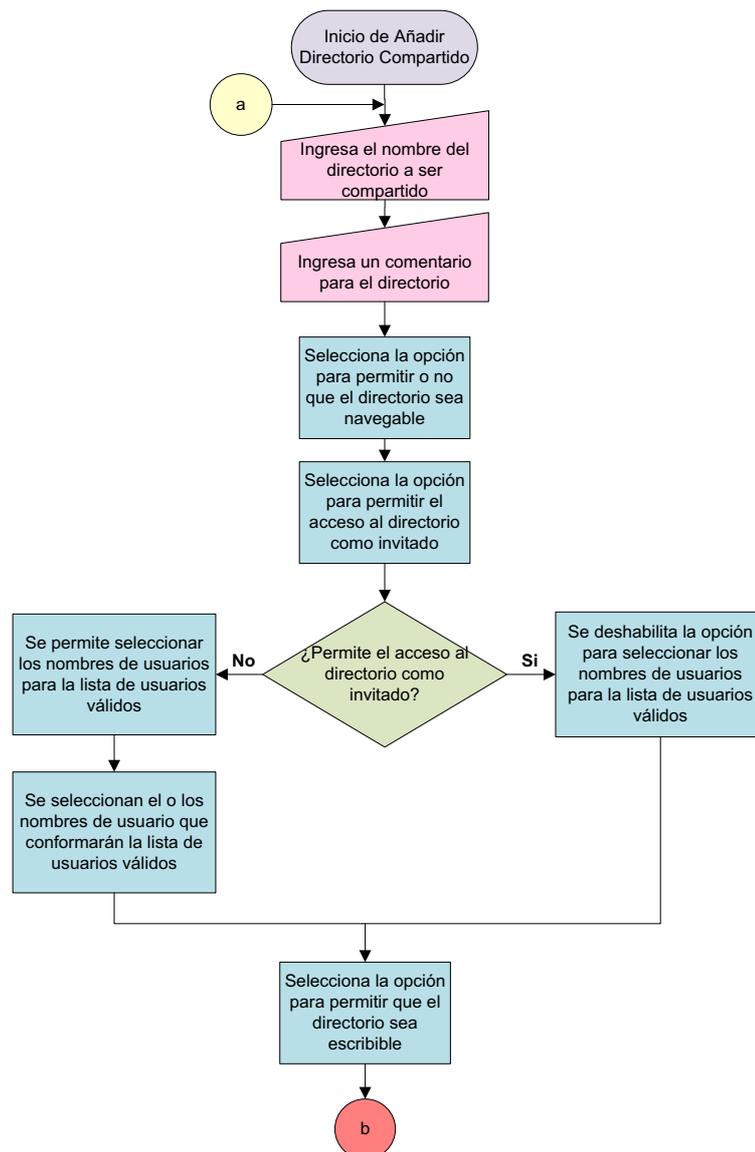
Figura 3.43 Diagrama de Flujo – Proxy – Manejo de Usuarios.

3.1.4.3 Codificación del Módulo de Servicio de Directorios Compartidos

En éste módulo se encuentran las siguientes secciones:

- **Visualización de Información General:** En ésta sección únicamente se obtendrá información del servicio y se la mostrará en la interfaz web, es decir que en ésta sección el usuario no tendrá que realizar ninguna acción, por ésta razón no se requiere de un diagrama de flujo.
- **Manejo de Usuario para el Servicio:** Esta sección permite crear, modificar y eliminar usuarios del servicio de Directorios Compartidos. No se realizará un diagrama de flujo debido a que la lógica empleada en ésta sección es la misma que se utilizó para el manejo de usuarios del servicio Proxy – Caché.

- Adición de Directorios al Servicio:** Esta sección permite agregar un directorio al servicio para que pueda ser compartido, dentro de los parámetros que se requiere configurar están: nombre del directorio a ser compartido, un comentario que describa al directorio, especificar si es navegable, escribible, si permite el acceso como invitado, según sea el caso agregar los usuarios en la lista de escritura o usuarios válidos. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



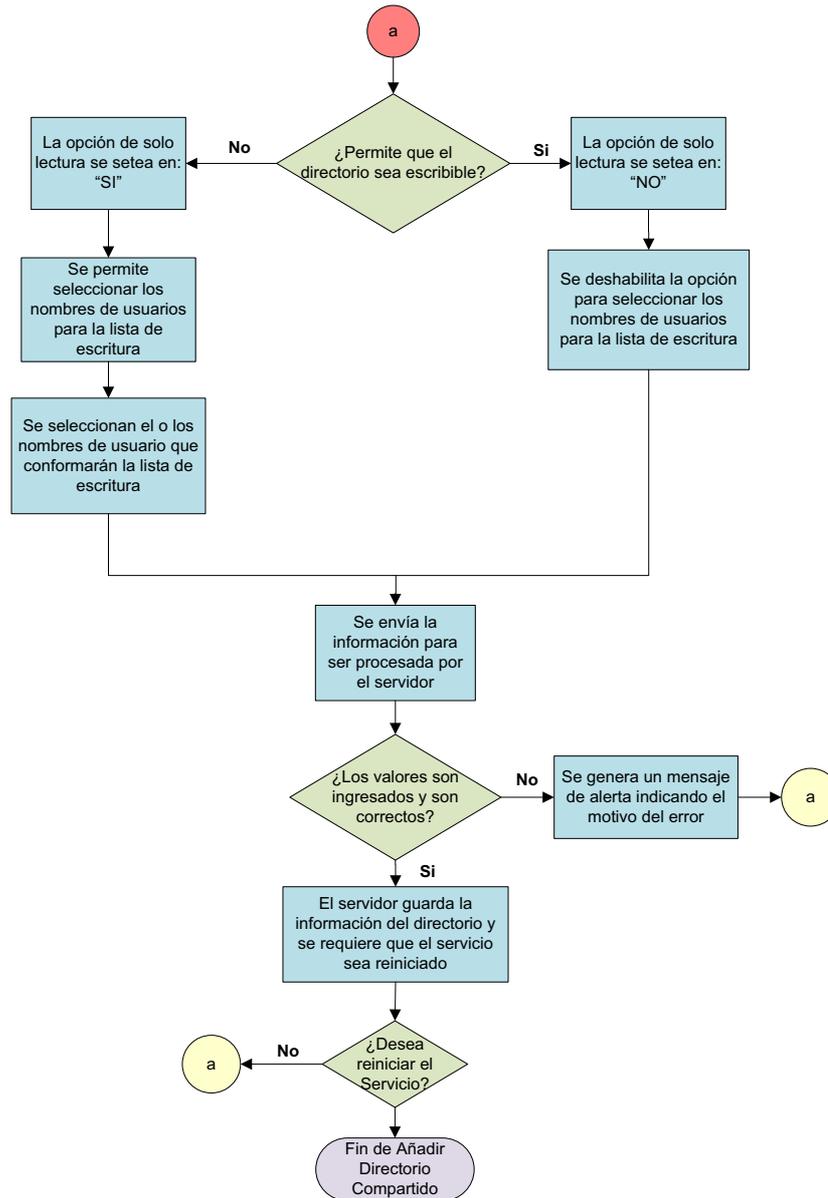
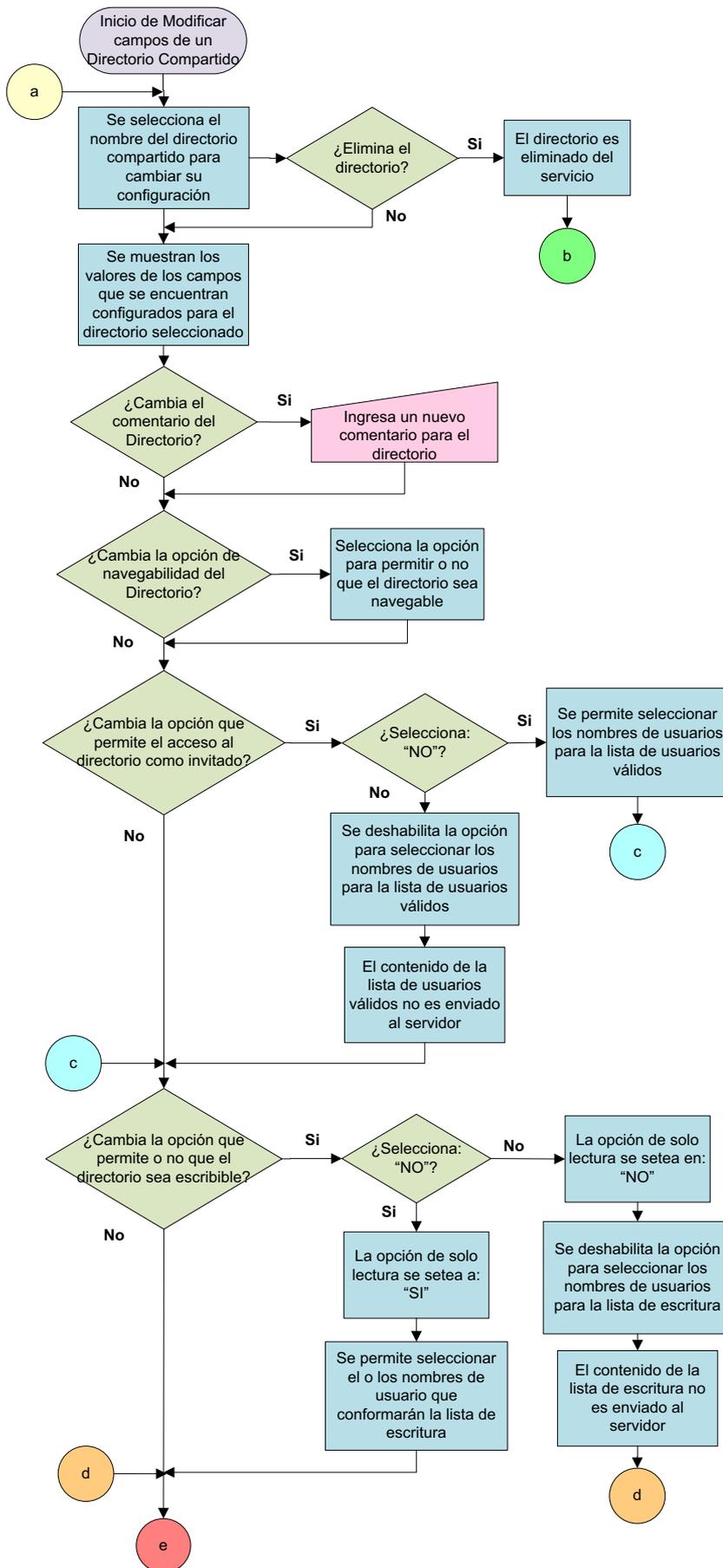


Figura 3.44 Diagrama de Flujo – Directorios Compartidos – Añadir un Directorio.

- Modificación de las Características de los Directorios Compartidos:**
 Permite modificar los parámetros configurados para un directorio en particular. Los parámetros que se permite modificar son todos los mencionados en la sección para agregar un directorio a excepción del nombre de directorio compartido. Además permite borrar el directorio del servicio. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



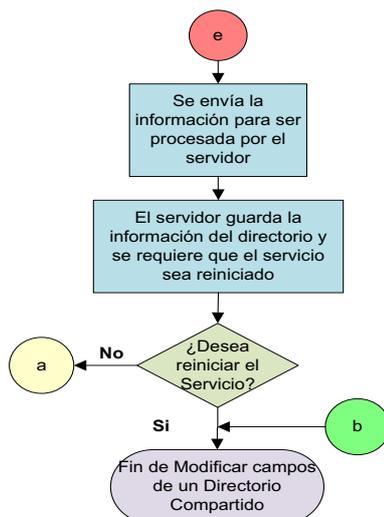
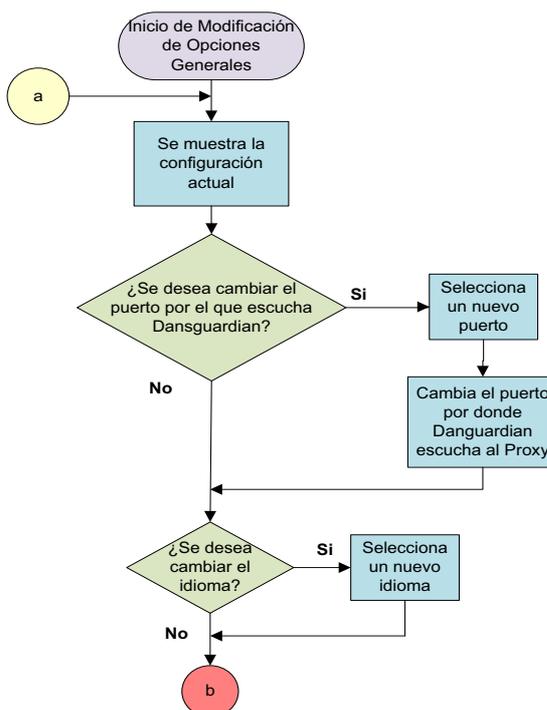


Figura 3.45 Diagrama de Flujo – Directorios Compartidos – Modificar los campos de un Directorio Compartido.

3.1.4.4 Codificación del Módulo de Servicio de Filtrado Web

Este módulo contiene las siguientes secciones:

- **Modificación de Opciones Generales:** Esta sección permite configurar ciertos parámetros del servicio de Filtrado Web, como: el idioma y el puerto que utiliza el servicio. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



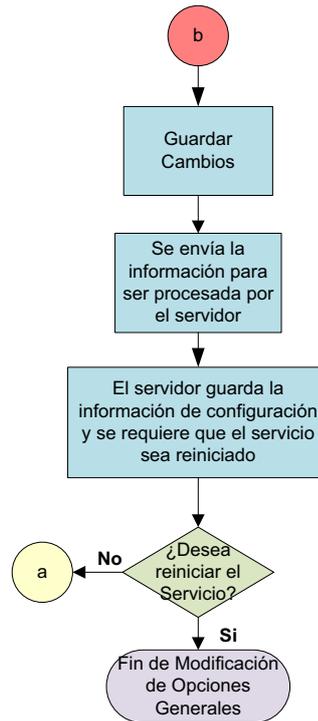
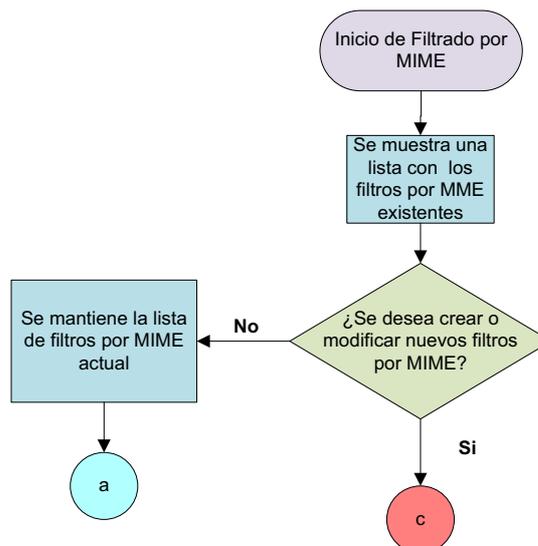


Figura 3.46 Diagrama de Flujo – Filtrado Web – Modificación de Opciones Generales.

- **Filtrado por MIME:** Permite filtrar el acceso Web por MIME, permite seleccionar o ingresar un tipo de MIME. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



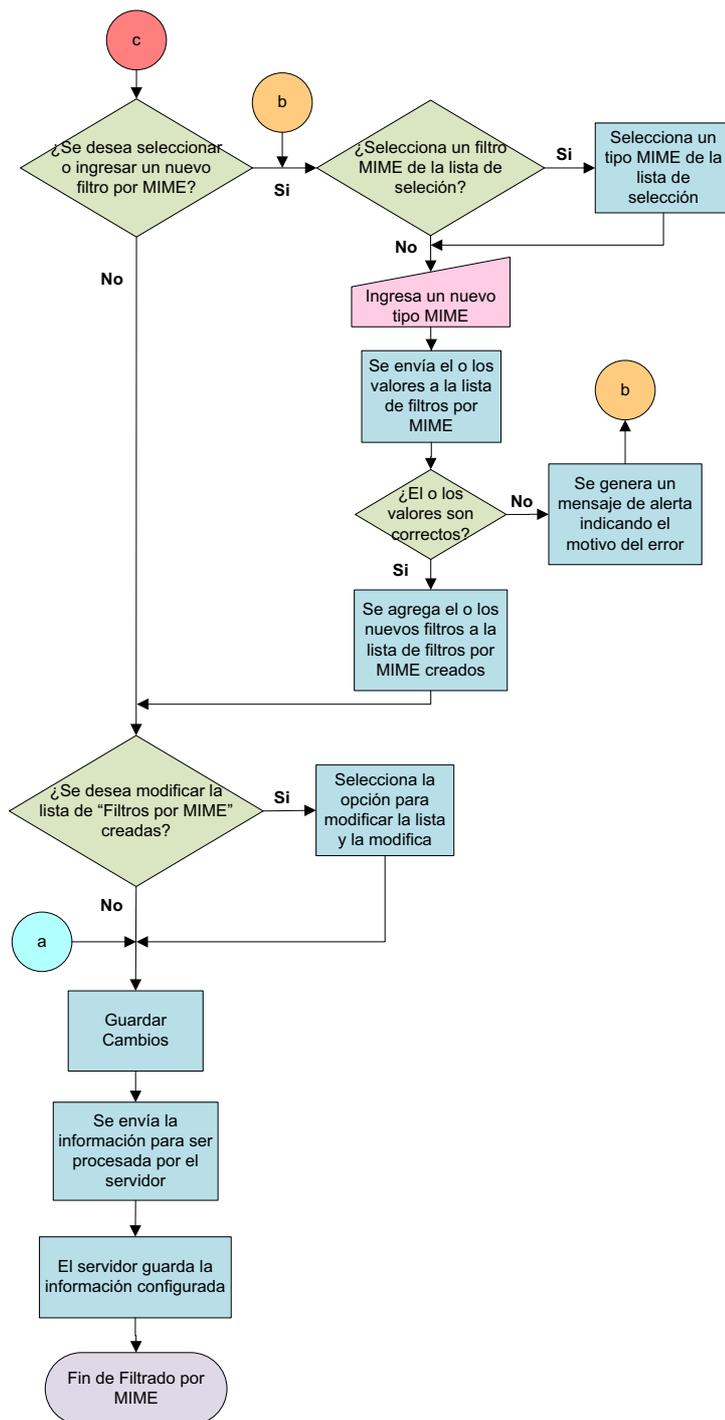


Figura 3.47 Diagrama de Flujo – Filtrado Web – Filtrado por MIME.

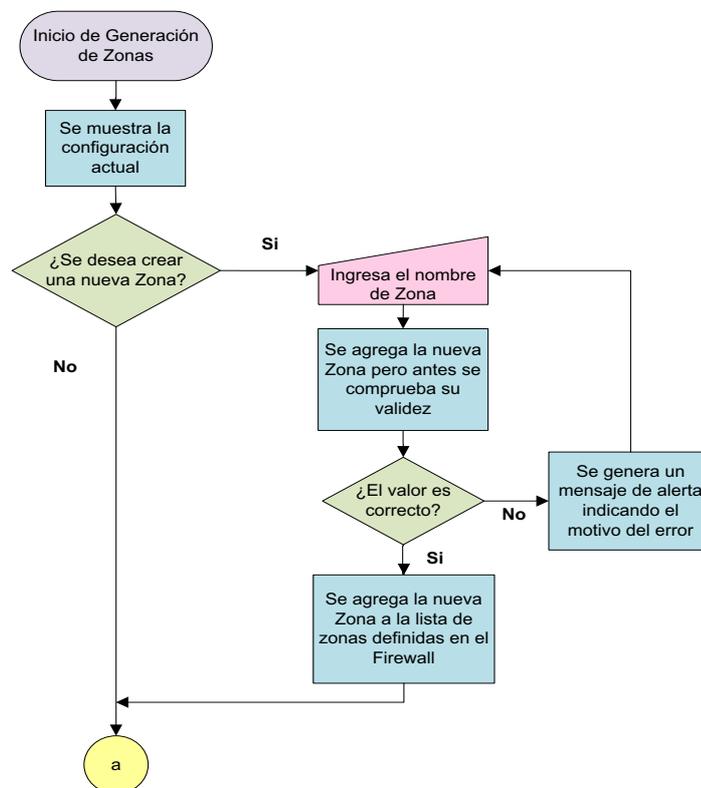
En las siguientes secciones no se presentarán diagramas de flujo, únicamente se indicará su funcionalidad y los parámetros que se pueden configurar, debido a que la lógica empleada para todas éstas secciones es la misma que la de la sección anterior.

- **Filtrado por Extensión:** Esta sección permite filtrar el acceso Web por extensión, mediante el ingreso de una extensión.
- **Filtrado por Site:** Permite filtrar el acceso Web por site, mediante la selección o ingreso de un site.
- **Filtrado por Frase:** Permite filtrar el acceso Web por frase, mediante el ingreso o selección de una frase.

3.1.4.5 Codificación del Módulo de Servicio de Firewall

Dentro de éste módulo se encuentran las siguientes secciones:

- **Generación de Zonas:** Esta sección permite crear y modificar las zonas a administrar por el firewall, el único parámetro que se debe ingresar es el nombre de la zona, si se requiere modificar las zonas se cuenta con una opción que permite hacerlo. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



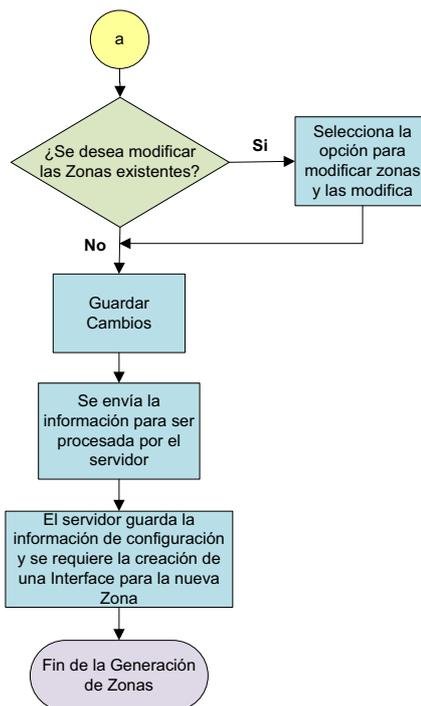
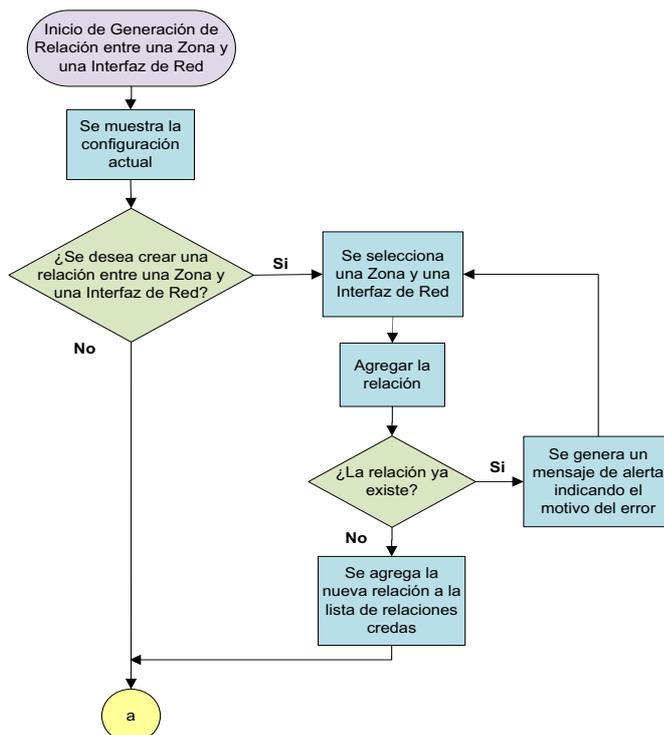


Figura 3.48 Diagrama de Flujo – Firewall – Generación de Zonas.

- **Generación de Relación entre una Zona y una Interfaz de Red:** Esta sección permite relacionar una interfaz con una zona, para esto se requiere seleccionar una zona y la interfaz con la que se desea generar la relación. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.



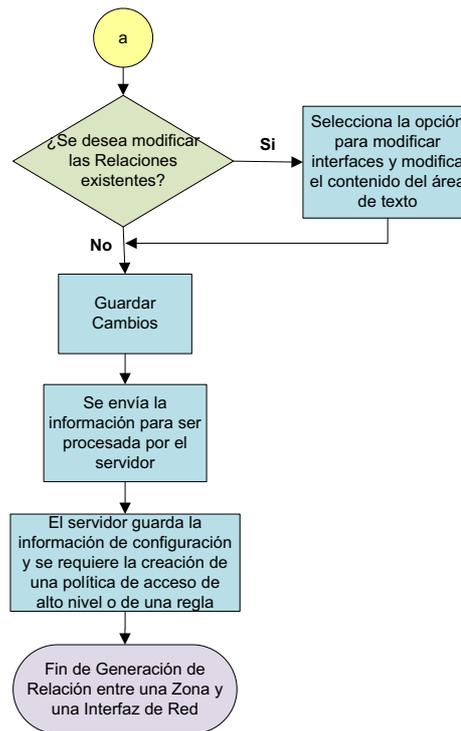


Figura 3.49 Diagrama de Flujo – Firewall – Generación de Relación entre una Zona y una Interfaz de Red.

- **Generación de Políticas de Acceso de Alto Nivel:** Permite crear políticas de acceso para admitir la comunicación entre las diferentes zonas, se debe seleccionar la zona de origen, destino y la política. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.

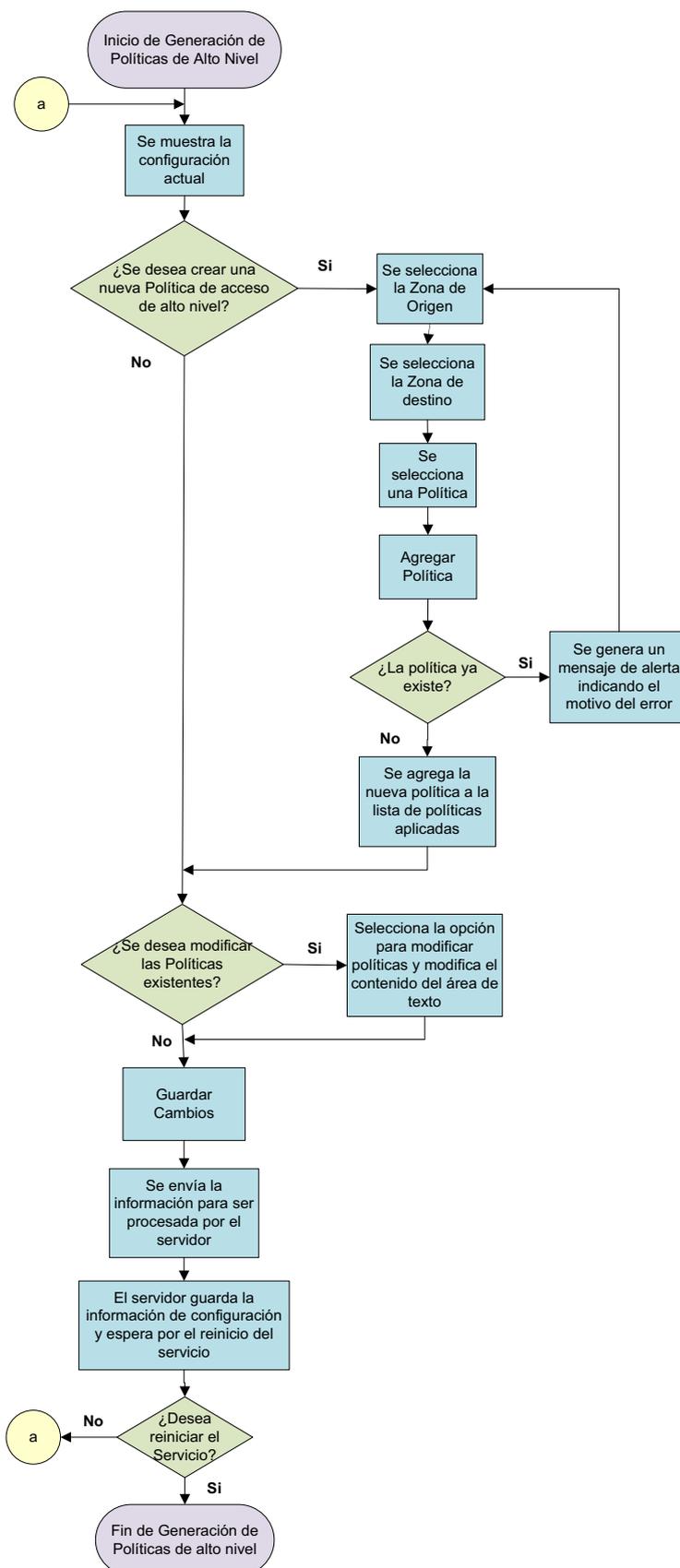


Figura 3.50 Diagrama de Flujo – Firewall – Generación de Políticas de Alto Nivel.

- Generación de Reglas de Acceso Específicas:** Esta sección permite crear reglas de acceso específicas para las solicitudes de conexión, para esto se debe seleccionar una acción que posteriormente deberá ser modificada por el usuario. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.

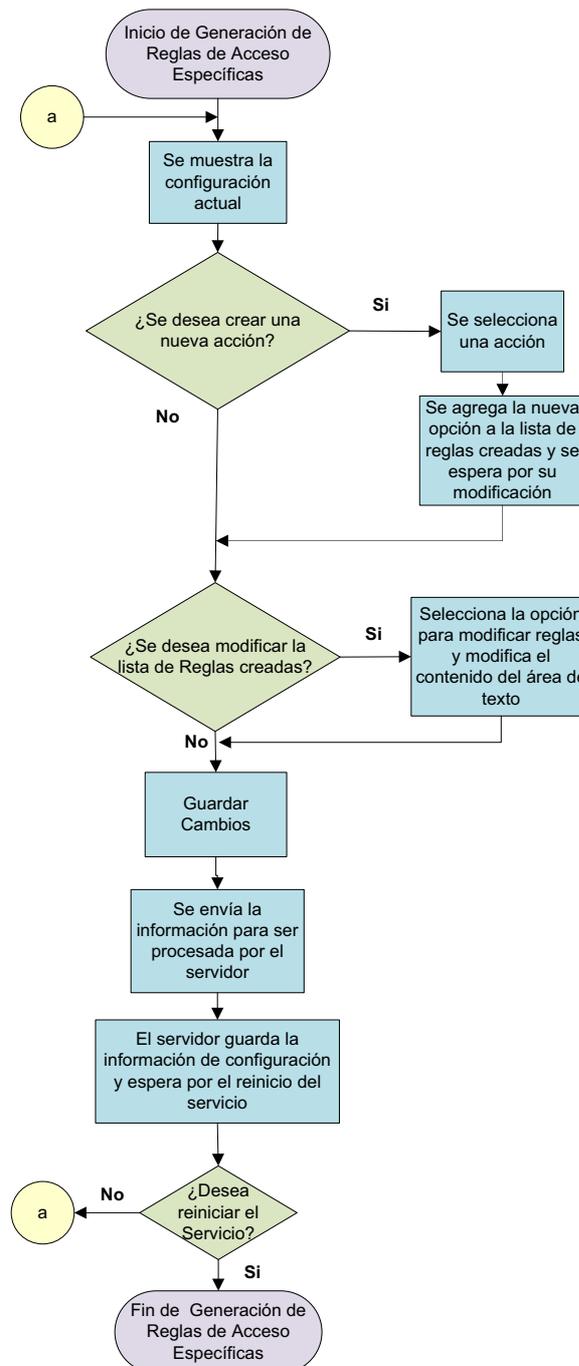


Figura 3.51 Diagrama de Flujo – Firewall – Generación de Reglas de Acceso Específicas.

- **Generación de Enmascaramiento:** Permite generar el proceso de enmascaramiento, para esto se debe seleccionar una interfaz a Internet que posteriormente deberá ser modificada por el usuario. A continuación se presenta el diagrama de flujo que describe la lógica de la funcionalidad de ésta sección.

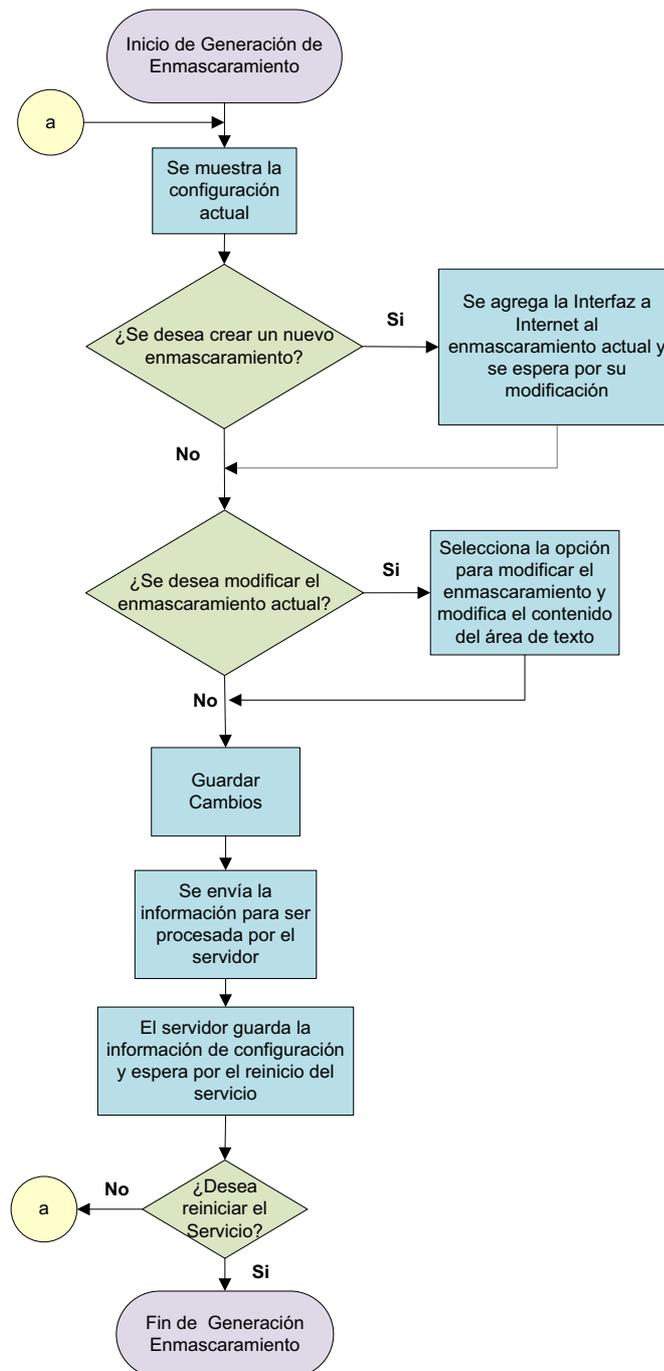


Figura 3.52 Diagrama de Flujo – Firewall – Generación de Enmascaramiento.

3.1.4.6 Codificación del Módulo de Servicio de Estado del Sistema Operativo

Para la codificación de cada una de las secciones presentes en éste módulo se han empleado los comandos propios de Linux, cuya descripción se encuentra en el anexo B. La figura 3.53 muestra el diagrama de flujo en donde se describe la lógica empleada en el desarrollo de la sección correspondiente al Estado del Disco Duro, ésta misma lógica se ha empleado en cada una de las secciones que conforman el presente módulo, por lo que los diagramas de flujo para las secciones correspondientes a: RAM y CPU, Información de Red e Información Adicional, no serán detallados.

