95
21

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

INSTALACION Y GESTION REMOTA DE SOFTWARE (PROYECTO INGRESO)

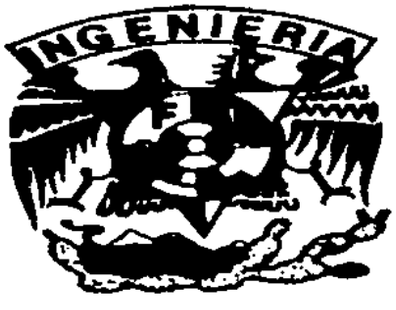
T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACION

P R E S E N T A :

ESPERANZA SOLIS ESCAMILLA



DIRECTOR DE TESIS: DR. URIEL TIRADO RIOS

COODIRECTORA: ING. MARIA JAQUELINA LOPEZ BARRIENTOS

MEXICO, D. F. 1997

TESIS CON
FALLA DE ORIGEN

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

DEDICATORIAS

Dedico este humilde trabajo para quien me dio la oportunidad de vivir, a quien generalmente solo imploro cuando necesito ayuda pero siempre esta a mi lado dispuesto a escucharme, como agradecimiento a su generosidad, a mi mejor amigo ... Dios.

Para Juan y Alicia, mis queridos padres, como un tributo a todos los sacrificios, cuidados, y desvelos que han tenido que pasar por verme realizar mis sueños, pero sobre todo por darme un hogar lleno de amor.

Para la única persona que ha soportado mi cansancio, mal humor y desesperación con mucha paciencia, compresión y amor. Para ti mi amado Ivan, que has estado junto a mi cuando más te he necesitado, gracias por tu apoyo, por tu tiempo, por tus consejos, por tu amor...sin ti nada habría sido posible.

AGRADECIMIENTOS

A la Universidad más importante de América Latina, la Universidad Nacional Autónoma de México por permitirme cumplir el sueño de formar parte de ella.

Al H. Instituto Mexicano del Petróleo por brindarme la oportunidad y los medios para la realización de este trabajo.

A mi director de tesis, el Dr. Uriel Tirado Ríos por el tiempo, paciencia y apoyo que siempre me brindó.

A mi coodirectora de tesis, la Ing. Ma. Jaquelina López Barrientos que tan amablemente acepto colaborar en este proyecto, gracias por su amistad, ayuda y comprensión.

A la Coordinación de Informática del IMP, en especial al Ing. José Hernández Monroy, a la Ing. Ma. del Pilar Vidriales García y al Ing. Andrés Sánchez Báez, por su valiosa ayuda y confianza.

A mis queridos amigos del Departamento de Soporte Informático del IMP: Paco, Carlos, Chucho, José Luis y Ma. Elena, por su amistad y apoyo incondicional.

A dos grandes amigos del Departamento de Diseño Asistido por Computadora del IMP: al M. en C. José Manuel Cervantes Martínez y al Ing. Moisés Payán Beltrán quienes siempre estuvieron dispuestos a ayudarme.

AGRADECIMIENTOS

A mis hermanos Pedro y Mimis por sus cuidados, apoyo y ejemplo.

A mis queridos sobrinos Gus y Ryan por alegrarme los días con sus sonrisas y cariño.

A mis adorados abuelitos por sus oraciones y por su inmenso amor.

A mi tía Ventura y mi primo Dany que me abrieron las puertas de su casa cuando más lo necesité, por su aprecio y por compartir conmigo el orgullo de ser universitaria.

Y A todas la personas que de una u otra forma me han brindado su apoyo . . . gracias.

CONTENIDO

CONTENIDO

7. Capa de aplicación del proyecto INGRESO

INTRODUCCIÓN

A lo largo de mis estudios profesionales he adquirido conocimientos teóricos y prácticos que me han formado como profesionista. El Instituto Mexicano del Petróleo me ha dado la oportunidad de aplicar estos conocimientos a la solución de un problema real de Ingeniería en Computación.

La idea de este proyecto nace en el Departamento de Soporte Informático de la Coordinación Informática ante la necesidad de automatizar la instalación y administración de software adquirido para trabajar en modo monousuario (stand alone) haciendo uso de la red de cómputo de la Coordinación. El nombre del proyecto es INGRESO, debido a su objetivo: INstalación y Gestión REmota de SOftware.

En este documento presento los trabajos desarrollados y los resultados obtenidos dentro del marco de este proyecto. En la primera parte proporciono una introducción al proyecto INGRESO.

El primer capítulo está destinado a definir y ubicar en su medio ambiente al proyecto INGRESO, al cual he llamado : *Entorno del proyecto INGRESO*. Enseguida presento una definición detallada del problema a resolver, sus posibles alternativas de solución y justifico la elección de una de ellas; a este segundo capítulo lo he titulado: *Definición del problema*.

En una segunda parte defino aquellos conceptos teóricos indispensables para el seguimiento de la tesis.

En el tercer capítulo, abordo la noción de *Arquitectura de las redes*, piedra angular en la interconexión de sistemas abiertos, así como los trabajos de normalización realizados alrededor de este concepto. Termino esta primera parte con el capítulo 4, donde presento el *Modelo Cliente-Servidor*. Ya que bajo este modelo se encuentra el sistema desarrollado en el proyecto INGRESO.

En la tercera parte, este trabajo de tesis cubre el análisis, el diseño y la implementación del proyecto INGRESO.

En el capítulo 5, *Descripción funcional del proyecto INGRESO*, se trata de seguir una metodología de diseño, comenzando con un análisis funcional del proyecto, en él se identifican sin ambigüedades las funcionalidades de INGRESO. Este análisis es refinado hasta identificar los módulos o subsistemas que comprenderá el proyecto. Una vez identificado lo que deberá hacer cada módulo, es tiempo de pasar a la implementación. En el capítulo 6, presento la *Capa de transporte del proyecto INGRESO*, utilizada para la comunicación entre estaciones de red, así como la caja de herramientas construida sobre ella para facilitar este fin. Por último, el capítulo 7 corresponde a la *Capa de aplicación del proyecto INGRESO*. Es en esta capa donde residen básicamente mi servidor de aplicaciones, así como sus clientes.

CAPÍTULO 1

ENTORNO DEL PROYECTO INGRESO

1.1. CONTEXTO DEL PROYECTO INGRESO

La Coordinación Informática de la CIPE (Coordinación de Ingeniería de Proyectos de Explotación) del IMP ha adquirido muchas aplicaciones (software) que fueron diseñadas para trabajar en modo "stand alone", es decir fuera de red.

Además, por razones económicas se ha contratado un número limitado de licencias para cada paquete Sin embargo, es posible administrar este software utilizándolo en periodos distintos en varias computadoras personales (estaciones de trabajo de alguna red). La idea aquí es manejar los paquetes de software como hace con los libros un bibliotecario.

Por otro lado, por razones de espacio en disco no se puede tener indefinidamente todo el software instalado en las máquinas, así como un lector no puede adquirir todos los libros deseados. Una solución a este problema podría ser el contar con un servidor de aplicaciones. Esta opción es válida siempre y cuando todo el software considerado sea especialmente concebido para trabajar en red. Un servidor de aplicaciones permite la explotación tanto de la red como del software. Pero la realidad es que una gran cantidad de software que ha sido comprado, opera solo en modo monousuario.

La Coordinación ha pensado que sería de utilidad un servidor de paquetes o software, capaz de administrar en base al número de licencias contratadas las instalaciones de un paquete.

Este software vendría a completar el software adquirido para trabajar en red. A este proyecto se le ha llamado INstalación y Gestión REmota de SOftware (Proyecto INGRESO).

1.2. PRESENTACIÓN DEL *INSTITUTO MEXICANO DEL PETRÓLEO*

Los regímenes presidenciales de los licenciados Miguel Alemán, Adolfo López Mateos y Adolfo Ruiz Cortines promovieron en diferentes facetas el arranque industrial del país, impulsando la explotación de nuestras reservas petroleras, ampliando y desarrollando las instalaciones y plantas industriales de Petróleos Mexicanos como palanca para contribuir al desarrollo económico y social del país

Desafortunadamente el crecimiento de la industria petrolera se vio afectado por una considerable dependencia en tecnología del extranjero Se plantearon políticas para sustituir las importaciones de tecnología, de ingeniería, de mano de obra especializada y capacidad de construcción, pero no produjeron los resultados esperados.

Es así que en la administración del Lic. Jesús Reyes Heroles, en funciones de Director General de Pemex, se presentó la iniciativa al Ejecutivo Federal, proponiendo la creación de un organismo que apoyara a Petróleos Mexicanos en la solución de sus problemas tecnológicos y de recursos humanos.

Esta propuesta dio como resultado el Decreto Presidencial del 23 de agosto de 1965, expedido por el Lic. Gustavo Díaz Ordaz, mediante el cual se fundó el *Instituto Mexicano del Petróleo* como un organismo descentralizado, de interés público, con carácter preponderantemente técnico, educativo y cultural, con personalidad jurídica y patrimonio propios

Con la fundación del *IMP* se abrió en México la investigación y el desarrollo tecnológico en campos y temas hasta entonces no explorados

Actividades

En su artículo segundo, el Decreto Oficial establece que el Instituto Mexicano del Petróleo debe realizar las siguientes actividades:

A) La investigación científica básica y aplicada.

B) El desarrollo de disciplinas de investigación básica y aplicada.

C) La formación de investigadores.

D) La difusión de los desarrollos científicos y su aplicación en la técnica petrolera.

E) La capacitación del personal obrero para que pueda desempeñar labores en el nivel subprofesional dentro de las industrias petrolera, petroquímica derivada y química.

En el mismo decreto de creación se especificó que el Instituto Mexicano del Petróleo desarrollaría sus actividades por medio de laboratorios, plantas piloto, plantas comerciales y centros educativos en los campos de:

A) Geología y Geofísica.

B) Ingeniería petrolera, estudios de transporte y distribución de hidrocarburos, así como de problemas de economía petrolera.

C) Química, refinación y petroquímica.

D) Manejo, adaptación y diseño de equipo mecánico, electrónico y maquinaria.

E) Estudio y adaptación de tecnología.

F) Electrónica[1] aplicada a las industrias petrolera, petroquímica básica, derivada y química.

G) Seguridad Industrial.

H) Cursos de organización y administración industrial.

[1] Nótese que en aquel entonces no se contemplaba o al menos no se hacía referencia explícitamente a la computación como herramienta de apoyo y de investigación en la industria Petrolera.

Para asegurar los compromisos de Instituto, a fin de satisfacer las necesidades de Petróleos Mexicanos, así como su vinculación con los sectores académico y de investigación, la integración de su Consejo Directivo comprende a representantes de Pemex y de las principales instituciones de educación superior.

El inicio de su operación se propició con la participación de un grupo de profesionistas destacados de Pemex que contaban, además de gran capacidad técnica, con amplio conocimiento de la industria petrolera y con un conjunto de investigadores del sector académico con vasta experiencia en el desarrollo de proyectos de investigación científica.

De esta manera, se formaron grupos de geólogos, geofísicos, matemáticos, físicos, químicos, electrónicos e ingenieros petroleros y químicos, entre otros especialistas, que en colaboración con colegas conocedores de los problemas centrales de las industrias petrolera, petroquímica y química, comenzaron a definir las áreas o temas en que era conveniente desarrollar esfuerzos para apoyar las actividades de la industria y adelantar sus posibles demandas tecnológicas.

Hace 30 años no existían hombres de ciencia asociados a los propósitos concretos de una industria como la petrolera, en este aspecto el Instituto fue pionero. De hecho, puede decirse que la investigación aplicada y el desarrollo tecnológico en México, se impulsan con la creación del IMP, ya que los laboratorios y centros de investigación no académicos de entonces, no estaban ligados a una empresa productiva de la naturaleza e importancia de Pemex.

Con el mayor conocimiento de las necesidades de la industria y con el arribo de nuevos investigadores y especialistas, formados muchos de ellos a iniciativa y con el apoyo del Instituto, se empezó a contribuir significativamente al desarrollo de varios de los muy diversos proyectos que en el transcurso de su existencia ha tenido a su cargo.

Paralelamente a la creación de los grupos de investigación descritos, se desarrollaron los grupos de ingeniería de proyecto, como un componente fundamental en el desarrollo y aplicación de las tecnologías en estudio.

Asimismo, se creó una amplia estructura para capacitar a los trabajadores y establecer programas para la formación académica y de actualización profesional, tanto del personal de Pemex como del propio Instituto.

El Instituto inició sus operaciones en cuatro edificios (tres con laboratorios y uno administrativo) y una nave de incipientes talleres. Actualmente se cuenta, en la sede principal, con 33 edificios (20 con laboratorios, cuatro naves de plantas piloto y talleres y una torre administrativa) y en el Conjunto de La Reforma, en el estado de Hidalgo, con tres naves industriales de laboratorios.

Además, se cuenta con cuatro Zonas Foráneas que comprenden 32 centros de capacitación, oficinas y laboratorios de otras dependencias del Instituto, con el fin de servir a Pemex en las zonas petroleras más importantes.

La entidad inició sus actividades con 300 personas, técnicos que provenían de diferentes dependencias de Pemex e investigadores de diversas instituciones del sector educativo. Debido a las demandas de proyectos y servicios solicitados por Pemex, el personal llegó a alcanzar en el año de 1987 la cantidad de 6 mil efectivos.

Durante este tiempo y como resultado de las diferentes reestructuraciones que ha tenido el Instituto, se requirió de una mayor sistematización y automatización de los procesos administrativos, tanto en lo referente al ordenamiento de la información interna para el control y la toma de decisiones, como para la demanda de información de las diferentes Secretarías de Estado del gobierno federal.

En el proceso de reestructuracion reciente de Petróleos Mexicanos, el IMP ha procedido a adecuar su organización y aquellos procesos administrativos y servicios que se deben proporcionar interna y externamente. Asimismo, se inició un redimensionamiento racional de los efectivos del Instituto, contándose actualmente con una plantilla de 4,300 empleados.

1.2.1. Coordinación de Ingeniería de Proyecto

Dentro del Instituto Mexicano del Petróleo se encuentra la Coordinación de Ingeniería de Proyecto, la cual a su vez está dividida en las siguientes tres ramas: la Subdirección de Ingeniería de Proyectos de Plantas Industriales (*SIPPI*), la Subdirección General de Ingeniería de Proyectos (*SGIP*) y la Coordinación de Ingeniería de Proyectos de Explotación (*CIPE*). Esta división se muestra en la Figura 1.1.[2]

La tendencia mundial en el desarrollo de proyectos es la ingeniería integral asistida por computadora. El IMP cuenta con varios años de experiencia en la aplicación de esta tecnología.

La ingeniería de proyecto en el Instituto Mexicano del Petróleo presenta características singulares con relación a las organizaciones tradicionales de ingeniería, que se originan por su ubicación en una entidad dedicada preponderantemente a la investigación y al desarrollo tecnológico; de esta manera, en el IMP se ve la ingeniería no sólo como un area de suministro de servicios, sino también como una oportunidad de desarrollo de tecnología.

[2] La estructura organica del IMP actualmente sufre modificaciones, por lo que pudiera existir ligeras variaciones al momento de leer el presente documento.

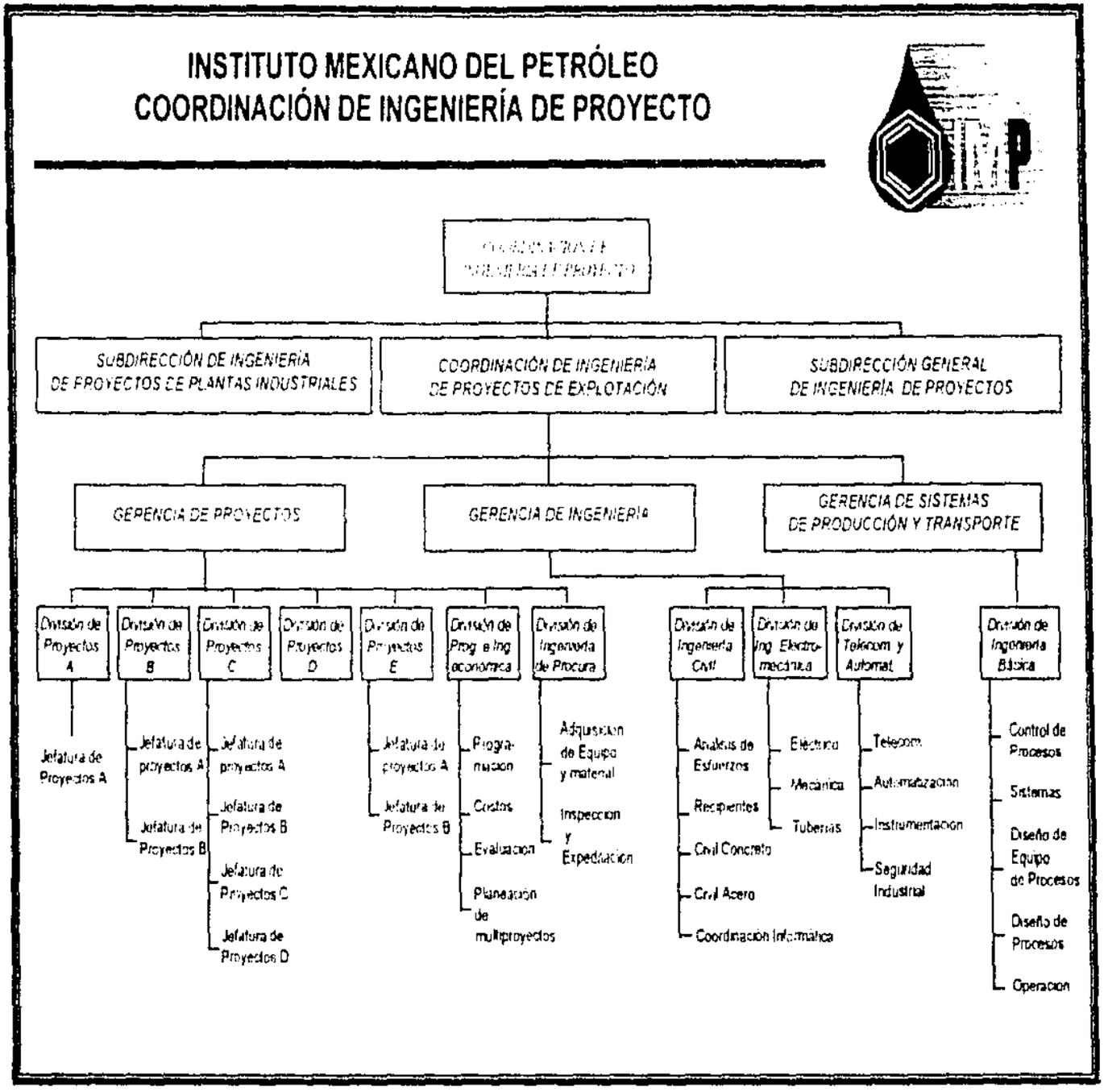


FIGURA 1.1. Coordinación de Ingeniería de Proyecto

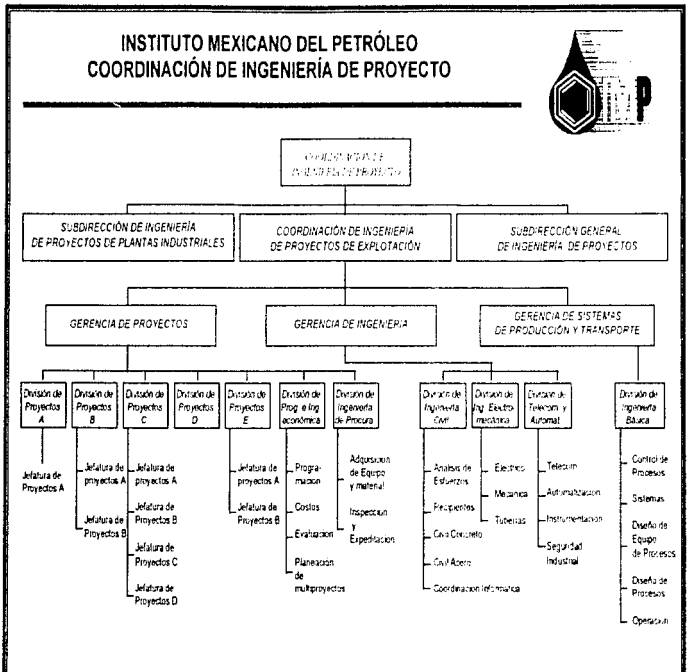


FIGURA 1.1. Coordinación de Ingeniería de Proyecto

La actividades de Ingenieria de Proyecto son:

1) Establecer las bases de diseño de los proyectos.
2) Efectuar la ingenieria básica de proyectos.
3) Llevar a cabo la ingenieria de proyecto.
4) Efectuar los trámites de compra de equipo y materiales.
5) Ayudar en la supervisión de construcción de las instalaciones proyectadas.
6) Elaborar las normas de proyecto y construcción.

Tomando en cuenta que el proyecto INGRESO se llevará a cabo en la Coordinación de Ingeniería de Proyectos de Explotación, -cuyos objetivos son el efectuar los estudios de factibilidad, la ingenieria de proceso y diseño de las instalaciones que requieren las industrias petroleras, petroquímica y química en general- en la Figura 1.1. se desglosa esta Coordinación, a continuación se ubica el área especifica donde se efectuará este trabajo.

Coordinación de Ingeniería de Proyectos de Explotación

La Coordinación de Ingeniería de Proyectos de Explotación se divide en tres Gerencias:

Gerencia de Proyectos
Gerencia de Ingeniería
Gerencia de Sistemas de Producción y Transporte

Como puede observarse en la Figura 1.1., la Gerencia de Ingeniería está integrada por:

División de Ingeniería Civil
División de Telecomunicaciones y Automatización
División de Ingeniería Electromecánica

Dentro de la División de Ingeniería Civil se encuentra la Coordinación Informática, la cual está dividida en tres departamentos, como se muestra en la figura 1.2.

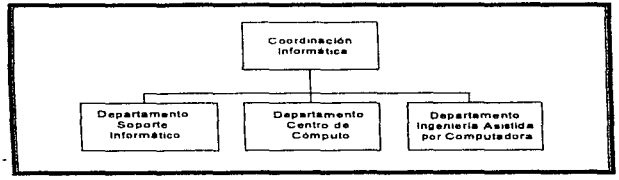


FIGURA 1.2. Coordinación Informática.

Precisamente, en el Departamento Soporte Informático perteneciente a la Coordinación de Informática, se llevará a cabo la implementación del proyecto INGRESO.

1.3. RECURSOS INFORMÁTICOS DEL *IMP*

Simbolo de la modernidad, los equipos de cómputo se han convertido en apoyo fundamental para el desarrollo tecnológico. Por ello, las unidades de computación y procesamiento de datos del *IMP* son constantemente actualizadas, al igual que lo son los programas adquiridos y los desarrollados, que han permitido la estructuración de una amplia gama de simuladores demandados por los diferentes grupos de investigación.

Con los laboratorios extensamente instrumentados y la expansión de las facilidades de cómputo, los diversos sistemas informáticos del *IMP*, constituyen uno de los sistemas computacionales más poderosos de Latinoamérica.

1.3.1. Software

El software ha llegado a ser el elemento clave de la evolución de los sistemas y productos informáticos. Lo que diferencia a una compañía de su competidora es la suficiencia y oportunidad de la información dada por el software.

El *IMP* adquiere mucho software, pero también desarrolla aplicaciones especializadas que resuelven las necesidades internas del Instituto.

Recientemente se llevó a cabo una revisión tanto de la situación Institucional del *IMP* como del avance que se ha dado en el campo del Software para PC's y se llegó al acuerdo de proponer como *paquetes institucionales* para computadoras personales los listados en la Tabla 1.1.

Sistemas Operativos	Dos, Windows NT, AIX y Novell (para redes)
Procesador de Palabras	Word y WordPerfect
Hoja de Cálculo	Lotus 1-2-3 y Excel
Base de Datos	Clipper 7.0 (1), DBASE y Oracle (2)
Graficación	Harvard Graphics, Freelance, Corel Draw y Power Point
Antivirus	Familia de Scan y Solomon
Front-End para Bases de Datos	SQL-Windows (2), UNIFACE

(1) Queda solo para aplicaciones ya existentes y actualmente en producción, como una facilidad de soporte a corto plazo.
(2) Recomendado como Base de Datos Institucional

TABLA 1.1. Paquetes Institucionales.

Es difícil establecer categorías genéricas para las aplicaciones del software que sean significativas. Conforme aumenta la complejidad del software, es más difícil establecer comportamientos nitidamente separados

Las tablas 1.2, 1.3 y 1.4 muestran el software principal con el que cuenta el *IMP*, e indican la amplitud de las posibilidades de aplicación.

ESPECIALIDAD	NOMBRE DEL SOFTWARE	OBJETIVO
INGENIERÍA DE PROYECTOS		
Programación	PRESTIGE	Administración de Proyectos
	TIME LINE	Administración de Proyectos
	SICOP	Sistemas de Control de Proyectos
INGENIERÍA BÁSICA		
Diseño de Proceso	SIMPROC	Simulación de Procesos
INGENIERÍA CIVIL		
Análisis de Esfuerzos	SUBMARINE PIPELINE	Análisis de Estabilidad Hidrodinámica
	ON BOTTON STABILISTY	
	ADA	Análisis de Ductos Ascendentes
INGENIERÍA ELECTROMECÁNICA	SMART-CAM	Simulación de Procesos de Manufactura de Máquinas-Herramientas
	YORK	Cálculo de Carga Térmica para Inmuebles

TABLA 1.2. Software de Ingeniería para PC's.

ESPECIALIDAD	NOMBRE	OBJETIVO
MULTIDISCIPLINARIOS	AUTOCAD VER 11	Diseño Asistido por Computadora dentro de Ingeniería
	MICROSTATION VER 4 0 Y 2 3	Diseño Asistido por Computadora
	ANSYS-PC/LINEARVER 4 4	Análisis Estructural de Propósito General
	SAP 86	Análisis Estructural de Propósito General
	3 DM	Modelado en Tres Dimensiones
ING ASISTIDA POR COMPUTADORA	CAD CORE TRACER VER 5 0	Vectorización de Planos
	CAD OVERLAY	Vectorización de Planos
	STATGRAPHICS	Estadístico

TABLA 1.3. Software de Propósito General en el Área de Ingeniería.

NOMBRE	APLICACIÓN
MARCS	Cálculo de Estructuras Costa - Fuera
VIPLEF	Análisis de Domos Salinos
3 DM	Modelado de Tres Dimensiones
NONSAP	Análisis Estructural No Lineal de Propósito General
WALKTHRU	Simulación en Tiempo Real en Modelos Tridimensionales en Computación
VMS 5.5.2	
VMS 6.0	
VMS 6.1	
DECWINDOWS	
PATHWORKS 4.2	
VAX C 3.2	
FORTRAN 6 0-1	
BLAST	
DISKEEPER/PLUS 6 0	
PASCAL 3.8	
BASIC 3.3	
COBOL 4 1	
DATA TRIEVE	
PCSA 3.0	
NCR FUSION 3 37	
ZIM 3 0	
ORACLE 7.0	

Tabla 1.4. Software de los sistemas VAX

PAQUETE	VERSION
1 APPLAUSE II	
2 AUTOCAD	10
3 AUTOCAD	11
4 BLAST PLUS PC	
5 BORLAND C ++	2
6 CHI-WRITER	4.2
7 CLIPPER	5
8 COPY II PC	6
9 COREL DRAW	2
10 DBASE IV	1.5
11 FLOW CHARTING	3
12 FORM TOOL	3
13 MICROSOFT FORTRAN	5.1
14 FREELANCE GRAPHICS	4
15 FREELANCE FOR WINDOWS	1
16 GRAND VIEW	2
17 HARVARD GRAPHICS	2.3
18 HARVARD PROJECT MANAGER	3.03
19 LOTUS 1-2-3	3.1
20 LOTUS 1-2-3 FOR WINDOWS	1.1
21 MICROGRAFX	1
22 MICROSOFT MS-DOS	5
23 THE NORTON ANTIVIRUS	1.5
24 THE NORTON COMMANDER	3
25 THE NORTON EDITOR	2
26 THE NORTON UTILITIES	6
27 PARADOX	3.5
28 PC TOOLS, CENTRAL POINT	7
29 PRINTCACHE	2.5
30 QUATTRO PRO	3
31 STATGRAPHICS	5.01
32 TIME LINE	4
33 TRANSLATE	1.1
34 TURBO C++	2
35 TURBO DEBUGGER & TOOLS	2
36 TURBO PASCAL PROFESIONAL	6
37 MICROSOFT MOUSE	
38 MICROSOFT WINDOWS	3
39 MICROSOFT WORD	3.1
40 WORDPERFECT	5.5
41 WORDSTART	5.1
42 X-TREE	6

Tabla 1.5. Discos de Software de la Coordinación de Informática

1.3.2. Hardware

Los avances de la microelectrónica han dado como resultado una mayor potencia de cálculo a la vez que una reducción del costo del hardware de las computadoras. Sin embargo, como la tecnología actual son las redes de computadoras, el *IMP* la ha adoptado y no sólo se requiere de PC's, sino tambien de sistemas operativos especializados, tarjetas de interfaz para red y cables, entre otros muchos detalles, los cuales arrojan una gran suma de dinero .

El *IMP* se esfuerza en actualizar el equipo de cómputo, pero la sofisticación del hardware y el gran número de equipo requerido, han dejado desfasada la capacidad de renovar continuamente el equipo.

En la tabla 1.6 se puede apreciar el equipo perteneciente a la Coordinación de Ingeniería de Proyectos de Explotación.

Por otro lado, el Instituto cuenta con dos super microcomputadoras VAX II. El nombre VAX vienen de Virtual Address eXtension, lo cual significa extensión de direccionamiento virtual. Las VAX tienen gran capacidad de direccionamiento virtual y de memoria virtual, lo que les permite procesar programas más grandes que el tamaño de su memoria.

La demanda de los proyectos y los desarrollos de ingeniería, orillaron a la adquisición de tres nuevos sistemas de tiempo compartido: VAX 4000 modelo 500. Las cinco máquinas cuentan con una buena cantidad de software a disposición de los usuarios como se puede observar en la Tabla 1.4.

CANTIDAD	MARCA/MODELO	PROCESADOR	RAM	DISCO DURO	MONITOR	CONEXIÓN A RED	SI HAY CONEXIÓN S=Servidor E=Est. de Trab.	NOMBRE O IDENT. DE LA RED
11	PRINTAFORM	8088	640KB	20	CGA			
4	PRINTAFORM	80286/1MHZ	1MB	40	CGA			
2	TEXA TURB10	80286/12MHZ	1MB	40	EGA			
1	ACER/915V	80286/16MHZ	5MB	80	VGA			
10	ACER/PS25DP	80386 SX/25	4MB	100	VGA	3	3E	NOVELL,WFG
5	ACER 1100/25	80386/25MHZ	4MB	100	VGA			
10	LEADING EDGE	80386/33MHZ	5MB	100	VGA			
22	ACER 1100/33	80386/33MHZ	8MB	100	VGA			
20	ACER 1100/33	80386/33MHZ	4MB	120	VGA	6	6E	NOVELL,WFG
17	ACER 1100/33	80386/33MHZ	4MB	100	VGA			
10	ACER 1200	80486/25MHZ	4MB	200	CGA	4	3E,1S	NOVELL,WFG,DECNET
20	ACER P433	80486/33MHZ	4MB	170	SVGA	2	2E	WFG
14	ACER P433	80486/33MHZ	4MB	170	SVGA	10	10E	DECNET
11	ACER P433	80486/33MHZ	8MB	170	SVGA	8	3S 5E	NOVELL,WFG,DECNET
33	ACER	Pentium 100MHZ	8MB	540	SVGA			
9	ACER	Pentium CDROM	16MB	1GB	SVGA			

Tabla 1.6. Equipo existente en la Coordinación de Ingeniería de Proyectos de Explotación.

El *IMP* recientemente ha adquirido sistemas RS/6000 de IBM, los cuales se basan en un chip RISC. Los procesadores RISC (Computadora de conjunto de instrucciones reducido, por sus siglas en inglés) corren a velocidades de los 50 MHz en adelante. El procesador RISC emplea un juego de instrucciones más simple que los microprocesadores regulares, y esto permite varias optimizaciones que pueden dar como resultado un rendimiento extremadamente alto. La Coordinación de Ingeniería de Proyectos de Explotación (CIPE) cuenta un equipo de estos.

1.3.3. Comunicación

Debido a la necesidad de comunicación entre el Instituto Mexicano del Petróleo y Petróleos Mexicanos, surgió la idea de poder comunicar los equipos de cómputo del *IMP* con los de *PEMEX*. Es así como el *IMP* decide implantar una red institucional, a la cual se le denominó **red IMP**. Por su parte PEMEX ya contaba con su red PEMEXPAQ.

