46
2ej

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ACATLAN

DESARROLLO DE UNA APLICACIÓN UTILIZANDO TECNOLOGÍA CLIENTE/SERVIDOR Y METODOLOGÍA ORIENTADA A OBJETOS

TESIS QUE PARA OBTENER EL GRADO DE:

LIC. EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN

PRESENTAN:

TORALES CHAVEZ, ALFREDO
Y
VARGAS GONZÁLEZ, JOSÉ G.

1996

TESIS CON
FALLA DE ORIGEN

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES "ACATLAN"

DIVISION DE MATEMATICAS E INGENIERIA
PROGRAMA DE ACTUARIA Y M.A.C.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MEXICO

SRES.: JOSE GPE. VARGAS GONZALEZ
ALFREDO TORALES CHAVEZ
Alumnos de la carrera de M.A.C.
P r e s e n t e .

Por acuerdo a su solicitud presentada con fecha 4 de mayo de 1995, me complace notificarles que esta Jefatura tuvo a bien asignarles el siguiente tema de Tesis Conjunta: "DESARROLLO DE UNA APLICACION UTILIZANDO TECNOLOGIA-CLIENTE/SERVIDOR Y METODOLOGIA ORIENTADA A OBJETOS", el-cual se desarrollará como sigue:

INTRODUCCION
CAP. I Ambiente Cliente/Servidor.
CAP. II Sistemas Operativos desde perspectivas de Cliente/Servidor.
CAP. III Comunicación.
CAP. IV SQL.
CAP. V Desarrollo de Sistema.
CONCLUSIONES.
BIBLIOGRAFIA.

Asimismo, fue asignado como Asesor de Tesis el Lic en M.A.C. Juan Carlos Rendón Aguilar, Profesor de esta Es-cuela.

Ruego a usted tomar nota que en cumplimiento de lo especificado en la Ley de Profesiones, deberán presentar servicio social durante un tiempo mínimo de seis meses como requisito básico para sustentar Examen Profesional, así-como de la disposición de la Coordinación de la Administración Escolar en el sentido de que se imprima en lugar visible de los ejemplares el título del trabajo realizado. Esta comunicación deberá imprimirse en el interior de la misma.

E.N.E.P. ACATLAN

A T E N T A M E N T E
"POR MI RAZA HABLARA EL ESPIRITU"
Acatlán, Edo. Méx. mayo 9 de 1996.

ACT. LAURA MA. RIVAS BECERRA
Jefe del Programa de Actuaría
y M.A.C.

JEFATURA DEL PROGRAMA DE ACTUARIA Y MATEMATICAS APLICADAS Y COMPUTACION

cg'

Agradezco a DIOS por darme la oportunidad de vivir y poder realizar este trabajo.

Este trabajo está dedicado a mi Padre, y si en este momento Dios me concediera un deseo, éste sería el poder verlo y decirle "PADRE YA TE CUMPLÍ".

Este trabajo se terminó gracias al apoyo de varias personas a las que ofrezco mi mayor agradecimiento como: mi madre, hermanos, maestros, compañeros de trabajo y amigos. Les puedo decir que la ayuda brindada no fue en vano.

La única misión de la U.N.A.M. es formar profesionales, que realicen su mejor desempeño en nuestra sociedad. Hoy en día, es necesario y urgente que cambiemos nuestra actitud y conciencia, porque no sólo nos debemos a nosotros mismos, sino a toda la sociedad que nos rodea como lo es nuestra familia y nuestra patria "México". Y lo más sorprendente es que sólo nos falta voluntad. Razona: "SI PUEDO" y lo lograrás.

Dios, ha puesto en cada alma un apóstol para que nos guíe por el sendero de la iluminación; sin embargo, muchos buscan la vida en lo externo sin reparar que lo que buscan está dentro de ellos.

Es mejor reflexionar de lo que "hiciste", que de lo que "pude hacer" sea esto bueno ó malo. Así, el hombre no se mide ni se valora por lo que tiene, sino por lo que desea tener.

ALFREDO

La vida es como la luz; siempre esperamos que nunca se acabe, aún cuando sabemos que existe la obscuridad.

Gracias a mi madre por ser mi Sol, a mi padre porque se que me esta viendo, a mi hijo José Alberto, porque Dios me ha dado la oportunidad de iluminar su camino y a Rosa por esperar a que veamos la luz de un nuevo día.

A toda mi familia que contribuyó a ser lo que soy, en especial a mis hermanos Javier, Yolanda, Pilar y a Luis Salazar. A José Luis porque siempre seguí su ejemplo.

A todos mis amigos de la Secretaría de Hacienda, en especial a Loly, por siempre decirme "usted dejese llevar" y he llegado hasta aquí, a Andrea por dejarme apoyar en su hombro y por último a Alfredo, por enseñarme a distinguir a los verdaderos amigos en los momentos difícilos.

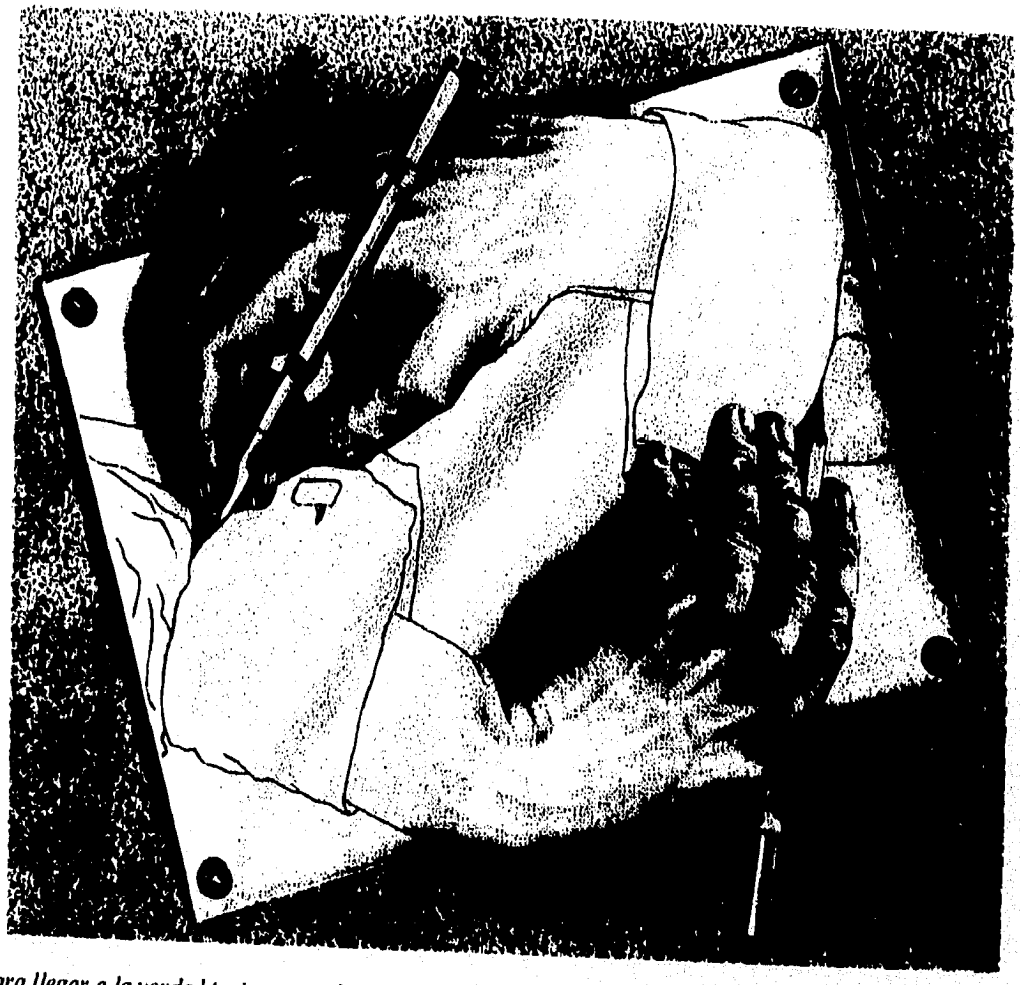
A mis amigos de Tacuba que gracias a su ejemplo se cual es mi camino.

La presente tesis esta dedicada a todas las "personitas" que están a mi alrededor.

José Gpe. Vargas González.

DESARROLLO DE UNA APLICACION UTILIZANDO TECNOLOGIA CLIENTE/SERVIDOR Y METODOLOGIA ORIENTADA A OBJETOS.

INTRODUCCION
PROLOGO



Para llegar a la verdad tuvimos que documentarnos, discutir y razonar, pero siempre tuvimos en mente que la verdad es única.

COMPLEJIDAD **Alfredo y José**

Sir Issac Newton lo admitió secretamente a algunos amigos:
comprendia cómo se comportaba **la gravedad**, *pero no cómo funcionaba.*

INTRODUCCION

Las avances tecnalógicos han permitido a las organizaciones adquirir una gran variedad de equipa de cómputo, que les permitía resolver determinadas necesidades. Sin embarga ahara se cuenta can el equipo interconectado canfarmando redes (Locales, Metropolitanas y Amplias) que permitan campartir recursos, información y procesamiento. En este sentido, es necesaria implantar un modelo que responda a las expectativas planteadas. En el capítulo uno, se describe el madelo cliente/servidor que respande a estas necesidades.

Un elemento fundamental en la arquitectura cliente/servidor son los sistemas aperativos, en el capítulo das se exponen los sistemas operativos de mayor demanda en el mercado informático desde una perspectiva del modela cliente/servidor.