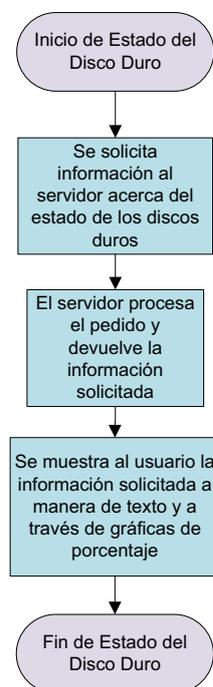


Figura 3.53 Diagrama de Flujo – Estado del Sistema Operativo – Estado del Disco Duro.

3.1.4.7 Codificación del Módulo de Servicio de Gráficas

Para el desarrollo de las gráficas presentes en éste módulo se ha empleado una herramienta denominada **RRDTOOL (Round Robin Database Tool o Herramienta para Bases de Datos Circulares)**, éste sistema permite almacenar y presentar datos en intervalos temporales, es decir que los archivos en donde se almacenarán éstos datos tendrán un tamaño fijo, esto se debe a que al ser una base de datos circular va a almacenar siempre la misma cantidad de datos, el procedimiento es como el de un círculo, se van colocando los datos hasta dar la

vuelta todo el círculo, al llegar al punto inicial los nuevos datos sobrescriben a los antiguos.

Generalmente **RRDTOOL** se utiliza para representar gráficas de temperatura, presión atmosférica, tráfico de red, carga de CPU, entre otras. Para el desarrollo de éste módulo se generarán gráficas para el uso de memoria, carga de CPU y rendimiento de red. Los gráficos se generan con extensiones .png, .gif, de forma que se los puede publicar en la interfaz Web del sistema de administración DAMA sin mayor complicación.

RRDTOOL contiene una serie de funciones que son compatibles con el lenguaje de programación Perl, que es el lenguaje empleado en el desarrollo del presente proyecto. Para la programación se ha empleado el módulo **RRDs** que permite el acceso a la funcionalidad de RRDTOOL desde Perl. De entre las funciones utilizadas para éste propósito están:

- **RRDs::create:** Permite crear una nueva base de datos.
- **RRDs::update:** Almacena los valores en la base de datos.
- **RRDs::graph:** Permite generar las gráficas en diferentes formatos.
- **RRDs::error:** Retorna el error cometido después de cada llamada a la función, si no devuelve ningún valor, quiere decir que la ejecución de la función anterior se ha realizado con éxito.

Las demás funciones que contiene **RRDTOOL** que no han sido utilizadas así como la estructura de las funciones descritas anteriormente, se las puede encontrar en la página oficial de éste sistema³⁶.

La lógica empleada en el desarrollo de éste módulo se detalla a continuación:

a) Primero obtiene la información que va a ser almacenada:

La forma en cómo se obtiene la información, depende del tipo de gráfico que se desee presentar, es decir:

³⁶ **RRDTOOL Documentation:** <http://oss.oetiker.ch/rrdtool/doc/index.en.html>

La información para la gráfica de Carga de CPU, como se puede observar en la figura 3.54, se obtiene del archivo `/proc/stat`, lo que se hace es obtener el porcentaje de utilización en un intervalo de tiempo, que para éste caso será un segundo, éste procedimiento se encuentra descrito en el anexo B.

```
# obtiene la información del cpu
@t1 = split(/\s+/, 'head -n 1 /proc/stat');
sleep(1);
@t2 = split(/\s+/, 'head -n 1 /proc/stat');

$cargal = $t1[1]+$t1[2]+$t1[3];
$carga2 = $t2[1]+$t2[2]+$t2[3];
$uso1 = $cargal + $t1[4];
$uso2 = $carga2 + $t2[4];
$t = 100 * (($carga2 - $cargal) / ($uso2 - $uso1));
$cpu = sprintf("%.1f", $t);
```

Figura 3.54 Obtención de información para la Gráfica de Carga de CPU.

La obtención de la información para la gráfica de Uso de Memoria se la hace mediante el uso del comando `free`, como se observa en la figura 3.55, la información acerca de éste comando se encuentra descrita en el anexo B.

```
# obtiene información de la memoria
open(IN, "free -b|");
while(<IN>){
  if($_ =~ /Mem:\s+(\d+)/){
    $mem_total = $1;
  }
  if($_ =~ /\s+ buffers\|cache:\s+(\d+)/){
    $mem_usada = $1;
  }
}
$mem = "$mem_total:$mem_usada";
$date = `date +%s`;
```

Figura 3.55 Obtención de información para la Gráfica de Uso de Memoria.

La información para la gráfica de Rendimiento de Red se la obtiene mediante el uso del comando `ifconfig`, como se muestra en la figura 3.56, éste comando se encuentra detallado en el anexo B.

```
my $in = `/sbin/ifconfig $_[0] |grep bytes|cut -d":" -f2|cut -d" " -f1`;
my $out = `/sbin/ifconfig $_[0] |grep bytes|cut -d":" -f3|cut -d" " -f1`;
```

Figura 3.56 Obtención de información para la Gráfica de Rendimiento de Red.

b) Se crea la base de datos:

Para crear la base de datos se hace uso de la función **RRDs::create**, la extensión de éste archivo es **.rrd**.

```
RRDs::create "$rrd/cpu.rrd",
  "-s 300",
  "DS:cpu:GAUGE:600:0,00:U",
  "RRA:AVERAGE:0.5:1:600",
  "RRA:AVERAGE:0.5:6:700",
  "RRA:MIN:0.5:1:600",
  "RRA:MIN:0.5:6:700",
  "RRA:MAX:0.5:1:600",
  "RRA:MAX:0.5:6:700",
```

Figura 3.57 Creación de la Base de Datos RRD.

c) La información es almacenada en la base de datos y cada cierto periodo de tiempo actualiza los datos. Para almacenar los datos se hace uso de la función **RRDs::update**.

```
RRDs::update( $rrd1, "-t", "cpu", "N:$cpu");
```

Figura 3.58 Almacenamiento de la información en la Base de Datos RRD.

d) Se toman los datos para crear la gráfica que posteriormente visualizará el usuario. Para crear las gráficas se hace uso de la función **RRDs::graph**.

```

RRDs::graph "$img/dcpu-$_[0].png",
>   "-s -1$_[0]",
>   "-e $last",
>   "-h", "80", "-w", "600",
>   "-l 0",
>   "-u 100",
>   "-a", "PNG",
>   "-v", "Carga en (%)",
>   "DEF:cpu=$rrd1:cpu:AVERAGE",
>   "AREA:cpu#32CD32:Carga CPU",
>   "LINE2:cpu#32CD32",
>   "GPRINT:cpu:MAX:Máximo\\:\\: %3.1lf %%",
>   "GPRINT:cpu:MIN:Minimo\\:\\: %3.1lf %%",
>   "GPRINT:cpu:AVERAGE:Promedio\\:\\: %3.1lf %%",
>   "GPRINT:cpu:LAST:Último\\:\\: %3.1lf %%";

```

Figura 3.59 Creación de gráficas mediante el uso de la función RRDs :: graph.

3.1.4.8 Codificación del Módulo de Página Inicial

Para la codificación del presente módulo, se ha empleado un resumen acerca del estado del servidor, es decir se implementó una copia de la información relevante del módulo del Estado del Sistema Operativo, de manera que se utilizó la misma lógica, ésta se encuentra descrita en la sección 3.1.4.6 del presente capítulo.

- **Manejo de Archivos de Configuración:** Una tarea que es común en la codificación de los módulos que conforman el sistema DAMA, es la forma en cómo se lee y escribe en los archivos de configuración de cada uno de los servicios instalados en el servidor. Para ello se utiliza Perl, que es un lenguaje que permite la fácil manipulación de archivos de texto a través de la generación de un script, Perl cuenta con un conjunto de funciones que facilitan el manejo de texto dentro de un archivo.

A continuación se muestra un ejemplo en el que se demuestra la funcionalidad de Perl para el manejo de archivos de texto, en el cual se ha utilizado algunas de las funciones que permiten manipular el contenido dentro de un archivo.

```
#!/usr/bin/perl

open (ON,"/home/Dama/squid/squid.conf");
my $ip:
my $puerto:
while(<ON>){

    if($_ =~ /^http_port/){
        @smb = split(/\s/, $_);
        ($ip,$puerto) = split(/:./,$smb[1]);
    }
}
close(ON);

open (ON,"/home/Dama/dansguardian/dansguardian.conf");
my $cabi = 0;
my $cwb = 0;
my $caba = 0;
while(<ON>){
    if($caba == 1){
        push(@squida,$_);
    }
    if($_ =~ /^filterport/){
        @dans = split(/\s/, $_);
        if($dans[2] == $puerto){
            if($dans[2] == 3128){
                $dans[2] = 8080;
            }else{
                $dans[2] = 3128;
            }
        }
        my $s = join(" ",@dans);
        push(@squida,"$s\n");
        $cwb = 1;
    }
    if($_ =~ /^proxyport/){
        @squid = split(/\s/, $_);
        if($dans[2] == 3128){
            $squid[2] = 8080;
        }else{
            $squid[2] = 3128;
        }
        my $s = join(" ",@squid);
        push(@squida,"$s\n");
        $caba = 1;
    }
    if($cwb == 0){
        push(@squida,$_);
    }
}
close(ON);

open (ON,">/home/Dama/dansguardian/dansguardian.conf");
foreach $d (@squida){
    print ON "$d";
}

@log = `sudo /usr/local/sbin/reiniciosquid`;!
```

Figura 3.60 Ejemplo de Manipulación de archivos en Perl.

Al ejemplo se lo ha dividido en 4 secciones, cada una con su color respectivo para poder analizarlo de mejor manera. La tarea que realiza el ejemplo es cambiar los puertos del servicio de filtrado web, tanto el puerto por el cual

trabaja el filtrado web como el puerto por el cual trabaja el proxy. De esta manera:

La primera sección corresponde a la línea destinada a proporcionar la ubicación del intérprete, ésta comienza con un signo **#** seguido del signo **!** y a continuación se ingresa la ruta del intérprete dentro del sistema operativo, ésta línea no finaliza con un **;** como las líneas a continuación referidas al script. Este aspecto es muy importante ya que a pesar de que todo el script se encuentre desarrollado correctamente si ésta línea falta, el resto no funcionará.

La segunda sección emplea la instrucción *Open* para indicar que se va a abrir un archivo en modo lectura para obtener la información requerida. Como se observa en la figura 3.61, se obtendrá información de dos archivos. La información en los archivos se halla almacenada de la siguiente manera:



Figura 3.61 Información almacenada en los archivos de texto.

Dentro de esta sección se han empleado distintas funciones propias de perl, por ejemplo, ***split*** divide a una cadena en base a un patrón, ***push*** permite insertar un elemento al final de un array, ***join*** une una cadena de caracteres en base a un patrón, ***if*** con su respectiva sentencia condicional en este caso se hace uso de expresiones regulares y operadores lógicos. El signo **^** dentro del patrón de la expresión regular indica que se verifiquen que las líneas comiencen con la cadena de caracteres que le sigue.

La tercera sección realiza la tarea de escribir, en éste caso en el archivo de configuración `dansguardian.conf`, para esto se utiliza la instrucción ***open***

seguido de la opción **>** para indicar que se va a abrir un archivo en modo de escritura borrando la información existente, mediante la instrucción **print** seguido de **ON** que corresponde al nombre de la variable que almacenará la información del archivo abierto, se procede a almacenar la información obtenida, para el ejemplo la información se encuentra almacenada en la variable **\$d**, luego de ello se procede a cerrar el archivo y se finaliza el script. La salida proporcionada es la siguiente:



Figura 3.62 Nueva información almacenada en los archivos de texto.

La última sección  se encarga de ejecutar el script que permite copiar la información de los archivos temporales a los archivos originales y además permite reiniciar los servicios de proxy y dansguardian.

Como se puede observar, el desarrollo de este script de ejemplo demuestra la facilidad de manejo de archivos de texto y la manipulación de su información, Perl tiene un gran potencial para tratar tareas mucho más complejas que este ejemplo es por esta razón que es utilizado dentro de sistemas donde la manipulación de archivos de texto es primordial.

CAPÍTULO 4

4 PRUEBAS Y EVALUACIÓN DE LA INTERFAZ WEB

4.1 IMPLEMETACIÓN DE UNA RED LOCAL TIPO SOHO

Para la realización de las pruebas y evaluación del sistema de administración Web DAMA, se implementará una red local tipo SOHO, la cual constará de entre 2 a 20 hosts.

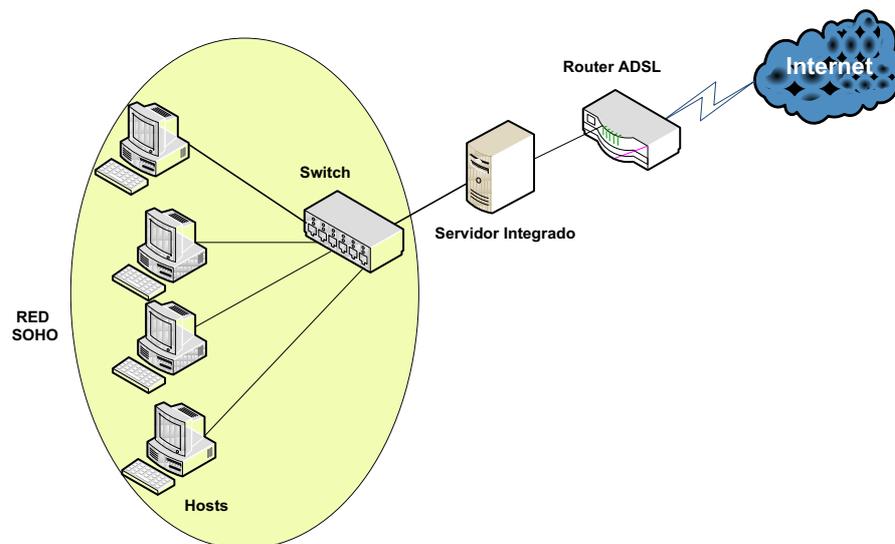


Figura 4.1 Esquema de la red local a implementar en las pruebas del Sistema de Administración Web DAMA.

La figura 4.1 muestra el esquema típico de una red tipo SOHO, como se puede observar tanto para la implementación de la red SOHO como para el perfecto funcionamiento del sistema DAMA, se requerirá de varios dispositivos hardware, dentro de los cuales se encuentran:

- *Hosts*: Corresponden a las máquinas que conformarán la red local SOHO, el sistema operativo que se encuentre instalado en cada uno de éstos es independiente.
- *Switch*: También conocido como conmutador, éste dispositivo también forma parte de la red local SOHO, permitirá conmutar la información del servidor hacia cualquier host que pertenezca la red local y viceversa.

- *Servidor Integrado:* Corresponde al servidor que contiene todos los servicios a los que se podrá acceder desde la interfaz Web, además aquí se alojará el sistema de administración DAMA. Cabe mencionar que éste servidor contará con dos tarjetas de red, una para la red local y la otra para la salida a Internet.
- *Router ADSL:* Éste dispositivo de red permitirá enrutar los paquetes de información desde una red a otra. Éste router corresponde al proporcionado por la empresa proveedora del servicio de Internet.

La red antes descrita corresponde a la red tipo SOHO implementada por los desarrolladores para realizar las pruebas en el transcurso del desarrollo, se empleará el mismo esquema para las pruebas a realizarse en un laboratorio disponible en la Facultad de Ingeniería Eléctrica y Electrónica para evaluar su comportamiento.

4.2 PRUEBAS Y EVALUACIÓN

4.2.1 PRUEBAS DE ACEPTACIÓN

Esta sección corresponde a la última fase de la metodología XP (programación extrema) empleada en el desarrollo del presente proyecto. En base a las historias de usuario así como del análisis de las iteraciones, se realizarán pruebas de aceptación, que como se explicó en la sección 1.8.5.3.2 del capítulo 1, servirán para verificar el correcto funcionamiento del sistema y también para determinar el cumplimiento de lo descrito en cada una de las historias de usuario definidas en el capítulo 2.

Para el desarrollo de las pruebas de aceptación, se emplearán tablas, al igual que en las historias de usuario, para de ésta manera poder codificar la información que permitirá confirmar que las historias de usuario han sido implementadas correctamente.

Caso de Prueba de Aceptación	
Número Caso de Prueba:	Número Historia de Usuario:
Nombre Caso de Prueba:	
Descripción:	
Condiciones de Ejecución:	
Entradas / Pasos de Ejecución:	
Resultado Esperado:	
Evaluación de la Prueba:	

Tabla 4.1 Formato de Caso de Prueba de Aceptación.

En donde:

- *Número Caso de Prueba:* corresponde al número del caso de prueba de aceptación.
- *Número de Historia de Usuario:* corresponde al número o nombre de historia de usuario, al cual se aplicará la prueba de aceptación. Este código corresponderá a los definidos en cada una de las historias de usuario descritas en la sección 2.1.3.2 del capítulo 2.
- *Nombre Caso de Prueba:* es el nombre de la prueba de aceptación.
- *Descripción:* es la descripción de las actividades a realizarse en cada una de las secciones de los módulos del sistema, en base a las historias de usuario descritas para cada una de ellas.
- *Condiciones de Ejecución:* como su nombre lo indica, ésta sección corresponde a la descripción de las condiciones necesarias para la ejecución de cada una de las secciones que conforman el sistema de administración DAMA.
- *Entradas / Pasos de Ejecución:* corresponde a las acciones que realiza el usuario en la sección que se está analizando en ése momento. Las entradas deben ser descritas una a una y en forma ordenada.

- *Resultado Esperado:* es el resultado que se espera obtener, de acuerdo a las acciones o pasos seguidos por el usuario en la sección del sistema que se esté analizando.
- *Evaluación de la Prueba:* es la valoración de los resultados obtenidos en base a los resultados que se esperaba obtener.

En base al formato de la tabla 4.1, se procede al desarrollo de las pruebas de aceptación de cada una de las secciones que forman parte de los módulos del sistema de administración DAMA.

Para el número de caso de prueba, se empleará una abreviatura, la cual será definida en cada módulo, por ejemplo para el módulo del servicio de DHCP, se empleará la abreviatura **CPAD (Nro.)**, en donde *CPA* hace referencia a Caso de Prueba de Aceptación, *D* corresponde al servicio DHCP y *(Nro.)* corresponderá al número de prueba de aceptación dentro de cada módulo.

MÓDULO DEL SERVICIO DHCP

La abreviatura empleada para el número de caso de prueba del presente módulo será: *CPAD (Nro.)*. A continuación se presentan las tablas que corresponden a los casos de prueba de aceptación de cada una de las secciones que conforman el módulo del servicio DHCP:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAD01	Número Historia de Usuario: HUD01- HUD02- HUDM01- HUDM02
Nombre Caso de Prueba: Ingreso configuración básica DHCP	
Descripción: Se configura los parámetros básicos del servicio DHCP.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Una vez realizada la configuración básica, el usuario deberá realizar la configuración dinámica del sistema. Al reiniciar el servicio DHCP, las personas que emplean los servicios del sistema, deberán negociar los parámetros de red para acceder nuevamente a los servicios.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio DHCP). 2. El usuario ingresa en los sitios correspondientes: la dirección IP para la interfaz que proveerá el servicio, dirección de red a la cual pertenece la interfaz, máscara de red y dirección broadcast de red. 3. El usuario da clic en el botón Guardar Cambios. 4. El usuario realiza la configuración dinámica del servicio, ingresa: el rango de direcciones IP que se asignará (dirección IP máxima y dirección IP mínima), tiempo mínimo de alquiler de la dirección IP en segundos y tiempo máximo de alquiler de la dirección IP en segundos. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio DHCP. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Al guardar los cambios el sistema DAMA muestra un mensaje de alerta indicándole al usuario que los datos ingresados se han agregado al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que emplean los servicios del sistema quedarán temporalmente sin conexión. • Se actualiza la información correspondiente a la nueva configuración y puede ser visualizada en la interfaz web. • Las personas que emplean los servicios del sistema accederán a ellos luego de obtener la información de red actual. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los datos se han agregado correctamente al servicio DHCP. • La red local funciona correctamente con los cambios realizados. • El sistema cumple con todos los resultados esperados. 	

Tabla 4.2 Prueba de Aceptación – Ingreso configuración básica DHCP.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAD02	Número Historia de Usuario: HUD02-HUDM02
Nombre Caso de Prueba: Ingreso configuración dinámica DHCP	
Descripción: Se configura la asignación dinámica de direcciones IP.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Al reiniciar el servicio DHCP, las personas que emplean los servicios del sistema, deberán negociar los parámetros de red para acceder nuevamente a los servicios.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio DHCP). 2. El usuario realiza la configuración dinámica del servicio, ingresa: el rango de direcciones IP que se asignará (dirección IP máxima y dirección IP mínima), tiempo mínimo de alquiler de la dirección IP en segundos y tiempo máximo de alquiler de la dirección IP en segundos. 3. El usuario da clic en el botón Guardar Cambios. 4. El usuario reinicia el servicio DHCP. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicándole al usuario que los datos ingresados se han agregado al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que emplean los servicios del sistema quedarán temporalmente sin conexión. • Se actualiza la información correspondiente a la nueva configuración y puede ser visualizada en la interfaz web. • Las personas que emplean los servicios del sistema accederán a ellos luego de obtener la información de red actual. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los datos se han agregado correctamente al servicio DHCP. • La red local funciona correctamente con los cambios realizados. • El sistema cumple satisfactoriamente con los resultados esperados. 	

Tabla 4.3 Prueba de Aceptación – Ingreso configuración dinámica DHCP.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAD03	Número Historia de Usuario: HUD03
Nombre Caso de Prueba: Ingreso configuración estática DHCP (A)	
Descripción: Se configura los parámetros para asignar una dirección IP fija a un host conocido.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Al reiniciar el servicio DHCP, las personas que emplean los servicios del sistema, deberán negociar los parámetros de red para acceder nuevamente a los servicios.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio DHCP). 2. El usuario realiza la configuración estática del servicio, ingresa: la dirección IP, dirección MAC y nombre del host. 3. El usuario da clic en el botón Guardar Cambios. 4. El usuario reinicia el servicio DHCP. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicándole al usuario que los datos ingresados han sido procesados por el servidor. • La nueva configuración se muestra en la lista de asignación de IP's estáticas. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que emplean los servicios del sistema quedarán temporalmente sin conexión. • Se actualiza la información correspondiente a la nueva configuración. • Las personas que emplean los servicios del sistema accederán a ellos luego de obtener la información de red actual. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los datos se han agregado correctamente al servicio DHCP y a la lista de direcciones IP's con asignación estática que se muestra en la interfaz web. • Los hosts seleccionados obtienen la dirección IP asignada. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.4 Prueba de Aceptación – Ingreso configuración estática DHCP (A).

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAD03	Número Historia de Usuario: HUD03
Nombre Caso de Prueba: Ingreso configuración estática DHCP (B)	
Descripción: Se selecciona un host que ya ha utilizado el servicio DHCP para asignarle esa dirección IP como estática.	
Condiciones de Ejecución: El usuario debe añadir un host mediante la selección de una de las tres opciones presentadas en la interfaz web. Al reiniciar el servicio DHCP, las personas que emplean los servicios del sistema, deberán negociar los parámetros de red para acceder nuevamente a los servicios.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio DHCP). 2. El usuario realiza la configuración estática del servicio para aquellos host que ya han utilizado el servicio DHCP, seleccionando: la dirección IP, dirección MAC o nombre del host. 3. El usuario da clic en aceptar en el mensaje de confirmación, para agregar al host a la lista de direcciones IP's estáticas. 4. Para eliminar algún host de la configuración estática, el usuario da clic en el botón eliminar. 5. El usuario reinicia el servicio DHCP. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de confirmación para agregar o no el host dentro de la configuración estática del servicio DHCP. • El sistema DAMA muestra un mensaje de alerta indicándole al usuario que la información ha sido procesada por el servidor. • El host seleccionado se muestra en la lista de asignación de IP's estáticas. • Al eliminar un host de la lista, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que la información ha sido procesada por el servidor. • Al reiniciar el servicio, el sistema DAMA le indica al usuario que las personas que emplean los servicios del sistema quedarán temporalmente sin conexión. • Se actualiza la información correspondiente a la nueva configuración. • Las personas que emplean los servicios del sistema accederán a ellos luego de obtener la información de red actual. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • La información del host seleccionado se ha agregado correctamente al servicio DHCP y a la lista de direcciones IP's con asignación estática que se muestra en la interfaz web. • Los hosts seleccionados obtienen la dirección IP asignada. • Al eliminar un host desde la interfaz web, éste se borra de la configuración estática del servicio DHCP. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.5 Prueba de Aceptación – Ingreso configuración estática DHCP (B).

- *Direccionamiento IP*

Ésta sección abarca la historia de usuario HUD04. Permite visualizar una lista que muestra los hosts con asignación dinámica, ésta lista contiene la siguiente información: la dirección IP, dirección MAC, el nombre del host y los detalles acerca de la concesión. En cuanto a la evaluación de la prueba, el sistema muestra la información correspondiente a cada uno de los aspectos anteriormente mencionados. El usuario deberá ingresar como administrador al módulo del servicio DHCP para poder visualizar ésta información.

Dirección IP	MAC	Hostname	Parámetros de Concesión
192.168.2.2	00:1b:24:9a:e7:70	Maya	Inicio de Concesión: 2 2010/12/14 23:50:19 Final de Concesión: 3 2010/12/15 05:53:39 Estado de Concesión: active
192.168.2.4	00:1e:33:a1:d8:97	No asignado29	Inicio de Concesión: 2 2010/12/14 23:27:19 Final de Concesión: 3 2010/12/15 05:30:39 Estado de Concesión: active
192.168.2.3	00:1e:33:a1:d8:97	No asignado6	Inicio de Concesión: 2 2010/12/14 20:40:13 Final de Concesión: 2 2010/12/14 20:42:25 Estado de Concesión: free

*Formato de los parámetros de concesión:
 Inicio de Concesión: día de la semana año/mes/día hora:minutos/segundos GMT (Standard Time)
 Estado de Concesión: estado (active o free)

Figura 4.2 Prueba de Aceptación - DHCP - Direccionamiento IP.

MÓDULO DEL SERVICIO PROXY - CACHÉ

La abreviatura empleada para el número de caso de prueba del presente módulo será: *CPAP (Nro.)*. A continuación se presentan las tablas que corresponden a los casos de prueba de aceptación de cada una de las secciones que conforman el módulo del servicio Proxy - Caché:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAP01	Número Historia de Usuario: HUP01
Nombre Caso de Prueba: Ingreso de opciones generales	
Descripción: Se configura las opciones generales de Proxy-Caché.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Cuando se ha seleccionado un puerto distinto al actual y se ha reiniciado el servicio, el usuario deberá configurar las reglas del firewall para redireccionar el tráfico, para evitar que las personas que emplean los servicios obtengan acceso a lugares no permitidos.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio Proxy-Caché). 2. El usuario realiza la configuración de las opciones generales del servicio, modificando ya sea el puerto por el cual se conecta el servicio, el idioma, el tipo de proxy (transparente o con autenticación), el espacio empleado por el servicio para almacenar la información de las páginas visitadas o el nombre que permitirá identificar al servicio. 3. El usuario da clic en el botón Guardar Cambios. 4. El usuario reinicia el servicio Proxy - Caché. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicándole al usuario que los datos ingresados han sido procesados por el servidor. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que emplean los servicios del sistema quedarán temporalmente sin conexión. • Se actualiza la información correspondiente a la nueva configuración y se muestra en la interfaz web. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los datos se han agregado correctamente al servicio Proxy – Caché. • La red local funciona correctamente con los cambios realizados, si una persona que emplea el servicio, accede hacia algún lugar restringido, el servicio le muestra una página en donde se observa los cambios realizados en idioma y nombre de identificación del proxy. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.6 Prueba de Aceptación – Ingreso de opciones generales.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAP02	Número Historia de Usuario: HUP02
Nombre Caso de Prueba: Ingreso de listas de control de acceso	
Descripción: Se ingresa y modifica listas de control de acceso para el servicio Proxy-Caché.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Se deberá verificar en las listas de control de acceso creadas, si la ACL a ingresar ya existe. El usuario deberá crear una política de acceso para la nueva ACL antes de reiniciar el servicio.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio Proxy-Caché). 2. El usuario verifica si la ACL a crear ya existe. 3. Si la ACL no se encuentra agregada, el usuario la crea, para esto ingresa: el nombre de la ACL, selecciona una opción para la ACL, una IP o dominio y puede o no seleccionar las opciones no almacenar en caché y conexión directa. 4. El usuario da clic en el botón Agregar a Lista. 5. El usuario da clic en la opción modificar lista y las modifica si desea. 6. El usuario da clic en el botón Guardar Cambios. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Al agregar la nueva ACL a la lista, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que puede modificar las listas o guardar los cambios. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que los datos ingresados han sido procesados por el servidor y puede aplicar las políticas de acceso. • Una vez guardados los cambios, el sistema ubica al usuario en la sección Políticas de Acceso y muestra el nombre de la ACL agregada en un área de texto. • Se actualiza la información correspondiente a la nueva configuración y se visualiza en la interfaz web. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Las ACLs se han agregado correctamente al servicio Proxy - Caché. • El nombre de la nueva ACL se muestra un área de texto, en la sección Políticas de Acceso. • Las ACLs que se visualizan en la interfaz web son aquellas que se han generado previamente. • La red local funciona correctamente con los cambios realizados. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.7 Prueba de Aceptación – Ingreso de listas de control de acceso.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAP03	Número Historia de Usuario: HUP03
Nombre Caso de Prueba: Ingreso de políticas de acceso	
Descripción: Se ingresa y modifica políticas de acceso para las ACLs existentes.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Se deberá verificar en lista de políticas aplicadas, si la política a ingresar ya existe. El usuario previamente debe haber creado una ACL. El usuario podrá modifica las políticas aplicadas sin necesidad de crear una nueva.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio Proxy-Caché). 2. El usuario selecciona una opción para permitir o denegar el acceso a la ACL creada, para esto selecciona: permitir acceso o negar acceso. Además puede solicitar autenticación para el acceso, dando clic sobre ésta opción. 3. El usuario da clic en el botón Agregar Política. 4. El usuario da clic en la opción modificar políticas, verifica el orden de las políticas, ya que la aplicación de éstas se realiza en el orden en el que se encuentran. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio Proxy - Caché. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Se agrega la nueva política a la lista de políticas aplicadas. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que los datos ingresados han sido procesados por el servidor. • Al reiniciar el servicio Proxy-Caché, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • Se actualiza la información correspondiente a la nueva configuración y se muestra en la interfaz web. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Las políticas de acceso se han agregado correctamente al servicio Proxy - Caché. • Las políticas que se visualizan en la interfaz web son correctas. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.8 Prueba de Aceptación – Ingreso de políticas de acceso.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAP04	Número Historia de Usuario: HUP04
Nombre Caso de Prueba: Ingresar, modificar y eliminar usuarios del servicio Proxy-Caché	
Descripción: Se ingresa, modifica y elimina usuarios que acceden al servicio Proxy-Caché.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Únicamente para ésta configuración no se requiere el reinicio del servicio.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio Proxy-Caché). 2. El usuario ingresa: nombre de usuario, contraseña y confirma la contraseña. 3. El usuario da clic en el botón Agregar usuario. 4. Para eliminar una persona del servicio el usuario lo selecciona de una lista para eliminarlo. 5. Mediante un mensaje de confirmación el usuario determina si elimina o no a una persona que accede al servicio. 6. Para cambiar la contraseña de una persona que accede al servicio, el usuario selecciona un nombre e ingresa la nueva contraseña y su confirmación. 7. El usuario da clic en el botón Cambiar. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicándole al usuario que una persona ha sido agregada al servicio. • Al eliminar un usuario el sistema DAMA muestra un mensaje de confirmación indicándole si desea eliminar o no a una persona del servicio. • El sistema DAMA muestra un mensaje de alerta indicando que se ha eliminado a una persona del servicio. • El sistema DAMA muestra un mensaje de alerta indicando que se ha cambiado la contraseña de una persona. • Se actualiza la información correspondiente a la nueva configuración y se muestra en la interfaz web. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los usuarios se agregan correctamente al servicio Proxy - Caché. • La modificación de la contraseña, así como el proceso de eliminación, funciona correctamente. • Una nueva persona accede al servicio mediante su nombre y contraseña previamente creados. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.9 Prueba de Aceptación – Ingresar, modificar y eliminar usuarios del servicio Proxy-Caché.

Como se puede observar en la Figura 4.3, al crear una ACL que tiene como destino dominio la página de **facebook** y cuyo acceso se encuentra denegado, al intentar ingresar a ésta página, el servicio Proxy – Caché le mostrará una página de error como la que se visualiza en la Figura 4.4.

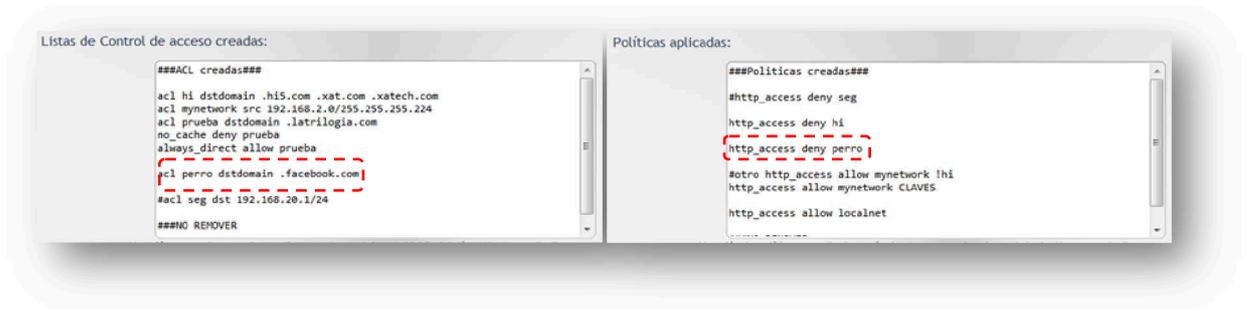


Figura 4.3 Prueba de Aceptación – Proxy-Caché.

Además en la Figura 4.4 se puede observar que cuando el usuario recibe el mensaje de error del servicio, ésta página contiene algunos de los datos configurados en la sección de Opciones Generales, como por ejemplo los cambios realizados en idioma y nombre de identificación del proxy. En éste caso el idioma seleccionado fue **español** y el nombre de identificación **Damaproxy**.



Figura 4.4 Proxy-Caché – Resultado esperado.

MÓDULO DEL SERVICIO DE DIRECTORIOS COMPARTIDOS

La abreviatura empleada para el número de caso de prueba del presente módulo será: **CPAS (Nro.)**. A continuación se presentan las tablas que corresponden a los casos de prueba de aceptación de cada una de las secciones que conforman el módulo del servicio de Directorios Compartidos:

- *Información General*

Ésta sección abarca la historia de usuario HUS01. Permite visualizar tanto el grupo de trabajo como los hosts permitidos para el acceso al servicio. En cuanto a la evaluación de la prueba, el sistema muestra la información correspondiente a cada uno de los aspectos anteriormente mencionados. El usuario deberá ingresar como administrador al módulo del servicio de Directorios Compartidos para poder visualizar ésta información.



Figura 4.5 Prueba de Aceptación - Filtrado Web - Información General.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAS01	Número Historia de Usuario: HUS02-HUSM01
Nombre Caso de Prueba: Ingresar, modificar y eliminar usuarios del servicio de Directorios Compartidos	
Descripción: Se ingresa, modifica y elimina usuarios que acceden al servicio de Directorios Compartidos.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. En el caso de que el host de la persona que accede al servicio emplee el sistema operativo Windows, la contraseña y nombre de usuario que se ingrese deben ser los mismos que posee el host, para facilitar el acceso al servicio. Para acceder a los directorios compartidos un host debe encontrarse en el mismo grupo de trabajo, el cual se muestra en la primera sección de éste módulo. Únicamente para ésta configuración no se requiere el reinicio del servicio.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Directorios Compartidos). 2. El usuario ingresa: nombre de usuario, contraseña y confirma la contraseña. 3. El usuario da clic en el botón Agregar usuario. 4. Para eliminar una persona del servicio el usuario lo selecciona de una lista para eliminarlo. 5. Mediante un mensaje de confirmación el usuario determina si elimina o no a una persona que accede al servicio. 	

