PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad de Ingeniería de Sistemas

DISERTACION DE GRADO PREVIA LA OBTENCION DEL TITULO DE INGENIERO EN SISTEMAS

"GUIA DE ADMINISTRACIÓN, IMPLANTACIÓN Y DESARROLLO DE SITIOS Y SERVIDORES WEB APLICADAS A LA INTRANET DE LA ESCUELA DE SISTEMAS DE LA PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR - SEDE AMBATO"

Andrés Rubén López Andrade
Xavier Francisco López Andrade

DIRECTOR DE TESIS:
INGENIERO DAVID GUEVARA

Ambato, 1999

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

Unidad de Ingeniería de Sistemas

DISERTACION DE GRADO PREVIA LA OBTENCION DEL TITULO DE INGENIERO EN SISTEMAS

"GUIA DE ADMINISTRACIÓN, IMPLANTACIÓN Y DESARROLLO DE SITIOS Y SERVIDORES WEB APLICADAS A LA INTRANET DE LA ESCUELA DE SISTEMAS DE LA PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR - SEDE AMBATO"

Director : _____
Ing. David Guevara

Revisores: _____
Ing. Patricio Chambers

_____
Ing. Wigberto Sánchez

Andrés Rubén López Andrade
Xavier Francisco López Andrade

Ambato, 1999

DEDICATORIA

A mis abuelos, Nohemí, Romelia y Teófilo, que me enseñaron, con el ejemplo, el amor al trabajo y el respeto a la gente.

A mis padres, Martha y Jaime, por mostrarme que la vida es bella mientras hay amor y comprensión y por convertirse, al pasar los años, en amigos verdaderos.

A mis hermanos, Valeria, Francisco y Gustavo, por haber compartido tantos momentos felices juntos, que directa o indirectamente apoyaron al desarrollo de este trabajo.

A mi hija Micaela, por que ella es el motivo principal de mi vida.

Finalmente a mi esposa Heidie, por estar junto a mí en todos estos años, convirtiéndose en un apoyo firme gracias a sus palabras de aliento, además por darme la mayor alegría que puedo recibir en estos momentos, mi segundo pedazo de cielo.

Andrés

DEDICATORIA

A mis padres Jaime y Martha, a mis hermanos Andrés y Valeria, porque con ellos he aprendido que el trabajo, el esfuerzo y la dedicación son razones poderosas para vivir.

Francisco

AGRADECIMIENTO

A nuestros profesores, que nos enseñaron tantas cosas útiles, no solo para la carrera que escogimos, sino también para nuestro desarrollo personal.

A nuestro director de tesis, por su colaboración incondicional en el desarrollo de este trabajo.

A nuestros revisores, que con sus sugerencias y comentarios ayudaron al desarrollo y terminación de este trabajo.

Al personal docente de la Pontificia Universidad Católica – Sede Ambato, por su apoyo desinteresado y por que siempre supieron ser amigos.

Francisco y Andrés

INDICE

INDICE DE FIGURAS

PREFACIO

La información es la manera en que las organizaciones coordinan sus actividades y cumplen con sus objetivos. Las tecnologías, para manejar y distribuir la información, han cambiado durante los años, pero las funciones requeridas para las organizaciones humanas se han mantenido constantes. El esfuerzo para encontrar vías para autentificar requerimientos electrónicos es meramente un intento de cumplir con lo que hacemos con sellos, reconocimientos de firmas y notarios públicos en el mundo de los documentos en papel. La necesidad de asegurar la información en las redes es la misma necesidad que teníamos antes con firmas y sellos en documentos oficiales.

De cualquier modo, los requerimientos de seguridad no solo son las únicas funciones que estimulan las organizaciones a manejar la información. Si la información no permite algún uso futuro de valor o con un valor tangible, entonces no se necesita que sea asegurada. La información organizacional generalmente lleva contenidos que permiten realizar acciones que permitan ganar o perder recursos. Una organización amplifica su habilidad para controlar aquellos recursos mediante la división del trabajo entre varios individuos para cumplir con el objetivo general. Para que la organización sea efectiva, actividades y progresos deben ser coordinados. Una razón importante para compartir información en las organizaciones es el acuerdo en la coordinación de esas tareas y objetivos.

Un requerimiento para una coordinación efectiva es la consistencia de la información. No es muy eficiente si la existencia o localización de información importante permanece desconocida para un individuo que la necesita. Tampoco no es muy eficiente

si un equipo de trabajo trata de llegar a un consenso cuando cada miembro opera desde una información distinta que puede ser incompatible o inconsistente con los otros miembros del equipo. Bastante información se corrompe y requiere atención para ser actualizada. Muchas de las estructuras y procedimientos organizacionales han sido redefinidas a través de los años, para resolver estos problemas de información basada en papel.

El problema de la integridad y unidad de la información es mucho más simple cuando el contenido no cambia muy seguido, las actividades coordinadas no son muy largas o complejas y la información es centralmente recogida y distribuida. De todos modos, estos no son características comunes de muchas empresas hoy. Los ambientes distribuidos mas comúnmente encontrados ahora necesitan poder coordinar información de muchas maneras, y requieren un grupo distinto de estructuras administrativas y de procesos que muchas organizaciones han heredado.

Una Intranet ofrece muchas nuevas opciones para una más efectiva coordinación de actividades organizacionales en un ambiente distribuido de decisiones. A pesar de todo, las infraestructuras basadas en documentos inhiben las variadas características de muchas Intranets y no controlan efectivamente otras. Lo que se necesita es una infraestructura que usa la Intranet que cumple con los requerimientos para coordinaciones organizacionales mediante la correcta administración de las decisiones en un ambiente universitario.

La construcción de una Intranet efectiva requiere atención hacia tres áreas distintas: administración, técnica y contenido. La administración consiste en los roles, políticas,

procesos y organizaciones que se necesitan para manejar el ciclo de vida de una Intranet formal. La infraestructura técnica consiste en las redes, hardware, y software requeridas para apoyar los contenidos, programas de desarrollo, publicaciones y acceso hacia la Intranet. Y el Contenido requiere que los procesos sean desarrollados para apoyar necesidades especiales tales como conversaciones iniciales, creaciones de bases de datos o programas y el desarrollo de "impactantes" páginas Web.

Mucha de la atención a las Intranets se ha centrado en la implementación de la infraestructura técnica y el desarrollo de contenidos especiales. El primero de ellos es una extensión de los sistemas y administración de redes y el segundo una extensión natural de las publicaciones técnicas, procesos de programación y recursos.

1 ADMINISTRACIÓN DE INTRANETS

1.1 CAPÍTULO 1

1.1.1 HISTORIA

La red de Internet surgió de la necesidad del gobierno de los Estados Unidos de resolver un problema de estrategia militar en el período de la Guerra Fría: ¿Cómo se podrían comunicar las autoridades después de una guerra nuclear? RAN Corporation, una de las empresas encargadas de la estrategia militar estadounidense propuso una solución: crear una red de comunicaciones que no dependiera de un organismo central, que estuviera integrada por nodos o puntos de enlace de igual rango y con la misma capacidad de originar, transmitir y recibir mensajes, de modo que si alguno de estos nodos recibiera un ataque o dejara de funcionar, el resto de la red pudiera seguir en operación. Los mensajes en esta red se dividirían en paquetes, cada uno con su propia dirección; se originarían en algún nodo en particular, saltarían de lado a lado hasta llegar a otro nodo específico, de manera individual. La ruta que siguieran los paquetes realmente no importaba; lo importante era que llegaran. Si una ruta hubiera sido destruida, el paquete encontraría otra para llegar a su destino.

Esto permitió que la tecnología primitiva para redes fue desarrollada en los años 1950 y 1960 para capacitar a usuarios múltiples compartir computadoras tipo "Mainframes"

mediante el uso de terminales conectadas por cables a las computadoras centrales. En esta situación, no habían conexiones entre computadoras (puesto que los usuarios tenían terminales sin capacidad propia para realizar cálculos ni almacenar datos), ni mecanismos avanzados para compartir información mediante computadoras (la comunicación era entre cada usuario y la máquina central, no entre usuarios). A finales de la década de 1960 y a principios de los años 1970, varias tecnologías y estándares fueron desarrollados para permitir la conexión y la comunicación entre verdaderas computadoras, y el compartir información mediante redes de computadoras se hizo posible.

El experimento más importante en este proceso de compartir información mediante redes, comenzó hace más de veinte años cuando el Departamento de Defensa de los Estados Unidos empezó a financiar una serie de conexiones (basadas en líneas telefónicas y cables de fibra óptica) entre las computadoras de varios laboratorios nacionales y universidades de los Estados Unidos. Con el fin de intercambiar información acerca de proyectos de desarrollo e investigación financiados por el gobierno, y a compartir el tiempo de procesamiento en las inmensas computadoras que se necesitaban para ciertos problemas científicos. Esta red resultó tan productiva que otras redes nacionales e internacionales de telecomunicaciones entre centros de cómputo académicos y gubernamentales empezaron a aparecer. El Web de comunicaciones integrado por estas redes conformó el Internet inicial, y los protocolos y estándares utilizados en esta red todavía sirven como la base para la Internet actual

inmensamente mayor, así como para las Intranets internas de las corporaciones que son más pequeñas.

La interconexión de computadoras en los negocios comenzó, a pequeña escala, en la década de 1980 con el enlace de computadoras personales mediante cables dentro de una oficina, para compartir aparatos periféricos caros como discos duros grandes e impresoras de alta calidad. Estas primeras Redes de Area Local (LANs), se hicieron más elaboradas rápidamente, con énfasis creciente en la capacidad de compartir información entre usuarios de una oficina, y el desarrollo de sistemas avanzados de mensajería como correo electrónico y "groupware" para organizar el flujo de información. Rápidamente se hizo común en los negocios compartir información entre sucursales enlazando LANs de la oficina local con las Redes de Area Amplia ("Wide Area Networks", o WANs), utilizando canales de telecomunicaciones (líneas telefónicas, fibras ópticas, y enlaces por microondas y por satélite, etc.) para los enlaces de larga distancia entre LANs.

La tendencia a conectar redes de computadoras se hizo sumamente difícil debido a la incompatibilidad entre el hardware, el software y los estándares desarrollados para uso comercial: las Apple Macintosh no podían ser conectadas con PCs compatibles con IBM, las redes Ethernet no podían conectarse con redes de Token-Ring, archivos de Microsoft Word no podían ser leídos por programas de Word Perfect. La conquista de estas incompatibilidades ha sido el logro sobresaliente en redes en los años 1990, y los

estándares y servicios desarrollados originalmente para Internet han jugado un rol de liderazgo en la resolución de estos conflictos.

La tecnología de Internet es complicada y el uso de Internet por muchos años fue una tarea para expertos técnicos. Un paso crítico fue dado a principios de 1990 con el desarrollo del "World Wide Web" (WWW), una serie de protocolos y estándares de interfase que ofrecieron una interfaces gráfica y fácil de usar para el usuario para manejar las tareas de Internet, tanto como las interfaces de Apple Macintosh y Windows de Microsoft escondieron los detalles técnicos del uso de computadoras personales. Utilizando servicios y protocolos de Internet y una interfase de WWW, ahora es posible para personas no especializadas conectar una computadora a casi cualquier otra computadora en el mundo, siempre de ambas computadoras estén accesibles mediante líneas telefónicas y el uso de protocolos de comunicaciones de Internet.

Este crecimiento y renovación de la Sociedad Internet continua hasta nuestros días. En la actualidad, el Internet no solo esta conformada por redes interconectadas usando el protocolo IP, sino recientemente redes basadas en protocolos diferentes al IP han desarrollado módulos que las integran con las redes IP tradicionales.

1.1.2 DEFINICIÓN

Intranet es una infraestructura de comunicación. Esta basada en los estándares de comunicación de Internet y el contenido estándar de la red mundial (World Wide Web). Por lo tanto, las herramientas usadas para crear una Intranet son idénticas a aquellas usadas para el Internet y sus aplicaciones. La característica más notable de una Intranet es que el acceso a la información publicada esta restringido solo a los clientes y usuarios de una organización y, en nuestro caso, para la Universidad Católica del Ecuador – Sede Ambato. Por lo que se puede definir a una Intranet como:

La utilización de todas o parte de las tecnologías y de las infraestructuras de Internet para las necesidades de transporte y de tratamiento de los flujos de información internos de un grupo de usuarios.

Una Intranet puede estar constituida por muchos servicios:

- Foros de discusión entre equipos de proyectos al interior de la Universidad
- Interconexión segura de redes locales de la Universidad entre sus distintos edificios
- Creación de Webs Internos

- Creación de un servidor Web para ser utilizado al interior de la Universidad y desde el Internet por una comunidad cerrada de alumnos y profesores.
- Implementación de Correo Electrónico
- Video Conferencias
- Clases virtuales
- Etc.

La tarea más importante de la Intranet es proveer comunicación entre las computadoras; una computadora solo necesita encapsular sus datos en un paquete llamado *"Internet Protocol (IP) packet"*, y direccionarlo correctamente. Las computadoras interconectadas son las que tendrían la responsabilidad de asegurar la comunicación que se hubiera establecido. *"Cada computadora en la Intranet podrá conversar, al mismo nivel, con cualquier otra computadora de la Red".*

1.1.3 DISEÑO DE UNA INTRANET

Esta sección trata del proceso mediante el cual se diseñará una Intranet. Aunque sin duda se dará cuenta de que su diseño cambiará a medida que se implemente el Web y las necesidades de los usuarios deseen mas cambios, no se debe pasar por este proceso antes de iniciar un trabajo serio. La planeación de una Intranet redundará en una construcción más efectiva y en menos tiempo. Para esto se necesitará poseer conocimientos del manejo de un navegador Web y que se tenga una familiaridad con WWW (World Wide Web) como un todo. Aquí se conocerá las herramientas que se pueden utilizar para implementar un Servidor.

1.1.3.1 EL CÓMO Y POR QUÉ DE UNA INTRANET

Esta tesis esta planificada para utilizar el término *Intranet* para referirse a la manera en que la Universidad Católica de Ambato aprovecha World Wide Web y la tecnología Internet relacionada para llevar a cabo su trabajo esencial: ayudar a difundir la información y los servicios necesarios para los cuales está destinada la Universidad Católica de Ambato.

Muchas instituciones, organizaciones y compañías han instalado Servidores Web y los hacen accesibles a través de Internet, con la idea de poner información, servicios corporativos a disposición de otros o para vender cosas en el Web. Sin embargo, el

objetivo inicial de los pioneros del Web en el CERN, en Ginebra, era crear un medio para que los científicos del CERN pudieran compartir información con mayor facilidad. Con los que se concluye que la primera Internet fue una Intranet que se diseñó con el objetivo de distribuir información dentro de una organización, para su propio personal, y se concentró en la manera como el Web y la tecnología relacionada puede emplearse para profundizar el propósito por el cual existe la organización.

1.1.3.2 LOS USUARIOS DE INTRANET

La definición de las personas que utilizarán la Intranet de la Universidad Católica de Ambato es muy diferente de las personas que usan el Web público de una organización, cuyo enfoque principal es mostrar información a *gente del exterior* y dicha información es completamente general y valiosa para las relaciones públicas. Por otro lado el enfoque de una Intranet es hacia las personas de interior o estrechamente vinculadas con la Institución: Estudiantes, profesores y administradores de la Universidad Católica de Ambato. Se podrá acceder a información acerca de los departamentos, actividades, horarios, listados, etc.

Cuando se empiece a ponderar el diseño de la Intranet, la primera consideración debe definir con claridad la audiencia que pretende, es decir, sus usuarios. El propósito fundamental de la Universidad Católica de Ambato es la educación e investigación, y sus clientes primarios son estudiantes, profesores e investigadores, los cuales son

miembros de la Institución; en este caso sus usuarios son las personas dentro de la Institución quienes elaboran y usan los servicios del Intranet.

Estas son distinciones cruciales que se deben tener en cuenta al considerar y diseñar su Intranet. La decisión sobre la manera de diseñarla y la información que almacenará debe basarse en su audiencia objetivo: sus usuarios internos.

1.1.3.3 INFORMACIÓN Y SERVICIOS QUE LOS USUARIOS NECESITAN

Existen varios servicios que la Universidad Católica de Ambato puede ofrecer y que pueden ser parte de una Intranet, básicamente podemos enumerar tres de ellos:

- Servicios de Recursos Humanos y Bienestar Estudiantil
- Servicios de Administración
- Servicios de Sistemas de Información

a) Servicios de Recursos Humanos y Bienestar Estudiantil

Cuando la Universidad Católica de Ambato tiene que tratar con personal y estudiantes, hay una buena cantidad de trabajo de escritorio involucrado. Buena parte de este trabajo es la información que sus clientes necesitan, por ejemplo:

- Consultas e ingresos del inventario de la Biblioteca de la Universidad Católica de Ambato.

- Manuales de empleados y personal administrativo, reglas de comportamiento, reembolso de gastos, vacaciones y cosas por el estilo.
- Boletines de prensa con avisos gubernamentales acerca de salarios mínimos, políticas, empleos, horarios.
- Manuales de estudiantes, reglas de comportamiento, solicitudes, multas, justificaciones.
- Historiales de empleados y estudiantes, asistencias e información general.
- Comunicaciones de las Asociaciones de Estudiantes y reuniones de tipo general de los clubes deportivos y sociales.

Incluso esta información puede ya estar almacenada electrónicamente y se pueden transferir directamente al Sitio Intranet para su uso.

b) Servicios Administrativos

Existen ciertos registros del mobiliario, equipos y bienes de los que dispone la Universidad Católica de Ambato y que pueden ser ingresados en el Intranet tales como:

- Listado de mobiliarios de oficina, máquinas o equipos de computación.
- Mapas de imágenes del campus universitario que proporcionan instrucciones detalladas de donde se encuentran ubicadas las Facultades, Oficinas, Aulas y dependencias de la Universidad Católica de Ambato.

- El directorio telefónico del personal administrativo de la Universidad Católica de Ambato.
- Formularios para búsqueda y actualización de inventarios, pedidos o requisiciones y otras tareas.

Los formularios pueden ser requeridos a través del navegador usando una *Interfaz Común de Gateway* (CGI Common Gateway Interface) que interactúa con una base de datos residente en el Servidor, y que, además, puede ser enviada a través del Correo Electrónico. En nuestro caso estamos usando la tecnología ASP (Active Server Pages) Páginas activas del Servidor, que combina los controles Active-X, Visual Java, Visual Basic, etc. Permitiendo que se puedan ejecutar programas en el Servidor Web.

c) Servicios de Sistemas de Información

El departamento de Sistemas de la Universidad Católica de Ambato proporciona ya servicios de Información a los Usuarios a través de programas o bases de datos que acceden al Servidor Principal. Esta información ya puede ser ingresada a la Intranet generando ciertas ventajas como:

- *Preguntas más frecuentes:* en las que cualquier duda general acerca de la Universidad Católica de Ambato puede ser resuelta, tales como: Horarios de Clase, Profesores de las distintas facultades, Requisitos de admisión, etc. Estas preguntas generan respuestas preestablecidas que pueden conformar el núcleo de

la ayuda en Intranet, incluso con la ayuda de navegadores se pueden utilizar formularios como los de ¡Yahoo!, Lycos o Altavista para buscar respuestas.

- *Bases de Datos:* para introducir, actualizar o borrar información que se posee en las bases de datos de personal, administración, etc. Estas operaciones se llevarán con facilidad a través de formularios Web soportados por scripts CGI que tienen acceso a estas bases de datos. La ventaja es que los usuarios donde quiera que estén siempre tendrán la misma interfase que reconocen y con la cual se sienten cómodos.

- *Compartir Documentos* tales como hojas de cálculo, textos y que se permita que varias personas puedan compartir en reuniones o planificaciones. Una configuración apropiada del Servidor y de los navegadores Web puede permitir que el Directivo de la Universidad Católica de Ambato haga clic en un hiperenlace y pueda ver datos actualizados de estadísticas, notas, asistencia, etc.

- *Compartir Datos* para que profesores y alumnos puedan compartir archivos de datos desde sus aplicaciones o puedan presentarse deberes o tareas comunes a ellos.

1.1.3.4 DISEÑADOR DE LA RED

Este es el momento de determinar quien va a tener la responsabilidad de organizar y diseñar la Intranet. En el pasado podría haber sido del departamento de Sistemas, que poseía el monopolio de la información y procesamiento de datos, hubiera sido fácil asignar la responsabilidad de la configuración y diseño de una Intranet. En la actualidad, sin embargo, Internet e Intranet están basados en la computación distribuida y el departamento de Sistemas no puede controlar la información de cada computadora de la organización. Esto permitiría que los usuarios de la Universidad Católica de Ambato puedan crear sus propias páginas Web y ponerla a disposición de todos.

1.1.3.5 CONTROL DE LA INTRANET

Con el punto anterior se debe tomar en cuenta el Control de la Intranet, este es un punto delicado que debe ser tomado en cuenta muy seriamente, por lo tanto se deben establecer ciertos puntos que deberán ser resueltos:

¿Se requiere que la Intranet tenga una apariencia y organización común, o sólo importa el contenido?

¿Alguien va a aprobar cada una de las piezas de información antes de que vayan a la Intranet o cualquiera puede publicar cualquier cosa que le plazca?

Si los usuarios son libres de colocar cualquier cosa que se les plazca. ¿El Administrador de la Red se preocupará por el material inapropiado y / o el uso inadecuado de los sistemas de información y los recursos de personal de su organización?

¿Se aceptará y alentará la inevitable evolución de la Intranet, a medida que los usuarios y el Administrador de la Red descubran nuevas cosas que podrían llevar a cabo?

Si se ha utilizado la Red Mundial de Internet en cualquier grado, se reconoce como la feria de vanidades más grande del mundo: Las personas pueden, y lo hacen, colocar lo que se les antoje. Es un arma de dos filos. Se puede encontrar cosas en verdad asombrosas en Internet, muchas de las cuales pueden ser ofensivas. La manera como uno se sienta con respecto a este tipo de anarquía matizará de manera inevitable el enfoque que utilice para asignar la responsabilidad y establecer normas para la Intranet.

Al mismo tiempo, la naturaleza fundamental de Web como servicio distribuido, proporciona oportunidades incomparables para el desarrollo individual y organizacional; por lo que imponer una estructura rígida y autoritaria en la Intranet quizá inhiba esta clase de creatividad que puede acarrear innovaciones en el funcionamiento de la Universidad Católica de Ambato.

1.1.3.6 MODELOS ORGANIZACIONALES

Sobre la base del enfoque filosófico que se elija en la asignación de responsabilidades para la Intranet, hay varios modelos que se pueden seguir. Aquí hay tres:

Modelo centralizado con un solo Servidor Web administrado por un departamento específico de la Universidad Católica de Ambato, y un proceso formal para el desarrollo e instalación de nuevos servicios.

Modelo descentralizado donde todos son libres de configurar un Servidor Web y colocar recursos de elección en él.

Modelos mixtos con elementos de los modelos anteriores.

a) Modelo Centralizado

En este modelo jerárquico todos los servicios Web están centralizados. Sólo un sistema computacional de la Universidad Católica de Ambato opera el Servidor Web. Se responsabiliza a un individuo o grupo específico de la configuración, diseño y administración del Servidor. Todas las páginas Web (documentos, formularios, etc.) están diseñadas de manera central a solicitud de los usuarios. De este modo, si el departamento de Administración quisiera colocar en Web reglamentos de profesores, se requeriría una solicitud formal, lo que incluye solicitudes de contenido y diseño. El equipo dedicado a Web diseñaría y refinaría, en forma conjunta con el departamento de Administración, la página de reglamento de profesores y una vez completado el proceso pondría la página a disposición del Servidor Web. He aquí el modelo del proceso de aprobación, antes de colocar cualquier información en el Servidor Web central.

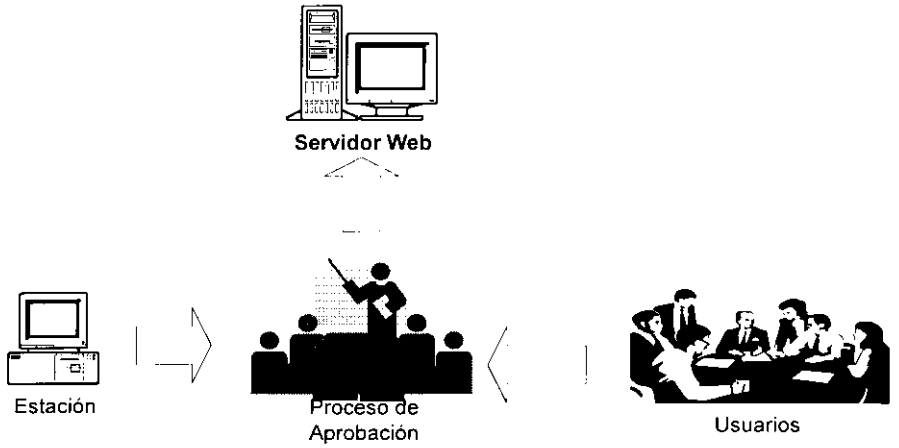


Figura # 1 Modelo Centralizado

Hay varias ventajas y desventajas en este modelo:

- Al concentrar la administración del Servidor Web, se tiene la oportunidad de estructurar una Intranet con un diseño consistente y un conjunto estándar de reglas. Así los usuarios verán un Servidor coherente y bien planificado.
- Simplifica la instalación y administración del Servidor Web e Intranet. Con lo cual todas las actualizaciones pueden ser hechas por una sola persona y la seguridad y respaldo de los datos se concentran en una sola máquina.
- Desde el punto de vista filosófico, atenta contra todo lo sucedido en el procesamiento de datos durante las dos últimas décadas y contradice todo lo que Internet representa.
- La administración centralizada puede entramparse con facilidad en asuntos organizacionales y de jerarquía, lo cual deteriora el potencial de respuesta rápida a los usuarios.

Este modelo se basa en el Web, si el sistema se detiene, todo se detiene. Esta política requiere tomar una decisión entre un tiempo inactivo costoso y u hardware de duplicación oneroso.

b) Modelo descentralizado

En este modelo el software del Servidor Web esta disponible para que todos los usuarios puedan tener libertad para establecer Servidores Web en sus propias estaciones y desarrollen sus propios documentos. Un usuario con experiencia moderada en computación puede hacer que un Servidor Web funcione en una tarde.

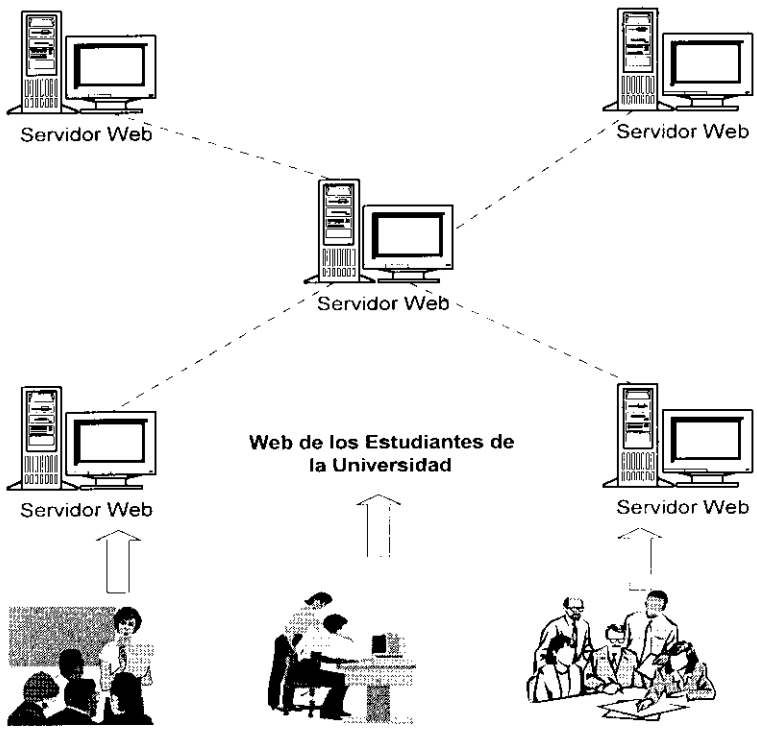


Figura # 2 Modelo Descentralizado

Como en el caso del modelo centralizado, en éste también hay puntos débiles y fuertes:

- El usuario instala su propio Servidor Web y es el más apto para saber con precisión el servicio que desea proporcionar con él.
- Se permite que la información que se tiene se puede compartir y distribuir rápidamente con un mínimo de esfuerzo
- Se establece una "anarquía" donde los usuarios establecen páginas Web relacionadas o no relacionadas sin orden alguno. Un paseo rápido por Internet muestra como esta anarquía conduce de lo sublime a lo absurdo, hasta lo, en verdad, obsceno.
- Este modelo tiene más peso en las instituciones académicas y de investigación, porque se permite la libertad individual de los investigadores y desarrolladores.
- Se establece vínculos entre páginas que no tienen relación entre sí, permitiendo que se genere confusión con el propósito principal de las páginas.

c) Modelo Mixto

En alguna parte entre los dos modelos anteriores esta el sitio de este modelo y es en donde la mayoría de las organizaciones sitúan su Intranet. Por ejemplo, se establece una política amplia en el sentido de que el propósito general de la Intranet es dar soporte a un grupo específico de usuarios y se permita cualquier cosa coherente con este propósito. En este caso se confía este aspecto al modelo centralizado al tiempo que se utilizaría aquellos aspectos del modelo descentralizado que dejan la mayor parte de los detalles a los usuarios.

1.1.3.7 PROPÓSITOS Y METAS DE LA INTRANET

a) Propósitos.-

Para trazar los propósitos de la Intranet es necesario integrar las ideas en una *Declaración de Propósitos*, como primer paso concreto hacia la realización de las ideas. Podemos citar algunos ejemplos de posibles propósitos:

- Proporcionar información a los alumnos
- Da información acerca de los reglamentos de la Universidad Católica de Ambato.
- Proporcionar un navegador como interfaz para las Bases de Datos.
- Utilizar el Intranet como un medio de compartir archivos desde aplicaciones comunes.

Cada declaración de propósitos debe ser específica y limitada, permite definir las tareas por realizar, en términos que se puedan captar con facilidad. Implica también que se puede dividir el trabajo de diseño y creación entre varios equipos otorgando un punto de orientación hacia el cual se pueda dirigir el timón de desarrollo de la Intranet.

b) Metas

Una vez que se cuente con la Declaración de Propósitos se necesita desarrollar metas de implementación más concretas u objetivos específicos para la información y los servicios que la Intranet proporcionará a los usuarios, por ejemplo:

- Ofrecer información sobre notas de alumnos.
- Proporcionar anuncios generales de la Universidad Católica de Ambato.
- Establecer un tablero de mensajes virtual, mediante el uso de formularios electrónicos.

Con las metas de implementación definidas, estas pueden ser trasladadas a conjuntos de tareas específicas con el fin de establecerlas y manejarlas. Pero a medida de que la Intranet empiece a funcionar, empezarán a surgir nuevas ideas que se podrán implementar; por lo tanto los planes no deben ser rígidos, deben tener un cierto margen para evolucionar.

1.1.3.8 DISEÑO Y DISPOSICIÓN DE UNA WEB

Con los propósitos y metas establecidos, hay que dedicarse al diseño y disposición de la Intranet. Esto se puede hace mediante dos partes relacionadas: lógica y física

a) Diseño Lógico

Este es el proceso de arreglar la información de la Intranet de acuerdo con un plan general. Esta tarea debe empezar con los temas principales colocados en cierto orden lógico, a menudo la información se desglosa de manera natural en fragmentos lógicos, de manera que estos se pueden reflejar estas divisiones en el diseño lógico. Basándose en los datos anteriores, un bosquejo lógico puede ser:

- Página de Alumnos de la P.U.C.E.S.A.
- Propósito

  Proporcionar información a los Alumnos
- Principales subdivisiones

  Reglamentos

  Información acerca de Notas y Trabajos

  Horarios de Cursos
- Servicios en General

Con esto ya se ha definido la página de los alumnos de la Universidad Católica de Ambato y siguiendo este orden lógico se pueden definir toda la Intranet en general. La jerarquía se presta para un diseño mas humano e intuitivo según los intereses, predilecciones y las necesidades de un usuario en un momento determinado; es por eso que es una buena idea permitir que los usuarios participen en ciertos aspectos de diseño y disposición.

b) Diseño Físico

Hay varias maneras de disponer los aspectos físicos de la Intranet, de hecho esto esta definido en el capitulo referente a la construcción de una Intranet. En el modelo centralizado existe un solo Servidor Web al cual solo tienen acceso sus administradores.

Todo se lleva a cabo en un solo sitio lo cual simplifica en gran medida la administración, mantenimiento, respaldos, y por supuesto, la seguridad del Servidor.

Es paradójico, pero el modelo descentralizado de Administración Web es más simple y sencillo de manejar. El administrador de la red no toma decisiones acerca de los sitios que se crean, los usuarios son libres de instalar Servidores, documentos e información de la Intranet.