En el capítulo tres se aborda el tema de comunicaciones, donde hablaremos del software y hardware necesario para establecer la camunicación entre computadoras, tratando los puntos de: el modelo OSI, protocolas y procesos de comunicación, manejadares de base de datos y los dispositivos necesarios para conectar diferentes redes.

En la actualidad a las organizaciones les interesa primordialmente contar can bases de datos confiables y de mantenimiento eficiente. En el capítulo cuatro se trata SQL, que es un vehícula natural para resolver las necesidades anteriormente planteadas a través de sus pocas instrucciones, introduciéndolas en farma interactiva (por medio de una herramienta, ya sea para usuarios finales o de administración a bases de datos) o que estén contenidas en aplicacianes desarrolladas. SQLWindows es una herramienta visual para el desarrollo de aplicaciones que par su facilidad y total manejo de la programación orientada a objetas, la hace una herramienta viable para el desarrallo de aplicaciones cliente/servidor.

En el capítulo cinco y último de este trabajo se explica la principal diferencia que existe entre la metadología estructurada y la arientada a objetos, y esto estriba, en que la primera analiza los pracesos y su descamposición y la segunda analiza los objetos y su comportamiento.

PROLOGO

La arquitectura cliente/servidor, es una forma de cómputo en red en el que ciertas funciones solicitadas por el cliente son procesadas por un servidor. En este escenario donde los principales protagonistas son el *cliente*, el *servidor* y la *red*, en un ambiente donde el lenguaje es el proceso *distribuido* y su rol es la convivencia en los *sistemas abiertos* son los conceptos que desarrollaremos en el presente trabajo.

Existen además otros personajes claves en la arquitectura cliente/servidor, coma lo es el sistema operativo y su control del hardware, de procesos y recursos. Los sistemas operativos modernos (Windows NT, Netware), poseen un control de procesos como el uso de múltiples hilos, el uso de RPC (procedimientos remotos) y un sistema de archivos donde el nivel de seguridad es bastante confiable. Unix es considerado como el sistema operativo estándar por el manejo de comunicaciones y el protocolo TCP/IP.

El software y hardware de comunicación hoy en día es bastante complejo y diverso, y si no se llevará un estándar será casi imposible establecer diálogo entre computadoras. Organizacianes como la ISO (Organización Internacional de Estándares), OSF (Fundación de Software Abierto) y otras, son piezas claves para la buena coordinación de los productos de comunicaciones. Las bases de datos ya no manipulan pequeñas cantidades de datos (objetos), por lo que las técnicas de OLTP y datawarehouse son utilizados en el procesamiento de datos.

El SQL (Lenguaje Estructurado de Consultas), el hecho de que sea muy fácil de usar no significa que no sea poderoso, y por lo mismo que es fácil de usar es muy popular coma lenguaje para herramientas de usuario final. Muchos Manejadores de Base de Datos lo han adoptado y al igual que el software y hardware de comunicación existen diferentes organizaciones por generar un estándar.

Las metodologías tradicionales ya no son suficientes para soportar los grandes cambios de la realidad en el ámbito de desarrollo de sistemas . Por lo que la tecnología orientado a objetos es la mejor alternativa para solucionar de una forma integral el desarrollo de sistemas.

LISTA DE FIGURAS Y TABLAS

CAPITULO I

INTRODUCCION A LA TECNOLOGIA CLIENTE/SERVIDOR

-¡Dios mío ! ¡Dios mío ! ¡ Que extraño es hoy todo! Tan solo ayer todo transcurría normalmente. ¿Habré cambiado durante anoche? Reflexionas. ¿ Era yo la misma cuando me levante esta mañana? Casi creo que me sentí un poca distinta. Pero si no soy la misma, entonces la pregunta es. ¿ Quien soy?

LEWIS CARROLL

ALICIA EN EL PAIS DE LAS MARAVILLAS.

CAPITULO 1

AMBIENTE CLIENTE/SERVIDOR

La idea básica del modelo computacional usado por varios años ha sido el de grandes computadoros centroles "mainfrome" (modelo tradicional). Desde hace más de veinte años, la industria de la computación ha venido usando a los poderosos mainframes en dande los usuarios comparten tanto procesadores como facilidades de almacenamiento de datos y periféricos. El acceso a estos mainframes era estrictamente cantrolado por los departamentos de sistemas y el único medio para accesar los datos era a través de tarjetas perforadas o terminales tontas, las cuales no tenían un procesador interno y cualquier proceso relacionado con la transacción o consulta de datas era desarrollado por el computador central. Además, las técnicas y metodologías utilizadas eran rígidas debido a la centralización de recursos y a la carencia de interfaces amigables con el usuario, ver figura 1.1.

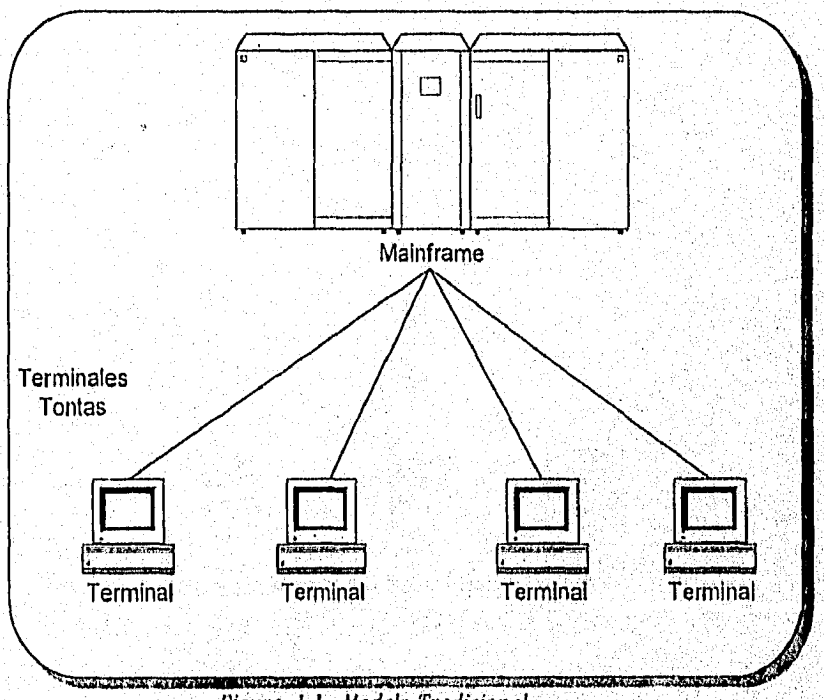


Figura 1.1. Modelo Tradicional.

En los últimos años se ha visto una evolución tanto en la recuperación de datos, como en las metodologías de análisis y en la amigabilidad de las interfaces con el usuario. Al aparecer las computadoras personales a principios de los ochenta y dadas las facilidades que ofrecen en cuanto a capacidades de proceso, almacenamiento de datos y software amigable, todo en equipos de escritorio, se dio un crecimiento fenomenal en la base instalada de estos equipos.

Simultáneamente los mainframes siguieron evolucionando, aunque no tan rápidamente y si bien redujeron costos en hardware y software, éstos no han llegado a ser tan bajos como para que algunas organizaciones puedan adquirirlos o al menos mantenerlos.

Por otro lado, las computadoras personales por sí solas no son capaces de manejar las principales necesidades de proceso de datos en la mayoría de las organizaciones. Por ello, surgen como una primera solución, las redes de área local "LAN" (Local Area Network), ver figura 1.2, que permiten entre otras cosas, la conexión de computadoras personales con mainframes y minicomputadoras, ver figura 1.3.

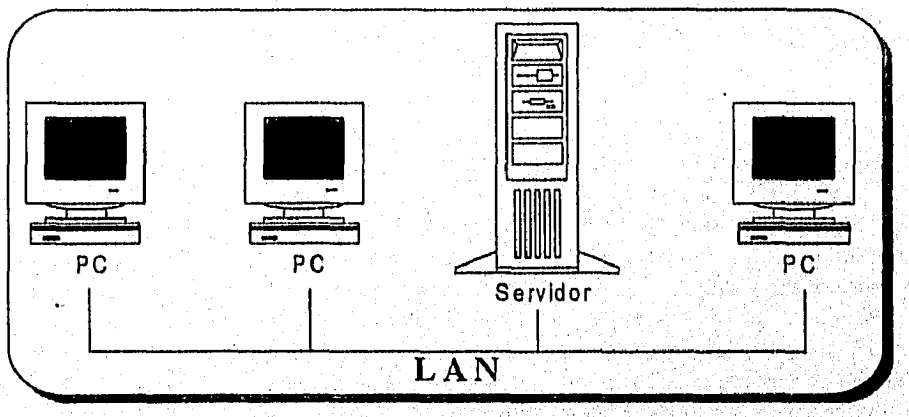


Figura 1.2. Red Local.

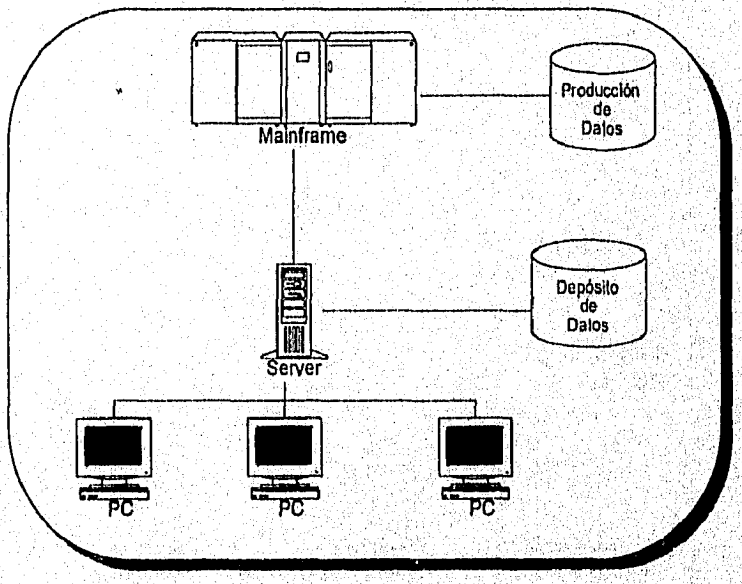


Figura 1.3. Red local, Conexión Mainframe-computadoras.