<p>6. Para cambiar la contraseña de una persona que accede al servicio, el usuario selecciona un nombre e ingresa la nueva contraseña y su confirmación.</p> <p>7. El usuario da clic en el botón Cambiar.</p>
<p>Resultado Esperado:</p> <ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicándole al usuario que una persona ha sido agregada al servicio. • Al eliminar un usuario el sistema DAMA muestra un mensaje de confirmación indicándole si desea eliminar o no a una persona del servicio. • El sistema DAMA muestra un mensaje de alerta indicando que se ha eliminado a una persona del servicio. • El sistema DAMA muestra un mensaje de alerta indicando que se ha cambiado la contraseña de una persona. • Se actualiza la información correspondiente a la nueva configuración y se muestra en la interfaz web.
<p>Evaluación de la Prueba:</p> <ul style="list-style-type: none"> • Los usuarios se agregan correctamente al servicio de Directorios compartidos. • La modificación de la contraseña, así como el proceso de eliminación, funciona correctamente. • Una nueva persona puede acceder a los directorios compartidos mediante su nombre y contraseña previamente creados. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados.

Tabla 4.10 Prueba de Aceptación – Ingresar, modificar y eliminar usuarios del servicio de Directorios Compartidos.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAS02	Número Historia de Usuario: HUS03
Nombre Caso de Prueba: Añadir un directorio al servicio	
Descripción: Se añade un directorio al servicio para ser compartido.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Para que una persona pueda acceder a los directorios compartidos, debe formar parte de los usuarios que acceden al servicio. Para poder compartir directorios mediante éste servicio un host debe encontrarse en el mismo grupo de trabajo, el cual se muestra en la primera sección de éste módulo.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Directorios Compartidos). 2. El usuario agrega un directorio al servicio, para esto ingresa: nombre del directorio, un comentario, define si el directorio será navegable, si permite el acceso como invitado, define si el directorio será solo de lectura o escritura, según sea el caso podrá ingresar la lista de usuarios válidos y la lista de escritura. 3. El usuario da clic en el botón Guardar Cambios. 4. El usuario reinicia el servicio de Directorios Compartidos (Samba). 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Al guardar los cambios el sistema DAMA muestra un mensaje de alerta indicando que el directorio ha sido agregado al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega en el archivo de configuración del servicio y se muestra en la interfaz web cuando se selecciona el nombre del directorio en la sección Modificar Directorio. • El directorio puede ser compartido en base a la configuración realizada. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • El directorio se agrega correctamente al servicio de Directorios Compartidos. • El directorio cumple con los parámetros configurados y es compartido, según las especificaciones definidas. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.11 Prueba de Aceptación – Añadir un directorio al servicio.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAS03	Número Historia de Usuario: HUS04
Nombre Caso de Prueba: Modificar las características de un directorio compartido	
Descripción: Se modifica las características de un directorio compartido, además se tiene la opción de borrar el directorio del servicio.	
Condiciones de Ejecución: El usuario debe llenar todos los parámetros para la configuración. Un host debe encontrarse en el mismo grupo de trabajo, el cual se muestra en la primera sección de éste módulo.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Directorios Compartidos). 2. El usuario modifica las características de un directorio compartido, para esto: selecciona el nombre del directorio y puede modificar ya sea el comentario, la opción de navegabilidad, la opción de acceso como invitado, la opción de lectura o escritura, según sea el caso la lista de usuarios válidos o la lista de escritura. 3. Para eliminar un directorio del servicio el usuario da clic en el botón Eliminar. 4. Mediante un mensaje de confirmación el usuario determina si elimina o no el directorio del servicio. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio de Directorios Compartidos (Samba). 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Al seleccionar el nombre del directorio compartido, la interfaz web muestra su configuración actual y además se visualiza el botón Eliminar. • Al modificar la lista de usuarios válidos o la lista de escritura, el sistema DAMA, mediante un mensaje de confirmación, permite eliminar los nombres de personas que tiene acceso al directorio seleccionado. • Al modificar el directorio seleccionado y guardar los cambios el sistema DAMA muestra un mensaje de alerta indicando que el directorio ha sido modificado. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y se muestra en la interfaz web, si se selecciona el nombre del directorio modificado. • El directorio puede ser compartido en base a la configuración realizada. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • La modificación del directorio se agrega correctamente al servicio de Directorios Compartidos. • El directorio cumple con los parámetros modificados y es compartido, según las especificaciones definidas. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.12 Prueba de Aceptación – Modificar las características de un directorio compartido.

MÓDULO DEL SERVICIO DE FILTRADO WEB

La abreviatura empleada para el número de caso de prueba para el presente módulo será: *CPAW (Nro.)*. A continuación se presentan las tablas que corresponden a los casos de prueba de aceptación de cada una de las secciones que conforman el módulo del servicio de Filtrado Web:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAW01	Número Historia de Usuario: HUW01-HUWM01
Nombre Caso de Prueba: Modificar las opciones generales del servicio de Filtrado Web	
Descripción: Se configura los parámetros básicos del servicio de Filtrado Web.	
Condiciones de Ejecución: Cuando se ha seleccionado un puerto distinto al actual y se ha reiniciado el servicio, el usuario deberá configurar las reglas del firewall para redireccionar el tráfico, para evitar que las personas que emplean los servicios obtengan acceso a lugares no permitidos.	
Entradas / Pasos de Ejecución: <ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Filtrado Web). 2. El usuario modifica los parámetros básicos del servicio, ya sea seleccionando el puerto por donde se provee el servicio o el idioma de la página web que se mostrará a las personas cuando se les niegue el acceso a algún contenido web. 3. El usuario da clic en el botón Guardar Cambios. 4. El usuario reinicia el servicio de Filtrado Web (Dansguardian). 	
Resultado Esperado: <ul style="list-style-type: none"> • Al guardar los cambios el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y se muestra en la interfaz web. 	
Evaluación de la Prueba: <ul style="list-style-type: none"> • Los cambios realizados se visualizan en la interfaz web. • El servicio se provee por el puerto especificado. • Las personas que acceden a contenidos web no permitidos, visualizan una página web de acceso denegado, en el idioma definido. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.13 Prueba de Aceptación – Modificar las opciones generales del servicio de Filtrado Web.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAW02	Número Historia de Usuario: HUW02
Nombre Caso de Prueba: Ingresar Filtros por MIME	
Descripción: Se filtra el contenido web mediante el ingreso o selección de un tipo MIME.	
Condiciones de Ejecución: Se requiere que el usuario ingrese y/o seleccione un tipo MIME. Un tipo MIME debe ser ingresado según el formato especificado. El usuario puede descomentar el tipo MIME que desee de la lista de filtros que se visualiza en la interfaz web.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Filtrado Web). 2. El usuario ingresa o selecciona un filtro por MIME, para esto: ingresa un tipo de MIME y/o selecciona uno de la lista. 3. El usuario da clic en Agregar a Lista. 4. Para modificar la lista de filtros por MIME, habilita la opción Modificar Lista. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio de Filtrado Web (Dansguardian). 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si el filtro seleccionado o ingresado ya existe en la lista de filtros, el sistema DAMA muestra un mensaje de alerta indicando que el filtro ya existe y no agrega el valor. • El sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados a la lista de filtros por MIME. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y los filtros agregados se visualizan en la interfaz web, en la lista de filtros por MIME. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los nuevos filtros por MIME se visualizan en la interfaz web. • Se filtra el acceso al contenido web según la lista de filtros por MIME. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.14 Prueba de Aceptación – Ingresar Filtros por MIME.

Para efecto de pruebas como se puede observar en la figura 4.6, se aplicó un filtro por MIME, el cual filtrará todo contenido **flash** de las páginas web a las que el usuario ingrese.

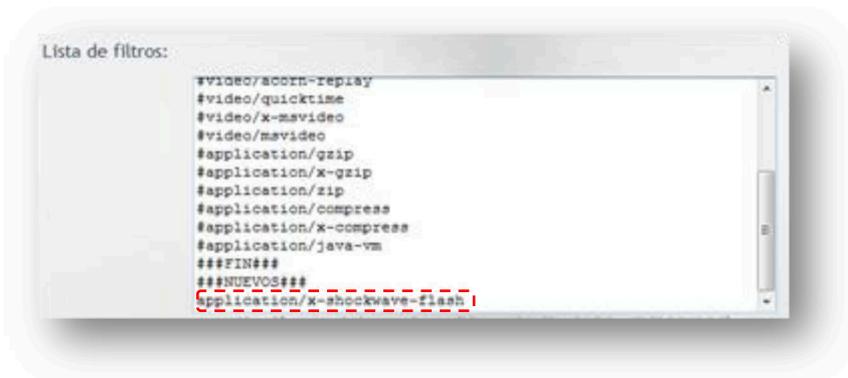


Figura 4.6 Prueba de Aceptación – Ingresar Filtros por MIME.

En la figura 4.7 se pueden observar dos imágenes, la primera corresponde a la página a la que se ingresó (**www.youtube.com**) antes de la aplicación del filtro y la segunda figura corresponde el ingreso a la misma página después de la aplicación del filtro por MIME.

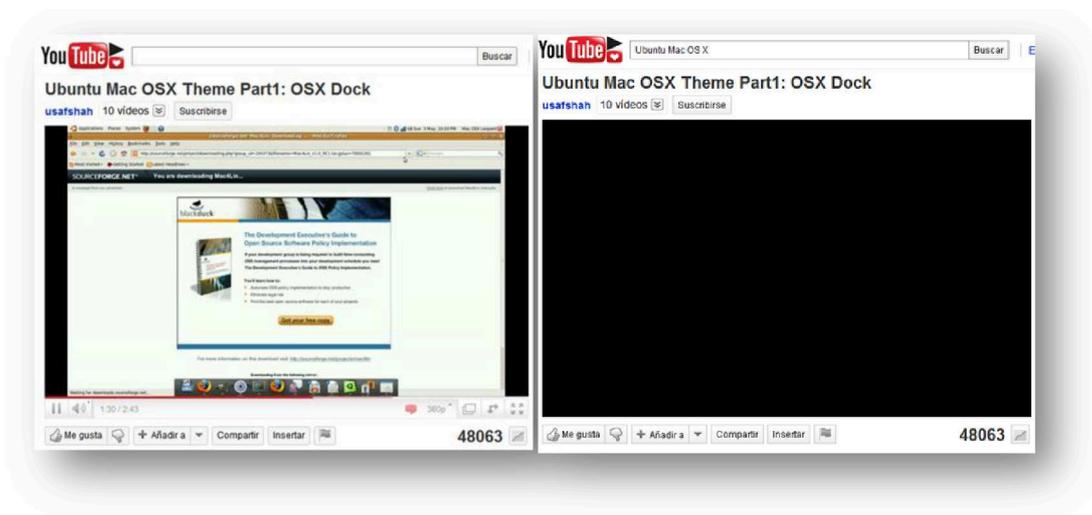


Figura 4.7 Filtros por MIME – Resultado esperado.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAW03	Número Historia de Usuario: HUW03
Nombre Caso de Prueba: Ingresar Filtros por Extensión	
Descripción: Se filtra el contenido web mediante el ingreso una extensión.	
Condiciones de Ejecución: Se requiere que el usuario ingrese una extensión. La extensión debe ingresarse según el formato especificado. El usuario puede descomentar las extensiones que desee de la lista de filtros que se visualiza en la interfaz web.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Filtrado Web). 2. El usuario ingresa una extensión para realizar el proceso de filtrado. 3. El usuario da clic en el botón Agregar a Lista. 4. Para utilizar las extensiones que se encuentran comentadas en la lista de filtros por extensión, el usuario habilita la opción Modificar Lista. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio de Filtrado Web (Dansguardian). 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si el filtro ingresado ya existe en la lista de filtros, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que el filtro ya existe y no agrega el valor. • El sistema DAMA muestra un mensaje de alerta indicando que el filtro ha sido agregado a la lista de filtros por extensión. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando al usuario que los datos han sido agregados al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando al usuario que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y los filtros agregados se visualizan en la interfaz web en la lista de filtros por extensión. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los nuevos filtros por extensión se visualizan en la interfaz web. • Se filtra el acceso al contenido web según la lista de filtros por extensión. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.15 Prueba de Aceptación – Ingresar Filtros por Extensión.

Para efecto de pruebas como se puede observar en la figura 4.8, se aplicó un filtro por extensión, el cual filtrará la descarga de todo contenido cuya extensión sea **.mp3**.

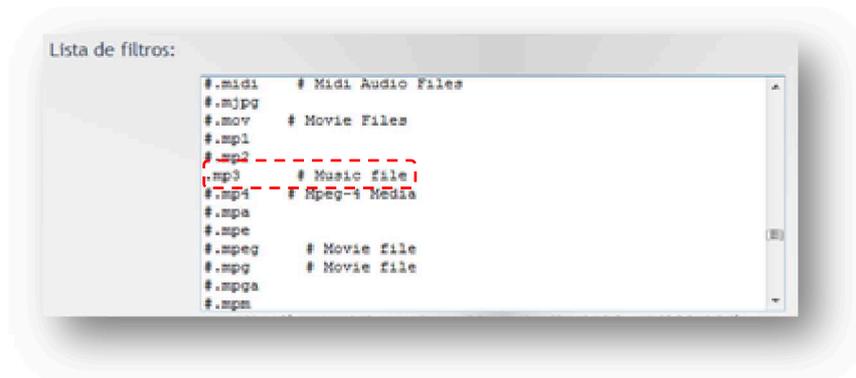


Figura 4.8 Prueba de Aceptación – Ingresar Filtros por Extensión.

En la figura 4.9 se puede observar la página que el servicio de filtrado web le mostrará a la persona que acceda al servicio cuando intente descargar algún archivo cuya extensión se encuentre bloqueada, además ésta página de acceso denegado le indicará la razón por la cual no puede acceder a dicho contenido, en éste caso le indica que la extensión a la que desea acceder se encuentra bloqueada.

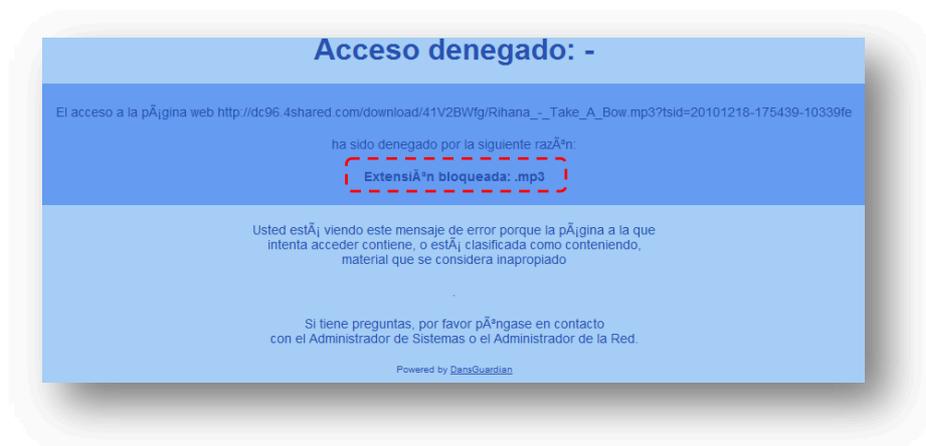


Figura 4.9 Filtros por Extensión – Resultado Esperado.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAW04	Número Historia de Usuario: HUW04
Nombre Caso de Prueba: Ingresar Filtros por Site	
Descripción: Se filtra el contenido web mediante el ingreso y/o selección de un site.	
Condiciones de Ejecución: Se requiere que el usuario ingrese y/o seleccione un dominio. El dominio debe ingresarse según el formato especificado. El usuario puede descomentar los dominios que desee de la lista de filtros que se visualiza en la interfaz web.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Filtrado Web). 2. El usuario ingresa o selecciona un filtro por site, para esto: ingresa un dominio y/o selecciona uno de la lista de dominios del blacklist. 3. El usuario da clic en Agregar a Lista. 4. Para utilizar los dominios que se encuentran comentados en la lista de filtros por site, el usuario habilita la opción Modificar Lista. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio de Filtrado Web (Dansguardian). 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si el filtro ingresado ya existe en la lista de filtros, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que el filtro ya existe y no agrega el valor. • El sistema DAMA muestra un mensaje de alerta indicando que el filtro ha sido agregado a la lista de filtros por site. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y los filtros agregados se visualizan en la interfaz web, en la lista de filtros por site. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los nuevos filtros por site se visualizan en la interfaz web. • Se filtra el acceso a los sites no permitidos y la página que las personas visualizan les informa que ése dominio se encuentra bloqueado. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.16 Prueba de Aceptación – Ingresar Filtros por Site.

Para efecto de pruebas como se puede observar en la figura 4.10, se aplicó un filtro por site, el cual filtrará el acceso a un sitio web, en éste caso se filtra el acceso al sitio **www.hotmail.com**.

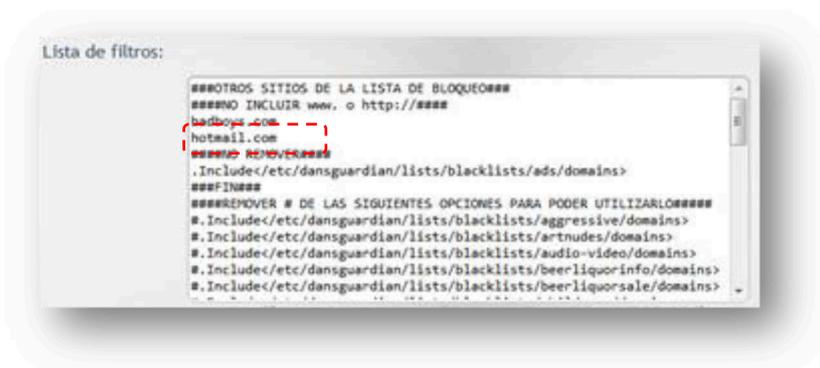


Figura 4.10 Prueba de Aceptación – Ingresar Filtros por Site.

En la figura 4.11 se puede observar la página que el servicio de filtrado web le mostrará a la persona que acceda al servicio cuando intente ingresar a un sitio web que se encuentre bloqueado, además ésta página de acceso denegado le indicará la razón por la cual no puede acceder a dicho contenido, en éste caso le indica que el sitio no es permitido.



Figura 4.11 Filtros por Site – Resultado esperado.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAW05	Número Historia de Usuario: HUW05
Nombre Caso de Prueba: Ingresar Filtros por Frase	
Descripción: Se filtra el contenido web mediante el ingreso y/o selección de una palabra o frase.	
Condiciones de Ejecución: Se requiere que el usuario ingrese y/o seleccione una palabra o frase. La palabra o frase debe ingresarse según el formato especificado. El usuario puede descomentar las frases o palabras que desee de la lista de filtros que se visualiza en la interfaz web.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Filtrado Web). 2. El usuario ingresa o selecciona un filtro por frase, para esto: ingresa una palabra o frase y/o selecciona un contenido de palabras o frases a filtrar de la lista de frases del Phraselist. 3. El usuario da clic en Agregar a Lista. 4. Para utilizar las frases o palabras que se encuentran comentadas en la lista de filtros por frase, el usuario habilita la opción Modificar Lista. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio de Filtrado Web (Dansguardian). 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si el filtro ingresado ya existe en la lista de filtros, el sistema DAMA muestra un mensaje de alerta indicándole al usuario que el filtro ya existe y no agrega el valor. • El sistema DAMA muestra un mensaje de alerta indicando que el filtro ha sido agregado a la lista de filtros por frase. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados al servicio. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden al servicio quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y los filtros agregados se visualizan en la interfaz web, en la lista de filtros por frase. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Los nuevos filtros por frase se visualizan en la interfaz web. • Se filtra el acceso al contenido web en base a los filtros por frase agregados y la página que las personas visualizan les informa que el acceso está denegado debido a que se encontró una frase no permitida. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.17 Prueba de Aceptación – Ingresar Filtros por Frase.

Para efecto de pruebas como se puede observar en la figura 4.12, se aplicó un filtro por frase, el cual filtrará el acceso a todo contenido web que incluya la palabra **amor**.



Figura 4.12 Prueba de Aceptación – Ingresar Filtros por Frase.

En la figura 4.13 se puede observar la página que el servicio de filtrado web le mostrará a la persona que acceda al servicio cuando intente ingresar a un contenido web que incluya la frase no permitida, además ésta página de acceso denegado le indicará la razón por la cual no puede acceder a dicho contenido, en éste caso le indica que se ha encontrado una frase no permitida.



Figura 4.13 Filtros por Frase – Resultado esperado.

MÓDULO DEL SERVICIO DE FIREWALL

La abreviatura empleada para el número de caso de prueba para el presente módulo será: *CPAF (Nro.)*. A continuación se presentan las tablas que corresponden a los casos de prueba de aceptación de cada una de las secciones que conforman el módulo del servicio de Firewall:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAF01	Número Historia de Usuario: HUF01-HUFM01
Nombre Caso de Prueba: Creación y Modificación de Zonas	
Descripción: Se crea y modifica zonas a ser administradas por el firewall.	
Condiciones de Ejecución: Se requiere que el nombre de la zona contenga hasta cinco caracteres. El usuario no debe remover las líneas indicadas de la lista de zonas definidas en el firewall, especialmente aquella cuyo nombre corresponde a “loc2” ya que ésta zona es utilizada para acceder a la interfaz web, caso contrario el sistema deja de funcionar correctamente.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Firewall). 2. El usuario ingresa la zona a ser administrada por el firewall, para esto ingresa el nombre de la zona. 3. El usuario da clic en el botón Agregar Zona. 4. Para modificar las zonas que se encuentran en la lista de zonas definidas en el firewall, el usuario habilita la opción Modificar Zonas. 5. El usuario da clic en el botón Guardar Cambios. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si la zona ingresada ya existe, el sistema DAMA muestra un mensaje de alerta indicando que la zona ya existe y no agrega el valor. • El sistema DAMA muestra un mensaje de alerta indicando que la zona ha sido agregada. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados al servicio. • La interfaz redirige al usuario a la sección de Interfaces para la generación de relaciones entre interfaces de red y zonas. • La nueva información se muestra en la interfaz web, en la lista de zonas definidas en el firewall. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Las zonas agregadas se visualizan en la interfaz web, en el área de texto destinado para ello. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.18 Prueba de Aceptación – Creación y Modificación de Zonas.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAF02	Número Historia de Usuario: HUF02-HUFM01
Nombre Caso de Prueba: Creación de relaciones entre Interfaces y Zonas	
Descripción: Se crea y modifica relaciones entre interfaces y zonas.	
Condiciones de Ejecución: El usuario empleará únicamente las zonas que han sido creadas. Se requiere que el usuario seleccione una zona y una interfaz. El usuario no debe remover las líneas indicadas de la lista de relaciones creadas, caso contrario el sistema deja de funcionar correctamente.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Firewall). 2. El usuario crea una relación entre una zona y una interfaz de red, para esto: selecciona una zona y una interfaz de red. 3. El usuario da clic en el botón Agregar Relación. 4. Para modificar las relaciones creadas, el usuario habilita la opción Modificar Interfaces. 5. El usuario da clic en el botón Guardar Cambios. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si la relación creada ya existe, el sistema DAMA muestra un mensaje de alerta indicando que la zona ya está siendo utilizada y no agrega la relación. • El sistema DAMA muestra un mensaje de alerta indicando que la relación ha sido agregada. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados al servicio. • La interfaz redirige al usuario a la sección de creación de políticas de alto nivel. • La nueva información se muestra en la interfaz web, en la lista de interfaces definidas en el firewall. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Las relaciones creadas se visualizan en la interfaz web, en el área de texto destinado para ello. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.19 Prueba de Aceptación – Creación de relaciones entre Interfaces y Zonas.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAF03	Número Historia de Usuario: HUF03-HUFM01
Nombre Caso de Prueba: Creación de Políticas de Alto Nivel	
Descripción: Se crea y modifica políticas de acceso para poder establecer comunicaciones entre las diferentes zonas de manera general.	
Condiciones de Ejecución: El usuario empleará únicamente las zonas creadas y que han sido relacionadas con una interfaz. Se requiere que el usuario seleccione una zona origen, zona destino y una política. El usuario no debe remover las líneas indicadas de la lista de políticas aplicadas, caso contrario el sistema deja de funcionar correctamente.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Firewall). 2. El usuario crea una política de acceso, para esto: selecciona una zona de origen, zona de destino y una política. 3. El usuario da clic en Agregar Política. 4. Para modificar las políticas aplicadas, el usuario habilita la opción Modificar Políticas. 5. El usuario da clic en el botón Guardar Cambios. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • Si la política creada ya existe, el sistema DAMA muestra un mensaje de alerta indicando que la relación entre las zonas ya existe y no agrega la política. • El sistema DAMA muestra un mensaje de alerta indicando que la política ha sido agregada. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido agregados al servicio. • La interfaz redirige al usuario a la sección de reglas de acceso. • La nueva información se muestra en la interfaz web, en la lista de políticas aplicadas. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Las políticas aplicadas se visualizan en la interfaz web, en el área de texto destinado para ello. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.20 Prueba de Aceptación – Creación de Políticas de Alto Nivel.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAF04	Número Historia de Usuario: HUF04-HUFM01
Nombre Caso de Prueba: Creación de Reglas de Acceso Específicas	
Descripción: Se crea y modifica reglas de acceso específicas para las solicitudes de conexión.	
Condiciones de Ejecución: El usuario empleará únicamente las zonas creadas y que han sido relacionadas con una interfaz. Se requiere que el usuario seleccione una acción, para su posterior modificación. El usuario no debe remover las líneas indicadas de la lista de reglas creadas, caso contrario el sistema deja de funcionar correctamente. Las reglas deben ser ingresadas en el orden correcto.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Firewall). 2. El usuario crea una regla de acceso, para esto selecciona una acción. 3. El usuario da clic en Agregar Acción. 4. El usuario ingresa la regla para la acción seleccionada en el área de texto que corresponde a la lista de reglas creadas. 5. El usuario da clic en el botón Guardar Cambios. 6. El usuario reinicia el servicio de Firewall. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicando que la acción ha sido agregada y que debe crear la regla en el área de texto para la acción seleccionada. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido procesados y que la aplicación de reglas sigue el orden mostrado en el área de texto. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden a los servicios del sistema quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y las nuevas reglas se visualizan en la interfaz web en la lista de reglas creadas. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • Las reglas creadas se visualizan en la interfaz web, en el área de texto destinado para ello. • Las reglas se aplican de acuerdo al orden establecido en el área de texto que corresponde a la lista de reglas creadas. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.21 Prueba de Aceptación – Creación de Reglas de Acceso Específicas.

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAF05	Número Historia de Usuario: HUF05-HUFM01
Nombre Caso de Prueba: Creación de Enmascaramiento	
Descripción: Se agrega la interfaz con conexión a Internet para el proceso de enmascaramiento.	
Condiciones de Ejecución: Se requiere que el usuario agregue la interfaz con conexión a Internet, para su posterior enmascaramiento. El usuario no debe remover las líneas indicadas de la lista de enmascaramiento actual, caso contrario el sistema deja de funcionar correctamente.	
Entradas / Pasos de Ejecución:	
<ol style="list-style-type: none"> 1. El usuario accede al sistema como administrador (Módulo del servicio de Firewall). 2. El usuario da clic en Agregar Interfaz, para añadirla a la lista de enmascaramiento actual. 3. En el área de texto que pertenece al enmascaramiento actual, el usuario ingresa la información de red a quien proveerá el servicio. 4. El usuario da clic en el botón Guardar Cambios. 5. El usuario reinicia el servicio de Firewall. 	
Resultado Esperado:	
<ul style="list-style-type: none"> • El sistema DAMA muestra un mensaje de alerta indicando que la interfaz ha sido añadida y que debe agregar la información necesaria para el proceso de enmascaramiento. • Al guardar los cambios, el sistema DAMA muestra un mensaje de alerta indicando que los datos han sido procesados. • Al reiniciar el servicio, el sistema le indica al usuario que las personas que acceden a los servicios del sistema quedarán temporalmente sin conexión. • El sistema DAMA muestra un mensaje de alerta indicando que el servicio se ha reiniciado. • La información correspondiente a la nueva configuración se agrega al servicio y se visualiza en la interfaz web, en la lista de enmascaramiento actual. 	
Evaluación de la Prueba:	
<ul style="list-style-type: none"> • El enmascaramiento actual se visualiza en la interfaz web, en el área de texto destinado para ello. • El sistema DAMA cumple satisfactoriamente con todos los resultados esperados. 	

Tabla 4.22 Prueba de Aceptación – Creación de Enmascaramiento.

MÓDULO DEL SERVICIO ESTADO DEL SISTEMA OPERATIVO

Éste módulo tiene como fin proporcionar información acerca del estado del sistema operativo al usuario, por ésta razón en las siguientes pruebas de aceptación no se definen entradas o pasos de ejecución, ya que en éste caso el usuario no interactúa con la interfaz web, debido a que las únicas actividades que realizará el usuario son visualizar la información que le presenta el sistema DAMA y apagar o reiniciar el servidor, así como reiniciar los servicios que presenta el sistema.

La abreviatura empleada para el número de caso de prueba para el presente módulo será: *CPAE (Nro.)*. A continuación se presentan las tablas que corresponden a los casos de prueba de aceptación de cada una de las secciones que conforman el módulo del servicio de Estado del Sistema Operativo.

- Disco Duro:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAE01	Número Historia de Usuario: HUE01-HUE02- HUEM01
Nombre Caso de Prueba: Visualizar Estadísticas de uso del Disco Duro y Particiones	
Descripción: Se visualiza información acerca del estado del disco duro, así como de discos duros adicionales y particiones, en caso de que los tenga.	
Condiciones de Ejecución: El usuario debe acceder al sistema como administrador (Módulo del Servicio Estado del Sistema Operativo). La información que corresponde a discos duros adicionales y particiones solo se podrá visualizar en caso de que existan.	
Entradas / Pasos de Ejecución:	
Resultado Esperado: <ul style="list-style-type: none"> • Para el estado del disco duro la información a visualizar es: porcentajes de utilización, tamaño total y disponible tanto para el sistema operativo como para el directorio personal. • Para el estado de discos duros adicionales o particiones la información a visualizar es: porcentaje de utilización, tamaño total y disponible, tanto para el disco duro adicional como para la partición. • Al reiniciar y apagar el servidor el sistema DAMA le indica al usuario que el servidor se está reiniciando o apagando respectivamente. • Las gráficas de barras que se presentan en ésta sección, deben reflejar el porcentaje de utilización. 	
Evaluación de la Prueba: <ul style="list-style-type: none"> • El sistema DAMA proporciona los valores esperados tanto para el disco duro como para los discos duros adicionales y particiones. 	

Tabla 4.23 Prueba de Aceptación – Visualizar Estadísticas de uso del Disco Duro y Particiones.

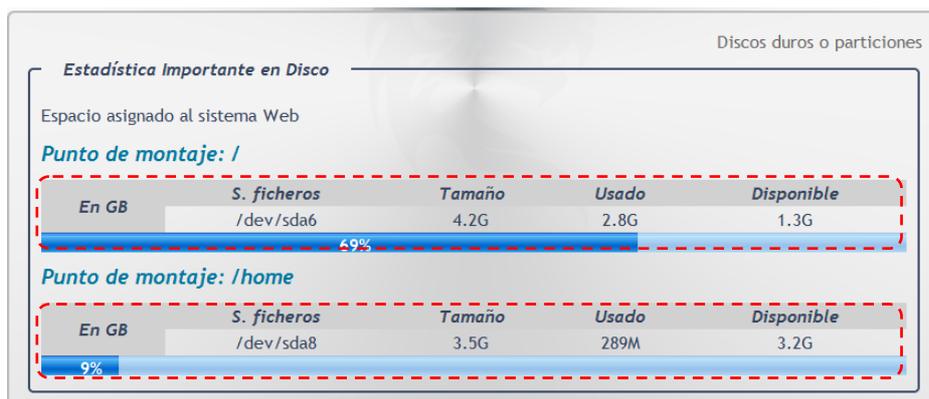


Figura 4.14 Prueba de Aceptación – Estado del Sistema Operativo – Disco Duro.

- RAM y CPU:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAE02	Número Historia de Usuario: HUE03-HUE04- HUEM02
Nombre Caso de Prueba: Visualizar Estadísticas de Memoria y de uso de CPU	
Descripción: Se visualiza información acerca del estado de memoria como del estado de CPU.	
Condiciones de Ejecución: El usuario debe acceder al sistema como administrador (Módulo del Servicio Estado del Sistema Operativo).	
Entradas / Pasos de Ejecución:	
Resultado Esperado: <ul style="list-style-type: none"> • Para el estado de memoria, la información a visualizar es: porcentaje de utilización, tamaño total y disponible, tanto para la memoria física del sistema operativo, memoria utilizada por el kernel en ese momento, memoria física asignada para intercambio como para la memoria física total. • Para el estado de CPU, la información a visualizar es: porcentaje de uso, vendedor, modelo velocidad, tamaño de caché y número de núcleos, en caso de existir más de un núcleo, se proporcionará ésta información para cada uno de ellos. Además se presenta la temperatura actual del CPU. • Al reiniciar y apagar el servidor el sistema DAMA le indica al usuario que el servidor se está reiniciando o apagando respectivamente. • Las gráficas de barras que se presentan en ésta sección, deben reflejar el porcentaje de utilización. 	
Evaluación de la Prueba: <ul style="list-style-type: none"> • El sistema DAMA proporciona los valores esperados tanto para el estado de memoria como para el uso de CPU. 	

Tabla 4.24 Prueba de Aceptación – Visualizar Estadísticas de Memoria y de uso de CPU.



Figura 4.15 Prueba de Aceptación – Estado del Sistema Operativo – RAM y CPU.

- Información de Red:

Caso de Prueba de Aceptación	
Número Caso de Prueba: CPAE03	Número Historia de Usuario: HUE05- HUE06- HUE07
Nombre Caso de Prueba: Visualizar Información de Red	
Descripción: Se visualiza información de cada interfaz del servidor, así como de las rutas empleadas por el sistema para enviar paquetes.	
Condiciones de Ejecución: El usuario debe acceder al sistema como administrador (Módulo del Servicio Estado del Sistema Operativo).	
Entradas / Pasos de Ejecución:	
Resultado Esperado: <ul style="list-style-type: none"> • Se proporciona información detallada tanto para la interfaz con salida a Internet, interfaz local como para las rutas que emplea el sistema para enviar los paquetes. • Al reiniciar y apagar el servidor, el sistema DAMA le indica al usuario que el servidor se está reiniciando o apagando respectivamente. • Las gráficas de barras que se presentan en ésta sección, deben reflejar el porcentaje de utilización. 	
Evaluación de la Prueba: <ul style="list-style-type: none"> • El sistema DAMA cumple satisfactoriamente con los resultados esperados. 	

Tabla 4.25 Prueba de Aceptación – Visualizar Información de Red.

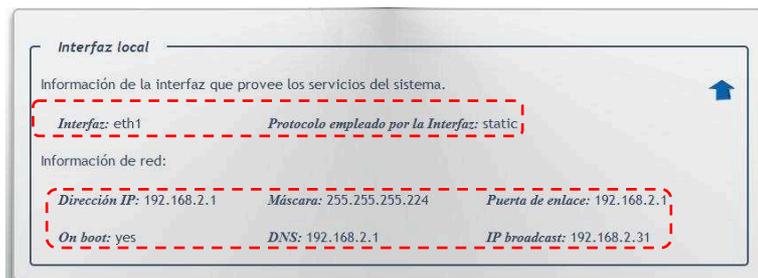


Figura 4.16 Prueba de Aceptación – Estado del Sistema Operativo – Información de Red.