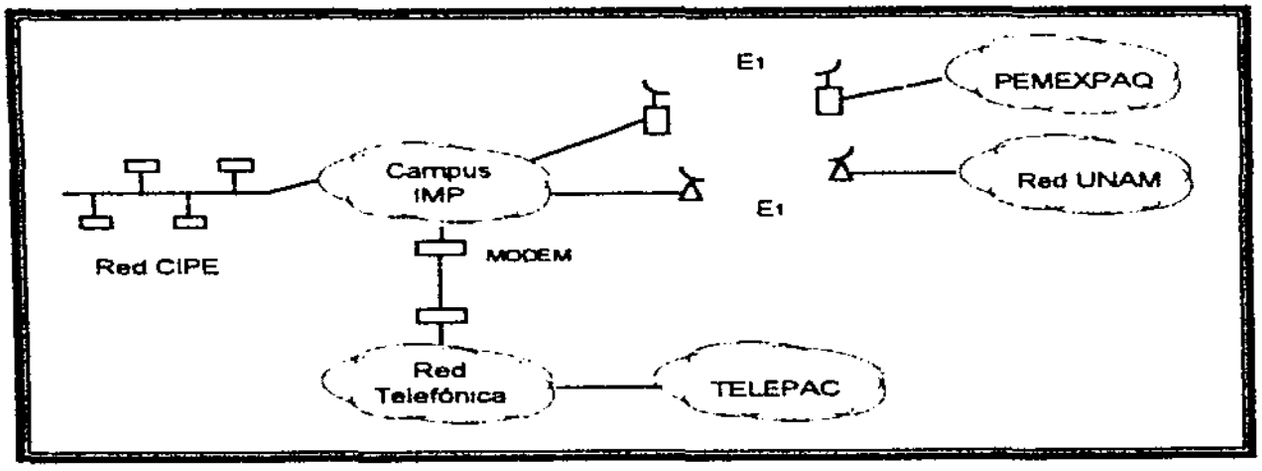


Figura 1.3. Red *IMP*

Actualmente la red IMP cubre todo el campus IMP sede con un cableado de fibra óptica que enlaza a todos los edificios, dos enlaces de microondas que enlazan a la red PEMEXPAQ y a la red INTERNET vía red UNAM (véase figura 1.3).

INTERNET es una red compuesta por más de 20,000 redes, conecta alrededor de 2,000,000 computadoras y la acceden alrededor de 25,000,000 personas diariamente. Se puede enviar correo electrónico a más de 100 países y hay conexiones Internet en más de 60 países en todos los continentes.

Entre los servicios que presta la red IMP se pueden mencionar:

- Enlace de las redes de Administración, Ingeniería de Proyecto, Explotación y Producción, Transformación Industrial y Capacitación-Servicios Técnicos.
- Enlace a la red INTERNET y a la red PEMEXPAQ.
- Transferencia de archivos entre redes remotas.
- Permite el correo electrónico (local e internacional).
- Permite el establecimiento de sesiones remotas y teleproceso.
- El servicio de información en línea a través de servidores Word Wide Web.
- Adquisición de software libre y con cargo.
- Noticias internacionales de actualidad, etc.

CAPÍTULO 2

DEFINICIÓN DEL PROBLEMA

2.1. DESCRIPCIÓN DEL PROBLEMA

La Coordinación Informatica de CIPE maneja una biblioteca (en diskettes) de los paquetes tipo monousuario utilizados en la Coordinación, los cuales son facilitados a los usuarios cuando los necesitan.

Cada paquete cuenta con un número limitado de licencias contratadas, esto significa que cada aplicación puede estar instalada legalmente sólo en un cierto número de computadoras personales.

La instalación y desinstalación de un paquete en una estación de la red, es responsabilidad del usuario. Si éste desea tener un paquete determinado hay que ir por los discos, registrarse y correr hacia su computadora a instalarlo. Cuando el usuario desinstale un paquete por liberar espacio en disco o por que no lo necesite actualmente y más tarde vuelve a requerirlo, tendrá que repetir el proceso.

Esta forma de instalación causa varios problemas. La distribución de las licencias es casi imposible de detectar, aún cuando se lleve un control de a quién se le otorgó una licencia. ¿Cómo saber si se ha liberado una licencia?, es decir, si algún usuario

desinstaló un paquete. ¿Cómo evitar que un paquete se instale en distintas máquinas con sólo una licencia registrada?

Otro problema evidente es que un usuario puede monopolizar la licencia de un paquete sin hacer uso de él; mientras que a otro usuario que realmente lo requiera, no se le otorgará por no haber licencias disponibles.

Aún contando con *n* licencias (*n* = número de máquinas disponibles) para cada paquete, no es posible tener al mismo tiempo todo el software instalado en las computadoras, por falta de espacio en disco duro.

El IMP ha invertido mucho en software *stand alone* (monousuario) y aunque quizás ya existan productos para red de este software, se desea seguir utilizándolo.

2.2. ALTERNATIVAS DE SOLUCIÓN

☞ Una solución a este problema podría ser el contar con un servidor de aplicaciones. La noción de servidor de aplicaciones es utilizada dentro del modelo cliente-servidor para designar aquella máquina y software que permiten administrar aplicaciones (programas) especialmente concebidas para trabajar en red. Esto significa que pueden ser ejecutadas al mismo tiempo por varios usuarios. En este caso la aplicación misma se encargada de resolver los conflictos generados cuando los usuarios tratan de acceder recursos críticos (recursos que existiendo en un número limitado pueden ser usados por varios usuarios). Para ello incluyen capacidades de bloqueo de archivos y registros que evitan que más de un usuario pueda utilizar simultáneamente el mismo archivo o registro de base de datos.

Dentro de este grupo de software distinguimos las llamadas aplicaciones intrinsecas para redes, las cuales distribuyen diversas tareas del proceso entre el servidor y las estaciones de trabajo. También reciben el nombre de aplicaciones cliente-servidor o aplicaciones front-end/back-end.

Aunque es posible poner en un servidor de aplicaciones el software que ha sido diseñado para trabajar en modo monousuario, las posibilidades de que dos usuarios puedan usarlo al mismo tiempo son mínimas . Esta solución no es adecuada, ya que este software no posee la capacidad necesaria para proteger los archivos y datos en un entorno multiusuario (integridad de la información).

☞ Como se dijo en la definición del problema, las aplicaciones se adquieren para un número limitado de computadoras personales, por razones sobre todo de carácter económicas. Pero este software puede reproducirse e instalarse en todas las estaciones de trabajo de la red. Sin embargo, esta alternativa representa un acto de piratería que

puede implicar serias consecuencias jurídicas tanto para la persona que lo hace, como para el Instituto.

☞ Por otro lado, dependiendo del proyecto en que se trabaje es común que se utilice sólo un cierto número de herramientas para cada una de sus faces. Es posible entonces, después de la terminación de cada una de estas faces, liberar estas licencias para que puedan asignarse a otros usuarios. Esta solución implica que los usuarios sean responsables de desinstalar las aplicaciones que ya no requieran y que la Coordinación este enterada. Para esto, la Coordinación podría contar con una persona encargada de verificar cada determinado intervalo de tiempo si se ha liberado alguna licencia, y así mantener el registro actualizado. Esta persona también podría encargarse de vigilar que las aplicaciones distribuidas estén siendo utilizadas, de no ser así se exigiría la desinstalación.

☞ También por razones de espacio en disco es evidente que no se puede tener indefinidamente instalado sobre una PC todo el software (que algún día quizás utilice el propietario). Una solución a este problema es "subir y bajar" este software de disco duro a diskette (o cinta) y viceversa, cada vez que sea necesario. Así solo se tendrá en disco duro las aplicaciones o herramientas que se necesiten actualmente y una gran pila de diskettes o cintas con el software que algún día quizás se utilice.

✓ Analizando las propuestas anteriores, se pueden integrar algunas ideas de éstas y plantear una última alternativa de solución: automatizar el proceso de instalado y desinstalado de paquetes, tomando en cuenta el modelo cliente-servidor para gestionar las licencias de cada paquete.

2.3. SOLUCIÓN RETENIDA

El tradicional sistema de instalación de paquetes tipo monousuario causa varios problemas, todos ellos con la tendencia en una misma dirección: la necesidad absoluta de pasar por medios electrónicos el proceso de instalado y desinstalado de paquetes.

Para ello se propone el diseño de un servidor de instalaciones a distancia de paquetes (software). Este sistema tendrá como función administrar el número de licencias contratado para cada paquete. A priori se detectan dos funciones principales: instalar y desinstalar paquetes. Para evitar que un usuario monopolice la licencia de un paquete sin hacer uso de él, se implementarán funciones de recuperación de licencias cuando se detecte que el paquete no ha sido utilizado después de un tiempo x o bien se le permitirá al usuario que por si mismo desinstale el paquete cuando ya no lo necesite o desee liberar espacio en disco.

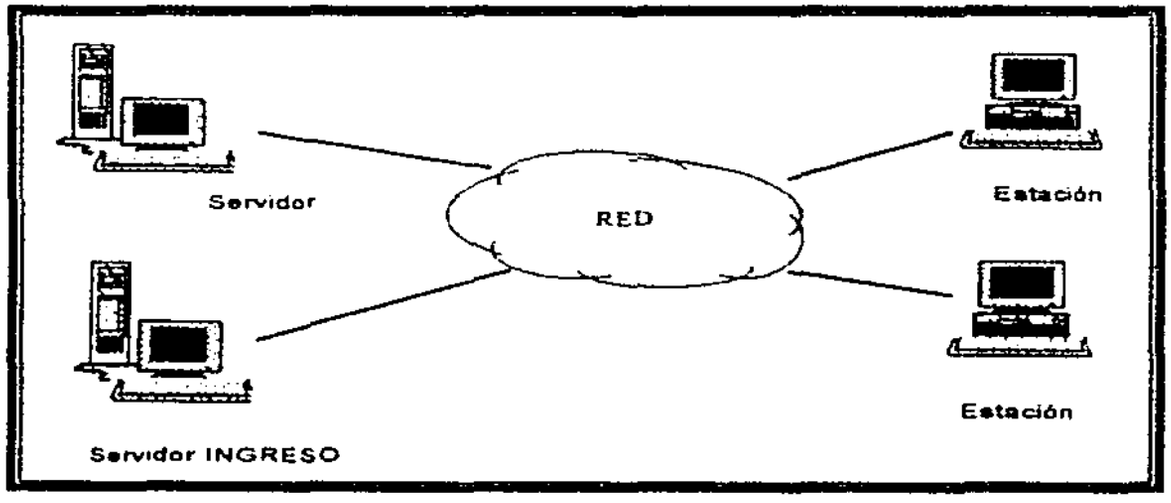


FIGURA 2.1. Servidor INGRESO.

CAPÍTULO 3

ARQUITECTURA DE LAS REDES

3.1. IMPORTANCIA DE LA ARQUITECTURA DE LAS REDES

Debido a que la industria informática ha desarrollado sus sistemas según diversos estándares, siempre ha sido difícil conectar sistemas distintos para formar redes útiles. Esto cambiará a medida que más y más fabricantes hagan que sus productos cumplan con una serie de estandares que posibiliten que los dispositivos se entremezclen de diversas maneras, dando así a los usuarios de los distintos sistemas el mismo acceso a los recursos de la red (productos *interoperativos*). De este modo, cualquier producto a desarrollarse debe ser lo suficientemente versátil como para adaptarse no solamente al nuevo ambiente, sino también para poder ajustarse a nuevos factores que surjan en el futuro.

Un sistema de computación se diseña a partir de algún tipo de arquitectura que describa las características del sistema.

Una arquitectura de red es una estrategia comercial para solucionar problemas de compatibilidad en una línea de productos, mediante un conjunto de especificaciones técnicas que definen las interrelaciones entre las partes de un sistema y la distribución de funciones entre los componentes de una red. Una buena arquitectura debe definir qué se conecta con que y cómo se conecta.

Los objetivos que una arquitectura debe satisfacer son los siguientes:

A) Hacer la red transparente para el usuario final y programador de aplicaciones (ocultar complejidades).

B) Cualquiera de los elementos de la red (productos y sistemas) deben contar con la posibilidad de ser trasladados hacia y desde las instalaciones especificas, con un mínimo de modificaciones al producto durante la transición.

C) Permitir que sistemas centrales múltiples u otros dispositivos inteligentes sean conectados a la misma red.

D) Habilitar terminales funcionalmente diferentes.

Dicho de otra forma, una arquitectura de red es un conjunto de protocolos, reglas y criterios de diseño que definen cómo va a funcionar una red y cómo han de diseñarse los productos hardware y software para funcionar en ella. Si se cumplen adecuadamente las reglas, los productos de diversos fabricantes podrán funcionar con la red sin problemas.

3.2. PROTOCOLOS

En las redes de computadoras continuamente se intercambia información entre entidades. Una *entidad* es cualquier componente de la red (software o hardware) capaz de producir o consumir información. Para que el intercambio de información pueda realizarse en forma ordenada, es necesario un conjunto de reglas llamado *protocolo*.

3.2.1. Funciones de los protocolos

Los protocolos son necesarios para reglamentar una serie de aspectos relacionados con el intercambio de información. Las funciones principales de los protocolos se describen a continuación.

- El protocolo, primeramente es responsable de establecer especificaciones sobre cuál es la *unidad* de información que va a ser intercambiada entre las entidades participantes.

- Un segundo aspecto definido por el protocolo es la creación de *convenciones,* como la definición del código de representación de las unidades que son intercambiadas, formatos utilizados, cuáles son las velocidades y qué controles pueden ser aplicados para reglamentar la transferencia.

- Para que una red de computadoras sea un medio de comunicación efectivo, el protocolo debe definir cómo se realizará la identificación entre entidades, a esta función se denomina *direccionamiento.*

- Por las características físicas presentes en la red, se pueden introducir errores en la información intercambiada. Además, los mecanismos utilizados en la operación de la red, tal como el control de congestionamiento, pueden causar la pérdida de unidades de información. Otra función de un protocolo es definir cómo se hará la *recuperación de errores*.

- Ciertas redes pueden operar internamente un modo datagrama, este servicio es utilizado para el intercambio de mensajes, pero no garantiza el ordenamiento de paquetes, es decir, no hay relación entre el orden de transmisión y el orden de llegada al nodo de destino. Por lo tanto, para ofrecer un servicio más confiable, el protocolo debe controlar la *secuencialización de los paquetes* (servicio de circuito virtual)

- El protocolo debe también definir un mecanismo de *control de flujo*, que a su vez define cuál es la técnica utilizada, cuál la unidad sobre la que se ejerce el control y cómo se relaciona el control de flujo con los demás mecanismos, por ejemplo el mecanismo de secuencialización. Relacionado con el control de flujo está el *control de prioridades*, en caso de que exista más de un nivel de prioridades entre los posibles diversos flujos de información regidos por el protocolo.

- Cuando se tiene un gran volumen de datos a transferir durante un cierto periodo de tiempo, se puede permitir la optimización de los recursos de la red, del control de flujo y de la recuperación de errores, a través de una *conexión* entre las dos entidades. El protocolo establece las reglas para el establecimiento y término de las conexiones.

3.2.2. Organización de la red en capas

Debido a la complejidad que la comunicación en una red presenta cuando se encara como un sistema, las funciones se organizan en forma jerárquica, es decir en niveles (o capas). Es más fácil construir productos de una manera estructurada, en la que cada capa añada funcionalidad hasta lograr un sistema que haga lo que se desea (divide y vencerás). Si se tiene que hacer modificaciones y el sistema esta bien construido, sólo se tendra que modificar una capa en lugar de rediseñar todo el sistema.

Los protocolos de red como el modelo OSI definen niveles de comunicación entre distintos sistemas de la red. Los niveles definen desde cómo se va a presentar la información en la pantalla del usuario, hasta el nivel en el que se crean los paquetes y se envían a través del sistema de cableado de la red.

Una característica común a las arquitecturas, es esta estructura en niveles en cada nodo de la red. Aunque el número exacto de capas de funciones apropiadas todavía no es restringido, ha surgido de la experiencia un cierto consenso. Las capas básicas de las arquitecturas son: de transmisión, administración de servicios y funciones de aplicación.

- ***Funciones de transmisión.*** La capa de transmisión se ocupa del direccionamiento (ruteo) y movimiento de datos entre el origen y el destino. Esta capa maneja el camino y las conexiones a nivel de enlace. Es responsable del formato específico de los datos. Incluye también control de error para asegurar que los datos se entreguen correctamente y si no es así, para corregirlos o retransmitirlos. La ruta exacta que deben seguir los diferentes mensajes la determina esta capa.

- ***Administración de Servicios.*** La segunda capa proporciona transformaciones específicas según el dispositivo de las características del emisor a las necesidades del receptor. Esta capa es clave para la confección de redes, ya que proporciona las funciones de traducción entre las características de terminales físicas y programas de aplicación. La capa de administración de servicios tiene también otros dos papeles fundamentales: control de la red y control de sesión. El control de sesión vigila el estado de la sesión, coordinando la conexión y brindando apoyo para comenzar, borrar y resincronizar los flujos de datos relacionados con la sesión.

- ***Aplicaciones*** La capa de aplicación se ocupa de las funciones del usuario final y puede ser un programa de aplicación o un dispositivo de E/S, tal como una terminal. Los servicios de la capa de administración de servicios se invocan por pedidos, desde la capa de aplicación.

Por lo tanto, las capas de las arquitecturas definen cómo se establece una sesión, cómo se empaqueta la información y cómo se envía a través de la red. Parte de las reglas definen cómo se entrega la información a la placa de interfaz de red y cómo la transmite dicha placa en el cable de la red. En el extremo de recepción para la transmisión, el proceso será inverso. Si se envía un mensaje a otro usuario, el mensaje aparecerá en su pantalla una vez que ha sido desempaquetado y procesado por los protocolos.

Algunas capas de las arquitecturas tiene múltiples protocolos. Estos protocolos no son alternativos, o sea, no se pueden usar solamente uno o dos de ellos. Cada protocolo realiza una función diferente que se encarga de un aspecto específico en el establecimiento y el mantenimiento de las comunicaciones.

Cada nivel puede ser utilizado por un programador para crear distintos tipos de aplicaciones que funcionen sobre la red.

3.2.3. Servicios

Como se acaba de mencionar, el sistema de comunicaciones de las redes de computadoras están normalmente organizado en niveles, el nivel más elemental esta constituido por la conexión física entre los equipos de la red. A partir de este nivel básico, los sucesivos utilizan el servicio ofrecido por el nivel inmediatamente inferior, sumando a éste nuevas funciones que a su vez, son ofrecidas al nivel inmediatamente superior en forma de un servicio más sofisticado.

Para poder llevar a cabo la comunicación entre dos niveles adyacentes, es necesaria la definición de una *interface* entre los mismos. Esta interface define las operaciones (servicios) disponibles, cómo acceder a ellos y cuáles son los formatos y convenciones utilizados.

Los servicios de usuario proporcionados por un nivel son especificados por un grupo de operaciones llamadas *primitivas*. Asociado con cada primitiva, está un grupo definido de parámetros.

Normalmente, una transferencia particular comienza por la petición primitiva (con los valores de los parámetros permitidos) del usuario del nivel, a través de la interface del nivel.

Las primitivas de servicios a usuarios asociados con un nivel son transferidos entre los niveles, utilizando una estructura de datos conocida como un bloque de control de eventos o ECB.

Entre los servicios de aplicación podemos encontrar:

- ***Soporte de nombres y/o direcciones***. En un sistema de red es importante la identificación de un proceso de aplicación del usuario y la localización en la red en la cual el proceso reside. La identificación de un proceso de aplicación del usuario es normalmente por medio de un nombre simbólico, mientras que la localización en la red es mediante una dirección. Esto permite a las aplicaciones LAN dirigir los mensajes a los adaptadores especificos e indicar que adaptador ha originado el mensaje.

- ***Soporte de datagramas***. Permiten enviar y recibir datagramas. Un datagrama es un mensaje que se envia de una computadora a otra sin garantía de recepción.

- ***Soporte de circuito virtual***. Se establece una conexión y permite el intercambio simultáneo información conservando el orden de emisión de los paquetes. Existe un control de flujo para evitar pérdida de información en caso de dispositivos funcionando a velocidades distintas.

- ***Servicios generales***. Se encargan de reiniciar el adaptador de interfaz de red, obtener su estado y otras funciones de control.

3.3. MODELO DE REFERENCIA OSI

La organización de estandarización más importante en la actualidad es la International Standard Organization, la cual desarrolló un modelo de referencia para la comunicación de datos conocida como "Open Systems Interconnection" (OSI, Interconexión de sistemas abiertos).

Este modelo es estratificado y se estructura en siete capas. Las primeras cuatro capas realizan el transporte de la información. Las tres capas inferiores, constituyen un estándar muy difundido, que se conoce como X.25, y tienen que ver con el tratamiento de la información.

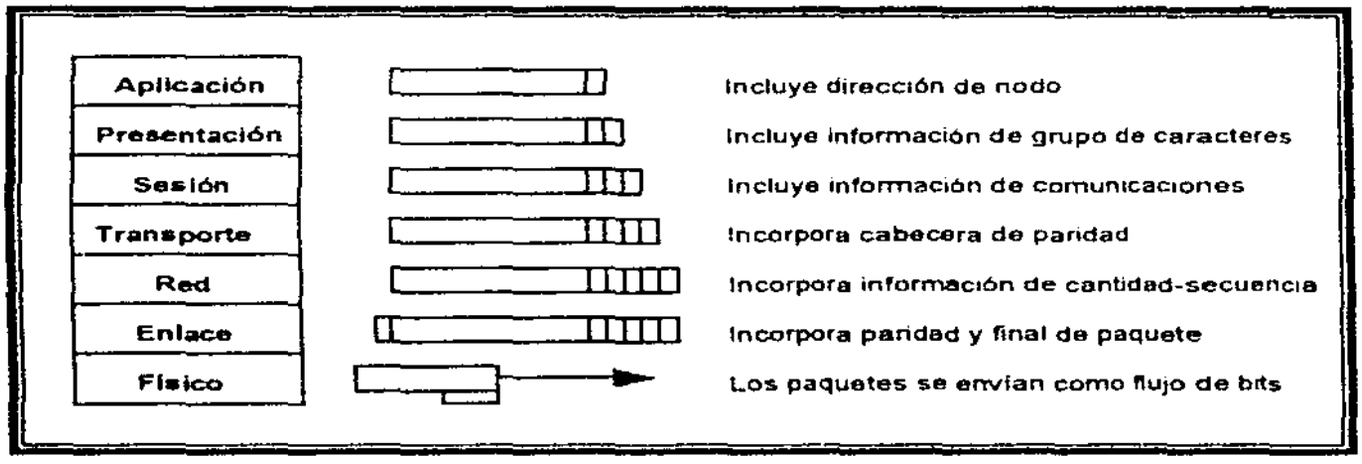


Figura 3.1. El protocolo por capas OSI y su efecto sobre los paquetes.

En cada capa se definen reglas y procedimientos específicos que todo programador y fabricante de productos para redes habrá de utilizar para diseñar productos interoperativos.

Las cuatro primeras capas, conforman el Subsistema de Transporte y ofrecen los servicios de circuitos virtuales y datagramas.

El modelo se presenta normalmente como una sucesión de procesos que tienen lugar cuando se pasan mensajes o datos desde una aplicación que se está ejecutando en una estación de trabajo a la red física. A la inversa, describe los procesos contrarios para cuando se recibe un paquete de la red y se procesa para su uso por una aplicación.

A continuación se describe cada nivel de protocolo por capas OSI tal y como se ilustra en la Figura 3.1. Para describir cada capa se utiliza un supuesto mensaje que un usuario envía a otro.

Capa de aplicación. La capa de aplicación actúa como una ventana a través de la cual la aplicación adquiere acceso a todos los servicios proporcionados por el modelo. Establece y controla el ambiente en el cual las aplicaciones pueden realizar sus operaciones. Todas las otras capas existen en función de brindar soporte a ésta. Su propósito es proporcionar a las aplicaciones servicios de comunicación. El sistema operativo de red y sus aplicaciones están en sí disponibles para el usuario en este nivel. Estas incluyen transferencia, acceso y administración de archivos, también servicios generales de intercambio de mensajes y documentos, como el correo electrónico. Esta sección contiene los programas de usuario y de aplicación.

El emisor escribe un mensaje y lo dirige al receptor.

Capa de presentación. La capa de presentación es la responsable de la transformación, el formato y la sintaxis (reglas que gobiernan el formato) de los datos que son transmitidos en una sesión: compresión de texto, codificación, decodificación, conversión de formatos de archivo, etc. Las estaciones de trabajo interconectadas pueden representar los caracteres, números, directorios y otra información de formas distintas. El nivel de presentación puede servir como un traductor entre las estaciones y fija el formato de información que se visualiza en las pantallas. Por ejemplo, si el mensaje va de un PC basado en el DOS a un Macintosh, la forma en que se escriben los caracteres en la pantalla es ligeramente distinta.

La capa de presentación añade la información de formato y pasa el mensaje a la capa de sesión.

Capa de sesión. Ofrece a los usuarios el acceso a la red. También permite a dos usuarios establecer una conexión, llamada sesión. Para esto el usuario debe entregar una dirección con la cual desea conectarse. El establecimiento de una sesión implica el intercambio de parámetros, tales como la identificación del usuario, modo de transmisión, opciones de fiabilidad, etc. La capa de sesión coordina el intercambio de información entre las estaciones de trabajo. Por ejemplo, en el caso de que un sistema sea más lento que el otro o la transferencia de paquetes no sea ordenada.

Las funciones de la capa de sesión se pueden dividir en dos categorías:

- Servicio de Administración de Sesión. Determinación y cancelación de contrato entre dos entidades de la Capa de presentación.
- Servicio de Diálogo de Sesión. Control del intercambio de datos, entre esas dos entidades, comprendiendo sincronización, delimitación y recuperación de operaciones con los datos.

Este nivel incorpora información sobre el protocolo de comunicación que se utiliza y envía el mensaje a la capa de transporte.

Capa de transporte. La capa de transporte proporciona el control entre nodos usuarios a través de la red. Cada nodo de la red debe enviar el mensaje hacia un punto perteneciente a la ruta más conveniente para llegar al destino final. El nivel de transporte actúa como la interface entre los niveles superiores (de aplicación orientada) y los niveles inferiores (dependientes de la red). Esta capa releva a las sesiones de cualquier consideración de detalle referente a la forma en la cual se realiza la transferencia de datos. Los servicios proporcionados por este nivel se dividen en cinco clases, que van desde la clase 0, la cual proporciona solamente las funciones básicas necesarias para establecer la conexión y transferir datos; hasta la clase 4, la cual proporciona un control total de errores y procedimientos para el control del flujo.

Este nivel divide la información en segmentos más pequeños y le asigna una paridad a cada segmento para la comprobación de errores. Almacena una copia hasta que la estación receptora confirma la recepción. Envía los segmentos del mensaje al nivel de red.

Capa de red. La capa de red controla la operación interna de la red. Convierte en paquetes la información, se hace cargo de cómo se encaminan; del control de congestionamiento y de la contabilidad. El tamaño de cada paquete viene determinado por el método de acceso al cable o el sistema operativo.

Se incorporan cabeceras para almacenar el número total de paquetes y su secuencia. Envía los paquetes a la capa de enlace.

Capa de enlace. Asigna a cada paquete una paridad para comprobación de errores y añade ésta al bloque del paquete. Incorpora una cabecera de dirección al principio de cada paquete. Almacena una copia de cada paquete al nivel físico.

Capa física. El propósito de la capa física es llevar los datos de una computadora a otra. En específico, la capa física traduce bits de datos a un formato adecuado para su transmisión, o recibe una transmisión y la vuelve a traducir a bits. Esta capa ve a los datos como un flujo de bits. La capa física cubre cuatro áreas:

- Eléctrica: Los voltajes y corrientes necesarios.
- Mecánica: La forma y el tamaño de los conectores.
- Funcional. El significado de un *pin* conector y su voltaje correspondiente.
- De procedimiento: La secuencia de cambios funcionales que indican ocurrencias de evento.

Es importante aclarar que la capa física no es lo mismo que los medios físicos.

La capa física de la estación receptora recibe la información.

Lo más importante acerca del modelo de referencia OSI es que en realidad no dice cómo se construirán las capas. No le interesa lo que suceda dentro de ellas; sólo se concentra en la manera como las capas funcionan entre sí. Es decir, lo que le interesa al modelo es la comunicación (la estructura con la que se transmiten los mensajes) y no la instalación (lo que se usa para pasar dichos mensajes).

3.4. ESTÁNDARES DE COMUNICACIÓN

Los estándares se dividen en tres grupos principales:

- **Los estándares *de jure*** (de ley), están ratificados por algún organismo internacional, como la Organización de Estándares Internacionales (ISO) o el Comité Consultatif International Téléphonique et Télégraphique (CCITT), un organismo regional como la Asociación de Fabricantes Europeos de Computadoras (ECMA), o una organización como el Instituto de Ingenieros Electricistas y Electrónicos (IEEE) o el Instituto Americano de Estándares Nacionales (ANSI). Estos estándares son adoptados con frecuencia, por organismos gubernamentales y se convierten en estándares con los que hay que cumplir por ley cuando se utilizan equipos de comunicaciones o de computación. Las organizaciones como CCITT, ISO, ANSI, IEEE, ECMA, etc., desarrollan los estándares *de jure*.

- **Los estándares *de facto***: se convierten en tales por el hecho de tener un alto número de adherentes (no siempre voluntarios). Un ejemplo de esto es la Arquitectura de Sistema de Red (SNA) de IBM, que define la manera como las computadoras y aparatos como terminales e impresoras pueden comunicarse entre sí. Este estándar es aceptado simplemente debido a la base de usuarios instalada de IBM. Otros ejemplos son el DOS 3.1 y NetWare de Novell. Todos ellos se han convertido en estándares de facto, ya que ocupan posiciones muy importantes dentro del mercado. Algunos de estos tipos de estándares tienen factores cualitativos interesantes que obran como elementos de influencia en el mercado correspondiente, provocando el surgimiento de usuarios exigentes que obligan a los proveedores a ofrecer más beneficios en sus productos.

- **Los estándares *propietarios***: en ocasiones un producto no funciona tan rápidamente como se necesita o de una manera en particular si se emplean los estándares existentes, los estándares *propietarios* son creados por el fabricante. Un producto de cómputo basado en un estándar propietario puede ser un riesgo para quienes lo compran, ya que el fabricante puede ir a la quiebra o simplemente dejar de venderlo. Se considera que un estándar es abierto cuando éste puede ser utilizado por otros fabricantes, y si es adoptado por otros fabricantes, es posible que se convierta en un estándar *de facto*. Por ejemplo, los estándares de comunicación de IBM son propietarios por naturaleza, pero han sido adoptados y apoyados por tantos fabricantes que ahora son estándares *de facto*.

Tanto los estándares *de jure* como los *de facto* están disponibles al público, es decir, se pueden comprar copias de la especificación. Los estándares *propietarios* pertenecen a un solo individuo o compañía y son controlados por éste; los detalles no están disponibles al público.

CAPÍTULO 4

MODELO CLIENTE-SERVIDOR

4.1. SISTEMAS DISTRIBUIDOS

Para responder hoy en día a la naturaleza descentralizada de los negocios, muchas organizaciones están mejorando su eficiencia adoptando estructuras distribuidas. La tendencia actual indica que no importará la distancia y la gente podrá trabajar en casa, por ejemplo, un vendedor podría tomar un pedido, introducirlo en una PC portátil y servirse de una red inalámbrica de área amplia por medio de tecnología de telefonía celular y enviarlo a la oficina central.

4.1.1. Antecedentes de los Sistemas Distribuidos

Las primeras computadoras que se comercializaron se distinguieron por ser enormes, muy costosas e incapaces de soportar múltiples flujos de trabajo. Para las organizaciones que podían adquirirlas únicamente compraban una por departamento. Los procesadores de estos primeros sistemas no eran conectados por ningún tipo de enlace de comunicación. Cada computadora poseía su propia base de datos, frecuentemente varias computadoras contenían datos en común.

Por ejemplo, una computadora dedicada al almacén y una computadora dedicada a la contabilidad podían contener la misma información de clientes, la primera para las entregas y la segunda para las facturas. Cuando la dirección de un cliente cambiaba debía reflejarse en ambas bases de datos, es aquí donde surgen los problemas, los datos no se actualizaban al mismo tiempo en ambas bases de datos, provocando la inconsistencia de los datos. A este tipo de procesamiento se le llamó *descentralizado*.

Los sistemas progresaron gradualmente y los sistemas operativos fueron más "comprensivos". Para disminuir la redundancia de los datos y promover que los usuarios compartieran datos, se crearon los *sistemas centralizados*.

En los sistemas centralizados una gran computadora -un host o un servidor- puede atender a múltiples computadoras conectadas a ella (ver la figura 1.4). Cada usuario considera al sistema como si le perteneciera por completo, y puede ejecutar programas, manipular datos y hasta crear programas.

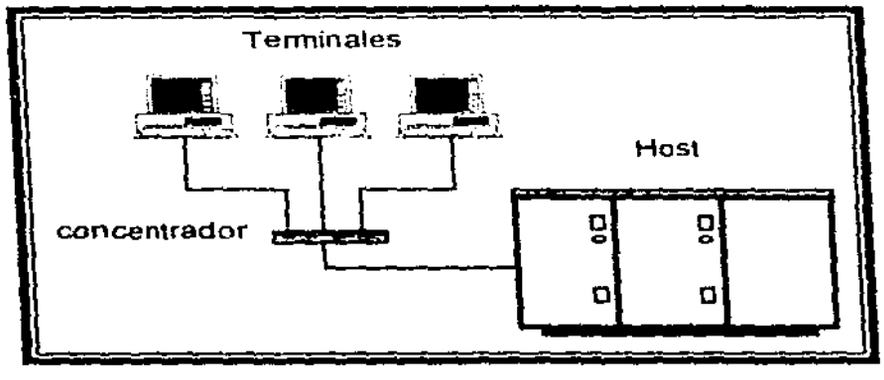


Figura 4.1. Sistema centralizado.