1.1.3.9 EQUIPOS PARA INTRANETS

Para instalar una Intranet, basándose en los diseños lógicos y físicos, se deben escoger las máquinas idóneas; se necesitan como mínimo los siguientes equipos:

- Un servidor Web - HTTP
- Estaciones de Trabajo
- Routeadores
- Sistema de Red

a) Servidor Web - HTTP

Un servidor Web será el sustento de nuestra Intranet, este servidor debe cumplir con las características adecuadas para dar un servicio estable y funcional, permitiendo conexiones seguras para nuestros usuarios.

Existen varios tipos de servidores Web, disponibles, ordenados por plataforma, algunos son del dominio público, otros son comerciales, los cuales se describen a continuación:

- UNIX

- *NCSA httpd*

  Éste es el servidor más popular, genéricamente conocido como httpd. Ofrece las funciones básicas que la gente espera del Web, tales como la habilidad para servir documentos desde directorios virtuales, ejecutar scripts en el servidor, seguridad basada en direcciones IP o autenticación de contraseña, y la habilidad de generar listados de directorios sobre la marcha.

  Además, incluye un número de características únicas, notablemente la habilidad para poner scripts ejecutables en cualquier parte del árbol de documentos, y "Server side includes", esto es un sistema bajo el cual se pueden colocar claves, direccionamientos y pedazos de código ejecutable directamente dentro del documento Hipertexto con el objeto de cambiar su apariencia sobre la marcha.

- *Apache*

  Algunos grupos aprovecharon el estatus de dominio público del código del servidor NCSA para agregarle características o mejorar su desempeño. Los miembros del no comercial Apache HTTP Server Project le agregaron características útiles a NCSA http, incluyendo la habilidad de personalizar mensajes de error, usar scripts ejecutables como páginas de bienvenida, y

soporte para negociar el contenido para encontrar la representación de un documento según las preferencias del cliente. Además, permite crear múltiples "servidores virtuales" en sitios separados con sus propios nombres de servidor y árboles de documentos corriendo en el mismo servidor.

Este servidor es el más popular, siendo el más utilizado dentro del Internet, su punto fuerte radica en que su código es del dominio público y fácilmente se encuentran "parches" o actualizaciones del sistema en la Red.

- *EIT enhanced HTTP Server*

  Este servidor está basado en NCSA http 1.3. Le agregaron la habilidad de priorizar las peticiones, configurar los mensajes de error, volver a arrancar un servidor caído, y soporta periodos programados de tiempo fuera.

- *CERN*

  Éste es el servidor original de Web. Debido a que fue creado y mantenido en el hogar del Web, ha sido probado para más avanzado del protocolo Web. El resultado de esto es que está superarmado. Desdichadamente, esta sobrecarga de características lo hace más complejo para instalar y configurar, particularmente en lo que respecta a seguridad.

Este servidor tiene tres características principales que otros no tienen. La bprimera es que puede actuar como un Proxy a través de un Muro de Fuego. Muchos sitios han instalado sistemas Muro de Fuego para mejorar la seguridad de su red. El trabajo de los Muro de Fuego es impedir la conexión entre la red local dentro del Muro de Fuego y el Internet en el exterior, excepto cuando se trate de determinadas conexiones en las que sí se tiene confianza. A menudo el efecto lateral es que los usuarios dentro del Muro de Fuego no pueden usar sus navegadores de Web fuera.

Cuando se instala el servidor Proxy en la máquina Cortafuegos se resuelve el problema ya que el servidor actúa como el intermediario entre el interior y el exterior. Los navegadores dentro del Cortafuegos mandan peticiones de documentos al Proxy del servidor CERN en la máquina Cortafuegos. El servidor CERN reenvía la petición al verdadero dueño de la información. Por supuesto, esto solo funciona si tienen navegadores que soporten acceso Proxy, actualmente casi todos lo hacen.

La segunda característica propia del servidor CERN, disponible solo cuando se usa como Proxy, es que puede hacer cache de los documentos remotos en forma local. Cuando un navegador pide un documento que esté en el cache local, el servidor lo entrega de ahí, en vez de traer el documento remoto. Esto tiene un

gran beneficio en desempeño, sobre todo cuando los documentos son enviados por una conexión lenta, como el cable trasatlántico. La tercer característica es "negociación de contenido", soporte para múltiple representaciones de un documento. En ciertas circunstancias, el servidor CERN puede elegir sobre varias representaciones del mismo documento (incluyendo idiomas alternos y formatos de archivo) para encontrar el que el cliente prefiere.

- *Plexus*

Este servidor esta desarrollado en Perl, que es un lenguaje interpretado, a diferencia de la mayoría de los demás servidores, que están escritos en C. Plexus es algo así como un paquete de herramientas para el servidor, todo lo necesario para un servidor básico está en el paquete, pero algunas características, como son formas y seguridad solamente están esbozadas.

Sus características únicas son la búsqueda de nombres de archivos usando las expresiones regulares de Perl, habilidad para que un solo servidor escuche a muchos puertos, y soporte para configurar varios servidores en la misma máquina, cada uno con su propio nombre virtual de servidor y árbol de documentos. Además, provee soporte para comunicaciones seguras. comunicaciones encriptadas con versiones compatibles de Windows

- *GN*

Fue diseñado para ser un servidor de doble propósito. Soporta los protocolos Gopher y HTTP. Se presenta como servidor de Gopher para los clientes Gopher y como servidor Web para clientes Web. Los dos servicios comparten el mismo árbol de documentos y archivos de datos. GN es una buena opción para servidores en que se encuentren en transición de Gopher a Web.

- *WN*

Este servidor esta basado en GN Lo diferente en este servidor es que cada directorio del sitio contiene un archivo que es una base de datos plana listando los archivos que se servirían, e información como títulos, autores y palabras de búsqueda. La ventaja de esto es que permite al servidor hacer búsquedas rápidamente.

Una desventaja potencial es que las páginas no pueden ser vistas a menos que estén explícitamente mencionadas en la base de datos. Esto incrementa la seguridad pero lo hace menos fácil de mantener. Por razones de seguridad WN no genera listados de directorio sobre la marcha como CERN, NCSA y Plexus.

WN viene con un rico conjunto de herramientas de manipulación y creación de documentos Hipertexto, búsqueda por palabras clave, y la creación de índices y

tablas de contenido. Además, soporta scripts ejecutables y provee inclusiones de servidor estilo NCSA.

- *Netsite Communications Server*

Muchos miembros originales de los equipos NCSA Mosaic, NCSA httpd y CERN Server fundaron la compañía privada Netscape Communications Corporation. Ellos hicieron el Netsite Communications Server, éste es similar a NCSA httpd en características, pero es más rápido y mejor soportado. Su desempeño se mejora en condiciones de carga pesada, particularmente cuando se sirven a múltiples peticiones simultaneas generadas por el Netscape Navigator, que es el navegador de la misma compañía. Además, provee una forma más rápida de cargar y correr scripts ejecutables.

Este servidor mejora el conjunto básico con comunicaciones seguras utilizadas para negocios. Cuando se opera en modo seguro, toda la comunicación entre el navegador y el servidor se encripta usando un seguro sistema de criptografía basado en llave pública. Esto solo funciona cuando se trata con un navegador compatible como el Netscape Navigator versión 3.0 o el Microsoft Explorer versión 3.0 o mejores.

Además, la configuración se realiza mediante una interfase gráfica, a diferencia de los archivos de texto que se ofrecen normalmente para la configuración de los servidores de dominio público.

- *Open Market Web Server*

Este servidor agrega muchas características al conjunto básico de los servidores de dominio público. Entre ellas se encuentra la habilidad para escoger diferentes documentos para servir a los clientes basándose en combinaciones arbitrarias del nombre del cliente, su dirección IP, la hora del día, el contenido del encabezado HTTP, u otras condiciones. Además, ofrece control de acceso al nivel de archivo así como para directorios enteros.

Tiene interfase gráfica para instalación simple. Para algo más sofisticado requiere que el usuario edite los archivos de configuración.

- Macintosh

- *WebSTAR*

Este producto fue shareware en sus inicios, era MacHTTP para Macintosh y llegó a ser un producto comercial. Corre bajo sistema 7 o superior y soporta Macintosh basadas en 68000 y PowerPC. Como el software de Macintosh, la configuración se hace en cajas de dialogo gráficas en vez de los archivos de

configuración. Soporta scripts ejecutables usando el AppleScript lenguaje. Con MacPerl, los scripts ejecutables escritos para máquinas UNIX pueden correr con mínimas modificaciones.

- Microsoft Windows 3.1

- *Win-httpd*

  Comenzó como una aportación de NCSA httpd y gradualmente evolucionó en características propias. Implementa las principales características de NCSA httpd versión 1.3 para UNIX. Un módulo de compatibilidad permite que muchos scripts ejecutables incluyendo Perl, que originalmente fueron escritos para UNIX puedan funcionar con pocas modificaciones. Soporta scripts ejecutables basados en archivos batch de DOS y Visual Basic.

- Windows NT y Windows 95

- *HTTPS*

  Los usuarios de Windows NT pueden usar HTTPS, es un servidor gratuito del EMWAC (European Microsoft Windows NT Academic Centre), o Website, un servidor comercial creado por Enterprise Integration Technologics y vendido por O'Reilly and Associates. HTTPS soporta múltiples conexiones simultaneas, formas, scripts ejecutables e imágenes sensibles al ratón. La mayor limitación de este servidor es que no tiene control de acceso a directorios.

- *WebSite*

  Corre en Windows 95 así como Windows NT. WebSite ofrece una interfaz gráfica para configurarse, crear documentos e instalar scripts. Adicionalmente ofrece características únicas como la habilidad para correr Microsoft Excel, bases de datos relacionales y otras aplicaciones con OLE (objet linking and embedding) dentro del servidor.

- OS2

- *GoServe*

  Es un servidor gratuito para el sistema operativo OS/2. Corre en versiones 2.0 y superiores, así como OS/2 Warp. Como GN GoServe fue diseñado como servidor dual a peticiones Gopher y HTTP. Soporta scripts ejecutables escritos en el lenguaje de programación REXX. En los scripts disponibles están algunos que implementan control de acceso a directorios basado en direcciones IP y/o contraseñas.

b) Estaciones de Trabajo

Las estaciones de trabajo serán los medios por los cuales los usuarios accederán a nuestra Intranet. La principal característica de estos equipos será el permitir usar un sistema operativo, un navegador, una tarjeta de red o un módem. Aunque, para nuestra Intranet es importante la tarjeta de Red; si es que existen usuarios remotos podríamos

considerar que usen un módem, pero si permitimos a los estudiantes ingresar desde sus casas entonces permitir el acceso telefónico sería indispensable

Existen varios tipos de estaciones de trabajo en el mercado, pero hay dos tipos importantes: Compatibles con I.B.M. y Macintosh

- Compatibles con I.B.M.

Este tipo de computadoras es el mas conocido en nuestro medio, aproximadamente el 90% del mercado es cubierto por computadoras que son compatibles con el modelo inicial que IBM lanzó en 1981. Las características que debe llevar una estación promedio son:

Procesador Pentium

Velocidad 150 mhz.

Disco duro (mínimo 300 Mb.)

Memoria RAM 16 Mb

Monitor

Teclado

Ratón

Tarjeta de Red (Ethernet o Token Ring)

Opcionales : tarjeta de sonido, parlantes, impresora

Aunque estas son las estaciones típicas de un laboratorio en la Universidad, se pueden intentar un nuevo tipo de estaciones Java que se las puede administrar desde un sitio central y pueden ser fácilmente configuradas desde el servidor Web o el servidor primario del Dominio. Sus características estándares pueden ser:

Procesador Pentium

Velocidad 266 mhz.

No Disco duro

Memoria RAM 32 Mb

Monitor

Teclado

Ratón

Tarjeta de Red (Ethernet o Token Ring)

- Macintosh

Este tipo de computadoras es muy escaso en nuestro medio, pero debido a que una Escuela de la Universidad Católica los usa, debemos incrementar este tipo de computadoras en nuestra Intranet. Las características de un equipo compatible con nuestra red son:

Procesador y Memoria

350 o 400 Mhz Power PC G3

64 Mb de memoria

Monitor

Soporte de Gráficos

Disco Duro de mínimo 6 Mb.

Tarjeta de Red (incluida en la maquina)

1.1.4 CONSTRUCCIÓN DE UNA INTRANET

Para construir una Intranet se necesita considerar varios aspectos fundamentales y que sin ellos sería inútil el establecer un sitio Web.

Lo primero es establecer un espacio físico en donde vayan los Servidores, computadoras y equipos de comunicación de nuestra Intranet, tomando en cuenta las siguientes consideraciones:

- Un Laboratorio o aula en donde vayan ubicadas las computadoras. Este laboratorio será en donde todos los estudiantes de la Universidad Católica de Ambato podrán ingresar con libre acceso para usar las estaciones y navegar dentro de nuestra Intranet.
- Una oficina en donde estén ubicados todos los equipos de comunicación y nuestro Servidor Web. Es decir, será la oficina del Administrador de la Red, por obvias razones debe estar en un lugar seguro y fuera del alcance de personas no autorizadas, no es necesario que este muy lejos del Laboratorio pero sí que este dentro del mismo piso del Edificio.
- Una red de cableado, Token Ring o Ethernet, que permita la comunicación entre los Servidores y las computadoras.
- Líneas Telefónicas que permitan que nuestra Intranet se comunique con Internet, estas pueden ser dedicadas o conmutadas, depende del Administrador de la Red

decidir cual medio de comunicación será el adecuado para la Universidad Católica de Ambato.

Esto no debe impedir que nuestra Intranet pueda incluir o permitir accesos desde computadoras o Servidores que, físicamente, estén fuera la Universidad. Lo que aquí se recomienda es solo el esquema básico de la Intranet.

Pero es más fácil seguir un esquema en donde se vean todos los pasos para construir una Intranet:

Primero optaremos por instalar el Servidor, este contendrá toda la información disponible para la Intranet. El Administrador de la Red escogerá el Servidor más adecuado para las necesidades de la Universidad Católica de Ambato, para esto debe considerar los siguientes elementos:

a) Marca

Es importante que para escoger un Servidor, no se debe adquirir un equipo de una marca desconocida o "clon". Es preferible escoger una marca de equipos conocida y que brinde seguridad a los usuarios.

b) Procesador

En general un procesador rápido podrá ejecutar todas las tareas propias de un Servidor, además, podrá cumplir con todos los requerimientos de las estaciones y de los usuarios.

c) Almacenamiento Físico

Cuando de trata de almacenamiento físico es mejor escoger discos duros de gran capacidad, 6 u 8 gigabytes, mejor si el Servidor puede manejar sistemas de protección de información en caso de pérdidas o daños en los discos duros (Smart Array) y cambios de discos duros cuando estos están en funcionamiento (Hot Swap).

d) Memoria

Se necesita que el Servidor tenga al menos 64 megabytes de memoria en RAM (Random Access Memory).

e) Capacidad de Expansión

Es muy importante que el Servidor permita mejoras en la tarjeta madre y en el procesador, que su arquitectura pueda ser mejorada según pase el tiempo y la tecnología avance.

f) Soporte Técnico

Un Administrador de Red y Web debe tener un buen equipo de soporte interno y externo, los cuales permiten que en casos de emergencia el Servidor y la información puedan ser recuperados en el menor tiempo posible, además, que este Soporte brinde soluciones en caso de necesitarlas, soluciones tales como: problemas del año 2000, mejoramiento de hardware y software, adecuación de redes, etc.

Después de tomar en cuenta estas consideraciones, se debe escoger un Sistema Operativo que maneje el Servidor, para esto existen tres clases de Sistemas Operativos robustos:

- Windows NT
- UNIX
- Linux

Él escogerlo dependerá del Administrador de la Red basándose en el Hardware del Servidor y un muy estudiado informe de Costo / Beneficio de cada Sistema Operativo. El Sistema Operativo escogido será el que sirva de plataforma Operativa y de Control DNS del Dominio de la Intranet.

El protocolo *TCP/IP* es el centro de la Intranet. No es necesario que sea el único protocolo y, en muchos casos, las empresas utilizan *TCP/IP* sobre otros protocolos como *IPX* (Internet Packet Exchange) de Netware o NetBui de IBM. A pesar de esto, la mayoría de las aplicaciones Intranet necesitan TCP/IP. Hay dos opciones: Si sus clientes están ejecutando UNIX, OS/2, Windows for Workgroups, Windows 95 ó Windows NT Workstation, es posible ejecutar el paquete de software TCP/IP distribuido con estos sistemas operativos. Si se está ejecutando Netware para servicios de impresión y archivos y muchos de los estudiantes aún están ejecutando DOS o Windows 3.1 en sistemas de menor capacidad, podríamos considerar un Gateway IPX para IP entre su servidor Web y el resto de la red.

Seguidamente, es preciso adquirir el componente software fundamental de la Intranet: un servidor Web. Los servidores HTTP de libre distribución y versiones Shareware, protegidos por derechos de autor, generalmente ofrecen todo lo que se necesita para tener una Intranet operativa. Microsoft provee de un servicio gratis para crear un Servidor Web en un Servidor Windows NT; este servicio se llama Internet Information Service (IIS), que provee de las funciones básicas de un Servidor Web, es muy útil para una Intranet pequeña como la de la Universidad Católica de Ambato; pero no ofrece mucha seguridad cuando el Servidor esta conectado a Internet y puede ser susceptible de ataques de personas externas a nuestra Institución.

Los servidores comerciales de gama media incorporan herramientas de monitorización y mantenimiento del Web mientras que los de gama alta, implementan excelentes

medidas de seguridad, recursos de criptografía para transacciones seguras y hasta enlaces con bancos de datos corporativos.

El control de acceso al servidor Web puede ser de dos tipos. Estos normalmente se definen distintos niveles de acceso caracterizados por los permisos otorgados sobre la información contenida en el servidor. Este tipo de control es similar a la distribución de derechos para usuarios de la red. Esto es lo que se llama filtrado de IP o nombre de huésped, en virtud del cual algunas direcciones IP poseen los permisos necesarios para acceder a determinadas páginas en el servidor. Además, existe también un método de verificación al nivel de usuario o acceso controlado por contraseña a las páginas Web.

Si todavía no se lo ha hecho, ahora es el momento de determinar el tipo de información que desea compartir en la Intranet. Una de las tareas más importantes y más difíciles de planificar inicialmente es la estructuración del contenido y la elección del criterio de navegación en la Intranet.

Algunas veces, al proyectar el diseño de páginas Web complejas, es interesante tener en mente una idea clara de la estructura del modelo del documento impreso. Por ejemplo, tal vez se quiera que su aplicación sea semejante a un catálogo, una revista o un libro de consulta. Cada aplicación puede tener un propósito diferente cuya estructura ha de estar perfectamente estudiada y desglosada en un organigrama antes de ser codificada en un lenguaje de Hipertexto.

Una vez planificado y estructurado el contenido real, debe ser preparado para su *almacenamiento en la Web*. Esto significa que debe ser codificado con un lenguaje de programación de Hipertexto o usar alguna herramienta automatizada diseñada para tal fin. Este almacenamiento o publicación debe ser realizada por el equipo de Administración de la Red, así se garantizará que la información publicada este de acuerdo con las normas y estándares de la Universidad Católica de Ambato; todas las páginas Web, así como la administración y configuración del Servidor se llevan a acabo en un solo lugar, lo cual simplifica en gran medida la administración, mantenimiento, respaldos del sistema y la seguridad del Servidor.

Es posible crear documentos con lenguaje HTML directamente, emplear herramientas de traducción de documentos en otros formatos a HTML, utilizar un sistema automatizado de generación de código o usar un banco de datos de documentos existentes que ofrece una traducción HTML automatizada en tiempo real para usuarios Web. Para cumplir con este objetivo tenemos que tomar en cuenta las siguientes herramientas:

- Herramientas que convierten datos electrónicos de documentos heredados (documentos que residen en el Servidor o en PC´s) en documentos HTML
- Herramientas para crear, modificar o convertir datos en gráficos

- Herramientas CGI para interactuar con el Servidor a través de una Interfaz de Gateway Común.
- Herramientas de Correo Electrónico compatibles con el estándar MIME.
- Navegadores que permitan leer la información preparada con las herramientas anteriores.

1.1.4.2 HERRAMIENTAS PARA EL DESARROLLO DE LA INTRANET

Existen varias herramientas en el mercado y en Internet que nos pueden ayudar a construir nuestra Intranet. Varias de estas herramientas son gratuitas y otras son parte de los paquetes comerciales. Depende de nosotros decidir cuales podemos utilizar.

a) Navegadores Web

Los navegadores son la pieza clave del software de las computadoras de los usuarios de Intranet, permiten que el usuario pueda navegar a través de las páginas y utilizar las aplicaciones.

Producto	Organización	Sitio Web
Cello	Cornell University	www.law.cornell.edu
CyberDog	Apple	www.apple.com
HotJava	Sun	www.sun.com
Internet Explorer	Microsoft	www.microsoft.com

Producto	Organización	Sitio Web
Mosaic	NCSA	www.ncsa.uiuc.edu
Netscape Navigator	Netscape	www.netscape.com
PowerBrower	Oracle	www.oracle.com

b) Herramientas de Edición H.T.M.L.

Estas herramientas son importantes para la creación de los sitios Web. El administrador de la Red puede escoger una o dos de estas herramientas para crear el sitio principal Intranet; pero si la Intranet es desarrollada por un equipo o los usuarios, estos pueden escoger sus propios programas de desarrollo.

Es importante que estas herramientas sean compatibles con los lenguajes de programación de Internet como Java, Perl, visual Basic, etc., y, además, puedan manejar los agregados comerciales para gráficos, sonidos, video, animaciones, etc.

Producto	Organización	Sitio Web
Claris Home Page	Claris	www.law.cornell.edu
Domino	Lotus	www.lotus.com
Front Page	Microsoft	www.microsoft.com
HotMetal pro	Soft Quad	www.sq.com
Netscape Composer	Netscape	www.netscape.com

Producto	Organización	Sitio Web
Site Mill	Adobe	www.adobe.com
Word Internet Assistant	Microsoft	www.microsoft.com

c) Servidores Web, de Noticias y de Correo

Dependiendo del tipo de servidor (máquina) que el Administrador de Red haya escogido para manejar la red en conjunto, podemos escoger entre los siguientes productos para desarrollar el sitio Intranet:

Producto	Organización	Sitio Web
Apache Http Server	Apache	www.apache.org
Commerce Server	Netscape	www.netscape.com
Domino	Lotus	www.lotus.com
Enterprise Server	Netscape	www.netscape.com
Exchange Server	Microsoft	www.microsoft.com
Internet Information Server	Microsoft	www.microsoft.com
Linux	Linux	www.linux.com
Ms Mail Server	Microsoft	www.microsoft.com
Netscape Mail Server	Netscape	www.netscape.com

Producto	Organización	Sitio Web
Netscape News Server	Netscape	www.netscape.com
Netware Web Server	Novell	www.novell.com
Proxy Server	Netscape	www.netscape.com
Solstice Mail Server	Sun	www.sun.com

d) Herramientas de Desarrollo Web / Intranet

Estas herramientas sirven para unir todos las páginas, aplicaciones, documentos desarrollados por los usuarios. Estas herramientas sirven para establecer una relación coherente entre el software - servidor y el software – cliente:

Producto	Organización	Sitio Web
BackStage	Macromedia	www.macromedia.com
Borland C++	Borland	www.borland.com
Café	Symantec	www.symantec.com
Connection for Java	Open Horizon	www.openhorizon.com
Intrabuilder	Borland	www.borland.com
Interact	ProtoView Developers	www.protoview.com
Visual J++	Microsoft	www.microsoft.com
Java Work Shop	Sun	www.sun.com

Producto	Organización	Sitio Web
Neuron Web Element	Neuron Data Systems	www.blazesoft.com
Orbix	Iona Technologies	www.iona.com
SniFF	Take Five Software	www.takefive.com
Tango	EveryWare Development	www.everyware.com
Visual Edge	IBM	www.ibm.com
WebObjects	NeXT Software	www.next.com

e) Herramientas para Interconexión con Bases de Datos

Estas herramientas son importantes para conectarse con las bases de datos de los servidores de la Red, sin importar si son parte o no de la Intranet. Casi todas las aplicaciones Intranet se basan en interacciones con Bases de Datos centralizadas.

Producto	Organización	Sitio Web
Access Internet Asistant	Microsoft	www.microsoft.com
Cold Fusion	Allaire	www.allaire.com
DB2 World Wide Web Connection	IBM	www.software.ibm.com
DbWeb	Microsoft	www.microsoft.com
IDBC Sql Sever	Microsoft	www.microsoft.com
Informix 4GL CGI Interface	Informix	www.informix.com

Producto	Organización	Sitio Web
Kit		
Oracle Web Server	Oracle	www.oracle.com
Sybase + Web SQL	Sybase	www.sybase.com

f) Aplicaciones de Correo Electrónico

Estas aplicaciones sirven para que los usuarios puedan acceder a su correo interno y externo de Internet.

Producto	Organización	Sitio Web
Chamaleon	Netmanage	www.netmanage.com
Eudora	Qualcomm	www.qualcomm.com
Exchange	Microsoft	www.microsoft.com
Outlook Express	Microsoft	www.microsoft.com
Netscape Messanger	Netscape	www.netscape.com
Pegasus Mail	Pegasus	www.pegasus.usa.com

1.1.5 USUARIOS DE INTRANET

La definición de las personas que utilizarán la Intranet de la Universidad Católica de Ambato es muy diferente de las personas que usan el Web público de una organización, cuyo enfoque principal es mostrar información a *gente del exterior* y dicha información es completamente general y valiosa para las relaciones públicas. Por otro lado el enfoque de una Intranet es hacia las personas de interior o estrechamente vinculadas con la Institución: Estudiantes, profesores y administradores de la Universidad Católica de Ambato. Se podrá acceder a información acerca de los departamentos, actividades, horarios, listados, etc.

Cuando se empiece a ponderar el diseño de la Intranet, su primera consideración debe definir con claridad la audiencia que pretende, es decir, sus usuarios. El propósito fundamental de la Universidad Católica de Ambato es la educación e investigación, y sus clientes primarios son estudiantes, profesores e investigadores, los cuales son miembros de la Institución; en este caso sus usuarios son las personas dentro de la Institución quienes elaboran y usan los servicios del Intranet.

Estas son distinciones cruciales que se deben tener en cuenta al considerar y diseñar su Intranet. La decisión sobre la manera de diseñarla y la información que almacenará debe basarse en su audiencia objetivo: sus usuarios internos.

1.1.5.1 PLANEACIÓN Y CREACIÓN DE USUARIOS Y GRUPOS DE TRABAJO

El departamento de sistemas de la Universidad Católica de Ambato es el encargado de determinar las políticas de planeación y creación de usuarios y grupos de trabajo en el Intranet. Estas políticas deben ir acordes con las políticas de usuarios del Sistema Operativo, en realizada son casi las mismos, lo que se debe determinar en este caso es la responsabilidad de la configuración y el diseño de la Intranet. Con las últimas técnicas para la creación de páginas HTML cualquier usuario con una PC disponible debe estar en la capacidad de crear una página Web y ponerla a disposición de todos.

Se deben crear los siguientes tipos de usuarios en la red:

- Administrador de Red

  Es el encargado de la Seguridad General, se necesita uno para manejar todos los recursos de los servidores en el ámbito de la Universidad ya que es suficiente para manejar una red pequeña.

- Operadores de Red

  Los operadores de red son ayudantes del Administrador de la Red, estos pueden manejar ciertos servicios y recursos críticos de la red como impresoras o recursos compartidos. Los operadores son los creadores de las nuevas cuentas de usuarios de la red.

- Usuarios administrativos

Son los empleados de la Universidad, los cuales tendrán un nombre de usuario asignado, tendrán privilegios tales como recursos compartidos, programas de la universidad y correo electrónico privado.

- Usuarios estudiantes

  Son los estudiantes de la universidad, solo tendrán derecho a entrar a la Intranet de la universidad y a usar correo electrónico. El acceso a ciertos recursos será temporal dependiendo de las necesidades en ciertas materias.

- Visitantes

  Los visitantes tendrán solo acceso de lectura a Intranet

1.1.5.2 AUTENTICACIÓN DE USUARIOS

Autenticación significa simplemente confirmar la identidad aparente de un usuario que solicita un servicio a través de una red. La mayoría de los servidores usa una dirección IP autenticada para identificar al usuario.

Existen tres métodos tradicionales para verificar la identidad de una persona:

- El primer método utiliza "algo conocido", que puede ser un elemento de información secreto como una contraseña
- El segundo método identifica a una persona como "algo que posee ésta", como una llave de una cerradura o una tarjeta inteligente.

- El tercer método emplea "algo incorporado", como una huella dactilar o el modelo de la retina de una persona.

En nuestro caso, vamos a utilizar el primer método ya que es el más conocido y el más sencillo. Utilizaremos los conocimientos de los usuarios para que ellos mismos puedan entrar a nuestra Intranet, se les proveerá de un nombre de usuario y ellos ingresarán una clave secreta que solo ellos conocerán.

a) Reglas de nombres de usuarios

Cuando se va a crear por primera vez un nombre de usuario se debe pensar en dos casos:

- Usuario Administrativo o Empleado
- Usuario Estudiante

Cuando es un empleado administrativo se deben seguir los siguientes pasos para crear el usuario en el servidor:

- Tomar la primera letra del primer nombre del empleado
- Tomar todo el apellido del empleado hasta completar ocho caracteres junto con el nombre anterior
- En el caso de duplicación de un nombre de usuario entonces tomar la primera letra del segundo nombre e intercalarla entre las cadenas de caracteres obtenidas de los dos pasos anteriores.

Ejemplo:

Nombre del Usuario	:	Xavier Francisco López Andrade
Usuario Asignado	:	xlopez
Nombre del Usuario	:	Andrés Ruben López Andrade
Usuario Asignado	:	alopez
Nombre de Usuario	:	Xavier Enrique López Peralta
Usuario asignado	:	xlopez (duplicado y no asignado)
		xelopez (modificado y asignado)
Nombre de Usuario	:	Luis Enrique Zambrano Peralta
Usuario asignado	:	lzambran

Si es un usuario estudiante se deben seguir los siguientes pasos para crear el usuario en el servidor:

- Tomar la primera letra del primer nombre del estudiante
- Tomar todo el apellido del estudiante hasta completar seis caracteres junto con la letra anterior
- Tomar el último dígito del año de la matrícula del estudiante y añadirlo a la cadena anterior

- Incluir "A" si es el primer semestre y "B" si es el segundo semestre de ese año y completar 8 caracteres en el usuario
- En el caso de duplicación de un nombre de usuario entonces tomar la primera letra del segundo nombre e intercalarla entre las cadenas de caracteres obtenidas entre los pasos 1 y 2.

Ejemplo :

Nombre del Estudiante	:	Xavier Francisco López Andrade
Año de matrícula	:	1997
Semestre	:	Octubre 1997 – Febrero 1998
Usuario	:	xlopez7B

Nombre del Estudiante	:	Andrés Ruben López Andrade
Año de matrícula	:	1993
Semestre	:	Marzo 1993 – Julio 1993
Usuario	:	alopez3A

Nombre de Usuario	:	Xavier Enrique López Peralta
Año de matrícula	:	1997
Semestre	:	Octubre 1997 – Febrero 1998
Usuario asignado	:	xlopez7A (duplicado y no asignado)
		xelope7A (modificado y asignado)

Todos estos usuarios deben ser creados en el servidor principal de la Intranet y, además, deben tener un documento de respaldo firmado que se archivara en un lugar seguro. Si el empleado se retira de la Universidad entonces el usuario será borrado y nunca mas se podrá utilizar un usuario igual para otra persona o empleado, pero si en el caso de que el empleado regresara entonces el usuario será reactivado. En el caso de un estudiante, si este no regresa en el siguiente semestre entonces la clave será borrada y podrá ser reutilizada con otro estudiante.

b) Contraseñas

Las contraseñas deben ser propias de cada usuario, se deben establecer las siguientes reglas para las contraseñas:

- La contraseña será secreta.
- La contraseña debe contener entre 6 y 10 caracteres de extensión.
- Los caracteres de la contraseña deben ser letras, números o una combinación de los dos; no se aceptan los demás caracteres del código ASCII.
- No se pueden aceptar contraseñas tales como cadenas de letras o números seguidos tales como abcdef, 123456, 987654, etc.
- La contraseña no puede repetirse sino hasta después de 3 contraseñas previamente utilizadas.
- El tiempo de duración de una contraseña será de 30 días.

- Cualquier violación de seguridad causada por utilización de una contraseña mal utilizada será de exclusiva responsabilidad del usuario.
- En caso de pérdida se solicitará una nueva clave al Administrador de la Red o en su defecto a un operador de Red previamente autorizado.

Cuando se registra un nuevo usuario, se debe elaborar un formulario([1]) en donde se indica toda la información necesaria para el ingreso a la red. Esta solicitud debe ser autorizada por el jefe del empleado o por el director de la Escuela en caso de los estudiantes.

Cuando el formulario llegue a manos del Administrador de Red u Operador asignado para estas funciones, este debe ingresar toda la información, asignar los recursos disponibles y como último paso asignar una contraseña; la contraseña asignada será el mismo nombre de usuario, con la condición de que el sistema pedirá un cambio de contraseña al primer ingreso del usuario en la Red.