Así, la computadora personal, entre otros, adoptó el papel de ser una terminal amigable, actuando como interface entre el usuario y el mainframe, es aquí donde surge la computación cliente/servidor. Los servidores de redes locales, tanto de archivos como de impresoras fueron evolucionando. Sin embargo, el compartir impresoras o archivos no significa que tengamos un sistema distribuida. Si bien es cierto que la gran base instalada de PC's, LAN's y software asociado a ellas (procesadores de palabras, hojas de cálculo, manejadores de base de datos) han promovido que se realicen distintos tipos de procesos en estos equipos, estos sólo han permitido compartir recursos y obtener soluciones a nivel departamental. A reserva de ahondar ahora, lo cual precisaremos más adelante, podemos decir que en la arquitectura cliente/servidor existen procesos separados que residen en plataformas diferentes, Tabla 1.1, e interactúan por medio de una red.

UNA PLATAFORMA ES UN CONJUNTO FORMADO POR :
• **Hardware**
• **Software (Sistema Operativo)**
• **Software de Subsistema (Manejador de Base de Datos)**
• **Software de Aplicación**
• **Software de Utilerías**
EXISTEN DIFERENTES TIPOS DE PLATAFORMAS
• **Mainframe o Minis enlazadas:** **Funciona como servidor gigante de archivos o servidor de base de datos, manejan aplicaciones tanto batch como interactivas y la interface con el usuario es tipo carácter, además que son equipos propietarios.**
• **Minis o Estaciones de Trabajo de alto desempeño :** **Dan servicio a nivel departamental, pero a diferencia de las workstations, las microcomputadoras son equipos propietarios y también manejan aplicaciones tanto batch como interactivas**
• **Microcomputadoras:** **Existe una gran base instalada en ellas.**

Tabla 1.1. Plataforma.

Las principales modelos que existen son: *centralizado* y *distribuido*.

MODELO CENTRALIZADO

Desde los inicios de la computación, un modelo comúnmente empleado por grandes empresas e instituciones educativos para organizar y administrar sus recursos de cómputo ha sido el centralizado, en el cual todos los recursos para procesamiento de datos se localizan en una área especialmente diseñada para tal efecto, este centro de procesamiento se localiza generalmente en las oficinas centrales de la organización. En él, grandes computadoras (*mainframes*) y dispositivos periféricos de alto desempeño se concentran para dar atención a todos los usuarios de la organización, es decir, todo el procesamiento de datos se efectúa en este lugar de manera centralizada, mientras que los usuarios finales sólo tienen acceso a él mediante terminales conectadas directamente a los equipos, ver figura 1.1. Esta forma de organización está determinada principalmente por los altos costos de equipos y en algunos casos por los escasos medios de comunicación.

INFRAESTRUCTURA

La infraestructura utilizada para implantar un modelo centralizado, está constituida básicamente por líneas de cable que permiten enlazar cada una de las terminales en los diferentes departamentos u oficinas de la empresa con el centro de cómputo.

MANEJO DE INFORMACION

El acceso a las fuentes de información de la organización es efectuado a través de un estricta control de personal que puede accederlas dependiendo de su puesto, departamento e inclusive de la fecha y hora.

A diferencia de otros esquemas, en el modelo de cómputo centralizado dodos los mecanismos que se emplean para su adquisición, la información conserva las características de ***consistencia y confiabilidad***, pero en muchas de los casos al momento de efectuar una consulta la información pierde características aún más relevantes como lo son ***rapidez y oportunidad***.

PROCESAMIENTO

El manejo de bases de datos centrales y comunes a toda la organización, es una práctica diario, éstas se localizan y se procesan en el centro de cómputo e implican un diseño sumamente compleja dadas las múltiples referencias que existen entre todas los elementos que conforman las bases de datos.

La utilización de sistemas operativas multitarea y multiusuario es común, sin embargo dada la carga excesiva de trabajo para el sistema en horas pico, la respuesta a peticiones de usuarios alcanzan, tiempos inaceptables. Muchas de estos sistemas son diseñadas para trabajar baja un ambiente muy particular, de esta forma se logra un mayor rendimiento y eficiencia. Sin embargo, dado su diseño el sistema operativo pierde versatilidad provocando una constante actualización del mismo.

El manejo de las bases de datos se convierte en una tarea que requiere de complejos algoritmos para la actualización de validación, procesos que consumen excesivos recursos provocando que el sistema en conjunto llegue a ser ineficiente.

Dada la dependencia que existe entre computadoras centrales y periféricos para llevar a cabo todas las tareas de procesamiento de la organización, la tolerancia a fallas del sistema en conjunto es mínima, ya que la falla de algún mecanismo de procesamiento repercutirá directamente en toda el modelo, llegando muchas veces a detener todos los procesos de la organización.

SERVICIOS

El principal servicio lo constituye el mismo procesamiento, dada que este involucrará generalmente la salida de resultados de los datos procesados a papel, cinta y en la mayoría de los casos a pantalla.

En este esquema, el uso de dispositivos periféricos se encuentra restringida a ser sola operado par el personal encargado de atender el centro de procesamiento, cualquier petición de los usuarios para su uso es canalizada, autorizada y atendida por los respectivos encargados.

El empleo de medios magnéticos de almacenamiento, tiempo de procesador y muchos otros recursos está limitado de acuerdo al departamento y puesto de la persona que lo requiere. De está forma cualquier requerimiento extra de estos recursos, deberá ser solicitado y aprobado por los administradores del sistema.

ADMINISTRACION

La tareas de administrar el sistema operativo y aplicaciones juegan un papel crítico ya que para toda la organización son las misma y en caso de ocurrir un error en alguna de ellas el modelo en conjunto fallará, para ello la tarea se debe llevar a cabo por personal altamente calificado, mientras que la parte de infraestructura se encuentra controlada por personal técnico encargado de vigilar el correcto funcionamiento del diferente equipo e instalaciones con que se cuenta para procesamiento y comunicacianes.

Son los administradores los responsables de definir prioridades de procesamiento, uso de periféricas y manejo de información. Así mismo, este grupa es el encargado de determinar qué equipo de cómputo, dispositivos periféricos y aplicaciones se deberán adquirir y desarrollar, siguiendo una política de estandarización. De ésta forma se evitará la redundancia de periféricos y aplicaciones, así como problemas de compatibilidad.

El manejo y aperación de las grandes computadoras de la organización, como ya se había mencionado, recae en personal especializado en administrar el sistema aperativo con el cual el sistema trabaja, osí como conocer a fondo el diseño y estructura de cada una de las aplicaciones utilizadas en la organización. Esta tarea se ve un tanto reducida, ya que no existe una gran variedad de aplicaciones, y comúnmente se dificulta par la complejidad del sistema operativo y de las aplicaciones utilizadas.

MODELO DISTRIBUIDO

Este esquema puede ser definido formalmente como aquel en el cual múltiples procesadores autónomos, posiblemente de diferentes tipos, son interconectados por una red de comunicación para interactuar entre sí, con la finalidad de llevar a cabo tareas de procesamiento común. Para alcanzar está meta los sistemas son integrados lógicamente en varias grados por sistemas operativos para procesamiento distribuido y aplicaciones distribuidas, toles como bases de datos distribuidas. Sin embargo, el esquema no sólo incluye procesadores y sistemas operativos, en un concepto más general el modelo distribuido es capaz de integrar los más diversos equipos periféricos, aplicaciones, tecnologías de diseño de redes y herramientas de administración.

Entre las caracterísficas más relevantes de este modelo se encuentran:

- Soporte para un número arbitrario de sistemas y aplicaciones.
- Diseño modular de la arquitectura físico.
- Uso de sistemas operativos y aplicacianes para procesamiento distribuido.
- Saporte de terceros tanto en hardware como en software.

Mientras que algunos de los beneficias que se derivan de adoptar el modelo, son:

- Incremento de desempeño.
- Incremento de la confiabilidad y disponibilidad.
- Modularidad y control local.
- Costos reducidos de mantenimiento.

INFRAESTRUCTURA

Basado primordialmente en redes de computadoras, el modelo de cómputo distribuida, utiliza todos las recursos y capacidades de éstas como la herramienta principal de su infraestructura, para comunicar a los diferentes recursos que se verán involucrados en el esquema.

Dada la rápida evolución de tecnologías en el campo de la computación, así como en el de las comunicaciones, la red de computadoras que se utilizará para implantar el esquema, como ya se había mencionado, requiere de ser lo suficientemente robusta, eficiente y adaptable a nuevas tecnologías, con la finalidad de garantizar la continuidad del modelo.

El esquema distribuido no sólo permite la integración de las más variadas tecnologías de diseña y construcción de procesadores y periféricas, sino también de las diferentes tecnologías de diseños de redes. Permitiendo de ésta forma, integrar redes de comunicaciones ya existentes dentro de una organización al modelo, que representa una ventaja sobre el esquema centralizado, sin embargo, resulta conveniente que la red a utilizarse sea realmente eficiente, dado que los resultados del modelo dependerán en gran medida de ésta.

Dadas las facilidades de integración de tecnologías de redes y equipos, la infraestructura del modelo distribuido puede ilustrarse con figura 1.4.