Dado que los sistemas centralizados deben dar apoyo a muchos usuarios a la vez, están diseñados y optimizados para proporcionar una respuesta rápida y un alto flujo de datos. Por otro lado, por su naturaleza centralizada estos sistemas permiten tener un algo grado de control de acceso y seguridad. La principal desventaja de la centralización es que si el sistema central falla, el sistema entero se estropeará. En este aspecto, los sistemas descentralizados son más confiables, ya que la falla de un nodo afecta sólo a una parte del procesamiento.

La tecnología que nos puede proporcionar algunos de los beneficios de los ambientes centralizados y descentralizados son las *redes de computadoras*.

Una red es un sistema de comunicacion en el que múltiples computadoras conectadas entre si transfieren y comparten datos y recursos (como almacenamiento en disco e impresoras). Los usuarios finales de estos sistemas cuentan con una computadora (estación de trabajo) capaz de ejecutar una gran variedad de aplicaciones por si sola. Pero también es capaz de conectarse a un host o a un servidor obteniendo mayor poder de procesamiento y la ventaja accesar datos remotos, así como la facilidad de ejecutar aplicaciones remotas. Es decir, las aplicaciones y los datos requeridos por los usuarios de las estaciones de trabajo a menudo están residentes en su propio disco local, y si estos recursos ocupan grandes cantidades de espacio en disco se encuentran en un servidor.

El concepto de *sistemas distribuidos* nace a finales de los años ochenta. Aunque existe una gran confusión en la literatura respecto a la diferencia de la *computación en red*[1] y un sistema distribuido, se puede decir que la distinción está en que en un sistema distribuido la existencia de múltiples computadoras conectadas en red es transparente (no visible) para el usuario.

[1] A la integración de los recursos disponibles en red, como servidores, PCs e incluso las macrocomputadoras, se le ha llamado "*computación en red*"

4.1.2. Caracteristicas de los Sistemas Distribuidos

El usuario de un sistema distribuido no se entera de la existencia de múltiples procesadores, él ve al conjunto como a un único procesador virtual. En efecto, un sistema distribuido es un caso especial de la computación en red, cuyo software tiene un alto grado de transparencia. La diferencia entre otros sistemas de computación en red y un sistema distribuido yace en el software utilizado, más que en el hardware.

En este tipo de sistemas los datos están distribuidos entre el grupo de computadoras conectadas en red. Cada computadora es autónoma, es decir, puede ejecutar aplicaciones locales, y participa en la ejecución de al menos una aplicación global. Las aplicaciones globales requieren accesar datos remotos utilizando subsistemas de comunicación.

La asignación de trabajos a procesadores, movimientos de archivos de donde están almacenados a donde son necesitados, y todas las demás funciones del sistema deben ser automáticas. Los programas no se ejecutan necesariamente en la estación de trabajo en la cual el comando fue dado.

En algunos sistemas de computación en red, el usuario debe registrar explícitamente su entrada a una máquina, pedir explícitamente los trabajos remotos, explícitamente mover archivos de un lugar a otro y generalmente manejar toda la administración de la red personalmente. Con un sistema distribuido ninguna de estas cosas tiene que hacerse explícitamente; todo esto lo hace automáticamente el sistema y el usuario no se da por enterado. Tanto como en un sistema distribuido como en los demás sistemas de computación en red, se necesitan mover archivos de un lado a otro. La diferencia yace en quién invoca el movimiento, el sistema o el usuario.

Un sistema distribuido es mas poderoso que uno convencional. Primero, este puede ser más confiable, porque cada función es replicada (duplicada o copiada) distintas veces. Cuando un procesador falla, otro puede tomar su trabajo. Cada archivo es almacenado en distintos discos, así si un disco falla no se destruye ninguna información. A esto se le conoce como *tolerancia a fallas*. Segundo, un sistema distribuido puede hacer mas trabajo en la misma cantidad de tiempo, porque la computación puede llevarse a cabo en paralelo.

Características deseables de un sistema distribuido

Los puntos listados en seguida distinguen y hacen más eficiente a un sistema distribuido.

- ***Múltiples elementos procesando.*** Estos pueden interactuar entre si y también correr independientemente. Por tal motivo, cada elemento o nodo debe contener procesador y memoria
- ***Interconexión de hardware.*** La comunicación tiene que darse entre los elementos procesando, así un sistema distribuido tiene interconexión de hardware (alambrica o inalámbrica).
- ***Los elementos procesando fallan independientemente.*** Un sistema distribuido puede no ser tolerante a fallas si cuando ocurre una falla todos los elementos se estropean simultáneamente. El sistema puede ser estructurado de tal forma que los elementos procesando fallen independientemente, en la práctica esto implica que las interconexiones estén estructuradas inteligentemente y que los datos almacenados en el nodo que falla estén disponibles en otro.
- ***Datos distribuidos*** en distintas computadoras conectadas en red y replicados para evitar pérdidas.

- ***La distribución de recursos es transparente para los usuarios del sistema***, es decir, los usuarios ejecutan operaciones locales y remotas sin hacer distinción entre ellas.

Desventajas de los Sistemas Distribuidos

Las principales dos desventajas de los sistemas distribuidos son las siguientes:

- ***Seguridad y acceso*** . Dado que los estos sistemas están distribuidos a todo lo largo y ancho de las organizaciones, son más difíciles de controlar que un sistema centralizado. Dado que las computadoras en la red son manejadas básicamente por sus usuarios, éstos generalmente están más ocupados en su trabajo que en manejar la red y suelen ser descuidados al manejar quién tiene derecho a sus recursos. Y en el caso de existir servidores de archivos, se necesitan administradores de sistemas, administradores de base de datos, administradores de grupos de trabajo y personal de soporte, para encargarse del acceso y seguridad de los recursos.
- ***Costo.*** El problema de administrar grandes grupos de usuarios de PCs interconectadas se vuelve serio, por lo tanto se requiere de personal dedicado a ésta tarea a parte del gran personal de mantenimiento.
- ***Rendimiento.*** Las transacciones entre múltiples nodos son lentas. Si una transacción debe accesar más de un nodo, el tiempo de respuesta aumenta. Es como si fuera a comprar una computadora, el tiempo de respuesta será rápido si todo el equipo lo compra con un proveedor que si cada componente lo adquiere en distintas tiendas. Por otra parte, las transacciones que modifican datos en múltiples nodos son lentas. Para evitar pérdidas generalmente se evitan modificaciones concurrentes. Probablemente los nodos que necesiten de los registros que se están modificando tendrán que esperar un largo tiempo .

4.2. ARCHIVOS DISTRIBUIDOS

Antes de comenzar con la discusión acerca de los archivos distribuidos, definamos lo que es un archivo.

> Un archivo es un conjunto de información almacenado como una unidad identificable con nombre propio en un medio de almacenamiento periférico (como un disco). Un archivo tiene ciertas propiedades o atributos.

Los archivos son algunos de los objetos distribuidos en una red. Frecuentemente las personas se refieren a la distribución de archivos como una *base de datos distribuida*. Sin embargo, la simple distribución de archivos no es suficiente para tener una base de datos distribuida. Para tener una verdadera base de datos distribuida, debe existir un sistema coordinador que maneje los datos contenidos en los archivos. Más tarde en este capítulo se detallarán los requerimientos de las bases de datos distribuidas y su significado.

En un sistema distribuido, los usuarios pueden localizar y utilizar archivos remotos como si fueran residentes. Los sistemas de software que proporcionan estos servicios son llamados **sistemas de archivos distribuidos** (DFS, por sus siglas en inglés). Los objetivos de un DFS son descritos a continuación.

- ***Acceso transparente a los archivos distribuidos.*** El acceso transparente significa que un usuario situado en un nodo de la red puede ser capaz de accesar archivos distribuidos en otros nodos como si fueran locales. El usuario utiliza los comandos del sistema local para accesar archivos remotos.

- ***Independencia de sistemas operativos.*** En la construcción de un sistema distribuido, un usuario debería ser capaz de configurar sistemas heterogéneos. Esto puede significar que diferentes sistemas operativos y sistemas de archivos sean implicados. No solamente los diseñadores deberían ser capaces de construir un sistema compuesto de diferente hardware y software, sino también estas diferencias deberían ser transparentes al usuario.

- ***Independencia de sistemas de archivos.*** Con la independencia de sistemas de archivos, los diferentes sistemas de este tipo, tales como DOS, UNIX y VMS, pueden ser utilizados en una red. La diferencia entre estos sistemas de archivos debería ser transparente. Por ejemplo, los comandos del sistema de archivos local deberían ser funcionales cuando accesarán archivos en un nodo remoto teniendo un sistema de archivo diferente.

- ***Independencia de arquitectura.*** El DFS debería permitir que cualquier configuración de la red -estrella, bus, árbol, anillo- fuera conectada. Ni la arquitectura ni el software deberían limitar la capacidad de archivos distribuidos.

- ***Resolución de contención.*** El DFS debe proporcionar un mecanismo apto para prevenir la alteración del contenido de los archivos. Esto puede resultar cuando dos o más usuarios intentan accesar y modificar el mismo archivo.

- ***Seguridad.*** Un DFS debe proporcionar seguridad como requisito indispensable. Los archivos deberían asegurarse solamente para acceso local o acceso remoto. Cuando el acceso remoto a un archivo es permitido, el DFS debe se capaz de conceder o negar peticiones basadas en contraseñas. Inherente a este requerimiento está la capacidad para proporcionar identificadores a los usuarios.

- ***Información en directorio de archivos.*** El DFS es responsable de hacer transparentes las peticiones del usuario. Esto significa que debe mantener un directorio de archivos remotos y sus localizaciones. Cuando un usuario pide el acceso a un archivo, el directorio es consultado para buscar el nodo(s) que alberga al archivo.

- ***Independencia de localización.*** Un archivo puede ser localizado en cualquier nodo de la red. También, un archivo debe poder transladarse de un nodo a otro sin interrumpir aplicaciones.

Existen diferentes implementaciones de DFS. Por ejemplo, en los sistemas UNIX, existe una extensión de los servicios de archivo del sistema operativo llamado NFS (Network File System). Desarrollado por Sun Microsystems y adoptado por otros fabricantes de UNIX. Es un sistema bastante complejo.

4.2.1 Colocación de Archivos

Un factor critico que afecta la ejecución en un sistema distribuido es la colación de los archivos.

Archivos centralizados.

Todos los archivos o la mayoría están residentes en uno o más nodos, esta situación elimina muchos problemas como la actualización de archivos y seguridad. La desventaja de tal implementación es la posibilidad de los cuellos de botella, es decir, el congestionamiento que los usuarios causan al querer accesar el mismo archivo. Si los datos están centralizados en un único nodo y este falla, todos los demás se verán afectados al no poder accesar a los archivos.

Los archivos pueden almacenarse en una computadora con un sistema operativo especial que permite a PCs autorizadas tener acceso a estos archivos. Por su función a esta computadora se le llama *servidor de archivos* (figura 4.2). Por lo tanto, un *servidor de archivos* es una computadora que, a traves de algún tipo de tecnologia de computación, se conecta a un grupo de PCs *clientes*, permitiéndoles tener acceso a archivos, en lugar de sólo a sectores de disco (a diferencia de su antecesor: servidor de disco). Esta perspectiva lógica del servidor ha facilitado la protección contra daños al sistema, permite una manera más avanzada de controlar el acceso de los usuarios y facilita la creación de software multiusuario.

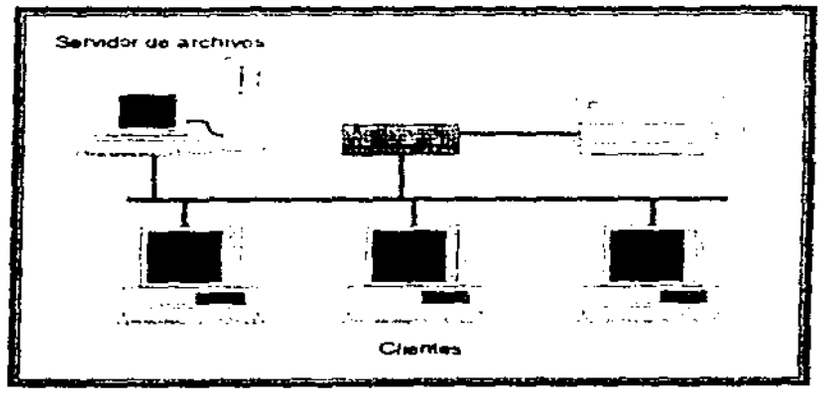


Figura 4.2. Servidor de Archivos en una topologia de bus.

Los actuales lideres del mercado en sistemas basados en servidor de archivos son:

- ✧ Novell (con NetWare)
- ✧ Banyan (con VINES)
- ✧ Microsoft (con Windows NT Advanced Server)
- ✧ IBM (con NetWare y LAN Server)
- ✧ Apple (con AppleShare)

El mayor inconveniente de los sistemas de servidor es que requieren una computadora dedicada especialmente a ejecutar el sistema operativo de red. A parte del sistema operativo, necesitan tambien tarjetas de interfaz para red y cables. Este servidor es con frecuencia la PC más poderosa en la compañía, con la mayor capacidad de almacenamiento en disco, la memoria más grande y el procesador más rápido. Por supuesto, esto está bien si se tiene un gran número de PCs, pero si sólo se tienen unas cuantas no es muy redituable. Aquí es donde entran en acción las *redes punto a punto.*

Archivos compartidos.

Con las redes punto a punto, el dueño de una PC puede configurar su sistema de manera que los usuarios en otras PCs tengan acceso a directorios especificos en su sistema, permitiendo que usen los archivos que contienen, también puede compartir su impresora. La idea de las redes punto a punto es que cada PC de la red puede ser a la vez cliente y servidor. Las redes punto a punto generalmente no son tan rápidas como los sistemas basados en servidores de archivos y dado que cada PC se encuentra bajo el control del usuario, éste puede ser descuidado al manejar el acceso. Las redes punto a punto permiten también que las PCs sean sólo clientes o servidores. (Esto los hace funcionalmente equivalentes a los sistemas de servidor de archivos). Cuando una PC de una red punto a punto funciona como servidor y puede ejecutar aplicaciones (procesadores de texto, hojas de cálculo, etc.) al mismo tiempo, se le llama *servidor no dedicado.*

La naturaleza descentralizada de las LANs de punto a punto facilita reorganizarlas conforme la situación lo demande.

La siguiente es una lista de los líderes actuales en el mercado de las redes de punto a punto:

- Apple (con el Sistema 7.0 de Macintosh)
- Artisoft (con LANtastic)
- Novell (con Personal NetWare)
- Microsoft (con Windows para Grupos de Trabajo)
- DCA (con 10Net)
- Sitka (con 10Net)

Archivos replicados.

Los archivos que son comunes para las aplicaciones en distintos nodos, pueden ser replicados (copiados o duplicados) en varios nodos para reducir el acceso a un sólo nodo. Pero si los archivos replicados pueden modificarse por los usuarios, mantener la consistencia de los datos en tiempo real es sumamente complicado, como se verá en el tema siguiente.

Archivos fragmentados.

Se dice que un archivo está fragmentado cuando los datos que lo forman están distribuidos en varios nodos de la red. Cuando el usuario accesa un archivo de este tipo, lo observa como un archivo lógico, cuando en realidad es la composición de varios archivos.

4.3. BASES DE DATOS DISTRIBUIDAS

A cualquier colección de datos se le puede llamar *base de datos*. En el campo de la computación, este término designa a un archivo o serie de archivos que almacenan datos en un formato recuperable para su posterior consulta o modificación. Se dice también que una base de datos es un conjunto de registros relacionados entre sí, donde un registro guarda datos acerca de un objeto o un tema en particular. Un registro esta formado por campos. Por ejemplo los campos. nombre, direccion, número telefónico, etc., constituirían un registro sobre una persona en especial.

Existen bases de datos centralizadas y bases de datos distribuidas. Una *base de datos centralizada* es la que se encuentra en un único nodo de una red de computadoras. En este tipo de base de datos el control es centralizado, es decir, asegurado por el administrador de la base de datos, así como la seguridad del sistema y privacidad. Para ahorrar espacio en disco se evitan las múltiples copias de la información, por lo tanto, no hay redundancia.

Una *base de datos distribuida* es la que tiene sus elementos de datos almacenados en más de un nodo de una red de computadoras, cada uno puede participar en la administración global y tiene autonomía local. En cada uno de estos nodos existe un sistema de administración de bases de datos distribuidas (SABDD) que es responsable de:

1. Administrar los accesos a la información local.
2. Comunicarse con los SABDD remotos para controlar, de una forma cooperativa, la ejecución de las transacciones que acceden a datos almacenados en más de un nodo.

Una transacción es una operación que ejecuta acciones de lectura, actualización y los procesos necesarios para ejecutar la tarea deseada por el usuario. Por ejemplo, la transacción:

"Aumento del 7% en los salarios de los profesores que trabajan en el laboratorio de Medición e Instrumentación de la Facultad de Ingeniería y trabajan tiempo completo".

4.3.1. Tipos de bases de datos distribuidas

Las *bases distribuidas* se pueden catalogar en dos tipos:

1. ***Multipartes.*** Los archivos se distribuyen en una o más áreas de disco en el mismo servidor, o en dos o más servidores. Estos archivos forman conceptualmente una sola base de datos. Para el usuario no es evidente que una consulta de la base de datos se dirige a diferentes secciones en diferentes servidores según se requiera. Como ejemplos de este tipo de sistema tenemos a los servicios StreetTalk de VINES de Banyan y los servicios de Directorio NetWare (NDS) de Novell. Bajo NDS, cada servidor posee una subsección o *partición* de toda la base de datos.

2. ***Replicadas.*** Cada base de datos es una copia de la base de datos maestra (hay una base de datos maestra y todas las demás son clones). Su manejo es bastante complejo, para lo cual es necesario contar con reglas que definan cómo manejar cambios indicados desde múltiples locaciones. Notes de Lotus es básicamente un sistema de base de datos que emplea este modelo, aunque se presenta como un producto groupware (software de grupo). NDS de Novell también puede ser una base de datos distribuida de tipo replicada. Cada partición tiene una versión maestra que se copia en los otros servidores según se requiera.

Si la base de datos es distribuida y no centralizada, las tareas individuales pueden acceder a sus datos de forma más rápida, fiable y con menos cuellos de botella. Una de las razones es que frecuentemente se procesan en paralelo. Ademas, un fallo de una base de datos raramente causará el fallo del sistema entero si se construye con redundancia.

Aunque la redundancia de los datos mejora el tiempo de respuesta, suministrando múltiples fuentes de información, los requisitos de replicación para los archivos distribuidos también producen problemas logísticos y de sobrecarga, puesto que todas las copias de los archivos deben ser actualizadas. El uso de una base de datos distribuida introduce el problema de *control de concurrencia*. El control de concurrencia implica la sincronización de las bases de datos de forma que todas las copias tengan la misma y correcta información disponible para los accesos.

El método convencional para el control de concurrencia se basa en lo que se conoce como *bloqueo y estampado* de tiempo. En intervalos regulares, se inician los siguientes tareas:

1. La base de datos es "bloqueada", de forma que se asegure el control de concurrencia; no se permite E/S.
2. Se realiza la actualización requerida.
3. La base de datos es desbloqueada.
4. Los archivos son validados para asegurar que todas las actualizaciones se han hecho correctamente.
5. Se reconoce que la actualización se ha terminado.

Todas las tareas de bloqueo están monitorizadas por un reloj maestro (es decir, estampados de tiempo).

Se han desarrollado algunas técnicas para aumentar la velocidad de actualización y resolver el problema de concurrencia. Una de éstas, llamada *protocolo del escritor exclusivo*, mantiene la consistencia de los archivos replicados permitiendo que sólo una tarea de escritura actualice un archivo en exclusiva. Por lo tanto, elimina la gran sobrecarga de los procedimientos de bloqueo y estampado de tiempo.

Actualmente existen herramientas poderosas para que la replicación de archivos sea mas sencilla y confiable SQLBase® y Centura Ranger® de Gupta® introducen un nuevo modelo de replicación store-and-forward (almacenar-y-enviar) especificamente diseñado para datos descentralizados. La solucion store-and-forward intenta sincronizar la base de datos objetivo con la de origen. La replicación de SQLBase soporta bases de datos heterogéneas, asegura la integridad de los datos y la consistencia en las transacciones. Centura Ranger consiste de un núcleo local de base de datos (el DBMS SQLBase) con tecnología sofisticada de replicación. Con el poder de replicación, Centura Ranger proporciona una solución para usuarios "movibles " u ocasionalmente conectados.

Por otro lado, Replication Server® de Sybase®, proporciona un acceso flexible para distribuir información, sincronizando réplicas de datos en plataformas heterogéneas a lo largo de una red cliente-servidor. Replication Server resuelve 6 requerimientos indispensables.

1) Alta disponibilidad de datos, el sistema no se estropeará por falta de datos, ya que hay réplicas necesarias.
2) Entrega de información consistente, se mantiene la integridad de los datos.

3) Alto desempeño, utiliza la red eficientemente a través de rutas inteligentes.
4) Admón. centralizada, el administrador puede monitorear y manejar cada componente del sistema de replicacion.
5) Acceso a datos heterogeneos
6) Autonomia local, cada sitio puede participar en el ambiente de replicacion.

4.3.2. Reglas de implementación de las bases de datos distribuidas

Ya se han dado los objetivo de los sistemas de archivos distribuidos, un grupo similar de objetivos o reglas se ha establecido para las bases de datos distribuidas. Estas reglas se explican a continuación. Note que en algunas ocasiones las reglas son comparables a los objetivos de los sistemas de archivos distribuidos y que estas reglas aumentan las capacidades de los sistemas de archivos remotos. Actualmente, no hay un sistema que se adhiera a todas estas reglas, pero aquellos que quieran implementar una base de datos distribuida deberán proponerse cumplir con la mayoria de ellas. Un ambiente que comprenda las 12 reglas listadas requerirá una inversión considerable.

Regla 1. Autonomia local. Los usuarios de un nodo determinado son responsables de la operación del sistema y administración de los datos de ese nodo. Un nodo local tiene cierta cantidad de independencia respecto a esas operaciones locales. Sin embargo, esta independencia no es ilimitada. Como un individuo en una sociedad libre tiene independencia individual, pero la independencia se extiende solamente donde no afecte a otros miembros de la sociedad. Asi, un nodo puede operar si sus acciones no son perjudiciales para la operación del sistema distribuido. "Tanto entre las naciones como en los individuos (y las bases de datos distribuidas) el respeto al derecho ajeno es la paz".

Regla 2. No confiar en un sitio central. Significa que todos los nodos en el sistema distribuido deberan ser considerados como nodos iguales. Además, no habrá un nodo del cual otros nodos deban depender, tal como un único nodo que contiene un directorio o diccionario de datos centralizado.

Regla 3. Operaciones continuas. La conexión, desconexión o falta de nodos en la red no debe afectar la disponibilidad de otros nodos. Naturalmente, la falla de un nodo probablemente romperá el acceso a los usuarios para localizar ese nodo, sin embargo, los usuarios de otros nodos pueden continuar el uso de la base de datos distribuida, y su ruptura limitará el acceso a los datos almacenados sólo en los nodos faltantes.

Regla 4. Independencia de localización. Los datos pueden ser colocados en cualquier lugar de la red y que su localización es transparente a las necesidades de acceso a él. Los datos pueden ser trasladados de un nodo a otro, sin importar la distancia.

Regla 5. Independencia de fragmentación. Los datos que los usuarios observan como un archivo lógico, pueden estar distribuidos transparentemente en múltiples nodos. Por ejemplo en una empresa, la información de empleados y su salario puede estar distribuida en nodos de distintas regiones. Físicamente estos archivos están separados, pero lógicamente su combinación puede dar un sólo archivo de personal corporativo y nómina. El administrador de personal debe ser capaz de hacer averiguaciones exhaustivas de todos los empleados, como el salario promedio y recibir la respuesta consolidada de todos los fragmentos. Además, la averiguación debe hacerse de la misma forma que si la tabla no estuviese fragmentada. El sistema de administración de bases de datos distribuidas es el responsable de formar un sólo archivo con varios fragmentos.

Regla 6. Independencia de réplicas. A las copias de un archivo en distintos nodos se les llaman replicas. El archivo puede ser duplicado en dos o más nodos y esta réplica es transparente para los usuarios y aplicaciones. La réplica es preferible para los archivos que necesitan ser accedidos por varios nodos, tal como un directorio de la red. Las réplicas pueden mejorar la ejecución y disponibilidad. El sistema administrador de bases de datos distribuidas es el responsable de actualizar las réplicas y mantener la consistencia de los archivos.

Regla 7. Procesamiento de preguntas distribuidas. Significa que un usuario puede formular una pregunta que involucre datos de otros nodos. El sistema administrador de bases de datos distribuidas debe determinar y llevar a cabo la estrategia de acceso más optima a los nodos que trabajarán cooperativamente con una porción de la pregunta.

Regla 8. Administración de transacción distribuida. Las transacciones que se extiendan a varios nodos deberán ser permitidas. Esto quiere decir que una transacción comenzada sobre un nodo puede modificar registros en otros nodos. Si se quiere modificar un archivo y algo falla antes de actualizar registros, los datos se pierden, pero la base de datos en si queda intacta. Pero ¿qué tal si ya se habían escrito los primeros 1,024 bytes y no se puede seguir adelante por fallas en la red? El archivo queda parcialmente modificado, y los datos de ese registro quedan alterados. El sistema administrador de bases de datos de cada nodo debe tomar en cuenta los problemas implicados y tener recuperación de fallas.

Regla 9. Independencia de Hardware. La red distribuida puede consistir de hardware de diferentes fabricantes.

Regla 10. Independencia de Sistemas Operativos. Cuando el hardware de diferentes fabricantes suministra los nodos de la red, es casi seguro que diferentes sistemas operativos sean utilizados

Regla 11. Independencia de redes. Otra consecuencia de la regla 9. puede implicar se usen diferentes arquitecturas de red, software y protocolos. El problema se soluciona por medio de un enfoque estructurado de los estandares El enfoque estructurado que más se utiliza para los sistemas de comunicaciones es el Modelo de Referencia OSI. si los fabricantes diseñan sus sistemas de red de acuerdo a este modelo, la interconexión es más facil. La independencia de red implica que las múltiples clases de software de red pueden utilizarse juntas para conectar a los nodos, y que las habilidades de las bases de datos distribuidas no sean afectadas.

Regla 12. Independencia de SABDD. Una variedad de sistemas de administración de bases de datos distribuidas (SABDD) pueden utilizarse en un sistema distribuido. Cada SABDD tiene diferente método de acceso a datos y diferente lenguaje de manipulación, tiene diferentes mecanismos de recuperación y guarda los datos en diferente formato. Los SABDD deben hacer transparentes estas diferencias para los usuarios y las aplicaciones. Además, un usuario será capaz de accesar datos administrados por cada una de los SABDD sin tener que aprender diferentes lenguajes de acceso a datos. Especificamente, los usuarios serán capaces de accesar datos distribuidos utilizando la misma interfaz. Esta regla implica que los sistemas de bases de datos serán coordinados y acoplarán las diferencias de lenguaje. La implementación de esta regla es muy compleja.

Actualmente, la mejor forma de implementar una base de datos distribuida es utilizando hardware y software de un sólo fabricante, que tenga un sistema de base de datos que soporte las habilidades distribuidas.

4.4. MODELO CLIENTE - SERVIDOR

Para comprender la estructura del modelo Cliente-Servidor, es necesario hablar de tres tipos de productos de mensajería electrónica. La mensajería electrónica es el conjunto de tecnologías por medio de las cuales es posible enviar mensajes de un usuario a otro, para su posterior lectura. La mensajería electrónica puede llevarse entre dos personas; entre dos programas, entre un programa y una persona o entre una persona y un programa.

El mercado de la mensajería LAN se divide en productos (Véase la figura 4.3):

- **De aplicación *frontal (Front-End)***. Es la parte de aplicación del software que se encarga de manejar la manera como el usuario ve los servicios que proporciona la aplicación posterior.

- **De aplicación *posterior (Back-End)***. Este software proporciona los servicios que son presentados por la aplicación posterior. Se encuentra en otra computadora, aunque también puede estar en un módulo separado de la misma PC.

- ***Middleware.*** Es un nuevo tipo de software que se utiliza para conectar una aplicación frontal con una aplicación posterior. Se trata de un componente diseñado para ocultar la red subyacente y hacer que el establecimiento, control y administración de las conexiones de red sean invisibles para el usuario.

Los productos de aplicación frontal dependen de los servicios de aplicación posterior.

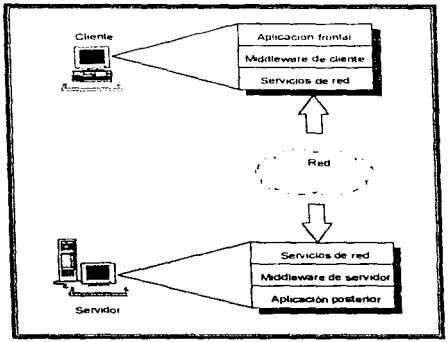


Figura 4.3. Aplicación frontal, aplicación posterior y middleware.

Las primeras aplicaciones para red eran deficientes, debido a que si ofrecían una base de datos para usuarios múltiples, por ejemplo, tenían la aplicación frontal del usuario y el mecanismo de la base de datos (el código de programa que accede a los archivos de la base de datos) en la misma PC. En el servidor sólo se encontraba la base de datos.

A este tipo de aplicaciones se les llamó *aplicaciones basadas en el cliente*, debido a que la PC hacía todo el trabajo de mensajería de datos (lectura, búsqueda de registros en los datos leídos, etc.). El servidor era solamente el depósito de datos. Dicho de otra forma, los componentes de aplicación frontal y posterior se encontraban en la misma PC, y los datos en el servidor o en una unidad local.

Si tomamos como ejemplo una biblioteca y usted es un usuario que desea buscar en una enciclopedia todo lo referente a computadoras, tendrá que tomar la enciclopedia, buscar la información y extraer los datos que le interesen. Usted hace todo el trabajo; la tarea del bibliotecario se reduce a encontrar la enciclopedia. En este ejemplo, usted es la computadora cliente en la red, el bibliotecario es el servidor de archivos y la enciclopedia es el archivo de la base de datos que hay que buscar.

Actualmente han surgido nuevos sistemas avanzados en donde la aplicación frontal se encuentra en el cliente, la aplicación posterior en el servidor y ambos se encuentran unidos por alguna clase de middleware. El componente servidor puede proporcionar servicios a muchos clientes de manera simultánea. A estos sistemas se les ha llamado sistemas *cliente-servidor*. (Véase la figura 4.4)

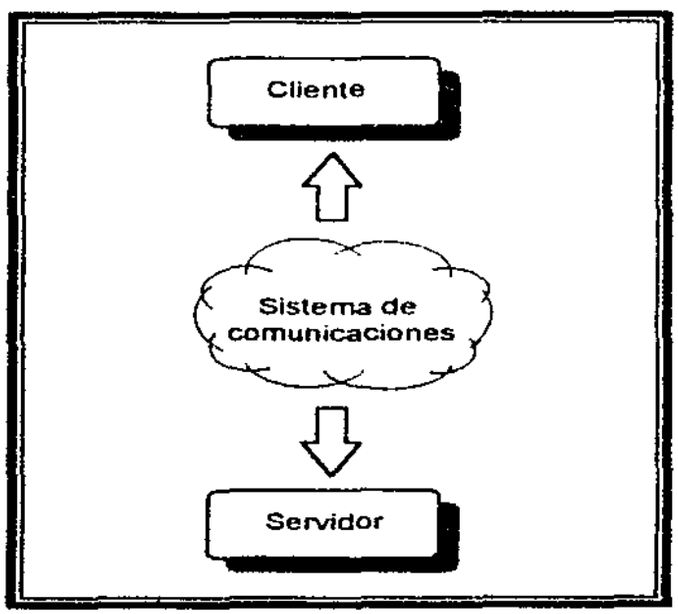


Figura 4.4. Modelo Cliente-Servidor.

Volviendo al ejemplo de la biblioteca, si usted puede pedirle al bibliotecario (servidor) que encuentre los datos por usted, sólo necesita esperar resultados. El bibliotecario hace toda la búsqueda. Como el bibliotecario es más efectivo que usted para buscar información, el trabajo puede hacerse más rápido. Esto es así generalmente porque el servidor está basado en una PC más poderosa que la de usted.

Las bases de datos no son las únicas aplicaciones que pueden ejecutarse en los sistemas de cliente-servidor. Por mencionar algunas tenemos a los servidores de correo electrónico, sistemas de creación de imágenes por computadora y de supervisión de servicios de red.

Ventajas de los sistemas cliente-servidor

- ***Seguridad.*** La seguridad es mayor porque los datos de la base de datos del servidor son accedidos de manera indirecta. Los usuarios no pueden ver en realidad los archivos de datos a menos que se les dé acceso explícito.
- ***Rendimiento.*** El rendimiento puede incrementar, ya que un servidor de aplicación posterior bien diseñado puede proporcionar una mejor coordinación de usuarios múltiples. En el caso de los servidores de base de datos, las PCs cliente no tienen que leer todos los datos de la base de datos para encontrar lo que requieren; pueden enviar solicitudes al servidor, y el servidor entrega solamente lo que necesitan.
- ***Efectividad.*** La efectividad es mucho mayor en relación con el costo. Los clientes sólo necesitan una máquina con el poder suficiente para ejecutar la aplicación frontal. Cuando se requiere de un alto rendimiento, el servidor de base de datos puede implementarse en una PC muy poderosa y proporcionalmente costosa.