En el caso de reemplazo de una contraseña, el usuario deberá firmar un registro de cambio de contraseña, antes de recibir una nueva contraseña. Y se le asignara una nueva contraseña que tendrá una validez de un día, tiempo suficiente para que el usuario cambie su contraseña.

---

[1] Ver un ejemplo de esta solicitud en los anexos

1.1.6 POLÍTICAS DE ADMINISTRACIÓN DE LA RED

En conformidad al reglamento orgánico de la Universidad Católica del Ecuador – Sede Ambato, corresponde a la Unidad de Ingeniería de Sistemas administrar la red central y sus servicios. Esta función incluye los aspectos de: planificación técnica, operación y control de su funcionamiento, desarrollo de proyectos asociados a ella, elaboración y proposición de políticas, representar a la Universidad ante organismos de carácter técnico en relación con su temática, definir y administrar los servicios que a través de la red puedan ponerse en funcionamiento.

1.1.6.1 POLÍTICAS DE INTERCONEXIÓN

La red central corresponde a la interconexión entre los diferentes edificios del campus central, incluyendo la conexión al edificio de la Universidad, en el centro de Ambato; a la sede universitaria del Tropezón y la comunicación hacia el exterior. La administración de las redes locales externas a la Unidad de Ingeniería de Sistemas, será responsabilidad de sus propios usuarios; excepto en aquellos aspectos normativos de diseño e instalación, que deberán contar con la autorización de la Unidad de Ingeniería de Sistemas, a objeto de asegurar el cumplimiento de estándares y normas para que su interconexión no provoque problemas.

La manutención del equipamiento de red existente en cada edificio, que ha sido provisto en forma centralizada, será de responsabilidad del organismo correspondiente.

Cada estación conectada a la red tiene una dirección única (dirección IP), y no está permitido cambiar dicha dirección sin autorización de la unidad administradora. La Unidad de Ingeniería de Sistemas administra la red de la universidad, 10.10.0.0. y asigna rangos de redes para su administración local.

El equipamiento computacional de la Universidad debe tener nombres bajo el dominio PUCESA.EDU.EC o de los sub-dominios que correspondan. No se permite el cambio de nombres ya que implica un intento de cambio de identidad.

1.1.6.2 POLÍTICAS DE USO DE LOS SERVICIOS DE RED

El uso de la red debe ser con fines universitarios o personales, sin propósitos comerciales a menos que exista un consentimiento expreso de la Universidad.

Se debe respetar la propiedad del software, según se establece en las leyes ecuatorianas.

El uso de cuentas es personal, por lo que no se debe utilizar cuentas de terceros, ni tampoco autorizar a terceros a utilizar las cuentas personales.

Se debe respetar la privacidad de la información y las cuentas, por lo que infringe las normas cualquier intento de suplantación o violación de la seguridad. Al mismo tiempo, el envío y recepción de mensajes es de carácter privado y la Unidad de Ingeniería de Sistemas no está facultada para accesar esta información.

Los usuarios se deben identificar apropiadamente, y se considera fuera de estas normas el cambio de identidad.

Infringe estas normas cualquier intento de interferencia o daño al equipamiento, servicio y/o usuarios. Esto incluye entre otros, la propagación de virus computacionales, la apropiación de información privada, el molestar a otros usuarios y la generación de sobrecarga.

La publicación de información en el WEB de los organismos debe estar coordinada por el encargado asignado por cada unidad.

La publicación de información en el WEB no puede ser de carácter comercial, a menos que exista un expreso consentimiento de la Universidad.

No se permite la publicación de información y/o imágenes que atenten contra la moral.

1.1.6.3 POLÍTICAS DE USOS INACEPTABLES DE BÚSQUEDA EN INTERNET.

El acceso a Internet a través de las máquinas localizadas en la Unidad de Ingeniería de Sistemas requiere que todos los USUARIOS se adhieran a los siguientes lineamentos de USOS INACEPTABLES de Internet:

- Utilizar correo electrónico, chats, Internet, Mirc con contenidos no académicos.
- Visualización de materiales sexualmente explícitos sexualmente, pornográficos y / o obscenos.
- Manejo de lenguaje obsceno, abusivo o sexualmente explícito.
- Materiales que fomenten delincuencia o daño en propiedad ajena.
- Acceder a materiales de otras personas y / o información o archivos sin permiso.
- Violentar las leyes de derechos de autor o propiedad intelectual sin previa autorización y / o sin la utilización de citas adecuadas.
- Utilizar los recursos de Internet proporcionados por la escuela para actividades comerciales o de lucro, proselitismo político.
- Utilizar expresiones de racismo.
- Utilizar el equipo en actividades / juegos no académicos que propicien el desgaste de los recursos de la escuela.

Ser usuario de la Universidad Católica del Ecuador – Sede Ambato es un PRIVILEGIO, la lista anterior de usos inaceptables no es exhaustiva. Un usuario que viola los términos y condiciones de uso de Internet e Intranet o comete otros actos de conducta no deseada y no listada pero considerada como inapropiada en el uso de los recursos de la Universidad Católica del Ecuador – Sede Ambato, estará sujeto a las sanciones que van desde suspensión del servicio, acceso restringido, restitución del

daño, suspensión de la escuela de manera temporal o suspensión definitiva. La violación de algunas de las prohibiciones listadas puede resultar en: Acceso restringido a Internet, pérdida de cuenta, acción disciplinaria o legal.

1.1.6.4 POLÍTICAS DE SEGURIDAD

Una política de seguridad (o acceso) consiste en la especificación de los requisitos de control de acceso a la información, equipos y otros activos de la Universidad. Estos requisitos de control determinan qué activos son accesibles, es decir, leer, modificar, eliminar, descargar, y por quién.

a) Tipos de Políticas de Seguridad

Se deben considerar dos tipos de políticas de seguridad: Alto nivel y Bajo nivel

Políticas de Alto Nivel se considera de nivel más elevado y define los requisitos de control de acceso desde la perspectiva del usuario. Establece el resultado final deseado de distintas peticiones de control de acceso a activos de información por parte de los usuarios.

Políticas de Bajo Nivel es una política de nivel inferior que define la forma en que una protección de seguridad específica pone en práctica el control de acceso. En general es

una respuesta a una necesidad de una política de alto nivel, por lo que ambas políticas deben mantener una consistencia.

Siempre estas políticas conllevan una mala reputación y establecerlas producen incidentes y roces entre los usuarios. Implica, además, reglas burocráticas que dificultan las tareas normales de los usuarios, pero que establecen las normas básicas para las operaciones cotidianas de la Red.

b) Creación de Políticas de Seguridad

Cuando se quieren crear políticas de seguridad, es necesario hacerse ciertas preguntas, aplicándolas siempre a cuestiones más específicas de nuestro entorno local. Si se desean establecer políticas de alto nivel las preguntas son muchas, pero que en general pueden ser:

Activos	Servicios	Compartimentos	Puntos de Entrada
¿Qué activos están sometidos a un riesgo en la red local?	¿Qué servicios pueden poner en peligro los activos de la red?	¿Existen compartimentos especializados a nivel local?	¿Están documentados todos los puntos de entrada?
¿Es la divulgación de la información una cuestión importante a nivel local?	¿Qué servicios de entrada son necesarios?	¿Se gestiona la información delicada?	¿Pueden los usuarios llamar a la red local?
¿Es la modificación de la información	¿Qué servicios de salida son	¿Forman parte los compartimentos de	¿Existen actualmente

Activos	Servicios	Compartimentos	Puntos de Entrada
una cuestión importante a nivel local?	necesarios?	seguridad de la estructura de la organización?	conexiones con Internet?
¿Existen requisitos en tiempo real?	¿Puede predecir cuáles serán las necesidades futuras de servicio?	¿Se precisan subredes internas aisladas?	¿Qué protocolos pueden entrar en la red local?

Tabla 1 Creación de Políticas de Seguridad

Para políticas de bajo nivel, es importante centrarse en detalles de implementación para cada entorno específico:

Rendimiento	Auditoría	Administración	Filtrado
¿Puede aumentarse el rendimiento del tiempo de proceso? ¿Cómo se mide?	¿Cuáles son los eventos críticos para la seguridad en el entorno?	¿Cuál es el enfoque de administración propuesto?	¿Cuáles son los requisitos para el Proxy?
¿Cómo se responde a las necesidades de mayor rendimiento de los usuarios?	¿Qué información debe auditarse?	¿Son necesarios protocolos de encriptación?	¿Qué información se precisa para el filtrado y los servicios Proxy?

Tabla 2 Políticas de Seguridad de Bajo Nivel

Es obvio que estas políticas deben ser puestas por escrito, debido a que muchas veces se confunden las políticas y, además, pueden tomarse a la ligera sin que sean efectivamente establecidas.

c) Requisitos para Políticas de Seguridad

Para poder desarrollar un entorno adecuado para las políticas de seguridad, el Administrador de la Red deben tomar en cuenta el conjunto de necesidades de la Universidad, esperando que este entorno no sea rígido y pueda ser aplicado sin que afecte las condiciones normales de los usuarios de la red.

A continuación vamos a ver ciertos requisitos para los servicios descritos en el capítulo de los *Cortafuegos* y que pueden servirnos para nuestra Intranet

- *Requisitos de entrada*

Es necesario determinar cuales servicios de entrada deben permitirse y cuales denegarse, pueden determinarse que las entradas de Telnet sean negadas y las de correo electrónico sean permitidas. Todo depende de las necesidades de la Universidad.

- *Requisitos para los servicios de Salida*

Permitir la salida a Internet se puede considerar de poca importancia, pero esto no es cierto; Muchas veces estos son los servicios que reducen la velocidad de nuestra Red y que, fácilmente, pueden producir los famosos "embotellamientos" y "caídas de la red". Es decir, que estos servicios deben ser tomados con precaución y deben estar en concordancia con las necesidades de los estudiantes y empleados administrativos de la Universidad.

- *Requisitos de Auditoría*

Siempre las decisiones sobre lo que debe o no debe ser auditado están sujetas a condiciones de rendimiento de la red. Muchos de los sistemas auditores provocan que la velocidad de conexión de nuestra Intranet se reduzca dramáticamente. En todo caso esto debe ser hecho con un proceso formal y no con un análisis rápido e impreciso.

d) Ejemplos de Políticas de Seguridad

Los siguientes son ejemplos de políticas de seguridad que pueden tomarse para la Intranet de nuestra Universidad, pero estos son solo ejemplos de lo que debe decidirse en un comité o grupo Administrador de la Red.

- *Políticas para Correo Electrónico*

Correo de Entrada

- Se permite el correo procedente de usuarios externos no identificados de Internet a los usuarios de Intranet
- Se prohibe el correo procedente de sitios pornográficos de Internet hacia los usuarios de Intranet.
- Se limita el tamaño de los correos entrantes a máximo 2 megabytes, cualquier correo sobre este tamaño será rechazado.

Correo de Salida

- Se permite el correo electrónico de los usuarios de Intranet hacia Internet.

- Se limita el tamaño de los correos salientes a máximo 2 megabytes, cualquier correo sobre este tamaño será rechazado.

- *Políticas para World Wide Web*

- Se prohiben las sesiones de WWW de usuarios externos no identificados hacia los programas internos de Intranet.
- Se permiten las sesiones de WWW de los estudiantes, profesores o empleados administrativos hacia los programas internos de Intranet.
- Se permite las sesiones de WWW de usuarios externos no identificados hacia www.pucesa.edu.ec
- Se prohiben las sesiones de WWW de usuarios internos hacia sitios pornográficos de la Red
- Se restringen las sesiones de comercio electrónico de usuarios internos, se debe solicitar por escrito una autorización del grupo Administradores de la Red.

- *Políticas de Servicios de Entrada*

- No se deben permitir solicitudes de Telnet de entrada, se pueden permitir sesiones restringidas si es autorizado por el Administrador de la Red.
- Se prohiben las sesiones de entrada de FTP hacia Intranet.
- Se prohiben sesiones de entrada de Finger hacia Intranet.

- *Políticas de Servicios de Salida*

  - Se permite solicitudes de Telnet de salida
  - Se permiten sesiones de FTP hacia el Internet, pero en máquinas con programas Antivirus activado.
  - Se permiten sesiones de Finger hacia Internet
  - Se permiten sesiones de Finger hacia InternetLenguaje H.T.M.L. y Páginas Web

1.2 LENGUAJE H.T.M.L. Y PÁGINAS WEB

1.2.1 GUÍA DEL LENGUAJE H.T.M.L.

1.2.1.1 INTRODUCCIÓN

H.T.M.L. es el lenguaje estándar para los navegadores de la Red Internet, este lenguaje es usado para que un usuario a través de un navegador o explorador pueda ver las paginas de Internet o Intranet en su computadora.

A través del H.T.M.L. se pueden ver documentos o información que se encuentra almacenada en un servidor Web que puede estar en la Intranet de la Universidad o para el público en general en la Internet.

1.2.1.2 DEFINICIÓN

H.T.M.L. que significa Hyper Text Mark-up Language (Lenguaje de Marcadores y Enlaces de Hipertexto), es un lenguaje de descripción de página y a su vez es un sistema para estructurar documentos. Contiene a su vez el texto a visualizar y las marcas que permiten dar formato en la ventana del navegador Web. Es un lenguaje muy sencillo que permite describir Hipertexto, es decir, texto presentado de forma estructurada y agradable, con *enlaces* (*Hipervínculos*) que conducen a otros documentos o fuentes de información relacionadas, y con *inserciones* multimedia (gráficos, sonido...) La descripción se basa en especificar en el texto la estructura lógica del contenido (títulos,

párrafos de texto normal, enumeraciones, definiciones, citas, etc.) así como los diferentes efectos que se quieren dar (especificar los lugares del documento donde se debe poner cursiva, negrita, o un gráfico determinado).

Estos documentos o textos pueden ser mostrados por los navegadores de paginas Web en Internet, como Netscape, Mosaic o Microsoft Explorer. El escoger un navegador o navegador depende del tipo de máquina y el sistema operativo que el usuario este manejando.

1.2.1.3 VERSIONES

Existen varias versiones de H.T.M.L. en el Internet, cada versión se ha mejorado las instrucciones y se han añadido nuevas características, todo eso dependiendo de los estándares que las grandes compañías de software han establecido para sus navegadores. Las versiones más populares son:

a) Versión H.T.M.L. 2.0

Cuando se produjo la explosión de Internet el estándar de HTML que circulaba era el 2.0 (establecido en noviembre de 1995), de modo que cualquier navegador que se utilice será capaz de leerlo. Esta versión ya esta caduca y solo sirve para ciertos casos de computadoras obsoletas o versiones antiguas de sistemas operativos

b) Versión H.T.M.L. 3.0 y 3.2

Aunque la versión 2.0 cumplía bien el objetivo para el que se creó, carecía de herramientas para tener un control mínimamente complejo de los documentos. No se consideró necesario que lo tuviera, ya que por aquel entonces Internet era un fenómeno más bien circunscrito a la actividad académica y el contenido predominaba sobre el diseño. Debido a ello, Netscape (líder del mercado de navegadores por aquel entonces) empezó a incluir etiquetas nuevas no incluidas en ningún estándar.

Por estos problemas, el IETF (el comité que suele decidir todos los estándares dentro de Internet) comenzó a elaborar el borrador del HTML 3.0, que resultó ser demasiado grande para la época, lo que dificultó su aceptación en el mercado.

Esto llevó a una serie de compañías (entre ellas Netscape, Microsoft, IBM, Sun, etc.) a crear un nuevo comité llamado W3C, Consorcio de la Red mundial de Internet, que es el que actualmente elabora las nuevas versiones del HTML. Su primer trabajo consistió en crear el borrador del HTML 3.2, que incluía muchas de las mejoras que los principales navegadores (Netscape y Explorer) habían introducido en Internet, como son las tablas, los applets, etc.

Este borrador fue aprobado en enero de 1997 e inmediatamente el W3C se puso a trabajar en la elaboración del siguiente estándar: el 4.0.

c) HTML 4.0

En julio de 1997 se presenta el borrador de este estándar. Por fin se estandarizan los marcos (*frames*), las hojas de estilo y los *scripts* (entre otras cosas). El 17 de diciembre de 1997 dicho borrador fue finalmente aprobado.

Ahora este es el nuevo estándar y solo las versiones mas nuevas de los navegadores lo soportan.

1.2.1.4 ESTRUCTURA BÁSICA DE H.T.M.L.

H.T.M.L. se compone de una estructura formada por directivas escritas en formato texto que están incluidas en un archivo, el cual puede ser hecho en cualquier programa editor de textos. El lenguaje HTML se basa en la sintaxis SGML (toma siglas). Esto quiere decir que cualquier directiva que hagamos en HTML estará encerrada entre dos etiquetas "<" y ">" que distinguen a estas del texto normal; Siempre cada directiva tiene una directiva de inicio *<directiva_de_inicio>* y otra de finalización *</directiva_de_finalización>*.

A veces es necesario ofrecer datos adicionales en una directiva. Por ejemplo, cuando se define un *hipervínculo* hay que especificar su destino. Para ello se incluyen parámetros en la directiva inicial (la de apertura), de la siguiente forma: `<directiva_de_inicio parametro1 parametro2 ...>`. La directiva de cierre, en caso de ser necesaria, queda como antes: *< directiva_de_finalización >*.

Un típico documento H.T.M.L. esta formado así:

Documento H.T.M.L.

Encabezamiento

Cuerpo

Fin del Documento H.T.M.L.

O en su forma escrita en al documento:

```
<HTML>
<HEAD>
Apariencia General del Documento H.T.M.L.
</HEAD>

<BODY>
Texto del documento, menciones a gráficos, enlaces, etc.
</BODY>
</HTML>
```

Un documento H.T.M.L. comienza con la etiqueta `<HTML>`, y termina con `</HTML>`. Dentro del documento (entre las etiquetas de principio y fin de html), hay dos grupos bien diferenciados: el *encabezamiento*, delimitado por `<HEAD>` y `</HEAD>`, que sirve para definir diversos valores válidos en todo el documento; y el *cuerpo*, delimitado por `<BODY>` y `</BODY>`, donde reside la información del documento.

a) Encabezamiento

La utilidad más importante del encabezamiento es la directiva <TITLE>, que permite especificar el título de un documento HTML. Este título no forma parte del documento en sí: no aparece, por ejemplo, al principio del documento una vez que este se presenta con un programa adecuado, sino que suele servir como título de la ventana del navegador que nos la muestra. Por ejemplo, en el encabezamiento de una página Web cualquiera se puede escribir la siguiente línea:

<TITLE>Mi Primera Página**</TITLE>**

Dentro del encabezado podemos incluir otras directivas adicionales. La directiva **<META>** indica al navegador de Internet las palabras clave y contenido de nuestra página Web. Muchos de los buscadores de páginas Web de Internet (Yahoo, Lycos, Altavista, etc.) utilizan el contenido de esta directiva para incluir la página en sus bases de datos. La directiva **<META>** lleva generalmente dos parámetros, **name** y **content**.

Ejemplos :

<META name = "Pagina de Andrés" **content** = "Mi página personal, Música y Libros">

Indica al navegador el nombre de la página y sus contenidos principales.

<META name = "keywords" **content** = "Andrés música películas vínculos Ecuador Ambato">

Indica al navegador las palabras clave para los buscadores de Internet.

Otro uso de la directiva **<META>** es la de indicar documentos con "refresco automático". Si se indica una ***URL*** se sustituirá el documento por el indicado una vez transcurridos el número de segundos especificados. Si no se incluye ninguna ***URL*** se volverá a cargar en el navegador el documento en uso transcurridos los segundos indicados. Esto es útil para páginas que cambian de contenido con mucha frecuencia o para redireccionar a la persona que visita nuestra pagina Web a una nueva dirección donde se encuentra una versión actualizada de nuestra pagina Web.

Ejemplo :

<META http-equiv= "refresh" **content =** "15; URL= http://www.microsoft.com " >

Transcurridos 15 segundos se accederá a la pagina Web de Microsoft.

La directiva **<BASE>** indica la localización de los archivos, gráficos, sonidos, etc. a los que se hace referencia en nuestra página Web. Si no se incluye esta directiva el navegador entiende que dichos elementos se encuentran en el mismo lugar donde se encuentra nuestra página Web.

Ejemplo :

<BASE href = "http://www.plasticaucho.com.ec/Andres/">

b) Cuerpo

El cuerpo de un documento HTML contiene el texto que, con la presentación y las directivas que se decidan, se presentará ante el usuario. Dentro del cuerpo son aplicables todos las directivas y efectos que se van a mencionar en el resto de este capítulo. Dichos efectos se especifican exclusivamente a través de directivas. Esto quiere decir que los espacios, tabulaciones y retornos de carro que se introduzcan en el archivo fuente no tienen ningún efecto a la hora de la presentación final del documento. Por ejemplo, escribiendo:

```
Estas
    palabras
forman      una
  frase.
```

Producimos exactamente lo mismo que con:

Estas palabras forman una frase.

A la hora de la verdad lo que se ve es:

Estas palabras forman una frase.

c) Opciones

<**Body**> tiene una serie de opciones que permitirán cambiar la apariencia global del documento en general:

background= "nombre de archivo gráfico"

Indica el nombre de un archivo gráfico que servirá como "fondo" de nuestra página. Si la imagen no rellena todo el fondo del documento, esta será reproducida tantas veces como sea necesario.

bgcolor = "código de color"

Indica un color para el fondo de nuestro documento. Se ignora si se ha usado el parámetro **background**.

text = "código de color"

Indica un color para el texto que incluyamos en nuestro documento. Por defecto es negro.

link = "código de color"

Indica el color de los textos que dan acceso a un **Hipervínculo**. Por defecto es azul.

vlink = "código de color"

Indica el color de los textos que dan acceso a un **Hipervínculo** que ya hemos visitado con nuestro navegador. Por defecto es púrpura.

El ***código de color*** es un número compuesto por tres pares de cifras hexadecimales que indican la proporción de los colores "primarios", ***rojo***, ***verde*** y ***azul***. El código de color se antecede del símbolo #.

Ejemplos :

#000000	Color	Negro
#FF0000	Color	Rojo
#00FF00	Color	Verde
#0000FF	Color	Azul
#FFFFFF	Color Blanco	

El primer par de cifras indican la proporción de color Rojo, el segundo par de cifras la proporción de color Verde y las dos últimas la proporción de color Azul. Cada par de cifras hexadecimales nos permiten un rango de 0 a 255. Combinando las proporciones de cada color primario obtendremos diferentes colores.

De cualquier forma la mayoría de los editores de HTML nos permiten obtener el código de color correspondiente escogiendo directamente el color de una paleta.

d) Directivas del cuerpo de H.T.M.L.

Las directivas que van en el cuerpo del documento tienen que estar de acuerdo con el formato señalado en las páginas anteriores; cuando una directiva no existe o no es compatible con el navegador que el usuario esta utilizando será simplemente ignorada y no se ejecutará nada de lo establecido.

Hemos dividido las directivas dependiendo si estas afectan a un párrafo en general o a una línea o palabra en particular:

e) Directivas del Párrafo

Estas son las directivas más importantes:

Etiqueta <P>

Sirve para delimitar un párrafo. Inserta una línea en blanco antes del texto.

Etiqueta <CENTER> </CENTER>

Permite centrar todo el texto del párrafo.

Etiqueta <PRE WIDTH = #> </PRE>

Representa el texto encerrado en ella con un tipo de letra de paso fijo, respeta todos los retornos de carro, tabulaciones, espacios, etc. Que se han escrito en el documento

H.T.M.L. Es muy útil a la hora de representar código fuente. El parámetro WIDTH especifica el número máximo de caracteres en una línea.

Etiqueta <DIV ALIGN =dirección> </DIV>

Permite justificar el texto del párrafo a la izquierda (`ALIGN=LEFT`), derecha (`RIGHT`), al centro (`CENTER`) o a ambos márgenes (`JUSTIFY`)

Etiqueta <ADDRESS> </ADDRESS>

Sirve para escribir direcciones (no electrónicas)

Etiqueta <BLOCKQUOTE > </BLOCKQUOTE>

Para citar un texto ajeno. Se suele implementar dejando márgenes tanto a izquierda como a derecha, razón por la que se usa habitualmente.

f) Cabeceras

En un documento de HTML se pueden indicar seis tipos de cabeceras (tamaños de letra) por medio de las directivas **<H1><H2><H3><H4><H5>** y **<H6>**. El texto que escribamos entre el inicio y el fin de la directiva será el afectado por las cabeceras. La cabecera **<H1>** será la que muestre el texto en mayor tamaño. No resulta recomendable utilizarlas para aumentar o disminuir el tamaño del tipo de letra, ya que cada navegador los muestra de manera diferente. Se usan para dividir correctamente en secciones nuestra página, tal y como se hace en un documento de texto normal.

Ejemplo	Se vería como
<H1>Texto de Prueba**</H1>**	Texto de prueba
<H2>Texto de Prueba**</H2>**	Texto de Prueba
<H3>Texto de Prueba**</H3>**	Texto de Prueba
<H4>Texto de Prueba**</H4>**	Texto de Prueba
<H5>Texto de Prueba**</H5>**	Texto de Prueba
<H6>Texto de Prueba**</H6>**	Texto de Prueba

Los textos marcados como "cabeceras" provocan automáticamente un retorno de carro sin necesidad de incluir la directiva <**BR**>. Por ejemplo:

Ejemplo	Se vería como
<H3>Pagina de Francisco**</H3>**Esta es mi pagina personal.	**Pagina de Francisco** Esta es mi pagina personal

g) Directivas de Líneas

Para indicar atributos del texto (negrilla, subrayado, etc.) tenemos varias directivas. Algunas de ellas no son reconocidas por determinados navegadores de Internet, es por ello que según el navegador que este el usuario este utilizando, se verá el resultado correctamente o no.

Etiqueta <B> </B>

Pone el texto en negrita.

Etiqueta <I> </I>

Representa el texto en cursiva.

Etiqueta <U> </U>

Para subrayar algo.

Etiqueta <S> </S>

Para tachar.

Etiqueta <TT> </TT>

Permite representar el texto en un tipo de letra de paso fijo.

Etiqueta

Letra superíndice.

Etiqueta

Letra subíndice.

Etiqueta <BIG> </BIG>

Incrementa el tipo de letra.

Etiqueta <SMALL> </SMALL>

Disminuye el tamaño del tipo de letra.

Etiqueta <BLINK> </BLINK>

El texto señalado parpadea.

Etiqueta <HR> </HR>

La directiva **<HR>** muestra una línea horizontal de tamaño determinable. Tiene los siguientes parámetros opcionales :

align = posición

Alinea la línea a la izquierda (left), a la derecha (right) o la centra (center).

noshade

No muestra sombra, evitando el efecto en tres dimensiones.

size = número

Indica el grosor de la línea en pixels.

width = num / %

Indica el ancho de la línea en tanto por ciento en función del ancho de la ventana del navegador. También se puede especificar un número que indicaría el ancho de la línea en pixels.

Ejemplo :

<HR align= center size= 20 width= 50%>

La directiva **<HR>** sin ningún parámetro mostraría una línea horizontal que ocuparía todo el ancho de la página.

1.2.1.5 JUEGO DE CARACTERES DEL DOCUMENTO.

Todos los exploradores de páginas Web actuales soportan todos los caracteres gráficos de la especificación *ISO 8859-1*, que permiten escribir textos en la mayoría de los países occidentales.

De cualquier forma y como muchos sistemas tienen distintos juegos de caracteres ASCII, se han definido dos formas de representar caracteres especiales usando solamente el código ASCII de 7 bits. Para hacer referencia a estos caracteres se les asigna un código numérico o un nombre de "*entidad*". Asimismo hay caracteres que se utilizan para las directivas de HTML, por ejemplo < y >. Estos caracteres pueden ser representados por un código numérico o una entidad cuando deseemos que aparezcan en el documento "tal cual". Las entidades comienzan por el símbolo **&** (ampersand o "y" comercial) y terminan con el símbolo **;** (punto y coma).

A continuación veamos una tabla con las principales entidades:

Caracter	Código	Entidad	Caracter	Código	Entidad
!	&#33;	--	"	&#34;	--
#	&#35;	--	$	&#36;	--
%	&#37;	--	&	&#38;	--
'	&#39;	--	(	&#40;	--
)	&#41;	--	*	&#42;	--
+	&#43;	--	,	&#44;	--
-	&#45;	--	.	&#46;	--
/	&#47;	--	:	&#58;	--
;	&#59;	--	<	&#60;	--
=	&#61;	--	>	&#62;	--
?	&#63;	--	@	&#64;	--
[	&#91;	--	\	&#92;	--
]	&#93;	--	^	&#94;	--
_	&#95;	--	`	&#96;	--
{	&#123;	--	|	&#124;	--
}	&#125;	--	~	&#126;	--
		nbsp	¡	&#161;	iexcl
¢	&#162;	cent	£	&#163;	pound

¤	&#164;	curren	¥	&#165;	yen
¦	&#166;	brvbar	§	&#167;	sect
¨	&#168;	uml	©	&#169;	copy
ª	&#170;	ordf	«	&#171;	laquo
¬	&#172;	not	-	&#173;	shy
®	&#174;	reg	¯	&#175;	macr
°	&#176;	deg	±	&#177;	plusmn
²	&#178;	sup2	³	&#179;	sup3
´	&#180;	acute	µ	&#181;	micro
¶	&#182;	para	·	&#183;	middot
¸	&#184;	cedil	¹	&#185;	sup1
º	&#186;	ordm	»	&#187;	raquo
¼	&#188;	frac14	½	&#189;	frac12
¾	&#190;	frac34	¿	&#191;	iquest
À	&#192;	Agrave	Á	&#193;	Aacute
Â	&#194;	Acirc	Ã	&#195;	Atilde
Ä	&#196;	Auml	Å	&#197;	Aring
Æ	&#198;	AElig	Ç	&#199;	Ccedil
È	&#200;	Egrave	É	&#201;	Eacute
Ê	&#202;	Ecirc	Ë	&#203;	Euml
Ì	&#204;	Igrave	Í	&#205;	Iacute

ø	&#248;	oslash	ù	&#249;	ugrave
ú	&#250;	uacute	û	&#251;	ucirc
ü	&#252;	uuml	ý	&#253;	yacute
þ	&#254;	thorn	ÿ	&#255;	yuml

Por lo tanto la palabra *página* la podríamos escribir como :

página

p**á**gina

p**á**gina

Es por ello que si deseamos que cualquier navegador de páginas Web pueda visualizar las letras acentuadas de nuestro documento debemos utilizar sus correspondientes entidades o códigos para representarlas.

Se ha incluido un archivo de ejemplo con todos las directivas que se han explicado hasta el momento.

1.2.1.6 LISTAS DE ELEMENTOS

Existen tres tipos de listas, numeradas, sin numerar y de definición.

Las listas numeradas representarán los elementos de la lista numerando cada uno de ellos según el lugar que ocupan en la lista. Para este tipo de lista se utiliza la directiva

<OL> **</OL>**. Cada uno de los elementos de la lista irá precedido de la directiva **<LI>**. La directiva **<OL>** puede llevar los siguientes parámetros :

start = número

Indica que número será el primero de la lista. Si no se indica se entiende que empezará por el número 1.

type = tipo

Indica el tipo de numeración utilizada. Si no se indica se entiende que será una lista ordenada numéricamente.

Los tipos posibles son :

1	=	Numéricamente	(1,2,3,4,...etc.)
a	=	Letras minúsculas	(a, b, c, d, ... etc.)
A	=	Letras mayúsculas	(A,B,C,D,... etc.)
i	=	Números romanos en minúsculas.	(i, ii, iii, iv, v,... etc.)
I	=	Números romanos en mayúsculas.	(I,II,III,IV,V,... etc.)

Ejemplo	Resultado
<OL> **<LI>**España **<LI>**Francia **<LI>**Italia	España Francia Italia Portugal

<LI>Portugal **</OL>**	
<OL type = A > **<LI>**España **<LI>**Francia **<LI>**Italia **<LI>**Portugal **</OL>**	 España Francia Italia Portugal

Las listas sin numerar representan los elementos de la lista con una viñeta o marca que antecede a cada uno de ellos. Se utiliza la directiva **<UL> </UL>** para delimitar la lista, y **<LI>** para indicar cada uno de los elementos. La directiva **<UL>** puede contener el parámetro **type** que indica la forma de la viñeta o marca que antecede a cada elemento de la lista. Los valores de **type** pueden ser **disk**, **circle** o **square**, con lo que la viñeta o marca puede ser un disco, un circulo o un cuadrado.

Ejemplo	Resultado
<UL type = disk **>** **<LI>**España **<LI>**Francia **<LI>**Italia **<LI>**Portugal **</UL>**	 España Francia Italia Portugal

Las listas de definición muestran los elementos tipo Diccionario, o sea, término y definición. Se utiliza para ellas la directiva **<DL></DL>**. El elemento marcado como término se antecede de la directiva **<DT>**, el marcado como definición se antecede de la directiva **<DD>**.