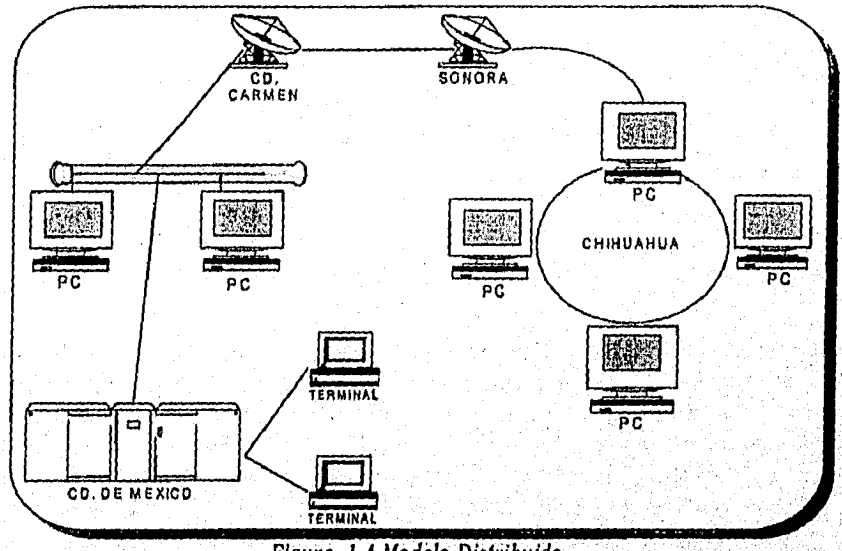


Figura 1.4 Modelo Distribuido.

MANEJO DE INFORMACION

Al igual que se distribuyen equipos de cómputo y periféricos, la información se halla distribuida a lo largo de todo la organización, encontrándose en aquellos lugares donde comúnmente es utilizada, donde sin embargo, no importando su localización, cualquier persona puede tener acceso a ella.

A diferencia del modelo centralizado, la información bajo este esquema conserva las características de oportunidad y rapidez, ya que las consultas se harán normalmente a bases de datos locales. Sin embargo dado que la información generalmente es procesada en forma distribuida y pese a que las herramientas para su manejo empleen complejos algoritmos de validación de datos, ésta es susceptible de perder características como ***consistencia y confiabilidad***.

Con este esquema, el usuario final tiene también la posibilidad de acceder a diversos bancos de información, que existen alrededor del mundo, y la consulta a estos se efectúa a través de aplicaciones y herramientas capaces de manipular la información contenida en ellos, sin importar su localización geográfica. Entre las fuentes externas a los cuales se puede acceder para obtener información actualizada y oportuna, se encuentran diversos redes de computación públicos y privadas que existen alrededor del mundo, destacando entre ellos Internet que es la red más grande e importante, que

cuenta entre sus miembros o universidades, instituciones de investigación, agencias gubernamentales e instituciones privadas. Las herramientas para consultar estas fuentes, comúnmente permiten una fácil interacción con el usuario, al contar con interfaces amigables que harán del proceso de adquisición de información una tarea sencilla para cualquier persona.

PROCESAMIENTO

Nuevos conceptos que han sido integrados a sistemas operativos, así como complejos mecanismos de comunicación entre computadoras a través de una red, permitiendo llevar a cabo el procesamiento de grandes cantidades de datos y complejos algoritmos utilizando para ello múltiples procesadores, que pueden ser de diferentes tipos y encontrarse distribuidos a lo largo de la organización, esto forma de procesar datos, conocida como procesamiento distribuido, es totalmente transparente al usuario final ya que serán los sistemas operativos y aplicaciones los que llevarán a cabo la distribución y control de todas las tareas que esto implique, entre aquellos sistemas que se dispongan para ese efecto dando la apariencia de que todas las operaciones necesarias las está llevando a cabo un solo procesador.

Dado que el procesamiento distribuido sería llevado a cabo por diversos procesadores utilizando como medio de comunicación la red de cómputo, ésta deberá ser lo suficientemente confiable para que el procesamiento pueda efectuarse, pues se requerirá de llamadas y peticiones de control entre todos aquellos procesadores que estén participando en el procesamiento.

SERVICIOS

Sustentar el modelo distribuido en redes de computadoras, permite ofrecer al usuario final una amplia gama de servicios basados en la utilización de aplicaciones que operan bajo la red de cómputo instalada, entre ésta variedad de facilidades se encuentran algunas que para el usuario final revisten vital importancia, como lo es el caso del correo electrónico, ya que éste adquiere gran relevancia cuando la organización pasa a formar parte de algunas de las redes existentes alrededor del mundo. Bajo estos circunstancias, el correo electrónico proporciona mayores beneficios en relación con otros medios de comunicación tradicionales tales como mensajería, fax, telefonía, ya que el usuario no sólo tendrá la posibilidad de entablar comunicación con personas de su organización, sino que también le será posible hacerlo con otras personas e instituciones de forma segura y eficiente, no importando en que lugar del mundo se encuentren.

Otro servicio que reviste igual importancia es la transferencia de información, el modelo cuenta con herramientas eficientes y confiables para transferir información entre sitios remotos sin importar que tipo de información deba transferirse, abarcando desde artículos periodísticos, reportes de clima, imágenes sobre condiciones atmosféricas hasta archivos de aplicaciones guardadas bajo formatos especiales.

Compartir equipos periféricos, tales como impresoras y graficadores de alta calidad y desempeño es una tarea que se facilita tanto para los administradores como para los usuarios finales, haciendo uso de la red de cómputo éstos podrán ser utilizados por todo aquel personal que así lo requiera, sin considerar donde se localicen los recursos. La integración y distribución de recursos en el modelo se efectúa mediante sistemas operativos y aplicaciones, los cuales una vez configurados permitirán al usuario hacer uso de aquellos recursos que le sean más convenientes o adecuados al momento de requerirlo en forma transparente.

Dado la veracidad del modelo distribuido, integrar recursos y servicios no previstos, no implicará rediseñar o modificar totalmente la estructura del modelo, sólo se requerirá variar configuraciones en sistemas operativos y aplicaciones. Integrar nuevos equipos periféricos no implicará contar con un distribuidor o fabricante único, periféricos de terceros pueden ser empleados sin ocasionar problemas de compatibilidad y sin recurrir a gastos excesivos.

ADMINISTRACION

Una de las tareas más importantes para alcanzar las metas previstas cuando se utiliza cómputo distribuido, la constituye la administración de su infroestructura. En este esquema donde el uso exhaustivo de una red de computadoras es común, la administración de este recurso ocupa un lugar primordial, contar con las herramientas adecuadas, así como con el personal capacitado, garantizará el correcto desempeño y funcionamiento de cada una de las partes que forman la red de cómputo. Herramientas comunes son aplicaciones que utilizan protocolos especialmente diseñados para monitorear el estado actual de la red y dar al administrador una idea precisa de cual es el problema y en que parte éste está ocurriendo, para así poder tomar decisiones sobre una solución rápida y adecuada, además de contar con un equipo de personal especializado, permitirá dar pronta solución a problemas técnicos en la infraestructura de la red.

MODELO COOPERATIVO

Es una variante de los sistemas distribuidos, el modelo cooperativo que consiste cuando el procesamiento de la aplicación se ejecuta en la plataforma óptima de hardware.

El termino proceso cooperativo, es una definición literal, implica los componentes individuales, en lo que aplicaciones se refieren, es cuando diferentes aplicaciones de los mismos se llevan a cabo en varias computadoras de la red y estas son transparentes para el usuario, ver figura 1.5.

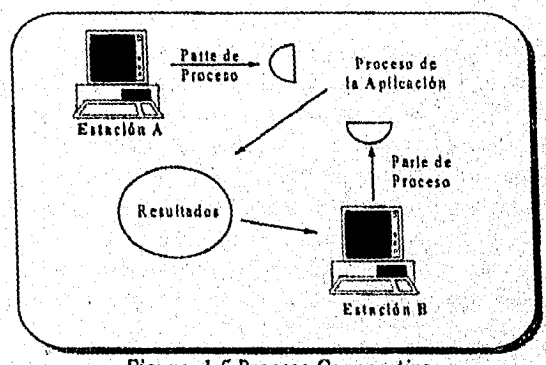


Figura 1.5 Proceso Cooperativo.

Existen tres tipos de procesos para el Modelo Cooperativo, que son : Peer-to-peer, RPC y Cliente/Servidor.

1) PEER-TO-PEER "PAR-A-PAR"

Este modelo tiene una forma similar al modelo cliente/servidor. La gran diferencia radica que en el modelo Peer-to-peer los componentes de procesos son considerados de igual forma y pueden intercambiar los papeles; el cliente realiza las funciones del servidor y viceversa ver figura 1.6.

ADMINISTRACION

Una de las tareas más importantes para alcanzar las metas previstas cuando se utiliza cómputo distribuido, lo constituye la administración de su infraestructura. En este esquema donde el uso exhaustivo de una red de computadoras es camún, la administración de este recurso ocupa un lugar primordial, contar con las herramientas adecuadas, así coma con el personal capacitada, garantizaró el correcto desempeño y funcionamiento de cada una de las partes que forman la red de cómputo. Herramientas comunes son aplicacianes que utilizan protocolos especialmente diseñados para monitorear el estado actual de la red y dar al administrador una idea precisa de cual es el problema y en que parte éste estó ocurriendo, para así poder tomar decisianes sabre una solución rápida y adecuada, además de contar con un equipa de personal especializado, permitiró dar pronta solución a problemas técnicos en la infraestructura de la red.

MODELO COOPERATIVO

Es una variante de los sistemas distribuidos, el madelo cooperativo que consiste cuanda el procesamiento de la aplicación se ejecuta en la plataforma óptima de hardware.