- Información Adicional:

Ésta sección abarca las historias de usuario HUE08 y HUE09, permite visualizar información acerca de usuarios, tiempo activo del sistema e información del kernel. En cuanto a la evaluación de la prueba, el sistema muestra información detallada de cada uno de los aspectos anteriormente mencionados.

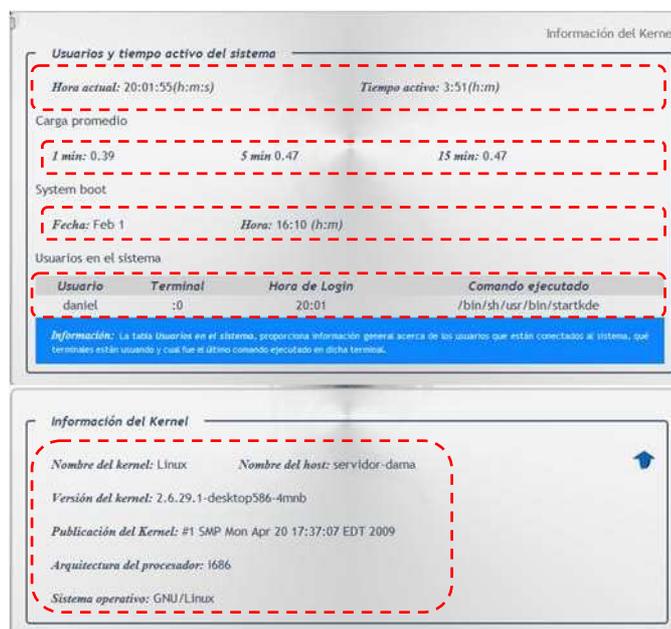


Figura 4.17 Prueba de Aceptación – Estado del Sistema Operativo – Información Adicional.

MÓDULO DEL SERVICIO DE GRÁFICAS

Éste módulo comprende las historias de usuario HUG01, HUG02, HUG03 y HUGM01. Al momento de realizar las pruebas el sistema respondió a cada una de las especificaciones descritas en cada una de las historias de usuario mencionadas. La única actividad que el usuario debe realizar en éste módulo es dar clic sobre la gráfica por día que se visualiza primero para desplegar las demás gráficas solicitadas por el usuario. Para poder ingresar a éste módulo, el usuario debe acceder al sistema como administrador.

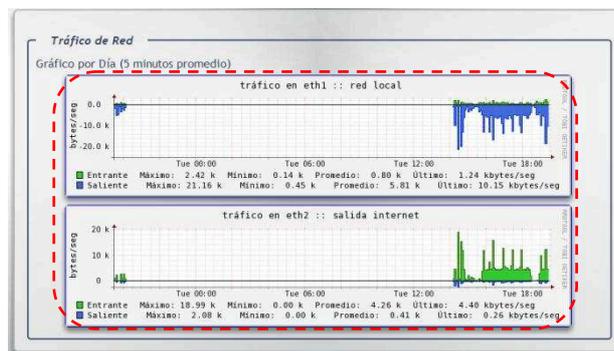


Figura 4.18 Prueba de Aceptación – Módulo del Servicio de Gráficas.

MÓDULO DE PÁGINA INICAL

Éste módulo corresponde a la página de bienvenida al sistema DAMA, la información que aquí se presenta cumple con las especificaciones descritas en la historia de usuario HUB01. En el proceso de pruebas se pudo comprobar que cualquier persona que accede al sistema puede visualizar la página principal, pero únicamente el administrador puede acceder a los módulos de los servicios que forman parte del sistema DAMA, ya que para poder ingresar a cada uno de ellos se requiere de un nombre de usuario y contraseña, que como se especificó en la sección 2.1.3.5 del capítulo 2, únicamente serán de conocimiento del administrador de la red.

Figura 4.19 Prueba de Aceptación - Página Inicial – Autenticación del Administrador de red.

CAPÍTULO 5

5 CONCLUSIONES Y RECOMENDACIONES

En base al desarrollo e implementación del sistema de administración web DAMA, se detallan las siguientes conclusiones y recomendaciones.

5.1 CONCLUSIONES

- Las pequeñas empresas tienen una gran representación en el sector económico del país, a pesar de esto los desarrolladores de tecnología muestran poco interés en generar soluciones que se adapten a las necesidades de éste tipo de organizaciones, en la mayoría de casos las soluciones tecnológicas que permiten la administración de servicios existentes en el mercado, se han desarrollado para grandes empresas que por lo general cuentan con un número mayor a 200 usuarios, lo cual para una empresa tipo SOHO (Small Home Small Office) implica una gran inversión. Esto involucra que empresas de éste tipo tengan pocas opciones de TIC's (Tecnologías de Información y Comunicación), sobre todo soluciones que permitan la administración y configuración de servicios dentro de la red, que no sean sobredimensionadas a sus necesidades, de fácil gestión y que no requieran de mucha inversión.

Tomando en cuenta éstos antecedentes se ha desarrollado el presente proyecto, en el que se hace uso de herramientas de libre distribución y adaptado a las necesidades requeridas por éste tipo de empresas, para lo cual se ha considerado al sistema operativo Linux como la principal alternativa para el desarrollo de la interfaz web DAMA, debido a sus características de gratuidad y funcionalidad; esto implica que el presente proyecto ayude a pequeñas empresas a contar con un sistema de administración de servicios vía web que cubra sus necesidades.

- La metodología de desarrollo XP se ha ajustado correctamente a las necesidades requeridas para la evolución del presente proyecto, ya que cada una de sus características han sido tomadas en cuenta, entre ellas tenemos: programación en pareja, desarrollo enfocado en la funcionalidad y perspectiva del usuario, diseños simplificados. Además la principal

ventaja que se ha aprovechado es la característica de la metodología que permite cambios continuos en el transcurso del desarrollo y su adaptabilidad a las necesidades del cliente a través de historias de usuario, iteraciones y pruebas de aceptación, que dan como resultado la generación de un software de calidad en un lapso de tiempo relativamente corto.

- El estándar IEEE 830-1998 permitió especificar de forma clara y detallada los pasos a seguir para obtener los requerimientos necesarios para el desarrollo del presente proyecto, además la aplicación de éste estándar da como resultado la integración de requerimientos de forma general, acogiendo el enfoque tanto de los desarrolladores, del usuario como del cliente, obteniendo de ésta manera un documento que acoge cada una de las características que el sistema desarrollado va a realizar y aquellas en las cuales se encontrará limitado.
- El desarrollo del presente proyecto ha requerido el empleo de diferentes lenguajes de programación cuyo funcionamiento en conjunto han permitido desarrollar una interfaz web funcional y amigable para el usuario. Se han empleado lenguajes de programación tanto en el lado del cliente como del servidor. Dentro de los lenguajes que se empleó en el lado del cliente están JavaScript, CSS y HTML, los cuales han cumplido una tarea específica dentro del desarrollo. CSS ha permitido desarrollar la parte visual de la interfaz web, HTML su contenido y JavaScript los procesos de validación de la información proporcionada por el usuario del sistema. Los lenguajes de programación que se emplearon en el lado del servidor son Perl y Bash. Perl ha constituido la parte medular, ya que dentro de las actividades más relevantes que ha permitido realizar, es la modificación de archivos dentro del servidor. Por otro lado Bash ha sido utilizado como un lenguaje de programación auxiliar, que ejecutó tareas propias del sistema operativo en el servidor.
- Al emplear una metodología de desarrollo como XP en la cual una de sus características más relevantes es el desarrollo del proyecto en un lapso de tiempo relativamente corto, se ha buscado emplear herramientas que no

requieran de un uso excesivo de tiempo para entender su funcionamiento, por lo que se optó por utilizar CGI (Common Gateway Interface), ya que su utilización es de fácil aprendizaje y no se requiere realizar una configuración previa en el servidor web, dado que al emplear el servidor web Apache para alojar la interfaz de administración, éste por defecto incluye un directorio para albergar éste tipo de interfaces, enfocándonos rápidamente en el lenguaje de programación Perl que como se mencionó es el más importante en el proceso de desarrollo.

- Todos los servicios que son configurados por el sistema de administración web DAMA cuentan con un archivo de configuración en el lado del servidor cuya manipulación se ha facilitado gracias al uso del lenguaje interpretado Perl, ya que al ser archivos en modo texto, Perl fácilmente permite leerlos y modificarlos mediante funciones propias del lenguaje.
- La interfaz web DAMA permite administrar varios servicios tales como servicio de DHCP, Proxy-Caché, Filtrado web, Firewall, directorios compartidos, estado del sistema operativo y visualización de estadísticas del servidor, a través de un explorador web empleado por el usuario, evitando la configuración de cada servicio de manera independiente en el lado del servidor, de ésta manera el usuario podrá realizar ésta tarea desde cualquier punto de acceso en la red.
- La navegabilidad en la interfaz web ha sido satisfactoria dado que al emplear una estructura de diseño web tipo red, el usuario puede acceder rápidamente a cada uno de los módulos que comprende el sistema, evitando confusiones en el proceso de navegación a través de la interfaz web, ya que como éste tipo de estructura sugiere el usuario puede identificar fácilmente el módulo y sección en el que se encuentra.
- Uno de los módulos importantes en el sistema de administración web DAMA es el módulo de gráficas, debido a que ayuda al webmaster o administrador de la red en el análisis de la información acerca del comportamiento de ciertos aspectos del servidor tales como rendimiento de red, uso de memoria y carga del CPU, para en base a ésta información

aplicar políticas que mejoren el rendimiento del servidor en la red que administra.

- Mediante el uso del sistema de administración web, específicamente del módulo de filtrado web, el administrador de la red o webmaster puede filtrar aquel contenido web que malgaste ancho de banda y que considere inapropiado para ser usado dentro de la empresa, contribuyendo de ésta manera al uso adecuado de la red con fines estrictamente laborales.

5.2 RECOMENDACIONES

- Se recomienda la unificación de una metodología ágil y el estándar IEEE 830-1998, para el desarrollo de éste tipo de proyectos, ya que de ésta manera se podrá obtener una visión clara de cómo se estructurará el desarrollo del proyecto y que actores formarán parte de él, además se podrá planificar de manera detallada el tiempo de desarrollo e implementación para cada una de las actividades a realizar.
- Es recomendable utilizar una metodología de desarrollo como XP, ya que mediante el uso de historias de usuario y casos de pruebas de aceptación se facilitará la codificación y modificación continua en el proceso de desarrollo con la finalidad de obtener un producto que satisfaga las necesidades del cliente.
- Se debe realizar un análisis previo de las características hardware y software necesarios para el desarrollo de un proyecto de éste tipo, para que de ésta manera se obtenga un producto final que no sea sobredimensionado.
- Se recomienda que el webmaster o administrador de red tenga conocimiento acerca de cómo se realiza la configuración de cada uno de los servicios disponibles, dado que de ésta manera podrá entender fácilmente como se realiza la configuración de servicios desde la interfaz web DAMA, evitando que se provoque una configuración errónea en algún servicio y se emplee tiempo en la detección del problema.

- Se sugiere que cuando se realicen cambios en el módulo del servicio DHCP, se los realice en horarios en los cuales los usuarios de la red mantengan sus hosts apagados o cuando la actividad en la red no sea crítica, ya que una vez reiniciado el servicio, después de realizar un cambio en la red, se requerirá que los usuarios renueven su concesión.
- Todas las configuraciones realizadas desde la interfaz web que provocan cambios en un servicio, requerirán que el usuario los reinicie para que se hagan efectivos, por lo cual se recomienda reiniciar el servicio modificado, caso contrario los cambios no se harán efectivos y se retornará a la configuración previa.
- Se recomienda el uso de un lenguaje interpretado como Perl para el desarrollo de éste tipo de proyecto, dado que la configuración de los servicios se realiza mediante la manipulación de texto, al contar Perl con un conjunto de funciones que han sido desarrolladas para manipular la información contenida en un archivo de texto, se facilita ampliamente el desarrollo de ésta actividad.
- Si bien se emplea un mecanismo de autenticación que restringe el acceso de cualquier tipo de usuario a la configuración de los servicios, se recomienda que los parámetros de autenticación sean de uso exclusivo y responsabilidad del administrador de la red o webmaster, para evitar que personas ajenas a ésta actividad modifiquen las configuraciones presentes en la interfaz web DAMA provocando de ésta manera el mal funcionamiento del servidor.

REFERENCIAS BIBLIOGRÁFICAS

• Libros y Manuales

1. POWEL A., Thomas, **Manual de Referencia HTML**, Primera Edición, McGraw-Hill, Interamericana De España, España, 1998.
2. CIMANO D., James, **Redes Locales Intranets**, Primera Edición, Editorial Paraninfo, Madrid-España, 1997.
3. AFERGAN, Michael; FARRAR, Brian, **Programación en WEB 6 en 1**, Primera Edición, Prentice-Hall Hispanoamérica, México, 1997.
4. TANENBAUM S., Andrew, **Redes de Ordenadores**, Segunda Edición, Prentice-Hall Hispanoamérica, 1991.
5. SERVATI, Ai; BREMNER, Lynn, **La Biblia de Intranet**, McGraw-Hill, México, 1998.
6. KORETSKY, Robert, **El Libro de Linux**, Primera Edición, España, 2005.
7. PETERSEN, Richard, **Manual de Referencia de Linux**, Sexta Edición, McGraw-Hill, 2009.
8. SHAH, Steve, **Manual de Administración de Linux**, McGraw-Hill, España, 2001.
9. SOMMERVILLE, Ian, **Ingeniería del Software**, Séptima Edición, Pearson Addison Wesley, Madrid – España, 2005.

• Direcciones Electrónicas

1. RAGGETT, Dave; LE HORS, Arnaud; JACOBS, Ian. **HTML 4.01 Specification.**

<http://www.w3.org/TR/html401/>

2. PÉREZ E., Javier. **Introducción a XHTML.**

<http://www.librosweb.es/xhtml/>

3. ALBEZA, Belén. **HTML+CSS.**

http://bergantells.net/~jander/weblog/tuto-html_a3.pdf

4. FERRER, Jorge; GARCÍA, Víctor; GARCÍA, Rodrigo. **Curso Completo de HTML.**

<http://lucas.hispalinux.es/Manuales-LuCAS/doc-curso-html/doc-curso-html.pdf>

5. KIOSKEA. **El Protocolo HTTP**

<http://es.kioskea.net/contents/internet/http.php3>

6. PROGRAMACIÓN WEB. **El Protocolo HTTP**

<http://www.programaciónweb.net/articulos/articulo/?num=409>

7. HASSAN E., Ahmed; HOLT C., Richard. **A Reference Architecture for Web Servers.**

<http://www.bauhaus-stuttgart.de/dagstuhl/HassanHolt.pdf>

8. DELGADO P., Javier. **Tutorial de CGI.**

<http://proton.ucting.udg.mx/tutorial/cgi/cgi.PDF>

9. VALSESIA, Edoardo. **Curso CGI**

<http://www.htmlpoint.com/cgi/index.html>

10. VALDÉS, Damián. **Lenguajes de Programación para La Web.**

<http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/>

11. MYGNET. **Crear un CGI en lenguaje C.**

http://www.mygnet.net/articulos/cgi/crear_un_cgi_en_lenguaje_c.15

12. KORPELA, Jukka. **Getting Started with CGI Programming in C.**

<http://www.cs.tut.fi/~jkorpela/forms/cgic.html>

13. ESCALONA C., María; GONZÁLEZ R., José. **Metodología y Técnicas en Proyectos Software para la Web.**

<http://www.lsi.us.es/docencia/get.php?id=1135>

14. BRITO A., Karennny. **Metodologías de Desarrollo para Aplicaciones Web.**

<http://www.eumed.net/libros/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm>

15. Proyectos Ágiles. Metodología Scrum.

<http://www.proyectosagiles.org/que-es-scrum>

16. Wikipedia. Scrum.

http://es.wikipedia.org/wiki/Scrum#Scrum_aplicado_al_desarrollo_de_software

17. Anónimo. Metodología Crystal Clear.

http://www.cs.umss.edu.bo/doc/material/mat_gral_130/Metodologia%20Crystal.doc

18. Tercer Planeta. Crystal Clear, una forma de trabajo.

<http://blog.tercerplaneta.com/2007/02/ms-all-de-las-capacidades-tcnicas-que.html>

19. PRIOLO, Sebastián. Programación Extrema.

ftp://190.5.199.3/jjurado/Mejora%20de%20Procesos/XP_6JAI.pdf

20. CASTILLO, Oswaldo; FIGUEROA, Daniel; SEVILLA, Héctor. Programación Extrema.

<http://programacionextrema.tripod.com/index.htm>

21. SÁNCHEZ G., Carlos. Metodología eXtreme Programming.

<http://oness.sourceforge.net/proyecto/html/ch05.html>

22. HUARACHI, Maritza. Metodología Ágil ASD (Desarrollo de Software Adaptable).

<http://www.profesionales.com.mx/images/18149/METODOLOGIA%20AGIL%20ASD.ppt>

23. Anónimo. Sitio Web: Estructura y Navegación.

<http://www.ite.educacion.es/formacion/materiales/107/cd/html/pdf/html13.pdf>

24. CORTÉS A., José. Estructura de la Web.

<http://www.joseacortes.com/recursosweb/8pasos/estructura.htm>

25. M., Verónica. Como organizar una Página Web, Diseño de la Estructura.

<http://www.lawebera.es/de0/organizar-pagina-web.php>

26. GALDÚ, Enrique. RRDtool, herramienta de bases de datos en round robin.

<http://www.uji.es/bin/ponencias/irrdtool.pdf>

27. VICENTE, Carlos. **RRDTool.**

http://www.allietti.com/tutorials/WALCs/walc2007/track_4/rrdtool.pdf

28. BENÍTEZ, Magali. **Tecnologías Web del lado del Servidor para Páginas Web Dinámicas.**

<http://www.duamu.com/re/articulo/1344/id/467/articulos-paginas-dinamicas--cgi-php-asp-jsp-y-muc.html>

29. Perl en Español. **Tutoriales.**

<http://perlenespanol.com/tutoriales/>

30. Anónimo. **Lenguaje de Programación Perl.**

<http://www.frt.utn.edu.ar/sistemas/paradigmas/Perl.htm>

A. ANEXO A

LENGUAJE PERL PARA EL MANEJO DE ARCHIVOS DE TEXTO EN EL LADO DEL SERVIDOR

PERL (Practical Extraction and Report Language) es un lenguaje desarrollado por Larry Wall con el objetivo de permitir la administración de sistemas UNIX de una manera más simple, en sus inicios fue desarrollado para trabajar exclusivamente en UNIX, en la actualidad puede trabajar en cualquier sistema operativo donde se encuentre el intérprete Perl. La utilidad principal de Perl es permitir la manipulación de archivos de texto permitiendo modificar contenidos, realizar búsquedas, crear archivos. Al contar con esta utilidad y tomando en cuenta que los servicios dentro de un sistema Linux se manejan a manera de archivos de configuración, Perl constituye un lenguaje idóneo para crear scripts que permitan manipular las configuraciones de los servicios dentro de un servidor Linux. Perl a diferencia de otros lenguajes constituye un lenguaje de interpretado que a diferencia de otros lenguajes no requiere pasar por un proceso de compilación para obtener un archivo ejecutable, la manera de compilación de Perl es a través de un intérprete el cual se encarga de interpretar los scripts desarrollados por el programador. Es por esta razón que a Perl se lo conoce como un lenguaje de scripts. Al ser Perl un lenguaje que trabaja en el servidor, está diseñado para manejar la memoria y los procesos de una manera robusta que no implique el uso de recursos que limiten la capacidad del servidor. Perl es ampliamente usado para desarrollar interfaces web dinámicas, a través de CGI's es por esta razón que es mencionado en el presente proyecto como el lenguaje principal para el desarrollo de páginas dinámicas que contienen formulación para obtener información por parte del administrador de la red.

CONSIDERACIONES AL DESARROLLAR UN SCRIPT EN PERL

Perl al igual que otros lenguajes de programación sigue ciertas normas que permiten al programador desarrollar sus scripts de una manera entendible, agregando comentarios y diferenciando las partes que realizan tareas importantes dentro del script. El desarrollo de un script en Perl no requiere de manera

obligatoria implementar estos mecanismos de desarrollo, pero lo que se busca al aplicarlos es que, si una persona intenta entender este script, lo pueda realizar de una manera más fácil, y no pierda su tiempo tratando de interpretar algo que no es entendible es por esta razón que hay que tener en cuenta los siguientes detalles:

- *Comentarios:* La manera más simple para proporcionar información sobre el propósito de un script lo constituyen los comentarios. Es recomendable indicar la función que realiza un script desarrollado así como también las personas que desarrollaron éste. Un comentario al inicio de cada tarea importante dentro de un script permitirá como se menciona facilitar la lectura del flujo de información.
- *Claridad:* Un script desarrollado sin seguir algunas normas se tornará complejo, incluso para la persona que desarrolla el script ya que si algo se modifica y si no se realizaron comentarios se perderá tiempo tratando de identificar el problema, de igual manera si se trabaja con un grupo de personas y no se proporciona un script entendible se perderá el tiempo para seguir con el desarrollo en entender el script escrito, por estas razones que la claridad en el desarrollo es un punto importante.
- *Indentación:* La indentación no es más que separar un bloque de líneas de código hacia la derecha mediante espacios o tabulaciones, con la finalidad de mejorar la legibilidad del script, de esta manera al generar por ejemplo, un bloque if { } se tendrá que realizar una indentación al contenido dentro de este bloque generalmente una tabulación mas en relación a la columna de las llaves.
- *Manejo de variables:* Con la finalidad de proporcionar una legibilidad al script desarrollado, los nombres de las variables deben ser claros, generalmente los contadores y variables simples llevan nombres de pocos caracteres, mientras que las variables que manejan información importante suelen tener un nombre similar a la información que manejan sin exceder el número de caracteres puede ser menor a 10. Es recomendable que cada variable cuente con su respectivo comentario en especial aquellas que son críticas para el desarrollo

del script, cuando se maneja dentro del script contenidos de un archivo estas variables se las escribe con letras mayúsculas permitiendo la diferenciación con el resto de variables.

Al momento de desarrollar un script en Perl a demás de las consideraciones mencionadas en los párrafos anteriores, hay que tener en cuenta que Perl es un lenguaje case-sensitive, es decir diferencia entre mayúsculas y minúsculas por lo que hay que tener presente este aspecto para evitar posibles errores de tipificación. Los comentarios se los inicia con el signo # seguido de la información a documentar finalmente cada línea dentro del script tiene que finalizar con el signo ; si este signo no es ingresado en alguna línea dentro del script el interprete generará errores y no ejecutará el script.

TIPOS DE DATOS

Perl define 3 tipos de datos para el manejo de la información:

- **Escalares**

Un escalar se lo define mediante el símbolo \$ seguido de su nombre de variable de esta manera `$perro` define la variable cuyo nombre es perro, la información que contiene un escalar puede ser un valor entero, un real, una cadena de caracteres o una combinación de estos. Por ejemplo:

```
$a = 3;      $a = "br";    $a = 3.3;    $a = "3br45";
```

Hay que tomar en cuenta que Perl interpreta la información como un escalar independiente de su contenido sea entero, carácter o combinación. Los datos pueden ser almacenados ya sea mediante comillas simples o comillas dobles la diferencia radica si se va o no a ingresar caracteres especiales dentro de la variable. Por ejemplo el caracter, `\n` o salto de línea. De esta manera si se quiere emplear caracteres especiales se empleará comillas dobles `$a = "perl \n";` mientras que para el resto de variables se puede emplear las comillas simples. `$b = 'perl';` al momento de imprimir cada variable en la misma línea la diferencia se aprecia, ya que `$a` cuenta con un salto de línea que provocará que lo que esté a continuación de esta variable sea mostrada en la línea siguiente.

Mediante paréntesis se puede definir un conjunto de escalares con sus respectivos valores, disminuyendo las líneas de código a una sola línea de esta manera: `($a,$b,$c,$d) = ("perro",23,23.5,'perl \n');` define 4 escalares con sus respectivos valores.

- **Arreglo de escalares**

Un arreglo no es más que un tipo de dato que almacena un conjunto de escalares independiente de su contenido, el arreglo se lo define mediante el símbolo `@` seguido de su respectivo nombre. La manera de almacenar la información es mediante un conjunto de escalares encerrados entre paréntesis. Por ejemplo:

```
@dias = ("lunes","martes","miércoles");
```

Se ha almacenado en el arreglo `@dias` los elementos lunes, martes y miércoles. Para acceder a un elemento dentro del arreglo se emplea el símbolo `$` seguido del nombre del arreglo y entre corchetes el valor del índice del elemento que se quiere obtener, hay que tomar en cuenta que el valor del índice inicia en 0. Por ejemplo:

```
$m = $dias[1];
```

Esta línea dará a la variable `$m` el valor martes ya que corresponde al segundo elemento cuyo índice es 1 dentro del arreglo `@dias`. Si se quiere obtener el valor del último índice del vector se empleará los caracteres `$#` seguido del nombre del arreglo, para obtener el valor del último índice del arreglo `dias` se emplearía `$m = $#dias;`. Para obtener varios valores almacenados dentro de un arreglo se emplea el símbolo `@` seguido de su nombre y entre corchetes los índices o el rango de índices. Por ejemplo:

```
@dias[1,2];           @dias[0..2];
```

Para el primer ejemplo se obtendrán los valores cuyos índices sean 1 y 2, mientras que para el segundo caso se obtendrán los valores cuyos índices van desde 0 al 2, para el ejemplo sería todos los elementos del arreglo.

- **Arreglos para asociación de escalares o HASH**

Los hash o arreglos asociados permiten crear relaciones entre dos elementos dentro de un arreglo. Un elemento proporcionará la información mientras el otro funcionará como llave (key) para obtener el elemento. De esta manera dentro de un hash se almacenan pares relacionados. La declaración de un hash se la realiza comenzando por un símbolo % seguido del nombre del hash. La asignación de escalares dentro del arreglo se la realiza ya sea mediante funciones dentro del script o de manera constante. Por ejemplo:

```
%colores = ("amarillo",1, "azul",2, "verde",3);
```

Esta línea crea las relaciones color con su correspondiente valor. Para el caso del color azul su valor correspondiente es 2, se aprecia claramente que la llave constituye el color y proporciona su valor correspondiente. Amarillo → 1, azul → 2, verde → 3. Correspondiendo a las parejas creadas. Para acceder a un elemento dentro del hash se debe emplear el símbolo \$ ya que lo que se va a obtener es un elemento escalar, a continuación se escribe el nombre asignado al hash y por último se abre y cierra llaves conteniendo a la llave correspondiente del elemento relacionado que se espera obtener. Por ejemplo:

```
$a = $colores{"amarillo"}; $b = $colores{"azul"};
```

Las variables \$a, \$b, obtendrán los escalares de los elementos a los cuales hacen referencia las llaves *amarillo* y *azul* dentro del hash. Estos valores son 1 y 2 respectivamente. Las llaves al igual que sus valores correspondientes pueden ir variando mientras transcurre el flujo del script. No es necesaria una declaración inicial para formar un hash constante. Generalmente un hash es empleado para almacenar información de manera dinámica mientras se ejecuta un script.

MANEJO DE ARCHIVOS

Perl es un lenguaje diseñado principalmente para generar información, ya sea de manera local mediante un script o empleando funciones que permitan la obtención de información a través de archivos de texto para proceder a su modificación y luego almacenarla en un nuevo archivo. Para esto Perl define funciones y variables que permiten manipular esta información. El presente proyecto maneja

ampliamente la manipulación de archivos de texto por lo que se enfocará a describir como se realiza este proceso.

- **Apertura de archivos:** La apertura de archivos se la realiza mediante la instrucción OPEN seguida de la información necesaria para la manipulación, su estructura es la siguiente.

open(VARIABLE," ARCHIVO");

- *VARIABLE:* corresponde al nombre de la variable que almacenará la información del archivo abierto.
- *ARCHIVO:* corresponde a la ubicación del archivo dentro del servidor así como también el modo de apertura para el archivo.

Un archivo puede ser abierto en los siguientes modos:

Modo	Explicación
<ARCHIVO	Abre el archivo en modo lectura.
>ARCHIVO	Abre el archivo en modo escritura borrando la información que exista.
>>ARCHIVO	Abre el archivo en modo escritura agregando la información al final.
+<ARCHIVO	Abre el archivo en modo lectura y escritura.
+>ARCHIVO	Abre el archivo en modo lectura y escritura borrando la información que exista.
PROGRAMA	Permite ejecutar "programa" y obtener su salida.
PROGRAMA	Permite ejecutar e ingresar datos para la entrada de "programa".

Tabla A.1 Modos de apertura de archivo.

- **Manipulación de archivos:** la manipulación de un archivo se la realiza almacenando la información dentro de la variable declarada en *open* a manera de reglones, es decir cada línea dentro del archivo es almacenada en la variable. Para esto se emplea:

- *El operador <>*: este operador contendrá la información del archivo, dentro de este operador se ingresará la variable definida en la instrucción open por ejemplo: **<IN>** *IN* corresponde a la variable declarada y ha sido encerrada por el operador.

Perl puede manejar tanto la entrada como salida estándar, para este caso se emplearán tanto STDIN como STDOUT dentro de la manipulación de archivos mediante la consola del sistema operativo. En el presente proyecto la manipulación es realizada mediante la interfaz CGI por lo que no se empleará más que la manipulación de archivos dentro del servidor.

- **Cierre de archivos:** Al igual que el resto de lenguajes existentes. Los archivos en Perl requieren ser cerrados para evitar el consumo de recursos dentro del sistema operativo. La instrucción que permite cerrar un archivo es CLOSE seguida del nombre de la variable declarada en la instrucción open. Por ejemplo:

close variable;

De esta manera un archivo queda cerrado, para este caso variable, esta instrucción permite identificar las acciones que se realizan a un archivo antes que se proceda a su cierre dentro de un script. Un punto importante dentro de la lectura de información contenida en un Archivo de texto es el almacenamiento, hay que tener presente que Perl almacena cada línea del archivo con un salto de línea por lo que al momento de su manipulación hay que tener presente esto para evitar manipulaciones erróneas de la información.

Bloques Condicionales e Iterativos (Bucles)

Perl maneja distintos bloques de acuerdo a como se quiera manipular la información para obtener un resultado en particular, es el caso de los bloques que permiten definir condiciones para cumplir una tarea o bloques para realizar tareas repetitivas hasta cumplir una condición. A continuación se define cada uno de estos bloques que permitirán para el caso del proyecto manipular la información para obtener un resultado esperado.

Bloques condicionales:

- *If (condición) {*

bloque de instrucciones;

}

El bloque if funcionará siempre que la condición entre los paréntesis sea verdadera, si esto no es verdad las instrucciones dentro del bloque nunca se ejecutaran.

- *If (condición) {*

bloque de instrucciones;

}else{

}

El bloque if else a diferencia del anterior puede ejecutar instrucciones cuando no se cumple la condición dentro de los paréntesis en *if ()*. El flujo de información es simple, si se cumple la condición se ejecutará el bloque de instrucciones caso contrario se ejecutará el bloque dentro de else.

- *If (condición) {*

bloque de instrucciones;

}elsif (condición){

bloque de instrucciones;

}

El bloque *if() ...elsif()...* permite agregar dos condicionales, es el caso de *if()* si cumple la condición ejecutará el bloque de instrucciones, caso contrario se verificará el siguiente condicional para decidir si ejecuta o no laguna instrucción.

- *unless (condición) {*

bloque de instrucciones;

}

Unless es un bloque propio de perl y realiza el proceso contrario al bloque if. Mientras que if se traduce como *si se cumple la condición*, unless se traduce como *si no se cumple la condición*. Su flujo es similar a if, para entrar al bloque debe satisfacer la condición.

Bloques iterativos:

- *for (variable,condición,variable++){*

bloque de instrucciones;

}

La instrucción for permite crear un bucle que ejecutará el bloque de instrucciones hasta que cumpla la condición especificada, variable será un escalar con valor entero que irá aumentando su valor en 1 mediante el operador ++, dentro de condición se utilizará variable para compararlo con una variable externa para definir si se cumple o no la condición y finalizar el bucle.

- *while (condición){*

bloque de instrucciones que modifican la condición de while;

}

While es base a su condición ejecutará el bloque de instrucciones con la particularidad que dentro del bloque existen tareas que modifican la condición para que en algún momento la condición pase de verdadera a falsa y se termine el bucle generado.

- *foreach variable (arreglo o (keys hash) o (values hash)){*

bloque de instrucciones que puede emplear variable;

}

Foreach permite tomar un escalar dentro de un conjunto de escalares para poder manipularlo dentro del bloque de instrucciones. Es decir mientras existe un valor en un arreglo o en un hash el cual es tomado por variable, el bloque funcionará.

Operaciones Básicas

Luego de mencionar los bloques principales que se pueden emplear dentro de un script en Perl, queda la duda acerca de cómo se puede generar las distintas condiciones, para ello se definen las conocidas operaciones básicas, estas operaciones se dividen en distintos grupos de acuerdo a la tarea que realizan:

- *Operaciones aritméticas:* Como su nombre lo indica se encargan de realizar operaciones matemáticas sobre un escalar que Perl a definido como un numero. Estas operaciones son:

Operador	Descripción
++	Antes de la variable: primero se incrementa el valor en 1 y luego se evalúa. Después de la variable: primero se evalúa y luego se incrementa en 1.
--	Antes de la variable: primero se decrementa el valor en 1 y luego se evalúa. Después de la variable: primero se evalúa y luego se decrementa en 1.
/	Realiza la división entre dos escalares numéricos.
*	Multiplica dos escalares numéricos.
-	Resta dos escalares numéricos.
+	Suma dos escalares numéricos.
**	Potencia de un escalar numérico.
%	Obtiene el modulo de la división entre dos escalares numéricos.

Tabla A.2 Operaciones aritméticas.

- *Operaciones lógicas*: Las operaciones lógicas permiten dentro de un script Perl implementar los condicionales para que estos puedan ser utilizados dentro de los distintos bloques, al igual que las operaciones aritméticas se define un conjunto de operaciones lógicas entre ellas:

Operador	Números	Cadena
Mayor que	>	Gt
Menor que	<	Lt
Mayor o igual	>=	Ge
Menor o igual	<=	Le
Igual	==	Eq
Diferente	!=	Ne
Comparar	<=>	Cmp
And “y”		
Or “o”	&	&
Negación	!	!

Tabla A.3 Operaciones lógicas.

Las operaciones lógicas se dividen en dos ramas, las destinadas a realizar comparaciones entre escalares numéricos y las encargadas de comparar cadenas de caracteres, su funcionamiento es igual no varía, solo se emplea un operador distinto. Para el caso en el cual se tiene que comparar cadenas de caracteres se toma en cuenta el orden de la cadena. Por ejemplo: con Lt se devuelve verdad si el orden de la cadena de la izquierda es menor que el de la derecha. Con el operador <=> el intérprete devuelve los valores: 0 cuando los valores comparados son iguales, 1 cuando el valor de la derecha es menor que el de la izquierda y -1 para el caso contrario, de la misma manera al comparar con el operados **cmp** el interprete devuelve los valores: 0 cuando las cadenas son iguales, 1 cuando la cadena de la derecha se encuentra al comienzo de la cadena de la izquierda y -1 cuando la cadena se encuentra al final.

- *Operaciones con cadenas y expresiones regulares:* Uno de los potenciales de Perl es el manejo de cadenas y expresiones regulares que permiten la manipulación de información dentro de un script. Por esta razón se han definido algunas operaciones que permiten realizar ciertas tareas sobre cadenas de caracteres para obtener una respuesta específica. Entre las operaciones para cadenas y expresiones regulares se tiene:

Operador	Descripción
.	Concatena dos cadenas de caracteres
X	Multiplica n veces una cadena de caracteres
++	Modifica el valor de una cadena de manera que se incremente sin dejar de considerarse una cadena.
=~	Indica que la cadena de la izquierda pasará por el proceso de verificación en base a una expresión regular.
!~	Realiza el proceso de negación para el proceso de verificación en base a una expresión regular.
//	Constituyen delimitadores de una expresión regular.
' '	Especifica cadena de caracteres sin tomar en cuenta caracteres especiales.
" "	Especifica cadena de caracteres tomando en cuenta caracteres especiales en el caso de que existan
` `	Realiza el proceso de ejecución del comando que se encuentre entre los signos.