Ejemplo	Resultado
<DL> **<DT>**WWW **<DD>**Abreviatura de World Wide Web **<DT>**FTP **<DD>**Abreviatura de File Transfer Protocol **<DT>**IRC **<DD>**Abreviatura de Internet Relay Chat **</DL>**	WWW Abreviatura de World Wide Web FTP Abreviatura de File Transfer Protocol IRC Abreviatura de Internet Relay Chat

Existen otros dos tipos de listas menos comunes. Las listas de Menú o Directorio se comportan igual que las listas sin numerar. La lista de Menú utiliza la directiva **<MENU></MENU>** y los elementos se anteceden de **<LI>** El resultado es una lista sin numerar mas "compacta" es decir, con menos espacio interlineal entre los elementos. La

lista de Directorio utiliza la directiva **<DIR></DIR>** y los elementos se anteceden de **<LI>**. Los elementos tienen un limite de 20 caracteres.

Todas las listas se pueden "anidar", es decir incluir una lista dentro de otra, con lo que se consigue una estructura tipo "índice de materias".

Ejemplo	Resultado
<UL type= disk> **<LI>**Buscadores **<UL>** **<LI>**Yahoo **<LI>**Ole **<LI>**Lycos **</UL>** **<LI>**Links **<UL>** **<LI>**Microsoft **<LI>**IBM **</UL>** **</UL>**	Buscadores Yahoo Ole Lycos Links Microsoft IBM

1.2.1.7 IMÁGENES

Para incluir una imagen en una página Web se utiliza la directiva **<IMG>**. Hay dos formatos de imágenes que todos los navegadores modernos reconocen. Son las imágenes ***GIF*** y ***JPG***. Cualquier otro tipo de archivo gráfico o de imagen (BMP, PCX, CDR, etc.) no será mostrado por el navegador, a no ser que disponga de un programa externo que permita su visualización.

La directiva **<IMG>** tiene varios parámetros:

src = "imagen"

Indica el nombre del archivo gráfico a mostrar.

alt = "Texto"

Mostrara el texto indicado en el caso de que el navegador utilizado para ver la página no sea capaz de visualizar la imagen.

lowsrc ="imagen"

Muestra una segunda imagen "superpuesta" sobre la primera una vez se carga la pagina. Este parámetro no es reconocido por la totalidad de los navegadores ya que esta en estudio su aplicación, así que en la mayoría de los casos será ignorado mostrándose solo la primera imagen (**src**). En Netscape muestra la imagen indicada por **lowsrc** en primer lugar, y posteriormente muestra la imagen indicada por **src**. Si las imágenes son iguales

pero tienen distinta "resolución" se conseguirá un efecto tipo "Fade". Si las imágenes son de distinto tamaño la imagen indicada en **src** se redimensionara al tamaño indicado por la imagen indicada en **lowsrc**.

align = TOP / MIDDLE / BOTTOM

Indica como se alineará el texto que siga a la imagen. TOP alinea el texto con la parte superior de la imagen, MIDDLE con la parte central, y BOTTOM con la parte inferior.

border = tamaño

Indica el tamaño del "borde" de la imagen. A toda imagen se le asigna un borde que será visible cuando la imagen forme parte de un Hipervínculo.

height = tamaño

Indica el alto de la imagen en puntos o en porcentaje. Se usa para variar el tamaño de la imagen original.

width = tamaño

Indica el ancho de la imagen en puntos o en porcentaje. Se usa para variar el tamaño de la imagen original.

hspace = margen

Indica él número de espacios horizontales, en puntos, que separarán la imagen del texto que la siga y la anteceda.

vspace = margen

Indica el número de puntos verticales que separaran la imagen del texto que le siga y la anteceda.

ismap / usemap

Indica que la imagen es un MAPA.

1.2.1.8 HIPERVÍNCULOS.

La característica principal de una página Web es que podemos incluir Hipervínculos. Un Hipervínculo es un elemento de la página que hace que el navegador acceda a otro recurso, otra página Web, un archivo, etc.

Para incluir un Hipervínculo se utiliza la directiva **<A></A>**. El texto o imagen que se encuentre dentro de los límites de esta directiva será sensible, esto quiere decir que si pulsamos con el ratón sobre él, se realzará la función de hipervínculo indicada por la directiva **<A></A>**. Si el Hipervínculo esta indicado por un texto, este aparecerá subrayado y en distinto color, si se trata de una imagen, esta aparecerá con un borde rodeándola. Esta directiva tiene el parámetro **href** que indica el lugar a donde nos llevará el Hipervínculo si lo pulsamos.

Por ejemplo, si tenemos

<A href = "http://www.microsoft.com/"> Pulse para ir a la página de Microsoft**</A>**

En nuestro navegador se verá como :

Pulse para ir a la página de Microsoft

Si se da un clic en este hipervínculo, automáticamente se irá hacia la página de Microsoft, que se encuentra detallada con la opción href.

Lo mismo se puede hacer con un gráfico:

<A href = "http://www.yahoo.com/" > **<IMG src** = "yahoo.gif"**></A>**

Y en el navegador se verá así:

Para búsquedas : 

Pulsando sobre la imagen se accedería a la pagina situada en *http://www.yahoo.com/.*

Un Hipervínculo también puede llevarnos a una zona de la página. Para ello debemos marcar en la página las diferentes secciones en las que se divide. Lo haremos con el parámetro **name**, ejemplo:

<A name = "seccion1"></A>

Esta instrucción marca el inicio de una sección dentro de la página. La sección se llamará *seccion1*. Para hacer un enlace a esta sección dentro de la página lo haríamos de la siguiente forma :

<A href = "#seccion1">Primera Parte</A>

O también :

<A href = "http://www.jet.es/mipagina.htm#seccion1">Primera Parte</A>

Un Hipervínculo puede hacerse a cualquier tipo de archivo. Con las directivas anteriores hemos visto como hacer enlaces a páginas Web o secciones dentro de una página Web, pero podríamos hacer un Hipervínculo a un grupo de noticias, o a otro servicio de Internet, ejemplo:

<A href = "news://news.ecua.net.ec/">Noticias de actualidad</A>

Asimismo podemos hacer que el Hipervínculo de como resultado el envío de un correo electrónico a una dirección de correo determinada, ejemplo:

<A href = "mailto: webmaster@puce.amb.edu" > Envíame tus sugerencias</A>

También podemos realizar un Hipervínculo a un archivo cualquiera. En este caso el navegador intentará "ejecutar" el archivo, y si no puede hacerlo nos preguntará si deseamos grabarlo en nuestra computadora. Esta es una forma sencilla de permitir a los visitantes de nuestra página copiar archivos a la computadora, ejemplo:

<A href = "paquete.zip">Clic aquí para llevarte una copia del programa.**</A>**

El parámetro **onMouseOver** permite que se muestre en la barra inferior del navegador un texto explicativo sobre el Hipervínculo, en vez de mostrar su dirección. Este parámetro no funciona en todos los navegadores, ejemplos:

<A href = "paquete.zip" **onMouseOver="window.status=**'Este vínculo copia el programa al disco duro' **; return true;**">Clic aquí para obtener una copia del programa.**</A>**

1.2.1.9 TABLAS

Las tablas nos permiten representar cualquier elemento de nuestra página (texto, listas, imágenes, etc.) en diferentes filas y columnas separadas entre si. Es una herramienta muy útil para "ordenar" contenidos de distintas partes de nuestra página. La tabla se define mediante la directiva **<TABLE></TABLE>.** Los parámetros opcionales de esta directiva son :

border = num.

Indica el ancho del borde de la tabla en puntos.

cellspacing = num

Indica el espacio en puntos que separa las celdas que están dentro de la tabla.

cellpadding = num

Indica el espacio en puntos que separa el borde de cada celda y el contenido de esta.

width = num ó %

Indica la anchura de la tabla en puntos o en porcentaje en función del ancho de la ventana del navegador. Si no se indica este parámetro, el ancho se adecuará al tamaño de los contenidos de las celdas.

height = num ó %

Indica la altura de la tabla en puntos o en porcentaje en función del alto de la ventana del navegador. Si no se indica este parámetro, la altura se adecuará a la altura de los contenidos de las celdas.

bgcolor = código de color

Especifica el color de fondo de toda la Tabla.

Para definir las celdas que componen la tabla se utilizan las directivas **<TD>** y **<TH>**. **<TD>** indica una celda normal, y **<TH>** indica una celda de "cabecera", es decir, el contenido será resaltado en negrita y en un tamaño ligeramente superior al normal. Los parámetros opcionales de ambas directivas son :

align = LEFT / CENTER / RIGHT / JUSTIFY

Indica como se debe alinear el contenido de la celda, a la izquierda (LEFT), a la derecha (RIGHT), centrado (CENTER) o justificado (JUSTIFY).

valign = TOP / MIDDLE / BOTTOM

Indica la alineación vertical del contenido de la celda, en la parte superior (TOP), en la inferior (BOTTOM), o en el centro (MIDDLE).

rowspan = num

Indica el número de filas que ocupará la celda. Por defecto ocupa una sola fila.

colspan = num

Indica el número de columnas que ocupará la celda. Por defecto ocupa una sola columna.

width = num ó %

Indica la anchura de la columna en puntos o en porcentaje en función del ancho de la ventana del navegador. Si no se indica este parámetro, el ancho se adecuará al tamaño de los contenidos. Este parámetro solo funciona en los navegadores modernos.

bgcolor = código de color

Especifica el color de fondo del elemento de la Tabla.

Para indicar que acaba una fila de celdas se utiliza la directiva **<TR>**. A continuación mostraremos un ejemplo de una tabla que contiene solo texto. Como se indicó anteriormente el contenido de las celtas puede ser cualquier elemento de HTML, un texto, una imagen, un Hipervínculo, una Lista, etc.

Ejemplo
<TABLE border = 4 cellspacing = 4 cellpadding = 4 width =80%>
<TH align = center>Buscadores
<TH align = center **colspan** = 2>Otros vínculos
<TR>
<TD align = LEFT>Yahoo
<TD align = LEFT>Microsoft
<TD align = LEFT>IBM

<TR>

<TD align = LEFT>Infoseek

<TD align = LEFT>Apple

<TD align = LEFT>Digital

</TABLE>

Buscadores	Otros Vínculos	
Yahoo	Microsoft	IBM
Infoseek	Apple	Digital

Las directivas **<TD>** y **<TH>** son cerradas según el estándar de H.T.M.L. , es decir que un elemento de tabla **<TD>** debería cerrarse con un **</TD>** , sin embargo los navegadores asumen que un elemento de la tabla, queda automáticamente "cerrado" cuando se "abre" el siguiente.

1.2.1.10 MAPAS

Un Mapa es una imagen que permite realizar diferentes Hipervínculos en función de la "zona" de la imagen que se pulse. Las directivas para crear mapas son **<MAP></MAP>** y **<AREA>**.

La directiva **<MAP>** identifica al mapa y tiene el parámetro **name** para indicar el nombre del mapa.

La directiva **<AREA>** define las áreas sensibles de la imagen. Tiene los siguientes parámetros obligatorios:

shape = "tipo"

Indica el tipo de área a definir.

coords = "coordenadas"

Indica las coordenadas de la figura indicada con shape.

href = "URL"

Indica la dirección a la que se accede si se pulsa en la zona delimitada por el área indicada.

Los tipos de área pueden ser los siguientes:

rect

Area rectangular. Se deben especificar las coordenadas "X1,Y1,X2,Y2" que corresponden a la esquina superior izquierda y esquina inferior derecha.

poly

Polígono. Se deben especificar las coordenadas "x1,y1,x2,y2,x3,y3,..." definiendo cada pareja (x,y) una esquina del polígono La última pareja de coordenadas se unirá a la primera para cerrar el polígono.

circle

Circulo. Se debe especificar en primer lugar las coordenadas "X1,Y1,R" que corresponden al centro del círculo y a continuación el valor del radio (en puntos).

Si dos áreas se superponen, se ejecutará la que se encuentre en primer lugar en la definición del mapa. Es importante definir una última área que abarque la totalidad del gráfico para direccionar a una URL "por defecto", con el objeto de contemplar el caso de que no se pulse sobre un área definida.

Aquí esta un ejemplo de cómo usar estas directivas:

```
<MAP name = "casa">
<AREA shape = "poly" coords = "2,62,57,62,28,1" href= "tejado.htm">
<AREA shape = "rect" coords = "21,101,35,138" href= "puerta.htm">
<AREA shape = "rect" coords = "2,64,57,138" href= "casa.htm">
<AREA shape = "circle" coords = "80,76,21" href= "arbol.htm">
<AREA shape = "rect" coords = "78,98,85,138" href= "tronco.htm">
<AREA shape = "rect" coords = "0,0,96,138" href= "dibujo.htm">
</MAP>
```

Para activar el mapa debemos indicar la imagen a mostrar, indicando que dicha imagen es tratada por un mapa. Para ello escribiríamos la siguiente directiva:

<IMG src = "grafico.gif" **usemap** = "#casa">

Existen utilitarios en Internet que ayudan al usuario a elaborar un mapa de manera sencilla, un buen programa es Mapthis que se puede bajar gratis de la Red Mundial.

1.2.1.11 FORMULARIOS

Los formularios nos permiten dentro de una página Web solicitar información al visitante y procesarla. En un formulario podremos solicitar diferentes datos (campos) cada uno de los cuales quedará asociado a una variable. Una vez se hayan introducido los valores en los campos, el contenido de estos será enviado a la dirección (URL) donde tengamos el programa que pueda procesar las variables. Para poder realizar este último paso de procesar las variables necesitaremos realizar un programa externo en algún lenguaje de programación como PERL, C++ o Visual Basic. A este programa externo se le suele llamar ***CGI*** (Common Gateway Interface). Aquí veremos como se pueden enviar las variables a nuestra dirección de correo electrónico, para otros programas se deberá consultar la sección referente a CGI

La declaración del formulario se pone entre las directivas **<FORM></FORM>**. En el interior de la declaración se indican los elementos (variables) de entrada. La directiva **<FORM>** tiene los parámetros **action** y **method**.

action = "programa"

Indica el programa que va a "tratar" a las variables que se envíen con el formulario. En nuestro caso enviaremos las variables por correo electrónico, con lo que el "programa" será **"mailto: direccion_de_correo"**.

method = POST / GET

Indica el método según el que se transferirán las variables. POST produce la modificación del documento de destino (como en el caso de enviar por correo las variables). GET no produce cambios en el documento destino (como en el caso de una consulta a una base de datos, por ejemplo una página de búsqueda en Internet).

Campos de Entrada

Para la introducción de las variables se utiliza la directiva **<INPUT>**. Esta directiva tiene el parámetro **type** que indica el tipo de variable a introducir y **name** que indica el nombre que se le dará al campo. Cada tipo de variable tiene sus propios parámetros.

type= text **name** = campo

Indica que el campo a introducir será un texto. Sus parámetros son:

maxlenght = número

Número máximo de caracteres a introducir en el campo.

size = número

Tamaño en caracteres que se mostrará en pantalla.

value = "texto"

Valor inicial del campo. Normalmente será " ", o sea, vacío.

type = password **name** = campo

Indica que el campo será una palabra de paso. Mostrará asteriscos (*) en lugar de las letras escritas. Sus parámetros opcionales son los mismos que para text.

type = checkbox **name** = campo

El campo se elegirá marcando una casilla. Se permite marcar varias casillas. Los valores de las casillas serán indicados por:

value = "valor"

checked

La casilla aparecerá marcada por defecto.

type = radio name = campo

El campo se elegirá marcando una casilla. Solo permite marcar una sola de las casillas. Los valores de las casillas serán indicados por:

value = "valor"

type = image name = campo

El campo contendrá el valor de las coordenadas del punto seleccionado de la imagen. Debe indicarse la imagen con el parámetro:

src = "fichero de imagen".

type = hidden name = campo

El usuario no puede modificar su valor, ya que el campo no es visible se manda siempre con el valor indicado por el parámetro:

value = "valor"

type = submit

Representa un botón. Al pulsar este botón la información de todos los campos se envía al programa indicado en **<FORM>**. Tiene el parámetro **value** = "texto" que indica el texto que aparecerá en el botón.

type = reset

Representa un botón. Al pulsar este botón se borra el contenido de todos los campos. El parámetro **value** = "texto" indica el texto que aparecerá en el botón.

Campos de Selección

Este tipo de campos despliegan una lista de opciones, entre las que debemos escoger una o varias. Se utiliza para ellos la directiva **<SELECT> </SELECT>**. Sus parámetros son:

name = campo

Nombre del campo

size = num

Número de opciones visibles. Si se indica 1 se presenta como un menú desplegable, se indica mas de uno se presenta como una lista con barra de desplazamiento.

multiple

Permite seleccionar mas de un valor para el campo.

Las diferentes opciones de la lista se indican con la directiva **<OPTION>**. Esta directiva puede incluir el parámetro **selected** para indicar cual es la opción por defecto. En caso de que no se especifique, se tomara por defecto la primera opción de la lista.

Áreas de texto.

Representa un campo de texto de múltiples líneas. Normalmente se utiliza para que se incluyan en los comentarios. La directiva usada es **<TEXTAREA> </TEXTAREA>**, y sus parámetros:

name = campo

Nombre del campo.

cols = núm.

Número de columnas de texto visibles.

rows = núm.

Número de filas de texto visibles.

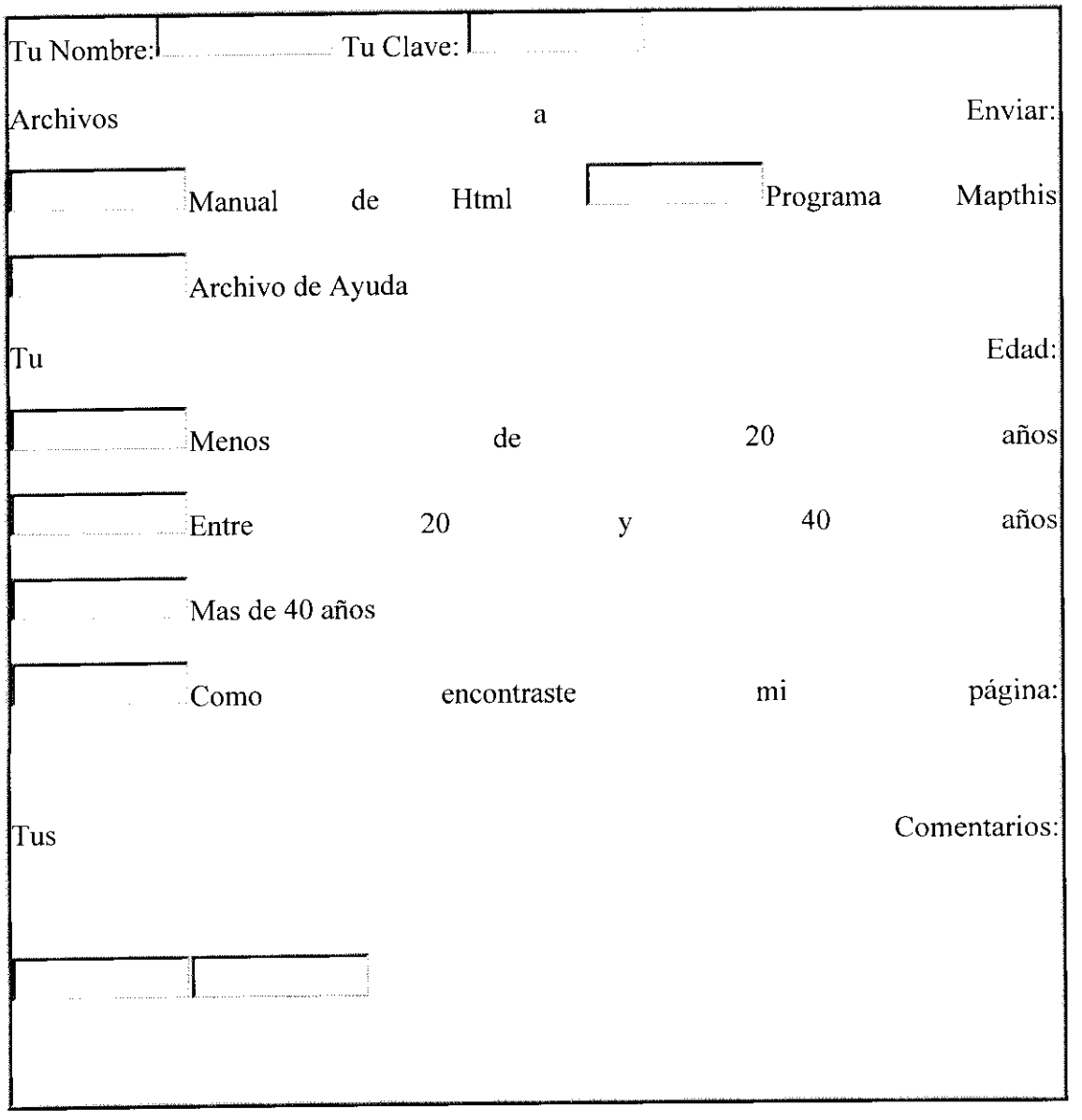
wrap = VIRTUAL / PHYSICAL

Justifica el texto automáticamente en el interior de la caja. La opción PHYSICAL envía las líneas de texto separadas en líneas físicas. La opción VIRTUAL envía todo el texto seguido.

Veamos a continuación un ejemplo de formulario utilizando todas las formas de introducción de datos.

<FORM action = "mailto: userid@pucesa.edu.ec" **method** = post >
Tu Nombre: **<INPUT type** = text **name** = nombre **size** = 30 >
Tu Clave: **<INPUT type** = password **name** = clave **size** = 8 >
<P>
Archivos a Enviar:
<INPUT type = checkbox **name** = archivo **value** = "Manual" > Manual de Html
<INPUT type = checkbox **name** = archivo **value** ="Mapthis" > Programa Mapthis
<INPUT type = checkbox **name** = archivo **value** ="Help" > Archivo de Ayuda
<P>
Tu Edad :
<INPUT type = radio **name** = edad **value** = "-20" > Menos de 20 años
<INPUT type = radio **name** = edad **value** = "20-40" > Entre 20 y 40 años
<INPUT type = radio **name** = edad **value** = "+40" > Mas de 40 años
<P>
<INPUT type = hidden **name** = lugar **value** = "pagina personal" >
Como encontraste mi página :
<SELECT name = donde >
<OPTION>De casualidad
<OPTION>Por el buscador Altavista
<OPTION>Por el buscador Yahoo
<OPTION>Me la comentaron
</SELECT>
<P>
Tus Comentarios:
**
**
<TEXTAREA name = comentario **rows** = 5 **cols** = 40 **wrap** = virtual
></TEXTAREA>
<P>
<INPUT type = submit **value** = "Enviar" >
<INPUT type = reset **value** = "Borrar" >
</FORM>

Ahora veamos el efecto producido en la página Web:



Si se rellena este FORM y se pulsa sobre el botón ***Enviar***, (estando conectado a Internet), se generará un mensaje de correo a una dirección de correo *userid@pucesa.edu.ec*. Si pulsas el botón ***Borrar*** se borraran los datos que hayas introducido en el Formulario.

El texto que se recibiría por correo electrónico sería parecido a este:

> Nombre=Pedro+Perez &clave=12345678 &archivo=Manual &archivo=Mapthis &edad=20-40 &lugar=pagina+personal &donde=Por+el+buscador+Altavista &comentario%94= Espero+que+me+mandes+los%0D%0Aficheros+antes+del+martes%0D%0A%0D%0AS aludos. %0D%0A

Podemos observar que en el correo se separan las variables con el símbolo **&**, los espacios se sustituyen por el signo + y se representan los códigos de retorno de carro y avance de línea del campo de texto con los caracteres ***%0D*** y ***%0A*** respectivamente.

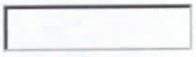
Si en vez de enviar estas variables por correo electrónico, fuesen enviadas a un programa (CGI), este programa podría tratarlas y dar como respuesta una nueva página Web.

IMPORTANTE: La opción **mailto:** en el parámetro **action** de la directiva **FORM** solo funciona correctamente con Netscape. En Microsoft Explorer esta opción da como resultado un correo en blanco. Para enviar un formulario por e-mail sin importar el navegador utilizado se ha de utilizar un programa externo que realice este proceso.

Se puede utilizar un Formulario como "lanzador" de links, es decir, por medio de una lista desplegable permitir al usuario escoger un Link y acceder a él. Para ello se utiliza el parámetro "OnClick". Ejemplo :

```
<FORM>
<SELECT name = "list" >
<OPTION SELECTED value= "http://web.jet.es/luisfd">Mi Pagina Web
<OPTION value= "http://www.microsoft.com">Microsoft
<OPTION value= "http://www.ibm.com">Ibm
<OPTION value= "http://www.novell.com">Novell
<OPTION value= "http://www.hp.com">Hewlett Packard
<OPTION value= "http://www.fujitsu.com">Fujitsu
<OPTION value="http://www.3com.com">3Com
</SELECT>
<INPUT TYPE=BUTTON value= "Ir a..."
onClick="top.location.href=this.form.list.options[this.form.list.selectedIndex].v
alue">
</FORM>
```

Este seria el efecto producido:



1.2.1.12 EXTENSIONES DEL HTML

Netscape y Microsoft han añadido al estándar de HTML diversas directivas para hacer más atractiva la visualización de las páginas Web. Estas directivas pueden no funcionar en algún navegador de HTML, pero el uso de ellas por parte de los dos "grandes" del software para Internet hace prever que serán inmediatamente incluidas en las nuevas versiones del resto de los navegadores

1.2.1.13 APPLET

La directiva **<APPLET></APPLET>** indica la ejecución de un programa (applet) externo escrito en lenguaje ***JAVA***. Java es un lenguaje creado por Sun Microsystems

que permite realizar operaciones multimedia sin incorporar nuevas directivas HTML. Los applets son muy variados, y cada uno de ellos realiza una tarea distinta. Hay applets para hacer que el texto se mueva dentro de la hoja, se contraiga y expanda, etc. Esta directiva tiene los siguientes parámetros :

codebase = URL

Dirección donde se encuentra el applet Java (Por ejemplo *http://www.ucm.es/java*). Si el Applet se encuentra en el mismo lugar que la pagina Web este parámetro no es necesario.

code = programa

Indica el nombre del programa (applet) Java a ejecutar.

width = num.

height = num.

Indican el espacio (ancho y alto) en puntos en el que el programa realizará su función.

Dentro de la directiva **<APPLET>** se incluye la directiva **<PARAM>** que envía al programa Java los parámetros necesarios para su funcionamiento. Esta directiva suele tener como mínimo los parámetros :

name = campo

Nombre de la variable a enviar.

value = valor

Valor de la variable a enviar.

Veamos un ejemplo en el que se ejecuta un programa Java que permite que un texto se desplace de un lado a otro de una zona de la pantalla:

```
<APPLET code="Laufschrift.class" width = 350 height = 25 >
<PARAM name = bg.color value = "0,255,0">
<PARAM name = message value = "**Bienvenido a mi pagina WEB">
</APPLET>
```

1.2.1.14 MARQUEE

La directiva **<MARQUEE></MARQUEE>** crea una marquesina con un texto en su interior que se desplaza. Funciona únicamente con Ms-Explorer. Sus parámetros son los siguientes :

align = top / middle / bottom

Indica si el texto del interior de la marquesina se alinea en la zona alta (top), en la baja (bottom) o en el centro (middle) de la misma.

bgcolor = "código de color"

Indica el color del fondo de la marquesina.

direction = left / right

Indica hacia que lugar se desplaza el texto, hacia la izquierda (left) o hacia la derecha (right).

height = num o %

Indica la altura de la marquesina en puntos o porcentaje en función de la ventana del navegador o explorador.

width = num o %

Indica la anchura de la marquesina en puntos o porcentaje en función de la ventana del navegador.

loop = num / infinite

Indica el número de veces que se desplazará el texto por la marquesina. Si se indica infinite, se desplazará indefinidamente.

scrolldelay = num.

Indica el número de milisegundos que tarda en reescribirse el texto por la marquesina, a mayor número mas lentamente se desplazará el texto.

Veamos un ejemplo de esta directiva :

<MARQUEE bgcolor = "#FFFFFF" **width** = 50% **scrolldelay** = 0 > Bienvenido a mi pagina personal en Internet.

</MARQUEE>

1.2.1.15 SONIDO DE FONDO

Una página Web puede tener un sonido que se active al entrar en la página. Esta característica de Ms Explorer utiliza la directiva **<BGSOUND>** y tiene los siguientes parámetros:

src = "fichero"

Indica el nombre del fichero que contiene el sonido (.wav, .mid).

loop = num / infinite

Indica el número de veces que se reproducirá el sonido. Si se indica infinite, el sonido se reproducirá de forma continua hasta que abandonemos la página.

Un ejemplo de esta directiva sería :

<BGSOUND src= "yesterday.mid" **loop=** infinite>

Para utilizar esta función en Netscape se utiliza la directiva **<EMBED>**. Esta directiva se utiliza realmente para "incrustar" un objeto en una pagina Web. Dicho objeto puede ser un fichero de sonido, un vídeo, un gráfico BMP, etc. Tiene los siguientes parámetros:

src = "fichero"

Indica el nombre del fichero que contiene el sonido (.wav, .mid) o el vídeo (.avi).

autostart = true

Incluirlo si deseamos que la reproducción se inicie inmediatamente.

loop = true

Incluirlo si deseamos que la reproducción no se detenga. (al terminar, vuelve a comenzar automáticamente).

volume = número

Volumen al que se reproducen los ficheros de sonido.

width = número **height** = número

Anchura y Altura de la representación del objeto. (Si es un sonido no es necesario este parámetro).

controls = smallconsole

Visualiza una serie de controles que nos permiten iniciar la reproducción del fichero, así como realizar una pausa o detenerlo.

Un ejemplo de esta directiva sería :

<EMBED src= "yesterday.mid" loop= true autostart= true volume=50 width=50 height=15 controls=smallconsole

1.2.2 LENGUAJE JAVA

1.2.2.1 INTRODUCCIÓN

Con la introducción del Internet en nuestros sistemas se perfila un cambio en la forma de concebir el trabajo con la computadora. En la concepción tradicional se disponía de una computadora y dispositivos de almacenamiento tales como disquetes, discos duros, cintas, cd´s, etc. En la actualidad una buena parte del software y de los datos residen en el Internet o serán recuperados de algún servidor de la Red. Esto conlleva algunas ventajas:

- Capacidad ilimitada de almacenamiento.
- Ahorro de almacenamiento ya que se limita a unas copias por servidor.
- La información esta permanentemente actualizada, independientemente del usuario que la use
- Se libera de problemas de seguridad al usuario.
- Las actualizaciones son más rápidas y fáciles. La tarea es ejecutada por especialistas y no por usuarios finales.
- El software es universal independiente de la plataforma usada: Macintosh, PC, UNIX, etc.

Pero también tiene sus desventajas:

- Necesidad de redes muy potentes (capacidad y velocidad).
- Costos de acceso a la información (módems, teléfonos, proveedores, etc.).
- Problemas de seguridad.
- El trabajo es dependiente de los servidores y de sus vías de comunicación.
- No existen aplicaciones realmente interactivas en Internet.

1.2.2.2 CARACTERÍSTICAS DEL LENGUAJE JAVA

El lenguaje JAVA resuelve algunos de estos problemas. Los diseñadores pensaron que este lenguaje debería resolver las condiciones siguientes:

- *Ser un lenguaje familiar*: Se trataba de diseñar un lenguaje basado en uno conocido y no algo completamente nuevo. En realidad esta basado en C y C++.
- *Eliminar problemas conocidos de otros lenguajes:* en realidad JAVA ha eliminado los tres problemas más importantes de C y C++, que son:
- *El uso de punteros:* Estos son los principales causantes de los programas desarrollados en C y C++ y son causantes de los bloqueos del sistema operativo. JAVA los eliminó y los problemas subyacentes han desaparecido.
- *La gestión de memoria:* Los programas tienen serios problemas para liberar la memoria utilizada cuando esta ya no es necesaria. JAVA utiliza un procedimiento que libera un espacio de memoria tan pronto como este ya no es utilizado.

- *El control de acceso a una matriz:* JAVA lleva control del tamaño de la matriz y no permite salirse del mismo. Esto evita muchos problemas difíciles de detectar.
- *El lenguaje debe ser orientado a objetos:* Debe trabajar con objetos y métodos que actúan sobre ellos. Las ventajas son:

  Los programas son más fáciles de modificar.

  Los programas son más fáciles de mantener.

  El código es mucho más portable y reutilizable.

  La estructura es mucho más clara y entendible.
- *El lenguaje debe ser portable:* JAVA es un lenguaje que se ejecuta en cualquier plataforma sin modificación alguna; aunque la calidad y el número de librerías de clases son muy limitados.
- *El lenguaje debe ser de alto rendimiento*: esto exige que los programas deben dar grandes prestaciones en cuanto a tiempos de ejecución y uso de memoria. Java, además, utiliza *threads* que son pequeñas aplicaciones que pueden ejecutarse a la vez y simulan un comportamiento recurrente que reparte el proceso del CPU.
- *El lenguaje debe ser lo más sencillo posible*: JAVA incorpora solo lo suficiente y necesario para la mayoría de las aplicaciones.
- *El lenguaje debe ser accesible a todos:* el sitio Internet de JAVA puede ofrecer gratuitamente todo el material necesario para:

  Aprender a programar en JAVA.