Desventajas de los sistemas cliente-servidor

- ***Complejidad.*** Los sistemas cliente-servidor generalmente no son fáciles de implementar, configurar ni manejar.
- ***Requerimientos y costo*** . Las aplicaciones de servidor tienden a ser grandes y complejas, por lo cual necesitan mucha memoria y rapidez de procesamiento. El servidor necesita ser una PC muy poderosa (costosa). Y si el número de usuarios es grande, el servidor deberá ser una máquina dedicada especialmente a este servicio

Hoy por hoy la gran mayoria de las aplicaciones en redes de computadoras se diseñan siguiendo el modelo *cliente/servidor* En resumen, bajo este modelo, un proceso llamado *servidor* es ejecutado en una computadora de la red donde se dedica a esperar solicitudes de servicio, las atiende (ofrece el servicio) y regresa a esperar más peticiones. Por otro lado, los procesos *cliente* en otras computadoras de la red, contactan al servidor cuando requieren del servicio de red correspondiente.

Por lo general, el servidor es un proceso que se ejecuta en una máquina multitareas, por ello al cliente no le basta con identificar en que computadora se encuentra el servidor, también debe ser capaz de identificar al proceso servidor en esa computadora. Las herramientas más populares para la programación en redes que resuelven ese problema son analizadas en el capitulo 6.

CAPÍTULO 5

DESCRIPCIÓN FUNCIONAL DEL SISTEMA INGRESO

5.1. ANÁLISIS FUNCIONAL

El análisis funcional del proyecto INGRESO establece el objetivo y las funciones primordiales del sistema, sin considerar cómo se llegará a esos fines. A partir de esto, se evalúa el flujo de la información (Entradas/Salidas) del sistema.

Entendiendo el comportamiento del sistema en el contexto de los sucesos que lo afectan, se establecen las características de la interfaz y las restricciones de recursos y tiempo. Este análisis será la base para el trabajo posterior de ingeniería.

5.1.1. Objetivo y Funciones primordiales del Proyecto INGRESO

En la primera parte de este trabajo se describieron los problemas que se generaban con el tradicional sistema de instalación y desinstalación de software tipo monousuario, dentro de la Coordinación de Informática del IMP. Para terminar con estos problemas se propusieron soluciones alternativas. La solución elegida consistió en automatizar el proceso de instalado y desinstalado de paquetes, bajo un esquema cliente-servidor (estudiado en el capítulo 4).

Al proyecto encargado de llevar a cabo esta solución se le ha llamado proyecto *INGRESO* (INstalación y Gestión REmota de SOftware).

Objetivo

El proyecto INGRESO tiene como objetivo principal diseñar e implementar un servidor de instalaciones a distancia de paquetes (software).

Funciones Primordiales

El servidor de instalaciones a distancia de software debe ser capaz de:

- Llevar a cabo las peticiones de instalaciones de software de los clientes, administrando el número de licencias contratado para cada paquete.
- Impedir que un usuario monopolice la licencia de un paquete sin hacer uso de él. Para llevar a cabo esto, se debe de detectar cuando un paquete instalado por el servidor no ha sido utilizado después de un tiempo x, y entonces desinstalar el paquete.
- Los objetivos anteriores implican otra tarea: tener un control claro de las licencias otorgadas. Es decir, poder identificar a cada usuario que retiene una licencia, así como el tiempo que tiene con ella.
- Detectar al cliente que desinstale por si mismo el paquete cuando ya no lo requiera o desee liberar espacio en disco. En este caso, entrará a la disposición de los demás clientes la licencia liberada.
- Autentificar a los usuarios. Únicamente usuarios autorizados podrán accesar al servidor.

- Encriptar los datos transferidos entre el cliente y el servidor, como protección contra intromisiones no autorizadas.
- Identificar cuando una licencia ha sido extraviada (por ejemplo, la máquina fue separada de la red, fue reformateada por virus, etc.). Podemos considerar este caso como pérdida involuntaria de licencia.
- Actualización de software previamente instalado.

5.1.2. Funcionamiento general del sistema

Como se estableció en un principio, el proyecto INGRESO debe ser implementado bajo un modelo *cliente/servidor*.

Bajo este modelo, un proceso *back-end* que llamaré *servidor INGRESO* será ejecutado en una computadora de la red donde se dedicará a esperar solicitudes de instalación, las atenderá y regresa a esperar más peticiones.

Por otro lado, los usuarios podrán contactar al *servidor INGRESO* cuando requieran su servicio a través de procesos *front-end* o *Cliente* cargados en otras computadoras de la red.

Podemos hablar entonces de *transacciones*, donde cualquier petición al servidor es una transacción.

5.1.3. Entradas y Salidas del sistema en general

Entradas del sistema.

El Instituto Mexicano del Petróleo organiza la actividades de su personal por medio de *proyectos*[1] Cada persona es asignada a un *número de proyecto*, que corresponde a una actividad especifica

El usuario del proceso *cliente* introducirá como entradas una clave de acceso y el número de proyecto al cual esta inscrito, para poder efectuar una petición de instalación.

Tanto la clave de acceso, como la clave de proyecto minimizan el número de acciones de entrada de datos que debe realizar el usuario, ya que con ellas el *servidor INGRESO* será capaz de obtener más datos. Éste contendrá bases de datos que complementen la información dada por el usuario.

Con la clave de usuario, el Servidor podrá saber el nombre del usuario, clave de empleado, cargo (becario/empleado) y su priondad ante los demás. Por otro lado, con el número de proyecto, obtendrá el nombre del proyecto, el nombre del responsable y los usuarios asignados. La administración de éstas bases de datos estará a cargo del Administrador de las bases de datos.

Es indispensable que el servidor almacene otra base de datos: la del software disponible. Esta información especificará el número de licencias de cada paquete.

[1] Los proyectos pueden ser facturables o no. Un proyecto es facturable cuando los gastos que genere se cargan a terceros

Si los datos dados por el usuario son válidos, podrá elegir dentro de una lista el nombre del paquete (software) que desea instalar

El sistema aparte de datos también tiene como entradas *sucesos*[2], ya que el sistema debe tener un proceso para monitorear a las estaciones de trabajo para averiguar cuando un paquete no ha sido utilizado después de un tiempo determinado, o cuando ha sido desinstalado por el usuario. También la pérdida involuntaria de un paquete debe ser considerada como un suceso de entrada, este caso puede darse cuando la máquina Cliente es separada de la red o reformateada por virus.

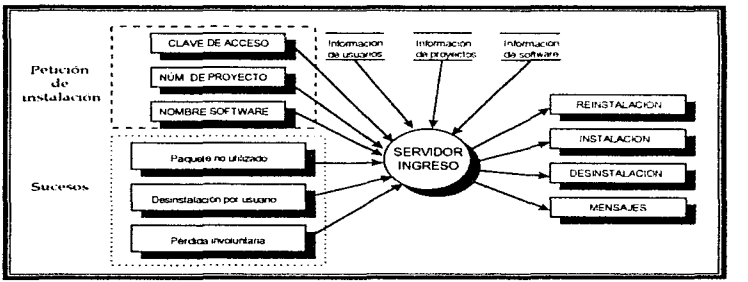


Figura 5.1. Diagrama de bloques de Entradas/Salidas.

[2] Un suceso representa algún aspecto de control del sistema y realmente no es más que un dato booleano. Los datos booleanos pueden ser: si o no, verdadero o falso, existencia o no existencia.

Salidas del sistema

El *servidor INGRESO* proporcionará los servicios de instalado ó desinstalado de paquetes. Los mensajes de error u operación satisfactoria forman también parte de las salidas del sistema. Otra salida que puede ser considerada es la reinstalación, y seria el resultado de modificar los parámetros actuales del paquete instalado. Las entradas y salidas al sistema se muestran en la figura 5.1.

5.1.4. Planeación de la Interface

En la Coordinación Informática del IMP el software de soporte más utilizado es Windows for Group® y Windows 95®, por lo cual la interfaz del sistema del servidor INGRESO será adaptada al ambiente Windows, pretendiendo ser gráfica y amigable.

Los usuarios del servidor de aplicaciones deberán introducir su clave de acceso y el número del proyecto dentro de una caja de diálogo. Por seguridad el sistema no desplegará los caracteres en pantalla.

El usuario podrá especificar el paquete que desea instalar, utilizando el ratón para seleccionarlo de entre un conjunto predefinido, o utilizando la "iluminación deslizante". Solamente serán desplegados aquellos paquetes que tengan aún licencias disponibles.

Los mensajes de error serán significativos, es decir, describirán el problema en un lenguaje que comprenda el usuario y proporcionarán una información constructiva para poder resolver el problema.

Las pantallas para administrar las bases de datos del servidor serán sencillas y de fácil uso. A través de menús descendentes se desplegarán cajas de diálogo. las cuales accesarán las bases de datos y por medio botones efectuarán operaciones específicas.

5.1.5. Recursos para el desarrollo del proyecto

Los recursos que el Instituto Mexicano del Petróleo proporciona para el desarrollo del proyecto INGRESO se describen a continuación.

Recursos de Hardware

- 2 computadoras personales 486 o superior, con 8 Mb de memoria principal como mínimo y con capacidad gráfica
- 2 tarjetas de red Ethernet (10 Mb)
- 1 impresora láser (lenguaje PCL)
- Red local Ethernet (10 Mb, cable coaxial o par trenzado)

Recursos de Software

- Sistema operativo de red (Novell 3.12 o superior)
- Windows for group 3.11
- Windows 95
- Sistema operativo DOS 6.0 o superior
- Software necesario para el desarrollo de aplicaciones NetWare y Windows

El lenguaje de programación orientado a objetos *Borland C++® para Windows* es la herramienta de programacion que he seleccionado para el desarrollo del proyecto, algunas de las razones se describen a continuación.

El proyecto INGRESO trabajará bajo un ambiente Windows, Borland C++® para Windows proporciona una biblioteca de clases llamada *ObjectWindows* (OWL) que simplifica enormemente la programación para Windows.

Una de sus principales ventajas es que oculta de manera efectiva muchos de los detalles de programación para Windows, con lo cual se puede concentrar realmente el esfuerzo en la creación de programas para Windows, en vez de tener que detenerse en los numerosos e intrincados detalles que usualmente están relacionados con esta labor.

Borland C++® para Windows puede compilar programas creados en lenguaje C. Aunque muchas características del lenguaje C son consideradas como obsoletas[3], estos programas pueden adaptarse a la programación orientada a objetos. Esto es importante, ya que muchas herramientas que serán útiles para el desarrollo del proyecto INGRESO están implementadas en el lenguaje de programación C.

Por ejemplo, el sistema de entrada/salida básico en red (Network Basic Input/Output System: NetBIOS), del cual se hablará con detalle posteriormente, cuenta con mayor referencia para su implementación haciendo uso del lenguaje de programación C. NetBIOS permite establecer una sesión entre dos estaciones de trabajo para llevar a cabo la transmisión de los datos. Para establecer la comunicación entre el servidor de aplicaciones y las estaciones de trabajo se hará uso del NetBIOS.

[3] Por ejemplo, el estilo de definición de una función en C como la siguiente es ilegal en los compiladores C++: int func(p1, p2) int p1, p2; { /* ... */ }

Por otra parte, para promover el desarrollo de software para un determinado sistema, debe disponerse de interfaces para programación de aplicaciones (API, Applications Programing Interfaces). Netware y Windows ofrece un conjunto de interfaces bien definido que pueden ser usados por equipos de desarrollo de software para adaptar o desarrollar aplicaciones que funcionen en el entorno distribuido de una red.

Por una parte, Windows® proporciona un gran número de funciones API indispensables para el desarrollo del proyecto INGRESO. Entre las más importantes, se encuentran las funciones para manejar las conexiones en la red.

Estas funciones permiten redireccionar dispositivos remotos (discos duros e impresoras) como locales. Todas las funciones API de Windows® pueden ser compilables con Borland C++ ® para Windows.

Por otra parte, Netware®[4] dispone de las siguientes interfaces:

- Un módulo de interfaz con una biblioteca en C que se conecta con el sistema operativo y ofrece una interfaz compatible con el ANSI C. Los desarrolladores pueden usar llamadas a funciones normales para acceder a los recursos de Netware.
- Un conjunto extendido de llamadas a funciones que pone las prestaciones específicas de Netware al alcance de aplicaciones basadas en servidor. Estas funciones extendidas se utilizan para acceder a las prestaciones de seguridad, facturación, gestión de colas y control de transacciones.

[4] El sistema operativo Netware fue desarrollado en el lenguaje de programación C.

- Una biblioteca de interfaz en C para el soporte de Netware de las comunicaciones interprocesos (IPC) estándares para aplicaciones basadas en servidor, incluyendo NetBIOS, Unix Transport Library Interface (TLI), Named Pipes y SPX.
- El Btrive, SQL y servicio de gestión de correos (MHS) de Netware ofrecen importantes interfaces de programación, que pueden ser empleadas para crear aplicaciones distribuidas.

5.1.6. Restricciones de tiempo

El tiempo máximo especificado de duración del proyecto es del 1 de Agosto de 1996 al 31 de Julio de 1997 (ver la Tabla 5.1.).

Las etapas a cubrir en este tiempo son las siguientes:

- Análisis funcional del sistema.
- Análisis funcional detallado (Diseño)
- Codificación del sistema
- Pruebas y mantenimiento al sistema
- Optimización
- Desarrollo de documentación

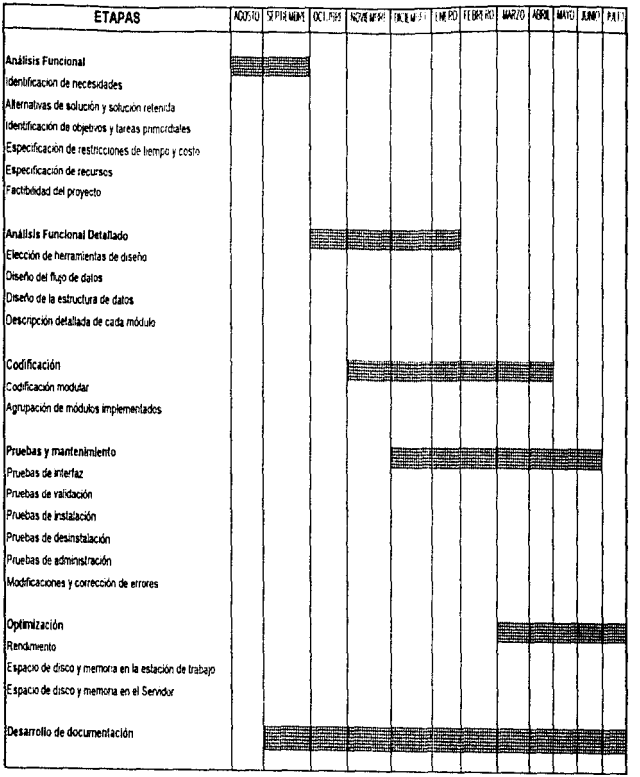


Tabla 5.1. Asignación de tiempos del proyecto INGRESO

5.2. ANALISIS FUNCIONAL DETALLADO

El análisis funcional detallado de un sistema, es una actividad de construcción de modelos, utilizando un método de análisis en particular.

Mediante una notación que es única del método de análisis, se crean modelos que reflejan el flujo y el contenido de la información, haciendo particiones funcionales del sistema según los distintos comportamientos, se establece la esencia de lo que se debe construir.

El análisis funcional detallado del proyecto INGRESO muestra un panorama del sistema, utilizando la metodología del *diseño orientado al flujo de datos*, la cual emplea las características del flujo de información para derivar la estructura de un programa.

5.2.1. Diseño Orientado al Flujo de Datos (DFD)

Un factor importante para la elección de un método de diseño es la variedad de áreas en las que pueda aplicarse. El diseño orientado al flujo de datos puede utilizarse en un amplio rango de áreas de aplicación. De hecho, debido a que todo el software puede representarse mediante un *diagrama de flujo de datos*, podría teóricamente aplicarse a cualquier desarrollo de software. Los autores Stevens W., G. Myers y L.Constantine fueron los primeros que propusieron el diseño de software basado en el flujo de datos a través de un sistema. Los primeros trabajos fueron refinados y presentados en los libros de Myers y de Yourdon y Constantine.[ROG93]

El *diagrama de flujo de datos* (DFD) es una técnica gráfica empleada en esta metodología, que representa el flujo de información y las transformaciones que se aplican a los datos al moverse desde una entrada hasta la salida.

En la figura 5.2 se ilustra la notación básica que se usa para crear un diagrama de flujo de datos DFD.

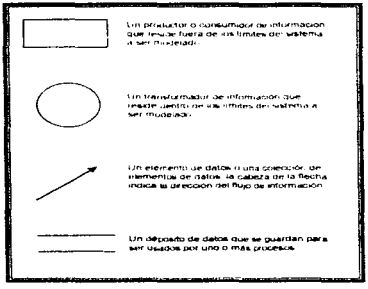


Figura 5.2. Notación DFD básica.

El diagrama de flujo de datos se puede usar para representar sistemas a cualquier nivel de abstracción. Los DFDs pueden ser refinados en niveles que representen un mayor flujo de información y un mayor detalle funcional.

La figura 5.1. muestra un DFD de nivel 0, también *llamado modelo fundamental del sistema*, y representa al sistema completo. A partir del DFD del nivel 0, el sistema puede dividirse en pequeños módulos, para ser analizados por separado.

La facilidad de uso y comprensión del diseño orientado al flujo de datos, ha provocado un gran interés en algunos investigadores por enfocarlo al *diseño orientado a objetos (DOO)*. Este método de diseño introduce un nuevo conjunto de términos, notaciones y procedimientos para la derivación del diseño de software. Autores como Ward describen una correspondencia entre las dos técnicas de diseño [POG93]. Por ejemplo, se puede representar una abstracción de datos (una *clase o subclase* para el DOO) como una burbuja. Los flujos hacia y desde esa burbuja siguen representando flujo de datos, pero implican operaciones asociadas a la clase. Esta relación se tomará en cuenta en la implementación del sistema.

[POG93] Roger S. Pressman, *Ingeniería del Software*, Mc Graw Hill, 1993, pp 386, 444

ESTA TESIS NO DEBE SALIR DE LA BIBLIOTECA

El sistema del Servidor INGRESO debe detectar cuando un paquete ha sido desinstalado por el usuario o cuando no ha sido utilizado después de un tiempo específico, este tipo de información es continua en el tiempo. Para poder representar el flujo de datos continuo en el tiempo, autores como Ward, Mellor, Hatley y Pirbhai han ampliado la notación básica. Estas ampliaciones se muestran en la figura 5.3.

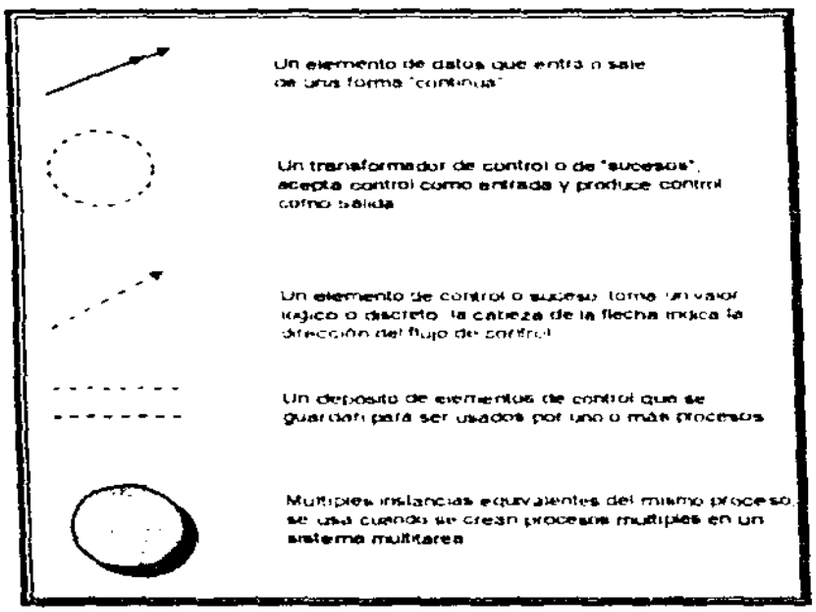


Figura 5.3. Ampliación de la Notación DFD básica.

5.2.2. Diseño Orientado al Flujo de Datos del Proyecto INGRESO

Como ya se había mencionado en el funcionamiento general del sistema, existirá un programa que se cargará en cada una de las estaciones de trabajo (*programa Cliente*), que se encargará de establecer la sesión con el Servidor INGRESO. Por otro lado, otro programa será el administrador de instalaciones (*programa Servidor*), el cual contestará la llamada de los usuarios. El diseño de cada programa se efectuará por separado, haciendo referencia a la interacción entre los dos.

5.2.2.1. Programa Cliente

El programa Cliente colaborará de manera importante para llevar a cabo los objetivos del Servidor. El análisis del sistema Cliente se divide en las dos tareas principales del proyecto, la instalación y la desinstalación de software.

Instalación

Utilizando la notación del *diseño orientado al flujo de datos*, se presenta gráficamente el flujo de información y los primeros procesos generales que intervienen para el programa que se cargará en las estaciones de trabajo (figura 5.4.).

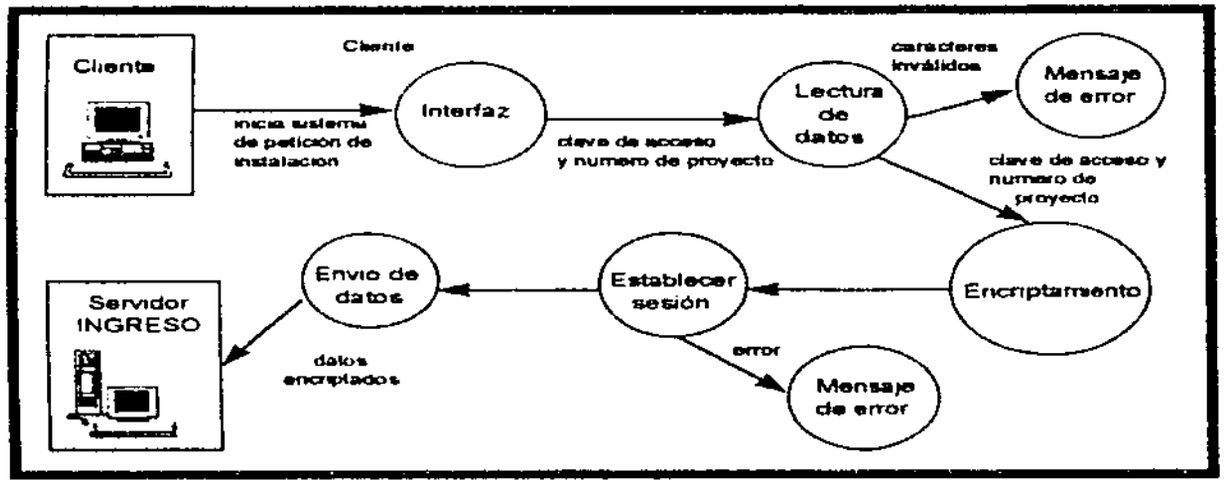


Figura 5.4. Primera parte del DFD del sistema Cliente.

Cuando la estación de trabajo ejecute el programa *Cliente*, iniciará el sistema de petición de instalación. El primer módulo del programa llamado **Interfaz**, se encargará de desplegar una caja de diálogo en ambiente Windows, la cual permitirá al usuario introducir su clave de acceso y número de proyecto.

El modulo lectura_de_datos permitira extraer las entradas del ambiente grafico, para verificar que los caracteres introducidos sean validos, de no ser así, se activará el módulo mensaje_de_error.

La clave de acceso y el número de proyecto pasarán por un proceso de encriptamiento antes de pasar por el cable, para evitar que derivaciones no autorizadas del cable permitan determinar estos datos.

Por medio del módulo establecer_una_sesión se tratará entablar comunicación con el Servidor INGRESO. Si se establece la sesión, los datos encriptados se envian al Servidor por medio del proceso envio_de_datos, de no ser así se activará el módulo mensaje_de_error

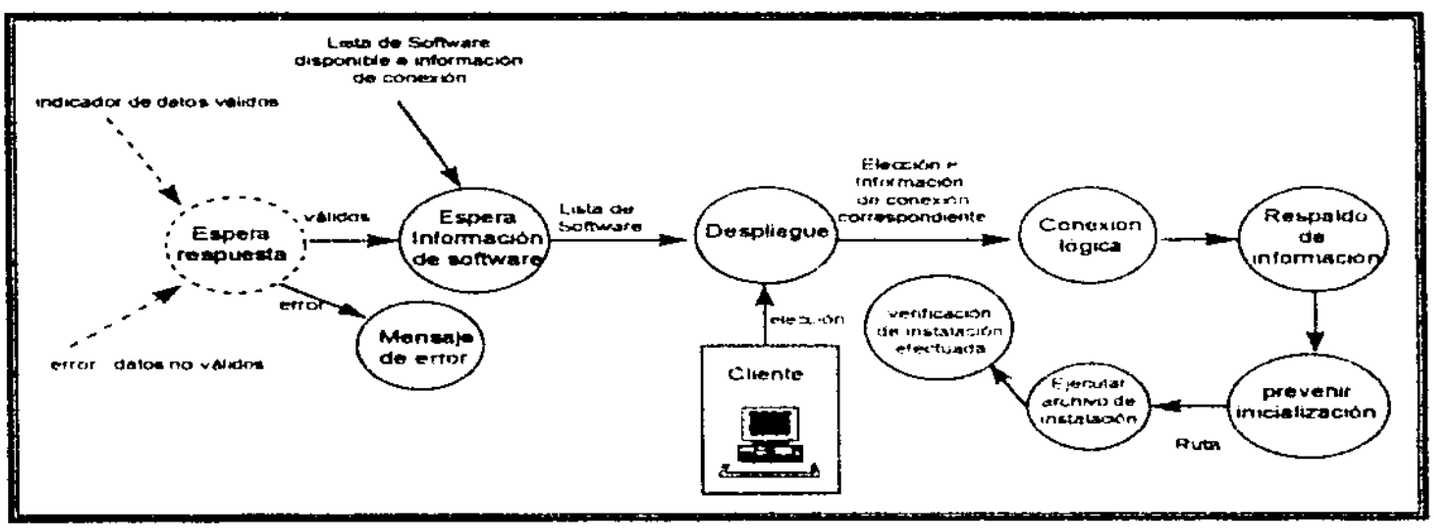


Figura 5.5. Segunda parte del DFD del sistema Cliente

En la figura 5.5. se muestra la segunda parte del DFD del sistema Cliente. En el diagrama se puede observar que una vez que el sistema Cliente ha enviado los datos del usuario el módulo espera_respuesta se queda pendiente del Servidor. Las posibles

entradas a éste módulo son elementos de control o suceso un indicador de datos válidos o un indicador de error Si se determina que es un error, el mensaje_de_error se despliega. De otra forma, los datos enviados fueron aceptados, y el módulo espera_información_de_software recibira del Servidor una lista de software disponible e información necesaria para una conexión lógica posterior.

El módulo despliegue mostrará la lista de software por medio de una caja de diálogo. El usuario se deslizará por la lista de paquetes a través de la iluminación deslizante o por medio del ratón, y una vez iluminada su elección, podrá presionar un botón de para iniciar la "Instalación" o uno para "Cancelar" la operación.

Una vez que el usuario ha tomado la decisión de efectuar la instalacion, su elección de software pasará por el módulo llamado Conexión_lógica, el cual recibirá como entrada la *información de conexión* del paquete en particular, antes recibida. Este módulo se encargará de efectuar una conexión lógica con el disco duro del Servidor, especificamente con el subdirectorio que contiene el archivo de instalación del software deseado. El contenido de la información de conexión se detallará más adelante.

Antes de llevar a cabo la instalación, se debe recordar que el proyecto INGRESO deberá ser capaz de llevar a cabo una desinstalación cuando un usuario este monopolizando una licencia de un paquete sin hacer uso de él.

Para que ésta función pueda realizarse, el sistema Cliente deberá guardar en un archivo una lista de los archivos y directorios que están almacenados en su disco duro antes de efectuarse la instalación.En ésta lista los archivos se listarán junto con su tamaño. Por otro lado, también se debe hacer una copia del archivo WIN.INI y SYSTEM.INI, utilizando otros nombres. El módulo respaldo_de_información se encargará de ésta tarea.

La mayoría de los programas de instalación reinician la computadora después de la instalación, esto lo hacen para asegurar que los cambios efectuados a los archivos de inicialización del sistema tomen efecto.

Si existe una inicialización, la conexion con el servidor INGRESO se perderá. Pero es necesario que el servidor esté enterado si la instalación pedida por el usuario se llevó a cabo.

Antes de ejecutar el archivo de instalación el modulo prevenir_inicialización añadirá una linea al archivo correspondiente de inicialización de Windows® para que ejecute un programa de "*verificación*" Este programa estara presente en el Cliente y se activara en la próxima reinicializacion de la PC.

Si la instalacion se llevo a cabo, y el programa de instalación requirió una inicialización del sistema, el programa de *verificación* se conectará al servidor INGRESO para darle aviso de que la instalación se llevó a cabo (el servidor por su lado, deberá restar una licencia). Cuando lo haya hecho, la linea añadida al archivo de inicialización de Windows será borrada.

Por lo tanto, el diálogo entre el servidor y clientes puede darse en más de una sesión, o conexión entre ambos.

Si el programa de instalación no reinicia la PC, el programa de verificación se ejecutará después de la instalación.

Desinstalación

Para que el programa de verificación detecte que la instalación se realizó, tendrá que efectuar una comparación de cada archivo y directorio actual (después de la instalación), con los listados en el archivo elaborado antes de la instalación. Comparará el tamaño actual de cada archivo con el estampado en la lista previa. Si existen diferencias, el sistema notará que el archivo ha cambiado, y lo incluirá en un *reporte de salida*. Si estos son los mismos los ignorará. Algún archivo que no este presente el la lista previa, fue obviamente añadido durante la instalación, así que el sistema escribirá este nombre en el reporte.

Cuando esta comparación de archivos y directorios es completada, el sistema Cliente comparará los respaldos del los archivos WIN.INI y SYSTEM.INI con los archivos actuales. Las secciones nuevas o cambios efectuados en estos archivos se incluirán en otro reporte para el usuario.

Estos reportes serán indispensables para asegurarse que una instalación se realizó y principalmente para llevar a cabo una *desinstalación* posterior. Primero, el reporte que incluye una lista de todos los archivos y directorios añadidos por el programa, listará los archivos por borrar. Sin embargo, desinstalar no significará literalmente borrar cada archivo contenido en la lista. Es posible que otro programa que se instale posteriormente, trabaje sobre los mismos archivos.

Por ejemplo, suponer que el programa A instala el archivo FOO.DLL en el directorio WINDOWS\SYSTEM. Si un programa B utiliza el mismo DLL, al ser instalado ya no es necesario sobreescribir el mismo archivo. Si se borra el DLL bajo un programa de desinstalación del programa A, se deshabilitará el programa B.

Los archivos que son añadidos en un directorio creado por el programa de instalación, pueden ser borrados sin preocupación. Los archivos añadidos en un directorio "publico" como el WINDOWS o el WINDOWS\SYSTEM, deben ser tratados con más cuidado. No deben borrarse, sin embargo el usuario basandose en la lista de archivos añadidos a estos directorios, puede renombrarlos o moverlos. Entoces probar que todas las aplicaciones funcionen correctamente.

Por otra parte, el reporte de secciones y claves añadidas a los archivos INI importantes también deben manejarse con cuidado. Ya que algunos programas de instalación modifican el valor de claves existentes dentro de los archivos INI.

Si el programa de instalación añadió toda una nueva sección al WIN.INI o toda una nueva rama al árbol de registros, se pueden eliminar estos bloques sin problemas. Pero si fueron añadidas líneas en secciones existentes o valores en claves existentes, la solución se complica.

Sin embargo, para no borrar estas estas lineas añadidas, se pueden poner como comentarios, añadiendo punto y coma al principio de la linea Si este cambio causa problemas, simplemente se puede quitar el punto y coma Este metodo no es aplicable para las claves de registro

Si el programa de instalación cambio el valor de una clave o registro de un archivo INI, no se puede cambiar a ciegas este valor Se puede hacer una copia de la linea en el archivo INI, y comentarla, y en la otra linea cambiar el valor de la variable por el anterior, y probar si no afecta a otro programa.

Los reportes de archivos nuevos y modificaciones, serán proporcionados al Cliente, como apoyo para una desinstalación posterior

Una de las funciones del proyecto INGRESO, es impedir que un usuario monopolice la licencia de un paquete sin hacer uso de él. Para llevar a cabo esto, se debe de detectar cuando un paquete instalado por el servidor no ha sido utilizado después de un tiempo x, y entonces "desinstalar el paquete"[5] o al menos recuperar su licencia notificando al usuario.

Un contador será añadido por cada paquete instalado. y a partir de la fecha de instalación se incrementará con cada día transcurrido. El contador antes de incrementarse verificará que el archivo .EXE más importante de la aplicación aún exista, de esta forma se podrá detectar al Cliente que desinstaló por si mismo el paquete Si el archivo no se encuentra, se llamará al Servidor para que éste libere una licencia.

Existirá un programa residente en el Cliente, que detecte cuando una aplicación instalada por el servidor está por ejecutarse, en este caso el contador correspondiente será inicializado a cero.