Tabla A.4 Operaciones para caracteres y expresiones regulares.

Hay que tener presente como se realizan las operaciones de caracteres dependiendo del operador, para el caso del operador **x** la expresión se repetirá el número de veces que se indique. Por ejemplo: "a" x 10; repetirá 10 veces el carácter a. para el caso del operador **++** al igual que los números realiza un incremento pero en este caso siguiendo el orden del alfabeto. Por ejemplo. "aa"++; dará como resultado "bb" siguiendo el orden mencionado. El resto de

operadores no requiere una explicación adicional que la mencionada en la tabla A.4.

Expresiones Regulares

Las expresiones regulares permiten manipular la información que se obtiene al leer un archivo de texto, esto se lo realiza mediante la evaluación de cada línea de texto dentro del archivo con un patrón, el cual permitirá determinar si existe alguna concordancia. Por ejemplo si se crea el patrón */^amor/*, y se lo evalúa con cada línea de texto se determinará que al encontrar una concordancia se generará una respuesta verdadera la cual puede ser utilizada para realizar alguna tarea. O también para el caso contrario. De la misma manera una expresión puede ser utilizada para realizar procesos de reemplazo de una cadena cuando concuerda con el patrón o modificar caracter por caracter generando un proceso de traducción. Todas estas funciones se las puede realizar con la ayuda de expresiones regulares y ciertos elementos adicionales que permiten agregar funcionalidad al manejo de estas expresiones.

El uso más simple de una expresión regular es la comparación de una cadena con un patrón:

Cadena =~ /patrón/;

Un patrón no es más que una expresión regular construida con la finalidad de encontrar alguna concordancia dentro de varias líneas de texto con el patrón. Hay que tener presente que las expresiones regulares manejan tanto mayúsculas como minúsculas, de esta manera una expresión regular escrita sin tomar en cuenta este aspecto provocará un flujo errado de la información. Las expresiones regulares pueden ser de distintos tipos:

- *Sustitución:* Como su nombre lo indica realiza un proceso de sustitución en base al patrón de búsqueda, si encuentra el patrón en el texto, reemplaza el texto por el indicado en la expresión. A las expresiones de sustitución se les puede añadir ciertas opciones para modificar la manera de realizar este proceso. La sintaxis para realizar una sustitución es la siguiente:

Texto (cadenas de caracteres) =~ s/patrón de búsqueda/texto de sustitución/;

Texto (cadenas de caracteres) =~ s/patrón de búsqueda/texto de sustitución/g;

- s = indica que se realizará un proceso de sustitución.
- Mientras que las opciones se las coloca al final de la expresión regular es el caso de g en el segundo ejemplo. Entre las opciones se tiene:
 - g:** para realizar la sustitución sobre todas las concordancias que se encuentren sobre el texto.
 - i:** para realizar sustituciones sin tomar en cuenta si el texto se encuentran en mayúsculas o minúsculas.
 - e:** para evaluar la parte de la sustitución dentro del script.
- *Traducción:* El proceso de traducción se lo realiza para reemplazar un patrón encontrado dentro del texto por una nueva secuencia de caracteres. Hay que diferencia la traducción con la sustitución, la traducción se la realiza caracter por caracter mientras que la sustitución reemplaza cadenas de caracteres. La sintaxis para realizar una traducción es la siguiente:

Texto (cadenas de caracteres) =~ tr/patrón de búsqueda/cadena de traducción/;

Aunque la traducción y sustitución parecen muy similares son totalmente diferentes, esto se puede apreciar en el siguiente ejemplo:

- a) *Texto (cadenas de caracteres) =~ s/perro/conejo/;*
- b) *Texto (cadenas de caracteres) =~ tr/vista/12345/;*

En el caso a) cada vez que se encuentre la cadena *perro* en el texto esta será reemplazada por la cadena *conejo*.

En el caso b) si se encuentra el caracter v en el texto, éste será traducido al número 1, de la misma manera para el resto de caracteres $i \rightarrow 2$, $s \rightarrow 3$, $t \rightarrow 4$, $a \rightarrow 5$. No es un reemplazo en la cadena completa es un remplazo de caracter.

Las expresiones regulares emplean ciertos caracteres especiales que añaden características adicionales al generar un patrón de búsqueda. Por ejemplo, para determinar que el patrón tenga concordancia con dígitos, con letras del alfabeto,

con tabuladores, con saltos de línea entre otros, a continuación se muestra una tabla con los caracteres especiales disponibles y su función:

Caracter	Descripción
^	Es empleado para buscar una concordancia al inicio de la cadena de búsqueda.
\$	Es empleado para buscar una concordancia al final de la cadena de búsqueda.
()	Es empleado para delimitar un conjunto de patrones de comparación.
[]	Es empleado para delimitar un conjunto de caracteres que pueden encontrarse dentro de la cadena de búsqueda.
{ }	Es utilizado para determinar la longitud de un caracter o cadena.
-	Es utilizado para especificar un rango ya sea números o letras.
.	Especifica que se trata de cualquier carácter.
\d	Representa un dígito.
\w	Representa un carácter alfanumérico.
\t	Representa un tabulador.
\n	Representa un final de línea.
\r	Representa un retorno de carro.
\	Es empleado para poder utilizar caracteres especiales en un patrón. Se lo coloca antes de un caracter especial, por ejemplo: \\d.

Tabla A.5 Caracteres especiales.

Los caracteres especiales permiten reducir la longitud de una expresión regular. Por ejemplo, para el caso de un patrón que quiera encontrar todos los dígitos en una cadena, se tendría que expresar cada dígito de la siguiente manera [1234567890], ahora con la ayuda del carácter especial guion el patrón queda reducido a [0-9], de esta manera se abarca todos los dígitos decimales. Este es un claro ejemplo de cómo los caracteres especiales reducen el desarrollo de una expresión regular.

Las expresiones regulares también cuentan con un conjunto de operadores destinados a realizar una tarea específica dentro de un patrón, que al igual que los caracteres especiales buscan disminuir la longitud y ayudar al desarrollo de un patrón de búsqueda más simple. Hay que tener en cuenta que estos operadores se los utiliza dentro de una expresión regular y son distintos a los mencionados en párrafos anteriores:

Operador	Descripción
*	Se encarga de repetir un patrón de búsqueda 0 o más veces.
+	Se encarga de repetir un patrón de búsqueda 1 o más veces.
?	Se encarga de dar el carácter de opcional a un patrón dentro de una expresión regular.
\$1..9 Por ejemplo: \$1	Es utilizado para referenciar un patrón de búsqueda en una expresión regular, puede ser empleado en una traducción o sustitución.
\1..9 Por ejemplo: \$1	Cumple la misma función que el operador anterior con la diferencia que se lo utiliza en la sección destinada a los procesos de búsqueda.

Tabla A.6 Operadores para patrones en una expresión regular.

Como parte final de este anexo se va a proporcionar un listado de las principales instrucciones dentro de Perl.

Instrucción	Definición
abs	Devuelve el valor absoluto del valor ingresado como parámetro.
chmod	Cambia los permisos del nombre del archivo pasado como argumento.
chop	Elimina y devuelve el último carácter de una cadena de caracteres.
chown	Cambia el propietario del archivo pasado como parámetro.

close	Cierra un archivo abierto por la instrucción open.
delete	Borra el valor de un arreglo asociativo a través de su clave.
die	Imprime en la salida del error estándar un mensaje el cual es pasado como parámetro cuando ocurre un error mientras se está ejecutando el script.
fork	Realiza una llamada fork.
index	Devuelve la posición de la primera ocurrencia de una cadena en otra.
int	Devuelve la parte entera del parámetro.
join	Une las cadenas pasadas con un separador, tanto las cadenas como el separador son dados como argumentos de la instrucción.
keys	Devuelve todas las claves de un arreglo asociativo.
length	Devuelve la longitud en caracteres del parámetro ingresado.
local	Declara como locales las variables indicadas.
mkdir	Crea un directorio en la dirección dada.
open	Abre un archivo cuya dirección y variable que almacenará la información son ingresadas como parámetros.
values	Devuelve todos los valores de un arreglo asociativo.
print	Imprime la información dada como parámetro en la salida correspondiente, para el caso de CGI en una página HTML.
require	Permite el ingreso de código externo dentro del script.
return	Empleado en subrutinas para devolver un valor que puede ser utilizado dentro del script.
sleep	Utilizado para poner en "pausa" la ejecución del script un número de segundos ingresados como parámetro.
sort	Empleado para ordenar un arreglo dado como parámetro.
split	Utilizado para dividir una cadena de caracteres en base a un carácter ingresado como parámetro.
system	Ejecuta la información que es ingresada como parámetro.

Tabla A.7 Instrucciones más empleadas en Perl.

B. ANEXO B

HERRAMIENTAS PARA PROVEER SERVICIOS EN LINUX

El sistema operativo Linux es utilizado principalmente como un sistema operativo para servidores, el cual provee servicios en una red. Estos servicios pueden ser destinados a un sinnúmero de actividades como por ejemplo para seguridad mediante un firewall, o para servicios de administración de red como DHCP, cada servicio dentro del sistema operativo es ejecutado como un daemon³⁷ el cual se encarga de escuchar peticiones que se realicen al servidor para ejecutar una tarea, es el caso de un servidor DHCP que está continuamente escuchando peticiones en la red para proveer una dirección IP a un cliente que lo solicite.

El directorio `/etc/init.d` almacena las configuraciones automáticas de cada servicio para permitir su ejecución, el sistema operativo emplea este directorio para activar los diferentes servicios cuando el sistema operativo arranca, de igual manera desactivará los servicios cuando el sistema se apague. En el caso de la distribución de Linux Mandriva la manipulación de los servicios se la realiza de dos maneras: empleando una interfaz gráfica o mediante línea de comandos.

Si se emplea la interfaz gráfica, la manipulación de servicios es simple ya que solo se debe manipular botones, obviamente la configuración de los servicios se los realiza en el sitio donde se encuentra el servidor. Si se utiliza la línea de comandos la manipulación de servicios ya no es tan simple puesto que se tiene que conocer los distintos comandos que permiten activar o desactivar servicios dentro del sistema operativo. A continuación se describe la manera de manejar los servicios a través de la línea de comandos.

³⁷ **Daemon.**- Constituye un proceso que se ejecuta en segundo plano, no realiza una interacción con el usuario del sistema, generalmente inician al arrancar el sistema operativo y permanece activo hasta que un evento lo reinicie, pare o el sistema operativo se apague.



Figura B.1 Interfaz gráfica para servicios en Mandriva Linux.

- **Inicio de un servicio por línea de comandos:**

Para este caso se emplea el comando *service* seguido del nombre del daemon que ejecuta el servicio (cada daemon se encuentra localizado en */etc/init.d*) mas la palabra *start*, de esta manera el servicio inicia.

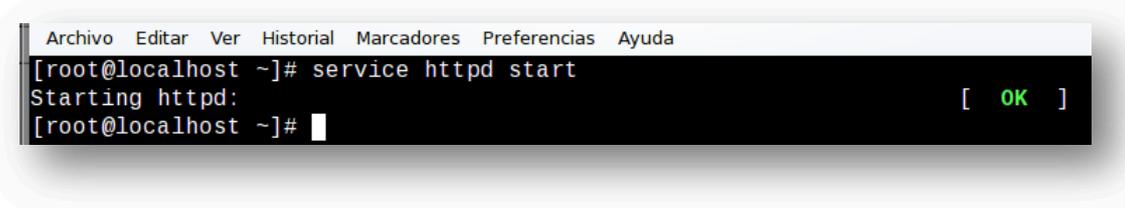


Figura B.2 Inicio del servidor web httpd (apache).

- **Parar un servicio**

Se emplea el comando *service* seguido del nombre del daemon que ejecuta el servicio más la palabra *stop*, de esta manera el servicio se detiene.

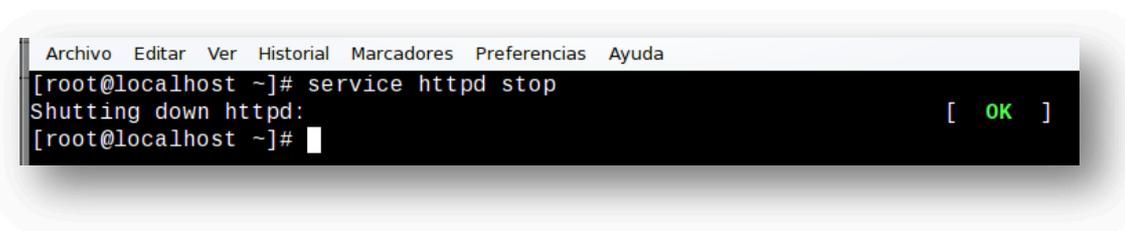
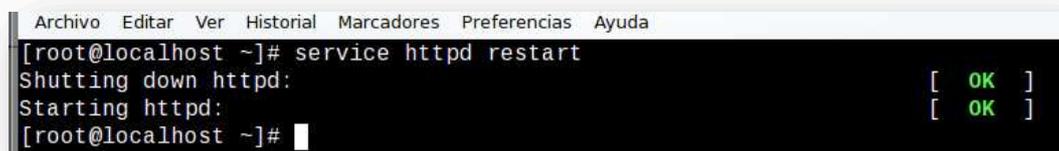


Figura B.3 Servidor httpd (apache) detenido.

- **Reiniciar un servicio**

Para este caso se emplea el comando *service* seguido del nombre del daemon que ejecuta el servicio más la palabra *restart*, de esta manera el servicio reinicia.



```

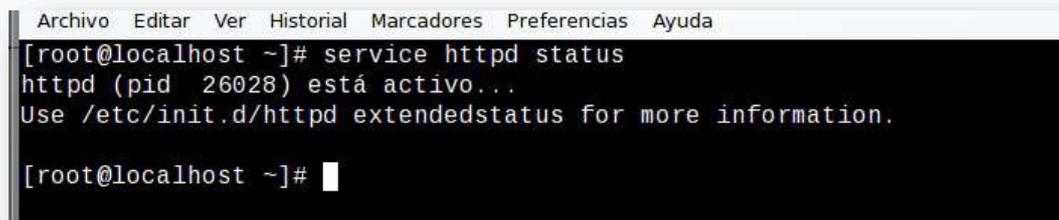
Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# service httpd restart
Shutting down httpd: [ OK ]
Starting httpd: [ OK ]
[root@localhost ~]# █

```

Figura B.4 Reinicio del Servidor web httpd (Apache).

- **Estado de un servicio por línea de comandos**

Se emplea el comando *service* seguido del nombre del daemon que ejecuta el servicio más la palabra *status*.



```

Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# service httpd status
httpd (pid 26028) está activo...
Use /etc/init.d/httpd extendedstatus for more information.

[root@localhost ~]# █

```

Figura B.5 Estado del servidor web httpd (apache).

La administración de servicios por línea de comandos a diferencia de la interfaz visual permite que se pueda utilizar desde algún tipo de aplicación, por ejemplo empleando scripts desde una página web dinámica o desde alguna aplicación desarrollada en un lenguaje de programación. Esto es una gran ayuda para el administrador de la red ya que puede crear sus propias aplicaciones para administrar los servicios de manera remota sin necesidad de estar en el sitio donde se encuentra el servidor.

DHCPD

El servicio dhcp en sistemas Linux toma el nombre de DHCPD y su función es permitir que los clientes obtengan información que les permita formar parte de una red, para ello una estación de trabajo dentro de una red debe contar con un

cliente dhcp que intercambiará información con un servidor dhcp, el servidor da al cliente su dirección IP, puerta de enlace, dirección del servidor DNS, su correspondiente mascara de red y en el caso de que exista un servidor proxy proporcionara su correspondiente dirección. La utilización de un servidor dhcp permitirá al administrador de la red centralizar el control sobre la configuración de red para las estaciones de trabajo. El servidor dhcp puede asignar direcciones IP de dos maneras: la estática que asigna una dirección IP fija a un host y la dinámica que emplea un rango de direcciones IP para asignar a los distintos hosts.

Instalación y Configuración del Servidor DHCP

La instalación del servidor DHCP para la distribución Linux Mandriva se la realiza en pocos pasos, esta distribución cuenta con un gestor de software que permite obtener las herramientas correctas así como sus dependencias, para garantizar el funcionamiento correcto del servicio.



Figura B.6 Gestor de software para Mandriva Linux.

- Inicialización del gestor para la instalación de software:

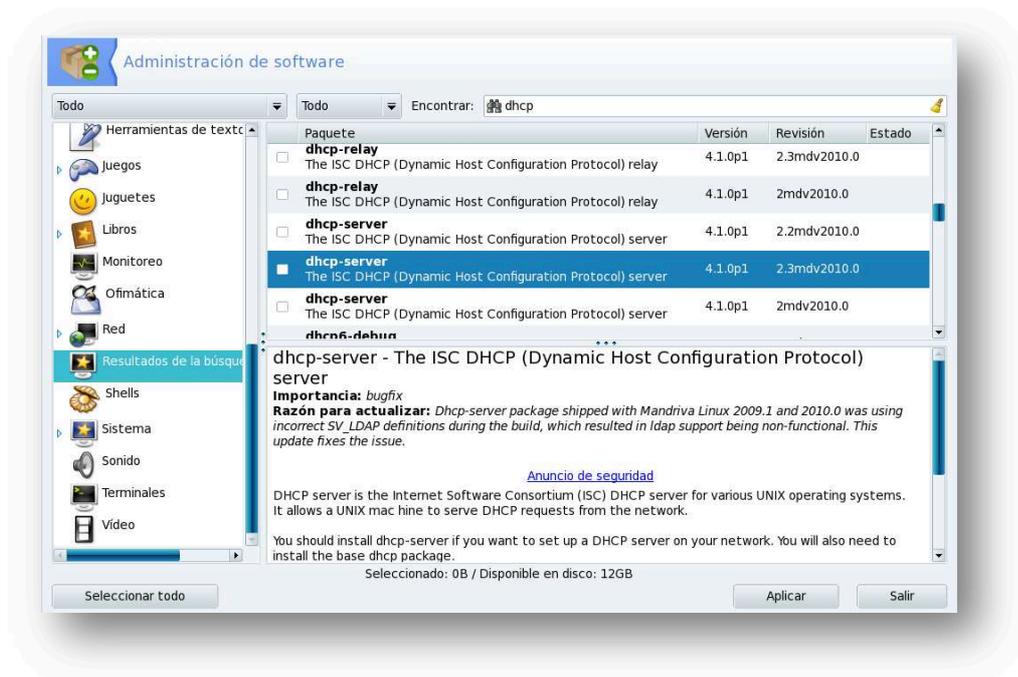


Figura B.7 Búsqueda e instalación del servicio DHCP.

- El gestor proporciona un buscador para ingresar el nombre del servicio o software que se quiera instalar, al ingresar el nombre del servicio en este caso DHCP se indicará si este se encuentra o no instalado mediante un indicativo que varía dependiendo del estado del software. Si no se encuentra instalado se tendrá que seleccionar el software y a continuación el gestor indicará las dependencias que necesita el software, el gestor se encargará de instalar el software y sus dependencias.

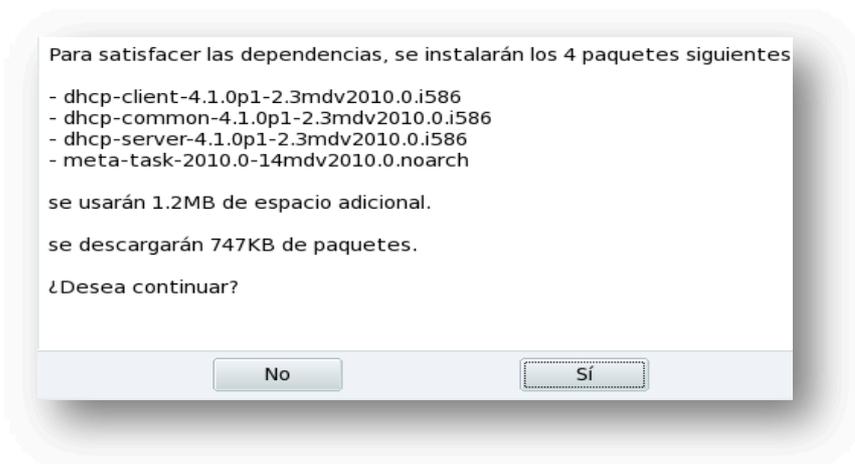


Figura B.8 Dependencias para DHCP.

Las opciones y argumentos del servidor DHCP se especifican en el archivo `/etc/sysconfig/dhcpd` por lo general estos archivos cuentan con información a manera de comentarios donde se indican que opciones y argumentos pueden ser empleados, en este archivo se especificarán las diferentes opciones para el funcionamiento del servidor DHCP, para el presente proyecto se cuenta con dos interfaces de red que cumplirán tareas específicas, una de ellas proveerá el servicio DHCP mientras otra obtendrá el servicio de internet, para ello se necesita especificar que el servicio DHCP solo funcione en la interfaz respectiva, aquella que provee el servicio a la red local, esto se lo realiza en el archivo `dhcpd` mediante la configuración `INTERFACES` seguida del parámetro correspondiente por ejemplo, `INTERFACES="eth1"`.

Para configurar la manera en la que el servidor proveerá el servicio DHCP se emplea el archivo `dhcpd.conf` el cual se encuentra localizado en el directorio `/etc/dhcp.conf`, en este archivo se indicará toda la información que el servidor dará a los clientes cuando requieran conectarse a la red. Esta información está formada por declaraciones, parámetros y opciones, es aplicada de manera global a todos los clientes, si se requiere proveer otro tipo de información a un cliente específico se tiene que declarar dentro del archivo otra configuración de manera que este cliente no empleará la configuración global.

- *Declaraciones:* Las declaraciones se encargan de especificar la información que el servidor DHCP va a emplear para proporcionar el servicio. Por ejemplo la declaración *range* indica el rango de direcciones disponibles para los clientes DHCP.
- *Parámetros:* Indica la información que el servidor empleará para tratar a un cliente. Por ejemplo, *fixed-address* especifica una dirección fija para un cliente.
- *Opciones:* Proporciona al servidor la información necesaria para entregar al cliente cuando este requiera acceder a un servicio de red. Por ejemplo, *option domain-name* especifica el nombre de dominio del cliente.

En la tabla B.1 se puede apreciar las declaraciones, parámetros y opciones para el archivo de configuración `dhcpd.conf`.

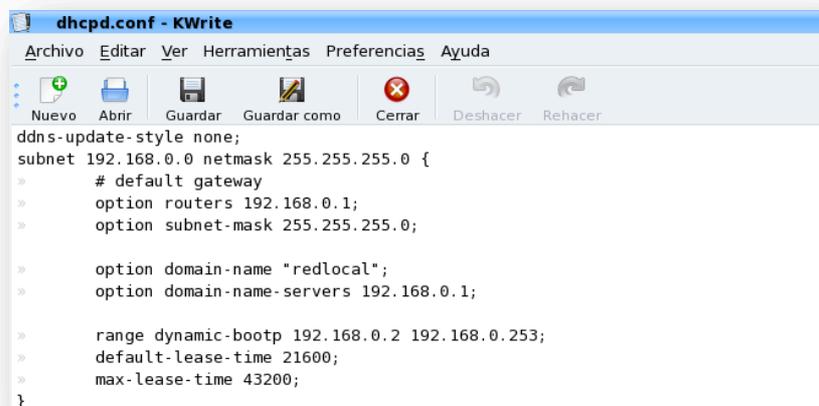
Sintaxis	Descripción
Declaraciones	
shared-network <i>nombre</i>	Indica si algunas subredes comparten la misma red física.
subnet <i>subnet-número-subred mascaradared</i>	Hace referencia a una subred completa de direcciones.
range <i>dirección inicial dirección final;</i>	Proporciona el rango de direcciones.
host <i>nombredehost</i>	Hace referencia a un host particular.
Group	Permite etiquetar un grupo de parámetros, declaraciones para aplicarlas a subredes y hosts.
allow unknown-clients; deny unknown-clients;	No asigna de forma dinámica direcciones a clientes desconocidos.
allow bootp; deny bootp;	Determina si responde a consultas bootp.
allow booting; deny booting;	Determina si responde a consultas de clientes.
Parámetros	
default-lease-time <i>tiempo;</i>	Asigna el tiempo en segundos para el arrendamiento a un cliente.
max-lease-time <i>tiempo;</i>	Asigna el tiempo máximo de arrendamiento a un cliente.
hardware <i>tipo-hardware dirección-hardware;</i>	Especifica el tipo de hardware de red y su dirección.
filename "<i>nombredearchivo</i>";	Especifica el nombre de un archivo de arranque inicial.
server-name "<i>nombre</i>";	Especifica el nombre del servidor en el que está arrancando un cliente.
next-server <i>nombre-servidor;</i>	Especifica el servidor que carga el archivo de arranque inicial determinado por el nombre de archivo.
fixed-address <i>dirección ;</i>	Asigna una dirección fija a un cliente.

get-lease-hostnames <i>marca;</i>	Determina si se buscan y usan direcciones IP de clientes.
authoritative; not authoritative;	Niega solicitudes de direcciones no validas.
server-identifier <i>hostname;</i>	Especifica el servidor.
Opciones	
option subnet-mask <i>dirección-ip;</i>	Especifica una máscara de subred del cliente.
option routers <i>dirección-ip, dirección ip,...;</i>	Especifica una lista de direcciones de IP para enrutador en la subred del cliente.
option domain-name-servers <i>dirección-ip , dirección-ip,...;</i>	Especifica una lista de servidores de nombres de dominio por el cliente.
option log-servers <i>dirección-ip, dirección-ip,..;</i>	Especifica listas de servidores de registro utilizados por el cliente.
option host-name <i>cadena;</i>	Especifica el nombre de host del cliente.
option domain-name <i>cadena;</i>	Indica el nombre de dominio del cliente.
option broadcast-address <i>dirección-ip;</i>	Especifica la dirección de difusión del cliente.
option nis-domain <i>cadena;</i>	Especifica el dominio del servicio de información de red del cliente.
option nis-servers <i>dirección-ip, dirección-ip,..;</i>	Especifica servidores NIS que usa el cliente.
option smtp-server <i>dirección-ip, dirección-ip,..;</i>	Presenta una lista de servidores SMTP utilizados por el cliente.
option pop-server <i>dirección-ip, dirección-ip,..;</i>	Presenta una lista de servidores POP utilizados por el cliente.
option nntp-server <i>dirección-ip, dirección-ip,..;</i>	Presenta una lista de servidores NNTP utilizados por el cliente.
option www-server <i>dirección-ip, dirección-ip,..;</i>	Presenta una lista de servidores web utilizados por el cliente.

Tabla B.1 Declaraciones, parámetros y opciones en DHCP.

Direcciones IP Dinámicas con DHCP

Cuando se tiene dentro de una red clientes que no siempre están conectados a la red o no se quiere asignar manualmente las direcciones IP para que un cliente acceda a la red, se utiliza la asignación dinámica de direcciones IP a través del servidor DHCP. Para ello se tiene que ingresar los parámetros y declaraciones correspondientes para que el servidor realice esta tarea. La declaración *range* permite asignar el conjunto de direcciones IP que el servidor DHCP va a emplear para designar a un cliente, también se debe definir los tiempos de concesión o arriendo de direcciones IP mediante los parámetros *default-lease-time* y *max-lease-time*, aparte de estos parámetros el servidor DHCP requiere saber la red a la cual va a proveer el servicio en el caso de contar con varias subredes se definirá la red asignada para cada una de ellas, Para generar una subred se emplea la declaración *subnet*, dentro de esta declaración se ingresa las declaraciones, parámetros y opciones necesarias, de esta manera la configuración de un servidor DHCP dinámico quedaría de la siguiente manera.



```

dhcpd.conf - KWrite
Archivo Editar Ver Herramientas Preferencias Ayuda
Nuevo Abrir Guardar Guardar como Cerrar Deshacer Rehacer
ddns-update-style none;
subnet 192.168.0.0 netmask 255.255.255.0 {
> # default gateway
> option routers 192.168.0.1;
> option subnet-mask 255.255.255.0;
>
> option domain-name "redlocal";
> option domain-name-servers 192.168.0.1;
>
> range dynamic-bootp 192.168.0.2 192.168.0.253;
> default-lease-time 21600;
> max-lease-time 43200;
}

```

Figura B.9 Configuración del servicio DHCP dinámico.

Direcciones IP estáticas con DHCP

La asignación de direcciones IP fijas por parte del servidor DHCP requiere ciertos parámetros que permitan identificar al cliente que va a obtener siempre la misma dirección IP, para ello se emplea la declaración *host* que permitirá configurar la dirección IP fija asignada a un cliente, dentro de esta declaración se deben ingresar los parámetros y opciones necesarias, se debe especificar la dirección

física del dispositivo de red al que se le va a asignar la dirección estática mediante el parámetro *hardware*, al igual que su respectiva dirección IP mediante el parámetro *fixed-address*.

La manera de identificar la dirección MAC³⁸ de una estación de trabajo varía dependiendo del tipo de sistema operativo que utilice, para el caso de una estación de trabajo con sistema operativo Windows la manera de verificar la dirección Mac es la siguiente: se tiene que ejecutar la consola de comandos cmd, en esta consola se tiene que ingresar el comando *ipconfig /all* de esta manera se desplegará los dispositivos de red disponibles junto con su correspondiente dirección física.

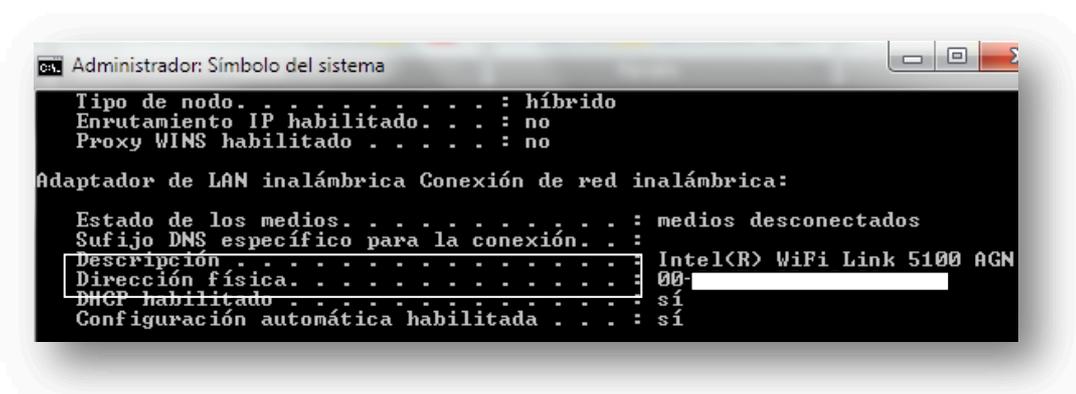


Figura B.10 Determinación de la dirección física en Windows.

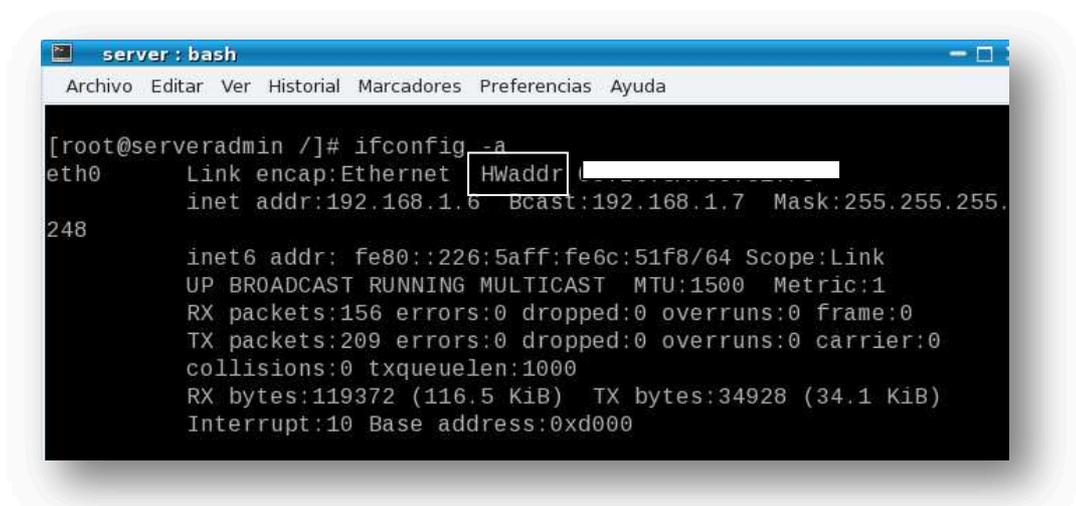


Figura B.11 Determinación de la dirección física en Linux.

³⁸ **Media Access Control (MAC).**- Control de Acceso al Medio, constituye un identificador único formado por 48bits, representados con notación hexadecimal para identificar una interfaz de red, la dirección mac también es conocida como dirección física de red.

Para el caso de una estación de trabajo bajo el sistema operativo Linux se tiene que emplear la consola de comandos en donde se tiene que ingresar el comando *ifconfig -a* al igual que Windows desplegará la información correspondiente a cada interfaz de red.

La configuración básica de asignación de direcciones IP fijas se muestra en la siguiente imagen donde se aprecia la manera de ingresar las declaraciones, parámetros y opciones dentro del archivo de configuración *dhcpd.conf*, cada parámetro, declaración y opción realizará su tarea correspondiente.

```

>         max-lease-time 43200;
}
host daniel {
>         option hist-name "daniel.serveradmin.com"
>         hardware ethernet 00:1e:33:a1:d8:97;
>         fixed-address 192.168.1.12;
>     }

```

Figura B.12 Configuración de una dirección IP estática.

El servidor almacena cada una de la asignaciones IP dentro de un archivo localizado en */var/lib/dhcp* con el nombre *dhcp.leases*, en este archivo se especificarán la dirección IP asignada, los tiempos de inicio y fin de arrendamiento, al igual que el nombre del host respectivo junto con otros parámetros adicionales. Esto permite al administrador de la red saber cómo se están manejando las direcciones IP para los host dentro de la red.

```

starts 5 2010/05/07 19:59:48;
ends 6 2010/05/08 01:59:48;
tstp 6 2010/05/08 01:59:48;
cltt 5 2010/05/07 19:59:48;
binding state free;
hardware ethernet 00:1b:24:9a:e7:70;
}
lease 192.168.0.12 {
starts 4 2010/05/13 22:25:33;
ends 5 2010/05/14 04:25:33;
tstp 5 2010/05/14 04:25:33;
cltt 4 2010/05/13 22:25:33;
binding state free;
}

```

Figura B.13 Arrendamientos de direcciones IP.

SQUID

SQUID es un servidor proxy-caché³⁹ de código abierto que permite proveer a clientes de una red local de servicios disponibles en una red como internet. El servidor proxy maneja las solicitudes que un cliente realiza cuando requiere emplear servicios web, de esta manera actúa como intermediario entre el cliente y la internet, permitiendo realizar procesos que añadan seguridad a la red, así como también logra acelerar el tiempo de acceso a páginas web debido a que mantiene copias actuales de las paginas a las que se acceden con mayor frecuencia, a medida que los clientes acceden a distintas páginas web el servidor almacena en la caché de SQUID una copia de la solicitud, si un cliente solicita nuevamente la página, el servidor verifica si la tiene y la dirige al cliente, si no la tiene obtendrá una copia original del sitio web. La actualización del contenido en caché se lo realiza mediante un algoritmo que permite verificar si una página almacenada y que ha sido solicitada por el cliente se encuentra actualizada, si esta se encuentra actualizada será dirigida al cliente caso contrario se realiza el proceso de actualización de la información.