  Interpretar programas JAVA.

Compilar programas JAVA.

Acceder a múltiples ejemplos (applets) de programas JAVA.

1.2.2.3 TIPOS DE PROGRAMAS JAVA

En JAVA existen dos tipos de programas:

Applets: son pequeños problemas que se integran en las páginas Web, aportando interactividad y nuevas posibilidades a las mismas. Al acceder a uno de estas páginas, mediante un navegador compatible con JAVA, se carga el programa compilado (Applet) y se ejecuta en una zona de la página, previamente establecida. Esta ejecución utiliza los recursos que incorpora el propio navegador. De esta forma, a través de la red se puede ejecutar un programa residente en un servidor remoto.

a) Restricciones de los Applets

Por razones de seguridad, los applets tienen varias restricciones. Un applet no puede:

- Cargar librerías o definir métodos nativos.
- Leer ni escribir ficheros en el sistema anfitrión que está ejecutándolo.
- Hacer conexiones de red, excepto al sistema anfitrión del que procede.
- Ejecutar cualquier programa en el sistema anfitrión que está ejecutándolo.
- Leer ciertas propiedades del sistema.

Aplicaciones: son programas normales que se ejecutan directamente por el sistema operativo del usuario. La diferencia con los applets radica en el hecho de que pueden utilizar todos los recursos del leguaje y del usuario.

1.2.3 PROGRAMACIÓN CGI

1.2.3.1 ¿QUE SON LOS CGI?

Un CGI (Common Gateway Interface) es un programa que se ejecuta en el servidor por petición del navegador de un cliente. El CGI produce un resultado, el cual se envía al navegador que provocó la ejecución del programa.

Los CGI dan dinamismo a la Web. Las páginas Web puras (archivos HTML) son archivos de texto y, por tanto, estáticos. Sin embargo, si en lugar de pedir una página Web el navegador ejecuta un programa, éste puede generar la página "al vuelo" y decidir en el momento cómo va a ser la página.

Por ejemplo, imagine una página Web que muestre la hora como texto. Está claro que no se puede poner la hora con el editor de páginas Web, ya que cada vez que alguien vea la página la hora será distinta. La solución es crear un programa que se ejecute cada vez que alguien quiera ver la página. El programa genera la página Web en el momento en que se ejecuta y así coloca la hora correcta. Por ello los CGI añaden dinamismo a las páginas Web.

En principio los CGI se pueden realizar con cualquier lenguaje de programación, ya que pueden ser ejecutables (archivos .exe). Sin embargo, lo más recomendable es utilizar otros lenguajes más populares como el VisualBasic o JavaScript, nuestra

recomendación es que utilice el ASP (Active Server Pages) para programar aplicaciones de servidor.

1.2.3.2 EL DIRECTORIO CGI-BIN

Los CGI no pueden ser colocados en cualquier directorio y esperar que funcionen. Un CGI ha de estar en un directorio que tenga permisos de ejecución de scripts.

El directorio cgi-bin es el único por defecto preparado para admitir CGI. Dispone de permisos de ejecución y tiene eliminados los permisos de lectura y listado para proteger mejor los fuentes. Normalmente colocará ahí sus propios CGI.

Si, aún así, desea colocar CGI propios en otros directorios, puede hacerlo siempre que les asigne los permisos adecuados.

1.2.3.3 EL PERMISO DE ESCRITURA

Muchos de los CGI que están disponibles en Internet o que usted pueda crear, desearán escribir datos en archivos de texto u otro tipo de bases de datos. Sin embargo, por defecto no hay ni un solo directorio público en la Web con permisos de escritura.

La recomendación es que sus archivos de datos residan en el directorio **Data** que está fuera del sitio Web, aunque puede ser leído por los CGI. De esa forma evitará que

cualquiera pueda leer sus archivos de datos desde la Intranet. El directorio Data ya tiene los permisos de escritura, de forma que no deberá hacer nada especial si sigue esta recomendación.

De todas formas, hay CGI disponibles freeware en Internet que escriben en archivos que se encuentran en el mismo directorio del Script. Esta es una práctica poco recomendable. Para que estos CGI funcionen es necesario que al directorio en el que se encuentran se le asignen permisos de ejecución y de escritura simultáneamente. Y esto constituye un importante agujero de seguridad para su sitio Web.

1.2.3.4 ¿POR QUÉ NO FUNCIONAN LOS CGI?

La experiencia dice que en la gran mayoría de los casos es por culpa de los permisos. Si el CGI intenta hacer algo para lo que no tiene permiso (principalmente, escribir algún dato), terminará silenciosamente.

Otra causa de los problemas puede ser la extensión del archivo (tiene que ser .pl y no .cgi). Asegúrese de que el script no utiliza características de UNIX que no estén en Windows NT, como el MAIL.

Otra fuente de problemas son las rutas de directorio. En UNIX se escriben de la siguiente forma:

```
$mypath = '/usr/bin/mydir';
```

Sin embargo, en Windows NT hay que usar el separador de directorios \ y comenzar las rutas absolutas por la letra de unidad. Es decir, la ruta anterior podría quedar de esta forma:

```
$mypath = 'c:\usr\bin\mydir';
```

Suponiendo que existiese el directorio c:\usr. Hay muchos directorios estándar en UNIX que no existen en Windows NT.

Cuando utilice rutas de este tipo tenga mucho cuidado con el tipo de comillas que utiliza para la cadena. Si utiliza comillas dobles (a veces es necesario si el path utiliza una variable) entonces debe utilizar \\ para evitar la interpolación. Nuestro ejemplo quedaría de la siguiente forma con comillas dobles:

```
$mypath = "c:\\usr\\bin\\mydir";
```

Las rutas relativas pueden ser problemáticas en Windows NT. Le recomendamos para mayor seguridad que utilice siempre rutas de directorio absolutas.

Piense en las sentencias de su script que escriben en archivos o leen de ellos como las mayores candidatas a provocar el error.

Hay veces que el navegador nos juega malas pasadas porque guarda en el cache un mal funcionamiento del script y nos da un error sin intentar comunicar con el servidor. Esto puede localizarse por la rapidez con la que el navegador nos da el error.

Se recomienda tener a mano los dos navegadores (Explorer y Netscape) y alternarlos para ver si el problema es igual en ambos. En ciertas ocasiones un navegador se empeña en seguir mostrando error cuando el problema ya está resuelto.

El problema de programar CGI es lo difícil que resulta la fase de depuración. Para depurar un script que no funciona le recomendamos:

- Paciencia. Esto es lo más importante
- Escribir sentencias print por doquier para examinar valores de variables.
- Comentar sentencias hasta que funcione y luego ir quitando los comentarios de uno en uno hasta encontrar la que hace fallar el script

1.2.3.5 EL MÓDULO WIN32:ODBC PARA ACCESO A BASES DE DATOS

Excelente módulo para acceder a bases de datos ODBC de las que se dispone de su DSN. Puede encontrar la documentación en Instalación del módulo Biblioteca en el capitulo 2 de esta tesis.

Ejemplo:

```
use Win32::ODBC;
$data = new Win32::ODBC($dsn); #Creamos el objeto
if (not defined($data) ) {
print "Error de conexión con la base de datos";
exit;
}
$res = $data->Sql($sentenciaSQL); #Ejecutamos la sentencia SQL. Devuelve
undef si va bien.
if ( defined($res) ) {
print "Error en la sentencia SQL";
exit;
}
$data->FetchRow(); #Sacamos la primera fila del resultado
%val = $data->DataHash; #%val contiene los nombres de los campos como
claves
for $campos (sort keys %val) {
print "$campos "; #Mostramos los nombres de los campos
}
#Este bucle muestra los valores hasta terminar las filas
do {
%val = $data->DataHash;
for $campos (sort keys %val) {
print "$val{$campos} "; #Se muestran los valores
}
print "\n";
}
while ( $data->FetchRow());
```

1.2.4 BASES DE DATOS EN EL WEB

1.2.4.1 INTRODUCCIÓN

El sistema de acceso a bases datos a través de la Web utilizando la tecnología Microsoft, se denomina ADO (ActiveX Data Objects). Aquí se expone una introducción comprensiva en castellano. Para ver una completa referencia acerca de todo el sistema de bases de datos de Microsoft, consulte en www.microsoft.com/data. La referencia completa de métodos y propiedades de los objetos está también en la biblioteca en-línea de MSDN (msdn.microsoft.com). Ahí verá múltiples ejemplos en los que comprobará la flexibilidad del sistema ADO (al mismo resultado se puede llegar por vías distintas). Por motivos pedagógicos, aquí se expone una utilización más rígida.

1.2.4.2 TERMINOLOGÍA

DSN Data Source Name. Es un identificador único de la base de datos en el sistema. Al definir un DSN hay que especificar tanto la ruta completa del archivo de base de datos como el controlador adecuado a la misma (MSAccess, FoxPro, etc.). Una vez creado, es todo lo que necesitamos saber acerca de la base de datos para poder abrirla, consultarla, modificarla, etc.

ADO ActiveX Data Objects. Es una familia de objetos componentes dedicados a facilitar el acceso a bases de datos. El ProgID de cada uno de ellos se forma combinando ADODB. con el nombre del objeto (por ejemplo ADODB.Recordset,

ADODB.Connection, ADODB.Command, etc.). Por tanto, en VBScript los objetos se crean con sentencias tipo Set mirst = Server.CreateObject("ADODB.Recordset"), etc.

1.2.4.3 CREAR UN DSN

Una vez que tenga preparado su archivo de bases de datos (por ejemplo mibase.mdb) colóquelo mediante FTP en el directorio Data de su dominio. Después vaya al panel de control de su dominio y obtenga un DSN para dicho archivo. El DSN será el identificador con el que podrá conectar con su base de datos.

1.2.4.4 LOS OBJETOS COMPONENTES DE ADO

Todo el sistema ADO se basa en una serie de objetos cuyas propiedades y métodos hay que utilizar. Estos objetos están registrados en el sistema, sin embargo, no están predefinidos. Es decir, hay que crearlos utilizando Server.CreateObject

1.2.4.5 CONNECTION

Representa una conexión a una base de datos. Este es el primer objeto que debemos crear para poder conectar con la base de datos. Tanto el objeto ADO que nos permite acceder a los datos (Recordset) como el que nos permite realizar consultas (Command) disponen de una propiedad llamada ActiveConnection que es una referencia al objeto connection que enlaza con la base de datos a la que queremos atacar. De ahí que el primer paso sea crear la conexión.

a) Propiedades y métodos más relevantes

ConnectionString

Es una cadena de caracteres con la información necesaria para establecer una conexión con la fuente de datos. Por tanto, es la propiedad básica de este objeto. Aunque hay hasta 7 argumentos distintos que se pueden suministrar en esta cadena, los básicos son el DSN que identifica al archivo de base de datos y el login y password si existen. Los argumentos se separan con punto y coma.

Ejemplo: "DSN=midsn; UID=milogin; PWD=micontraseña"

Open

Abre la conexión con la base de datos. Si antes hemos asignado la propiedad ConnectionString, este método no necesita parámetros.

Close

Cierra la conexión con la base de datos.

Ejemplo:

```
<%
Set miconexion = Server.CreateObject("ADODB.Connection")
miconexion.ConnectionString = "DSN=midsn"
miconexion.Open
' .....
'.....
miconexion.Close
%>
```

1.2.4.6 RECORDSET

Este es el objeto ADO más importante ya que es con el que accederemos directamente a los datos de las tablas, tanto para leerlos como para modificarlos.

Un objeto Recordset representa una tabla, que puede ser una tabla física de la base de datos o bien una obtenida mediante una operación como un filtrado o sentencia SQL. En cualquier caso, el objeto representa a la tabla con todos sus registros, aunque sólo uno de ellos es el activo. El registro activo es en el que podemos leer o modificar los valores de los campos. También se le llama cursor.

a) Propiedades y métodos más importantes

Para una mejor compresión y puesto que son numerosos, los hemos dividido por categorías de utilidad.

Propiedades que hacen referencia al origen de los datos:

Estas dos propiedades deben ser asignadas a todo Recordset, pues le dicen la base de datos y la tabla de la que obtener sus datos.

- ActiveConnection

  Como se ha comentado antes a esta propiedad se le debe asignar un objeto connection que se haya creado previamente. Indicará al Recordset la base de datos en la que buscar su tabla.

- Source

Indica al objeto Recordset la tabla a la que representará. A la propiedad Source se le asigna normalmente una cadena de caracteres con el nombre de la tabla. Sin embargo, también es posible asignarle una sentencia SQL y entonces el objeto Recordset referenciará al resultado de aplicar dicha sentencia.

Ejemplo:

Aquí creamos en primer lugar un objeto Connection y luego un Recordset al que se asigna dicho objeto. Este ejemplo es una especie de esqueleto para los demás que sigan: Se supone que debe hacerse algo parecido a esto para abrir el Recordset. Es decir, en los restantes ejemplos escribimos sólo el código que iría en el lugar de los puntos suspensivos de este script.

```
<%
Set miconexion = Server.CreateObject("ADODB.Connection")
miconexion.ConnectionString = "DSN=midsn"
miconexion.Open
Set mirecordset = Server.CreateObject("ADODB.Recordset")
mirecordset.ActiveConnection = miconexion
mirecordset.Source = "Clientes"
mirecordset.Open
..... ' Operaciones con los datos
.....' de la tabla Clientes.
mirecordset.Close
miconexion.Close
%>
```

Propiedades que hacen referencia al número de registros:

- RecordCount

  Número de registros de la tabla a la que representa el objeto recordset. Ejemplo:

  ```
  <h3>Tenemos <%= rstClientes.RecordCount%> clientes registrados en nuestra base de datos</h3>
  ```

- EOF

  Acrónimo de End Of File. Vale TRUE si estamos en el último registro y FALSE si no. Se usa mucho como condición en bucles while, los cuales se ejecutan hasta llegar al último registro.

- BOF

  Acrónimo de Begin Of File. Vale TRUE si estamos en el primer registro y FALSE si no.

b) Métodos para mover el cursor (registro activo):

- MoveFirst

  Mueve el cursor al primer registro de la tabla

- MoveLast

  Mueve el cursor al último registo de la tabla

- MoveNext

  Mueve el cursor al siguiente registro

- MovePrevious

  Mueve el cursor al registro anterior

  Ejemplo de un bucle que recorre todos los registros de una tabla

```
<%
mirecordset.MoveFirst
do while not mirecordset.EOF
..... ' Tratamiento
..... ' de datos
mirecordset.MoveNext
loop
%>
```

c) Lectura y modificación de los campos del registro activo

La sintaxis para acceder a los datos de un campo del registro activo de un recordset es

mirecordset("Domicilio")

Esto se usa tanto para leer como asignar valores. En nuestro ejemplo, el objeto mirecordset representa a una tabla uno de cuyos campos tiene el nombre "Domicilio". Así, con la expresión

```
dom = mirecordset("Domicilio")
```

Leemos el valor del campo domicilio del registro activo y se lo asignamos a la variable dom. Con la expresión

```
mirecordset("Domicilio") = "C/ Bretón de los Herreros 19, 1º M"
mirecordset.Update
```

Asignamos un valor al campo Domicilio del registro activo. Tras la edición del registro, es necesario llamar al método Update. El motivo es que los cambios en el registro activo se realizan sobre un buffer (espacio de almacenamiento intermedio) y no sobre el registro propiamente dicho.

Ejemplo:

El recordset rstClientes representa a nuestra tabla de clientes que, entre otros, tiene los campos Provincia e IVA . El siguiente bucle recorre todos los clientes, comprueba su provincia y le asigna el IVA correspondiente: 0 para Canarias y 16 para los demás

```
<%
rstClientes.MoveFirst
do while not rstClientes.EOF
if rstClientes("Provincia") = "Las Palmas" or rstClientes("Provincia") =
"Tenerife" Then
rstClientes("IVA") = 0
else
rstClientes("IVA") = 16
end if
rstClientes.Update
rstClientes.MoveNext
loop
%>
```

d) Métodos para agregar o eliminar registros de la tabla

Delete

Eliminar el registro activo es muy fácil. Basta con invocar este método. Por ejemplo, este bucle elimina todos los clientes morosos de nuestra base de datos:

```
<%
rstClientes.MoveFirst
do while not rstClientes.EOF
if rstClientes("Deuda") > 0 Then
rstClientes.Delete
end if
rstClientes.MoveNext
loop
%>
```

AddNew y Update

Crear un nuevo registro involucra dos métodos: Primero AddNew crea un nuevo registro en blanco. Después asignamos valores a los distintos campos del registro. Por último invocamos el método Update para que se haga efectiva la incorporación del nuevo registro con los valores asignados.

En este ejemplo incorporamos un nuevo cliente a nuestra base de datos

```
<%
rstClientes.AddNew
rstClientes("Nombre") = "Pepe Gotera"
rstClientes("Dirección") = "Rue del Percebe, 13"
rstClientes("Localidad") = "Sildavia"
rstClientes("Profesión") = "Chapuzas a domicilio"
```

A algunos programadores esto les puede parecer confuso y poco estructurado. De hecho, parece preocupante que un webmaster diseñador gráfico y un programador de aplicaciones de bases de datos tengan que trabajar ambos sobre el mismo archivo.

Sin embargo, esto es sólo apariencia a primera vista. Una reflexión más profunda nos revela que la página ASP engloba tanto a las páginas Web como a los scripts CGI. En efecto, una ASP puede consistir sólo en código HTML (y entonces es lo mismo que una página Web normal) o sólo en código ejecutable (lo mismo que un CGI). Por tanto, la mezcla de código y HTML es opcional, pero muy utilizada por resultar práctica (por ejemplo, un formulario que se llama a sí mismo).

1.2.5.3 ¿QUÉ VENTAJAS Y DESVENTAJAS TIENE EL ASP CON RESPECTO A LOS CGI?

a) Ventajas:

- Se puede elegir el lenguaje de programación de entre estos tres: JavaScript, Visual Basic Script y PerlScript. Incluso se pueden tener scripts en distintos lenguajes dentro de la misma página ASP. El lenguaje escogido vale con cualquier navegador, ya que se trata de código que se ejecuta en el servidor. Al navegador sólo le llega HTML.
- Se pueden utilizar componentes del lado de servidor. Esto permite programar al estilo VisualBasic creando objetos y utilizando sus métodos y propiedades.

- Se puede acceder a bases de datos con objetos recordset de un modo muy parecido a como se hace en Visual Basic.
- Tiene persistencia de variables en memoria (entre distintas visualizaciones de páginas Web) que se pueden asociar a cada sesión de usuario o a la aplicación en su conjunto. Esto resuelve de forma elegante uno de los mayores problemas de la programación en la Web: El servidor Web no tiene memoria entre la visualización de una página Web y la siguiente.

b) Desventajas:

- Ninguna. Los partidarios del Perl pueden seguir utilizando dicho lenguaje al 100% (incluyendo bibliotecas) y aprovechar las nuevas características del ASP sin más que utilizar como lenguaje de programación el PerlScript.

1.2.5.4 CONCEPTO DE APLICACIÓN ASP

Una página ASP es sólo un script. Se denomina aplicación ASP al conjunto de páginas ASP, páginas Web, archivos gráficos, etc, dispuestos en un directorio y sus subdirectorios. Todo este conjunto de scripts y datos es el equivalente en la Web a la aplicación tradicional que al final se compila en un ejecutable.

Las aplicaciones ASP tienen una raíz que es el directorio en el que residen, que debe tener permisos de ejecución de scripts y una aplicación en la metabase del IIS. Los

servidores virtuales de Arsys disponen de un panel de control con funcionalidad para asignar permisos a directorios y crear aplicaciones.

1.2.5.5 CONCEPTOS BÁSICOS

a) Las marcas de código <%...%>

Todo lo que se escriba en una página ASP es por defecto código HTML que se envía directo al navegador del cliente, es decir, lo mismo que una página Web. Si se desea escribir código a ejecutar en el servidor, hay que encerrarlo entre las marcas <% y %>. Por ejemplo, fíjese en el siguiente fragmento de página ASP, donde hemos puesto en rojo la parte de los códigos ejecutables:

```
<%
nombre = Request.Form("nombre")
passw = Request.Form("passw")
if nombre = "root" and passw = "489602" then
boss = True
else
boss = False
End if
%>
<form method=POST action = p1.asp>
<p>Escriba su nombre:</p>
<p><input type="text" name="nombre"></p>
<p><input type="submit" name="enviar" value=" enviar "></p>
</form>
```

Cuando se solicite la página ocurrirá lo siguiente:

En primer lugar se preprocesa la página ASP en busca de códigos Server Side Include (por ejemplo <!--#include "pagina.html" -->). Esto hace que con una sentencia como la anterior se puedan incluir unas páginas en otras.

Se ejecuta el código de servidor. El código HTML que aparezca se considera que debe ser mostrado directamente.

Se envía el resultado al navegador del cliente, pero excluyendo el código de servidor. De esa forma el cliente no puede ver el código ejecutable de la página ASP.

b) Las marcas <SCRIPT> y </SCRIPT>

Una alternativa a las marcas <% y %> son las marcas <script> y </script>. Por ejemplo, el código anterior quedaría de la siguiente forma

```
<SCRIPT LANGUAGE=VBScript RUNAT=Server>
nombre = Request.Form("nombre")
passw = Request.Form("passw")
if nombre = "root" and passw = "489602" then
boss = True
else
boss = False
Endif
</SCRIPT>
<form method=POST action = p1.asp>
<p>Escriba su nombre:</p>
<p><input type="text" name="nombre"></p>
<p><input type="submit" name="enviar" value=" enviar "></p>
</form>
```

La marca <SCRIPT> se utiliza también para el código del lado de cliente (el JavaScript que se ejecuta en el navegador del cliente), por lo que es necesario el modificador RUNAT=Server con el fin de indicar que hay que ejecutarlo en el servidor. Además, existe el modificador LANGUAGE en el que indicamos el lenguaje en el que hemos escrito el código. Son válidos JavaScript (o JScript que es lo mismo), VBScript y PerlScript.

c) Elección del lenguaje de programación

Si utiliza para el código las marcas <SCRIPT> y </SCRIPT> puede elegir el lenguaje de programación mediante el modificador LANGUAGE. Esto le permite tener scripts en distintos lenguajes dentro de la misma página ASP.

Si desea elegir el lenguaje del código entre las marcas <% y %> debe comenzar la página ASP con una sentencia como la que sigue:

```
<%@ LANGUAGE=JavaScript %>
```

Esto hará que todo el código que se escriba entre <% y %> se considere en JavaScript. El lenguaje por defecto (el utilizado si no se escribe la sentencia anterior) es el Visual Basic Script.

d) Intercalar HTML entre el código ejecutable

Cuando se utilizan las marcas <% y %> para encerrar el código ejecutable, es posible intercalar HTML. Considere el siguiente código que corresponde a una página ASP completa. Puede copiar el código y guardarlo con el nombre p1.asp (se requiere este nombre para que funcione el formulario) en un directorio de su dominio que tenga permisos de ejecución de scripts y una aplicación definida. Se muestra en rojo el código ejecutable y en negro el H.T.M.L.

```
<html>
<head>
<title>Pequeña prueba</title>
</head>
<body>
<%
nombre = Request.Form("nombre")
if nombre = "" then
%>
<form method=post action = p1.asp>
<p>Escriba su nombre:</p>
<p><input type="text" name="nombre"></p>
<p><input type="submit" name="enviar" value=" enviar "></p>
</form>
<% else %>
Hola <b> "& nombre"& </b>. Bienvenido al mundo del ASP.
<%End if %>
</body>
</html>
```

Fíjese como se muestra una sentencia if then-else-end if que se encuentra fragmentada y entre sus partes aparece código HTML. La interpretación es clara: Si la condición es cierta se mostrará el código HTML que existe entre el if then y el else (el que

corresponde a un formulario). Si la condición es falsa, se mostrará el que existe entre el else y el End if (una frase).

e) Intercalando variables en el HTML

La construcción <%= variable %> intercalada entre el código HTML, hace que se muestre el valor de dicha variable. Por ejemplo, en el script anterior aparece la frase:

Hola <%= nombre%> . Bienvenido al mundo del ASP

Que hace que se muestre en negrita el valor de la variable nombre, que es lo que se haya metido en el formulario.

f) Formularios que se llaman a si mismos

Los datos de los formularios son gestionados por el programa que se incluye en la sección ACTION de la marca FORM. Como las páginas ASP son a la vez páginas Web y ejecutables en el servidor, podemos tener en ellas formularios que se llaman a si mismos. Curioso y en algunos casos muy útil como veremos a continuación. Las ventajas de esto son:

Compacto. Tenemos todo en un sólo archivo.

Permite aprovechar las variables del objeto Session para que el formulario tenga "memoria" de los valores introducidos la vez anterior y mostrarlos por defecto.

Un ejemplo de formulario que se llama a si mismo es el ejemplo mostrado anteriormente. Dijimos que había que guardarlo con el nombre p1.asp porque es el nombre que aparece en el ACTION del formulario. La lógica que sigue dicho script es la habitual en este tipo de casos:

Primero se obtienen las variables del formulario como si hubiese sido enviado por el usuario.

Después se comprueba si el ASP ha sido llamado directamente o si se trata de un envío de formulario (llamada del ASP a si mismo). En el ejemplo, la comprobación se hace examinando la variable que contiene el dato a introducir en el formulario.

Si el ASP ha sido llamado directamente entonces lanzamos el código que muestra el formulario.

Si el ASP ha sido invocado al pulsar el botón del formulario, entonces ejecutamos el código que procesa el formulario.

Es decir, el script ASP que se llama a si mismo tiene dos caras: En una muestra la página inicial y en la otra el resultado de procesar dicha página.

1.2.5.6 INCLUSIÓN DE PÁGINAS

a) Incluir páginas (Server Side Include)

De los comandos SSI (Server Side Include), las páginas ASP sólo soportan el #include, aunque ciertamente se trata del más importante. Este comando es útil si tenemos código (bien sea HTML o ejecutable) que se va a repetir en distintas páginas Web.

Por ejemplo, tenemos un pie de página común para todas nuestras páginas Web. En lugar de escribir el código en todas ellas, creamos un archivo con el pie de página y lo incluimos en todas las páginas de nuestra Web. No sólo ahorraremos trabajo, sino que si queremos modificar el pie de página, bastará con modificar un archivo y automáticamente se reflejará el cambio en toda la Web.

En el caso del código de servidor, resulta muy útil crear bibliotecas de funciones que se puedan reutilizar más adelante. Una forma de reutilizar funciones es guardarlas en archivos e incluir estos cuando sea necesario. Esto es mucho mejor que copiar y pegar la función. Ahorra trabajo y al igual que antes, si realizamos una mejora en la función se reflejará en todo el código que la llame.

El Visual Basic Script es un lenguaje que no tiene facilidades de inclusión de código. Esto se puede suplir mediante este comando. Sin embargo, si se programa en PerlScript, se pueden utilizar las facilidades del lenguaje para incluir código de otros scripts o de bibliotecas mediante las palabras clave requiere o use.

b) Sintaxis

La sintaxis del comando tiene dos variantes:

```
<!--#include file="archivo.asp"-->
```

En la variante file hay que escribir la ruta del archivo relativa al directorio en el que se encuentra el ASP que se está ejecutando. En nuestro ejemplo, archivo.asp está en el mismo directorio que el ASP que lo incluye.

```
<!--#include virtual="/dirweb/archivo.asp"-->
```

En este caso se escribe la dirección Web comenzando desde el raíz del servidor Web. En nuestro ejemplo, estamos incluyendo el archivo cuya dirección Web completa es:

```
www.midominio.com/dirweb/archivo.asp
```

Es decir, en el primer caso nos referimos a archivos según la estructura de directorios y en el segundo según la estructura de la Web.

El archivo a incluir no tiene por qué ser un ASP, puede tener cualquier extensión. Si se trata de archivos que sólo están pensados para ser incluidos, es una buena práctica de programación el colocarles la extensión .in

El preprocesado de los #include es previo a la ejecución del script, por lo que no puede haber includes condicionados a sentencias if then. Hay que imaginarse que cada include es sustituido por el archivo completo y que entonces es cuando se procesa el código ASP.

1.2.5.7 OBJETOS PREDEFINIDOS

Cada script ASP dispone desde el momento en que es ejecutado de una serie de objetos predefinidos. Estos objetos no hace falta crearlos. Se encuentran disponibles para cualquier script y permiten realizar funciones básicas como averiguar los parámetros pasados al Script, enviar información al usuario, guardar variables persistentes, etc.

Aquí exponemos un breve resumen de los más importantes con sus propiedades y métodos más utilizados.

a) Response

Se utiliza para enviar la salida del script, es decir, lo que verá el usuario en su navegador.

Métodos mas importantes

- Response.Write(cadena)

  Envía la cadena de caracteres al cliente. Es equivalente a intercalar código HTML entre el ejecutable. Así el siguiente código

```
<%
if num > 1000 then
Response.Write ("<h1>Se ha pasado de mil</h1>")
End if
%>
```

  Es totalmente equivalente a este

```
<% if num > 1000 then %>
<h1>Se ha pasado de mil</h1>
<% End if %>
```

  De hecho el HTML intercalado se implementa internamente mediante sentencias Response.Write.

- Response.Redirect (Url)

  Redirige la página ASP a la URL especificada. Por ejemplo, esto que sigue es una página ASP completa que simplemente redirige a la página www.pucesa.edu.ec

```
<% Response.Redirect ("http://www.pucesa.edu.ec") %>
```

La redirección se implementa mediante cabeceras HTTP que son distintas que las enviadas cuando se muestra una página Web. Eso significa que si se utiliza Response.Write o se intercala cualquier código HTML, ya no funcionará un posterior Response.Redirect porque se habrán enviado las cabeceras de mostrar una página Web, no las de redirección.

No obstante, existe una forma de "echarse atrás" y enviar un comando de redirección tras haber enviado comandos write. Esto implica el uso de la propiedad Buffer y los métodos Flush y Clear, como se explica a continuación.

- Response.Flush

  Envía de inmediato los datos del buffer. Si se estable a True la propiedad Response.Buffer, la salida del script (enviada mediante comandos Response.Write o HTML intercalado) no se envía directamente al navegador, sino que queda en un buffer (espacio intermedio de almacenamiento) del servidor. El método Response.Flush envía los datos del buffer al navegador.

Otra forma de enviar los datos del buffer es simplemente dejar que termine el script o invocar el método End.

- Response.Clear

  Borra los datos del buffer. Si se estable a True la propiedad Response.Buffer, la salida del script (enviada mediante comandos Response.Write o HTML intercalado) no se envía directamente al navegador, sino que queda en un buffer (espacio intermedio de almacenamiento) del servidor.

  Utilizar un buffer sólo tiene sentido si pensamos que nos podemos arrepentir de los datos enviados. Este método permite precisamente eso, anular todos los comandos Response.Write y HTML intercalado que hayamos utilizado desde que pusimos a TRUE la propiedad Response.Buffer

- Response.End

  Termina el script y envía el buffer si está a TRUE la propiedad Response.Buffer.

- Response.Buffer

  Si se pone en TRUE, se utiliza un buffer para la salida de datos, en combinación con los métodos anteriormente descritos.

A pesar de la insistencia en el Buffer, es algo que prácticamente no se usa a no ser que sea realmente necesario, pues vuelve lento el mostrado de la página Web. El único método realmente importante del objeto Response es Write.

b) Request

El objeto Request se utiliza para recoger los parámetros de entrada del script, es decir, los datos de los formularios o las variables pasadas mediante la interrogación en la URL. No tiene métodos ni propiedades dignas de mención, sino sólo tres colecciones muy importantes: Form, QueryString y ServerVariables.

- Colección Request.Form

  Mantiene la colección de parámetros pasados al script mediante el método POST, que es el usado preferentemente en los formularios, de ahí su nombre.

  Request.Form nos devuelve la colección de parámetros. Por tanto, la expresión admite la sintaxis de las colecciones de Visual Basic, es decir:

  Request.Form("Param") nos da el valor del parámetro de nombre Param siempre y cuando exista un sólo valor para dicho parámetro. Por ejemplo, nos daría el texto introducido en un control de formulario cuyo código HTML podría ser del estilo a <input type=text name="Param">.

Request.Form(3) nos da el valor del tercer elemento de la colección, es decir, el valor del tercer control del formulario.

Request.Form.Count nos da el número de parámetros pasados en el formulario.

Como existe la posibilidad de parámetros multivaluados, como una lista de selección múltiple, la expresión Request.Form("Param") puede ser a su vez una colección: La colección de valores del parámetro.

Por ejemplo, si en nuestro formulario tenemos una lista de selección múltiple, cuyo nombre sea Lista, la expresión Request.Form("Lista")[2] nos daría el segundo elemento seleccionado de la lista y Request.Form("Lista").Count el número de elementos que ha seleccionado el usuario. Su valor será 1 si se trata de un parámetro monovaluado y 0 si es un parámetro no definido en el formulario.