El termino proceso cooperativo, es una definición literal, implica los componentes individuales, en lo que aplicaciones se refieren, es cuando diferentes aplicaciones de las mismas se llevan a cabo en varias computadoras de la red y estas son transparentes para el usuario, ver figura 1.5.

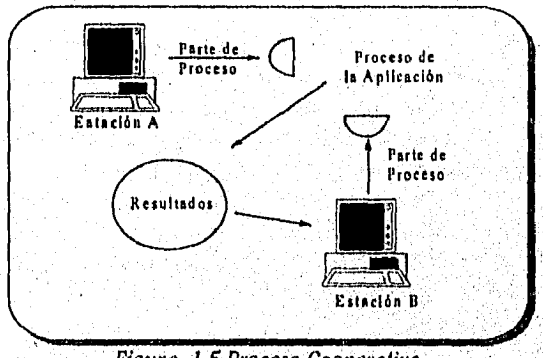


Figura 1.5 Proceso Cooperativo.

Existen tres tipos de procesos para el Modelo Cooperativo, que son : Peer-to-peer, RPC y Cliente/Servidor.

1) PEER-TO-PEER "PAR-A-PAR"

Este modelo tiene una forma similar al modelo cliente/servidor. La gran diferencia radica que en el madelo Peer-ta-peer los componentes de procesos son, considerados de igual forma y pueden intercambiar las papeles; el cliente realiza los funciones del servidor y viceversa ver figura 1.6.

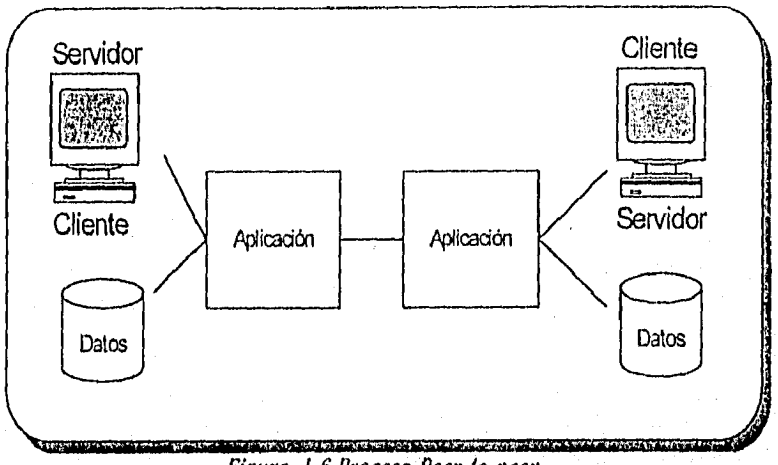


Figura 1.6 Proceso Peer-to-peer.

Esta flexibilidad, de cualquier computadora para actuar como servidor, es donde radica su desventaja, es difícil mantener la seguridad de los datos. Además reduce el rendimiento del sistema, ya que el trafico en la red es mayor. Por otro lado, en el modelo cliente/servidor la seguridad es más fiable ya que los recursos pueden ser centralizados y optimizados, encontrándose disponibles para algunos clientes.

2) RPC (Llamadas de Procedimientos Remotos "Remote Procedure Call")

El RPC es similar a una llamada de procedimiento local, frecuentemente usan código que es almacenado como un procedimiento o una subrutina. Cuanda un programa necesita ejecutar este código, el desarrollo de la llamada del procedimiento local pasa los parámetros necesarios, este entonces ejecuta el procedimiento almacenado. El RPC generaliza este modelo de un sistema monousuario a un sistema de red.

Con el RPC, el procedimiento o subrutina es localizado en otro equipo en la red, por lo tanto su ejecución se considera como *"procesa remata"*.

El mecanismo del RPC, crea esta información dentro de los mensajes de la red y transmite el mensaje al procedimienta remoto, ver figura 1.7. Aunque al usar RPC puede afrecer flexibilidad, es mucho más complejo que el interactuar con SQL (Lenguaje estructurodo de consulta,"Structured Query Language") que se explica en el capítulo IV.

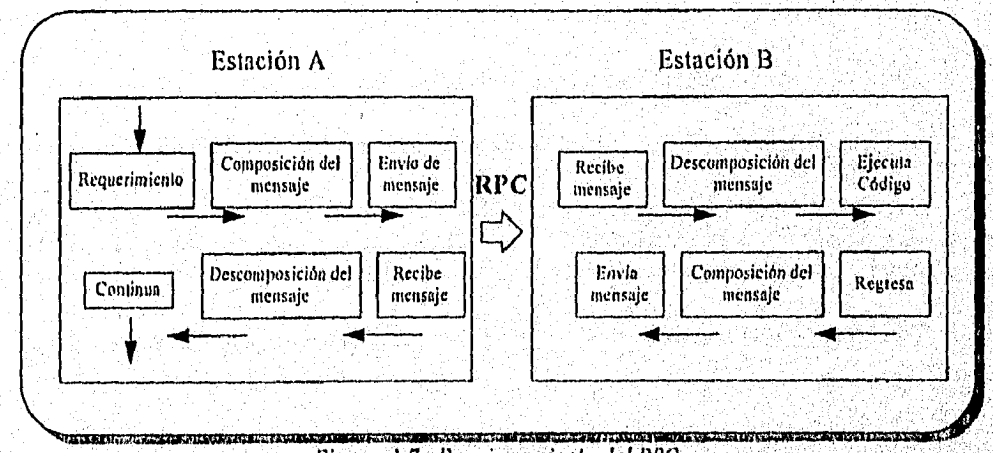


Figura 1.7. Funcionamiento del RPC.

3) CLIENTE/SERVIDOR

En un sistema cliente/servidor, uno o más clientes y uno o más servidores, juntos con el sistema operativo y los protocolos de comunicación, conforman el ambiente que permite y facilita el cómputo distribuido.

En una aplicación basada en esta arquitectura existen dos procesos independientes, en lugar de uno solo. De esta forma se puede repartir el trabajo a través de varios computadoras en una red. Estos dos procesos, el cliente y el servidor, se comunican mediante un protocolo bien definido. Esta técnica modular permite la comunicación entre distintas computadoras (servidor de archivos, estaciones de trabajo con alta calidad de graficación, etc), para que cada una de ellos se dedique a realizar el trabajo que realiza mejor.

De manera introductoria, se puede decir que un servidor es un sistema o un programa que provee de algún servicio a otros elementos a través de una red. Un ejemplo típico es un servidor de archivos, que permite el acceso o información remota a cualquier usuario a través de la red. Un cliente es un sistema o un programa que requiere y recibe alguna acción de un servidor.

Existen varias interpretaciones acerca de la definición exacta de que es la computación cliente/servidor, nosotros daremos la idea básica y mejor discutiremos las características que debe cumplir.

Básicamente la arquitectura cliente/servidor es aprovechar la computación que esta en procesos separados sobre separadas plataformas, interactuando una con otra permitiendo el aprovechamiento de recursos, mientras toman la mejor ventaja de los diferentes dispositivos.

Si bien no existe una definición precisa de cliente/servidor que sea aceptada por el ámbito informática, si existen diez características que deben cumplir. Las cinco primeras son obligatorias y las cinco restantes como opcionales, y precisando aún más como deseables, las cuales son[1]:

1 La arquitectura cliente/servidor, consiste en un proceso cliente y un proceso servidor, distinguiéndose una de otra, sin embargo pueden interactuar conjuntamente.

2 La parte del cliente y la del servidor, pueden operar en plataformas diferentes y esto generalmente ocurre, pero no necesariamente.

3 La plataforma del cliente o del servidor, puede ser actualizada sin que esto implique actualizar ambas.

4 El servidor dispone de servicios múltiples de concurrencia de clientes, en algunos sistemas el cliente puede tener acceso a múltiples servicios.

5 El sistema cliente/servidor incluye capacidad de administración de la red.

6 Un algunos casos no todos, una parte significativa de la lógica de aplicación reside en el cliente.

7 Usualmente la acción es iniciada por el cliente final y no por el servidor final, sin embargo, servidores de base de datos pueden tomar acciones basadas en disparadores (ver punto 4.1.5), como una política de la organización o proceso almacenado.

8 Una interfaz gráfica amigable GUI (Graphical User Interface), generalmente reside en el cliente.

[1] *IMPLEMENTING PRODUCTION QUALITY CLIENT/SERVER, Barbara Bochenski Pag 8, 1993*

9 La capacidad de un lenguaje estructurado de consulta (SQL), es característica de la mayoría de los sistemas cliente/servidor.

10 El servidor de base de datos, debe proveer seguridad y protección de datos.

DESCRIPCION FUNCIONAL

De manera general, para iniciar la comunicación entre un cliente y un servidor es necesario establecer una sesión. Por lo tanto, el servidor debe estar esperando ó "escuchando" que algún cliente trate de establecer una sesión, esto quiere decir que un cliente trate de "hablar" pero si no es "escuchado" la comunicación fracasará. Es muy posible que por algún momento el servidor también "hable" y que el cliente "escuche", pero esto sólo ocurrirá cuando el servidor así se lo indique al cliente.

Un servidor también se reserva el derecho de establecer comunicación con uno o más clientes. Así, el servidor se encargara de atender a cada cliente y establecer los mecanismos que seguirá para la distribución de sus servicios. Un servidor define operaciones que son exportadas a los clientes, los clientes invocan estas operaciones para que el servidor controle el manejo de datos.

Típicamente, una aplicación cliente comenzará una transacción, ejecutará una o varias operaciones en el servidor y terminará la transacción. Lógicamente, los servidores están estructurados como un ciclo infinito. El servidor simplemente recibe los requerimientos de los clientes para invocar operaciones en favor de esas transacciones. Para implantar las operaciones que exporta, el servidor puede requerir de otro servidor o puede manipular sus propios datos.