Si el contador llega al límite de tiempo preestablecido, automáticamente se le preguntará al usuario que el paquete no ha sido utilizado por lo que se le retirará la

[5] Una medida drástica sería eliminar los archivos ejecutables del paquete en cuestión, aun cuando no se eliminen sus demás componentes

licencia, el sistema puede desinstalarlo en ese momento si él lo desea, de otra forma queda bajo su responsabilidad la utilización del mismo. De cualquiera de las dos formas, se le dará aviso al Servidor para que sea liberada una licencia.

La estación de trabajo podrá iniciar una sesión con el servidor sin que el usuario se entere. Este tipo de programas conocidos como "watch-dogs" ejecutan alguna acción como resultado de un evento. En este caso, las sesiones con el servidor se establecerán cuando se detecte que un paquete no ha sido usado después de un tiempo determinado, cuando el archivo ejecutable más importante de la aplicación no fue localizado (esto puede significar una desinstalación por el usuario) o cuando se debe informar al servidor que el paquete fue instalado después de una reinicialización.

5.2.2.2. Programa Servidor

El problema es grande y complejo como para que se pueda comprender como un todo. Por esta razon, se divide en partes que se puedan entender fácilmente y establecer interfaces entre las partes, de forma que se realice la función global

A continuación en la figura 5.6 se hace una partición inicial para descomponer al software del Servidor de Instalaciones en tres principales módulos.

Figura 5.6. Primera división funcional del programa del servidor INGRESO

La partición llamada monitoreo de las estaciones de trabajo, tendrá como función: recibir confirmaciones de que un paquete ha sido instalado correctamente o que alguno ha sido desinstalado.

La partición interacción remota con el usuario, se encargará de contestar la llamada de los usuarios y de atender las peticiones de instalación.

Por último, la gestión de bases de datos se refiere a la administración de las bases de datos *usuanos*, *proyectos* y *software*. El encargado de esta parte es la persona nombrada como *Administrador de la base de datos*, quien se encargará de dar de alta, dar de baja, hacer cambios, hacer consultas, etc., de los datos correspondientes a cada base de datos. El sistema mismo podrá accesar la información proporcionada por el Administrador de la red para: validar usuarios, verificar disponibilidad de licencias, actualizar el número de licencias disponibles (por una instalación o desinstalación), etc.

En seguida se analizan a detalle cada una de las primeras tres particiones del sistema del Servidor INGRESO.

Interacción remota con el usuario

Este submódulo se encargará, como ya se había dicho, de contestar la llamada de los usuarios y de atender las peticiones de instalación.

El sistema del Servidor tendrá que estar atento a la llamada de los Clientes, una vez que recibe alguna podrá dar servicio. Para comprender la secuencia del Sistema INGRESO es necesario recordar el DFD de la figura 5.4., en el cual el usuario ya ha establecido una sesión con el servidor y le ha enviado sus datos. Aquí es donde entra en juego el diagrama siguiente de la figura 5.7.

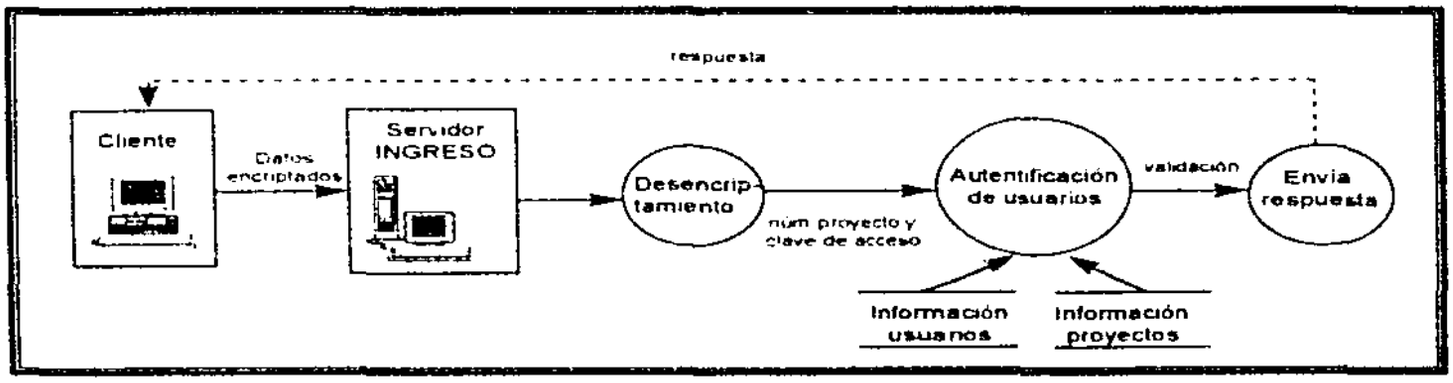


Figura 5.7. Primera parte del DFD de la división *interacción con el usuario* del programa Servidor.

Una vez que el sistema Cliente envía los datos del usuario (clave de acceso y número de proyecto) encriptados, el programa del Servidor los pasará por un módulo de desencriptamiento, obteniendo los datos originales.

Los datos originales pasarán por una verificación a través del módulo de **autentificación_de_usuarios**. el servidor de instalaciones confirmará la identidad del usuario y validará el número de proyecto, haciendo uso de las bases de datos **información_ usuarios** e **información_proyectos** y la relación entre ellas.

Si los datos no son válidos el módulo **envía_respuesta** se encargará de enviar un elemento de control para activar un mensaje de error en la estación de trabajo y cerrará la sesión. Si por el contrario, los datos son válidos se enviará un *indicador de datos válidos*, y el sistema Ciente espera la lista de paquetes disponibles, tal y como se mostró en la Figura 5.5.

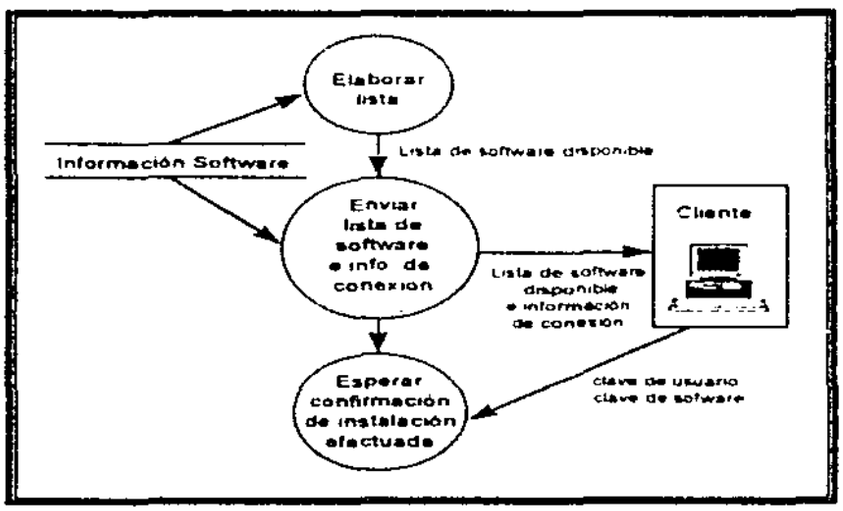


Figura 5.8. Segunda parte del DFD de la división *interacción con el usuario* del programa Servidor

El DFD de la figura 5.8. muestra el comportamiento del sistema Servidor si los datos recibidos son válidos. Primero, el modulo **elaborar_ lista** consultará la base de datos **Información_Software** para crear una lista de los paquetes que están presentes y tienen licencias disponibles.

La lista de paquetes disponibles entrará al modulo enviar_lista_de_software e_información_de_ conexión, el cual enviara al sistema Cliente la lista elaborada por el módulo anterior, y a traves de una consulta a la base de datos información_software enviará tambien la información de conexion de los paquetes especificados en la lista.

Antes de especificar que contiene la información de conexion, es necesario explicar la forma de ordenar los paquetes de software en el Servidor.

Para que la conexión lógica tenga sentido, en el Servidor los paquetes deben permanecer ordenados de acuerdo al nombre de la aplicación y por discos. Por ejemplo, si la letra de la unidad local en uso del Servidor fuera C, el software fuera Turbo C versión 2.0 y este fuera un paquete de 3 discos; en el servidor se guardaría cada disco de la siguiente forma:

C:\TC2\disco1
C:\TC2\disco2
C:\TC2\disco3

Cada directorio de paquete, en este caso TC2, será compartido en red por medio del Administrador de Archivos de Windows® o el Explorador de Windows95®, todos ellos con un *tipo de acceso de sólo lectura* y con *password*. El tipo de acceso de sólo lectura de un directorio, permite a los usuarios conectados a él leer archivos o ejecutar aplicaciones, pero de ninguna forma modificarlos o eliminarlos. Y el password es una clave de acceso al directorio que se utiliza como protección.

Una vez explicada la forma de guardar los paquetes en el servidor, podemos aclarar que la información de conexión contendrá: la ruta del directorio de cada paquete en particular, el nombre del archivo de instalación y el password correspondiente. Esta información permitirá una conexión lógica de la estación de trabajo con el Servidor y será dada de alta (entre otros datos) por la persona encargada de administrar la red, en la base de datos información_software.

Monitoreo a Clientes

Siguiendo con el analisis de la figura 5 8 , el Servidor de aplicaciones no cierra la sesión cuando envia la lista de software y la información de conexión, ya que si la aplicación no reinició la estación de trabajo tendrá que verificar si la instalación se ha efectuado.

El sistema Cliente confirmará al Servidor que la instalación se llevó a cabo. Si el programa de instalación necesita reiniciar la máquina Cliente, irremediablemente se perderá la conexión con el Servidor.

El sistema del Servidor esperará llamadas nuevamente teniendo la capacidad de identificar entre una petición de instalación y una confirmación de que un paquete fue instalado o desinstalado.

El cliente tratará de conectarse con el Servidor cada vez que se reinicie la PC. hasta que pueda confirmarle que la aplicación se instaló correctamente. Los parámetros enviados serán: la clave de usuario y la clave de software.

Si la instalación ha sido satisfactoria, el servidor deberá activar un módulo de actualización_de_licencias (o administración) para restar la licencia otorgada al cliente del total de licencias. También se debe llevar un registro de la clave del usuario que retiene la licencia en particular.

En el caso de una confirmación de desinstalación automática o una hecha por o el usuario, el servidor aumentará en uno el número de licencias disponibles correspondiente clave de software recibida y borrará el registro en el cual se especificaba que usuario retenia la licencia.

Gestión de Bases de Datos

Para que el Servidor INGRESO pueda administrar las licencias de los paquetes, autentificar a los usuarios y validar los proyectos, es necesario que almacene y maneje colecciones de datos (bases de datos) mutuamente relacionadas.

En el capítulo 4 se habló de las bases de datos distribuidas y centralizadas; en el caso del Servidor INGRESO, las bases de datos que manipulará serán centralizadas.

En el análisis de las entradas del sistema en general, se detectaron tres principales bases de datos:

1. INFORMACIÓN PROYECTOS
2. INFORMACIÓN USUARIOS
3. INFORMACIÓN SOFTWARE

El análisis de éstas bases de datos debe apoyarse en un *Modelo de Datos*, es decir, un grupo de herramientas conceptuales para describir los datos, sus relaciones, semántica y sus limitantes. Por medio del *Modelo Relacional* se especificará cada base de datos.

En el Modelo Relacional los datos y las relaciones entre los datos se representan por una serie de tablas. Cada tabla tiene asignado un nombre único, siendo éste el nombre de la base de datos. Cada renglón es llamado *registro*, y guarda datos acerca de un objeto o tema en particular. Un registro está formado por *campos* (columnas con nombres únicos).

Para comprender estos conceptos en seguida se muestra la tabla 5.2 correspondiente a la base de datos INFORMACIÓN PROYECTOS.

INFORMACIÓN PROYECTOS

Clave de proyecto	*Título*	*Responsable*
BBA-2002	Instalación y Gestión Remota de Software	Dr. Uriel Tirado Rios
BBA-2003	Administración de Impresión	Ing. José Hernández Monroy

Tabla 5.2. Información Proyectos

Identificando los conceptos

INFORMACION PROYECTOS es el nombre de la base de datos

Clave de proyecto, *Título* y *Responsable* son los campos.

Como ejemplo se han dado dos renglones, cada uno representa un registro.

Los campos de la base de datos INFORMACIÓN PROYECTOS necesitan tener una definición más precisa y rigurosa, que permita en el desarrollo del sistema tener una mejor comprensión. Para esto, se utiliza el llamado *Diccionario de Datos*.

El Diccionario de Datos contiene metadatos, es decir datos acerca de los datos. La estructura usada para desarrollar esta descripción del contenido, se ilustra en la Tabla 5.3., especificamente para la base de datos INFORMACIÓN PROYECTOS.

Nombre del atributo	*Tipo*	*Long*	*Descripción*	*Status*
Clave de proyecto	char	10	Clave asignada a un proyecto particular	llave
Título	char	60	Nombre del proyecto	no llave
Responsable	char	55	Nombre de la persona responsable del proyecto	no llave

Tabla 5.3. Diccionario de Datos de Información Proyectos

El tipo y longitud de cada campo es importante para la adecuación de una variable en el desarrollo del software. La descripción es una breve explicación de lo que representa en la realidad cada campo. Por último, el status indica si el campo es una *llave* o no. Un campo es una llave cuando por medio de el se pueden identificar los registros.

Las llaves también permiten relacionar (asociar) tablas. Dado que cada usuario del servidor debe estar inscrito en un proyecto, la tabla de INFORMACIÓN USUARIOS estará relacionada a la tabla INFORMACIÓN PROYECTOS por medio de la llave Clave de proyecto. Esta relación especificará que usuarios están asignados a cada proyecto.

La tabla correspondiente a la base de datos INFORMACIÓN USUARIOS y el Diccionario de Datos de la misma, se muestran a continuación en las tablas 5.4 y 5.5 respectivamente.

INFORMACIÓN USUARIOS

Clave de usuario	*Clave de proyecto*	*Nombre*	*Apellidos*	*Status*
140125	BBA-2003	Raúl	Vizcarra Mora	TRUE
140124	BBA-2002	Esperanza	Solis Escamilla	FALSE

Tabla 5.4. Información Usuarios

Nombre del atributo	*Tipo*	*Long*	*Descripción*	*Status*
Clave de usuario	char	10	Clave de acceso asignada al usuario	llave
Clave de proyecto	char	10	Clave del proyecto al que esta inscrito	llave
Nombre	char	15	Nombre del usuario	no llave
Apellidos	char	40	Apellidos del usuario	llave
Status	bool		Empleado (TRUE) o Becario (FALSE)	no llave

Tabla 5.5. Diccionario de Datos de Información Usuarios

Cada vez que se almacene un paquete en el Servidor, la persona encargada de administrar la red debe de dar ciertos datos del mismo. La base de datos encargada de administrar esta informacion sera INFORMACION SOFTWARE (tablas 5.6 y 5.7)

INFORMACIÓN SOFTWARE

Clave de Software	*Nombre*	*Num. de licencias*	*Ruta*	*Archivo instalación*	*Archivo EXE*	*Clave ID*
INGTC002	Turbo C v 2.0	4	C:\TC2	setup.exe	tc.exe	
INGEX004	Excel v 4.0	5	C:\EX4	install.exe	excel.exe	11796-OEM-0010006-64471

Tabla 5.6. Información Software

Nombre del atributo	*Tipo*	*Long*	*Descripción*	*Status*
Clave de Software	char	10	Contraseña de directorio e identificador	llave
Nombre	char	46	Nombre del producto de software	no llave
Núm. de licencias	int		Número de licencias compradas	no llave
Ruta	char	46	Directorio donde se encuentra el software	no llave
Archivo instalación	char	13	Nombre del archivo de instalación	no llave
Archivo EXE	char	13	Nombre del archivo .EXE más importante	no llave
Clave ID	char	24	Clave del producto de fabrica	no llave

Tabla 5.7. Diccionario de Datos de Información Software

La base de datos INFORMACIÓN SOFTWARE por si sola no administra las licencias de software. Asi que debe existir una relación con INFORMACIÓN USUARIOS que muestre las licencias otorgadas y los usuarios que las poseen. La tabla utilizada para este fin, es la base de datos INSTALACIONES (tabla 5.8).

INSTALACIONES

Clave de Software	*Clave de usuario*	*fecha*
INGTC002	140125	09-06-97
INGEX004	140124	21-03-97

Tabla 5.8. Instalaciones.

Nombre del atributo	*Tipo*	*Long*	*Descripción*	*Status*
Clave de Software	char	10	Identificador de software	llave
Clave de usuario	char	10	Clave de usuario que retiene una licencia	llave
fecha instalación	date		Fecha en que se confirmó la instalación	no llave

Tabla 5.9. Diccionario de Datos de Instalaciones

Para llevar un control de las licencias con las que se cuenta por cada paquete, se utilizará una tabla auxiliar: LICENCIAS DISPONIBLES (tabla 5.10). Esta tabla se accesará cada vez que el Servidor reciba una confirmación de instalación o desinstalación. El campo por modificar será *licencias disponibles*.

LICENCIAS DISPONIBLES

Clave de Software	*Total de licencias*	*Licencias disponibles*
INGTC002	4	2
INGEX004	5	0

Tabla 5.10. Licencias Disponibles.

La descripción de cada campo se muestra en el diccionario de datos de la tabla 5.11.

Nombre del atributo	*Tipo*	*Long*	*Descripción*	*Status*
Clave de Software	char	10	Identificador de software	llave
Total de licencias	int		Total de licencias contratadas	llave
Licencias disponibles	int		Licencias no otorgadas	no llave

Tabla 5.11. Diccionario de Datos de Licencias disponibles

CAPÍTULO 6

CAPA DE TRANSPORTE DEL PROYECTO INGRESO

6.1. Protocolos de transporte

Los protocolos de transporte funcionan en los niveles de transporte y de red del modelo en capas OSI. Ofrecen rutinas de bajo nivel para la transferencia de información durante una sesión de comunicaciones en red. En las siguientes secciones explicaré las principales características de algunos de los protocolos de más uso, y el papel que desempeñan en las redes.

6.1.1. IPX/SPX

IPX

El Internetwork Packet Exchange (IPX, intercambio de paquetes en red) , es una versión del protocolo de servicios de red de Xerox. Se creó específicamente para el NetWare de Novell. Comparado con el modelo de referencia OSI, IPX es un protocolo del nivel de red (ver figura 6.1). IPX se encarga del direccionamiento y el enrutamiento de mensajes hacia otras computadoras y de enviar los datos que entran a los procesos locales correctos. IPX proporciona un servicio de datagramas, lo cual implica que la transmisión de datos no es confiable, es decir, no se garantiza la entrega satisfactoria de un "paquete". Algunas garantías pueden proporcionarse por un protocolo del nivel superior, tal como SPX (Sequenced Packet Protocol) o NCP (NetWare Core Protocol).

Los mensajes son enviados por partes dentro de *paquetes*, éstos contienen la dirección completa de remite y de destino. Las direcciones están divididas en tres partes: la red, el nodo y el socket (conector).

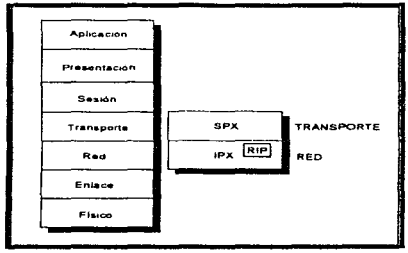


Figura 6.1. Comparación de IPX/SPX con el modelo de referencia OSI.

La parte de *red* de la dirección es una dirección lógica, que determina el segmento de cable de red físico al que está conectada la estación de trabajo. Esta parte de la dirección consiste de un número de 32 bits (4 bytes).

La *dirección del nodo* identifica la tarjeta adaptadora de red de la estación de trabajo y está formada por 48 bits (6 bytes).

Y finalmente, se utiliza un *socket* (conector) para identificar un proceso software concreto de destino. El número de socket es un número de 16 bits (2 bytes) asignado para cada proceso que quiere comunicarse utilizando los servicios IPX.

Internamente, el número de socket es utilizado para acceder estructuras de datos utilizadas para comunicarse con el protocolo IPX.

El socket es importante, ya que distintos procesos de software tales como NCP (NetWare Core Protocol), SAP (Service Advertising Protocol) y servicios RIP (Routing Information Protocol) pueden correr simultáneamente en un nodo NetWare. Algunos de los números de socket más conocidos son: 451hex para NCP, 452hex para SAP, 453hex para RIP y 455hex para NetBIOS.

La dirección completa de un proceso sobre un nodo NetWare consiste de los siguientes tres valores: <número de red, dirección del nodo, número de socket>

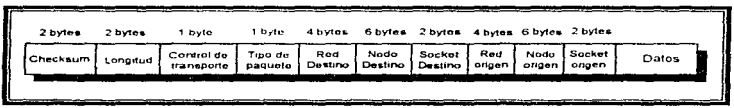


Figura 6.2. Estructura de un paquete IPX.

La figura 6.2. muestra la estructura de un paquete IPX. En seguida se describen los campos que integran un paquete IPX.

- *CheckSum*. Usualmente este campo está fijo en FFFFhex para indicar que CheckSum está inhabilitado. IPX confía en que el nivel de enlace de datos informe acerca de los errores de los paquetes porque los protocolos de enlace de datos, tales como Ethernet y Token Ring, tienen un hardware Cyclic Redundancy CheckSum (CRC).

- *Longitud*. Este campo indica la longitud de los paquetes IPX en bytes, incluyendo la cabecera IPX de 30 bytes más el campo de datos.

- *Control de transporte*. Se utiliza como contador de *routers*[1] (ruteadores) atravesados por el paquete IPX. Este campo se incrementa cada vez que el paquete atraviesa un router. Cuando el contador alcanza los 16 routers, el paquete se asume como perdido y es descartado.

- *Tipo de paquete*. El tipo de paquete identifica cuál protocolo del nivel superior debe recibir la porción de datos del paquete IPX. Algunos de los codigos más usuales de este campo son: 4 para PXP (Packet Exchange Protocol), 5 para SPX (Sequenced Packet Protocol) y 17 para NCP (NetWare Core Protocol). El tipo de paquete cero es reservado para un tipo de paquete no conocido.

- Los campos *red destino, nodo destino y socket destino* identifican de manera única a un proceso en el nodo destino.

- Los campos *red origen, nodo origen y socket origen* identifican a un proceso en el nodo de origen.

- *Datos*. Por último, este campo contiene los datos correspondientes al paquete para ser transmitidos. IPX originalmente heredó un límite de 576 bytes de la estructura de un paquete XNS (Sistema de Red Xerox). Sin embargo, nuevos drivers IPX pueden manejar paquetes más grandes.

[1] Los servidores de comunicaciones que encadenan las LAN, normalmente utilizan técnicas sofisticadas de inspección de paquetes para enrutar su tráfico hacia su destino, por eso se les llama *routers* (ruteadores).

Todos los nodos en la misma red física deben tener el mismo número de red. Si el número de red destino es el mismo que el numero de red local, el paquete IPX es enviado directamente al nodo sobre la red local. Si el número de red destino es diferente que el número de red local, y ésta es la primera vez que un paquete IPX es enviado a la red destino, una petición de ruteo del paquete se envía al *RIP* (Routing Information Protocol) para determinar la ruta más rápida.

La respuesta contiene la dirección de un router local capaz de apresurar el envío del paquete. El paquete IPX entonces es enviado a este router. Los ruteadores IPX tienen una tabla que contiene información de ruteo para todas las redes alcanzables para cada router. Típicamente, el router IPX también es el servidor de archivos NetWare, porque todos los servidores de archivo NetWare contienen un módulo de software ruteador IPX. Los routers IPX intercambian información de ruteo cada 60 segundos por default. El formato y naturaleza de la información está especificada por el RIP de Novell. Para ajustar este intervalo de tiempo se utiliza SERVMAN.NLM en el servidor de archivos NetWare v 4.x.

El protocolo IPX se ha optimizado para su uso en redes locales, ofreciendo un bajo consumo de recursos y un alto rendimiento. IPX no es un buen protocolo para las redes de gran alcance, siendo a menudo necesario utilizar TCP-IP o X.25 cuando se necesita un protocolo de transporte fiable en un área extensa.

SPX

Sequenced Packet Exchange (SPX, intercambio de paquetes secuenciados), se encuentra por encima del IPX (ver figura 6.1) y se asegura de que los paquetes se reciban en orden y sin errores.

SPX es una interfaz orientada a la conexión que ofrece comprobación de errores, ventanas y control de flujo. SPX reside en la capa de transporte y parcialmente en la de sesión del protocolo en capas. Cuando es necesaria una transmisión de datos fiable por parte de las aplicaciones se utiliza SPX. Se establece un circuito virtual entre las dos conexiones, y se utilizan procedimientos especiales para detectar los paquetes perdidos. Se ha de confirmar la llegada de los paquetes, si el paquete no se recibe se retransmite.

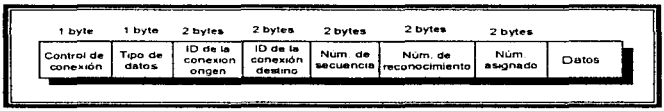


Figura 6.3. Estructura de un paquete SPX.

La figura 6.3. muestra la estructura de un paquete SPX. A continuación se describe a los campos que lo integran.

- *Control de conexión.* Este campo se utiliza para regular el flujo de datos a través de la conexión. Por ejemplo, la secuencia de bits 0001000, se usa como una señal de fin del mensaje.

- *Tipo de datos*. El campo se usa para indicar la naturaleza de los datos contenidos en el campo de datos SPX, y para identificar el protocolo del nivel superior al cual los datos deberán entregarse.

- El *ID de la conexión origen* y el *ID de la conexión destino* son los números del circuito virtual utilizados para identificar una sesión. Estos IDs se usan para demultiplexar circuitos virtuales separados sobre un único socket.

- *Número de secuencia*. Este campo se usa para enumerar cada paquete enviado, y lo utiliza SPX para detectar la pérdida de un paquete o si un paquete está fuera de secuencia.

- *Número de reconocimiento*. Indica el paquete próximo que el recibidor espera. Los campos de reconocimiento y secuencia mantienen a las computadoras que envían y reciben, informadas correctamente acerca de cuál paquete tiene que ser enviado y cuál tiene que ser recibido.

- *Número asignado*. Se utiliza para indicar cuántos buffers libres tiene disponibles el receptor en una conexión.

Usualmente, las estaciones de trabajo NetWare no utilizan el protocolo SPX; lo utiliza directamente el protocolo IPX. La confiabilidad en la transmisión la mantiene el protocolo NCP (NetWare Core Protocol). SPX se usa para estabilizar conexiones remotas entre el servidor de impresión e impresoras remotas. SPX también se utiliza en NetWare SQL y conexiones remotas del servidor de archivos NetWare a través de RCONSOLE.

6.1.2. TCP/IP

TCP/IP es el acronimo de Transmission Control Protocol and Internet Protocol (Protocolo de control de transmisiones y protocolo entre redes). TCP/IP es una serie de estándares que se generó en 1969, fue desarrollado originalmente por el Departamento de Defensa de los Estados Unidos, en la Agencia de Proyectos de investigación Avanzada (ARPA, Advanced Research Project Agency) para conectar sistemas de computadoras distintas en redes de gran alcance a distancia que funcionasen sobre lineas telefónicas alquiladas.

TCP/IP se utilizó primero en una red llamada ARPANet (Advanced Research Projects Agency Network) y los resultados fueron tan exitosos que hoy en dia TCP/IP se encuentra en una amplia gama de sistemas LAN, y ARPANet ha crecido hasta transformarse en lo que hoy conocemos como *Internet*, una inmensa colección de computadoras interconectadas a lo largo y lo ancho de todo el mundo.

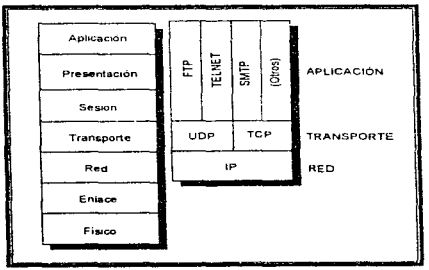


Figura 6.4. Comparación del modelo TCP/IP con el modelo OSI.

En la decada de los ochenta la Agencia DARPA ("Defense Advanced Research Projects Agency") estableció un acuerdo la Universidad de California en Berkeley para implementar el protocolo TCP/IP para el sistema operativo UNIX. A partir del código fuente de AT&T para una computadora VAX de Digital Equipment, la Universidad produjo su propia versión de UNIX conocida como UNIX 4.2 BSD (Berkeley Software Distribution) o UNIX de Berkeley. A partir de dicha década, los fabricantes de productos de comunicación se interesaron altamente en los protocolos TCP/IP, por lo cual han llegado a ser los estándares de facto más importantes para la interoperabilidad.

El modelo de red TCP/IP define sólo tres capas: RED, TRANSPORTE y APLICACIÓN, éstas se alinean con las del modelo de referencia OSI, como casi cualquier arquitectura de comunicaciones (ver figura 6.4.). La correspondencia entre las funciones de una capa OSI y aquellas de la capa equivalente de cualquier otra arquitectura no es exacta, pero los conceptos generales son muy similares. A continuación se describen en forma general las tres capas del TCP/IP:

CAPA DE RED

La capa de red llamada *IP* por "*Internet Protocol*" realiza la función de enviar los paquetes provenientes de la capa de transporte (ruteador de datagramas). Esto lo hace definiendo una estructura muy simple de direcciones (32 bits que se subdividen en 2 campos: uno para la dirección de red y otro para la dirección de máquina) y un mecanismo sin conexión que se encarga de enviar los datos sin asegurar que lleguen a su destino (servicio de datagramas).

Por conveniencia, las direcciones se dividen en cuatro grupos de 8 bits y se representan con su respectivo valor decimal separados por un punto ( ). A ésta notación se le llama "notación decimal punteada" En seguida se muestra una dirección IP en forma binaria y también como notación decimal punteada.

Dirección IP = 10010000 00010011 01001010 11001010

Dirección IP = 144 19 74 202

Los datagramas IP pueden ser fragmentados dentro de pequeñas unidades, las cuales contienen información suficiente para que puedan reensamblarse cuando lleguen al nodo destino. Los problemas encontrados al reensamblar son reportados al transmisor por el Internet Control Message Protocol (ICMP).

CAPA DE TRANSPORTE

La capa de transporte establece la comunicación entre los programas de aplicación, utilizando un mecanismo que se basa en el concepto conocido como buzón (ó "mail-box"). Este concepto significa que cuando un programa quiere comunicarse con otro, sólo precisa mandar sus datos a un puerto (un número entero) que identifica a un proceso que recibe información o acepta conexiones. Por otro lado, el programa que recibe información, lo hace leyendo los datos que fueron enviados a este puerto. Bajo TCP/IP los primeros 1024 puertos son reservados para ser usados como "puertos bien conocidos".[2]

[2] El protocolo FTP, por ejemplo, tiene reservado el puerto número 21 de TCP; asimismo el protocolo TELNET tiene reservado el puerto 23 de TCP y los RPC de Sun el 111 de TCP y UDP.

La capa de transporte ofrece dos protocolos, el primero es *TCP* por "*Transmission Control Protocol*", que se encarga de tomar mensajes arbitrariamente largos de las capas superiores y dividirlos en segmentos de 64 kilobytes o menos. A continuación TCP pasa los mensajes al IP para su transmisión, lo que puede implicar una división más (la cual es transparente para el TCP). El TCP también se encarga de mantener la secuencia de los mensajes que recibe y de reintentar las transmisiones fallidas.

TCP tiene 4 características fundamentales:

1 Confiabilidad en la transmisión los datos llegan en el mismo orden en que se enviaron y sin errores
2 Conexión de circuito virtual se establece una conexión para transmitir información y al finalizar se efectúa una desconexión
3. Flujo de bytes. El programa que lee de la conexión no sabe dónde empieza o termina un paquete de información, de hecho se lee como un flujo de bytes.
4 La conexión es "full-duplex". Ambos extremos de la conexión pueden leer y/o escribir en la conexión.

El segundo protocolo el *UDP* por "*User Datagram Protocol*" es una alternativa al servicio TCP para efectuar funciones simples y no críticas. UDP realiza una interfaz con la capa IP, para que las capas superiores puedan enviar mensajes (datagramas) cuando no se requiera de una entrega garantizada y no haya necesidad de establecer una sesión formal con el receptor.

Las características principales de UDP son:

1. No hay confiabilidad en la transmisión los datos pueden no llegar, llegar con errores, o en distinto orden en el que fueron mandados
2. La comunicación se realiza entre programas o procesos

CAPA DE APLICACIÓN

La capa de aplicación realiza sus funciones basadas en la capa transporte. Como ejemplos bien definidos de los protocolos de esta capa tenemos a *TELNET* que es un protocolo que se utiliza para simular una terminal a través de la red. Le permite desde su computadora ejecutar programas que se encuentran en otra computadora. Los datos en la pantalla del programa de la computadora remota aparecen desplegados en su pantalla, y los datos que usted introduce se envían a la computadora remota. TELNET le permite trabajar con los datos en donde se encuentran (de manera remota) y usar la red para controlar el acceso y transferencia de los resultados. En lugar de ejecutar localmente el programa en su máquina y tomar los datos de la computadora remota a través de la red.

Otro ejemplo es el protocolo *FTP* '*File Transfer Protocol*' o Protocolo de Transferencia de Archivos, que permite transferir archivos de una máquina a otra. Los archivos pueden ser binarios (cualquier tipo de archivo) o de texto. FTP se puede programar para que suceda un intercambio de archivo en una hora en particular (en la noche, por ejemplo, cuando los costos de comunicación disminuyen) y la transferencia se llevará a cabo sin la supervisión de un operador.