También es posible iterar sobre las colecciones anteriores mediante bucles For each. El siguiente ejemplo nos muestra todos los parámetros pasados al script y sus valores:

```
<%
For each param In Request.Form
Response.Write ("<p>")
Response.Write("El parámetro " & param & " toma el valor " &
Request.Form(param) )
Response.Write("</p>")
Next
%>
```

- Colección Request.QueryString

  Mantiene la colección de parámetros pasados al script mediante el método GET, que es el utilizado cuando se escribe la URL seguida del símbolo ? y de los parámetros con sus valores. Por ejemplo:

  http://www.pucesa.edu.ec/cgi-bin/miprog.asp?nombre=Andrés&apellido=López

  Invoca el script pasando las variables nombre (valor = Andrés) y apellido (valor = López). El símbolo & se utiliza como separador entre los distintos parámetros.

  Aunque no es lo habitual, también los formularios pueden utilizar el método GET. Para ello basta con ponerlo en la marca <FORM> de la página Web:

  <FORM method=GET action = "MiForm.asp">

  En este caso, los parámetros del formulario los obtendríamos en Request.QueryString en lugar de en Request.Form.

- Colección Request.ServerVariables

  Mantiene la colección de las variables de entorno del servidor Web. ¿Y qué es eso?. El lugar donde podemos encontrar valores muy interesantes que conoce el servidor

Web, como la dirección IP del usuario que está viendo las páginas, el login y contraseña si estamos en un directorio de acceso restringido, el nombre del dominio, el nombre del script, etc.

c) Server

Aquí empieza lo bueno. El objeto Server sólo tiene un método importante, pero es el más importante de todo el esquema ASP, ya que es el que permite crear objetos componentes y extender la funcionalidad del ASP de forma ilimitada. Con el Visual Basic Script del ASP y sin utilizar objetos componentes, no se puede hacer prácticamente nada; ni siquiera leer un archivo del disco. Toda la funcionalidad reside en objetos componentes ActiveX.

- El método Server.CreateObject

  Crea una instancia del componente de servidor que se le pase como parámetro.

- Server.CreateObject(ProgID)

  El parámetro ProgID es un identificador único del componente que suele darse en la forma Vendedor.Componente. Por ejemplo:

```
<%
Set herram = Server.CreateObject ("MSWC. Tools")
if herram.FileExists ("mipagina.html") Then
%>
<p>Esta es <a href="mipagina.html">mi página</a> </p>
```

```
<% else %>
<p>Lo siento. Mi página ha desaparecido</p>
<% end if %>
```

MSWC.Tools es el ProgID del objeto Tools de Microsoft, que viene con la instalación de ASP. No es un objeto predefinido como Response o Request, sino que hay que crearlo como cualquier otro. En VisualBasic los objetos se asignan a variables utilizando Set.

Después de creado, podemos acceder a sus métodos y propiedades. En este caso, utilizamos el método FileExists que nos dice si existe o no un archivo dada su URL relativa.

d) Session

Sólo para poder beneficiarse de este objeto predefinido en sus aplicaciones Web, hay proveedores que se pasan de UNIX a Windows NT. ¿Cuáles son entonces sus maravillosos métodos, propiedades o colecciones?. Veamos a continuación.

Lo interesante del objeto Session son las variables que nosotros podemos guardar en él. Esas variables permanecen entre distintas páginas Web y son únicas para cada usuario. Así, si usted guarda el nombre del usuario en una variable del objeto Session, podrá incorporarlo a todas las demás páginas, ya que ese dato no se pierde al terminar el

script. Además, aunque haya varios usuarios viendo simultáneamente las páginas no hay problema, ya que cada uno tiene un objeto sesión distinto.

```
Session("variable") = valor
```

Esto guarda el valor en la variable. Para recuperarlo más adelante en este u otro script distinto, sólo tendríamos que escribir Session("variable").

En PerlScript el objeto Session se implementa como un hash con la sintaxis $Session{variable} = valor

Ejemplo:

Puede guardar este script con el nombre p2.asp. Este ejemplo es prácticamente igual al p1.asp con la única diferencia de que se guarda el nombre del usuario en el objeto Session.

```
<html>
<head>
<title>Pequeña prueba del objeto Session</title>
</head>
<body>
<%
nombre = Request.Form("nombre")
if nombre = "" then
%>
<p>Por favor, escriba su nombre </p>
<form method = "Post" action="p2.asp">
```

```
<input type ="text" name = "nombre">
<input type = "submit" value = " Enviar ">
<%
else
Response.Write ("Su nombre es " & nombre & ". No se preocupe que no se me
va a olvidar.")
Session("Nombre") = nombre
end if
%>
</body>
</html>
```

Para escribir el nombre del usuario en páginas asp posteriores a esta, basta con que intercale el código

<%= Session("Nombre") %> en el HTML de la página.

e) Application

Es igual que el objeto Session y sirve para lo mismo (guardar variables) con la única diferencia de que las variables son únicas para la aplicación en su conjunto, no para cada sesión de usuario. Por tanto, en el objeto Application se deben guardar variables que vayan a ser comunes a todos los usuarios, no las específicas como su nombre.

Esto implica también que si deseamos modificar el valor de una variable guardada en el objeto Application, es necesario bloquear el objeto para evitar que desde otra sesión se acceda simultáneamente a la misma variable. Para ello existen los métodos Lock y Unlock

El método Application.Lock

Bloquea el objeto impidiendo que otros clientes modifiquen cualquier variable guardada.

El método Application.Unlock

Desbloquea el objeto permitiendo que otros clientes modifiquen las variables del objeto Application.

Ejemplo:

Una variable típica para guardar en el objeto Application es el número de visitas a la Web. La variable no debe estar asociada a un cliente en concreto, sino que debe ser general. En la página en cuestión pondríamos el siguiente código:

```
<%
Application.Lock
Application("NumVisitas") = Application("NumVisitas") + 1
Application.Unlock
%>
```

Cuando queramos mostrar el número de visitas recibidas, bastará con que insertemos

```
<%= Application("NumVisitas") %>.
```

f) Los eventos de los objetos Session y Application y el archivo global.assa

Los objetos Application y Session tienen eventos al estilo de los formularios de Visual Basic, sólo que mucho más limitados. De hecho cada objeto sólo tiene dos eventos: uno que se lanza cuando es creado y otro cuando es destruido.

Eso significa que podemos escribir código de inicialización que se ejecutará cada vez que un usuario acceda por primera vez a nuestras páginas (creación de un objeto Session) o cuando acceda el primer usuario (creación del objeto Application). Así mismo, podemos escribir código de limpieza justo cuando se termine la sesión de un usuario o al terminar la aplicación en su conjunto.

El código de estos eventos ha de estar en un archivo de nombre global.asa que debe estar en el directorio raíz de la aplicación. El archivo global.asa es opcional y puede contener tres tipos de información:

Eventos de creación y destrucción de aplicación y sesiones.

Marcas <OBJECT> para crear objetos de servidor con alcance de sesión o aplicación (Método alternativo al de Server.CreateObject).

Bibliotecas de tipos de objetos componentes.

Los eventos de Session y Application han de estar en marcas <SCRIPT> como se indica en el ejemplo a continuación:

Ejemplo:

```
<SCRIPT LANGUAGE=VBScript RUNAT=Server>
sub Application_onStart
Application("NumSesionesActivas") = 0
Application("NumSesionesTotales") = 0
End sub
sub Session_onStart
Application.Lock
Application("NumSesionesActivas") = Application("NumSesionesActivas") + 1
Application("NumSesionesTotales") = Application("NumSesionesTotales") + 1
Application.UnLock
End sub
sub Session_onEnd
Application.Lock
Application("NumSesionesActivas") = Application("NumSesionesActivas") -1
Application.UnLock
End sub
sub Application_onEnd
End sub
</SCRIPT>
```

Aquí se utilizan los eventos para guardar el número de sesiones activas y totales de nuestra aplicación, es decir, el número de clientes que nos están viendo en un momento dado y el total que nos han visitado. Como se trata de datos globales y no ligados a una sesión, hay que guardarlos en el objeto Application y cuando se quieren modificar, es necesario bloquear primero dicho objeto.

¿Cuánto dura una sesión y como funciona?

Lógicamente el servidor Web no puede saber si el usuario está todavía leyendo la última página que descargó de la Web o se ha ido ya a dormir. De ahí que la duración de una sesión se mida por el tiempo de inactividad del usuario. Para medir éste el objeto

Session dispone de una propiedad llamada Session.Timeout en la que el tiempo se mide en minutos.

Cuando un usuario de Internet ve por primera vez cualquier página ASP de nuestra aplicación, el servidor Web automáticamente le envía un cookie y crea para él un objeto Session con llamada al evento de creación que se encuentre en global.asa. Si pasan los minutos indicados en Session.Timeout sin que el usuario haya visto una nueva página, la sesión se da por concluida y el objeto Session del usuario es destruido.

El cookie es una pieza de información que el servidor Web envía al cliente para que se guarde en su disco duro. Cuando el cliente vuelve al sitio Web reenvía el cookie al servidor y de esa forma éste puede distinguir a los usuarios y hacerles un seguimiento individual.

g) Objetos componentes: alcance y creación

Los objetos que se crean mediante Server.CreateObject (ProgID) tienen un alcance a nivel de script, es decir, cuando termina el script los objetos son eliminados. Sin embargo, en ciertas ocasiones será conveniente que los objetos tengan duración de sesión o incluso de aplicación.

Hay dos formas de dar alcance de sesión o de aplicación a un objeto:

- Guardarlo en una variable del objeto Session o Application
- Crearlo en global.asa mediante una marca <OBJECT>

Ejemplo método 1:

```
<%
' Para crearlo con alcance de sesión lo guardamos en el objeto Session
Set Session("miobjeto") = Server.CreateObject("MPX.Tool")
.....
.....
' Para verlo después en otra página ASP
Set miobjeto = Session("miobjeto")
miobjeto.show
%>
```

Ejemplo método 2:

Este procedimiento es el más cómodo. En primer lugar escribimos lo siguiente en el global.asa

<OBJECT RUNAT=Server SCOPE=Session ID=miobjeto PROGID="MPX.Tool">

</OBJECT>

Y ahora ya nos podemos referir al objeto en cualquier página ASP sin más que llamarlo por su nombre:

```
miobjeto.show
```

En general, conviene no abusar del alcance de Session o de Application para los objetos ya que además de consumir recursos innecesariamente podemos perder cualidades.

Por ejemplo, si queremos conectarnos a una base de datos mediante un DSN no es conveniente dar alcance de Session al objeto Connection y dejar la base de datos abierta. Además de consumir recursos del servidor, perderemos la opción de pooling del ODBC.

1.2.5.8 LENGUAJE PERL

Es un lenguaje de ordenadores interpretado. Su nombre significa "perla" y se supone que es el acrónimo de *Practical extraction and report language* (lenguaje práctico para extracción e informes).

Al ser interpretado, los programas se denominan scripts y son archivos de texto. El Perl es el lenguaje más usado para programar CGIs porque al ser interpretado es más portable y porque sus características lo hacen ideal para ello.

Una de las ventajas del Perl es que al ser interpretado es muy portable. Sin embargo no lo es del todo. Hay funciones de la biblioteca estándar que sólo existen en sistemas Unix y no en Windows NT.

Debe tener esto en cuenta y que no se puede garantizar que los scripts en Perl que encuentre en Intenet vayan a funcionar sin cambios en Windows NT. Hay algunos que

sí, pero normalmente es porque están diseñados con idea de portabilidad desde el principio.

Además, los CGI de terceros casi siempre requieren una adaptación previa (customize) para que funcionen. El motivo es que suelen depender de parámetros como rutas de directorios, nombres de archivos, etc. Por suerte, la zona de personalización suele estar clara en la mayoría de los scripts y consiste simplemente en cambiar valores de ciertas variables.

Tenga en cuenta los siguientes puntos si piensa utilizar scripts en Perl que están escritos para Unix:

En Unix el intérprete de un script se lanza según la información de la primera línea, mientras que en Windows NT es según la extensión del archivo. Por ello, *todos los scripts CGI en Perl deben llevar la extensión .pl* y no la extensión .cgi.

En Windows NT no existe el comando MAIL que envía archivos por e-mail mediante una tubería. Aquí es necesario utilizar Blat.

El servidor Web de Microsoft (IIS) es "Non parsed header" (NPH), lo que significa que cuando ejecuta un CGI no envía ninguna cabecera. Muchos scripts para UNIX asumen que el servidor web enviará la cabecera, por lo que tal vez deba retocarlos. Los envíos de información deberían comenzar de esta forma (o con otra cabecera):

```
print "HTTP/1.0 200 OK\n";
print "Content-type: text/html\n\n";
```

El módulo de biblioteca CGI.pm resuelve estos problemas de forma elegante.

1.3 SERVICIOS Y POLÍTICAS PARA SERVIDORES WEB

1.3.1 PROTOCOLO TCP/IP

1.3.1.1 INTRODUCCIÓN

Los protocolos de red son los mecanismos que se emplean en computadoras o en otros componentes de la red para intercambiar información de forma comprensible para las dos partes que intervienen en una comunicación. Esta forma de comunicación tiene reglas y normas que están sujetas a un determinado convenio; este convenio debe ser preciso ya que las computadoras carecen de la flexibilidad para gestionar variaciones o excepciones. Los protocolos poseen tres elementos comunes que los definen: dirección, servicios y formato.

1.3.1.2 DIRECCIÓN

La dirección es un elemento del protocolo que identifica la siguiente ubicación de procesamiento a la cual debe enviarse la información. El protocolo TCP emplea un número de puerto para identificar la aplicación de servicio que recibe los datos entrantes.

El protocolo TCP no precisa de un elemento de dirección para enviar los datos porque lo transfiere en todos los casos al IP. La figura # 1 demuestra los tipos de gestión de direcciones que emplea el protocolo TCP/IP.

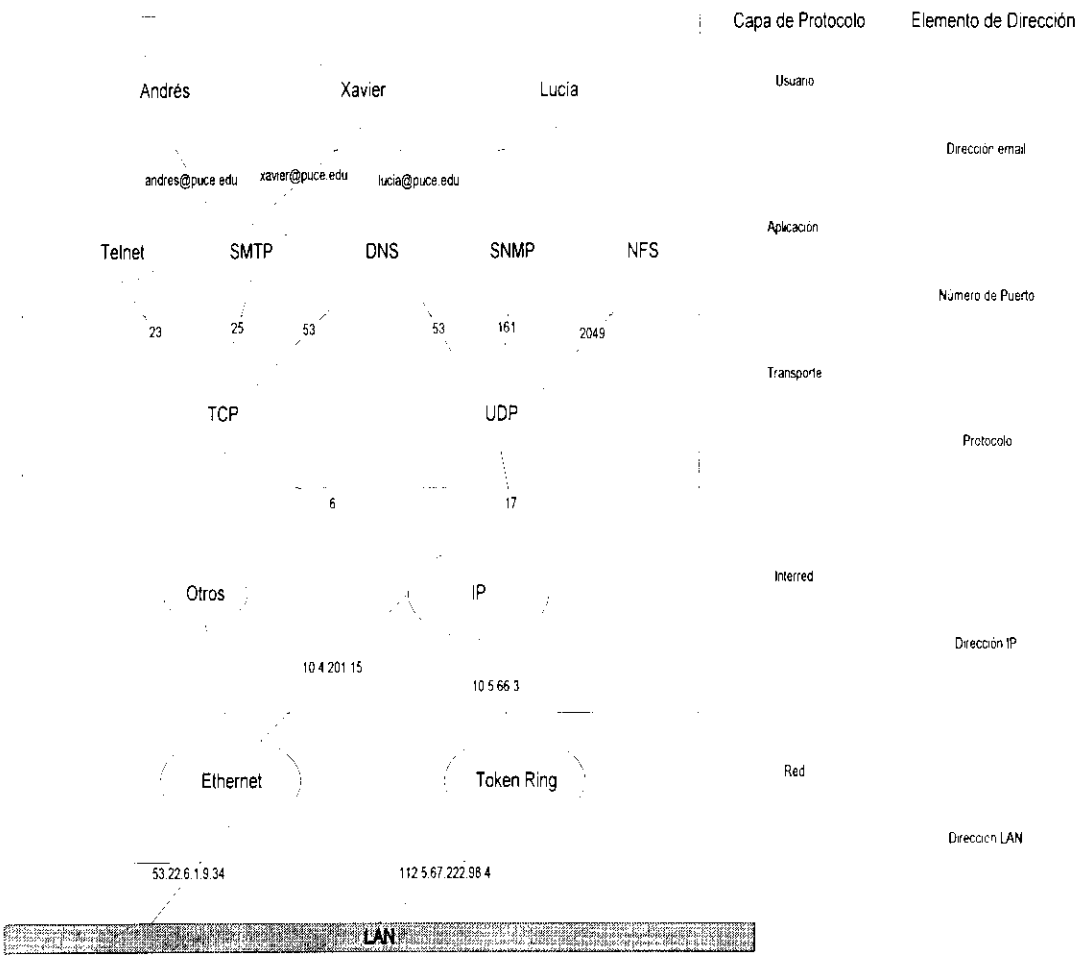


Figura # 3 Capas de la Red

1.3.1.3 FORMATO Y ESTRATIFICACIÓN DE PROTOCOLOS

Cada protocolo incorpora un formato definido par la información que envía o recibe. De este modo sabe donde debe buscar la información o los datos del usuario dentro del flujo de datos.

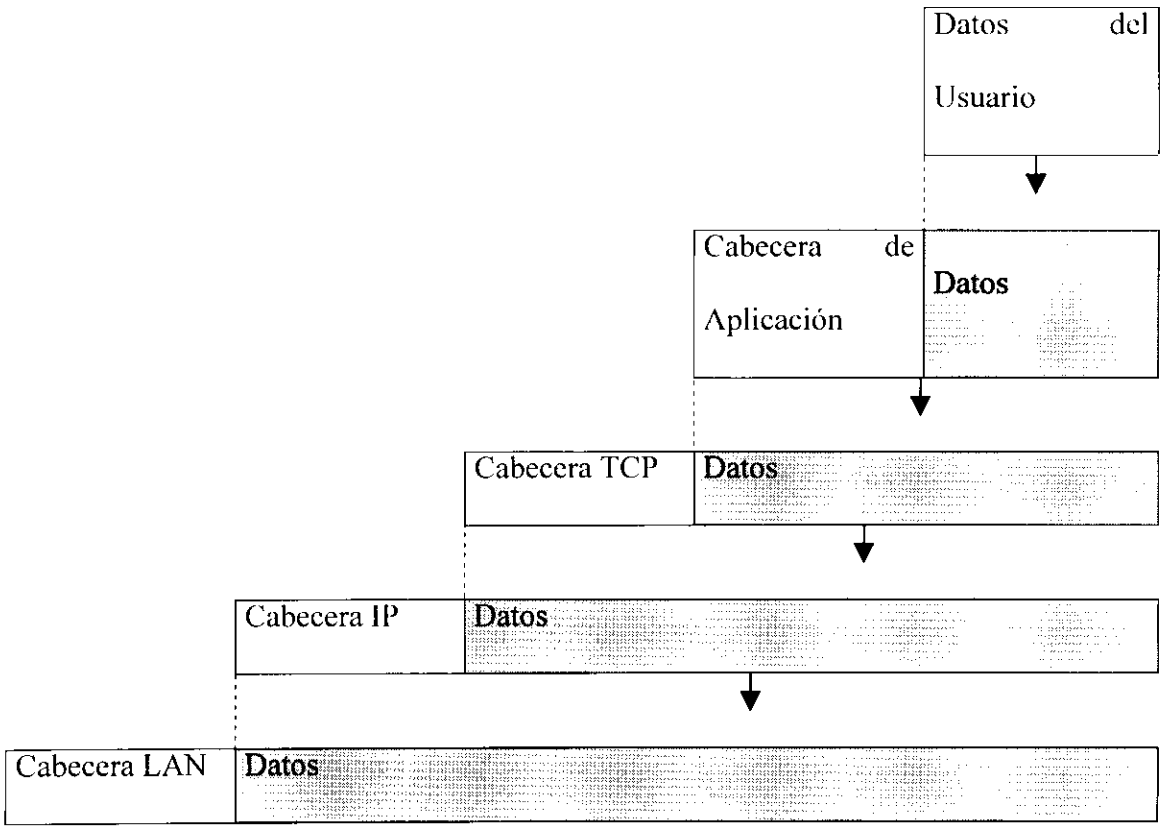


Figura # 4 Estructura de un Paquete TCP

La Figura # 4 muestra la jerarquía de los protocolos TCP/IP junto con los elementos de dirección asociados con los mismos que son necesarios para atravesarla. Cada nivel de protocolo añade a los datos su propia información de cabecera a medida que la información fluye desde un usuario hacia la red. Esta combinación se convierte entonces en los datos procesados por el protocolo que se encuentra en el nivel inmediatamente inferior. En el Host destinatario el proceso se invierte. En cada capa de

protocolo, las cabeceras son empleadas por el nivel en cuestión, tras lo cual se "separa" la cabecera y se envían los datos restantes a la cabecera inmediatamente superior.

La encapsulación de información de protocolo en la porción de datos del protocolo más bajo que recibe el nombre de estratificación de protocolos. Esto permite que, en las máquinas de origen y destino, la misma capa de protocolo comparta datos específicos al protocolo sin interferir con otros protocolos.

1.3.1.4 CAPA DE APLICACIONES

La capa de aplicaciones es aquella donde se lleva a cabo el servicio solicitado por el usuario. Algunos ejemplos de aplicaciones que operan a este nivel son Telnet, FTP, correo, navegadores WWW y ping. Esta capa se comunica con el usuario o el Host para obtener información sobre la dirección del Host remoto y sobre los datos a enviar. En el Host del usuario, la aplicación recibe el nombre de aplicación cliente. En el Host remoto la aplicación recibe el nombre de aplicación servidor.

1.3.1.5 CAPA DE TRANSPORTE

Los dos protocolos de la capa de transporte son TCP y UDP. Esta capa asegura que los datos del usuario llegan a al aplicación que se encuentra en el Host remoto. TCP es mucho más fiable que UDP. Esta capa ofrece un servicio orientado a la conexión.

1.3.1.6 CAPA DE INTERRED

Los modelos de protocolos incorporan una única capa de red, la que no se adapta completamente bien a la pila de protocolos TCP/IP. IP incorpora una capa de red LAN (Figura # 3) que proporciona un sistema de direcciones para permitir la comunicación entre redes. Dentro de TCP/IP, IP proporciona la capa de interred.

1.3.1.7 CAPA DE RED

La capa de red tiene la misión de entregar el paquete IP al Host apropiado. En esta capa, la dirección IP de los paquetes salientes se correlaciona con una dirección de red, que incluye la cabecera de la red. La red LAN retransmite el paquete hacia la red en donde el Host apropiado lo recoge.

Aplicación (7) Presentación (6) Sesión (5)	HTML, CGI, JPEG, GIF, Java, JavaScript	FTP (Transferencia de Archivos)	Telnet (toma de control a distancia)	SMTP (correo)	POP (correo)	NMTP (Noticias)	SNMP (Administración)	NFS (archivos compartidos)	DNS (resolución de nombres)			
	HTTP (Web)											
Transporte (4)	TCP						UDP					
Red (3)	IP											
Enlace (2) Física (1)	X25	Frame Relay	Ethernet	Slip PPP por línea especializada	PPP por red telefónica	RDSI	Token Ring	ATM	FDDI			

Tabla 3 Capas de la Red

1.3.1.8 TRANSMISSION CONTROL PROTOCOL (TCP)

TCP es el protocolo de transporte más habitual empleado en Internet. Proporciona una conexión virtual entre dos aplicaciones que se comunica entre sí. La aplicación de red envía una solicitud a TCP a fin de establecer una conexión con una aplicación situada en un Host remoto. El cliente TCP se comunica entonces con el TCP del Host remoto, tras lo cual se establece una conexión. La aplicación de red entonces puede escribir un flujo de datos a TCP que se entrega íntegramente a la aplicación remota en el orden original. Este protocolo no se preocupa de los detalles de la red excepto de la dirección IP y de los números de puerto TCP, además, es seguro debido a que confirma la comunicación entre ambas direcciones antes de enviar los datos.

a) Números de Puerto

TCP utiliza los números de puerto comprendidos entre 1 y 65.535 para identificar las aplicaciones que se encuentran en cada uno de los extremos de la conexión. Los números de puerto están reservados para las aplicaciones servidor; sin embargo los servidores pueden usar números de puerto más altos. Las aplicaciones cliente reciben dinámicamente números de puerto mayores que 1023. Pero ciertas aplicaciones tienen números de puerto ya asignados como por ejemplo Telnet que tiene asignado el número de puerto 23. De todos modos esta combinación de números de puerto origen y destino, así como el número del protocolo TCP se convierten en el identificador único de esta conexión, en donde se cumple que las direcciones IP en Internet deben ser únicas.

b) Servicios

TCP proporciona un servicio orientado a la conexión. Transporta fiablemente los datos del usuario hacia la aplicación remota. Cada segmento de los datos del usuario se etiqueta con una suma de comprobación y un número de secuencia, de 32 bits y aleatorio, para asegurar que ningún dato se modifique o extravíe. Con la capacidad dinámica del IP, los datos llegan desordenados pero TCP los ordena correctamente antes de entregarlos a la aplicación remota y si el dato se extravía entonces TCP pide que se lo retransmita inmediatamente. El número de secuencia también es de regreso hacia el cliente, con lo cual cada conexión posee dos números de secuencia exclusivos, uno para cada dirección.

c) Formato

La tabla # 4 muestra el formato de la cabecera TCP. La longitud es de 20 bytes además de las opciones.

Número de Puerto Origen	Número de Puerto Destino	Número de Secuencia	Número de Secuencia ACK	Longitud de Cabecera	Indicadores TCP	Tamaño de ventana	Suma de Comprobación	Puntero de Segmento	Opciones

Tabla 4 Cabecera TCP

1.3.1.9 USER DATAGRAM PROTOCOL (UDP)

Este es un protocolo desprovisto de elementos superfluos. El único servicio real que presta es asegurar que los datos son dirigidos hacia la aplicación apropiada; Es más eficaz que TCP debido a que no realiza todas las tareas que TCP presta. UDP es

necesario cuando uno quiere enviar los datos mediante una conexión rápida y breve de un paquete al contrario de TCP que necesita al menos tres paquetes para la misma tarea.

UDP no tiene corrección de errores, retransmisión ni reorganización de paquetes, esta simplicidad determina que UDP sea más vulnerable que TCP debido a que la ausencia de conexión y números de secuencia podría permitir que un atacante pueda enviar elementos perniciosos, mediante un envío unidireccional, al Host y modificar elementos o programas residentes en el Host.

UDP esta adaptado a las aplicaciones de pregunta - respuesta y sirve principalmente para las aplicaciones de servicios IP: DNS (Domain Server Name), RIP (Routing Information Protocol); también sirve para aplicaciones de "tiempo real" como telefonía y vídeo – conferencia que ocupan todo el ancho de banda posible y si un paquete se pierde o esta dañado no necesita de retransmisión; todo esto permite una transmisión más rápida y ligera pero menos segura.

a) Servicios

UDP no proporciona muchos servicios por su sencillez, fue diseñado con pocas características para permitir una excelente velocidad de conexión. Utiliza el mismo puerto que TCP.

b) Formato

La longitud de la cabecera es de 8 bytes, así:

Número de Puerto Origen	Número de Puerto Destino	Longitud UDP	Suma de Comprobación

En algunos casos los programadores tienden a desactivar la suma de comprobación para aumentar la velocidad de transmisión

1.3.1.10 INTERNET PROTOCOL (IP)

El propósito del protocolo IP es facilitar la comunicación entre dos Hosts. IP es lo que se conoce como un servicio de datagrama informal. Recibe esta calificación porque los protocolos de nivel superior no deben depender de IP para poder entregar un paquete a la vez. IP hace lo que puede par entregar el paquete en el Host destino solicitado pero, si falla en el intento, simplemente se deshace del mismo. Cuando una línea de comunicación esta ocupada o un router se ve incapaz de transmitir paquetes comienzan a descartarlos. Esto es normal debido a que TCP retransmite los paquetes perdidos durante la comunicación, si averigua que este no ha llegado a su destino.

IP tampoco garantiza que los paquetes lleguen en el orden con que se envían. Los paquetes IP pueden recorrer rutas diferentes hacia un mismo destino, de manera que un paquete enviado después puede llegar antes que otro paquete enviado previamente. Además los paquetes transmitidos pueden llegar desordenados, que demuestran que IP no es muy recomendable pero si muy robusto y se complementa con TCP para garantizar la comunicación.

1.3.1.11 DIRECCIONES IP

La dirección IP tiene una longitud de 4 bytes, para reconocerla con mas facilidad se representa habitualmente mediante una notación de puntos, en la que cada byte de dirección esta separado con un punto. Dado que cada byte puede representar un número hasta 255 cada número de IP puede ser representado entre 0 y 255. Cada Host conectado a Internet debe poseer una dirección exclusiva, que consta de dos elementos: una porción correspondiente a la red y otra correspondiente al Host. La Tabla #5 muestra de que forma se desglosan las direcciones de red en función del primer byte [2].

Clase de Dirección	Rango del primer byte	Número de bytes de red	Número de bytes de Host	Número de Direcciones de Red	Número de Direcciones de Host
A	De 0 a 1127	1	3	128	16.777.216
B	De 128 a 191	2	2	16.128	65.536
C	De 192 a 223	3	1	2.097.152	256

Tabla 5 Direcciones IP

a) Servicios

IP no ofrece muchos servicios. Como se ha mencionado, su función principal es entregar un paquete IP en el Host de destino, aunque no la garantiza. Debido a que es un

[2] Las direcciones IP mayores a 223 se usan para propósitos especiales de transmisión múltiple

servicio sin conexión, desconoce el concepto de paquete perdido, por lo que no soporta retransmisión.

Pero lo que si incorpora es la fragmentación de paquetes IP en paquetes de menor tamaño cuando un protocolo de red inferior no puede soportar el paquete completo. Cada fragmento recibe una copia de la cabecera IP y un segmento de los datos, entonces el Host destino se encarga de ensamblar el paquete antes de pasarlo a TCP cuando recibe dichos paquetes fragmentados.

b) Formato

La tabla # 4 demuestra el fragmento de un paquete IP, el campo correspondiente al protocolo identifica al protocolo superior que debe recibir los datos del paquete. La suma de comprobación es para asegurar la integridad de la cabecera IP mas no de los datos.

Versión	Longitud de Cabecera	TOS	Longitud de Paquete	Identificador de Paquete
Indicadores		Desplazamiento de Fragmentos	TTL	Número de Protocolo
Suma de Comprobación		Longitud de Cabecera	Dirección IP origen	Dirección IP destino

Tabla 6 Paquete IP

1.3.1.12 RUTEADO DE PAQUETES IP

El Host origen y todos los routers no necesitan conocer la ruta exacta para enviar el paquete IP hacia su destino final, solo se necesita conocer la dirección de red del siguiente router, esto se conoce como salto.

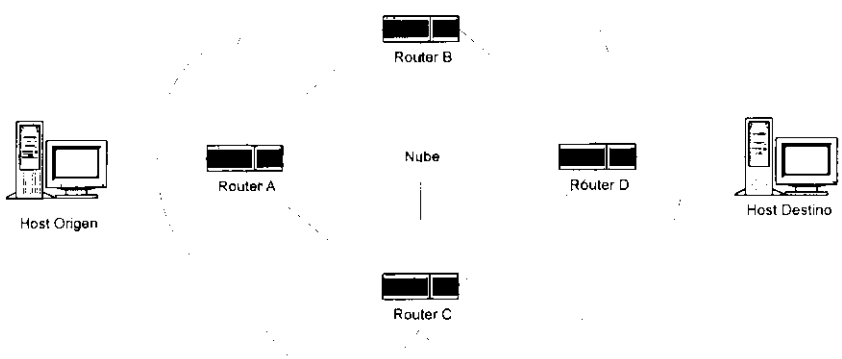


Figura # 5 Routeado de Paquetes

El Host origen sabe como dirigir hacia el router A todos los paquetes que no están destinados a los Hosts de la red local. La tabla de ruteado del router A tiene dos entradas correspondientes a la dirección del Host destino. La ruta principal es hacia el router B, pero si la línea esta inhabilitada o demasiado ocupada, puede pasar el paquete hacia el router C. El router B puede pasar los datos hacia el router D o el router C. Al final el router D recibe el paquete, el cual es transferido al Host destino. Para que esto funcione, los routers deben conocer a otros routers la ubicación del Host IP y las direcciones de red.