Bajo este esquema, se reparte el proceso de una aplicación entre un front-end (cliente) y un back-end (servidor), algunas funciones que lo distinguen como se muestra en la tabla 1.2.

FRONT-END	BACK-END
- Diseño de formas	- Almacenamiento
- Presentación	- Seguridad
- Lógica de la aplicación	- Administración de datos
- Manejo de datos	- Selección de registros
- Consultas	- Reorganización de la BD
- Menús	- Indexación
- Utilerías	- Ordenamientos
	- Actualizaciones en lote

Tabla 1.2. Comparación de funciones: Front-end vs Back-end.

SISTEMAS ABIERTOS "OPEN SYSTEM"

Muchas organizaciones han instalado una variedad de plataformas, y en cada una de ellas manejan información necesaria para la vida de las organización. El reto de los sistemas abiertos y con ellos los estándares, está en hacer que estas diferentes plataformas trabajen de manera conjunta.

Un sistema abierto es oquel basodo en un ambiente independiente a fabricantes, cuyas especificaciones son aceptadas, disponibles y estandarizadas. En estos sistemas los usuarios pueden mezclar hardware y software de diversos fabricantes en redes muy diversas. Los sistemas abiertos son de crucial importancia en ambientes de cómputo distribuidos ya que estos son construidos con sistemas de diversos proveedores de computadoras, redes, manejadores de base de datos y aplicaciones.

Un sistema abierto debe ofrecer : ***Interoperabilidad, Portabilidad e Integración.***

Interoperabilidad Significa que dos sistemos pueden trabajar conjuntamente a través de interfaces bien definidas, par ejemplo, una oplicación desarrollada en el sistema operativo UNIX puede interactuar con oplicacianes para los sistemas operativos OS/2 y MVS. Estas aplicacianes deben tener interfaces que permiten intercambiar información.

Portabilidad Significa que las aplicacianes deben ser independientes de las plataformas, de tal manero que combios que sufran las plataformas existentes no signifiquen modificaciones a las aplicaciones. Las oplicaciones desarrolladas en una plataforma deben ser transportadas y utilizables en otras plataformas.

Integración Se refiere a la facilidad con la cual un sistema puede ser usado. Un sistema integrado básicamente minimiza el esfuerzo necesario para utilizarlo.

Los estándares san necesarios para hacer a los sistemas abiertas una realidad. Un gran numero de estándares han sida desarrollados para cumplir con interoperabilidad, portabilidad e integración. Estos estándares son desarrollados por ISO ("Organización Internacional de Estandares, "International Standard Organization,"). Los estándares TCP/IP y XWindows son claros ejemplos de sistemas abiertos.

1.1 TENDENCIAS DE MERCADEO HACIA CLIENTE/SERVIDOR

La reingeniería de ubicación de las aplicacianes de misiones criticas, anteriormente eran colocadas en los equipos que se disponían, la tecnología cliente/servidor permite ahora migrar la aplicaciones a plotaformas más adecuados, siendo que estas deben ejecutarse en un medioambiente donde sean más eficientes. El modelo cliente/servidor permite separar tareas y estas desarrollarse en individuales plataformas, donde los desarrolladores deben revisar todas las toreas de una aplicación y determinar entonces la localizoción para su procesamiento, en un mainframe, minicamputadar, servidor o cliente. Ver figura 1.8.

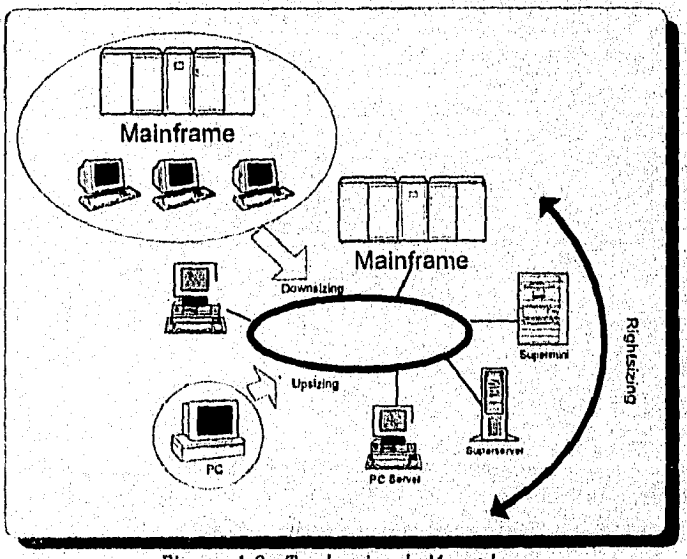


Figura 1.8. Tendencias de Mercadeo.

1 DOWNSIZING.

Consiste en migrar aplicaciones completas o funcianes de estas, de mainframes centralizados a plataformas menores buscando así acercar la aplicación al usuario final.

Downsizing toma este nombre de las técnicas de "adelgazamiento" de las sistemas madernos de producción *"Todo lo que no da valor agregado a la producción debe eliminarse, lo que se llama producción esbelta y que surge desde 1970"*[2].

En Estados Unidos, se realizó una entrevista[3] a 300 compañías por DATAQUEST, donde el 70% estaban en proceso de evaluación ó migración de aplicaciones ; más del 50% utiliza los equipos mainframe sola como servidor de archivos de base de datos. Gartner Graup (Stanford, Cannectiucut)[4] pronostica que para 1996 estarán instaladas cerca de cinco millanes de LAN's (Redes de área local) cerca de 160 millones de PC's y 75% de todas las empresas estarán conectadas por una LAN.
El beneficio patencial de utilizar una estrategia Downsizing es un menor costo, pero también existe factares camo la mejora del tiempo de respuesta, decremento en el tiempa de desarrollo de sistemas, incremento de flexibilidad, mejor control e implementación de cambios de estrategias en procesos de flujo de trabaja.

Sin embargo antes de tamar una decisión al llevar a caba esta estrategia, es necesario, coma la recamienda Theadora P. Klein asesor de la firma Boston Systems Groups, Inc. plantearse las siguientes preguntas:

- ¿La aplicación es departamental, divisional o involucra a toda la empresa?
- ¿Cuál es el tamaño de la base de datos y como es accesada?
- ¿La aplicación es fundamentalmente autónoma?
- ¿Qué tan familiar es la nueva tecnología a los usuarios y al equipo de sistemas de información?
- ¿Los datos de la aplicación son altamente confidenciales?
- ¿Qué nivel de tiempa perdida del sistema puede ser tolerado?

2 UPSIZING.

Este proceso se da cuanda una aplicación ha rebasado las capacidades del actual medioambiente donde se desenvuelve y es necesario migrar la aplicación a un media más poderoso, donde las necesidades de proceso de la aplicación queden cubiertas, tamando en cuenta que no llegue a interrumpir la utilización de la aplicación por parte de las usuarios.

El medio ambiente puede ser ampliado de varias farmas para aumentar sus capacidades, las cuales incluyen:

- Incrementa de memoria y almacenamiento en el servidor .
- Acceso al espacio del disco dura (swapping)[5] para darle más poder de proceso al servidor.
- Adicionando procesadores al servidor.
- Actualizanda el software y el cableado (hardware) de la red para hacerla más robusta.

Tal vez exista un mínimo desligamiento hacia los usuarios, por eso es recomendable, siempre que seo posible implantar sistemas abiertos tanta en hardware como en software.

---

[2] SALDIVAR JUAN, OP CIT.
[3] CLIENTE/SERVIDOR COMPUTING, Dauna Travis, Dowsizing, pay 8-9.
[4] IMPLEMENTING PRODUCTION-QUALITY CLIENT/SERVER SYSTEM, Barbara Bachenski, Pdg. 74
[5] Swaping: Transferencia de procesos de memoria RAM a disco y viceversa.

3 RIGHTSIZING.

Consiste en migrar aplicaciones completas o funciones de estas a los sistemas que respondan mejor a las necesidades de las mismas[6]. Este proceso es posible gracias a la flexibilidad ofrecida por los sistemas abiertos. En esto proceso se puede combinar las ventajas del mainframe con las de computadoras de escritorio.

La principal ventaja, radica en que permite tener una mejor visión que oprecie cuales aplicaciones no deben ser retiradas de los mainframe: por ejemplo, oquellas en las que se requiere del procesamiento de grandes volúmenes de transacciones provenientes de miles de usuarios concurrentes en diferentes plataformas. Por otra parte, existen aplicaciones que requieren de interfaces gráficas (GUI) para mejorar la productividad del usuario final y cuya implementación es propia de sistemas de escritorio.

4 SMARTSIZING.

La estrategia de Smartsizing, esta basada en la reingeniería del proceso de negocios, en contraste con el "Downsizing", la cual reimplementa los existentes sistemas automatizados sobre plataformas más pequeñas enlazadas por una red local (LAN), para disminuir costos e incrementar la productividad. Esta implica el usa de la tecnalogía de la información, que puede hacer que el proceso del negacio sea más eficiente e incrementar las ganancias. La reingeniería de negocios se enfaca a usar tecnología para modernizar tareas internas de fluja de trabajo, camo solicitudes de ordenes y necesidades de los clientes. La información tecnológica puede ser usada para incrementar la satisfacción de los clientes y los productos pueden ser desarrolladas más rápidamente.

1.2 INFRAESTRUCTURA DE SOFTWARE CLIENTE/SERVIDOR

El término cliente/servidor, en primera instancia es un termino complejo y abstracto, de la misma forma tanto las empresas de cómputo, como los especialistas en sistemas hacen referencia a él, como un modelo a como una arquitectura.