Ambos protocolos (FTP y TELNET) utilizan TCP. Un ejemplo de protocolo que utiliza a UDP es NFS o Sistema de Archivos en Red. NFS permite acceder a un servidor de archivos en forma transparente, es decir, los archivos remotos parecen locales.

Otro ejemplo es el estándar dominante para el correo electrónico, el *Protocolo Simple de Transferencia de Correspondencia* (*SMTP*), el cual permite enviar mensajes de texto. Si los mensajes contienen datos complejos como archivos de programa, mensajes de voz digitalizada o gráficas tienen que codificarse en una versión de texto simple antes de la transmisión. También se debe asegurar el mensaje lleve la dirección electrónica de la persona a la que se desea enviar.

6.1.3. NetBIOS

El Network Basic Input/Output System (NetBIOS), sistema de entrada/salida básico en red, fue desarrollado por Microsoft e IBM . Ofrece una interfaz de programación a alto nivel estándar para las redes punto a punto

NetBIOS es manejado por la capa de transporte y funciona fundamentalmente en el nivel de sesion, se encuentra entre la capa de presentación y la capa de sesión el modelo de referencia OSI (como se muestra en la Figura 6 5)

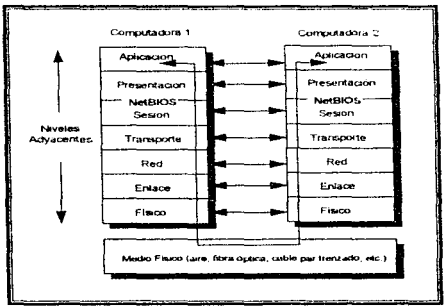


Figura 6.5. NetBIOS dentro del modelo de referencia ISO/OSI.

NetBIOS está incluido en cada tarjeta adaptadora de LAN para IBM PC en un chip ROM. Para otras tarjetas adaptadoras de red la ejecución del archivo NETBIOS.COM es un completo remplazo de NetBIOS en ROM Ambos ocupan 640K de memoria dentro de la PC en el espacio de direcciones de memoria.

La mayoría de los fabricantes en red ofrecen una implantación NetBIOS. Por ejemplo, el Programa de Red de IBM, LAN Mangar de Microsoft, LAN Server de IBM y otros fabricantes de LAN Manager utilizan la combinación de los protocolos NetBIOS, *Redirector* y *Server Message Block* (SMB).

El Redirector es una extensión al MS-DOS 3.1 y posteriores. Este componente de software de red recibe peticiones de datos y servicios por parte de un sistema operativo local, empleando un *SMB* (Bloque de Mensajes del Servidor) estructura los datos y los envía al servidor de la red, por medio de NetBIOS.

Fabricantes como Novell emulan las comunicaciones en el nivel de sesión de NetBIOS con sus propios productos para red. Las diversas implementaciones pueden presentar diferencias mínimas que llegan a ocasionar incompatibilidad entre los productos.

NetBIOS se distingue de los demás protocolos en que es bastante sencillo y, en un sentido estricto, es una interfaz más que una verdadera suite[1] de protocolos con una estructura en capas.

NetBIOS soporta el registro de nombres para las computadoras (se pueden dirigir mensajes a las estaciones con nombre, en lugar de a direcciones de red), el establecimiento y la terminación de sesiones, así como la transferencia de datos entre computadoras. NetBIOS es *bidireccional* (full duplex), lo que significa que los dos participantes en la sesión pueden enviar información simultáneamente. Estas funciones se están clasificadas dentro de los *servicios de NetBIOS* que se explican con detalle a continuación.

[1] *Suite* es un término que se utiliza a menudo en relación con los protocolos. Un suite de protocolos es un conjunto de éstos que funcionan juntos para proporcionar servicios completos de comunicación.

Servicios de NetBIOS

NetBIOS proporciona cuatro categorías de servicios de aplicación:

- ***Soporte de nombres***. El soporte de nombre permite que un adaptador LAN (tarjeta de red) sea distinguido de otros en su red respectiva por uno o más nombres, gracias a ellos los mensajes se dirigen a adaptadores específicos y permiten determinar qué adaptador a originado un mensaje. Un nombre puede ser un *nombre único* o un *nombre de grupo*. Si el adaptador registra su nombre como nombre único (utilizando el comando ADD NAME), entonces no puede operar otro adaptador sobre la red con el mismo nombre, fallará el intento de registrarlo. Si un adaptador es registrado como nombre de grupo (con el comando ADD GROUP NAME), entonces otro adaptador no puede usar el nombre como nombre único, el intento de registrarlo también fallará. Los nombres de grupo son utilizados para enviar mensajes a una colección de estaciones de trabajo. NetBIOS coloca los nombres en una tabla interna conocida como *la tabla de nombres NetBIOS*. Cada adaptador LAN tiene su propia tabla de nombres. Los sucesos de registros de nombres son reportados en esta tabla para las aplicaciones LAN, junto con un valor de un byte, este valor es un número sin signo conocido como *número de nombre*. El número de nombre es utilizado en varios comandos de NetBIOS asociados con el nombre. La primera entrada en la tabla es permanente y es asignada por el adaptador basada en el número serial interno. Todos los adaptadores LAN IBM tienen este número único el cual es asignado a 6-bytes, garantizando ser único para cada adaptador. Una vez utilizado un nombre puede ser borrado de la tabla a través del comando DELETE NAME, también el comando RESET ADAPTER de NetBIOS borra los nombres de la tabla (excepto el primer nombre). La tabla de nombres es temporal ya que está contenida dentro de la RAM, así que ejecutar un reset en el sistema (Ctrl-Alt-Del) o apagar la estación de trabajo tendrá el mismo efecto que el comando RESET ADAPTER. Cada nombre de red consiste de 16 caracteres.

- ***Soporte de datagramas***. Permite enviar pequeños mensajes a un nombre, nombre de grupo, o a la red entera, así como recibir mensajes desde un nombre o un nombre de grupo. Esto no proporciona una conexion y no garantiza la transferencia de mensajes o recepcion.

- ***Soporte de sesión***. NetBIOS crea una sesión de comunicación entre aplicaciones, ésta conexión es punto a punto (circuito virtual). Las aplicaciones pueden residir en la misma estacion de trabajo (sesión local) o en diferentes estaciones de trabajo (sesión remota), ésta última permite intercambiar información con otro usuario (nombre) sobre la red. Las sesiones se crean cuando una primera aplicación utiliza el comando LISTEN (escuchar) de NetBIOS, la aplicación esperará una llamada del nombre que se especifica en un campo llamado Call Name (si éste es asterisco recibirá llamadas de cualquier nombre). La segunda aplicación tendrá que invocar el comando CALL (llamar) especificando en su propio campo Call Name el nombre del adaptador de la primera aplicación. Cada aplicación entonces recibe una notificación de que la sesión se ha establecido y un valor sin signo de 1 byte (conocido como Local Session Number). Después de establecer una sesión, ambos lados pueden utilizar los comandos RECEIVE y SEND (o alguna de sus variantes) para transferir datos. Con el soporte de sesiones también se puede finalizar una sesión y obtener el estado de una sesión específica.

- ***Comandos generales***. Los comandos generales proporcionan servicios como inicializar el adaptador de interfaz de red (con el comando RESET), obtener su estado y otras funciones de control.

6.2. Caja de herramientas para NetBIOS

El sistema basico de entrada/salida en red (NetBIOS) es software de interface que permite a las computadoras comunicarse en una red de área local.

En el análisis del sistema se establecio a *Borland C++*(R) como lenguaje de programación del proyecto INGRESO, así como a NetBIOS como la base de la comunicación entre los sistemas de las estaciones de trabajo y el servidor. Para hacer esto posible se ha implementado una "*caja de herramientas para NetBIOS*", es decir, un módulo compilable que ofrece a los programas escritos en C una interface más comprensible a los servicios ofrecidos por NetBIOS.

Estudiando los conceptos de la programación de NetBIOS, se ha recopilado el material útil para los fines del proyecto. En éste capítulo se ilustran los principios y técnicas para el desarrollo de ésta caja de herramientas.

6.2.1. Empleo de los comandos de NetBIOS

Para acceder a NetBIOS, los programas hacen una interrupción de software y pasan datos que describen la operación que desean utilizar. A estos datos se les llama *Network Control Block* o NCB (bloque de control de red).

La aplicación utiliza un área de memoria de 64 bytes para construir un bloque de control NetBIOS (Ncb). La construcción de un Ncb consiste en llenar varios campos que son requeridos por el comando en particular que será usado. Los comandos son presentados a NetBIOS en la forma de un NCB.

Para evitar comportamientos impredecibles de la PC, todos los campos del Ncb deben ser inicializados con ceros antes de utilizar un comando. En Apéndice A (Listado A.1) se encuentra una función que se encarga de esto, la cual será mas entendible cuando se hable de la estructura del Ncb.

Después de que el Ncb es "llenado" , la aplicación utiliza el par de registros ES:BX para hacer una petición de interrupcion INT 5C, que es la interrupcion nativa de NetBIOS. Para IBM PCs, una metodo alternativo es utilizar la interrupción 2Ah de DOS, especificando un valor de 0400h o 0401h en el registro AX.

Un valor en AX de 0401h indica al DOS que no debe reintentar el comando si éste falla. Un valor en AX de 0400h indica al DOS que debería reintentar si el comando falla porque: el adaptador no tiene los recursos necesarios para completar el comando satisfactoriamente, el adaptador está ocupado y no puede manejar la petición, u otra estación de trabajo rechazó el intento de nuestra aplicación por entablar una sesión de comunicación con ésta.

La interrupción 2Ah es necesaria algunas veces para la coexistencia total con el Programa LAN PC IMB. Antes de utilizar esta interface, se debe verificar que la versión de DOS sea 3.1 o siguiente, y determinar si el Programa LAN PC IBM está instalado.

En el Apéndice A (Listado A.2), se muestra como utilizar la petición de interrupción utilizando el lenguaje de programación C.

6.2.2. Descripción breve de los campos Ncb

El Ncb está formado de 64 bytes con 13 campos y un área reservada de 14 bytes. La Tabla 6.1 muestra los campos del Ncb. La estructura del Ncb en lenguaje de programación C, es ilustrada en el apéndice A (Listado A.3)

Offset	*Nombre del Campo*	*Longitud en bytes*
+00	Command	1
+01	Return Code	1
+02	Local Session Number	1
+03	Name Number	1
+04	Buffer Address	4
+08	Buffer Length	2
+10	Call Name	16
+26	Name (Local)	16
+42	Receive Time Out	1
+43	Send Time Out	1
+44	Post Routine Address	4
+48	LANA Number	1
+49	Command Complete Flag	1
+50	Reserved Field	14

Tabla 6.1. Los campos Ncb.

A continuación se presenta una lista de campos NCB y la descripción de cada campo.

Command (NcbCommand). El campo Ncbcommand es un campo de 1 byte que contiene el *código de comando* para la operación deseada. Cada comando se ejecuta en cualquiera de los dos modos *wait* o *no-wait* (esperar o no-esperar). Si el bit de más alto orden del código de comando es cero se selecciona la opción *wait*. NetBIOS acepta la petición y regresa a la aplicación cuando el comando es completado. Si por el contrario es 1, se selecciona la opción *no-wait* y NetBIOS puede regresar el control a la aplicación aún cuando el comando no se haya ejecutado. El resto de los 7 bits especifican el comando que se desea ejecutar.

Return Code (NcbRetCode) El campo NcbRetCode es un campo de 1 byte que eventualmente contiene el codigo de regreso final de los comandos. Si este es cero después de la terminacion del comando, el comando fue completado satisfactoriamente. De otra forma, un problema fue detectado y regresará un código de error que corresponde a una explicacion (ver la tabla A.1 del apéndice A).

Local Session Number (NcbLsn). El campo NcbLsn es un campo de 1 byte que contiene el número de la sesión local asociada con un comando. NetBIOS asigna el valor del número de la sesión local en el orden 1,2,3,...254,2,3,4,254, y así sucesivamente. Los valores de cero y 255 nunca son asignados.

Name Number (NcbNum) El campo NcbNum es un campo de 1 byte que contiene el número de nombre de la tabla de nombres NetBIOS asociado con un comando. NetBIOS asigna el valor de el NcbNum de misma manera que el de NcbLSN. El primer número de entrada, NcbNum 1, es siempre el número del nombre del nodo permanente.

Buffer Address. El campo Buffer Address es un campo de 4 bytes que contiene un apuntador a memoria a un buffer de datos. En el caso de la PC IBM, los datos están en el formato OFFSET:SEGMENT.

Buffer Length (NcbLength). El campo NcbLength es un campo de 2 bytes indicando el tamaño del buffer apuntado por el campo Buffer Address.

CallName (NcbCallName). El campo NcbCallName es típicamente un campo de 16 bytes, pero no siempre, contiene un nombre remoto asociado con una petición. Los 16 bytes son importantes y utilizados. En algunos casos, la sesión puede ser local, en cuyo caso el nombre es un nombre local en lugar de un nombre remoto.

Name (NcbName). El campo NcbName (local) es un campo de 16 bytes, contiene un nombre local asociado con una petición. Los 16 bytes son utilizados. El primer caracter no puede tener un valor binario de cero o ser un asterisco. IBM reserva los valores de 00h a 1Fh para el caracter 16 y los valores "IBM" como los primeros tres caracteres de algún nombre.

Receive Time Out (NcbRto). El campo NcbRto es un campo de 1 byte utilizado con los comandos CALL y LISTEN. Este especifica el número de periodos de medios segundos que un comando RECEIVE (RECEIVE, RECEIVE-ANY) puede esperar antes de la terminacion del tiempo fuera y regresar un error. Especificando un valor de 00h indica que no hay tiempo fuera para comandos RECEIVE asociados con la sesión.

Send Time Out (NcbSto). El campo NcbSto es un campo de 1 byte utilizado con los comandos CALL y LISTEN. Este especifica el número de periódos de medios segundos que un comando SEND (SEND, SEND NO-ACK, CHAIN SEND NO-ACK) puede esperar antes de la terminación del tiempo fuera y regresar un error. Especificando un valor de 00h indica que no hay tiempo fuera para comandos SEND asociados con la sesión.

Post Routine Address. El campo Post Routine Address es un campo de 4 bytes conteniendo un apuntador a memoria para una rutina que es ejecutada cuando el comando se complete. NetBIOS solamente inspecciona este campo cuando el comando esta especificado con la opción *no-wait*, de otra forma, este es ignorado. En el caso de las PC IBM, los datos están en el formato OFFSET:SEGMENT.

LANA Number (NcbLanaNum). El campo NcbLanaNum es un campo de 1 byte indicando cual adaptador debe tomar el comando (en el caso de exista más de un adaptador). El adaptador primario es el adaptador de LAN cero, el adaptador alterno es adaptador de LAN número 1.

Command Complete Flag (NcbCmdCplt) El campo NcbCmdCplt es un campo de 1 byte que indica si un comando especificado con la opción *no-wait* fue completado, si el valor de este campo es FFh, el comando no fue completado. De otra forma, el campo contiene el código de regreso del comando final.

Reserved Field (NcbReservedArea). El campo NcbReservedArea es un área reservada de 14 bytes que NetBIOS puede utilizar para regresar información amplia de error. Los programas de aplicación podrían nunca utilizar este campo porque si no es utilizado correctamente el comportamiento de NetBIOS es impredecible.

6.2.3. Comandos NCB

NetBIOS ofrece 19 comandos para cubrir con los servicios del nivel de sesión de las aplicaciones distribuidas. Los comandos del NCB están divididos en cuatro categorías. La siguiente tabla 6.2 muestra los comandos de cada categoría.

Generales	*Soporte de nombre*	*Soporte de sesión*	*Soporte de mensajes*
RESET	ADD NAME	CALL	SEND DATAGRAM
CANCEL	ADD GROUP NAME	LISTEN	SEND BROADCAST DATAGRAM
STATUS (ADAPTER STATUS)	DELETE NAME	HANG UP	RECEIVE DATAGRAM
UNLINK		SEND	RECEIVE BROADCAST DATAGRAM
		CHAIN SEND	
		RECEIVE	
		RECEIVE ANY	
		SESSION STATUS	

Tabla 6.2. Comandos de NetBIOS.

Cada comando se especifica a NetBIOS por medio de un *código*, un número hexadecimal, que además de identificar al comando en particular, establece el modo de ejecución del comando. *wait* (esperar) o *no-wait* (no esperar). Por tal motivo la mayoria de los comandos tienen dos códigos, como se verá en su descripción

Cuando se emplea el código de comando de la opción wait se le está pidiendo a NetBIOS no regresar el control a la aplicación hasta que el comando se complete satisfactoriamente o el tiempo especificado para esperar en el campo NcbSto o en el campo NcbRto concluya. Por el contrario, con opción no-wait, NetBIOS puede regresa el control a la aplicación aún cuando el comando no se completo

En la caja de herramientas, un archivo de prototipos llamado NETBIOS.H contiene, entre otras definiciones, a los códigos de comando. El listado se puede observar en el Apéndice A (Listado A.4).

Haciendo un análisis, los comandos del soporte de datagramas no son útiles para los propósitos del proyecto, ya que no garantizan la transmisión de datos, y la longitud de los mensajes es máxima de 512 bytes.

A continuación se describirán únicamente a los comandos generales, los comandos de soporte de nombre y los comandos de soporte de sesión.

COMANDOS GENERALES

Los comandos generales permiten leer el status del adaptador y controlan otras funciones importantes.

RESET. Este comando inicializa el status local y limpia las tablas de sesión y nombre de NetBIOS (excepto el nombre del nodo permanente).
Código de comando: Hex 32 - opción esperar (wait)

CANCEL. Este comando pide que el comando que se encuentre en la dirección dada por el campo Buffer Address de la estructura NCB, sea cancelado.
Codigo de comando Hex 35 - opción esperar (wait)

STATUS (ADAPTER STATUS) Este comando proporciona información de la condición actual de un adaptador local o remoto por el nombre especificado en el campo NcbCallName. Si un asterisco (*) está especificado en el primer byte del campo NcbCallName, regresa la información para el adaptador local. Esta información es colocada en la dirección especificada por el campo Buffer Address, y la longitud del campo es actualizada para indicar el número de bytes de información recibida. El número de bytes requeridos es 60 +18(X), donde X es el número máximo de nombres para el adaptador.
Código de comando: Hex 33 - opción esperar (wait)
Hex B3 - opción no esperar (no-wait)

UNLINK. Este comando es utilizado solamente con un RPL (remote program load). Este comando se aplica solamente si una llamada a IBMNETBOOT fue hecha al momento de encender la computadora. La sesión con IMBNETBOOT es cancelada cuando este comando es utilizado.
Código de comando: Hex 70 - opción esperar (wait)

COMANDOS DE SOPORTE DE NOMBRE

Los comandos de soporte de nombre permiten a una computadora ser identificada en la red por un nombre. Un nombre puede ser un nombre único o un nombre de grupo. El número de nombres que cada implementación NetBIOS puede soportar varía. Un nombre de nodo permanente está siempre presente y consiste de 10 bytes de ceros binarios seguidos por el número ID único del adaptador. Este nombre de nodo permanente es único en toda la red. Este nombre no es mostrado como una entrada en la tabla local de nombres regresada por el comando STATUS.

ADD NAME. Este comando añade un nombre de 16 caracteres a la tabla de nombres. El nombre añadido es único no podrá ser añadido por otro en la red. El comando regresa el número del nombre en el campo NcbNum. Este número es utilizado en el soporte de mensajes y para comandos RECEIVE ANY.

Código de comando: Hex 30 - opción esperar (wait)
Hex B0 - opción no esperar (No-wait)

ADD GROUP NAME. Este comando añade un nombre de 16 caracteres a la tabla de nombres. Un nombre de grupo permite a un simple nodo comunicarse con muchos nodos. El nombre añadido no puede utilizarse como nombre único en la red, pero puede añadirse por muchos como nombre de grupo. El comando regresa el número del nombre en el campo NcbNum. Este número es utilizado en el soporte de mensajes y para comandos RECEIVE ANY.

Código de comando: Hex 36 - opción esperar (wait)
Hex B6 - opción no esperar (No-wait)

DELETE NAME Este comando borra un nombre de 16 caracteres de la tabla de nombres. Utilizando antes el comando HANG UP, el nombre será eliminado.

Código de comando

Hex 31 - opción esperar (wait)

Hex B1 - opción no esperar (No-wait)

COMANDOS DE SOPORTE DE SESIÓN

Los comandos de soporte de sesion permiten establecer una conexión lógica (sesión) en la red, enviar y recibir mensajes y sesiones, y obtener el status de la sesión. Los nombres son utilizados para estabilizar las sesiones, pero un número de 1 byte es utilizado para referirse a cada sesion estabilizada.

CALL. Este comando abre una sesión con otro nombre especificado en el campo NcbCallName utilizando el nombre local especificado por el campo NcbName. El nombre al que se llama debe tener un comando LISTEN pendiente para estabilizar la sesión. Se pueden entablar sesiones con un nombre local o remoto. Múltiples sesiones pueden establecerse con el mismo par de nombres. Todos los comandos SEND y RECEIVE para ésta sesión terminarán inmediatamente si no se concluyen durante los intervalos de tiempo fuera especificados. Los intervalos de tiempo fuera son dados en medios segundos (un valor de cero indicara que no hay tiempo fuera). Cuando el comando CALL es completado, un número de sesion local (NcbLsn) es asignado y utilizado para mantener la sesión.

Código de comando: Hex 10 - opción esperar (wait)

Hex 90 - opción no esperar (No-wait)

LISTEN. Este comando habilita una sesión para ser establecida con el nombre especificado en el campo NcbCallName. Si el campo NcbCallName tiene un nombre que comienza con asterisco (*), una sesión es estabilizada con algún nodo de la red que utiliza el comando Call para el nombre local especificado por el campo NcbName. Un comando LISTEN para un nombre específico tiene prioridad ante un comando LISTEN para cualquier nombre. Se pueden entablar sesiones con un nombre local o remoto. Múltiples sesiones pueden establecerse con el mismo par de nombres. Todos los comandos SEND y RECEIVE para esta sesión terminaran inmediatamente si no se concluyen durante los intervalos de tiempo fuera especificados.

Código de comando: Hex 11 - opción esperar (wait)

Hex 91 - opción no esperar (No-wait)

HANG UP. Este comando cierra la sesion indicada por el número de sesión local. Un 00 hexadecimal (comando completado) es regresado en un cierre normal y 0A hexadecimal (la sesión ha sido cerrada) o 08 hexadecimal (número de sesión local invalido) si la sesión ya ha sido cerrada o no existe. Cuando un comando HANG UP es recibido, todos los comandos locales pendientes RECEIVE son terminados y regresa el código de "la sesión ha sido cerrada" en el campo NcbRetCode.

Código de comando: Hex 12 - opción esperar (wait)

Hex 92 - opción no esperar (No-wait)

SEND. Este comando envía datos utilizando el número de sesión indicada en el número de sesión local (NcbLns). Los datos son tomados del buffer indicado por Address Buffer, para el número de bytes dado por NcbLength. El tamaño del buffer puede ser de 65,535 bytes en longitud. Cuando una sesión se cierra por la computadora remota, todos los comandos SEND pendientes sobre la sesión cerrada regresan un código "la sesión ha sido cerrada". Si un comando local HANG UP es utilizado con alguno de los comandos SEND pendientes, el comando HANG UP es llevado a cabo hasta que los comandos SEND

son completados o finalizados por error Si una sesión termina anormalmente, el código regresado corresponde a "sesión terminada anormalmente". Si el comando SEND expira en el tiempo fuera, la sesión inmediatamente termina y se regresa un código de "tiempo fuera del comando" . Si mas de un SEND esta pendiente, los datos son transmitidos en un orden "primero en entrar-primero en salir" (FIFO: first-in-first-out) dentro de la sesión.

Código de comando: Hex 14 - opción esperar (wait)

Hex 94 - opción no esperar (No-wait)

CHAIN SEND. Este comando envia datos por el número de sesión indicada en el NcbLsn. Los datos son tomados de los buffers para el número indicado de bytes. Dos buffers pueden unirse con este comando. Los datos en el segundo buffer son encadenados a los datos en el primer buffer y colocados como un único mensaje. El campo NcbCallName es utilizado para especificar la longitud y dirección del segundo buffer. La longitud está especificada en los primeros 2 bytes y la dirección en los siguientes 4 bytes.

Código de comando: Hex 17 - opción esperar (wait)

Hex 97 - opción no esperar (No-wait)

RECEIVE. Este comando recibe datos desde la sesión especificada. Si más de un comando RECEIVE esta pendiente, son pospuestos de acuerdo a la siguiente jerarquía: RECEIVE, RECEIVE-ANY-FOR-A-SPECIFIED-NAME, y RECEIVE-ANY-FOR-ANY-NAME Una vez que los comandos son ordenados por prioridad, se procesan en el orden: "el primero en entrar - el primero en salir". El valor de tiempo fuera de éste comando se especifica con los comandos LISTEN o CALL de la sesión.

Código de comando: Hex 15 - opción esperar (wait)

Hex 95 - opción no esperar (No-wait)

RECEIVE ANY. El comando RECEIVE-ANY recibe datos desde cualquier sesión asociada que envia datos con los comandos SEND o SEND CHAIN sobre una sesión asociada con un nombre local especificado. La aplicación local especifica el nombre local con el nombre NcbNum cuando utiliza este comando. Si el programa de aplicación fija en el campo NcbNum un FFh, el comando RECEIVE es para cualquier nombre remoto, más conocido como RECEIVE-ANY-FOR-ANY-NAME (se debe tener cuidado al utilizar este comando, ya que puede recibir mensajes para otros programas). Si más de un comando que puede recibir datos sobre una sesion está pendiente, éstos se procesan con la prioridad siguiente: RECEIVE, RECEIVE-ANY-FOR-A-SPECIFIED-NAME, y RECEIVE-ANY-FOR-ANY-NAME. Una vez que los comandos son ordenados por prioridad, se procesan en el orden: "el primero en entrar - el primero en salir". El valor de tiempo fuera de éste comando se especifica con los comandos LISTEN o CALL de la sesión.

Código de comando: Hex 16- opción esperar (wait)

Hex 96 - opción no esperar (No-wait)

SESION STATUS. El comando de SESION STATUS obtiene la condición actual de una o todas las sesiones asociadas con un nombre local. Si un asterisco está especificado como el primer byte de el campo NcbName, este comando obtiene la información para todos los nombres en la tabla de nombres NetBIOS. La mínima longitud del buffer valida es de 4 bytes.

Código de comando: Hex 34- opción esperar (wait)

Hex B4 - opción no esperar (No-wait)

La implementación de un comando como ya se había dicho, primeramente requiere de la inicialización a ceros de los campos pertenecientes a la estructura Ncb. Inmediatamente después se especifica el comando en el campo NcbCommand, se llenan los campos necesarios para el comando en particular y se efectúa la petición de éste a NetBIOS por medio de una interrupción. La interrupción que se utiliza en la caja de herramientas es la 5Chex.

CAPÍTULO 7

CAPA DE APLICACIÓN DEL PROYECTO INGRESO

7.1. DESCRIPCIÓN DETALLADA DEL SISTEMA

De acuerdo al modelo de referencia OSI, la capa de aplicación proporciona y controla el ambiente en el cual la aplicación adquiere acceso a todos los servicios proporcionados por el sistema.

7.1.1. Sistema Cliente

La máquina Cliente como máquina solicitante de servicios de la máquina Servidor, cuenta con una interfaz de usuario gráfica compatible con Windows 3.11® y Windows95®. La "interfaz de usuario" es la puerta hacia aplicaciones de software interactivas. La pantalla inicial del sistema Cliente dispone de tres comandos (figura 7.1).

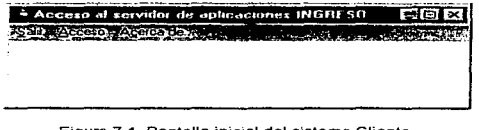


Figura 7.1. Pantalla inicial del sistema Cliente.

Es posible seleccionar cualquier comando de menú con el mouse o con el teclado (presionando ALT+ la letra que aparece subrayada en el nombre del menú).

El comando "Acerca de..." únicamente muestra información del programa (figura 7.2), el comando "Salir" termina el programa Cliente.

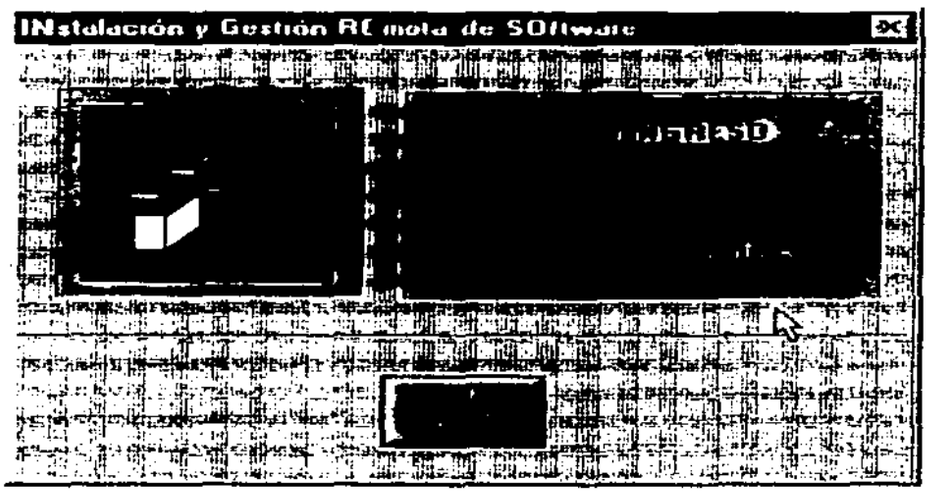


Figura 7.2. Comando "Acerca de".

Por último, la opción "Acceso" del menú despliega una caja de diálogo (figura 7.3), en la cual el usuario deberá introducir su clave de acceso y clave de proyecto, para poder establecer una sesión con el Servidor INGRESO por medio de NetBIOS[1].

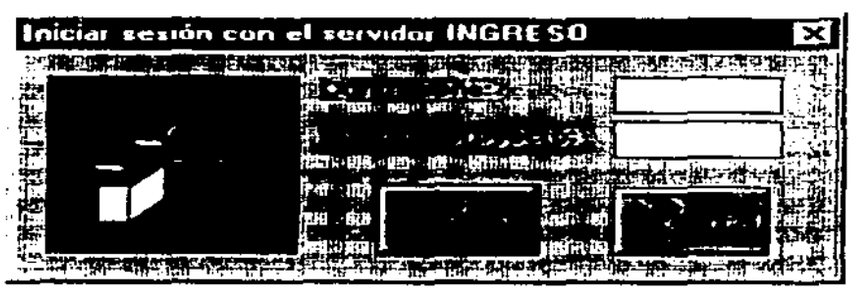


Figura 7.3. Comando "Acceso".

[1] Antes de ejecutar el sistema Cliente debe ser cargado el protocolo de transporte NetBIOS, el cual es el "lenguaje" de comunicación con el Servidor. Futuras versiones podrán operar sobre otro tipo de transporte.

Las dos entradas son indispensables, si el boton 'OK' es presionado, y falta alguna entrada, no se permitirá proseguir (figura 7.4). Por cuestiones de seguridad, la contraseña de usuario no será desplegada en pantalla, cada caracter será sustituido por un asterisco (*).

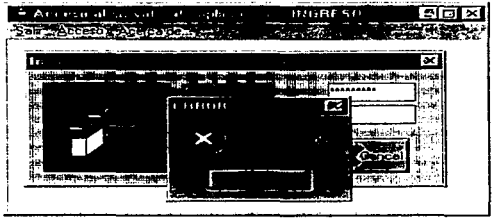


Figura 7.4. Error: Falta de datos.

Los caracteres introducidos serán validados[2], antes de intentar una conexión con el Servidor. Si alguna de las entradas contiene caracteres inválidos y se trata de cerrar la caja con ayuda del botón 'OK', la entrada incorrecta será borrada después de desplegar un mensaje de error (figura 7.5) y la caja permanecerá abierta esperando una entrada correcta.

Si se introducen datos y se cierra la caja de diálogo empleando el botón 'CANCELAR', se ignorarán las modificaciones introducidas a los controles de edición.

[2] Los caracteres válidos son:
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-"

Si los caracteres son válidos y el botón 'OK' es presionado, se pasará a "la capa de presentación" del modelo de referencia OSI. La cual es responsable de una transformación[3] de los datos que son transmitidos en una sesión. En este caso, la contraseña de usuario y el número de proyecto serán encriptados, es decir codificados, como una medida de seguridad.

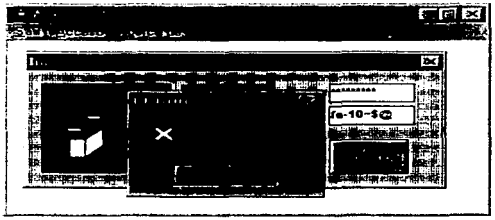


Figura 7.5. Error: Caracteres incorrectos.

A continuación se tratará de establecer una sesión con el Servidor INGRESO por medio de NetBIOS. NetBIOS es manejado por la capa de transporte y funciona fundamentalmente en el nivel de sesión, se encuentra entre la capa de presentación y la capa de sesión el modelo de referencia OSI.

El cursor se cambiará por el reloj de arena mientras se establece una sesión con el Servidor. Si NetBIOS detecta un error, se desplegará en pantalla. Los posibles errores se muestran en el apéndice B, lista B1.