1.3.1.13 FLUJO DE PAQUETES

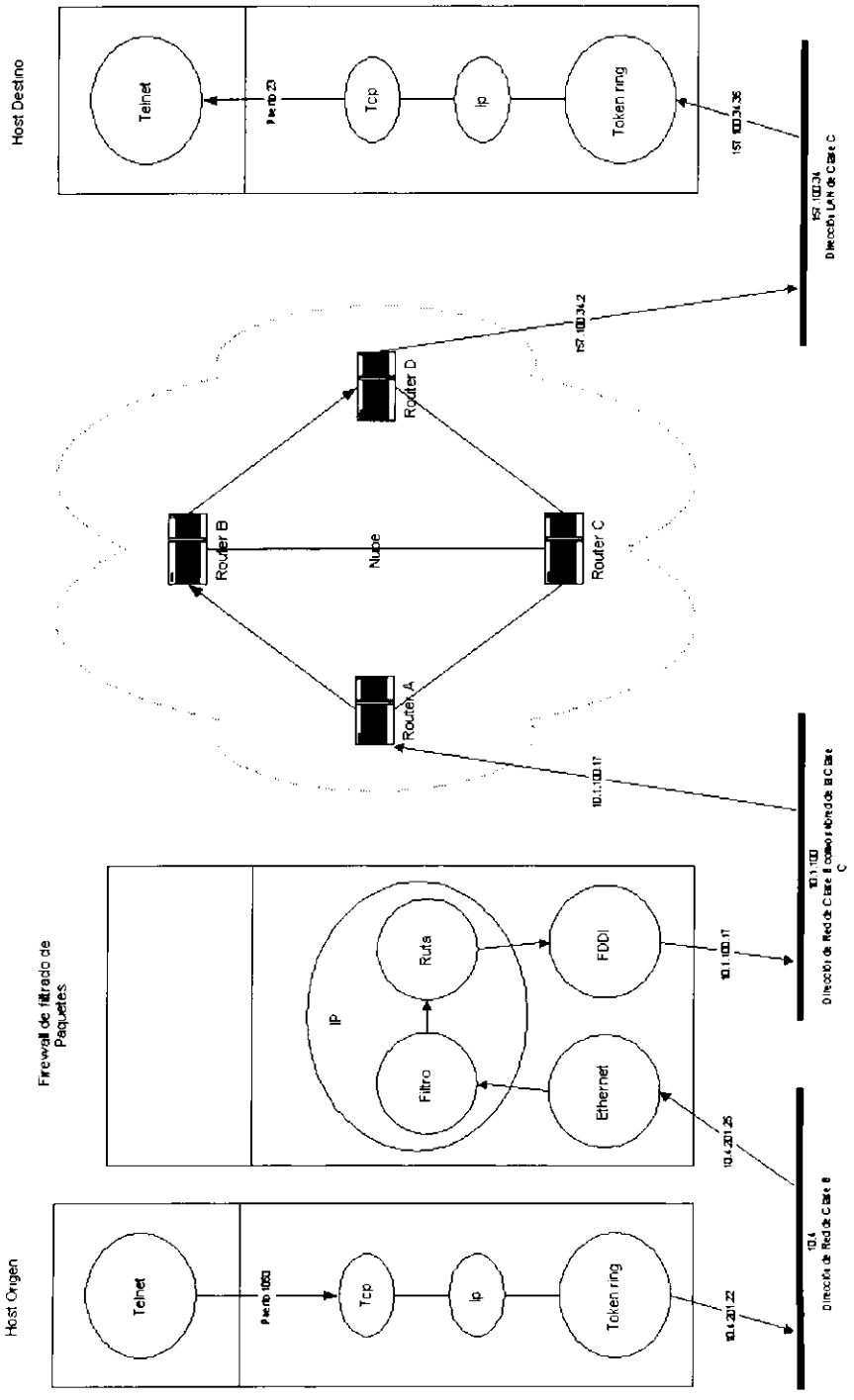


Figura # 6 Flujo de Paquetes

1. El usuario introduce el comando:

   telnet ambato.net.ec

2. El cliente Telnet busca la dirección 157.100.34.35 correspondiente a ambato.net.ec
3. El cliente Telnet solicita a TCP que inicie una conexión hacia el puerto 23 de la dirección 157.100.34.35.
4. TCP asigna el puerto disponible 1050 al cliente Telnet.
5. TCP construye una cabecera TCP con el puerto origen 1050 y un puerto destino 23, activa el indicador syn y desactiva el indicador ack.
6. TCP envía la cabecera a IP sin información del cliente Telnet.
7. IP toma la cabecera TCP y anexa una cabecera IP que contiene una dirección origen IP 10.4.201.22.
8. IP averigua que la dirección no se encuentra en la LAN local, por lo que dirige el paquete hacia el CortaFuegos.
9. IP envía el paquete hacia el controlador Ethernet junto con la dirección IP del Cortafuegos.
10. El controlador Ethernet añade al paquete una cabecera Ethernet que contiene una dirección Ethernet que corresponde a la dirección IP del Cortafuegos que se le ha proporcionado. El paquete se retransmite hacia la LAN Ethernet.
11. El controlador Ethernet del Cortafuegos acepta el paquete, "separa" la cabecera Ethernet y transfiere el paquete IP al software de IP.
12. El software de filtrado IP decide si debe dar paso al paquete o descartarlo.

13. Si el paquete es aceptado es dirigido hacia el router A de la forma descrita en los pasos 8 al 10 bajo una red FDDI.
14. El router A envía el paquete, por la ruta de preferencia, al router B.
15. El router B envía el paquete al router D, que lo envía al Host destino.
16. El software Token Ring del Host destino separa la cabecera Token Ring y envía el paquete IP a IP.
17. IP consulta la dirección y determina que es el Host local.
18. IP consulta el número del protocolo y ve un 6, que indica TCP. Separa la cabecera IP y transfiere el paquete a TCP.
19. TCP detecta que syn esta activado y ack desactivado, por lo que asume que es una nueva conexión [3].
20. TCP notifica a la aplicación servidora Telnet que ha recibido una solicitud de conexión desde el Host 10.4.201.22. Pregunta si se acepta la conexión.
21. La aplicación Telnet ordena a TCP que acepte la conexión.
22. TCP entonces crea una cabecera TCP con el bit syn activado y el bit ack también. Incluye un puerto de destino 1050 y un puerto origen 23. TCP entonces envía el paquete a IP.
23. IP entonces crea una cabecera IP con una dirección origen 157.100.34.35 y una dirección destino 10.4.201.22
24. Este paquete es sometido aun proceso similar a fin de regresar al Host origen.

1.3.2 DNS (DOMAIN SERVER NAME)

1.3.2.1 HISTORIA DEL DOMAIN NAME SERVER

En los años 1970 la red inicial de INTERNET "***ARPANET***" fue una organización pequeña de unos pocos cientos de Hosts. Un simple archivo "***/etc/hosts***" contenía toda la información que fue necesaria para poder resolver los nombres de máquina en direcciones IP.

La estructura de este archivo es:

; File /etc/hosts

; Numero IP Nombre Completo [Aliases1|2|3...]

pucesa.edu.ec mail.pucesa.edu.ec pucesa.edu.ec

Con el crecimiento de servidores, computadoras, etc. la solución de mantener en cada máquina la tabla de hosts realmente se imposibilita en tener un control exacto de toda la red, así nace el *Domain Name Server.*

[3] Cuando estos indicadores están configurados de otra forma, TCP registra su antememoria para encontrar alguna concordancia en direcciones de origen y destino. SI no hay ninguna, envía una señal de reiniciación y da fin a la conexión.

1.3.2.2 DEFINICIÓN

Es un mecanismo que permite convertir el nombre simbólico de las máquinas, por ejemplo: www.pucc.edu.ec en una dirección IP real: 192.188.55.11; y al revés, es posible obtener la dirección simbólica a partir de la dirección IP.

Funciona mediante una base de datos distribuida, el conjunto de nombres se reparte a través de un conjunto de servidores DNS de una red; en una Intranet pequeña o mediana, puede bastar un servidor DNS único.

Los nombres simbólicos posibles forman una jerarquía en árbol:

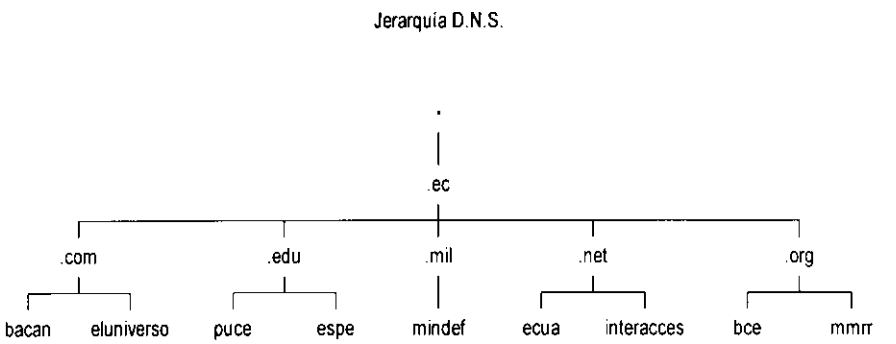


Figura # 7 Jerarquías

Ejemplo:

www.pucc.edu.ec

www.elcomercio.com.ec

www.pucesa.edu.ec

En la instalación del protocolo TCP/IP en un cliente se debe indicar el nombre o dirección del servidor DNS a donde acudir. Cuando un cliente quiere conocer la dirección IP correspondiente a una dirección simbólica, pregunta esto a su servidor DNS, si este conoce la respuesta se la da. Si no la conoce hay dos opciones:

Devuelve la dirección de otro servidor DNS, que conoce la respuesta o que conoce la dirección de otro servidor que conoce la respuesta o que conoce la dirección de otro servidor que conoce la respuesta.....

Efectúa la búsqueda él mismo entre sus homólogos y devuelve la respuesta al cliente.

El último caso es más frecuente, además, es el más eficaz; porque, de paso, el servidor DNS conserva la información para restituirla otra vez, debido a que cuando una máquina cliente pida la dirección de www.pucc.edu.cc hay muchas posibilidades de que vuelva a pedirla varias decenas de veces para descargar imágenes y páginas en los siguientes minutos.

Ciertos servidores DNS son delegados para administrar una parte del ámbito: se llama a esta parte "una zona". Esto significa que poseen los datos de esta parte de la base. Son la autoridad en esa zona y son responsables de la validez, coherencia, y actualización de

los datos de la base. Esto no impide tener una copia de otras partes de la base para aumentar su eficacia. Estos servidores se llaman primarios que carga sus datos a partir de un archivo escrito por un administrador. Un servidor es secundario cuando carga sus datos únicamente a partir de un servidor primario.

a) Ruta de Petición de Nombres con DNS

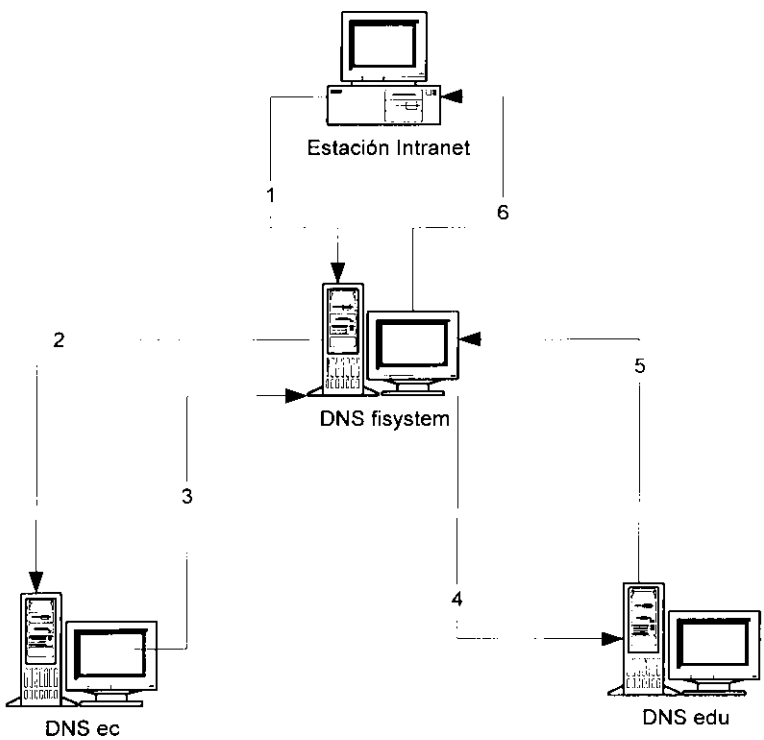


Figura # 8 Sistema D.N.S.

La estación Intranet pregunta la dirección de www.puce.edu.ec

El servidor local DNS fisystem pregunta donde puede encontrar las direcciones edu. a un servidor DNS ec.

Si el servidor no lo sabe pregunta a un servidor DNS edu.

El servidor DNS edu retorna la dirección 192.188.55.11

El servidor DNS fisystem retorna la dirección 192.188.55.11 a la estación Intranet.

1.3.2.3 CONFIGURANDO EL DNS

El primer paso es configurar el DNS de la Universidad para traducir las entradas en la tabla de "***hosts***" en equivalentes datos de DNS; esta configuración sirve para Servidores con Sistemas Operativos Unix o Linux, en el Sistema Operativo Windows NT hay un servicio especial DNS. Los archivos básicos de la configuración son:

named.boot	Archivo maestro de arranque "startup" del demonio DNS
named.data	Archivo que realiza la traducción de nombres a números
named.rev	Archivo que realiza la traducción de números a nombres
named.ca	Archivo de Root Name Server
named.lo	Resolución Local LOOPBACK

NAMED.BOOT

Este es el principal archivo de la configuración de DNS, sin este el mismo no correrá. Usualmente este archivo contiene el directorios donde los archivos de datos, reversos, cache, loop son localizados, además de un registro o línea por cada archivo a leer:

```
; @(#)65   1.4  com/etc/named.boot, , tcpip320, 9141320 8/14/90 15:50:31
;
; /etc/named.boot
```

```
;
;               boot file for name server
;
; type          domain                  source file or host
;
;
directory /etc/nameserver
domain          pucesa.edu.ec
cache           .               named.ca
primary                 pucesa.edu.ec           pucesa.ec.d
primary                 95.100.157.in-addr.arpa named.revzone
primary                 0.0.127.in-addr.arpa    named.local
;
```

NAMED.DATA

Las entradas en este archivo se denominan "DNS resource records", las entradas en este archivo puede ser en mayúsculas o minúsculas, los tipos de recursos son:

SOA	Indica la autoridad para el dominio
NS	Lista de Name Server para el dominio

Otros:

A	Tipo de registro mapea nombre a direccion
PTR	Mapea dirección a nombre
CNAME	Alias para un hosts denominado CANONICAL NAME
TXT	Información Textual [Opcional]
HINFO	Información del Host

MX Mail Exchange (Relay de Mail para host determinado)

El campo de SOA es contenido en cada uno de los archivos mencionados en el archivo "*named.boot*", en el mismo va:

El número de Serie "la Versión" de SOA,

El campo "refresh" cual indica el tiempo que deberá esperar para alimentar al secundario name server la información,

El campo "Retry" cada que tiempo el secundario intentara alcanzar el dominio en caso de que este abajo en el intervalo de "refresh"

El campo "expire" El tiempo en que el secundario esperara para poder seguir alimentando al dominio.

El campo "TTL" time to live, valor aplicado a cada unos de los recursos en los archivos. El tiempo que los otros server mantengan los datos de este dominio en cache.

```
;
; Archivo -> pucesa.hosts  util para mapear nombres a direcciones IP
;
;$ORIGIN pucesa.edu.ec.
@       IN  SOA pucesa.edu.ec.  root.pucesa.edu.ec. (
            97021401        ; Serial
            28800           ; Refresh every 8 hours
            7200            ; Retry every 2 hours
            604800          ; Expire after a week
            86400 )                 ; TTL

        IN  NS      pucesa
        IN  NS      pucesa.edu.ec.
        IN  MX 10   pucesa

pucesa    IN   A       204.225.202.73
www       IN   CNAME   pucesa
```

```
mail        IN   A      204.225.202.66
mail1       IN   A      204.225.202.18
mail2       IN   A      204.225.202.34
mail3       IN   A      204.225.202.50
mail5       IN   A      204.225.202.82
mail6       IN   A      204.225.202.98
mail7       IN   A      204.225.202.114
mail8       IN   A      204.225.202.130
mail9       IN   A      204.225.202.146
notes       IN   A      204.225.202.211
mail12           IN     A          204.225.202.215
mail11           IN     A          204.225.202.162
pucesafirewall        IN     A          204.225.202.242
```

NAMED.REV

Este archivo contiene por cada registro "A" en el named.data file, un puntero para direccionar números a nombres, igualmente contiene el recurso SOA.

```
; TRANSLATE arp IP A NOMBRES
@               IN  SOA  pucesa.edu.ec. root.pucesa.edu.ec.
(
                        97021401   ; versión
                        3600       ; refresh
                        3600       ; retry
                        864000     ; expire
                        172800 )   ; minimum

      IN   NS     pucesa.edu.ec.

73    IN   PTR    pucesa.edu.ec.
66    IN   PTR    mail.pucesa.edu.ec.
18    IN   PTR    mail1.pucesa.edu.ec.
34    IN   PTR    mail2.pucesa.edu.ec.
50    IN   PTR    mail3.pucesa.edu.ec.
82    IN   PTR    mail5.pucesa.edu.ec.
```

```
98    IN   PTR    mail6.pucesa.edu.ec.
114   IN   PTR    mail7.pucesa.edu.ec.
130   IN   PTR    mail8.pucesa.edu.ec.
146   IN   PTR    mail9.pucesa.edu.ec.
162   IN   PTR    mail11.pucesa.edu.ec.
211   IN   PTR    notes.pucesa.edu.ec.
215         IN   PTR    mail12.pucesa.edu.ec.
242   IN   PTR    pucesafirewall.pucesa.edu.ec.
```

NAMED.LO

El Name server necesita de este archivo para cubrir el "*loopback network*" propio de TCP/IP. Es una dirección especial que el server usa para direccionar el tráfico así mismo. La red de la Universidad es 204.225.202 y el Host: 204.225.202.2.

```
; Archivo -> named.local  define la direccion de  localhost
;
@       IN   SOA pucesa.edu.ec.  root.pucesa.edu.ec.
(
          97021401        ; Serial
          28800          ; Refresh every 8 hours
          7200           ; Retry every 2 hours
          604800          ; Expire after a week
          86400 )
       IN  NS       pucesa.edu.ec.

1      IN  PTR      localhost.
```

NAMED.CA

El archivo de cache sirve para indicar al Name server donde encontrar información del "*ROOT*" Domain. Esta información es recopilada del host nic.ddn.mil.

```
; named.ca - host que resuelven direcciones
;
; Initial cache data for root domain servers.
;
.                       99999999        IN      NS      NS.NIC.DDN.MIL.
                        99999999        IN      NS      KAVA.NISC.SRI.COM.
                        99999999        IN      NS      AOS.BRL.MIL.
                        99999999        IN      NS      C.NYSER.NET.
                        99999999        IN      NS      TERP.UMD.EDU.
                        99999999        IN      NS      NS.NASA.GOV.
                        99999999        IN      NS      NIC.NORDU.NET.
;
; Prep the cache (hotwire the addresses).  Order does not matter.
;
NS.NIC.DDN.MIL.         99999999        IN      A       192.112.36.4
KAVA.NISC.SRI.COM.      99999999        IN      A       192.33.33.24
AOS.BRL.MIL.            99999999        IN      A       192.5.25.82
C.NYSER.NET.            99999999        IN      A       192.33.4.12
TERP.UMD.EDU.           99999999        IN      A       128.8.10.90
NS.NASA.GOV.            99999999        IN      A       128.102.16.10
NS.NASA.GOV.            99999999        IN      A       192.52.195.10
NIC.NORDU.NET.          99999999        IN      A       192.36.148.17
```

1.3.2.4 ARRANCADO EL NAME SERVER

Bajo el directorio /etc existe un archivo ejecutable denominado "named" el cual debe ser insertado en los registros de arranque del sistema o manualmente:

```
/etc/named
```

Este comando asume que archivo de inicio es *"/etc/named.boot"*, ver información *named(1)* del SO.

Se puede configurar el archivo de log, para verificar funcionamiento del DNS, es responsabilidad del administrador del sistema el mantener este tipo de información, ya que los archivos crecen de manera indeterminada.

En el archivo "*named.pid*" se genera el número de proceso respectivo al demonio de DNS, si queremos dar una señal de restaurar el demonio:

```
kill -HUP  pid
```

Existe un utilitario para verificar el correcto funcionamiento del DNS, en muchos sistemas operativos el comando es "**nslookup**" en el mismo podremos darnos cuenta de como esta actuando el sistema DNS.

1.3.3 POLÍTICAS DE TELNET

1.3.3.1 INTRODUCCIÓN

Una de las funciones más impresionantes e útiles de Internet es conectarse a sistemas remotos como un terminal. De este modo un usuario normal puede ser parte de un sistema de supercomputadoras a través de una computadora personal. El programa para establecer esta conexión se llama *TELNET*

1.3.3.2 DEFINICIÓN

Telnet es un protocolo de los mas usados para conectarse a un Sistema UNIX mediante una línea de comandos. Telnet es usado cuando el usuario no dispone de una interfase gráfica en su terminal. Cuando se establece una comunicación con un sistema remoto, nuestro terminal envía y recibe respuestas a través de un puerto asignado por el servidor remoto, proveyendo una comunicación inmediata a través de un servidor anfitrión.

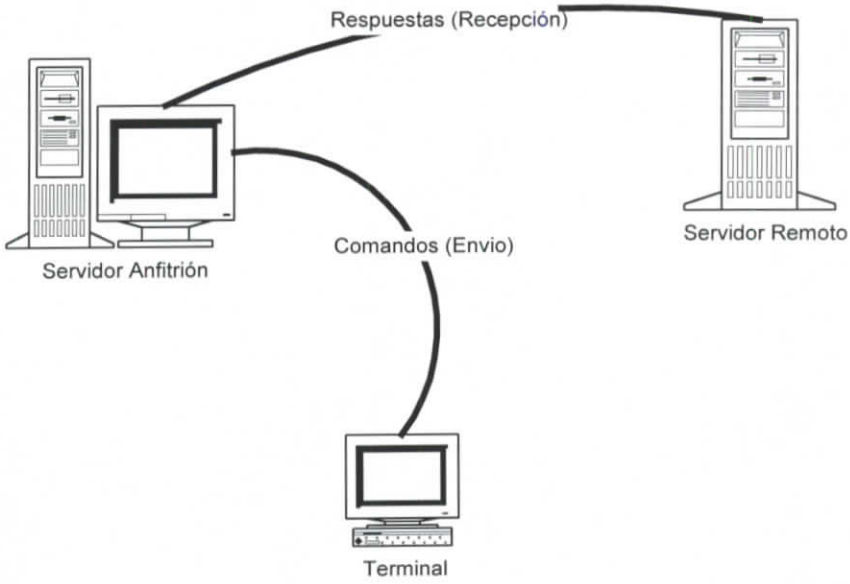


Figura # 9 Sistema Telnet

Telnet tiene un gran número de ventajas sobre FTP incluyendo la habilidad de comunicarse (chat) con otros usuarios, arreglar problemas con archivos de programas CGI, mostrar el contenido de directorios no vacíos y editar archivos sin bajarlos del servidor.

1.3.3.3 ACCESO

El acceso a un sistema remoto se puede hacer siempre y cuando se tenga autorización para hacerlo, para esto se debe obtener un usuario y una clave que serán otorgadas por el Administrador de la Red de la Universidad.

El proceso de asignación de permisos completo es:

- Establecer contacto con el Administrador de la Red de la Universidad o con la persona encargada de la seguridad de la Red.
- Definir la necesidad de investigación o utilización del servicio de Telnet en el Servidor Anfitrión.
- Abrir una cuenta de estudiante, profesor, investigador o empleado para recibir una identificación de usuario y una contraseña.
- Utilizar la identificación de usuario y contraseña para conectarse al Servidor Anfitrión

Es necesario establecer cuentas anónimas de usuario para el servicio de Telnet en el Servidor Anfitrión; esto sirve para proporcionar servicios generales a usuarios de Internet.

1.3.3.4 CONEXIÓN

Para establecer una conexión con Telnet, en cualquier Sistema Operativo, la sintaxis del comando es:

telnet servidor-anfitrión {puerto} <enter>

En donde el nombre del *servidor-anfitrión* debe ser reemplazado por un nombre válido o por una dirección IP válida del servidor remoto; este a su vez validará la identificación de usuario y contraseña. El *puerto* es opcional y solo debe ser utilizado en casos especiales que así lo requieran, generalmente para proveer información específica en el Servidor Anfitrión.

1.3.3.5 COMANDOS

Los comandos más comunes en Telnet son:

Comando	Uso
?	Muestra un resumen de los comandos Telnet.
Close	Cierra una conexión con el Servidor Anfitrión, abierta previamente.
Open host puerto	Abre una conexión con el Servidor Anfitrión, utilizando un puerto (opcional).

Comando	Uso
Quit	Abandona Telnet.
Status	Presenta información acerca del estado de Telnet.
Ls	Lista todos los directorios.
Cd directorio	Cambia de directorio.
Traceroute	Este comando traza una ruta entre el terminal y el dominio remoto, indicando el número de brincos que se realizan hasta llegar al Servidor Remoto.
Whois	El comando despliega la información actual de un servidor remoto o Host de Internet, incluye información acerca de la Administración, software, Dirección IP, etc.
Talk identificador_usuario	Comando que sirve para establecer una comunicación con otro usuario que este

Comando	Uso
	utilizando una sesión Telnet

1.3.4 POLÍTICAS PARA LA CREACIÓN DE MUROS DE FUEGO

1.3.4.1 DEFINICIÓN

Un Muro de Fuego o Cortafuegos o Firewall es un medio que sirve para regular el acceso a la red de computadoras de una institución. El papel de un Cortafuegos en una red de computadoras es similar al de un guardia de seguridad que protege la entrada de un edificio de oficinas. El Cortafuegos es el responsable de proteger la entrada a una red de una organización.

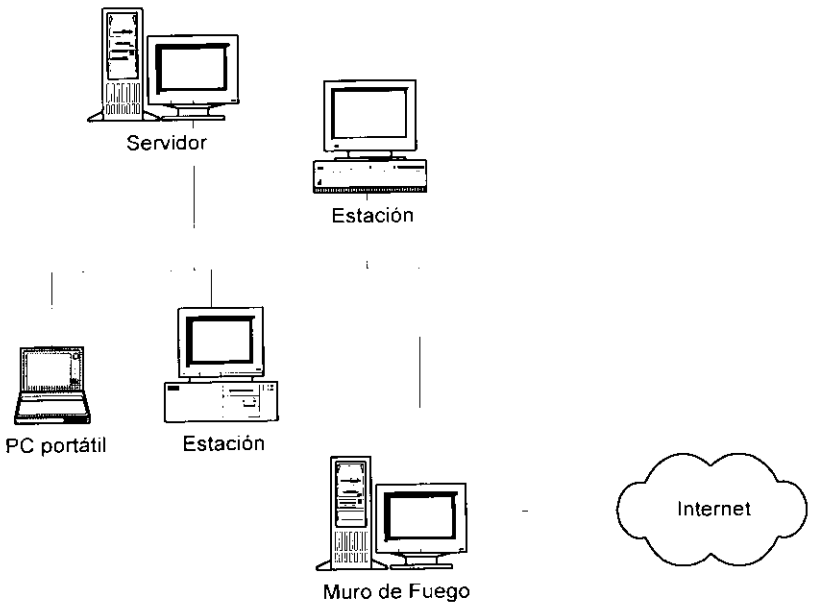


Figura # 10 Muro de Fuego

Para la red de computación de una organización o universidad, un Cortafuegos realiza las funciones de controlar el acceso y registrar los intentos de acceso. Para ello, consulta

la información identificativa asociada a la comunicación procedente del exterior. El Cortafuegos decide entonces permitir o no la comunicación de acuerdo con la política de seguridad configurada por el administrador de la Red, además anota todos los intentos en un registro electrónico.

1.3.4.2 UBICACIÓN DE UN CORTAFUEGOS

La ubicación de un Cortafuegos se la realiza en la entrada de la Intranet de una organización, en nuestro caso de la Universidad (ver Ilustración 1) Para que un Cortafuegos pueda desempeñar su función correctamente, todas las comunicaciones deben pasar a través de él. Es, en esencia, el punto en donde se produce la conexión con la red exterior. Actualmente el uso más habitual de un Cortafuegos, es separar la Intranet de una organización del ámbito público de Internet. Sin embargo, puede emplearse también dentro de una Intranet para proteger y aislar adicionalmente recursos especiales. Al estratificar la protección se protege adicionalmente estos recursos críticos internos de cualquier ataque procedente de Internet.

Los Cortafuegos son una parte importante de un sistema de seguridad equilibrado. No son, sin embargo, la solución de seguridad definitiva. Actúan como un filtro y, como tal, todavía pueden permitir la entrada de las amenazas en la red. Es conveniente evaluar cuidadosamente lose servicios de red que podrán atravesar el Cortafuegos. Además, debe proporcionarse seguridad adicional en otras áreas de la red como los controles de acceso basados en un Host.

La máquina en donde se establecerá un Cortafuegos debe ser un servidor, pero en una red pequeña como la de la Universidad Católica de Ambato se necesitaría solamente una pequeña máquina con un procesador Pentium; pero para efectos de velocidad y seguridad, la función de Cortafuegos deberá ser el único servicio soportado por el sistema.

1.3.4.3 SERVICIOS DE UN CORTAFUEGOS ESTÁNDAR

Las funciones o servicios básicos que ofrece un Cortafuegos son:

a) Control de Acceso

El objetivo principal de un Cortafuegos es proporcionar control de acceso a y desde la Intranet de una organización. Esto se lo consigue obteniendo tanta información como es posible de los paquetes o sesiones que pasan a través de él. Mediante esta información y una política de seguridad definida, el Cortafuegos decide autorizar o denegar el paso del paquete, si se niega, la sesión asociada al mismo no se completa satisfactoriamente. La cantidad y calidad de la información conseguida de una sesión determinará la probabilidad de éxito de un Cortafuegos.

b) Registro de Actividades

Un Cortafuegos eficaz puede registrar importantes elementos de información referentes a todos los intentos de sesión exitosos e infructuosos que se efectúan a través del

Cortafuegos. Esta información es valiosa cuando se trata de determinar cual fueron las actividades de un intruso y que áreas se afectaron. Esta información se conoce como un Registro de Auditoria.

1.3.4.4 INSTALACIÓN DE LOS CORTAFUEGOS

La mayoría del trabajo administrativo consiste en instalar el Cortafuegos y configurar las reglas de acceso. Por ejemplo los routeadores de la red deben configurarse para dirigir el tráfico hacia el Cortafuegos, y éste debe definirse con dos direcciones IP (una para cada interfaz) y con rutas hacia routeadores ubicados en las proximidades. La mayoría de Cortafuegos obligan a utilizar rutas estáticas en vez de dinámicas, permitiendo así una configuración mas adecuada de las reglas de acceso y actividades.

Se recomienda como mínimo unas dos semanas para establecer el Cortafuegos, aún así, se haya comprado un paquete preestablecido para el efecto. Si se configura el Cortafuegos desde cero, este tiempo deberá incrementarse más.

Una vez realizada la configuración inicial, las demás tareas del Administrador son supervisar y hacer copias de seguridad de los registros de actividades, agregar o modificar reglas de acceso según convenga y añadir información de autenticación de usuarios. También debe establecerse que el Cortafuegos pueda ser administrador remotamente permitiendo la configuración de varios Cortafuegos desde una ubicación central. El Administrador del Cortafuegos, en colaboración con el personal de la red y el

personal de gestión, debe determinar la lista de servicios que pueden entrar y salir de la red. Los riesgos asociados deben evaluarse y documentarse. Los riesgos inaceptables deben eliminarse, o en su defecto, incorporar una seguridad adicional a la red a fin de poder gestionar la amenaza asociada a los mismos.

1.3.4.5 AMENAZAS REALES PARA LA INTRANET

Las amenazas reales de Internet hacia nuestra Intranet pueden ser varias, complicadas y que pueden causar daños graves a nuestras redes. Entre las amenazas tenemos:

a) Sistemas abiertos

El concepto de sistemas abiertos esta enraizado en la noción de un enfoque estándar de la informática y de las redes que proporciona una mayor interoperabilidad, flexibilidad y potabilidad de software y de componentes del sistema.

A partir de los años 60 y 70 comenzaron a crearse islas locales de redes digitales dentro de las organizaciones empresariales, militares, gubernamentales y académicas. Uno de los aspectos más importantes de estas redes fue que no estaban conectadas con otras redes, lo que dio como resultado redes a cuyos activos no podían acceder fácilmente los atacantes del exterior. Sin embargo las recientes conexiones a Internet en los 80 y 90 han eliminado prácticamente las redes aisladas. Así, la seguridad de las redes modernas ha disminuido debido a su mayor capacidad de conexión, pues facilita el acceso a las mismas de los atacantes potenciales.

No se pretende decir que los sistemas abiertos son peligrosos y no recomendables, lo que se quiere decir es que los efectos de seguridad deben ser incrementados y tomados en cuenta. Los sistemas abiertos permiten mejorar la conexión y facilitar un mayor acceso a la red pero también incrementan los riesgos de sufrir ataques malintencionados.

b) El crecimiento de Internet

Otra tendencia que provoca el ataque hacia las redes es el inusual y espectacular crecimiento de Internet. Este crecimiento ha sido impulsado por los avances fundamentales en la tecnología de redes que se ha producido a una velocidad vertiginosa.