SQUID cuenta con un soporte para páginas web que emplean ciertos protocolos, por ejemplo, para conexiones seguras mediante SSL. En la tabla B.2 se indican los diferentes protocolos soportados por SQUID:

Protocolo	Descripción y puerto
HTTP	Protocolo empleado por páginas web, su puerto para SQUID es el: 3128.
FTP	Transferencia de archivos mediante páginas web al igual que HTTP emplea el puerto: 3128.
ICP	Protocolo de caché de internet, su puerto es: 3130.
HTCP	Protocolo de cache de hipertexto emplea el puerto: 4827.
CARP	Protocolo de enrutamiento de matrices.

³⁹ **Proxy-caché.-** Constituye un proxy que añade a sus funciones normales la característica de almacenar la información acerca de páginas y contenidos web, para mejorar el tiempo de acceso por parte de los usuarios hacia dicha información mediante el proxy.

SNMP	Protocolo simple de administración de red, emplea el puerto 3401.
SSL	Capa de conector seguro.

Tabla B.2 Protocolos soportados por SQUID.

Instalación del Servicio Proxy-Caché (SQUID)

De la misma manera que el servicio DHCP, se tiene que emplear el gestor de software disponible en Mandriva Linux. Para ello:

- Se ejecuta el gestor de software de Mandriva.
- Se ingresa el nombre del servidor en este caso SQUID para que el gestor busque las distintas versiones disponibles.

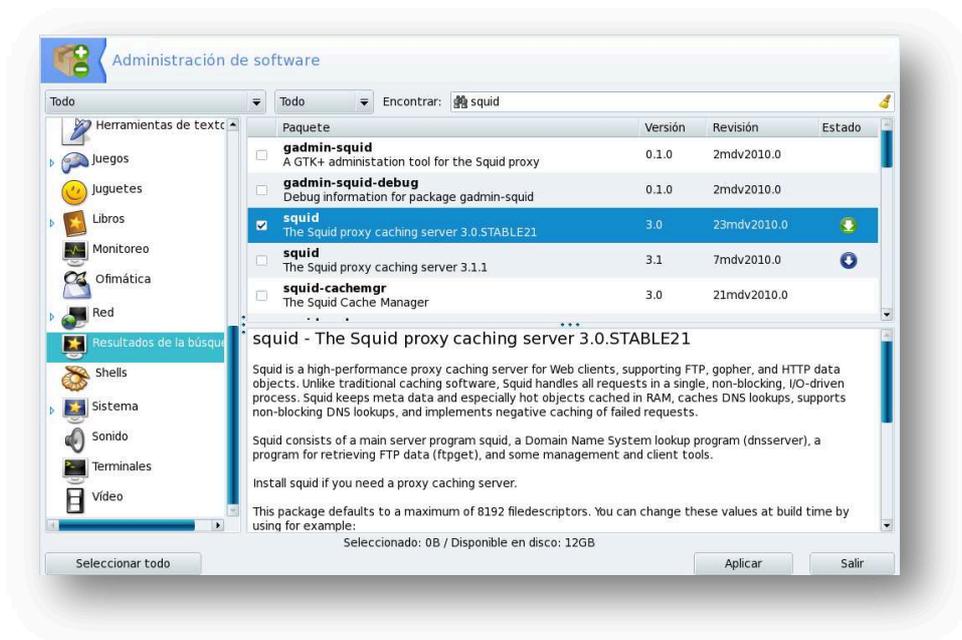


Figura B.14 Squid en el gestor de software Mandriva.

Se seleccionará el servicio y el gestor indicarán las dependencias necesarias de esta manera se procederá a la instalación.

Configuración del Servidor Proxy-Caché SQUID

Como la mayoría de servicios instalados en Linux, SQUID emplea un archivo de configuración denominado *squid.conf* el cual se encuentra ubicado en */etc/squid* y permite configurar todos los parámetros necesarios para poner en ejecución el

servidor proxy-caché en base a los requerimientos necesarios, por ejemplo, los puertos que se van a utilizar, a que usuarios se permitirá el acceso, que sitios web podrán ser visualizados por los clientes así como también definir el tamaño de caché que va a manejar el servicio.

Por lo general el puerto empleado por SQUID es el puerto 3128, este va a permitir que los clientes accedan al servicio Web para obtener información de internet, este valor viene por defecto en el archivo de configuración de SQUID pero puede ser modificado para emplear un puerto distinto mediante la entrada *http_port* en el archivo de configuración. Este archivo cuenta con información a manera de comentarios donde se indican las distintas opciones de configuración que se puede realizar, así como ejemplos que permiten obtener un mejor entendimiento del funcionamiento de cada entrada dentro del archivo.

- **Control de acceso al servidor SQUID**

Al actuar SQUID como un intermediario entre los clientes y el servicio web este permite implementar seguridades para permitir o restringir el acceso a contenido externo, para ello SQUID emplea distintas ACL's (Listas de Control de Acceso) para determinar que clientes pueden acceder al servicio, de esta manera SQUID actúa como un pequeño firewall para el servicio Web.

Para crear listas de control de acceso el archivo de configuración de SQUID cuenta con una entrada denominada **acl**, esta entrada permite crear una lista de acceso ya sea para un grupo de host o un dominio. Luego de haber creado la ACL se debe especificar si dicha lista va a contar con acceso al servicio, para permitir o negar el acceso a los servicios, se debe emplear la entrada **http_access**. Estas dos entradas son básicas y permitirán poner en marcha de manera rápida al servicio.



```

###ACL creadas###

acl hi dstdomain .hi5.com .xat.com .xatech.com
acl mynetwork src 192.168.0.0/255.255.255.0
acl prueba dstdomain .latrilogia.com
no_cache deny prueba
always_direct allow prueba

acl perro dstdomain .facebook.com

acl gato dstdomain .bangbros.com

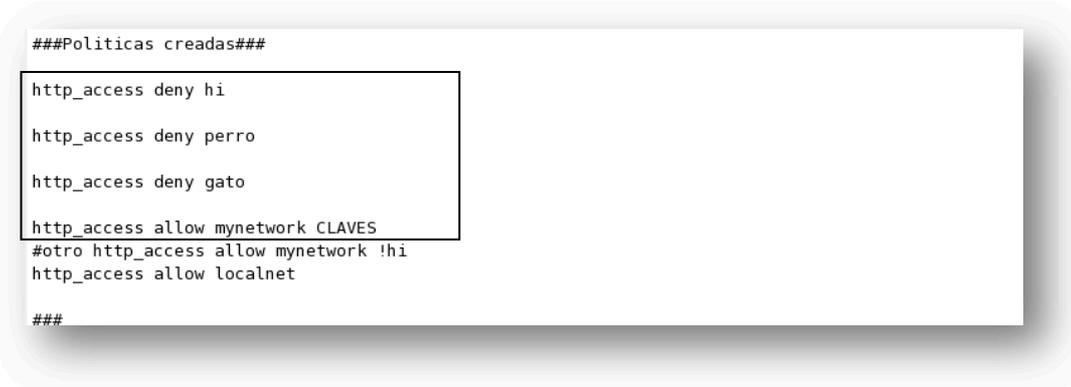
###

```

Figura B.15 Creación de ACL's.

Como se aprecia en la figura anterior la manera de crear una acl es muy simple, se digita la entrada **acl**, a continuación se ingresa un nombre que caracteriza a la lista para identificarla a lo largo del archivo, luego se ingresa la opción denominada **src** seguida del grupo de clientes o cliente al cual se proveerá el servicio junto con su respectiva máscara, de esta manera la lista de control de acceso quedará creada. Ahora se tendrá que determinar las políticas de acceso, ingresándolas en la parte correspondiente dentro del archivo de configuración, cada sección se encuentra claramente diferenciada dentro del archivo de configuración mediante líneas de comentarios.

El orden de agregación de las reglas de acceso o negación es muy importante ya que SQUID ejecuta las ordenes una a continuación de otra, si encuentra una coincidencia a alguna regla SQUID se detiene y no verifica las reglas restantes.



```

###Politiclas creadas###

http_access deny hi

http_access deny perro

http_access deny gato

http_access allow mynetwork CLAVES
#otro http_access allow mynetwork !hi
http_access allow localnet

###

```

Figura B.16 Reglas de acceso.

Para definir que listas tendrán acceso o no al servicio se emplea la entrada **http_access** seguida de la opción **allow** para permitir el acceso al servicio o **deny**

para negarlo, a continuación se ingresa el nombre de la lista de control de acceso creada previamente, de esta manera quedaría definida la política de acceso para una acl. (Observar figura B.16).

SQUID permite también controlar a que sitios externos se acceden, para esto se pueden crear reglas de acceso para dominios ya sean internos o externos, empleando las opciones **dstdomain**, **srcdomain** tanto para origen como para el destino, dentro de la declaración de una **acl**. Hay que tener presente que la regla será aplicada para todo el dominio. Existen más opciones de control de acceso las cuales se mencionan a continuación:

Opción	Descripción
src <i>direcc_ip/mascaradered</i>	Dirección IP cliente.
src <i>direccIP1- direcc-IP2/mascara de red</i>	Rango de direcciones.
dst <i>direcciónIP/mascaradesubred</i>	Dirección IP de destino.
myip <i>direcciónIP/mascaradered</i>	Dirección IP local.
srcdomain <i>dominio</i>	Búsqueda por dominio.
dstdomain <i>dominio</i>	Servidor de destino.
srcdom_regex [-i] <i>expresión</i>	Expresión regular que busca coincidencias con el nombre del cliente.
dstdom_regex [-i] <i>expresión</i>	Expresión regular que busca coincidencias con el destino.
url_regex [-i] <i>expresión</i>	Expresión regular que busca coincidencias con el URI completo.
urlpath_regex [-i] <i>expresión</i>	Expresión regular que busca coincidencias con la ruta URI.
port <i>puertos</i>	Un puerto o rango de puertos.
proto <i>protocolos</i>	Protocolo específico como HTTP, FTP.
method <i>método</i>	Métodos específicos como GET o POST.
browser [-i] <i>regexp</i>	Coincidencia de patrón en encabezados.
ident <i>username</i>	Coincidencia de cadena en la salida ident.
src_as <i>número</i>	Utilizada para enrutamiento de solicitudes

	a cachés específicas.
dst_as número	Utilizada para enrutamiento de solicitudes a cachés específicas.
proxy_auth nombredeusuario	Lista de nombres de usuario válidos.
snmp_community cadena	Una cadena para limitar el acceso a su agente SNMP.

Tabla B.3 Opciones disponibles en una ACL.

SQUID permite también generar un proceso de autenticación para la utilización del Proxy-caché, para ello emplea al igual que en casos anteriores comandos como `ident`, `acl`, `http_access`. Para permitir o negar el acceso al servicio.

- **Manejo de caché (memoria y disco)**

Para configurar el tamaño de disco y memoria destinados a la caché de SQUID se emplea el comando `cache_mem` el cual define el tamaño de espacio asignado para los distintos objetos que maneja el proxy como: objetos que están en transmisión, aquellos que se usan frecuentemente, y objetos de solicitudes fallidas. Por defecto el tamaño de la caché es de 8MB. Esta configuración se la modifica de ser el caso en el archivo `squid.conf`.

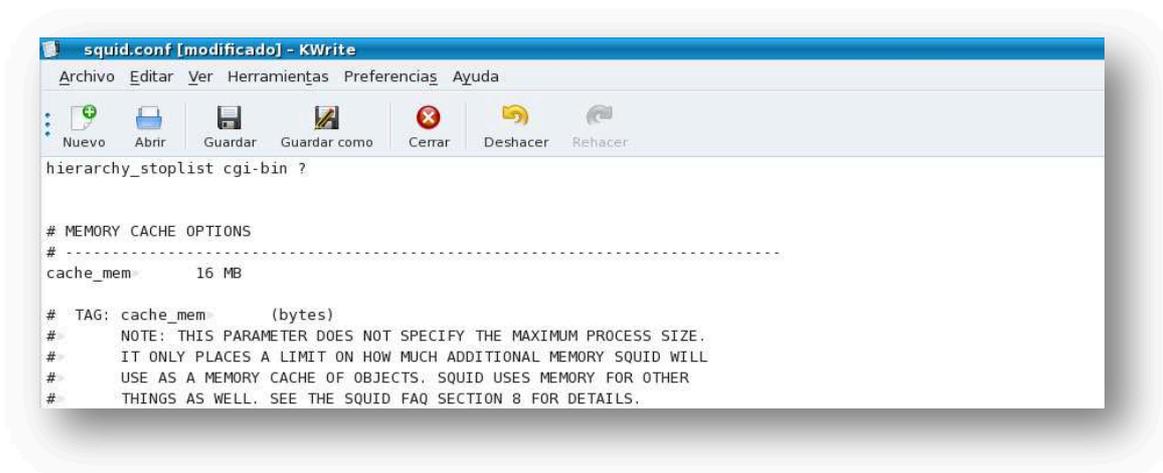


Figura B.17 Asignación del tamaño para caché en SQUID.

También se puede definir un rango de asignación de espacio mediante los comandos: `maximum_object_size` y `minimum_object_size` en el caso del disco. Para el espacio asignado en la memoria se emplean los comandos

maximum_object_size_in_memory y minimum_object_size_in_memory, por defecto el valor establecido para minimum es 0. Squid Permite configurar varias directivas que potencializan el uso del proxy-caché para emplearlo en grandes redes, para el caso del presente proyecto las directivas planteadas se enfocan a la implementación de un servidor proxy-caché básico sin dejar de lado las características de seguridad mencionadas.

```

**
#Default:
# minimum_object_size 0 KB

# TAG: maximum_object_size (bytes)
# Objects larger than this size will NOT be saved on disk. The
# value is specified in kilobytes, and the default is 4MB. If
# you wish to get a high BYTES hit ratio, you should probably
# increase this (one 32 MB object hit counts for 3200 10KB
# hits). If you wish to increase speed more than you want to
# save bandwidth you should leave this low.
#
# NOTE: if using the LFUDA replacement policy you should increase
# this value to maximize the byte hit rate improvement of LFUDA!
# See replacement_policy below for a discussion of this policy.
#
#Default:
# maximum object size 4096 KB

```

Figura B.18 Tamaños máximo y mínimo de disco.

SHOREWALL (Firewall)

Al momento de implementar un conjunto de servicios dentro de un servidor, se requiere aplicar ciertas reglas para manejar los paquetes de datos que ingresan o salen del servidor por las distintas interfaces de red. Para aplicar estas reglas en un servidor Linux, se cuenta con iptables, iptables no es más que una herramienta empleada por el sistema operativo para generar distintas políticas de acceso sobre los paquetes de datos que ingresan o salen a través de las distintas interfaces presentes en el sistema. Iptables puede ser configurado ya sea empleando una línea de comandos o a través de una herramienta de configuración. Cada distribución Linux puede contar con su respectiva herramienta de configuración, para el caso de la distribución Mandriva Linux empleada en el presente proyecto se dispone de la herramienta Shorewall. *Shorewall* (*shoreline firewall* o *cortafuegos de orilla*) no es más que una herramienta de configuración para iptables, la cual emplea un conjunto de archivos de texto simple donde se ingresarán las distintas reglas y

configuraciones necesarias para poner en marcha el firewall. La manera de ingresar las distintas reglas es muy intuitiva, ya que es similar a la forma de configuración de servicios a través de archivos de configuración. Shorewall disminuye en gran parte la complejidad de configurar iptables a través de una línea de comandos, facilitando al usuario el entendimiento de las reglas, dejando a la herramienta realizar los procesos de generación de reglas, shorewall obtiene la información de las líneas ingresadas en los archivos de texto y genera las cadenas⁴⁰ a interpretar por iptables.

Archivos de configuración

Por defecto los archivos de configuración para Mandriva Linux se encuentran ubicados en el directorio */etc/shorewall/* cada una de las políticas de acceso que se quiera aplicar a los paquetes de datos en el sistema deben ser ingresados dentro del archivo respectivo en este directorio. Los principales archivos de configuración son:

- Shorewall.conf
- Policy
- Interfaces
- Rules
- Masq
- Zones

• **Archivo */etc/shorewall/shorewall.conf***

De manera similar al manejo de servicios en un servidor Linux, shorewall cuenta con su correspondiente archivo de configuración, en este archivo se ingresan todos los parámetros que afectarán de manera global a la herramienta. Los parámetros que se encuentran por defecto en el archivo bastarán para que shorewall funcione sin problema.

⁴⁰ **Cadena.**- No es más que una tarea que el kernel del sistema operativo Linux empleará para realizar el proceso de administración de los paquetes que recibe o quiere enviar por las interfaces de red.

Para verificar que shorewall inicie al momento de encender el servidor se tiene que verificar la línea dentro del archivo de configuración denominada *STARTUP_ENABLED* este parámetro acepta dos valores *no* o *yes*, al ingresar el valor *yes* se indicará que shorewall siempre inicie al momento de arrancar el sistema. Este parámetro por defecto en Mandriva Linux ya se encuentra con el valor *yes*. Por esta razón solo se verificará que este parámetro se encuentre con el valor respectivo.

- **Archivo */etc/shorewall/zones***

En este archivo se definen las distintas zonas que van a ser administradas por la herramienta shorewall, cada zona es identificada por un nombre. Por ejemplo, *fw* identifica al firewall y es la zona por defecto para referirse a la maquina donde se está ejecutando shorewall. Un nombre de zona puede estar formado por letras, digitos o “_”, con un máximo de 5 caracteres. Existen ciertas palabras que son reservadas para el uso de shorewall: “all”, “none”, “any”, “SOURCE” y “DEST”. El orden en el cual se ingresan las zonas será empleado por shorewall para relacionar los distintos paquetes de datos con su respectiva zona. En la figura B.19 se muestra el archivo *zones* con las respectivas zonas creadas.

```
#
# Shorewall version 4 - Zones File
#
# For information about this file, type "man shorewall-zones"
#
# The manpage is also online at
# http://www.shorewall.net/manpages/shorewall-zones.html
#
#####
#####
#ZONE TYPE          OPTIONS          IN              OUT
#                   OPTIONS          OPTIONS          OPTIONS
net  ipv4
loc  ipv4
fw   firewall
#LAST LINE -- DO NOT REMOVE
```

Figura B.19 Archivo *zones*.

Se pueden distinguir varias columnas, cada una de ellas destinadas a una declaración específica:

La primera columna corresponde a la declaración de la zona con su nombre respectivo. Por ejemplo *fw* identifica a la zona del firewall, es decir la maquina donde se encuentra ejecutando shorewall. La zona *net* corresponde a la conexión a internet.

La segunda columna corresponde al tipo de zona, para el firewall el tipo de zona es *firewall*, solo puede existir una zona firewall. Otro tipo de zona y la mas empleada es *ipv4* corresponde a la zona estándar, si se deja esta columna vacía el valor por defecto corresponderá a este tipo. Estos dos tipos son los más empleados, adicionalmente se cuenta con 3 tipos de zonas: *ipsec*, *bport*, *vserver* que se emplean para definir tipos de zonas especiales.

Las siguientes columnas denominadas *options* solo podrán ser utilizadas cuando una zona sea declarada con un tipo *ipsec*. Para mayor información acerca de las distintas opciones se puede consultar la página oficial de shorewall en la sección *zones*⁴¹.

- ***Archivo /etc/shorewall/interfaces***

En este archivo se ingresan las distintas interfaces de red presentes en el firewall relacionando cada interfaz con su correspondiente zona, el orden en el cual se ingresan las interfaces no tiene importancia al momento de aplicar las distintas reglas de acceso. La figura B.20 muestra el archivo *interfaces*, como se puede observar, la distribución de la información en el archivo es similar al archivo *zones*.

⁴¹ Shorewall: **Shorewall-Zonas**. Tomado de: <http://www.shorewall.net/manpages/shorewall-zones.html>

```

# Shorewall version 4 - Interfaces File
#
# For information about entries in this file, type "man
shorewall-interfaces"
#
# The manpage is also online at
# http://www.shorewall.net/manpages/shorewall-interfaces.html
#
#####
#####
#ZONE INTERFACE BROADCAST OPTIONS
net wlan0 detect
net eth0 detect
loc pan0 detect routeback
#LAST LINE -- DO NOT REMOVE

```

Figura B.20 Archivo *interfaces*.

La primera columna identifica a la zona y corresponde a los nombres de zona ingresados en el archivo *zones*.

La segunda columna identifica a la interfaz de red que se relacionará con la zona. Por ejemplo: eth0, wlan0.

La tercera columna *broadcast* es empleada para ingresar la dirección de broadcast para la interfaz, shorewall cuenta con una opción denominada *detect* que permite detectar la dirección broadcast automáticamente.

La cuarta columna corresponde a *options* en esta se declaran las distintas opciones que se pueden aplicar sobre la interfaz, la opción importante y que servirá para el funcionamiento de los distintos servicios en el presente proyecto es *dhcp*, se empleará esta opción debido a que:

- Se estará ejecutando un servidor dhcp en la misma máquina que shorewall. y,
- Una de las interfaces obtendrá información de red de un servidor dhcp.

Existe un conjunto amplio de opciones adicionales aplicables a una interfaz las cuales pueden ser consultadas en la página oficial de shorewall en la sección *interfaces*⁴².

⁴² Shorewall: **Shorewall-Interfaces**.

Tomado de: Shore<http://www.shorewall.net/manpages/shorewall-interfaces.html>

- **Archivo */etc/shorewall/policy***

En este archivo se define las políticas de acceso “de nivel alto” a ser aplicadas para las conexiones establecidas entre las zonas definidas en el archivo *zones*. El orden de ingreso de las políticas es importante ya que según su ingreso se aplicarán las distintas reglas. Cuando se inicia una conexión entre zonas primero se verifica el archivo *rules* (se detallará más adelante) si no se encuentra una relación entre las zonas dentro de *rules*, se verifica la relación dentro del archivo *policy*, por cada par de zonas que quieren establecer una conexión se verifica dentro del archivo en orden, desde la primera hasta la última regla hasta encontrar una relación. En la figura B.21 se muestra el archivo *policy* con ciertas declaraciones.

```
# Shorewall version 4 - Policy File
#
# For information about entries in this file, type "man
shorewall-policy"
#
# The manpage is also online at
# http://www.shorewall.net/manpages/shorewall-policy.html
#
#####
#####
#SOURCE      DEST POLICY          LOG  LIMIT:
      CONNLIMIT:
#
#           LEVEL BURST      MASK
loc  net  ACCEPT
loc  fw   ACCEPT
fw   loc  ACCEPT
fw   net  ACCEPT
net  all  DROP info
all  all  REJECT  info
#LAST LINE -- DO NOT REMOVE
```

Figura B.21 Archivo *policy*.

La primera columna corresponde a la zona origen, cualquiera de las zonas definidas en el archivo *zones*, permitiendo también emplear “*all*” para referirse a todas las zonas.

La segunda columna se refiere a la zona destino, de igual manera se emplean las zonas definidas en *zones*, y “*all*” para referirse a todas las zonas.

La tercera columna se refiere a la política que se aplicará para la relación entre las zonas. Esta política será aplicada si no se encuentra una relación dentro del archivo denominado *rules*.

Destino	Función
ACCEPT	Permite el ingreso del paquete a través del firewall.
DROP	Ignora la solicitud de conexión.
REJECT	Niega el acceso y avisa al emisor.
QUEUE	Envía los paquetes al espacio del usuario.

Tabla B.4 Algunas políticas dentro del archivo *policy*.

La cuarta columna determina el manejo sobre los logs generados al momento de manejar las solicitudes de conexión entre las distintas zonas. Si se quiere almacenar información acerca de las distintas conexiones ignoradas o negadas se debe ingresar la palabra *info* en esta columna, de manera que toda la información acerca de las distintas conexiones sea almacenada en el archivo de registro del sistema (log) correspondiente.

La quinta columna se encarga de especificar el número de conexiones TCP permitidas en un intervalo de tiempo y el tamaño aceptable de burst. Por ejemplo: 4/sec:5, indica que se aceptan 4 conexiones por segundo con un tamaño de burst = 5.

- **Archivo */etc/shorewall/rules***

En este archivo se configuran las reglas que rigen sobre todos los paquetes que fluyen a través del firewall, es el archivo más importante, pues aquí se determinarán todas las reglas sobre los paquetes. Si se ha definido dentro del archivo destinado a políticas (*policy*) que ninguna conexión sea permitida entre las zonas, se tendrá que indicar en el archivo *rules* cada regla para permitir el establecimiento de una conexión para habilitar distintos servicios. Mandriva Linux configura las reglas en un archivo distinto a *rules*. Este archivo se encuentra ubicado en */etc/shorewall/rules.drakx*, es allí donde se ingresarán todas las reglas. De manera similar al archivo *policy* las reglas deben ser ingresadas en el orden a ser aplicadas ya que el firewall busca una relación y si la encuentra finaliza la búsqueda, por esta razón la organización de las reglas dentro del archivo es primordial para evitar fallos en el proceso de manejo de conexiones. En la figura B.22 se muestra tanto el archivo *rules* así como el archivo *rules.drakx*.

```

# Shorewall version 4 - Rules File
#
# For information on the settings in this file, type "man
shorewall-rules"
#
# The manpage is also online at
# http://www.shorewall.net/manpages/shorewall-rules.html
#
#####
#####
#####
#ACTION          SOURCE          DEST          PROTO DEST  SOURCE
                ORIGINAL    RATE          USER/ MARK  CONNLIMIT  TIME
#                DEST        LIMIT        GROUP          PORT  PORT(S)
#SECTION ESTABLISHED
#SECTION RELATED
INCLUDE    rules.drakx
#LAST LINE -- DO NOT REMOVE

```

```

ACCEPT fw net tcp http
REDIRECT loc 3128 tcp http

```

Figura B.22 Archivos *rules* y *rules.drakx*

El archivo *rules* se divide en tres secciones destinadas a realizar tareas específicas.

- La sección *ESTABLISHED*: se encarga de generar reglas para paquetes con estado established. Solo se permiten las siguientes acciones: ACCEPT, DROP, REJECT, LOG y QUEUE.
- La sección *RELATED*: esta sección genera reglas para los paquetes cuyo estado es related. Solo permite las acciones: ACCEPT, DROP, REJECT, LOG y QUEUE.
- La sección *NEW*: genera reglas para los paquetes cuyo estado es NEW, INVALID y UNTRACKED.

Si las reglas son aplicadas fuera de estas secciones shorewall asume que las reglas se encuentran en la sección *NEW*. Generalmente y para el caso del presente proyecto las reglas a aplicar serán de este tipo, es decir fuera de las secciones.

En la tabla B.5 se describen las columnas más importantes dentro del archivo *rules*.

ACTION	SOURCE	DEST	PROTO
Acción a realizar sobre la solicitud de conexión.	Fuente a la cual se aplicará la regla, pueden ser zonas o direcciones IP.	Empleado con DNAT y REDIRECT para especificar el destino, pueden ser zonas, direcciones IP , puertos	Especifica un tipo de protocolo, se puede ingresar el número o nombre de protocolo.

DEST PORT	SOURCE PORT(S)	ORIGINAL DEST	RATE LIMIT
Especifica el puerto destino. Puede ser un número de puerto, nombre o rango.	Puerto usado por el cliente.	Empleado con DNAT y REDIRECT. Para permitir o negar la acción a la(s) dirección(nes) IP ingresada(s) en esta columna.	Número de conexiones TCP permitidas en un intervalo de tiempo.

Tabla B.5 Principales columnas dentro del archivo *rules*.

La columna *action* cuenta con un conjunto de acciones, cada acción debe ser ingresada dependiendo de la tarea que se quiera realizar sobre una solicitud de conexión, luego de haber cumplido con una las reglas planteadas. Entre las principales acciones se tiene:

- *ACCEPT*: acepta la solicitud de conexión.
- *DROP*: ignora la solicitud
- *REJECT*: rechaza la solicitud informando al emisor.
- *DNAT*: (Destination Network Address Translation o Traducción de dirección de red de destino) Permite acceder desde una red externa a distintos puertos dentro de una red privada.

- *REDIRECT*: redirige las solicitudes de un puerto a otro en particular, aquellos puertos que se utilizan por distintos servicios los cuales se ejecutan en la maquina donde se encuentra el firewall.

Estas acciones son las más utilizadas dentro de shorewall en lo referente a creación de reglas. Existen ciertas acciones adicionales que permiten agregar mayor potencialidad al manejo de reglas, su funcionamiento es más complejo y está enfocado a usuarios con un conocimiento avanzado acerca del manejo de políticas de acceso.

- **Archivo */etc/shorewall/masq***

El archivo *masq* permite realizar procesos de enmascaramiento o NAT (Network Address Translation - Traducción de Dirección de Red), el enmascaramiento es empleado para proveer acceso a internet, a un conjunto de direcciones IP de una red privada, a través de una o más direcciones IP públicas, este procedimiento es muy utilizado en la mayoría de empresas donde se cuenta con limitadas direcciones IP publicas y un gran número de direcciones IP privadas. En la B.23 se observa el archivo *masq*, al igual que el resto de archivos se encuentra formado por un conjunto de columnas.

```
#
# Shorewall version 4 - Masq file
#
# For information about entries in this file, type "man
shorewall-masq"
#
# The manpage is also online at
# http://www.shorewall.net/manpages/shorewall-masq.html
#
#####
#####
#INTERFACE      SOURCE          ADDRESS          PROTO PORT(S)
eth2 192.168.0.0/24
                                     |IPSECMARK USER/
#
      GROUP
#LAST LINE -- DO NOT REMOVE
```

Figura B.23 Archivo *masq*.

La primera columna define la interfaz de red que tiene acceso a una dirección IP pública es decir que obtiene acceso a internet. Se puede definir ciertas direcciones IP. Por ejemplo, eth0:(dirección IP/máscara) o si se tiene una sola dirección IP solo se especificará la interfaz de red.

La segunda columna especifica quien va a solicitar el acceso a internet, es decir quién tendrá acceso a internet, puede constituir una interfaz de red, una o más direcciones IP, con la ayuda del signo ! se puede excluir ciertas IPs. Por ejemplo eth0:!192.168.0.1.

La tercera columna es opcional y define la(s) dirección(nes) IP a través de las cuales se puede obtener acceso a internet a través de la interfaz de red ingresada, para realizar proceso de SNAT.

La quinta y sexta columna son utilizadas para definir un protocolo y un puerto. La selección de un puerto puede ser ingresada solo si se ha definido el tipo de protocolo a utilizar. Al ingresar un tipo de protocolo se estará restringiendo el acceso a un tipo de conexiones en particular. Se puede ingresar ya sea el nombre o el número de puerto, de la misma manera para el protocolo. Existen columnas adicionales que realizan tareas distintas pero no son explicadas debido a que no serán utilizadas en el presente proyecto.

De esta manera se ha explicado de forma compacta el proceso para modificar un firewall a través de una herramienta como shorewall. La forma como se ha desarrollado este tema corresponde a los pasos que se tienen que seguir para configurar un firewall a través de shorewall.

Ejemplos acerca de la creación de reglas en shorewall

```
#ACTION SOURCE                DEST                PROTO  DEST
#                                PORT
ACCEPT  loc1                    local2             tcp    22
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

En este ejemplo se aceptan las conexiones que provienen de la zona denominada loc1 hacia el puerto 22 de la zona destino denominada local2.

```
#ACTION SOURCE          DEST          PROTO  DEST
#                               PORT
ACCEPT loc1:192.168.0.2    loc2:192.168.2.1    tcp    22
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

En este ejemplo se aceptan las conexiones que tengan como origen la dirección IP 192.168.0.2 dentro de la zona loc1. Para acceder al puerto 22 de la dirección destino 192.168.2.1 a través del protocolo TCP.

```
#ACTION SOURCE  DEST          PROTO  DEST  SOURCE  ORIGINAL  RATE
#                               PORT   PORT(S) DEST   LIMIT

DNAT  net      loc:192.168.1.2 tcp    53    -      -        10/sec:5
DNAT  net      loc:192.168.1.2 udp    53    -      -        10/sec:5
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

En este caso se realiza un proceso de DNAT para las direcciones externas (internet) que desean ingresar hacia un host de la red privada, cuyo destino es el puerto 53(DNS) a través del protocolo TCP y UDP. Para el caso de la primera regla se define la tasa de conexión a 10 conexiones por segundo, con un tamaño de burst = 5

```
#ACTION SOURCE  DEST  PROTO  DEST  SOURCE  ORIGINAL  RATE
#                               PORT   PORT(S) DEST   LIMIT

REDIRECT      loc   8080   tcp    80     -      -      -
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

Esta regla redirige las conexiones desde la red local en el puerto 80 hacia el puerto 8080 del firewall, donde puede existir un proxy o un servicio de filtrado web, para habilitar por medio de esta regla su uso de manera transparente.

SERVIDOR WEB APACHE

El principal servidor HTTP dentro de las distintas distribuciones Linux constituye el servidor web Apache, apache es un sistema muy poderoso y estable, configurable como la mayoría de servicios mediante sus respectivos archivos. Cuenta también con la posibilidad de ofrecer métodos que permitan agregar seguridades sobre el servicio web, mediante el manejo de módulos que trabajan en conjunto con apache para realizar tareas específicas. Apache es un servidor gratuito desarrollado y mantenido por Apache Server Project. Apache se basó en un servidor web desarrollado por el National Center for Supercomputing Applications,

de la universidad de Illinois. Desde entonces se ha vuelto el servidor web más utilizado dentro de internet con soporte para varias plataformas como Unix, Linux, Windows, de igual manera se han desarrollado varias herramientas de configuración así como documentación que permite realizar una configuración rápida del servicio, aunque el servicio ya está disponible con solo instalarlo.

Instalación del Servidor Web Apache en Mandriva Linux

La distribución Mandriva Linux empleada para el presente proyecto no tiene instalado el servidor apache por defecto por lo que se tiene que emplear al igual que en los servicios anteriores el gestor de software disponible en Mandriva:

- Dentro del gestor de software en la sección de búsqueda se ingresa el texto Apache, y se mostrarán las distintas versiones disponibles, se seleccionará una de ellas y el gestor mostrará las distintas dependencias necesarias para el funcionamiento de apache y se procederá a la instalación.

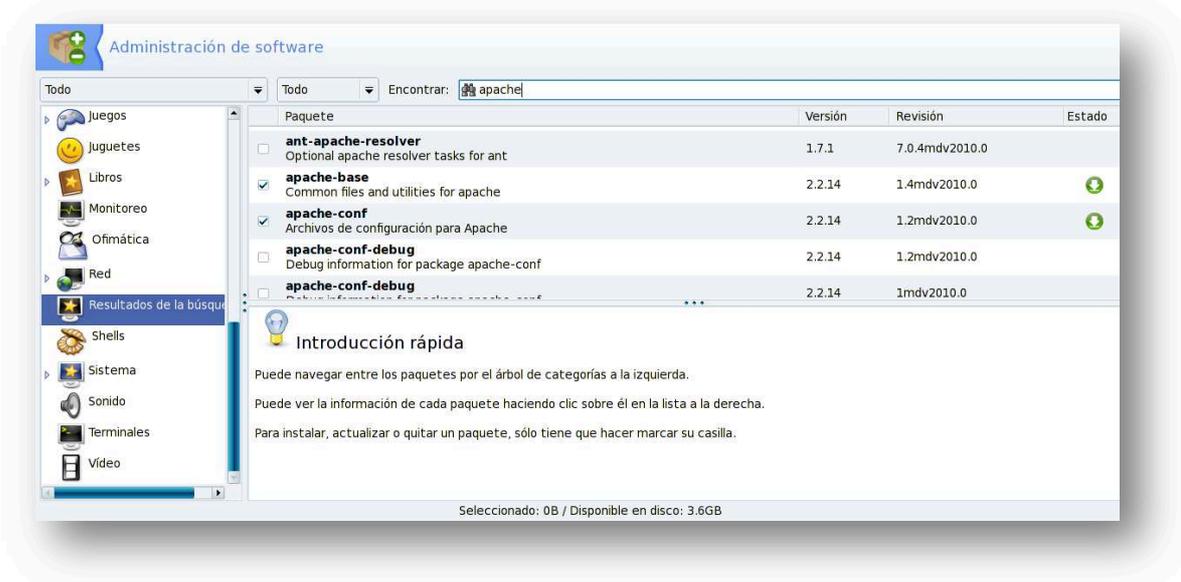


Figura B.24 Instalación del servidor web apache en Mandriva Linux.