En este punto, consideramos impartante puntualizar la diferencia entre modelo y arquitectura.

En un sentido estricto, de acuerda a la definición del diccionario:

MODELO. " Es un objeto que se reproduce o se imita. Representación de alguna cosa en pequeña escala. Técnicamente es la construcción de una o varias piezas para hacer el molde en el cual se vaciarán los objetos".

En tanto que:

ARQUITECTURA. "Arte de proyectar , construir y adornar. Farma o estructura"

Par lo tanto cliente/servidor, es la canjunción de varias piezas que primeramente se diseñarán a escala para posteriormente reproducirse como un esquema digno de imitarse. Pero, ¿Dónde esta la interconexión de componentes ?. Desde luego, que aquí entra la infraestructura de comunicación entre las elementos o piezas adicionales a su ubicación para construir el sistema.

[6] *Parra Ellia: Mainframe VS PC: No ala guerra frontal , personal Computing Mexico, numero 49, pag 42-49.*

Podemos decir que la infraestructura cliente/servidor es la integración distribuida y abierta de un sistema en red, con los recursos, medios y aplicaciones, que definidos modularmente en los servidores, administran, ejecutan y atienden peticiones de los clientes, todos interrelacionados física y lógicamente, compartiendo datos, procesos e información, estableciendo así un enlace de comunicación transparente entre los elementos que conforman la estructura.

Los elementos que conforman la estructura cliente/servidor son : ***el cliente***, ***el servidor*** y ***software de comunicación***, ver figura 1.9.

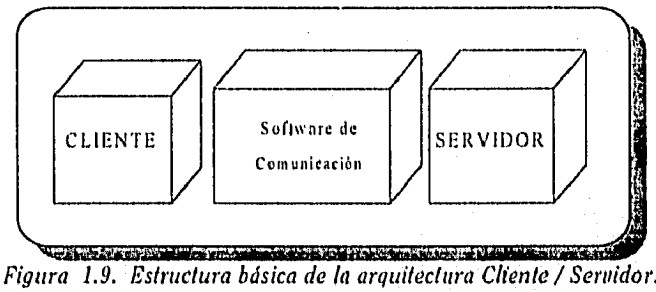


Figura 1.9. Estructura básica de la arquitectura Cliente / Servidor.

La computación cliente/servidor no se basa solo en los componentes del hardware, sin menospreciar sus cualidades, lo que hace posible esta tecnología es el ***software***. Al dividirse las tareas entre el cliente y el servidor, pueden ser procesadas en diferentes: plataformas, sistemas operativos y protocolos de comunicación, teniendo como resultado un desarrollo más rápido de la aplicación donde están envueltas. Entonces el usuario podrá accesar los datos con una interfaz gráfica agradable, que despertará su interés e invitación y sin la necesidad de capacitaciones exhaustivas.

EL CLIENTE: Es un consumidor de servicios provistos por uno o más servidores (acceso a la base de datos, servicio de impresión, correo electrónico), desarrolla parte y en ocasiones toda la lógica de la aplicación, brinda servicios de presentación al usuario a través de una interfaz gráfica.

EL SERVIDOR: Es diseñado con el objetivo de dar servicio, sus capacidades de hardware son mayores a las del cliente generalmente a través del DBMS que radica en él, puede recuperar, actualizar y almacenar los datos. Administra los recursos de la red, dispositivos y demás servicios con los que cuente.

SOFTWARE DE COMUNICACIÓN: Por medio de él se realiza la comunicación entre el cliente y el servidor, administra el flujo de datos a través de la red, permite la comunicación hacia otros sistemas (por medio de ruteadores, puentes y gateways). Es responsable de detectar problemas y recuperación de datos por colisiones.

Una aplicación envuelve varias tareas las cuales pueden ser llevadas a cabo en el modulo cliente o del servidor, con el fin de entender mejor como se realiza una aplicación la dividiremos en seis tareas, las cuales son :

1. **Interface para el usuario.** Cuenta con un dispositivo que acepta entradas del usuario y las despliega en la lógica de presentación. Este dispositivo puede simplemente desplegar caracteres recibidos de un teclado o un dispositivo que reciba señales de un mouse.

2. **Lógica de presentación.** Controla la interacción entre el usuario y la computadora. Esta es la especificación que el usuario realiza cuando selecciona un menú de opciones, un botón de selección o escoge un elemento de una lista.

3. **Lógica de aplicación**. Es una colección de decisiones, cálculos y operaciones que la aplicación puede llevar a cabo. Esta puede incluir los cálculos del pago a empleados, la decisión de aceptar una orden de compra, la evaluación de cargar una aplicación o el procedimiento de realizar una transacción la cual ha sido transferida.

4. **Lógica de datos**. Es la expresión de las operaciones desarrolladas en la base de datos que son necesarias para realizar la lógica del negocio. Como la mayoría de los manejadores de bases de datos se basan sobre el modelo relacional, las expresiones son instrucciones en SQL como: Select, Insert y Update.
5. **Servicio de datos**. Son las acciones que el DBMS toma para realizar la lógica de datos, incluyendo la manipulación, la definición y la transacción de los mismos. Para manipular datos el DBMS típicamente compila las instrucciones en SQL.

6. **Servicio de archivos**. Accesa el disco y trae los bits que se convierten en datos cuando son vistos a través del DBMS. Los servicios de archivos son usualmente realizados por funciones del sistema operativo.

En el ambiente cliente/servidor, una aplicación se puede almacenar, dentro de los clientes, en los servidores o parcialmente en ambos dependiendo de las características y capacidad del equipo, así como de las necesidades particulares en cuanto al accesa a datos e información; de esta manera, es responsabilidad del área de Sistemas así como de los diseñadores de la red, definir "desde la etapa de planeación", cómo serán estructurados las aplicaciones y componentes dentro del sistema en red.

A continuación se muestra una clasificación de alternativas, para organizar las aplicaciones en base al acceso a la información como se muestra en la figura 1.10.

Presentación Distribuida: Se separan las datos, procesos y presentación en el servidor y solo una parte de la presentación se ejecuta en el cliente. Donde varios usuarios pueden accesar los mismos datos desde los diferentes servidores y cada uno visualiza exclusivamente la información que solicita. Por ejemplo, la consulta e impresión de las listas de proveedores que maneja el departamento de compras de la empresa, adicionándole los datos personales del usuario. Este tipo de estructura hace uso del poder gráfico de los clientes y permite compartir las aplicaciones existentes.

Presentación Remota: Se concentran los datos y procesos en el servidor y la presentación se ejecuta en el cliente. Aquí se aprovechan las capacidades particulares de cada cliente, de tal manera que el usuario por su parte, puede accesar los datos que requiera, manipularlos con sus propias herramientas creando y/o modificando libremente la presentación y salida de la información. Por ejemplo, un vendedor de seguros realiza una consulta a la base de datos (servidor) de las diferentes primas que se tienen registradas para las automóviles modelo 95, al visualizar la información, utiliza su procesador de textos para una mejor presentación y así obtiene los reportes requeridos.

Función Distribuida: Se tienen los datos y procesos en el servidor, pero parte de los procesos conjuntamente con la presentación, se ejecutan con los recursos propios del cliente. Aquí el principal aspecto es el uso y manejo de ciertas subrutinas y/o programas dentro de lo servidores, mismas que pueden ser utilizadas tantos veces como sean requeridas, sin tener que reescribirlas o particularizarlas para cada cliente. Por ejemplo, se puede solicitar la consulta de las cifras de ventas por los últimos dos trimestres, a las diferentes subsidiarias (bases de datos), dónde internamente se efectúa un proceso de selección de datos, el concentrado de información se recibe en el servidor local y a su vez el cliente ejecuta el proceso correspondiente para obtener los datos de su empresa, así mediante ciertas instrucciones predefinidas conjunta la información y obtiene el informe requerido.

Acceso Remoto de Datos: Los datos se almacenan en el servidor, de esta forma los procesos y la presentación se ejecutan desde el cliente. Básicamente es el modo de acceso utilizado en el software de correo electrónico, dónde se solicitan consultas de información a bases de datos remotas, de la misma forma que se recibe y se envía información de otros clientes. Por ejemplo, consultas a las bibliotecas de las diferentes universidades nacionales y del extranjero.

Base de Datos Distribuida: Todos los datos se concentran en los servidores, pero cierta porción de información así como las procesos y presentación se accesan con los recursos del cliente. Este tipo de estructura permite la comparación de datos a cualquier nivel dentro de la empresa, dónde cada usuario puede manipular la información de su área o departamento y compartirlo con la de otros en forma simultánea. Por ejemplo, una empresa puede tener una base de datos única dónde concentre la información relacionado a las existencias en almacén de sus productos, de tal forma que las altas, bajas y cambios se efectúan en línea y al ser consultadas por cualquier persona dichas modificaciones se visualizan por todos los demás usuarios.