[3] Algunas de las posibles transformaciones son: compresión de texto, codificación, decodificación, conversión de formatos de archivo, etc.

En el caso de que un error de NetBIOS impida establecer una sesión y se cierre la caja de diálogo, si se vuelve a abrir, los datos seguirán intactos. Esperando que se intente nuevamente una conexión con el servidor.

Si la sesión se establece, el Sistema Cliente recibirá una lista de los paquetes con licencias disponibles desde el Servidor. Esta lista será mostrada en una caja de diálogo tipo "List Box" en la cual el usuario puede seleccionar el nombre del paquete a instalar (Figura 7.6). Si el botón 'OK' es presionado, el sistema Cliente intentará una conexión lógica al directorio del Servidor que contenga el paquete elegido. Si ocurre un error, se desplegara el mensaje correspondiente, los posibles errores al intentar una conexión lógica se listan en el apéndice B, lista B2. La caja de diálogo permanecerá abierta esperando se intente nuevamente la conexión o seleccione otro software.

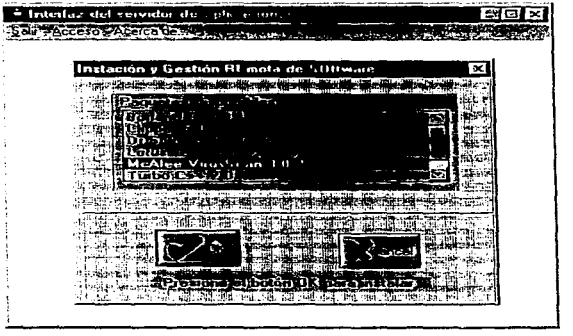


Figura 7.6. Lista de Software.

Una vez que la conexion logica se haya efectuado, el sistema Cliente ejecutará remotamente el archivo de instalación respectivo. Si el archivo de instalación no puede ejecutarse, el error se mostrará en pantalla, los posible mensajes por desplegar se muestran en el apendice B, lista B4.

Si la instalación se llevo a cabo, se generarán dos reportes de salida, uno para mostrar los cambios efectuados en los archivos WIN.INI y SYSTEM.INI, y otro para reportar los archivos añadidos o modificados con la instalación. Los dos son archivos temporales que se pueden visualizar con cualquier editor de texto. Los nombres de estos archivos serán notificados al usuario. Estos archivos le serán útiles en el caso de que quiera desinstalar por si mismo el paquete.

Por ejemplo, si el paquete elegido fuera McAfee VirusScan 3.0.2 el reporte de archivos añadidos o modificados sería el mostrado en el listado 7.1.

```
**********************************************
    REPORTE DEL SERVIDOR INGRESO
    Archivos nuevos o modificados
    hora: 2:45:00.53 pm
    Fecha: July 2, 1997
**********************************************


:\AUTOEXEC.BAT   274
:\BOOTLOG.PRV    19596
:\AUTOEXEC.003   205
:\AUTOEXEC.M01   206
:\WINDOWS\SYSTEM.INI        1938
:\WINDOWS\WIN386.SWP        7340032
:\WINDOWS\SECCAST.ICO       4710
:\WINDOWS\TMPDELIS.BAT  122
:\WINDOWS\WININIT.BAK   512
:\WINDOWS\SYSTEM\MCAFECOM.DLL 117664
:\WINDOWS\SYSTEM\MCKRNL.VXD     12415
:\WINDOWS\SYSTEM\S95EXT.DLL     78848
:\WINDOWS\SYSTEM\MCSCAN32.VXD 131201
:\WINDOWS\SYSTEM\MCUTIL.VXD     24703
:\WINDOWS\SYSTEM\VSHIELD.VXD    36992
:\WINDOWS\SYSTEM\FFASTLOG.TXT   10624
:\WINDOWS\HELP\SCANEXT.HLP      9083
```

Listado 7.1. Reporte de archivos añadidos o modificados .

```
\WINDOWS\HELP\SCRSCAN.HLP          8836
\WINDOWS\HELP\VIRUSCAN.CNT         1149
\WINDOWS\HELP\VIRUSCAN.HLP         31622
\WINDOWS\HELP\VSHLDCFG.HLP         12015
\WINDOWS\TEMP\~INSTALL.TMP         42315
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\VIRUSS~1.LNK          476
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\VIRUSS~2.LNK          324
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\VSHILL~1.LNK          484
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\CREATE~1.LNK          480
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\SCREEN~1.LNK          480
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\WHATS~1.LNK           359
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\README~1.LNK          370
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\MCAFEE~1.LNK          359
\WINDOWS\MENUIN~1\PROGRA~1\MCAFEE~1\VSNINST~1.LNK         536
\ARCHIV~1\MCAFEE\VIRUSS~1\DEISLI.ISU        17352
\ARCHIV~1\MCAFEE\VIRUSS~1\RESELLER.TXT      29142
\ARCHIV~1\MCAFEE\VIRUSS~1\WHATSNEW.TXT      38821
\ARCHIV~1\MCAFEE\VIRUSS~1\PACKING1.LST      2348
\ARCHIV~1\MCAFEE\VIRUSS~1\WCMDRSRL.INI      1793
\ARCHIV~1\MCAFEE\VIRUSS~1\VALIDATE.EXE      16326
\ARCHIV~1\MCAFEE\VIRUSS~1\WCMDR.INI         2150
\ARCHIV~1\MCAFEE\VIRUSS~1\WCMDR.EXE         21680
\ARCHIV~1\MCAFEE\VIRUSS~1\CLEAN.DAT         127400
\ARCHIV~1\MCAFEE\VIRUSS~1\NAMES.DAT         347633
\ARCHIV~1\MCAFEE\VIRUSS~1\SCAN.DAT          338471
\ARCHIV~1\MCAFEE\VIRUSS~1\SCRSCANP.DLL      71680
\ARCHIV~1\MCAFEE\VIRUSS~1\SCRSCANR.DLL      51760
\ARCHIV~1\MCAFEE\VIRUSS~1\MCALYZE.DAT       682787
\ARCHIV~1\MCAFEE\VIRUSS~1\DEFAULT.VSC       745
\ARCHIV~1\MCAFEE\VIRUSS~1\SCRSCAN.EXE       84480
\ARCHIV~1\MCAFEE\VIRUSS~1\SCAN.EXE          262658
\ARCHIV~1\MCAFEE\VIRUSS~1\SCANPM.EXE        558919
\ARCHIV~1\MCAFEE\VIRUSS~1\SLFSRVSW.DLL      20480
\ARCHIV~1\MCAFEE\VIRUSS~1\FANFORM.TXT       2486
\ARCHIV~1\MCAFEE\VIRUSS~1\DPMI16.DLL        3096
\ARCHIV~1\MCAFEE\VIRUSS~1\GETREPLY.EXE      2132
\ARCHIV~1\MCAFEE\VIRUSS~1\FDISK32.INF       14122
\ARCHIV~1\MCAFEE\VIRUSS~1\LDISK32.EXE       25088
\ARCHIV~1\MCAFEE\VIRUSS~1\FDISKDOS.BAT      594
\ARCHIV~1\MCAFEE\VIRUSS~1\FDISKDOS.PIF      967
\ARCHIV~1\MCAFEE\VIRUSS~1\FDISKDOS.IN       5
\ARCHIV~1\MCAFEE\VIRUSS~1\MCCOD32.DLL       147968
\ARCHIV~1\MCAFEE\VIRUSS~1\SCAN32.EXE        393360
\ARCHIV~1\MCAFEE\VIRUSS~1\SLFSRVSW.EXE      296448
\ARCHIV~1\MCAFEE\VIRUSS~1\VSECOM.DLL        111104
\ARCHIV~1\MCAFEE\VIRUSS~1\MCAFEE.SCT        139264
\ARCHIV~1\MCAFEE\VIRUSS~1\CHKVXD.EXE        6989
\ARCHIV~1\MCAFEE\VIRUSS~1\DPMI32.DLL        22016
\ARCHIV~1\MCAFEE\VIRUSS~1\DUNZIP32.DLL      99840
\ARCHIV~1\MCAFEE\VIRUSS~1\DZIP32.DLL        120832
\ARCHIV~1\MCAFEE\VIRUSS~1\MCKRNL32.DLL      63488
\ARCHIV~1\MCAFEE\VIRUSS~1\MCSCAN32.DLL      182272
\ARCHIV~1\MCAFEE\VIRUSS~1\MCUTIL32.DLL      163840
\ARCHIV~1\MCAFEE\VIRUSS~1\SHUTIL.DLL        111616
\ARCHIV~1\MCAFEE\VIRUSS~1\MCALYZE.DLL       25088
\ARCHIV~1\MCAFEE\VIRUSS~1\DEFAULT.VSH       952
\ARCHIV~1\MCAFEE\VIRUSS~1\VSHCFG32.EXE      71680
\ARCHIV~1\MCAFEE\VIRUSS~1\VSHWIN32.EXE      132608
```

Listado 7.1. Reporte de archivos añadidos o modificados (continuación) .

7.1.2. Sistema Servidor

El sistema del Servidor INGRESO, incluye siete opciones en el menú principal (Figura 7.7).

La primera opción despliega un submenú descendiente, el comando "Esperar llamada", prepara al sistema para esperar peticiones de instalación, confirmaciones de que una instalación se efectuó con éxito o la recuperación de una licencia. Esta opción bloquea cualquier otra aplicación. La opción "Salir" termina el programa servidor, solamente si NetBIOS no esta activo.

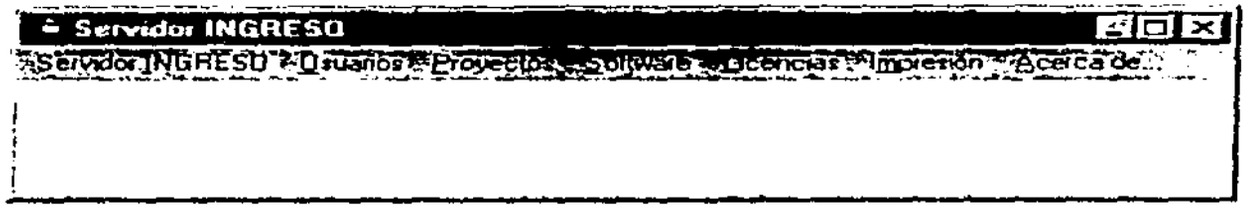


Figura 7.7. Menú Servidor INGRESO.

El menú de "Usuarios" permite administrar los datos relacionados con los usuarios autorizados para el uso del sistema (Figura 7.8). La tabla encargada de almacenar los datos de usuario es accesada por medio de cajas de diálogo que permiten consultar, añadir, modificar o eliminar registros.

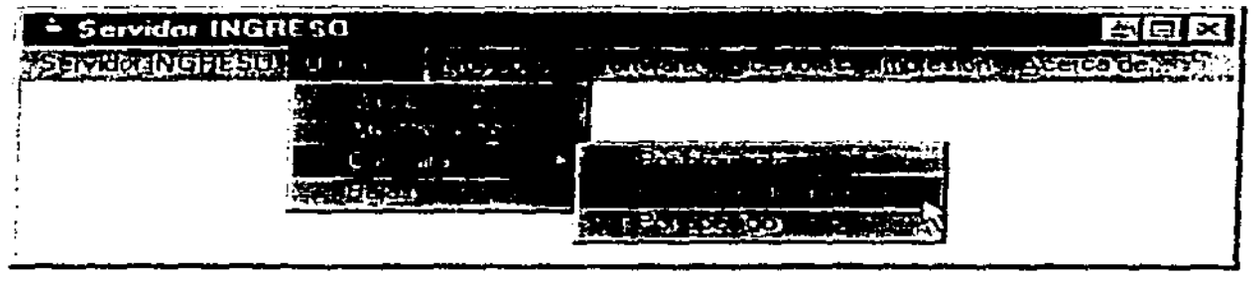


Figura 7.8. Menú Usuarios.

La opción "Altas" del menú de Usuarios, muestra una caja de diálogo que facilita capturar la información que identifica a un nuevo usuario (Figura 7.9).

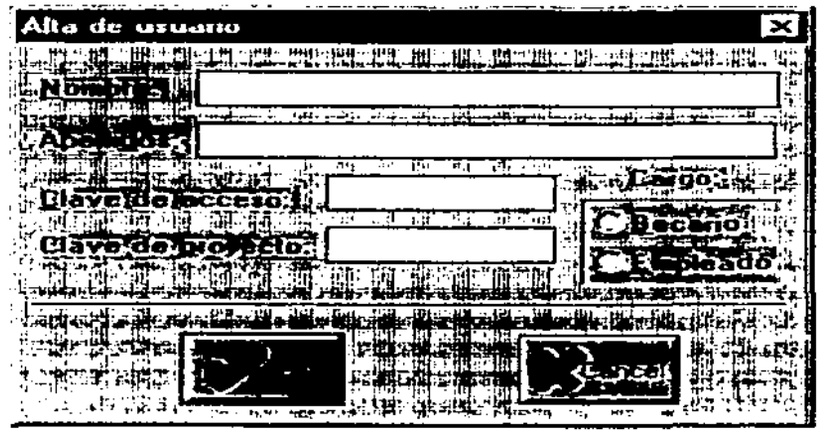


Figura 7.9. Alta de usuario.

Si el botón OK es presionado, se validará cada caracter que fue introducido si existe un caracter inválido[4], se mostrará en pantalla (Figura 7.10).

También se verificará que no falte ningún dato, de ser así se desplegará un mensaje semejante al de la Figura 7.4.

Si los caracteres son válidos, se confirmará que la clave de proyecto introducida exista en la base de datos, si no es así, se desplegará un error haciendo una petición de que primero sea dado de alta el proyecto.

Si el proyecto existe, se guardará un nuevo registro con los datos introducidos. El botón CANCEL cerrará la caja y no efectuará ningún cambio en la base de datos.

[4] Los caracteres válidos para la clave de acceso y la clave de proyecto son:
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789-"
Los caracteres válidos para el nombre y el apellido son todos los caracteres del abecedario en mayúsculas y minúsculas incluyendo ancentos.

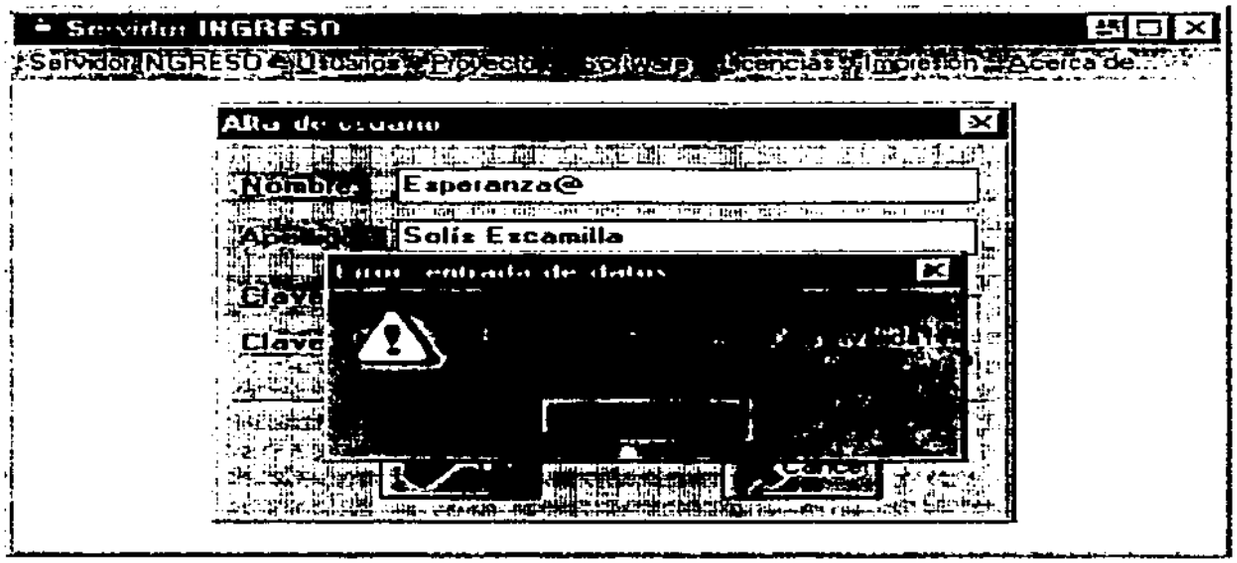


Figura 7.10. Alta de usuario: entrada inválida.

Si se desea consultar a todos los usuarios dados de alta, se puede elegir la opción "Consultas" del menú del Usuarios, pudiendo elegir entre tres tipos, ordenadas por: clave de proyecto, clave de usuario o apellidos (Figura 7.8).

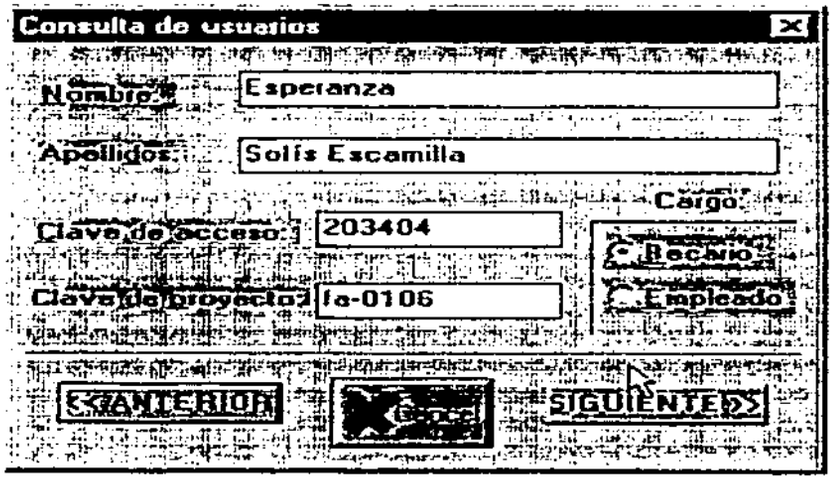


Figura 7.11. Consulta de usuarios.

La caja de diálogo para las tres consultas es la misma, lo único que cambia es el orden de los registros (Figura 7.11). Los botones 'ANTERIOR' y 'SIGUIENTE' permiten moverse de un registro de usuario a otro.

Si existen errores al capturar los datos de un usuario o existe la necesidad de modificar algún campo, la opción "Modificaciones" del menú Usuarios permite editar un registro y guardar los cambios (Figura 7.12). Los botones '<<' y '>>' tienen la misma función que los botones 'ANTERIOR' y 'SIGUIENTE' respectivamente.

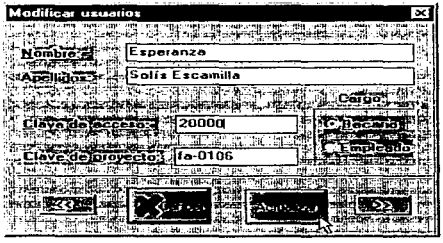


Figura 7.12. Modificación de usuarios.

Si el botón 'Actualizar' es presionado se pedirá una confirmación de que en realidad se desea guardar los cambios (Figura 7.13). Al elegir el botón 'OK' las modificaciones serán permanentes, si se presiona el botón 'CANCEL' todo cambio será ignorado.

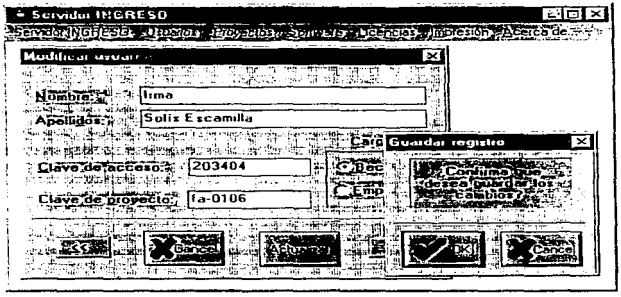


Figura 7.13. Modificación de usuarios: confirmación.

De manera similar, la opción "Bajas" del menú Usuarios desplegara una caja de diálogo que eliminará los datos de un usuario si se presiona el botón 'Eliminar' y es confirmada la petición (Figura 7.14).

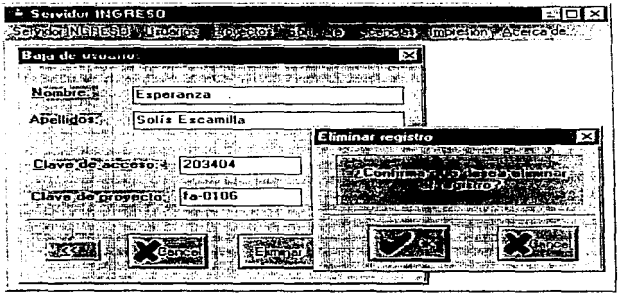


Figura 7.14. Eliminación de usuarios.

El menú de Proyectos incluye las mismas opciones que el menú de Usuarios, la única diferencia es que las consultas únicamente son ordenadas por clave de proyecto (Figura 7.15).

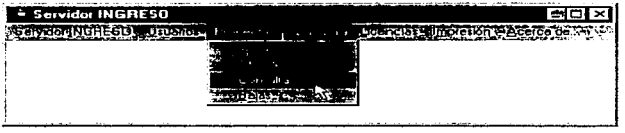


Figura 7.15. Menú Proyectos.

Todas las posibles operaciones a las base de datos proyectos y software, siguen la misma lógica que se aplicó a la base de datos de usuarios. La caja de diálogo para dar de alta a los proyectos se muestra en la figura 7.16.

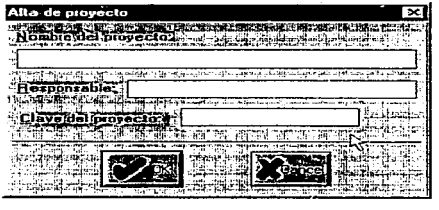


Figura 7.16. Alta de proyectos.

En la consulta de proyectos, si algún usuario fue dado de alta con la clave de proyecto consultada, su clave de usuario será mostrada dentro de una lista llamada "Claves de usuarios asignados" (Figura 7.17).

Figura 7.17 Consulta de proyectos

Si la clave de usuario es seleccionada con el mouse presionándolo con un doble 'click', se mostrarán los datos del usuario asociados a esa clave (Figura 7.18).

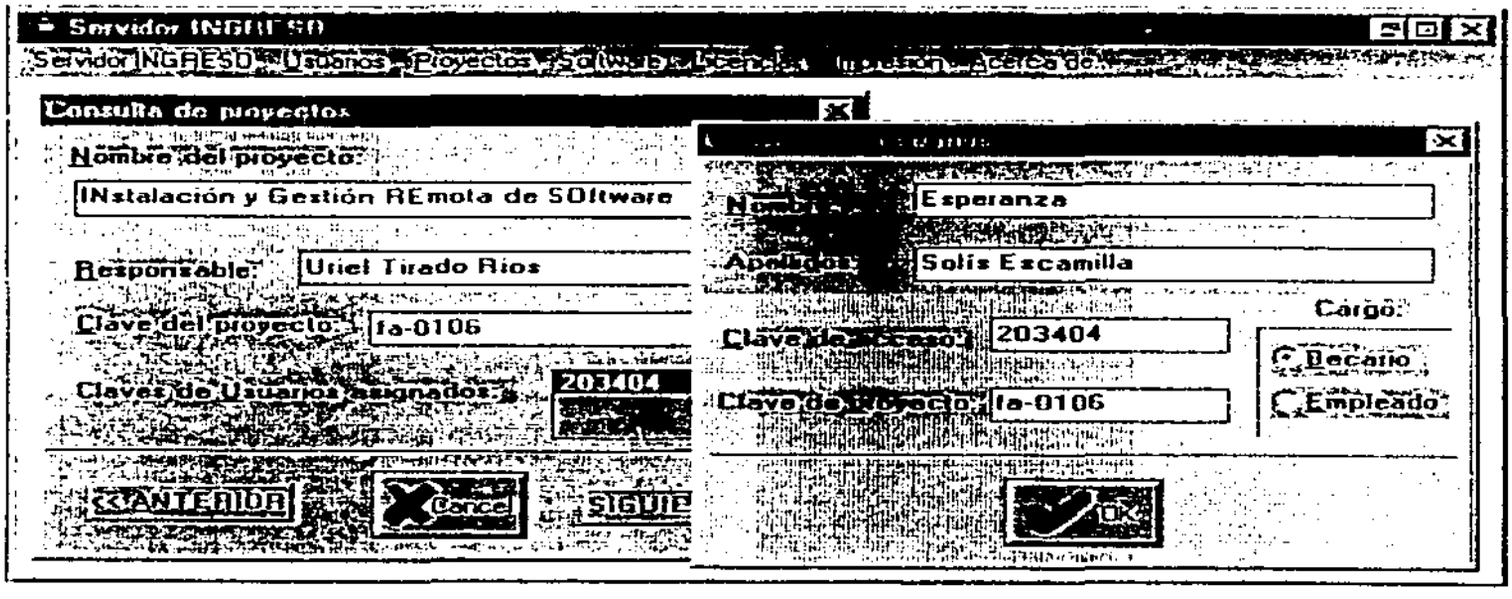


Figura 7.18. Consulta de proyectos: usuarios asignados.

Las cajas de diálogo para eliminar y modificar registros de proyectos, pedirán una confirmación para llevar a cabo la acción, de igual forma que para los registros de usuarios.

En el caso de que existan usuarios asignados a un proyectos que se quiera eliminar, la acción será denegada. Primero se tendrá que dar de baja a los usuarios o modificar la clave de proyecto a la cual estan asignados.

El menú de software cuenta con las mismas opciones que el menú de proyectos. Las consultas a Software son ordenadas únicamente por la clave de software (Figura 7.19).

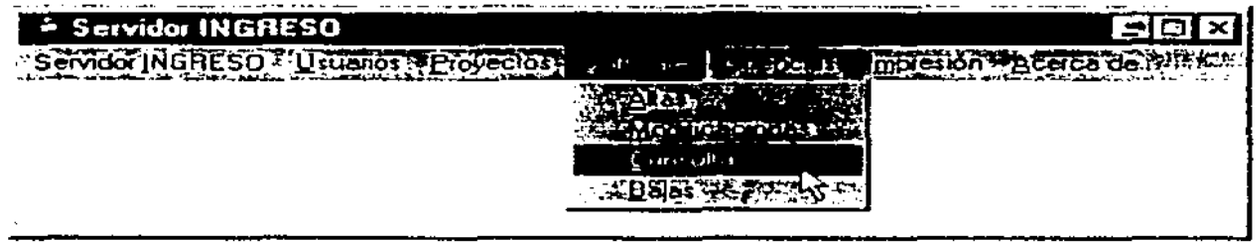


Figura 7.19. Menú software.

Las cajas de diálogo para dar de "Alta" software y hacer "Modificaciones" cuentan con un botón llamado 'EXAMINAR', el cual desplegará una caja de diálogo que permite buscar la carpeta (directorio) y el archivo correspondiente de instalación, al presionar el botón 'OK' se llenan los campos Directorio y Archivo sin necesidad de teclear. Esta caja cuenta con algunos botones de radio para elegir el tipo de archivos enlistados (Figura 7.20).

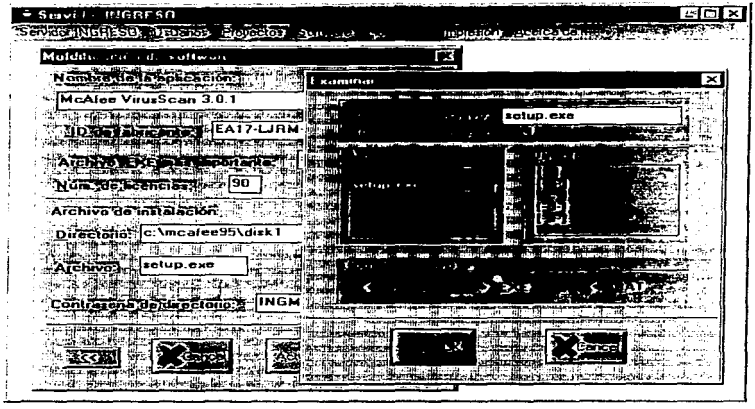


Figura 7.20. Modificación software: Examinar.

Al dar de alta a un paquete automáticamente se actualizará el control de licencias disponibles, esto puede ser consultado desde el menú de Licencias en la opción "Disponibles" (Figura 17.21), la cual mostrará la clave del software con el número de licencias contratadas y el número de licencias disponibles para instalar (Figura 17.22).

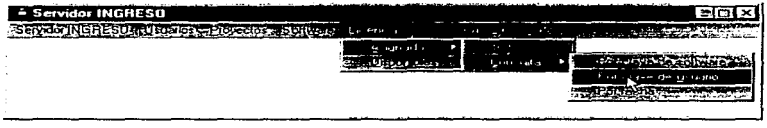


Figura 7.21. Menú Licencias.

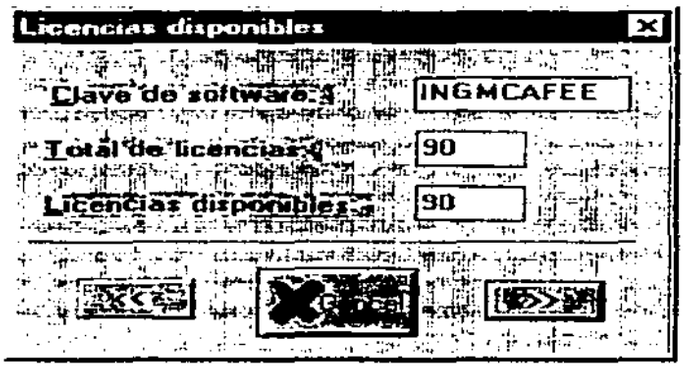


Figura 7.22. Licencias disponibles.

El número de "licencias disponibles" será actualizado cada vez que se lleve a cabo una instalación o se verifique una desinstalación. Los registros de ésta opción son ordenados por el campo "clave de software".

Las instalaciones confirmadas podrán verificarse en el menú Licencias en la opción "Asignadas". Esta opción despliga un menú descendiente con las opciones de consultas y bajas. Las consultas pueden hacerse por: clave de software, clave de usuario o fecha de instalación (figura 17.22). La caja de diálogo para las consultas se muestra en la figura 17.23.

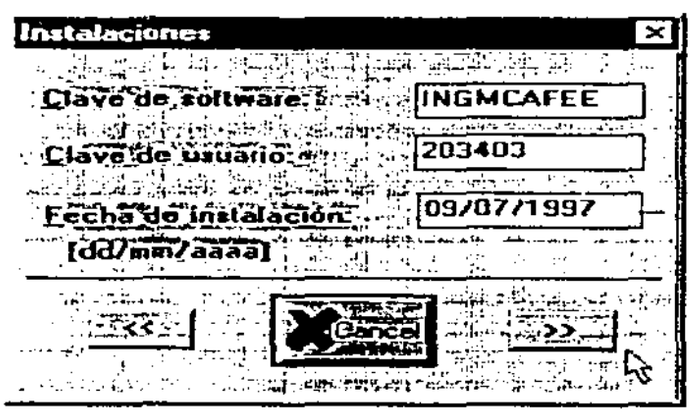


Figura 7.23. Instalaciones.

El administrador de la base de datos podrá dar de baja instalaciones, si una instalación es dada de baja, el número de licencias disponibles se actualizará.

Para la elaboración de la base de datos, se utilizo una libreria en C++ llamada CSDB, la cual consiste de una serie de clases que incorporan las herramientas necesarias para el desarrollo y manejo de base de datos, sin necesidad de utilizar un DBMS (Sistema Administrador de Bases de Datos). Todos los archivos de las bases de datos tienen un formato compatible con dBASE.

Por último, el menú "Imprimir" tiene dos opciónes "Configuración" e "Imprimir...", ésta última opción despliega un submenú que muestra las posibles impresiones.

Figura 7.24. Menú Impresión.

Las impresiones son reportes sencillos de información contenida en las bases de datos. Por Ejemplo, una impresión de "Lista de Software" se imprimiria de forma semejante al mostrado en el Listado 7.2.

SERVIDOR INGRESO

LISTA DE SOFTWARE

Fecha: July 21, 1997

CLAVE	LICENCIAS	LIC.LIBRES	NOMBRE
INGBC2	5	5	Borland C++ 3.1
INGCLIPPS	6	6	Clipper 5.0
INGDRSOL	15	15	Dr Solomon's Windows
INGFLW21	6	5	Freelannce Graphics 2.1 for Windows
INGMCAFEE	60	58	McAfee VirusScan 3.0.2
:	:	:	:

Listado 7.2. Lista de Software.

Al elegir del menú "Imprimir" la opcion "Configuración" aparece una caja de diálogo mediante la que se pueden elegir entre las opciones soportadas por el controlador de la impresora por omision. En la Figura 7.25 se muestra una caja de diálogo correspondiente a una impresora por omisión HP LaserJet IIISi.

Cada controlador de impresora cuenta con su propia caja de dialogo, dependiende de las opciones que ofrezca cada impresora. Al exhibir la caja de diálogo instalada para la impresora por omision, el Administrador de Impresión Windows se carga automáticamente en la memoria para preparar la impresión.

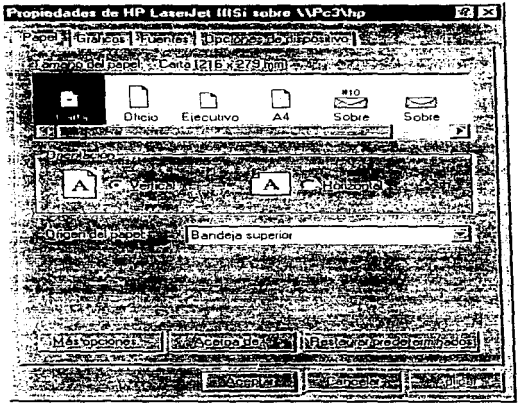


Figura 7.25. Configuración de Impresora.