Luego de haber instalado el servicio se generarán los distintos directorios donde se almacenarán los archivos de configuración, después de esto el servidor web quedará listo para funcionar con las configuraciones por defecto. Para verificar el estado del servicio se digita `service httpd status` en una consola de comandos, si el servicio está parado se lo inicia empleando el método explicado en la parte referente a inicio de servicios.

Para verificar el funcionamiento del servicio de manera local, hay que ingresar la URI ***http://localhost*** en un explorador web en el servidor, de esta manera se generará el documento HTML por defecto el cual indicará que el servidor web se encuentra ejecutando.

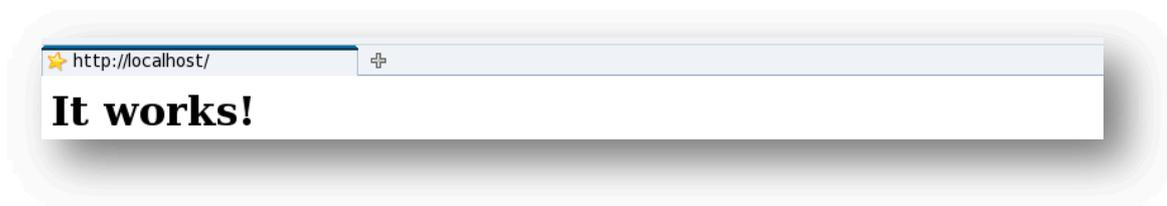


Figura B.25 Verificación del funcionamiento del servidor web.

La tabla B.6 muestra los distintos archivos y directorios que se crean al instalar apache sobre Mandriva Linux, junto con su correspondiente descripción.

Directorios para HTML	Descripción
/var/www/html	Almacena documentos HTML para una página web.
/var/www/cgi-bin	Almacena documentos CGI para páginas web dinámicas.
/var/www/html/manual	Manual de referencia (requiere instalación).
Archivos de configuración	
/etc/httpd/conf	Directorio de archivos de configuración del servidor apache.
.htaccess	Archivo de configuración basados en directorio; este archivo almacena las directivas para controlar el acceso a archivos dentro del directorio que declara.
/etc/httpd/conf/httpd.conf	Archivo de configuración del servidor apache.
/etc/httpd/conf.d	Almacena la configuración de los distintos módulos por ejemplo, ssl.conf para SSL.
/etc/httpd/modules	Directorio que almacena los módulos de apache.

Tabla B.6 Directorios y archivos para configuración del servicio web Apache.

Configuración de Apache

La configuración de Apache se la realiza en base a directivas que se aplicarán dentro del archivo de configuración del servicio conocido como *httpd.conf* como la mayoría de archivos de configuración este archivo contiene a manera de comentarios la explicación para cada una de las directivas que se pueden ingresar para configurar el servicio. Se puede configurar cosas tan básicas como definir el puerto por el cual el servidor web va a escuchar peticiones hasta configuraciones más complejas como generar hosts virtuales. Como la mayoría de servicios implementados en un servidor requiere el desarrollo de políticas de seguridad por ejemplo, emplea SSL para cifrar la información que se transmite entre un cliente y el servidor web, cada una de estas funciones se las implementa mediante su directiva correspondiente, una directiva básica tiene la forma *directiva opción opción...*, existen directivas más complejas que permiten crear bloques de instrucciones que afectarán por ejemplo a un directorio que contiene información que manipula el servidor. Por ejemplo la directiva *Directory* permite crear un bloque de directivas que afectarán a un directorio particular. Las directivas de bloque cuentan con una directiva de inicio y una de cierre, por ejemplo:

```
<Directory pathdelacarpeta>  
    Directiva opción  
    Directiva opción  
</Directory>
```

Esta directiva de bloque permite asignar distintas directivas a la carpeta que se encuentre en *pathdelacarpeta*, cuando el servidor web maneja varios directorios para almacenar contenido HTML, resulta conveniente almacenar cada directiva fuera del archivo de configuración de apache, para ello se emplea los archivos *.htaccess* donde se almacenará las directivas aplicadas a cada directorio que manipulará el servidor. A estos archivos se los denomina de configuración basados en directorio y permiten adicional a la disminución del código dentro del archivo de configuración agregar políticas de monitorización con la finalidad de determinar cómo los usuarios están manejando sus directorios dentro del servidor web cuando se les ha permitido tener control sobre estos.

El archivo de configuración de apache se halla definido en tres secciones: la primera trata acerca de las configuraciones globales, la segunda trata la configuración del servidor y la sección final el manejo de hosts virtuales.

- **Configuración global**

La primera sección del archivo *httpd.conf* corresponde a la configuración global, dentro de esta sección se realizarán modificaciones para controlar distintas operaciones en el servidor, entre estas se definirá los directorios donde se almacenará la información que genere el servidor, así como también se emplearán algunas directivas para controlar el rendimiento del mismo.

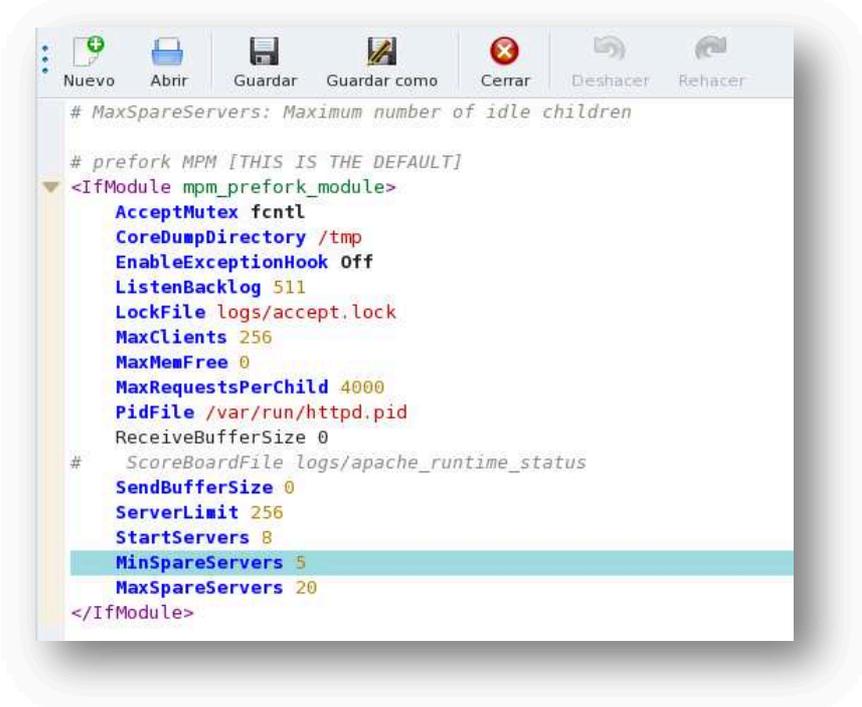
La directiva que permite definir el lugar donde se almacenarán los archivos

de configuración, de error y registro es **ServerRoot**, *ServerRoot /etc/httpd*. Este directorio es empleado como raíz cuando el servidor requiere emplear algún modulo o como punto de partida para otras directivas. Cuando se quiere emplear un puerto distinto al que apache configura por defecto (80), se emplea la directiva **Listen** seguido del puerto donde el servidor web escuchará las peticiones.

Apache permite extender sus capacidades mediante la implementación de módulos, algunos de estos módulos vienen definidos por defecto dentro del archivo de configuración, mientras otros pueden ser creados y almacenados dentro del directorio correspondiente. Un modulo está formado por un conjunto de directivas y se podrán utilizar luego que el servidor haya iniciado su servicio. Un ejemplo de modulo es *mod_mime*, este modulo contiene los diferentes tipos MIME así como las correspondientes directivas. Para cargar un modulo se emplea la directiva **LoadModule** con los argumentos *nombre de modulo* y *ubicación*. Como se mencionó en párrafos anteriores el directorio donde se almacenan los modulos es */etc/httpd/modules*, al cargar un modulo no es necesario indicar la ruta completa ya que se ha indicado mediante **ServerRoot** que el directorio raíz corresponde a */etc/httpd*.

Apache utiliza una arquitectura con módulos de multiprocesamiento conocidos como MPM (MultiProcessing Modules), estos módulos permiten a apache trabajar sobre distintos sistemas operativos así como también controlar ciertas

operaciones de multiprocesamiento, para el caso de sistemas Unix y Linux se definen los MPM prefork o worker, mientras que Windows emplea el mpm_winnt. El modulo prefork es utilizado para mantener la compatibilidad con sistemas antiguos y permite generar un subproceso por proceso. Mientras worker soporta varios subprocesos por proceso generando una carga mucho menor sobre los recursos disponibles en el sistema. El manejo de MPM permite configurar al servidor de acuerdo a las demandas de cargas de trabajo, por defecto cada una de las directivas viene con valores previos, específicos para operar con una carga ligera. Existe un conjunto de etiquetas para manejar MPM, para el caso del presente proyecto la modificación de estos parámetros no es necesaria ya que son suficientes los parámetros por defecto para cada etiqueta pues la carga que se va a manejar es mínima.



```

# MaxSpareServers: Maximum number of idle children

# prefork MPM [THIS IS THE DEFAULT]
<IfModule mpm_prefork_module>
    AcceptMutex fcntl
    CoreDumpDirectory /tmp
    EnableExceptionHook Off
    ListenBacklog 511
    LockFile logs/accept.lock
    MaxClients 256
    MaxMemFree 0
    MaxRequestsPerChild 4000
    PidFile /var/run/httpd.pid
    ReceiveBufferSize 0
    # ScoreBoardFile logs/apache_runtime_status
    SendBufferSize 0
    ServerLimit 256
    StartServers 8
    MinSpareServers 5
    MaxSpareServers 20
</IfModule>

```

Figura B.26 Modulo MPM prefork.

- **Configuración de Servidor**

La segunda sección dentro del archivo de configuración corresponde a *Server configuration*, esta sección permite realizar configuraciones para el servidor web principal, aquel que responde a la solicitudes que no son manejadas por un *<Virtualhost>*, al igual que la sección anterior cuenta con un conjunto de directivas para distintas operaciones, por ejemplo, **DocumentRoot** “/var/www/html” es una

directiva que permite determinar el directorio de inicio para el sitio web alojado en el servidor. La ruta empleada en esta directiva corresponde a la instalada por defecto en el sistema operativo Linux.

En esta sección se definen también los tiempos que empleará el servidor para manejar peticiones o conexiones, mediante las directivas:

Directiva	Definición
Timeout <i>segundos</i>	Es el tiempo (segundos) que espera el servidor antes de terminar la recepción o envío de una solicitud. Por defecto su valor es 300.
KeepAlive <i>on off</i>	Acepta conexiones persistentes en la misma conexión por defecto su valor es on.
MaxKeepAliveRequests <i>valor</i>	Máximo número de solicitudes a permitir durante una conexión persistente. Su valor por defecto es 100.
KeepAliveTimeout <i>segundos</i>	Tiempo de espera para recibir una nueva solicitud del mismo cliente en la misma conexión. Por defectos es 5

Tabla B.7 Directivas para manejo de tiempos.

La directiva más importante dentro de esta sección es aquella que nos permiten manipular directorios individuales para administrar su contenido mediante un control de acceso, así como también la creación de alias para modificar las rutas de acceso URI.

Para la creación de directorios individuales donde se puede almacenar documentos HTML, imágenes, CGI's, se emplea la directiva de bloque **<Directory>**, que encerrará un conjunto de directivas que se aplicarán al directorio especificado. Su sintaxis es:

<Directory rutadeldirectorio>

Directiva.....

Directiva.....

</Directory>

Dentro de **<Directory>** se emplean directivas de control de acceso así como directivas **Options** para habilitar ciertas características por ejemplo, la ejecución de interfaces CGI. Las directivas aplicadas a un directorio de nivel alto son heredadas por un directorio de nivel bajo. Cuando se quiere evitar este tipo de herencia se emplea la directiva **AllowOverride** con el argumento *all* de esta manera cada directorio que ha sido almacenado de manera particular con *.htaccess* puede emplear sus directivas y no heredarlas de la directiva *Directory*. Las directivas que permiten realizar un control de acceso sobre un directorio creado son:

- **allow**.- permite el acceso para los usuarios remotos o hosts especificados en la directiva.
- **deny**.- niega el acceso a los elementos especificados.
- **order**.- indica el orden en que se aplicarán las directivas de acceso.
- **require**.- empleada para control de autenticación.



Figura B.27 Manejo de directivas para el directorio cgi-bin.

Como se puede observar cada directiva cuenta con sus argumentos respectivos por ejemplo, **allow from all** permite el acceso a todos los hosts.

Cuando se quiere modificar la ruta de acceso a un documento HTML por ejemplo, si se encuentra en el directorio `/home/proyecto` y se quiere que en el explorador web se ingrese la URI `http://www.misitio.com/proyecto` y se acceda al contenido de proyecto se emplea la directiva **Alias**, esta directiva permite crear una relación entre la URI que se ingresa en el explorador y el directorio en donde se encuentra almacenada la información HTML. De esta manera la directiva quedaría de la siguiente manera:

Alias /proyecto /home/proyecto

Por defecto apache tiene un conjunto de páginas web de inicio denominadas **index** que son declaradas con la directiva **DirectoryIndex**. Algunas páginas web de inicio estándar son:

DirectoryIndex index.html index.xml index.cgi

Un alias permite relacionar únicamente directorios con contenido HTML, para el caso en el cual se requiera relacionar programas CGI's, se debe emplear otra directiva, ya que estos programas ejecutan script's y están almacenados en directorios especiales con los permisos requeridos para su funcionamiento, para el caso de Mandriva Linux el directorio correspondiente es `/var/www/cgi-bin`. Para relacionar este tipo de directorio se emplea la directiva:

ScriptAlias /cgi-bin/ /var/www/cgi-bin/

Un servidor web también debe ser capaz de permitir implementar políticas de autenticación que restrinja el acceso a usuarios específicos, para el caso de Apache se define un conjunto de directivas de autenticación que van a ser aplicadas a un bloque de directorio `<Directory>` o a un archivo `.htaccess` del directorio. Como se mencionó en párrafos anteriores la directiva **require** determina los usuarios que tendrán acceso al directorio. Entre las directivas de autenticación se tiene:

- **AuthName**.- proporciona el nombre del campo de autorización en el cual los nombres de usuarios y los passwords son válidos.
- **AuthType**.- indica el tipo de autenticación.

- AuthUserfile.- indica la ruta donde se encontrará el directorio de passwords, a manera de nombre de usuario y contraseña.
- AuthUsergroup.- indica la ruta al directorio que contiene los nombres de los miembros del grupo que tendrán acceso.

La tercera sección dentro del archivo de configuración corresponde a la configuración de host virtuales, el presente proyecto no emplea esta sección por lo que no es descrito, solo es mencionado como parte del archivo de configuración de apache *httpd.conf*.

Servidor Web Seguro SSL

La seguridad en un servidor web abarca dos aspectos. El primero referente a la limitación en cuanto al acceso al servidor web y el segundo en cuanto al flujo de transacciones seguras. Para el primer caso ya se ha mencionado la utilización de la herramienta disponible en Linux denominada SQUID. Mientras que para asegurar una transacción entre el explorador web del cliente y el servidor se deben emplear herramientas adicionales que permitan realizar tres tareas fundamentales: identificar los participantes de la transacción, verificar la integridad de los datos, y tener privacidad en la transmisión. Para identificar los participantes de la transacción se emplea procedimientos de autenticación, para verificar la integridad de los datos se emplean firmas digitales conteniendo un valor de digest⁴³ solo para los datos mientras que para la privacidad de los datos se realizan procedimientos de cifrado de información.

La implementación de esta política de seguridad en un servidor web apache se la realiza utilizando el protocolo de capa de sockets seguros conocida como SSL, para ello los datos se cifran con una llave pública, pero solo pueden ser descifradas con su correspondiente llave privada. El proceso de autenticación se lleva a cabo mediante la utilización de certificados digitales.

⁴³ **Digest.-** Es una función matemática que genera una secuencia de caracteres, de entre 128 y 256 bits de longitud, en base a un archivo de información de cualquier longitud.

Tanto el navegador web como el servidor cuentan con su correspondiente certificado. En este punto actúa una tercera parte que no es más que una autoridad certificadora que se emplea como una entidad independiente en la que tanto el explorador como el servidor confían y determinará que el cliente y el navegador son quien dicen ser.

Luego de pasar por el procedimiento de autenticación se determinará el tipo de cifrado para la información para garantizar la privacidad y también se determina el tipo de revisión de integridad digest para asegurar la integridad en los datos. Todos estos procedimientos son realizados en el servidor web Apache mediante la implementación de un modulo denominado `mod_ssl` que emplea la versión de fuente abierta de SSL llamada OpenSSL para realizar transacciones seguras. Este modulo agregar al servidor apache características SSL generando un acceso alternativo al servidor mediante un puerto diferente (**443**) y su correspondiente protocolo HTTPS. De esta manera para ingresar a un sitio web seguro se debe digitar la URI correspondiente empezando por el protocolo seguro `https://www.miseguro.com` como todos los módulos creados para apache `mod_ssl` cuenta con un conjunto de directivas para realizar distintas tareas, el archivo de configuración donde se aplicarán las directivas es **smb.conf** el ingreso de las directivas se las realiza mediante un bloque `ifDefine` habilitados por la marca `HAVE_SSL`. Para descargar el módulo SSL se tiene que declarar la siguiente directiva.

```
<ifDefine Have_SSL>  
  
    LoadModule ssl_module modules/libssl.so  
  
</ifDefine>
```

Al igual que el resto de archivos de configuración **smb.conf** cuenta con comentarios donde se explica la tarea que realiza cada directiva. Si se quiere utilizar una de estas directivas basta con quitar el signo `#` al principio de la directiva y esta quedará lista para ser empleada luego de reiniciar el servicio. Algunas directivas que llaman la atención dentro del archivo de configuración son aquellas que determinan donde se encuentran almacenados las diferentes claves para cifrado así como los certificados del servidor por ejemplo:

```

<ifDefine Have_SSL>

    #Server Certificate:

    SSLCertificateFile etc/httpd/conf/ssl.crt/cerver.crt

</ifDefine>

```

ssl.crt es el directorio donde se almacenan los certificados del servidor de la misma manera *ssl.key* almacenará las llaves pública y privada del servidor.

SAMBA

Samba es un software que fue desarrollado por Andrew Tridgell para permitir compartir recursos dentro de una red heterogénea donde existen maquinas con distintos sistemas operativos como GNU/Linux, Unix, Windows. Para ello samba emplea el protocolo SMB/CIFS (Server Message Block/Common Internet File System) que es compatible con los sistemas operativos mencionados, los recursos a compartir constituyen directorios e impresoras, con samba se puede crear ya sea un servidor SMB para que los usuarios accedan al servicio o se puede implementar el servidor SMB como cliente para acceder a recurso dentro de la red. De esta manera se puede compartir recursos desde una maquina Linux a maquinas Windows o viceversa, de la misma manera se realiza la compartición de impresoras. Actualmente samba es mantenido por *the SAMBA Team*. Bajo la Licencia Pública General GNU.

Instalación y Configuración de Samba

El proceso de instalación de samba es similar al resto de servicios que ya sean mencionado dentro del presente capitulo. Hay que tomar en cuenta que se deben instalar algunos paquetes para que el sistema GNU/Linux pueda trabajar como servidor de recursos o como cliente para ello:

- Se debe inicializar el gestor de software, se ingresará dentro de la sección de búsqueda la palabra samba.
- Hay que seleccionar los paquetes necesarios que permitirán el funcionamiento correcto del servicio para esto hay que seleccionar los

paquetes samba-server (no es más que el servidor Samba), samba-client (cliente para SMB), samba-common (directorios necesarios para el cliente y servidor).



Figura B.28 Servidor samba en el gestor de software de Mandriva.

- **Configuración de samba**

Para el presente proyecto samba va a ser configurado como un servidor que permitirá compartir directorios dentro de este, por lo que en su configuración se mencionará solo los parámetros necesarios para implementar este servicio. Hay que tener presente que samba puede funcionar de manera más compleja como un controlador primario de dominio dentro de una red.

El primer paso a configurar para compartir directorios con samba es la creación de usuarios, todos los usuarios van a contar con un nombre de usuario y contraseña que les permitirá ingresar a sus directorios compartidos. Los usuarios solo tendrán acceso a sus directorios mas no tendrán acceso al intérprete de comandos, por esta razón no se generará claves con *passwd* solo se empleará como intérprete */sbin/nologin* o bien */bin/false*. De esta manera el ingreso de un usuario se lo realizará de la siguiente manera:

```
useradd -s /sbin/nologin usuario
```

De esta manera se ha creado el usuario dentro del sistema GNU/Linux ahora el paso siguiente es crear el usuario en samba para ello se ejecuta:

smbpasswd -a usuario

Para sincronizar las cuentas entre los clientes Windows y el servidor samba es importante que los nombres de usuario y claves sean las mismas. Las claves de acceso serán solicitadas al ejecutar las cadenas mencionadas.

El archivo de configuración que permite compartir directorios en la red se encuentra en */etc/samba* con el nombre *smb.conf* en este archivo se podrá configurar los distintos parámetros para poner en marcha el servicio proporcionado por samba, entre los distintos parámetros se tiene:

workgroup = Nombredegrupo

Este parámetro permite agregar un grupo de trabajo en el cual se va a compartir los recursos. Para asignar un nombre al servidor y este pueda ser visualizado en la red se emplea el siguiente parámetro seguido del nombre correspondiente:

netbios name = servidorlocal

Otro parámetro que nos permite describir al servidor constituye *server string = servidor...* el cual contendrá una breve descripción del servidor.

Luego de este conjunto de parámetros que configuran ciertas características del servidor, se tiene que indicar a que usuarios se va a permitir el acceso, así como también se debe determinar la interfaz sobre la cual se va a permitir recibir solicitudes para el servicio. Con estos parámetros podremos añadir seguridades determinando quienes tienen acceso y evitando que el servidor pueda obtener solicitudes por interfaces que no son permitidas. Los parámetros que permiten añadir estas características son:

hosts allow = 192.168.0. 127.

Hosts allow permite definir que usuarios tendrán acceso al servicio, para el caso de un rango de direcciones se tiene que especificar la dirección IP de la red con un punto al final, en el ejemplo anterior, 192.168.0. Indica que los hosts permitidos van desde la dirección 192.168.0.1 hasta la dirección 192.168.0.254, el valor 127. Indica que se permite el acceso a la interfaz "loopback". Loopback es una interfaz

virtual que representa al propio host independiente de las direcciones IP asignadas a sus interfaces de red reales si existieran.

Para definir sobre que interfaz se escucha las peticiones se emplea el parámetro *interfaces* seguido de la correspondiente dirección IP/Máscara. Por ejemplo:

Interfaces =192.168.0.1/24

Hasta aquí se han definido tanto los parámetros del servicio, las políticas de acceso y la creación de usuarios para el servidor. Ahora es el momento de dirigirse a la parte en el archivo de configuración destinada a la creación de directorios denominada *Share Definitions*, todas las secciones dentro del archivo de configuración cuentan con información de ayuda a manera de comentarios iniciando con el signo #. La definición de un directorio compartido tiene la siguiente estructura:

```
[nombre_descriptivo]
    comment = descripción del directorio a compartir
    path = ruta del directorio
    opciones..... (Aplicables al recurso a compartir)
```

La estructura es muy simple y permite agregar un conjunto de opciones que añadirán características al recurso compartido. La sección *Share Definitions* tiene un conjunto de ejemplos para esta estructura, empleando diferentes opciones. Estos ejemplos se encuentran comentados con un signo ; inicial. En esta sección también se encuentra por defecto ingresadas las estructuras para compartir los directorios home de cada usuario que ha sido creado dentro del servidor, sin olvidar que solo el usuario tendrá acceso al directorio. La tabla B.8 muestra las principales opciones que se pueden ingresar dentro de la estructura para la definición de un directorio compartido.

Opción	Descripción
guest ok = <i>yes</i> o <i>no</i>	Permite acceso al recurso como invitado.
Public <i>yes</i> o <i>no</i>	Similar a la opción guest ok no requiere una clave para conectarse al recurso compartido.
Browseable = <i>yes</i> o <i>no</i>	El recurso aparece en la lista de recursos compartidos en el entorno de red de Windows.

valid users = <i>nombre1 , nombre2, @grupo</i>	Indica los usuarios que pueden acceder al recurso compartido, pueden ser usuarios o grupos.
Writable = <i>yes o no</i>	Determina si el recurso permite la escritura.
admin users = <i>nombre1 , nombre2, @grupo</i>	Define los usuarios que tendrán acceso con permisos de super-usuario sobre el recurso. Pueden ser usuarios o grupos.
write list = <i>nombre1 , nombre2, @grupo</i>	Determina los usuarios que tendrán permiso de escritura sobre el recurso compartido.
directory mask = <i>numero</i>	Define los permisos en el sistema que tendrán los directorios creados dentro del recurso. Por defecto: 0755 no permite la escritura para un grupo y otros.
create mask = <i>numero</i>	Define los permisos en el sistema que tendrán los archivos creados dentro del recurso. Por defecto: 0744 no permite la escritura y ejecución para el grupo y otros.
inherit permissions = <i>yes o no</i>	Al crear archivos o directorios estos heredan los permisos del directorio donde son creados.

Tabla B.8 Directivas para manejo de tiempos.

A continuación se muestra un ejemplo para crear un directorio compartido empleando las opciones mencionadas:

[Descargas]

```
comment = Almacena los archivos de Descarga.
path = /home/Descargas
writable = no
write list = daniel
valid users = daniel , root
browseable = yes
```

Se ha creado el recurso compartido Descargas que se encuentra en la ruta /home/Descargas puede ser examinada por los usuarios daniel, root, pero en modo de lectura con excepción del usuario daniel que tiene permisos de escritura.

De esta manera se ha mostrado como se realiza la configuración de un servidor samba básico que comparte recursos dentro de una red heterogénea. A continuación se proporciona tanto la configuración en el lado del servidor como la correspondiente visualización en el cliente.

```
[Descargas]
comment = Almacena los archivos de Descarga
path = /home/Descargas
writable = no
write list = daniel
valid users = daniel
browseable = yes
```



Figura B.29 Configuración y visualización de un recurso compartido.

DANSGUARDIAN FILTRADO DE CONTENIDOS

Una herramienta indispensable dentro de un servidor, constituye aquella que permite administrar el contenido web que los clientes pueden visualizar. GNU/Linux cuenta con una herramienta para realizar esta tarea, su nombre es DANSGUARDIAN y no es más que un software que permite realizar un filtrado de contenidos web, manejando políticas para determinar que páginas y que contenido dentro de ellas pueden ser visualizados por los clientes. Dansguardian al igual que la mayoría de herramientas Linux es código abierto (Open Source), está desarrollada en C++ y permite su manipulación en base a los archivos de configuración de manera flexible, adaptada a los requerimientos que un administrador de red necesite al momento de implementar políticas de acceso web.

Dansguardian trabaja en conjunto con un servidor proxy caché como es el caso de la herramienta SQUID aunque también puede trabajar con otro tipo de proxy. La herramienta dansguardian actúa entre el cliente y el servidor proxy-caché, de esta manera cuando un cliente realiza una solicitud al servidor dansguardian se encarga de recibir la solicitud y determinar si se permite o no el acceso a la información solicitada, en el caso de aceptar la solicitud esta es enviada hacia el servidor proxy el cual se encarga de realizar la solicitud hacia el servidor web destino. Como cada servicio trabaja escuchando en puertos distintos: SQUID 3128 mientras que DANSGUARDIAN 8080, la herramienta dansguardian realiza el proceso de redirección hacia el puerto correspondiente al proxy-caché (3128) para trabajar en conjunto. En el siguiente gráfico se ilustra el flujo de información a través de cada herramienta.

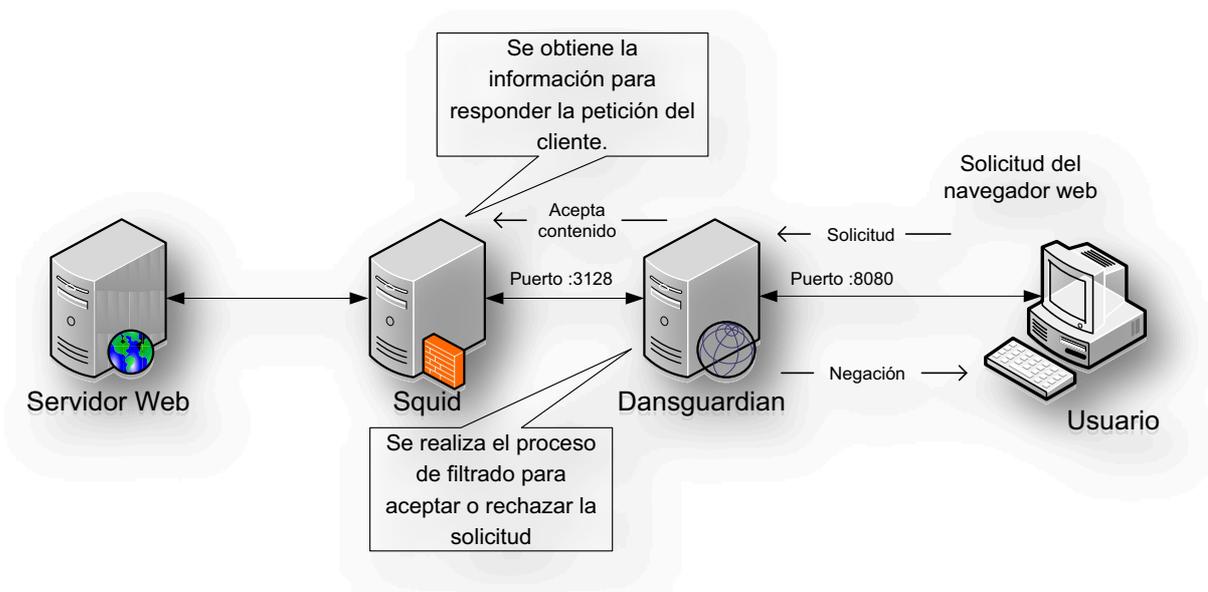


Figura B.30 Flujo de información a través de Squid y Dansguardian.

Instalación y configuración de dansguardian

La instalación de dansguardian es similar al resto de herramientas dentro de Mandriva, hay que tomar en cuenta que cada herramienta requiere de algunas dependencias para su funcionamiento, de esta manera:

- En la sección de búsqueda del gestor de software se debe ingresar la palabra dansguardian y el gestor mostrará las versiones disponibles de la herramienta, se seleccionará una versión, a continuación el gestor

mostrará las dependencias necesarias, se aceptará cada una y se procederá a la instalación respectiva.

La herramienta instalará cada archivo de configuración y directorio con el contenido respectivo, por lo general la mayoría de herramientas ya están listas para su funcionamiento con solo instalarlas. Para verificar que el servicio de dansguardian se encuentre en funcionamiento hay que emplear el comando `service` en una consola o mediante la interfaz gráfica disponible en Mandriva.

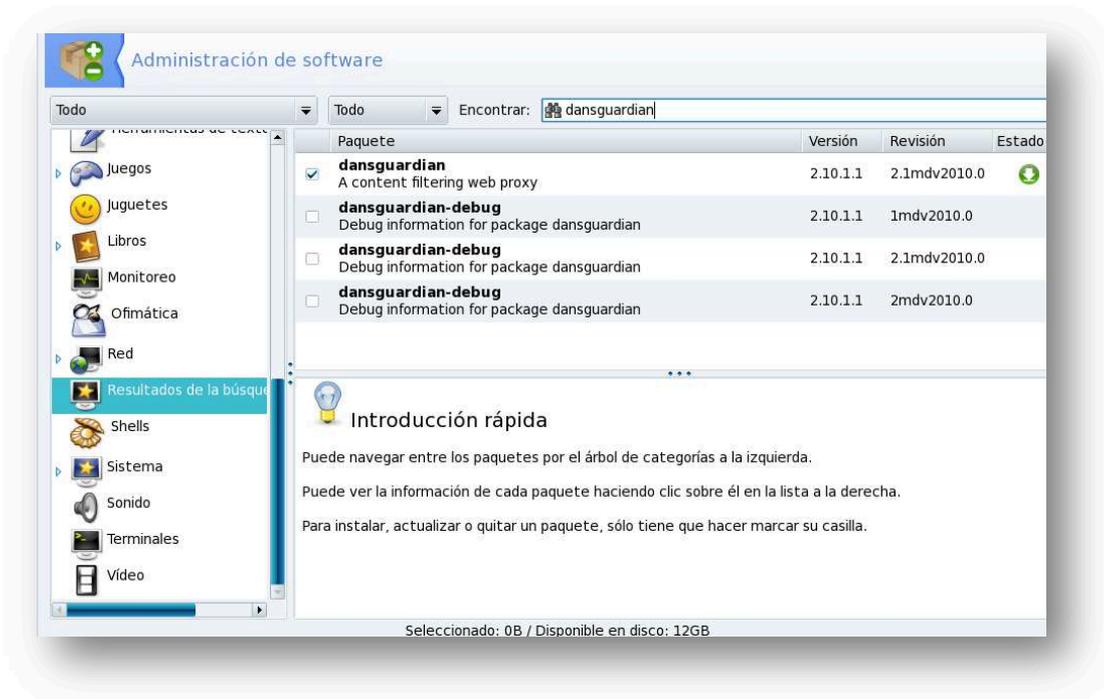


Figura B.31 Instalación de dansguardian en Mandriva.

Luego de verificar que el servicio se encuentre activo hay que tener en cuenta que dansguardian trabaja en conjunto con squid, por esta razón es necesario también verificar que el servicio de squid se encuentre funcionando, esto se lo realiza de manera similar al caso de dansguardian.

Otro punto importante constituye el manejo de puertos para cada servicio, como se mencionó para el caso de dansguardian el puerto por defecto es 8080, es por esta razón que se tiene que configurar previamente el explorador web cliente para verificar el funcionamiento de dansguardian.

- **Configuración básica**

La configuración básica del servicio dansguardian se la realiza en el archivo de configuración localizado en el directorio `/etc/dansguardian/dansguardian.conf` al igual que el resto de herramientas el archivo de configuración contiene información a manera de comentarios de las modificaciones posibles que se pueden realizar para la configuración. Las líneas más importantes para la configuración básica son:

UNCONFIGURED - Please remove this line after configuration

Esta línea se debe comentar mediante un signo # al inicio, para indicar que el archivo ya ha sido modificado.

La sección llamada *Network Settings* es utilizada para definir los puertos y direcciones IP con las que va a trabajar el servicio dansguardian:

Sintaxis	Descripción
filterip = <i>dirección IP</i>	Determina la dirección IP a utilizar por el servicio de dansguardian.
filterport = <i>puerto</i>	Especifica el puerto que dansguardian va a escuchar para realizar el proceso de filtrado.
proxyip = <i>dirección IP</i>	Determina la dirección IP del proxy.
proxyport = <i>puerto</i>	Especifica el puerto que dansguardian utilizará para conectarse al proxy.

Tabla B.9 Entradas para configurar direcciones IP y puertos.

Para un servidor que provee servicios a una red local y accede a internet utilizando interfaces de red distintas, la dirección *filterip* corresponderá a la dirección IP de la interfaz de red a la cual se enlazan los clientes de la red local. Después de estas directivas las siguientes líneas a configurar son:

languagedir = '/usr/share/dansguardian/languages'

languagedir indica el directorio donde se encuentran los diferentes lenguajes que soporta el servicio dansguardian, la ubicación del directorio destinado a lenguajes

puede variar ya que depende de la distribución Linux para el caso de Mandriva la ruta correspondiente es la indicada.

language = 'spanish'

language indica el idioma a emplear por dansguardian en este caso spanish, la determinación del idioma a emplear es importante debido a que de esta manera se asegurará que las páginas web que muestra el servicio puedan ser entendidas por el cliente que realiza la solicitud.