Así, cuando nuestras redes son pequeñas, hay individuos que navegan en el Internet buscando fallas o "agujeros" en los sistemas de seguridad, algunos lo hacen por pura diversión o para comprobar que son capaces de realizar actividades maliciosas sin ser descubiertos. Otros en cambio lo hacen por disminuir las capacidades de la competencia y establecer un monopolio único, pero entre estos casos están los peores que son los de terrorismo informático que causan daños casi irreparables a los sistemas involucrados.

1.3.4.6 TIPOS DE ATAQUES PROCEDENTES DE INTERNET

Las amenazas de Internet son muchas y muy variadas, explicarlas todas sería una descripción larga y tediosa, aquí solo explicaremos algunas simples que son las más representativas.

a) Ataques simples por negación del servicio

Este es un tipo de ataque que simplemente empieza a solicitar un servicio en especial, haciendo que se sature y deje de funcionar debido a un trabajo excesivo, saturando a las víctimas con tal volumen de datos que los sistemas se inhiben. Tal vez los buzones de correo electrónico son el objetivo más habitual de los ataques de negación del servicio, en especial porque prácticamente todo el mundo divulga su dirección Internet a cualquier persona interesada. Este tipo de ataque puede ser manual o automático, enviando mensajes de correo a un buzón hasta que este es incapaz de gestionar el volumen.

El desastre del "Gusano de Internet" en 1988 causó un grave ataque de negación de servicio, aunque la aplicación se replicaba a sí misma no causo daños a la información, pero si pérdidas en costos de procesamiento y la "caída" de muchas redes y computadoras.

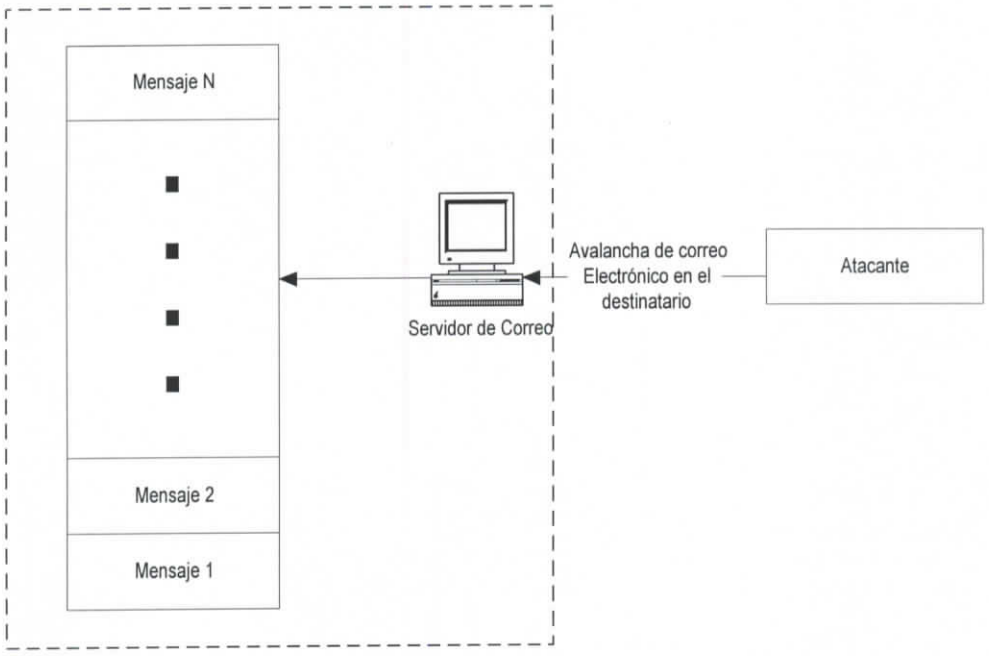


Figura # 11 Ataques por Correo Electrónico

Rechazar este ataque no es sencillo, pero existen varios métodos que si pueden reducir el intento, tales como reducir el número de conexiones a sistemas anfitriones internos.

b) Aspiración de paquetes de red

Si la red de área local esta conectada a varios equipos, es vulnerable a este tipo de ataques en los que alguien puede "escuchar" los datos que fluyen por la red. Los programas llamados aspiradores de paquetes, capturan paquetes que circulan por la red de área local y los presentan en forma legible. El remitente y el destinatario de esta información nunca sabrán que ha sido intervenida. Los administradores y encargados de la red han usado los aspiradores de paquetes como una herramienta útil de diagnóstico.

Si la red estará conectada a Internet, las probabilidades de que estos paquetes sean intervenidos por extraños se incrementa notablemente y puede dejar a los sistemas críticos vulnerables a cualquier ataque.

c) Ataque por suplantación de IP

Otro tipo habitual consiste en el intento de suplantar el IP de un usuario. Esta suplantación se produce cuando un atacante utiliza la dirección Internet exclusiva de un usuario desprevenido, especialmente en entornos de datos.

Una suplantación de IP se convierte en ataque grave cuando el atacante externo pretende tener una dirección IP interna de la red a la que desea atacar. Este tipo de ataque es realmente peligroso debido a que logra pasar por varios tipos de Cortafuegos sin ser detectados. Al combinar este ataque con otras suplantaciones más complejas (como el secuestro de sesiones) puede producir intrusiones exitosas.

d) Información útil sobre los ataques procedentes de Internet

En Internet existe información que detecta y avisa sobre los ataques a los cuales puede una Intranet o el Internet ser expuestos. Los foros comp.risk y alt.2600 de Usenet son buenas fuentes de información junto con el foro en ftp://crvax.sri.com/risks en donde están enumerados varios ataques.

Existe además una biblioteca manejadas por el Equipo Responsable de Emergencias de Computadoras (CERT en Inglés) que da muy buenos reportes de ataques y como prevenirlos, incluso presta asesoramiento a quien se lo solicite. Se lo puede ubicar en ftp://infor.cert.org/pub/cert_faq

1.3.4.7 TIPOS DE CORTAFUEGOS

Los Cortafuegos pueden diferir en su arquitectura y características, pero hay dos tipos básicos:

- Cortafuegos de Filtrado de Paquetes
- Cortafuegos de Gateway
- Cortafuegos Híbridos

a) Cortafuegos de Filtrado de Paquetes

Este Cortafuegos es un router de filtrado, su función normal es aceptar un paquete, identificar su próximo salto de ruta, por medio de un examen de la información que contiene el paquete IP y autorizar su paso o desecharlo dependiendo de si se encuentra una ruta válida o no. Este proceso a bajo nivel proporciona una mayor velocidad, aunque es menos seguro que un Cortafuegos de Gateway.

Este proceso se lleva al nivel de Kernel en el Sistema Operativo, a diferencia de las aplicaciones el Kernel no esta sujeto a programa alguno, permitiendo que la información se procese mucho más rápido, algunos routers realizan este procedimiento a nivel de la tarjeta de interfaz de la red con el fin de proporcionar un rendimiento mayor.

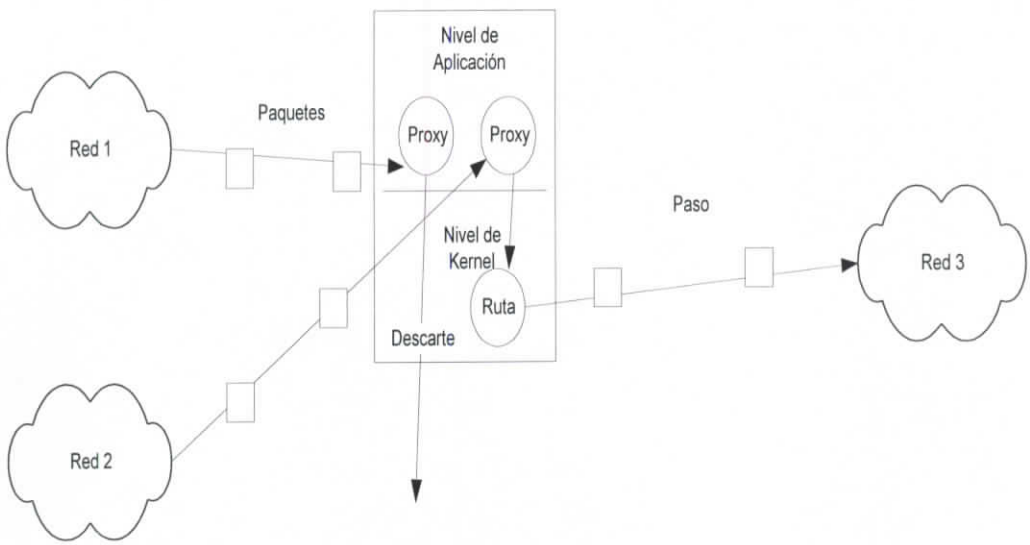


Figura # 12 Cortafuegos por Filtrado

b) Gateways a nivel de Aplicación

Estos Cortafuegos se aplican a nivel de software en una plataforma de Host, estos impiden el paso directo de los paquetes de una red a otra. Obligan a que la conexión original se haga a una aplicación Proxy que se encuentra dentro del Cortafuegos. Este decide si se establece la conexión o no. Este tipo de Cortafuegos impiden algunos ataques como el Sobreflujo de Buffer y proporcionan una mayor seguridad que los anteriores aunque son mas lentos y no soportan todos los servicios.

Una de las limitaciones de este Cortafuegos es que precisa de una aplicación individual para cada servicio de la red. Así pues si se establece un nuevo servicio, se tendrá que incrementar una aplicación Proxy o de lo contrario se denegará el acceso a los usuarios. Además existe un límite en cuanto al número de conexiones simultáneas que puede proporcionar el Servidor.

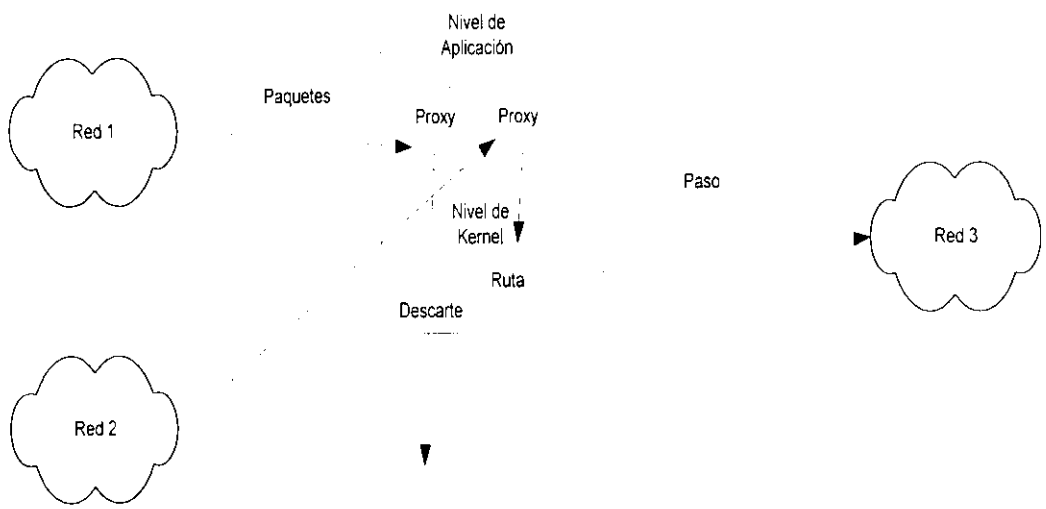


Figura # 13 Cortafuegos por Aplicación

c) Cortafuegos Híbrido

Una variante natural de los dos Cortafuegos descritos anteriormente incluye un filtro de paquetes y un Gateway a nivel de aplicación. Los paquetes recibidos son sometidos en primer lugar a las decisiones de filtrado del filtro de paquetes. A partir de ahí, los paquetes pueden desecharse, hacerse pasar a través del Kernel hacia su destino previsto o enviarse a un Proxy a fin de ser procesados posteriormente.

Un Cortafuegos híbrido es la mejor solución para una Intranet que necesita la seguridad que ofrece un Gateway a nivel de aplicación para ciertos servicios y la velocidad y flexibilidad de un filtro de paquetes para otro tipo de servicios. Este Cortafuegos es mas caro que un router normal pero con una combinación de routers y Gateways se puede obtener un resultado similar.

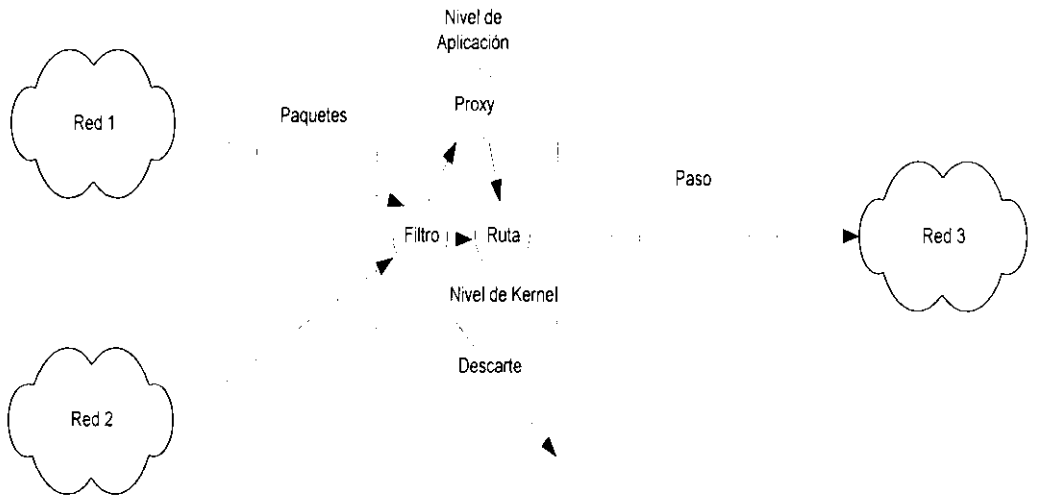


Figura # 14 Cortafuegos Híbrido

1.3.4.8 ADMINISTRACIÓN DE CORTAFUEGOS

Una vez escogido el Cortafuegos adecuado, se debe escoger el método para administrar los mismos, esta protección puede llegar a ser nula si no es administrado correctamente. Una interfaz simple y con un número mínimo de opciones de configuración reduce los errores de administración, pero produce una menor flexibilidad de configuración. Casi siempre todas las administraciones pueden ser locales o remotas.

Existen tres clases de interfaz de Administración de Cortafuegos:

- Administración basada en archivos de texto
- Administración basada en menús de texto
- Administración basada en GUI

La Administración basada en archivos de texto es de uso extendido en sistemas UNIX tradicionales dado que ofrece una interfaz de control a bajo nivel, utiliza un archivo específico donde se puede introducir parámetros de configuración específicos. La desventaja radica en que se pueden cometer errores al editar un fichero se pueden cometer errores de escritura u otros errores técnicos.

La administración basada en menús de texto presenta menús que reducen los errores pero que proporciona menor control para el administrador, impidiendo que este pueda ver el efecto de algunos cambios.

La interfaz gráfica de usuario (GUI) incorpora botones, menús desplegables, ventanas, mensajes de ayuda y combina todas las ventajas de las dos administraciones anteriores siendo más fácil de utilizar y no es susceptible de muchos errores.

a) Configuración de mecanismos para el control del acceso

Una vez que se ha instalado y configurado el Cortafuegos en nuestra Intranet para comunicarnos con el Internet, debemos comprobar el conjunto inicial de reglas de filtrado o los permisos de acceso a las aplicaciones Proxy. Estas reglas están definidas en las políticas de seguridad. Puede incluir controles para:

- Autorizar todas las sesiones internas TCP el acceso a Internet.
- Autorizar sesiones de correo procedentes de Internet el acceso a un servidor de correo Interno.
- Autorizar sesiones de noticias de correo procedentes de Internet el acceso a un servidor de noticias interno.
- Permitir a algunos mensajes ICMP (protocolo de errores y mensajes de control a nivel de IP) procedentes de Intranet a Internet.
- Autorizar a las sesiones de ingreso mediante encriptación.
- Denegar el acceso al resto de tráfico. (Acción predeterminada)

b) Tareas Diarias

Una vez instalado y configurado el Cortafuegos, el trabajo del administrador será el de vigilar el tráfico de la red en función de las políticas de seguridad. Podemos describir unas pocas tareas que el Administrador de la Red debe ejecutar:

Mantener las cuentas de los usuarios de la red el Administrador debe agregar, eliminar, y modificar las cuentas de los usuarios de la red, mediante un servidor de autenticación centralizado.

Actualizar permisos de acceso a los servidores según se producen cambios en las políticas de seguridad o se añaden nuevas redes a la Intranet

Reaccionar ante las alarmas se debe reaccionar ante las alarmas del sistema, al menos la mayoría serán inocentes o falsas. Se debe prestar atención a las que se repiten con frecuencia y adoptar los correctivos necesarios.

Revisar registros de actividad Al revisar estos registros se pueden detectar accesos no permitidos que las alarmas no detectaron y que ayudan a crear nuevas alarmas. Esta revisión se simplifica al pasar el tiempo.

Hacer copias de seguridad del Cortafuegos es importante hacerlas a intervalos regulares.

Tareas de mantenimiento de la plataforma el sistema operativo subyacente y el hardware necesitan un mantenimiento periódico.

c) Respuestas ante ataques

Si el Cortafuegos es atacado y su Intranet esta en peligro de ataque, se deben realizar los siguientes pasos:

- Mantener la calma, en realidad un gran porcentaje de los ataques son intentos inocuos de atravesar las defensas.
- Decidir interrumpir el ataque o no. Es importante saber la fuente del ataque, en este caso no se debe interrumpir el ataque hasta conocer bien al atacante. En el caso de que recursos críticos estén en riesgo, el ataque debe interrumpirse inmediatamente.
- Interrumpir o destruir las conexiones establecidas por el atacante, si esto no es factible, cerrar la conexión externa a Internet.
- Determinar los cambios a realizar en la política de seguridad para evitar futuros ataques de este equipo.
- Comunicar al equipo de trabajo sobre los detalles de este ataque.

d) Recuperación de desastres

Como un Cortafuegos es la conexión vital de los usuarios de nuestra Intranet con el mundo externo, se debe incluir un plan de recuperación de desastres que incluyan pasos a seguir cuando el Cortafuegos sea inoperante y respuestas alternativas en función de la interrupción del funcionamiento.

En el caso de que el Cortafuegos se rehuse a funcionar o sea víctima de un ataque efectivo que lo deje inutilizado, se deben intentar algunos de los siguientes pasos:

- Asegurarse que existan comunicaciones que entren al Cortafuegos, pero que ninguna sale del mismo.
- Intentar corregir el error desde la consola del Cortafuegos.
- Comprobar que los cables de alimentación y de datos del Cortafuegos están conectados correctamente
- Reiniciar el Cortafuegos, el software y la red desde la consola.
- Comprobar el estado del Cortafuegos y los mensajes de error que se reporten; comuníquese con el soporte del proveedor del Cortafuegos.
- Comunicar el incidente a los altos cargos de dirección.
- Desconectar el Cortafuegos principal y poner en marcha el substituto.
- Restaurar el Cortafuegos desde la última copia de seguridad adecuada.

1.3.4.9 SERVICIOS SOPORTADOS

Los servicios a los que nos referimos aquí son los protocolos, a nivel de aplicación, que el Cortafuegos reconoce y autoriza. Los servicios se identifican mediante el número de puerto utilizado, TCP o UDP, que son números menores que 1024 y que por regla general son fijos y están registrados como estándares de Internet.

Es importante que si el Cortafuegos no puede reconocer un puerto, servicio o protocolo, niegue terminantemente la conexión. Es tarea del Administrador de Red actualizar esta lista de puertos en función de los nuevos servicios que se pueden ofrecer a través de Internet o en la Intranet.

DNS *Protocolo TCP o UDP* *Puerto 53*

Por lo general este servicio no es usado por los usuarios sino por los servidores DNS local. El Cortafuegos debe permitir que el servidor DNS de la red propia el acceso a los servidores DNS externos e incluso su actualización con direcciones nuevas, negando a los servidores DNS externos poder actualizar los servidores DNS internos.

Finger *Protocolo TCP* *Puerto 79*

Este servicio fue desarrollado para permitir a los usuarios de una red poder localizar a otros usuarios, es posible averiguar nombres de entrada (login), nombres reales y datos de acceso. El Cortafuegos debe prohibir terminantemente el acceso desde el exterior de cualquier solicitud de este servicio. Una alternativa válida es tener un Proxy de Finger que muestre un mensaje estándar que diga "Esta red no soporta el servicio Finger" a las peticiones entrantes y autorice las peticiones hacia el exterior.

FTP *Protocolo TCP* *Puerto 21*

Este protocolo es estándar para transferir archivos entre sistemas. Dado que FTP se establece mediante contraseñas sencillas y puertos perfectamente identificados, el Cortafuegos debería permitir este tipo de conexiones, siempre y cuando se establezca una conexión con el cliente y no con el servidor.

Gopher *Protocolo TCP* *Puerto 70 y otros*

Gopher proporciona un servicio sencillo de menús textuales que proporcionan información en Internet. En nuestra Intranet no lo necesitaremos, por lo que se debe prohibir el acceso desde el exterior y permitir la salida al exterior de solicitudes de este tipo.

ICMP *Protocolo ICMP*

ICMP (Internet Control Message Protocol) es un protocolo de mensajes de control de Internet, soportado por encima de IP a nivel de TCP o UDP que lo utilizan para enviar mensajes de error o prueba a sistemas distintos. ICMP contiene campos de tipo y código estándar. Un Cortafuegos puede filtrar selectivamente estos mensajes según el tipo de código utilizado.

IRC *Protocolo TCP* *Puerto 6667*

La aplicación IRC (Internet Relay Chat) ofrece la posibilidad de participar en conferencias con múltiples usuarios en un entorno de texto. La principal amenaza no es

inherente al protocolo sino que es una amenaza social porque el atacante puede pedir u obligar a que el Administrador de la Red o un usuario proporcionen información de autenticación o reduzca los controles de seguridad; por lo que un Proxy IRC no es de mucha utilidad.

Mail *Protocolo TCP* *Puerto 25*

Este es el servicio más utilizado en Internet SMTP (Simple Mail Transfer Protocol), permite que el usuario mande un mensaje sin tener que establecer una conexión directa con el servidor destino. SI el Cortafuegos posee un servicio actualizado del SMTP o sendmail entonces ya se habrán cubierto los niveles de seguridad requeridos.

MBONE *Protocolo IP*

Multicast Bone (Mbone) es un servicio que se emplea para dirigir paquetes multitransmitidos a través de routers que no soportan multitransmisión. Se emplean para vídeo conferencias o radio Internet. El Cortafuegos debe identificar la dirección IP de este paquete y permitir solo aquellos que vienen de direcciones perfectamente identificadas.

Network News *Protocolo TCP* *Puerto 119*

Permite que los usuarios lean o participen en debates a través de la lectura de información en una serie de mensajes con un tema común. Es muy común usar un

servidor NNTP (Network News Transfer Protocol) interno para los estudiantes y profesores y permitir el acceso hacia servidores NNTP externos.

NFS *Protocolo UDP* *Puerto 2049*

Permite a los usuarios compartir sistemas de ficheros con otros usuarios. Este servicio proporciona muy poca seguridad y se recomienda no usarlo y prohibirlo terminantemente con conexiones al exterior y restringir su uso interno.

NTP *Protocolo UDP* *Puerto 123*

Network Time Protocol permite que los servidores estén sincronizados con sus relojes, no se conoce de alguna amenaza conocida por lo que se permite su libre acceso.

Port Mapper *Protocolo TCP o UDP* *Puerto 111*

Este servicio asigna los números de puerto para las aplicaciones en el servidor. No es muy utilizado y debe trabajar estrechamente con el Cortafuegos.

Rlogin *Protocolo TCP* *Puerto 513*

Este comando, mas que servicio, permite que un usuario pueda acceder remotamente al sistema local, pero solo se debe permitir el acceso si es que esta conexión se establece a través de un enlace seguro y con una dirección IP perfectamente identificada.

Telnet *Protocolo TCP* *Puerto 23*

Es el protocolo y aplicación estándar para la entrada (login) en sistemas remotos. Proporciona una conexión entre dos sistemas basados en caracteres. Como es una aplicación muy conocida, el Cortafuegos permite autenticar el usuario sin ningún problema.

SNMP *Protocolo TCP y UDP* *Puertos 161 y 162*

Simple Network Management Protocol es un protocolo que emplea una estación de gestión para supervisar y configurar dispositivos de red. Debe quedar terminantemente prohibido su uso en los accesos hacia la red interna.

WWW *Protocolo TCP* *Puerto 80 y otros*

Este es el servicio mas usado en Internet, usa el protocolo HTTP (Hyper Text Transfer Protocol) que permite que los usuarios vean texto, gráficos, animaciones y sonido en su estación. Es un servicio complejo que merece que el Cortafuegos realice monitoreos constantes.

2 TRABAJO PRÁCTICO

2.1 INTRODUCCIÓN

El trabajo práctico consiste en una aplicación completa que incluye:

- Un programa orientado a objetos para el mantenimiento de los datos
- Una página de consulta de los datos a través de un navegador de Internet
- Manual técnico de la aplicación en páginas Web
- Manual del usuario en páginas Web
- Explicación de las técnicas de programación utilizadas en el desarrollo del trabajo práctico

2.2 APLICACIONES INTRANET

2.2.1 BIBLIOTECA

Aplicación para el control del inventario de libros de la Biblioteca

2.2.1.1 INSTALACIÓN

a) Crear un ODBC

Para poder trabajar con el módulo Biblioteca se debe crear un ODBC para la base de datos que contiene los datos de la Biblioteca.

Desde el Panel de Control, doble clic en el icono ODBC Data Sources 32 Bits

Este es un componente de Office 97 que puede o no haber sido instalado en su máquina, si usted no tiene este icono, por favor contáctese con el Administrador de Sistemas para pedir su instalación.

- Añadir un nuevo ODBC (System DNS)Driver: Microsoft Access
- Nombre del Origen de la Base de Datos: Biblioteca
- Presionar en el botón Seleccionar y buscar la ubicación del archivo Biblioteca.mdb
- Presionar en Aceptar en las dos ventanas de dialogo.

Después de seguir estos pasos puede trabajar con la Aplicación Biblioteca.

2.2.1.2 APLICACIÓN BIBLIOTECA

El Módulo Biblioteca permite llevar el control de los libros que existen en la Biblioteca de la Pontificia Universidad Católica - Sede Ambato

a) Como trabaja

El módulo Biblioteca está desarrollado en Visual Basic 5.0 con Access 97 y guarda la siguiente información:

Datos del AutorCódigo

Nombre

Datos de los libros

Número del inventario

Título del libro

Editorial

Ciudad

Número de Páginas

Notas

Para ver mas detalles referirse al anexo 5.2

3 CONCLUSIONES Y RECOMENDACIONES

3.1 CONCLUSIONES

- La diferencia fundamental entre el Internet y la Intranet es que el Internet, de alguna manera, es una anarquía de información y la Intranet puede ser normada y controlada para beneficio de sus usuarios. Por lo tanto la creación de la Intranet precisa de normas y reglas específicas para que no sea desviada de su objetivo primordial cual es, informar con claridad, eficiencia y prontitud.

- Para crear una Intranet se necesitan herramientas que pueden ser encontradas fácilmente en el Internet. Estas herramientas ayudan a un usuario a crear sus propias páginas Web que podrían ser parte de la Intranet.

- La eficiencia de la Intranet radica en la correcta utilización de la tecnología, software y hardware, obviamente esta ligada al correcto mantenimiento de sus elementos y a la constante actualización de los mismos.

- La Intranet le permite a la organización difundir información comprensible a los usuarios, así estos pueden estar seguros de cómo actuar, conocer las reglas y normas a seguir, sus derechos y deberes frente a la corporación.

- La Intranet permite que la comunicación entre profesores y estudiantes establezca vínculos tecnológicos más efectivos que los que se consiguen en una clase magistral, por lo tanto el aprendizaje puede alcanzar altos niveles de formación profesional.

- Implementar una Intranet institucional fortalece la estructura organizativa interna de la misma, clarificando aspectos vitales para su desarrollo.

- El uso de una Intranet en la institución permite la sistematización y eficiencia del trabajo evitando dispersión y desperdicio de recursos humanos y materiales.

- La administración de la Intranet, al regirse por políticas y normas específicas requiere que las mismas sean enmarcadas en una normativa jurídica institucional decidida por los niveles jerárquicos correspondientes y revisadas periódicamente de acuerdo a las necesidades institucionales.

- La realización de este trabajo de investigación se hizo basándose en la necesidad de agrupar toda la información requerida para el desarrollo de una Intranet en un solo documento que servirá de consulta para conocer uno de los métodos de administración centralizada de información más utilizados en el mundo.

- Las aplicaciones prácticas realizadas en este trabajo se basaron en la necesidad de crear ejemplos que demuestren la utilidad que puede ofrecer la Intranet utilizando herramientas de desarrollo visual orientado a objetos, que además servirán como guía para el desarrollo de futuras aplicaciones.

3.2 RECOMENDACIONES

- Es prioritario el uso de la Intranet como respuesta a los requerimientos elementales de una Universidad que debe enfrentar los retos tecnológicos de su desarrollo de acuerdo con el cambio del milenio.

- Al crear una Intranet se aconseja tener ética profesional, un comité podrá supervisar que la información a ser publicada en la Intranet cumpla con los fines que persigue la Universidad, lo cual permitirá que el uso del sistema sea el adecuado y correcto

- No es fundamental saber programar, pero si es necesario tener nociones básicas de elaboración de páginas Web y lenguaje H.T.M.L. para la creación de la Intranet.

- Hay que propagar el uso de la Intranet universitaria, porque se podría incrementar la difusión de información, permitiendo así que se establezca una relación mas estrecha de conocimientos y experiencias entre profesores y estudiantes.

- Apoyar el estudio de las nuevas técnicas aplicadas en Internet para que los futuros profesionales estén preparados para el desarrollo de Intranets aplicadas en empresas publicas, privadas, sectores educativos, etc

- Aprovechar las aplicaciones desarrolladas en este trabajo para que los estudiantes, profesores y personal docente saquen mayor provecho de la Intranet de la Universidad.

- Utilizar el trabajo práctico como herramienta de consulta para el desarrollo de nuevas aplicaciones que ayuden a mejorar el servicio que presta la parte administrativa y docente de la Universidad.

- El objetivo propuesto se ha cumplido a cabalidad porque se ha desarrollado un manual de consulta para la creación de Intranets. Este debe utilizarse como un instrumento y guía de consulta para la creación de Intranets.

4 BIBLIOGRAFIA

4.1 LIBROS

La Magia de Internet, Allen L. Wyatt, McGraw – Hill, México 1995

El Proyecto Intranet, Frédéric Alin - Denis Lafont -Jean-François Macary, Ediciones Gestion, París 1997

Seguridad para Intranet e Internet, Edward Amoroso -Ronald Sharp, Prentice – Hall, Madrid 1997

Java , Enrique Castillo - Angel Cobo - Cristina Solares - Patricia Gómez. Editorial Paraninfo, España 1997

La Revolución Informática,Sebastián Dormido - Mariano Mellado, Salvat Editores, Barcelona 1981

Exploring Java WorkShop 2.0, Sun Microsystems, SunSoft Inc., Palo Alto, CA 1997

Visual Basic 5.0 - Manual del Programador, Microsoft Corporation, E.E.U.U. 1997

4.2 REVISTAS Y PUBLICACIONES

Análisis y Diseño Orientado a Objetos, Método de Grady Booch

Pc – Magazine en Español, Volumen 10 Número 2, Edición Ecuador - Perú

Oracle Magazine, Volume XIII Number 4, July / August 1999

4.3 SITIOS WEB

4.3.1 UNIVERSIDADES

http://www.puce.edu.ec

http://www.pucesa.edu.ec

http://www.espe.edu.ec

4.3.2 MOTORES DE BÚSQUEDA

http://www.yahoo.com

http://www.espanol.yahoo.com

http://www.altavista.com

http://www.lycos.com

http://www.webcrawler.com

http://www.hotbot.com

http://www.msn.com

http://www.ask.com

4.3.3 ORGANIZACIONES

http://www.w3.org

4.3.4 COMERCIALES

http://www.oracle.com

http://www.microsoft.com

http://www.sun.com

http://www.cnet.com

http://www.discovery.com

5 ANEXOS

5.1 FORMULARIO DE INGRESO DE USUARIOS

Universidad Católica del Ecuador
Programas Académicos Ambato
Formulario de Ingreso de Usuarios

Formulario N° **Fecha :**

Datos Generales
Nombres :
Apellidos :

Datos del Usuario
Nombre del Usuario :
Tipo de Usuario : Estudiante

Accesos

Recurso	Total	Lectura	Negado
Intranet	☐	☐	☐
Disco Compartido	☐	☐	☐
Impresora	☐	☐	☐
Internet	☐	☐	☐
	☐	☐	☐
	☐	☐	☐

Director	Usuario

5.2 DISCO COMPACTO

Para el trabajo práctico, por favor referirse al Disco Compacto adjunto.