CONCLUSIONES

Los trabajos realizados en esta tesis permitieron la realización de un prototipo para el proyecto INGRESO. En una etapa posterior toda experiencia será aprovechada para el desarrollo de un sistema más ambicioso.

El modelo Cliente/Servidor, en el cual esta basado el sistema, proporciona muchas ventajas, que se pretenden explotar

En primer lugar, se pueden integrar diferentes tipos de equipos y sistemas operativos en un ambiente único de procesamiento. Bajo este modelo, la red determina las necesidades del usuario y sólo le transmite información realmente útil. La implantación de esta arquitectura promueve el uso de sistemas abiertos dado que tanto clientes como servidores corren en diferentes plataformas de hardware y software, lo cual permite que las empresas compren productos de diferentes proveedores sin interferir en el desempeño de las demás aplicaciones y equipos de la red

Actualmente, se ha encontrado la nueva forma de aprovechar el modelo cliente/servidor : *Intranet*. Esta corriente del mercado viene con fuerza y propone utilizar la misma tecnología de Internet pero con información hacia dentro de las empresas. Esto significa poner sobre una red pública una red privada virtual, sin invertir en recursos adicionales. La información que se puede poner en Intranet son todas aquellas cosas tan cotidianas de las organizaciones y tan difíciles de actualizar y distribuir entre sus integrantes.

Se ha pensado que el proyecto INGRESO podría trabajar junto con otros sistemas administradores de recursos informáticos dentro de Intranet, la cual, además de ser un medio muy vivo de comunicación corporativa, es realmente útil para poner a disposición de todos los empleados la información de uso cotidiano, con un esquema de seguridad que garantiza integridad y confiabilidad de la información.

El problema con el modelo cliente/servidor, de tener que desarrollar un cliente, su código y un servidor con sus propios códigos, queda resuelto en Internet e Intranet, donde el cliente es universal y es un browser (buscador) con un lenguaje también universal (HTLM), al igual que el servidor.

También existe la tendencia de que la arquitectura cliente/servidor se vea impulsada y complementada por nuevos estándares del mercado, como podría serlo Java, el nuevo lenguaje de programación especialmente creado para Internet. Java marca una tendencia importante en cuanto a desarrollos y hoy día es soportada por la mayor parte de los proveedores de soluciones.

Se dice que la era de las computadoras ya quedó atrás y que ahora el paradigma cliente/servidor va a evolucionar a un modelo más amplio al que se le podrá llamar *cliente/red*. Esto consistirá en accesar a todos los servicios que proporciona una red desde quizá un teléfono celular, una agenda electrónica o cualquier otra forma de acceso remoto, totalmente independiente de los protocolos.

Sin embargo, las tendencias actuales son todavía cliente/servidor e Intranet. En todo el proceso de búsqueda de información de Internet, se cumple el proceso de cliente y servidor, siendo a veces un servidor cliente de otro.

Invariablemente, como se vea, siempre se terminará usando una arquitectura cliente/servidor.

Ante todo esto, futuras versiones del proyecto INGRESO tendrán que utilizar el protocolo de transporte de Internet: TCP-IP. El nuevo sistema se orientará a Intranet, sobre todo por los reducidos gastos que representa el implantarla y por los beneficios corporativos que ésta puede generar en la empresa. Con esta nueva estructura, el sistema podría implementarse no solo a nivel departamental, sino institucional. De esta forma, la base de datos tendrá que controlar además de los datos de usuarios, proyectos y software, el equipo, la coordinación, el departamento, la extensión, etc., en donde se encuentran ubicados.

APÉNDICE A

Caja de herramientas NetBIOS

Listado A.1. Inicialización de los campos de un Ncb.

```
void ClearNcb(struct Ncb *NcbPtr)
{
        int i;
        LPSTR CharPtr = (LPSTR) NcbPtr;
                for ( i = 0; i < sizeof(ZeroNcb); i++)
                        *CharPtr++ = '\x00';
}
```

Listado A.2. Petición de interrupción directa a NetBIOS.

```
#define USGC unsigned char

#define NetbiosInt21FunctionCode ((USGC) 0x2A)
#define NetbiosInt5C             ((USGC) 0x5C)

void  NetbiosRequest(struct Ncb *NcbPtrNear)
{
        struct SREGS SegRegs;
        union REGS InRegs,OutRegs;      /* definidos en dos.h */
        struct Ncb far *NcbPtrFar = (struct Ncb far *) NcbPtrNear;

        NcbPtrNear -> NcbLanaNum = 0;    /* adaptador primario */
        segread(&SegRegs);

        SegRegs.es = FP_SEG(NcbPtrFar);
        InRegs.x.bx = FP_OFF(NcbPtrFar);

        int86x(NetbiosInt5C, &InRegs, &OutRegs, &SegRegs); /* interrupción nativa NetBIOS */
}
```

Listado A.3. Una estructura del Ncb (bloque de control en red).

```
#define USGC unsigned char
#define USGI unsigned int
#define USGL unsigned long

struct Ncb
{
        USGC NcbCommand;            /* código de comando */
        USGC NcbRetCode;            /* código de regreso */
        USGC NcbLsn;                /* número local de la sesión */
        USGC NcbNum;                /* número asociado a un nombre */

        LPSTR NcbBufferOffset;      /* I/O buffer offset */
        USGI  NcbBufferSegment;     /* I/O buffer segment */

        USGI NcbLength;             /* longitud de datos en I/O buffer */

        char NcbCallName[16];       /* nombre remoto para CALL */
        char NcbName[16];           /* nombre local del adaptador de red */

        USGC NcbRto;                /* tiempo fuera RECEIVE en unidades de 1/2 segundos */
        USGC NcbSto;                /* tiempo fuera SEND en unidades de 1/2 segundos */

        LPSTR NcbPostRtnOffset;     /* offset of post routine */
        USGI  NcbPostRtnSegment;    /* segment of post routine */

        USGC NcbLanaNum;            /* número de adaptador a ejecutar un cmd */
        USGC NcbCmdplt;             /* 0xFF      comando pendiente, si no completo */

        char NcbReservedArea[14];   /* area de trabajo para la tarjeta de red */
        } ZeroNcb,                  /* prototipo NCB para cálculos sizeof */
```

00h	Buen regreso, ejecución satisfactoria
01h	Longitud del buffer inválida
03h	Comando inválido
05h	Tiempo fuera del comando
06h	Mensaje recibido incompleto
07h	Falla comando NO-ACK local
08h	Numero de sesión local invalido
09h	Recurso no disponible
0Ah	La sesión ha sido cerrada
0Bh	El comando fue cancelado
0Dh	Nombre duplicado en la tabla de nombres NetBIOS local
0Eh	Tabla de nombres NetBIOS llena
0Fh	El nombre tiene sesiones activas
11h	Tabla de sesión local NetBIOS llena
12h	Sesión abierta rechazada porque no hay LISTEN pendiente
13h	Número de nombre ilegal
14h	No puede encontrar al nombre llamado o no contesta
15h	Nombre no encontrado, o no puede especificarse con asterisco (*) o 00h como primer byte del NcbName, o el nombre ha sido borrado y no puede ser utilizado
16h	Nombre usado en un adaptador remoto
17h	Nombre borrado
18h	Sesion terminada anormalmente
19h	Conflicto detectado con el nombre
1Ah	Dispositivo remoto incompatible (PC Network)
21h	Interface ocupada
22h	Muchos comandos pendientes
23h	Numero inválido en el campo NcbLanaNum
24h	Comando completado mientras CANCEL esta ocurriendo
25h	Nombre especificado reservado para ADD GROUP NAME
26h	Comando no válido para cancelar
30h	Nombre definido por otro proceso (solamente Edition Extended OS/2)
34h	Ambiente NetBIOS no definido (solamente Edition Extended OS/2)
35h	Recursos del sistema operativo requerido (solamente Edition Extended OS/2)
36h	Exceden las aplicaciones máximas (solamente Edition Extended OS/2)
37h	No disponibles SAPs para NetBIOS (solamente Edition Extended OS/2)
38h	Petición de recurso no disponible (solamente Edition Extended OS/2)
40h	Error del sistema (PC Network)
41h	Portador impaciente desde un adaptador remoto detectado (PC Network)
42h	Portador impaciente desde este adaptador detectado (PC Network)
43h	Portador no detectado (PC Network)
4Eh	Estatus bit 12, 14, o 15 en más largo que un minuto (Token-Ring)
4Fh	Uno o más de estatus bits 8-11sobre (Token-Ring)
50h-F6h	Mal funcionalmiento del adaptador
F7h	Error implícito en DIR.INITIALIZE
F8h	Error implícito en DIR.OPEN.ADAPTER
F9h	Error interno del programa de sopote LAN IBM
FAh	Rechazo adaptador
FBh	EL programa de NetBIOS no ha sido cargado en la PC
FCh	DIR.OPEN.ADAPTER o DLC.OPEN.SAP parámetros rechazados o faltan
FDh	Inesperado cierre de adaptador
FFh	Estatus comando-pendiente

Tabla A.1. Códigos finales de regreso NetBIOS.

Listado A.4. Netbios.h: Archivo de encabezados de la caja de herramientas NetBIOS.

```
#ifndef NETBIOS_H
#define NETBIOS_H

#define TRUE    1
#define FALSE   0
#define SUCCESS 1
#define FAILURE 0

#ifndef USGC                                    // abreviatura de tipos
#define USGC unsigned char
#endif
#ifndef USGI
#define USGI unsigned int
#endif
#ifndef USGL
#define USGL unsigned long
#endif

#define NetbiosInt21FunctionCode ((USGC) 0x2A)  // interrupciones para acceder a NetBIOS
#define NetbiosInt5C             ((USGC) 0x5C)

//              CODIGOS DE COMANDOS

#define NETBIOS_RESET_WAIT_ONLY          ((USGC) 0x32)
#define NETBIOS_CANCEL_WAIT_ONLY         ((USGC) 0x35)
#define NETBIOS_ADAPTER_STATUS           ((USGC) 0x33)
#define NETBIOS_UNLINK_WAIT_ONLY         ((USGC) 0x70)
#define NETBIOS_TRACE                    ((USGC) 0x79)
#define NETBIOS_ADD_NAME                 ((USGC) 0x30)
#define NETBIOS_ADD_GROUP_NAME           ((USGC) 0x36)
#define NETBIOS_DELETE_NAME              ((USGC) 0x31)
#define NETBIOS_FIND_NAME                ((USGC) 0x78)
#define NETBIOS_CALL                     ((USGC) 0x10)
#define NETBIOS_LISTEN                   ((USGC) 0x11)
#define NETBIOS_HANG_UP                  ((USGC) 0x12)
#define NETBIOS_SEND                     ((USGC) 0x14)
#define NETBIOS_SEND_NO_ACK              ((USGC) 0x71)
#define NETBIOS_CHAIN_SEND               ((USGC) 0x17)
#define NETBIOS_CHAIN_SEND_NO_ACK        ((USGC) 0x72)
#define NETBIOS_RECEIVE                  ((USGC) 0x15)
#define NETBIOS_RECEIVE_ANY              ((USGC) 0x16)
#define NETBIOS_SESSION_STATUS           ((USGC) 0x34)
#define NETBIOS_SEND_DATAGRAM            ((USGC) 0x20)
#define NETBIOS_RECEIVE_DATAGRAM         ((USGC) 0x21)
#define NETBIOS_SEND_BDATAGRAM           ((USGC) 0x22)
#define NETBIOS_RECEIVE_BDATAGRAM        ((USGC) 0x23)
#define NETBIOS_INVALID_COMMAND          ((USGC) 0x7F)
```

Listado A.4. Netbios.h (Continuación).

```
#define MAX_ADAPTER_NUMBER      1
#define MAX_SESSION_COUNT       254
#define MAX_NAMES               254
#define MAX_COMMAND_COUNT       255

#define NO_WAIT                 ((USGC) 0x80)


//              ESTRUCTURA Ncb

struct Ncb
{
        USGC NcbCommand;        // código de comando
        USGC NcbRetCode;        // código de regreso
        USGC NcbLsn;            // número local de la sesión
        USGC NcbNum;            // número asociado a un nombre

        LPSTR NcbBufferOffset;  // I/O buffer offset
        USGI  NcbBufferSegment; // I/O buffer segment

        USGI NcbLength;         // longitud de datos en I/O buffer

        char NcbCallName[16];   // nombre remoto para CALL
        char NcbName[16];       // nombre local del adaptador de red

        USGC NcbRto;            // tiempo fuera RECEIVE en unidades de 1/2 segundos
        USGC NcbSto;            // tiempo fuera SEND en unidades de 1/2 segundos

        LPSTR NcbPostRtnOffset; // offset de post routine
        USGI  NcbPostRtnSegment; // segment de post routine

        USGC NcbLanaNum;        // número de adaptador a ejecutar un cmd
        USGC NcbCmdplt;         // 0xFF => comando pendiente, si no completo

        char NcbReservedArea[14]; // area de trabajo para la tarjeta de red
        } ZeroNcb;

#define MIN_NAME_NUM    2
#define MAX_NAME_NUM    254
#define ILLEGAL_NAME_NUM 0
#define MIN_LSN         1
#define MAX_LSN         254
#define ILLEGAL_LSN     0
```

Listado A.4. Netbios.h (Continuación).

```
//          CODIGOS DE REGRESO

#define NB_ILLEGAL_BUFFER_LENGH                 0x01
#define NB_INVALID_COMMAND                      0x03
#define NB_COMMAND_TIMED_OUT                    0x05
#define NB_MESSAGE_INCOMPLETE                   0x06
#define NB_NO_ACK_FAILURE                       0x07
#define NB_ILLEGAL_LSN                          0x08
#define NB_NO_RESORCE_AVAILABLE                 0x09
#define NB_SESSION_CLOSED                       0x0A
#define NB_COMMAND_CANCELED                     0x0B
#define NB_DUPLICATE_LOCAL_NAME                 0x0D
#define NB_NAME_TABLE_FULL                      0x0E
#define NB_NAME_HAS_ACTIVE_SESSIONS             0x0F
#define NB_LOCAL_SESSION_TABLE_FULL             0x11
#define NB_SESSION_OPEN_REJECTED                0x12
#define NB_ILLEGAL_NAME_NUMBER                  0x13
#define NB_CANNOT_FIND_CALLED_NUMBER            0x14
#define NB_NAME_NOT_FOUND_OR_ILLEGAL            0x15
#define NB_NAME_USED_ON_RMT_ADAPTER             0x16
#define NB_NAME_DELETED                         0x17
#define NB_SESSION_ENDED_ABNORMALLY             0x18
#define NB_NAME_CONFLICT_DETECTED               0x19
#define NB_INCOMPATIBLE_RMT_DEVICE              0x1A
#define NB_INTERFACE_BUSY                       0x21
#define NB_TOO_MANY_COMANDS_PENDING             0x22
#define NB_INVALID_ADAPTER_NUMBER               0x23
#define NB_CMD_COMPLETED_DURING_CANCEL          0x24
#define NB_RESERVED_NAME_SPECIFIED              0x25
#define NB_CMD_NOT_VALID_TO_CANCEL              0x26
#define NB_LANA_SYSTEM_ERROR                    0x40
#define NB_LANA_REMOTE_HOT_CARRIER              0x41
#define NB_LANA_LOCAL_HOT_CARRIER               0x42
#define NB_LANA_NO_CARRIER_DETECTED             0x43
#define NB_UNUSUAL_NETWORK_CONDITION            0x44
#define NB_ADAPTER_MAL_FUNCTION                 0x50
#define NB_COMMAND_PENDING                      0xFF
#define MAX_SESSION_BUFFER_SIZE                 8192

struct    SessionMsg {USGL TextLength,
                      char Text[MAX_SESSION_BUFFER_SIZE];
                                 };

#endif
```

APÉNDICE B

Mensajes de Error

1. Longitud del Buffer invalida
2. Comando invalido
3. Comando fuera de tiempo
4. Mensaje recibido incompleto
5. Falla del comando local No-Ack
6. Número de sesion local invalida
7. Recurso no disponible
8. La sesión ha sido cerrada
9. El comando fue cancelado
10. Nombre duplicado en la tabla de nombres NetBIOS local
11. Tabla de Nombres NetBIOS llena
12. El nombre ya tiene sesiones activas
13. La tabla de sesiones NetBIOS local esta llena
14. Petición de sesion es rechazada porque no hay comando Listen pendiente
15. Número de nombre inválido
16. No se puede encontrar el número llamado o no hay respuesta
17. Nombre no encontrado o ilegal
18. Nombre usado en un adaptador remoto
19. Nombre borrado
20. Sesión terminada anormalmente
21. Se detecto un conflicto en el Nombre
22. Dispositivo remoto incompatible
23. Interface ocupada
24. Muchos comandos pendientes
25. Número inválido de adaptador
26. Comando completado mientras que el comando Cancel ocurria
27. El Nombre especificado es reservado para un nombre de grupo
28. Comando no válido para cancelar
29. Error del sistema
30. Portador desde un adaptador remoto detectado
31. Portador desde este adaptador detectado
32. Portador no detectado
33. Condición Inusual de la red
34. Mal funcionalmiento del adaptador
35. Comando pendiente

Lista B.1. Posibles errores detectados por NetBIOS.

1 La función no fue soportada
2 El sistema estaba fuera de memoria
3 Error en la red
4 Apuntador incorrecto
5 Nombre de recurso de red inválido
6 Nombre de dispositivo local inválido
7 Contraseña invalida
8 Acceso denegado
9 El dispositivo ya estaba conectado

Lista B.2. Posibles errores al efectuar una conexión lógica.

1 El sistema estaba fuera de memoria
2. Error en la red
3 Apuntador incorrecto
4. Nombre de dispositivo local o de red inválido
5. Nombre de dispositivo inválido
6. Archivos abiertos La conexión no fue cancelada

Lista B.3. Posibles errores al efectuar una desconexión lógica.

1. Fuera de memoria o archivo executable dañado
2 Archivo no encontrado
3 Ruta no encontrada
4. Error de protección de red
5. Se requiere una libreria
6. Insuficiente memoria para comenzar la aplicación
7. Versión de Windows incorrecta
8. Archivo ejecutable inválido
9 La aplicación fue diseñada para un sistema operativo diferente
10. La aplicación fue diseñada para MS-DOS 4.0
11. Tipo de archivo ejecutable no conocido
12. La aplicación modo-real
13. Error al cargar una segunda instancia de un archivo executable
14. Archivo ejecutable comprimido
15. Libreria de enlace dinámico incorrecta
16. La aplicación requiere Microsoft Windows 32-bit

Lista B.4. Posibles errores al tratar de ejecutar el archivo de instalación.

```
Fatal Error 1110: CSWINDOW, WORK_WINDOW: null pointer.
Fatal Error 1115: CSWINDOW, ACTIVATE_WINDOW: null pointer
Fatal Error 1120: CSWINDOW, HIDE_WINDOW: null pointer.
Fatal Error 1125: CSWINDOW, OUT_CHAIN: null pointer.
Fatal Error 1130: CSWINDOW, WIN REMOVE: window doesn't exist
Fatal Error 1135: CSWINDOW, win_make: Trying to create window -1.
Fatal Error 1140: CSWINDOW, Not Enough memory for windows
Fatal Error 1145: CSWINDOW, win_current: Tracing error faulty number.
Fatal Error 1310: CSJULIAN: Can't calculate julian date for year=0.
Fatal Error 1410: CSDATE: Unknown date format!
Fatal Error 1510: CSFIELD: Type of set_max() differs from the field type.
Fatal Error 1515: CSFIELD: Type of set_min() differs from the field type.
Error 1610: CSHEAP: Heap is already open.
Fatal Error 1620: CSHEAP: Can't open: Heap not initialized.
Fatal Error 1630: CSHEAP: Not open: Can't allocate.
Fatal Error 1640: CSHEAP: Malloc() internal error: You found a bug!
Fatal Error 2110: CSMENU: MENU point_2_opt: invalid option.
Fatal Error 2115: CSMENU: to many options.
Fatal Error 2120: CSMENU submenu: invalid option.
Fatal Error 2125: CSMENU display option: invalid option.
Fatal Error 2130: CSMENU connect: invalid option.
Fatal Error 2510: CSTEMPLATE: invalid template!
Fatal Error 2515: CSTEMPLATE: invalid template string.
Fatal Error 2520: CSTEMPLATE: character %s not allowed in template!
Fatal Error 3110: CSPANEL: can't allocate field.
Fatal Error 3115: CSPANEL: Error, max number of fields reached!
Fatal Error 3120: CSPANEL: exit_field: field %s does not exist!
Fatal Error 3125: CSPANEL: protecting non-existing field!
Fatal Error 3130: CSPANEL: can't allocate field for double.
Fatal Error 3135: CSPANEL: can't allocate field for float.
Fatal Error 3140: CSPANEL: can't allocate field for long.
Fatal Error 3145: CSPANEL: can't allocate field for integer.
Fatal Error 3150: CSPANEL: can't allocate field for string.
Fatal Error 3155: CSPANEL: can't allocate field for character.
Fatal Error 3160: CSPANEL: can't allocate field for DATE.
DEBUG Warning 3310: CSMALLOC: File %s line %s farcalloc() returns NULL.
DEBUG Warning 3320: CSMALLOC: File %s line %s farmalloc() returns NULL.
DEBUG Warning 3330: CSMALLOC: File %s line %s calloc() returns NULL.
DEBUG Warning 3340: CSMALLOC: File %s line %s malloc() returns NULL.
DEBUG Warning 3380: CSMALLOC: File %s line %s realloc() returns NULL.
DEBUG Warning 3390: CSMALLOC: File %s line %s realloc() returns NULL.
DEBUG Error 3315: CSMALLOC: File %s line %s farfree(NULL).
DEBUG Error 3316: CSMALLOC: File %s line %s farfree( NOT USED ).
DEBUG Error 3325: CSMALLOC: File %s line %s free(NULL).
DEBUG Error 3326: CSMALLOC: File %s line %s free( NOT USED ).
DEBUG Error 3335: CSMALLOC: File %s line %s realloc(NULL)
DEBUG Error 3336: CSMALLOC: File %s line %s realloc( NOT USED ).
DEBUG Error 3345: CSMALLOC: File %s line %s farrealloc(NULL).
DEBUG Error 3346: CSMALLOC: File %s line %s farrealloc( NOT USED ).
```

Lista B.5. Posibles errores de la base de datos.

```
DEBUG Error 3350: CSMALLOC  Can't open log file "%s
DEBUG Error 3360: CSMALLOC  NearHeap fails test in file "%s line "%s
DEBUG Error 3370: CSMALLOC  FarHeap fails test in file "%s line "%s
Fatal Error 3840: EDSTR  string to long: "%s
Fatal Error 3845: EDSTR  alloc_min: Can't minimize allocation!
Fatal Error 3850: EDSTR  alloc_adjust: Can't adjust allocation!
Fatal Error 3855: EDSTR  can't allocate memory!
Fatal Error 7010: BASE out of memory
Fatal Error 8010: "%s is not a PAGE database file, or file is corrupted
Fatal Error 8020: PAGE "%s No pages in empty page chain
Fatal Error 8210: "%s is not a BTREE file, or file corrupted!
Fatal Error 8215: Can't open BTREE "%s: Is already open
Fatal Error 8216: Can't open BTREE "%s: Out of memory
Fatal Error 8220: Can't initialize BTREE: Out of memory
Warning 8225: Multiple Keys can not be set: BTREE "%s is already open; IGNORED.
Fatal Error 8226: BTREE minimum key length is 1: Can't Define "%s.
Fatal Error 8227: BTREE negative data length: Can't Define %s.
Fatal Error 8228: BTREE multiple keys NOT allowed with data length 0.
Error 8230: BTREE is closed: Can't Pack
Error 8235: BTREE "%s Pack(): Can't open temporary file
Error 8240: BTREE "%s Pack(): Out of memory
Error 8245: BTREE "%s Pack(): Disk Full!"
Error 8250: BTREE "%s Pack(): Read error
Fatal Error 8310: CSTABLE "%s Index key less then data key
Error 8320: CSTABLE "%s Attempt to use PREV() before SEARCH() or INSERT().
Error 8330: CSTABLE "%s Attempt to use NEXT() before SEARCH() or INSERT().
Fatal Error 8340: CSTABLE "%s Insufficient memory to allocate data block.
Fatal Error 8350: CSTABLE "%s Insufficient memory to allocate index block.
Fatal Error 8420: CSARRAY "%s Argument of SIZE(%s) too large.
Fatal Error 8430: CSARRAY "%s Not enough memory to allocate buffer.
Warning 8440: CSARRAY "%s You can't change SIZE()  Attempt ignored.
Fatal Error 8450: CSARRAY "%s Index %s out of range
Fatal Error 8460: CSARRAY "%s Class not properly set up.  SIZE() has to be called.
Fatal Error 8710: BUFFER "%s Out of memory.  Can't allocate buffer
Fatal Error 8715: BUFFER "%s Can't initialize class.  Not enough memory.
Fatal Error 8720: BUFFER "%s No buffers available
Fatal Error 8725: BUFFER class not open.  Can't set number of buffers
Fatal Error 8726: BUFFER class not open.  Cannot load buffer.
Fatal Error 8730: BUFFER "%s Buffer 2 file, page "%s  Can't perform fseek.
Fatal Error 8735: BUFFER "%s File 2 buffer, page "%s  Can't perform fseek.
Fatal Error 8740: BUFFER "%s Buffer 2 file, page "%s  Can't perform fwrite.
Fatal Error 8745: BUFFER "%s File 2 buffer, page "%s. Can't perform fread.
Fatal Error 8750: BUFFER "%s number_buff() needs to be called before assign().
Fatal Error 8755: BUFFER  Can't open. Is already opened.
Fatal Error 8756: BUFFER "%s  Number of buffers NOT set
Error 8790: BUFFER  "%s locate_buff() Priority "%s out of range [0..31]. ADJUSTED.
Error 8791: BUFFER  "%s load_buff() Priority "%s out of range [0..31]. ADJUSTED.
Error 8792: BUFFER  "%s change_stat() Priority "%s out of range [0..31]. ADJUSTED
Fatal Error 9310: TBASE  "%s Recordsize larger then 32765 bytes.
```

Lista B.5. Posibles errores de la base de datos (continuación).

```
DEBUG Error 9320: TBASE: "%s Invalid record "%s"      Waiting for keyboard hit
Fatal Error 9340: TBASE is already open. Can't open again
Fatal Error 9341: TBASE is open. Can't call define()
Fatal Error 9350: %s is not a TBASE file, or file corrupted!
Error 9390: TBASE: %s Out of memory during pack()
Warning 9504: "%s Can't call empty(). Database needs to be open. IGNORED
Fatal Error 9505: PAGE: %s Too much DATA read from header. Max is %s
Fatal Error 9506: PAGE: %s Too much DATA written to header. Max is %s.
Fatal Error 9510: PAGE: "%s Read_header: can't perform fseek
Fatal Error 9511: PAGE: "%s Write user data; database NOT open
Fatal Error 9512: PAGE: "%s Write user data; can't perform fseek
Fatal Error 9513: PAGE: "%s Write user data; disk full!"
Fatal Error 9515: PAGE: "%s Data_2_header(): can't perform fseek
Fatal Error 9520: PAGE: "%s Read_header: eof
Fatal Error 9521: PAGE: "%s Read user data; database NOT open.
Fatal Error 9522: PAGE: "%s Read user data; can't perform fseek.
Fatal Error 9523: PAGE: %s Read user data; can't perform fread
Fatal Error 9525: PAGE: "%s Data_2_header(): can't perform fwrite
Fatal Error 9530: PAGE: "%s Read_header: no error
Fatal Error 9535: PAGE: "%s Header_2_data(): Database is not open!
Fatal Error 9540: PAGE: "%s Write_header: can't perform fseek
Fatal Error 9545: PAGE: %s Header_2_data(): can't perform fseek.
Fatal Error 9550: PAGE: %s Write_header: can't perform fwrite
Fatal Error 9555: PAGE: %s Header_2_data(): can't perform fread.
Fatal Error 9560: PAGE: %s Can't open file during definition.
Error 9565: PAGE: %s Can't set page size. Database already open. IGNORED
Error 9570: PAGE: %s Can't open. Locked? Read only?
Error 9571: PAGE: %s Can't open. File doesn't exist
Error 9575: PAGE: "%s Maximum page size is "%s bytes. Page size is NOT changed.
Fatal Error 9580: PAGE: "%s Can't close database. Disk full?
Error 9585: PAGE: "%s Can't open database. Is already open
Fatal Error 9590: PAGE: "%s File not open. Can't read data from header
Fatal Error 9591: PAGE: "%s File not open. Can't write data to header
Error 9592: PAGE: "%s File not open. Can't execute 'save_as()'
Error 9595: PAGE: %s Out of memory during 'save_as()'
DEBUG Warning 9596: PAGE Can't close(). Class not open
Fatal Error 9610: VRAM: %s Wrong page size. File corrupted!
Fatal Error 9620: VRAM: %s Can't initialize, out of memory
Fatal Error 9630: %s is not a VRAM file, or file is corrupted.
Fatal Error 9631: VRAM: "%s vmalloc(%s), attempt to allocate more then "%s bytes.
Fatal Error 9640: VRAM: %s Chunk size of %s bytes is to big
Error 9666: VRAM: class NOT open. Can't execute "%s function
Error 9667: VRAM: %s already open. Can't execute "%s function
Fatal Error 9710: VBASE: %s Can't allocate empty-chain table.
Fatal Error 9720: VBASE: "%s Maximum-record-size of %s bytes is too big.
Fatal Error 9730: VBASE: %s Attempt to read invalid record "%s.
Fatal Error 9740: VBASE: "%s Attempt to write invalid record "%s
Fatal Error 9750: VBASE: "%s Can't open. Illegal number of indexes!
Fatal Error 9751: VBASE: "%s Can't open. Wrong page size.
```

Lista B.5. Posibles errores de la base de datos (continuación).

Fatal Error 9760 VBASE "as Attempt to append record with invalid length "as.
Fatal Error 9761 VBASE "as Attempt to write record with invalid length "as
Error 9768 VBASE "as Database NOT open Can't execute PACK()
Error 9771 VBASE "as Database NOT open. Can't execute EMPTY().
Warning 9772 VBASE "as Not enough memory to execute PACK()
Fatal Error 9780 "as is not VBASE database file, or file is corrupted

Lista B.5. Posibles errores de la base de datos (continuación).

BIBLIOGRAFÍA

- "*Advanced C*". Herbert Schildt, Osborne Mc Graw-Hill
- "*Borland C++ 3.1. Programación Orientada a Objetos*". Ted Faison, SAMS una División de Prentice Hall Hispanoamericana S.A.México, 1993.
- "*Business Data Communications*". David A. Stamper, 3ª Ed., The Benjamin/Cummings, Publishing Company, Inc., California, 1991.
- "*C Programmer's Guide to NetBIOS, IPX, and SPX*". W. David Schwaderer, SAMS Publishing, USA, 1992
- "*Computer Networks*". Andrew S. Tanenbaum, 2ª Ed. Prentice Hall,USA, 1989
- "*Comunicaciones y Redes de Procesamiento de Datos*". Néstor González Sainz, McGraw-Hill, México, 1987
- "*Distributed Systems*". Edited by Sape Mullender, acm PRESS, Frontier Series,1991.
- "*Ingeniería del Software un Enfoque Práctico*". Roger S. Pressman, 3ª Ed. Mc Graw Hill, España, 1993
- "*NetWare 4 for Professionals*". Bierer, Hatch, Higley, Gendreau, y Siyan, New Riders Publishing, Indianapolis, USA, 1993.
- "*Redes de Computadores, Aspectos Técnicos y Operacionales*". Ed. Paraninfo, S.A., Daniel A. Menasce, Daniel Schwabe, 1988, Madrid.
- "*Redes para Todos*". Mark Gibbs, 2ª Ed. Prentice-Hall Hispanoamericana, S.A., SAMS Publishing, México, 1995
- "*Soluciones Avanzadas*". Núm.2, Redes, Enero-Febrero 1993, Director/Editor Carlos Vizcaíno Sahagún, Publicación Bimestral de Xview, S.A. de CV, México.
- "*Uncharted Windows programming*". William H. Roetzheim,SAMS Publishing,1ª Ed., USA, 1993.
- "*Undocumented Windows A Programmer's Guide to Reserved Microsoft® Windows API Functions*". Andrew Schulman, David Maxey, Matt Pietrek. Adisson-Wesley Publishing Company, 2ª Ed., USA, 1992
- "*Windows Programmer's Guide to ObjectWindows Library*"..Namir Clement Shammas,SAMS Publishing,1ª Ed., USA, 1992.

PUBLICACIONES EN INTERNET

- *The GNU C Library - Top*
  http://www.th-muenchen.de/th/susehilf/gnu/libc/Top.html
  24 Octubre 1994

- *The standard C library: 'libc.a'*
  http://ebweb.tuwien.ac.at/gnu-docs/libc-1.12/libc_1.html

- *Middleware*
  http://www8.zdnet.com/pcmag/issues/1506/pcmg0089.htm
  David S. Linthicum
  Marzo 26, 1996

- *PC Magazine Utility: InCtrl3*
  http://www8.zdnet.com/pcmag/download/utils/inctrl3.htm
  Neil J. Rubenking
  Julio 96