Modificando estas pocas líneas el servicio dansguardian quedará listo para su funcionamiento, tan solo se requiere configurar un explorador web en una maquina cliente para que acceda al servicio web mediante la dirección IP del servidor y el correspondiente puerto. Por ejemplo, 192.168.0.1:8080. De esta manera al ingresar una URI prohibida el servicio dansguardian generará una página web indicando la razón por la cual se negó el acceso al recurso solicitado. No hay que olvidar que dansguardian y squid trabajan en conjunto por lo que hay que tener presente que si se modifican los puertos de estas herramientas el servicio dejará de funcionar.

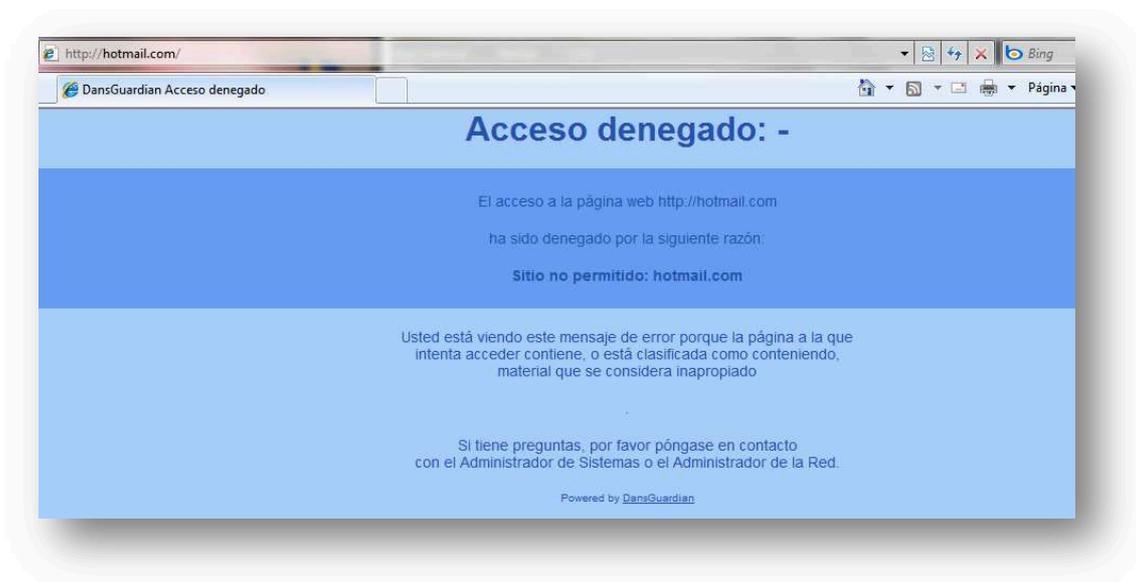


Figura B.32 Dansguardian filtrando contenido.

- **Configuración avanzada**

Debido a que squid y dansguardian escuchan en sus respectivos puertos, un cliente puede configurar su explorador web para que realice peticiones en el puerto de squid (3128), de esta manera saltará el filtrado de contenidos implementado con dansguardian. Para evitar este tipo de problemas hay que emplear una herramienta adicional que maneje los paquetes dentro del servidor, es el caso de Iptables que mediante sus reglas permitirá el redireccionamiento de puertos. Una regla simple modificará el flujo de paquetes para que las solicitudes que un cliente realice saltando el servicio dansguardian no pueda acceder a squid directamente, ya que será obligado a pasar por el filtrado realizado por dansguardian. La línea que se ingresará en el archivo rules dentro de shorewall es:

```
#ACTION SOURCE DEST PROTO DEST SOURCE ORIGINAL RATE
# PORT PORT(S) DEST LIMIT
REDIRECT loc 8080 tcp 3128 - - -
#LAST LINE -- ADD YOUR ENTRIES BEFORE THIS ONE -- DO NOT REMOVE
```

La regla es simple, los paquetes de la zona loc (red local) con protocolo TCP y que tengan como destino el puerto 3128 serán redireccionados al puerto 8080.

Al realizar el proceso de redireccionamiento el servicio queda listo para que todos los clientes puedan realizar solicitudes y pasen por el servicio proporcionado por dansguardian. Ahora, el siguiente paso es definir las políticas de acceso aplicadas a los clientes, para ello dansguardian realiza el proceso de filtrado empleando un conjunto de archivos llamados listas, estas listas se encuentran almacenadas dentro del directorio */etc/dansguardian/lists*, cada archivo tiene un nombre que lo identifica y es invocado al momento de realizar el proceso de filtrado. En la tabla B.10 se indican las listas más empleadas con su descripción correspondiente.

Archivo	Descripción
bannedphraselist	Almacena un conjunto de palabras prohibidas, las palabras van entre signos <>. Por ejemplo, <sexo>. Se puede utilizar combinaciones de palabras las cuales pueden contener espacios.
Bannedmimetyplist	Almacena un conjunto de tipos MIME prohibidos.
Bannedextensionlist	Almacena un conjunto de extensiones de archivos no permitidos.
Bannedregexpurllist	Almacena un conjunto de expresiones regulares empleadas para el proceso de filtrado.
Bannedsitelist	Almacena un conjunto de sitios prohibidos. Al ingresar un nombre de dominio se bloqueará todos sus subdominios.
Bannedurllist	Bloqueará partes específicas de un sitio web. Por ejemplo, un subdominio a diferencia de bannedsitelist.
Exceptionsitelist	Almacena un conjunto de nombres de dominio que no serán filtrados, el nombre de dominio no debe empezar con http:// o www.
Exceptioniplist	Almacena un conjunto de direcciones IP de los clientes que tiene acceso sin restricción. Por ejemplo, el administrador de la red.
Exceptionphraselist	Almacena un conjunto de palabras que pasarán el filtro, se tienen que

	guardar empleando los signos <> entre la palabra.
--	---

Tabla B.10 Listas empleadas para procesos de filtrado.

Si se quiere agregar una política de acceso es necesario ingresarla dentro del archivo de filtrado correspondiente. Por ejemplo, para negar el acceso al sitio web *www.youtube.com* se realizaría lo siguiente:

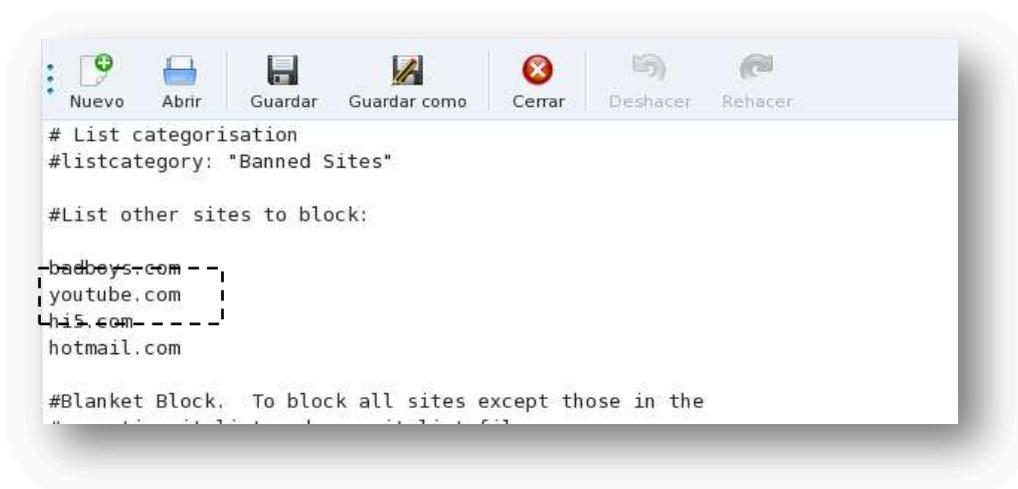


Figura B.33 Negación del acceso al dominio youtube.com.

Como se puede apreciar la manera de definir una política de acceso requiere exclusivamente la manipulación de los archivos donde se almacenan URI`s, palabras, tipos MIME, extensiones. Como ayuda cada lista cuenta con ejemplos a manera de comentarios, al igual que ciertos ejemplos que permiten comprender el funcionamiento de la lista correspondiente. Hay que tener presente que se debe realizar un proceso de reinicio del servicio de manera que los cambios realizados sean tomados en cuenta de inmediato por dansguardian.

ESTADO DEL SISTEMA

En adición al manejo de distintos servicios sobre un servidor Linux, también se hace necesario la implementación de un medio de visualización que permita obtener información acerca de cómo se encuentra el sistema operativo en distintos aspectos. Por ejemplo, el estado de la memoria, la capacidad del o los

discos duros, así como también la información de las interfaces que maneja la red. Por esta razón a continuación se mencionará un conjunto de comandos propios del sistema operativo Linux que permiten obtener esta información y manipularla de acuerdo a los requerimientos necesarios. La mayoría de comandos vienen por defecto en el sistema operativo, pero hay que tener presente que esto dependerá del tipo de distribución empleada. Los siguientes comandos serán desarrollados sobre la distribución Mandriva a través de una consola para verificar su funcionamiento, en el capítulo siguiente esta información será manipulada para mostrarla a través de una interfaz Web.

Estado de memoria

Para visualizar el estado de la memoria en lo referente a como está siendo empleada por el Sistema Operativo, se utiliza el comando **free**, este comando proporciona información detallada de la cantidad de memoria física que se está utilizando en el momento de su ejecución, además cuenta con varias opciones que agregan funciones adicionales, entre las distintas opciones se tiene:

- **[-b -k -m -g]:** estas opciones son empleadas para proporcionar la información de salida ya sea en bytes, Kilobytes, Megabytes o Gigabytes respectivamente.
- **-l:** muestra información detallada acerca de estadísticas de alto y bajo uso de memoria.
- **-o:** proporciona la información empleando un formato antiguo donde no consta la línea referente a *+/+ buffers/cache*.
- **-s:** actualiza la información cada cierto tiempo (en segundos) mostrándola en la salida correspondiente.
- **-t:** muestra la información acerca del uso de memoria añadiendo una nueva línea al final con la información de uso total de memoria.
- **-V:** muestra información acerca de la versión del comando.

La sintaxis es simple, se ingresa el nombre del comando en este caso *free* seguido de la opción respectiva. La ejecución de este comando no requiere una

opción obligatoria ya que por sí solo proporciona la información necesaria. En la siguiente figura se muestra la información desplegada por el comando.

```

Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# free
              total        used         free       shared    buffers     cached
Mem:          2979804    1924076    1055728          0        146500    1193132
-/+ buffers/cache:    584444    2395360
Swap:          433712          0         433712
[root@localhost ~]#

```

Figura B.34 Comando free.

Como se puede apreciar el comando proporciona la información a manera de tabla, los valores numéricos se encuentran en bytes. La primera fila corresponde a la asignación de memoria física al sistema operativo, en este caso un total de 2979804 bytes de los cuales se han usado 1924076 y se encuentran libres 1055728. La siguiente línea corresponde a la memoria que está siendo utilizada por el kernel en ese momento 584444 bytes. La siguiente línea corresponde a la memoria de intercambio SWAP, al igual que el resto de líneas proporciona la información correspondiente a su uso. Generalmente esta memoria tiene un uso de 0 bytes, este valor es ideal ya que indica que el sistema no está sobre exigido.

Estado del disco duro

En lo referente al estado del disco duro se proporciona estadísticas acerca de la cantidad de información que esta almacenada ya sea en los discos, en las particiones o en los espacios propios del sistema operativo como la raíz (/) y el directorio home (/home). Para obtener esta información estadística se emplea el comando **df**, este comando también cuenta con un conjunto de opciones para proveer información adicional. Por ejemplo:

- **-a**: incluye sistemas de archivos ficticios
- **-B**: usa bloques de tamaño de byte
- **-h**: imprime los valores en un formato entendible. por ejemplo 4G.
- **-T**: imprime el tipo de sistema de archivos.

- **--help**: muestra información de ayuda para el comando.
- **--version**: proporciona la versión del comando.

La figura B.35 proporciona la salida obtenida al ejecutar el comando **df -T**:

```

Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# df -T
S.ficheros  Tipo  Size  Used  Avail  Use%  Montado en
/dev/sdb5   ext3  6,1G  3,8G  2,1G  65%  /
/dev/sdb7   ext3  1,2G  563M  635M  47%  /home
/dev/sdb1   fuseblk 226G  212G  14G   95%  /media/Daniel
/dev/sda2   fuseblk 232G  104G  128G  45%  /media/disk
/dev/sda1   fuseblk 1,5G  160M  1,4G  11%  /media/TOSHIBA SYSTEM VOLUME
[root@localhost ~]#

```

Figura B.35 Comando df.

Como se puede observar el comando *df* proporciona información detallada acerca del uso del o los discos duros montados en el sistema. Cada columna cuenta con su respectiva cabecera que indica el tipo de información que se proporciona. Por ejemplo, *Avail* indica la cantidad de espacio disponible, en el caso del directorio raíz (*/*) para la figura B.35 se tiene disponible 2.1G, la letra G indica cómo se mencionó anteriormente la unidad de medida en este caso Gigabytes.

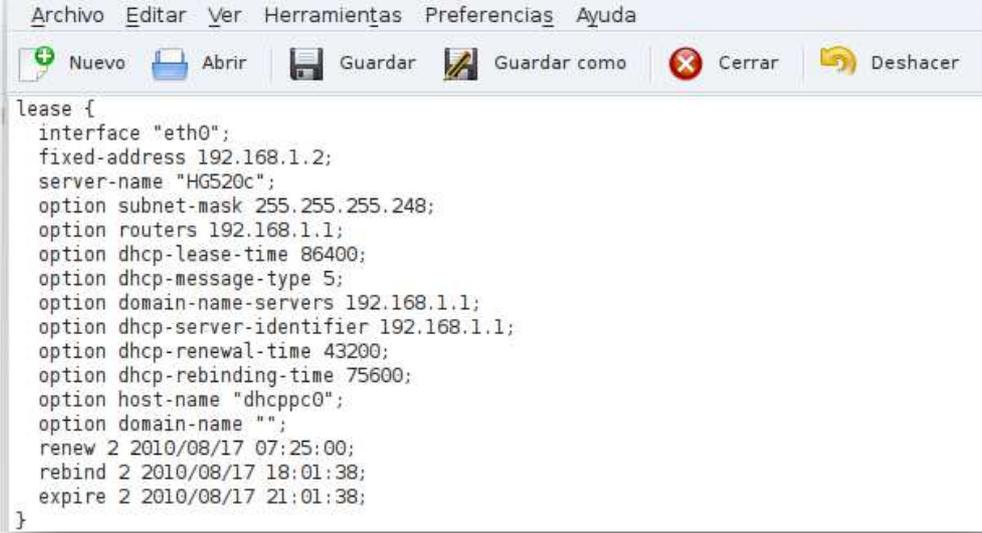
Estado de la red

La información correspondiente al estado de la red se enfoca a como se hallan configuradas las interfaces dentro del sistema operativo. Cada interfaz en el caso del presente proyecto tiene una configuración distinta. Se contará con una interfaz que obtendrá el servicio de Internet a través de un dispositivo proveedor, mientras que otra interfaz proporcionará los distintos servicios a la red local. Para el primer caso la configuración puede emplear el protocolo DHCP para obtener la información que le permita acceder al servicio. Mientras que la segunda interfaz se configurará de manera estática para que los usuarios accedan a los servicios disponibles en la red. Para obtener toda la información mencionada se requiere manipular archivos de configuración propios del sistema operativo así como ciertos comandos. Estos archivos y comandos se detallan a continuación:

- ***/var/lib/dhcp/dhclient-“interfaz”.leases***

Este archivo almacena la información correspondiente al cliente dhcp (interfaz que obtendrá el servicio de internet), para el caso de la siguiente figura la interfaz que

ha obtenido la información de un dispositivo externo es eth0. El contenido de este archivo corresponde a la información obtenida al realizar una solicitud a un servidor DHCP y obtener una concesión del servicio.



```

Archivo  Editar  Ver  Herramientas  Preferencias  Ayuda
+ Nuevo  [Icono] Abrir  [Icono] Guardar  [Icono] Guardar como  [Icono] Cerrar  [Icono] Deshacer

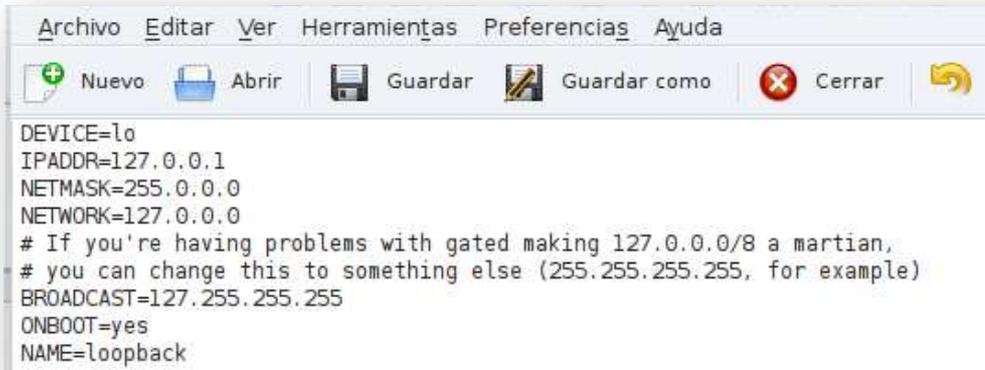
lease {
  interface "eth0";
  fixed-address 192.168.1.2;
  server-name "HG520c";
  option subnet-mask 255.255.255.248;
  option routers 192.168.1.1;
  option dhcp-lease-time 86400;
  option dhcp-message-type 5;
  option domain-name-servers 192.168.1.1;
  option dhcp-server-identifier 192.168.1.1;
  option dhcp-renewal-time 43200;
  option dhcp-rebinding-time 75600;
  option host-name "dhcpcp0";
  option domain-name "";
  renew 2 2010/08/17 07:25:00;
  rebind 2 2010/08/17 18:01:38;
  expire 2 2010/08/17 21:01:38;
}

```

Figura B.36 Información DHCP al obtener una concesión.

- ***/etc/sysconfig/network-scripts/ifcfg-"interfaz"***

Para verificar la configuración ya sea dinámica o estática de una interfaz se emplea el archivo de configuración ifcfg-"interfaz" en el caso de la figura B.37 la interfaz es loopback "lo", el termino interfaz (***ifcfg-"interfaz"***) va a variar dependiendo de la cantidad de interfaces que posea el host. Por ejemplo, eth0, eth1, wlan0, etc.



```

Archivo  Editar  Ver  Herramientas  Preferencias  Ayuda
+ Nuevo  [Icono] Abrir  [Icono] Guardar  [Icono] Guardar como  [Icono] Cerrar  [Icono] Deshacer

DEVICE=lo
IPADDR=127.0.0.1
NETMASK=255.0.0.0
NETWORK=127.0.0.0
# If you're having problems with gated making 127.0.0.0/8 a martian,
# you can change this to something else (255.255.255.255, for example)
BROADCAST=127.255.255.255
ONBOOT=yes
NAME=loopback

```

Figura B.37 ifcfg-lo, información de la interfaz virtual loopback.

Las líneas importantes dentro de este archivo son:

- **DEVICE:** el cual indica el nombre de la interfaz,
- **IPADDR:** que proporciona la dirección IP estática asignada a la interfaz.
- **NETMASK:** que indica su correspondiente máscara.
- Para el caso de una interfaz que no emplea asignaciones estáticas se cuenta con la línea **BOOTPROTO** la cual indica el tipo de protocolo utilizado para obtener acceso a la red. la información de concesión será almacenada en el archivo mencionado en párrafos anteriores, mientras que en este archivo solo se indicarán algunos parámetros.

- **Comando ifconfig**

Este comando proporciona la información de todas las interfaces presentes en el equipo. Es el comando más utilizado para obtener información de los dispositivos de red y verificar si estos se encuentran configurados. La salida del comando ifconfig se puede observar en la figura B.38:

```

Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:1E:33:A1:D8:97
          inet addr:192.168.1.2  Bcast:192.168.1.7  Mask:255.255.255.248
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:20673 errors:0 dropped:0 overruns:0 frame:0
          TX packets:15361 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:22942909 (21.8 MiB)  TX bytes:1783766 (1.7 MiB)
          Interrupt:29 Base address:0x4000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:19788 errors:0 dropped:0 overruns:0 frame:0
          TX packets:19788 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1612802 (1.5 MiB)  TX bytes:1612802 (1.5 MiB)

```

Figura B.38 Comando ifconfig.

Este comando proporciona un resumen de cada una de las interfaces presentes en el equipo, proporcionando información acerca de:

Su dirección IP, su correspondiente máscara, dirección de broadcast, así como también información acerca del flujo de paquetes a través de la interfaz. Finalmente se indica la cantidad de bytes que han sido transmitidos y recibidos por la interfaz añadiendo la cantidad en una unidad entendible. Para la figura B. 38 en el caso de RX se tiene 1.5 MiB.

Los tres elementos mencionados en esta sección permiten obtener la mayor cantidad de información referente a la configuración de cada interfaz. De esta manera se puede manipular toda esta información para generar un documento que proporcione los detalles de configuración de las interfaces presentes en el sistema.

- **comando ROUTE**

Este comando es utilizado para configurar y administrar las rutas IP en el sistema. La tabla de enrutamiento IP mostrada con el comando *route* es lo que siempre “mira” el kernel antes de enviar un paquete a la red a través de sus interfaces físicas. Es el mapa que usa IP para saber por dónde tiene que mandar el tráfico para que llegue a su destino. Al ejecutar el comando *route* se obtiene:

```

Archivo Editar Ver Historial Marcadores Preferencias Ayuda
[root@localhost ~]# route
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.1.0 * 255.255.255.248 U 0 0 0 eth0
link-local * 255.255.0.0 U 1002 0 0 eth0
default 192.168.1.1 0.0.0.0 UG 0 0 0 eth0
[root@localhost ~]#

```

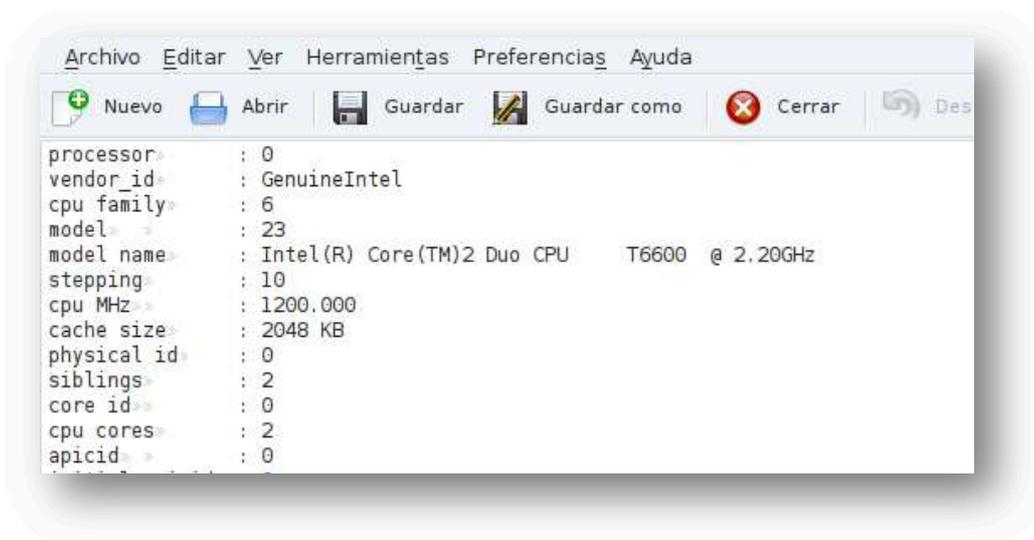
Figura B.39 Comando route.

La primera columna indica el destino, para la primera línea 192.168.1.0 se refiere a la red a la cual se encuentra enlazada la interfaz en este caso eth0. Si existieran mas interfaces se especificarían de la misma manera que la línea uno. La segunda línea corresponde a la dirección de loopback. La última línea cuyo destino es default corresponde a la salida que tendrán los paquetes para dirigirse a una red externa. La columna gateway se refiere a la dirección del dispositivo de enlace con otras redes. Cuando el valor es * indica que no se va a emplear un gateway ya que se trata de la misma red. Para la última línea se indica que la

puerta de enlace corresponde a 192.168.1.1 y corresponderá al default Gateway siendo esta un router o un computador que interconecta dos o más redes. La siguiente columna corresponde a las mascararas de red de las direcciones destino, en el caso de la primera y segunda línea corresponden a las mismas mascararas de las interfaces dado que la información no abandona la red. Para la última línea “default” la máscara correspondiente es 0.0.0.0 dado que se va a enrutar a todas las direcciones. Las siguientes columnas corresponden a parámetros adicionales, siendo la última columna la referente a la interfaz a la cual está asociada cada ruta en la tabla. Con esta tabla de enrutamiento se añade mayor información al estado de la red.

Estado de CPU

Al igual que el resto de sistemas operativos, Linux proporciona información acerca del procesador, su capacidad y el uso que le da el sistema operativo. Esta información es almacenada en un archivo el cual describe todas las características del procesador. La ubicación del archivo para el caso de mandriva linux es ***/proc/cpuinfo***. A continuación se muestra el contenido de este archivo para un hardware específico:



```

processor: 0
vendor_id: GenuineIntel
cpu family: 6
model: 23
model name: Intel(R) Core(TM)2 Duo CPU T6600 @ 2.20GHz
stepping: 10
cpu MHz: 1200.000
cache size: 2048 KB
physical id: 0
siblings: 2
core id: 0
cpu cores: 2
apicid: 0

```

Figura B.40 Archivo cpuinfo.

Los campos importantes dentro de este archivo, los cuales van a ser visualizados posteriormente mediante la interfaz web se detallan en la tabla B.11:

Campo	Descripción.
processor	Número de identificación para cada procesador. Si solo existe un procesador su valor será 0.
vendor_id	Proporciona el nombre del fabricante del procesador.
model name	Indica el nombre del procesador.
cpu mhz	Muestra la velocidad del procesador.
cache size	Indica la cantidad de memoria caché disponible.
core id	Número de identificación para el núcleo, es aplicable para el caso en el que un procesador posea más de un núcleo.
cpu cores	Proporciona la cantidad de núcleos que posee el procesador.

Tabla B.11 Algunos campos en el archivo `cpuinfo`.

Si el procesador cuenta con más de un núcleo el archivo `cpuinfo` proporcionará la información acerca de estas características. En la figura B.40 se puede observar esta característica mediante `cpu cores: 2` indicando que el procesador cuenta con 2 núcleos. Dentro del archivo ejemplo, el primer párrafo corresponde al núcleo 1, identificándolo mediante la línea `processor` y `core id` que tienen el valor 0 para referirse al primer núcleo. El conjunto de líneas no son más que un resumen de las características del procesador.

En lo referente al uso del procesador por parte del sistema operativo, se cuenta con un archivo dentro del directorio `proc` que proporciona un conjunto de tiempos con la finalidad de obtener mediante cálculos el uso de CPU en un intervalo de tiempo. Este archivo es `/proc/stat`:

```

Archivo  Editar  Ver  Herramientas  Preferencias  Ayuda
Nuevo  Abrir  Guardar  Guardar como  Cerrar
cpu 91690 1345 46921 918018 11626 794 156 0 0 0
cpu0 46360 547 23169 446524 6354 243 66 0 0 0
cpu1 45329 798 23751 471493 5271 551 89 0 0 0
intr 6907703 1787244 10439 0 0 0 0 0 0 1 11322 0 0 333093 0 0 0 0 3
ctxt 11313616
btime 1282598254

```

Figura B.41 Parte de la información disponible en `/proc/stat`.

Las líneas importantes dentro de este archivo son aquellas que inician con la palabra *cpu*, como se puede observar, para el caso de la figura B.41 se proporciona tanto la información total mediante la línea *cpu*, así como la correspondiente a cada núcleo del procesador mediante *cpu0* y *cpu1*, dado que el hardware empleado para la figura B.41 corresponde a un procesador de doble núcleo. Las columnas importantes y las cuales van a servir para calcular el uso de CPU son las 4 columnas luego de la palabra *cpu*, ya sea para el caso de la información total, así como para sus correspondientes núcleos. Cada columna tiene una unidad de medida denominada jiffies (1/100 de un segundo) que no es más que una unidad de tiempo.

- La primera columna se refiere al número de jiffies que se han empleado en modo usuario.
- La segunda columna se refiere al número de jiffies que se han usado en modo usuario con prioridad baja.
- La tercera columna se refiere al número de jiffies que se han utilizado por el sistema.
- La cuarta columna se refiere al número de jiffies de inactividad del sistema.

Las siguientes columnas proporcionan información relevante para el cálculo del uso de la CPU por lo que no son mencionadas. La fórmula a emplear es aquella que permite obtener el porcentaje en base a una cantidad total y su respectivo uso.

$$\% = \frac{uso}{total} \times 100$$

Para la variable uso, se tendrá que sumar los tiempos (jiffies) que ha empleado el sistema operativo, es decir las 3 primeras columnas. Mientras que para total se añade a la suma anterior el tiempo de inactividad. Ahora, se debe tener presente que si estos valores se los obtiene en un instante de tiempo no representan el uso de la CPU ya que se tiene que verificar su uso en un intervalo de tiempo, para ello se tiene que esperar un intervalo de tiempo para generar dos medidas y en base a la información obtenida tanto en el instante inicial como el final determinar si ha

existido una variación, es decir se ha utilizado la CPU. Con ello la fórmula quedaría implementada de la siguiente manera.

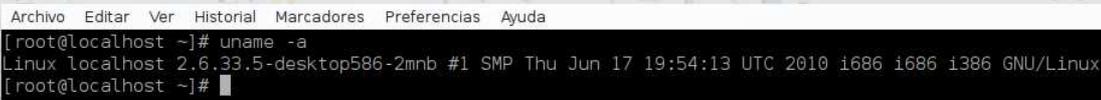
$$\% = \frac{(Uso_{t2} - Uso_{t1})}{(Total_{t2} - Total_{t1})} \times 100$$

Un archivo que genera información de interés se encuentra en el directorio `/proc/acpi/thermal_zone/THRM/temperature`, este archivo proporciona información acerca de la temperatura actual del procesador. Con todos estos directorios y cálculos se proporciona la mayor cantidad de información acerca del procesador.

Información adicional

Existen comandos que permiten obtener cierta información importante acerca del sistema, a continuación se mencionarán algunos de estos:

- **uname:** genera información acerca del kernel empleado por la distribución.



```

Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# uname -a
Linux localhost 2.6.33.5-desktop586-2mnb #1 SMP Thu Jun 17 19:54:13 UTC 2010 i686 i686 i386 GNU/Linux
[root@localhost ~]#

```

Figura B.42 Comando `uname -a`.

Opciones:

- **-a:** genera toda la información acerca del kernel.
 - **-s:** proporciona el nombre del kernel: para la figura *Linux*.
 - **-n:** proporciona el nombre de host: *localhost*
 - **-r:** proporciona la versión del kernel: *2.6.33.5-desktop586-2mnb*
 - **-v:** proporciona la fecha de publicación del kernel: *#1 SMP Thu Jun 17 19:54:13 UTC 2010*
 - **-m:** proporciona la arquitectura del procesador: *i686*
 - **-o:** proporciona el nombre del sistema operativo: *GNU/Linux*.
- **uptime:** muestra cuanto tiempo se encuentra encendido el sistema junto con otra información adicional.



```

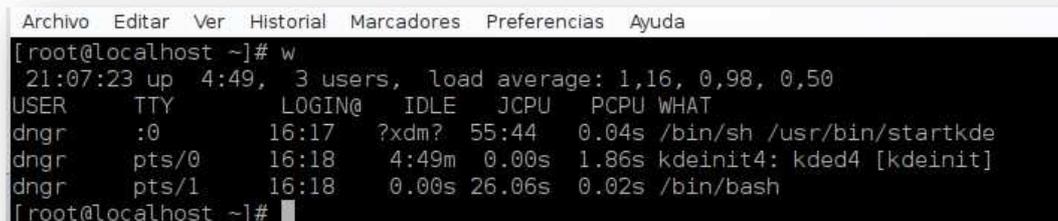
Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# uptime
 20:53:34 up  4:35,  3 users,  load average: 0.01, 0.04, 0.08
[root@localhost ~]#

```

Figura B.43 Comando uptime.

- primero se indica la hora actual: *20:53:34*.
- luego el tiempo que se encuentra encendido: *up 4:35* (4h 35min).
- proporciona los usuarios en el sistema: *3 users*.
- se indica la carga promedio en él: último minuto, 5 minutos y 15 minutos: *load average: 0.01, 0.04, 0.08*.

Dentro del comando *uptime* se indica el número de usuarios en el sistema pero no se indica cuales son estos usuarios, para este caso se emplea el comando *w*:



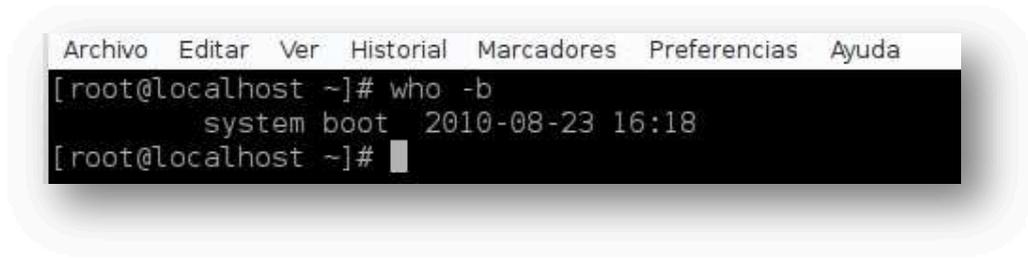
```

Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# w
 21:07:23 up  4:49,  3 users,  load average: 1,16, 0,98, 0,50
USER      TTY      LOGIN@   IDLE   JCPU   PCPU   WHAT
dngr      :0       16:17    ?xdm? 55:44  0.04s  /bin/sh /usr/bin/startkde
dngr      pts/0    16:18    4:49m  0.00s  1.86s  kdeinit4: kded4 [kdeinit]
dngr      pts/1    16:18    0.00s  26.06s 0.02s  /bin/bash
[root@localhost ~]#

```

Figura B.44 Comando w.

La primera columna indica los nombres de los usuarios en el sistema, la segunda columna indica el terminal que está utilizando cada usuario, LOGIN@ proporciona la hora en la cual los usuarios se loguearon en el sistema, las siguientes 3 columnas proporcionan información acerca del uso de CPU mientras que la última columna proporciona información acerca de lo que estaba ejecutando cada usuario, al momento de utilizar el comando *w*. El comando *who* muestra información similar al comando *w* pero de manera reducida, con la opción *-b* el comando *who* proporciona la fecha y hora en la que el sistema inicio.

A terminal window with a menu bar at the top containing 'Archivo', 'Editar', 'Ver', 'Historial', 'Marcadores', 'Preferencias', and 'Ayuda'. The terminal text shows the prompt '[root@localhost ~]#', the command 'who -b', the output 'system boot 2010-08-23 16:18', and the prompt '[root@localhost ~]#' followed by a cursor.

```
Archivo  Editar  Ver  Historial  Marcadores  Preferencias  Ayuda
[root@localhost ~]# who -b
system boot 2010-08-23 16:18
[root@localhost ~]# █
```

Figura B.45 Comando `who -b`.

De esta manera se concluye con la sección referente a la información acerca del sistema, con el objetivo que esta información pueda ser manejada posteriormente con la finalidad de proporcionar al usuario la información acerca de cómo está funcionando el sistema en los aspectos mencionados.