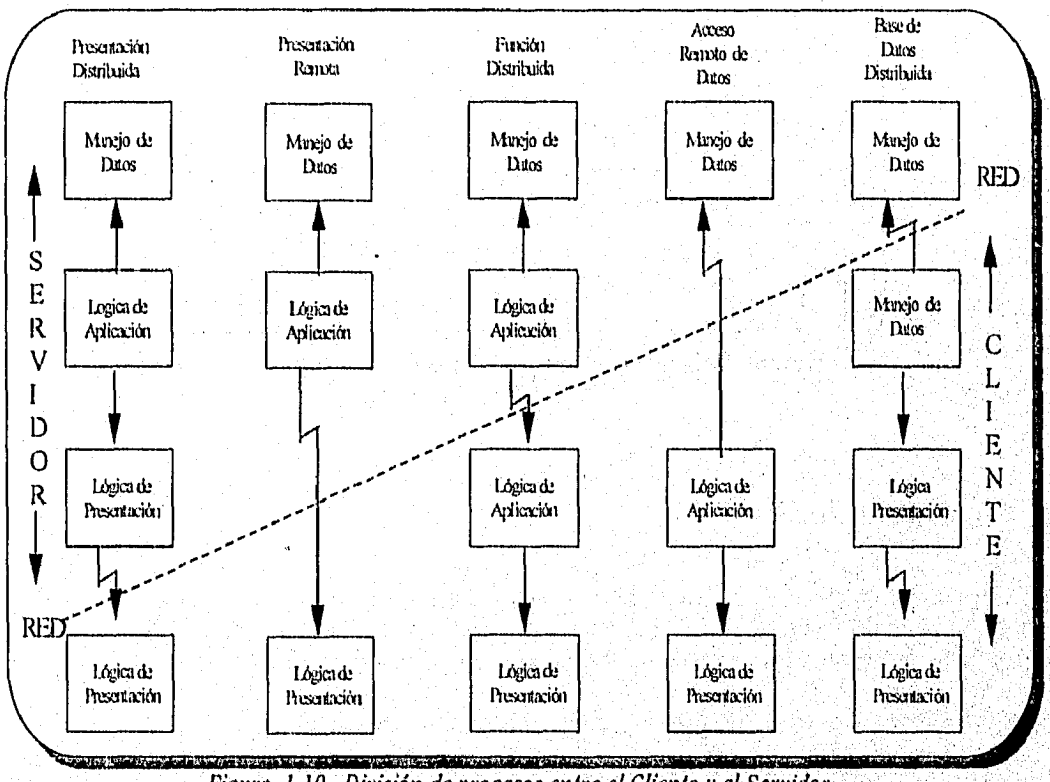


Figura 1.10. División de procesos entre el Cliente y el Servidor.

GUI

El medioambiente gráfico GUI (Graphic User Interface, "Interface Gráfica del Usuario"), contribuye a que sean más atractivos los sistemas cliente/servidor, porque hacen que estos sean mas fáciles de usar e incrementan la productividad. Las características GUI posibilitan al usuario a ser más productivo con menos entrenamiento, esto es porque la interfaz gráfica es más intuitiva. Estudios realizados para comparar la productividad y el aprendizaje utilizando aplicaciones con características GUI contra

aplicacianes de tipo carácter tradicional, han demostrado que un medioambiente gráfica aumenta más del 200 %[7] las capacidades del usuaria.

Las interfaces gráficas tiene cualidades de intercalar la atención de tareas, multitarea e intercambio de datas entre aplicaciones, entre otras. Las características GUI no son necesarias para determinar que exista un sistema cliente/servidor, pero son muy usadas en este tipa de sistemas, y si aún no forman parte integral, están sienda reconocidas camo altamente deseables.

El desarrollo de ambientes GUI consiste en la programación de manejo de eventos, dande el código responde a acciones del usuario; sin embargo, al crear un medioambiente gráfico es importante tomar en cuenta estándares abiertos e interfaces independientes a cualquier plataforma, camo se ilustra en la figura 1.11, ya que si un sistema cliente/servidor se crea en medioambientes abiertos y la interface gráfica que se utilice no lo es, tendrá como resultada un sistema propietario en la plataforma en que se creó.

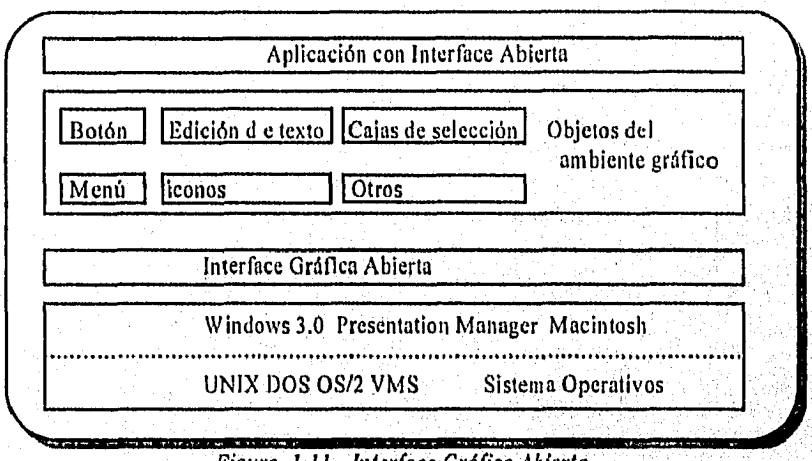


Figura 1.11. Interface Gráfica Abierta.

1.3 EL CLIENTE

La plataforma ideal en una arquitectura cliente/servidor opera en un medioambiente de sistemas abiertos, usando una disciplina de solicitudes de servicio, que esta basado en normas bien definidas, esto habilita la interacción de múltiples plataformas hardware y software.

Como utilizamos el termino **usuario**, lo definiremos para evitar ambigüedades.

En un sentido estricto *usuario en general*, es la persona que utiliza la computadora con cualquier fin.

Usuario de desarrollo: Es la persana que ocupa la computadora y utiliza programas que le permiten crear una aplicación (desde un programa especifico hasta un sistema).

Usuario final: Es la persona que ocupa la computadara, a través de una aplicación (sea desarrollada por el área de informática o por un praveedor de software).

El Software que reside en el cliente representa la parte del "Front-End" en una aplicación cliente/servidor, permite al usuario final interactuar con la computadora y contiene una parte de la lógica de la

[7] *Carners Group, 1994.*

aplicación. Técnicamente hablando las más importante característica es que hace un requerimiento de servicio al modulo del servidor. La mayoría de los módulos cliente cuentan actualmente con interfaces gráficas (GUI), que facilitan la interacción entre el usuario y la computadora.

Características del Cliente:

- Es el medio de enlace y/o comunicación entre el usuario y el computadora.
- Es la entidad que requiere o solicita el servicio.
- Requiere el uso de los recursos del computadora para realizar cualquier actividad.
- Es el medio por el cual se envía la solicitud y se reciben los resultados o notificaciones del servidor.
- Contiene la interface gráfica (GUI).
- Puede interactuar con uno o varios servidores.
- Es el responsable de mantener el diálogo con el usuario.
- Su costo comparado con un servidor es considerablemente más baja.

Funciones principales del cliente :

- Inicia y termina la solicitud.
- Permite el manejo de la pantalla y ventanas.
- Presenta la información y/o datos.
- Interpreta los comandos.
- Maneja procesos de ayuda.
- Recibe las entradas provenientes del mouse , teclado y touch-screen.
- Permite controlar las cajas de diálago.
- Habilita el manejo de multimedia (si es el caso).

En el modelo cliente/servidor, todos los servicios que el usuario necesita pueden ser accesado por uno computadora de escritoria (CLIENTE). Como correo electrónico, hojas de cálculo, procesadores de palabras, presentación de gráficas, y los datos corporativos. Así como el uso de dispositivos de almacenamiento masivo, cd-rom, etc.

Los facilidades GUI, para realizar intercambio dinámico de datos y unión dinámica de datos, entre aplicaciones. Todos estas tareas son realizadas por el modulo del Cliente, el usuario no notará que puede accesar dispositivos remotos, o almacenar datos en un dispositivo que no se encuentre físicamente en la computadora que se esta utilizando. El cliente hará todo esto de una forma transparente para el usuario, de tal manera que llegue a pensar que todo el sistema sola sea la computadora que esta utilizando.

HARDWARE

En la computadora que actúa como cliente, y donde reside el front-end, se ejecutan programas que son responsables de la presentación y manipulación de los datos. Estos programas generan requerimientos de datos que son solicitados al servidor, en ocasiones el servidor actúa como un cliente y se denomina agente, ya que solicita datos de otro servidor. La computadora que actúa como cliente debe ser bastante rápida para ejecutar los programas que presenten y manipulen las datos recibidos.

En el presente trabajo, nos referimos al termino *computadora del cliente* como una computadora de escritorio o computadora personal en la cual trabaja el usuario, ya sea que opere en ambiente Macintosh, DOS, OS/2 o estaciones de trabajo en UNIX.

Las computadoras más comunes que actúan como clientes están basadas en un microprocesador 386 con una velocidad desde 33 Mhz o mayores y manitar VGA[8].
La memoria y almacenamiento para que una computadora actúe como cliente eficientemente, depende directamente de la complejidad de la aplicación, para una aplicación basada en un mainframe en una arquitectura cliente/servidor, típicamente se necesitará poco o nada de proceso local y los requerimientos de hardware para una plataforma GUI son usualmente básicas. En un ambiente cooperativo, donde la aplicación se encuentra basada en el cliente, se requiere bastante memoria para llamar y ejecutar la aplicación lógica, alta velocidad de procesamiento, más poder en el microprocesador y uso de memoria cache[9] (memoria auxiliar rápida).

Por lo tanto se concluye que los requerimientos hardware están basadas en el procesamiento que requiere el software elegida o desarrollada, en la plataforma GUI conveniente y sobre todo cubrir las necesidades de los usuarios.

SOFTWARE

Antes de listar el software que existe para el modulo cliente, explicaremos por que es difícil hacer una división exacta.

Las Herramientas para usuarios finales que existen hoy en día están tendiendo a ser más sofisticadas y simples para crear aplicaciones que abarquen a toda la empresa. La compañía Research Inc., con sede en Cambridge MA, predice que el mercado de herramientas para Frant-Ends, en E.U. crecerá de $475 Millones de dólares en 1992 a $2.6 Billones de dólares en 1996[10]. Así misma, se irán incorporando capacidades orientadas a objetos, que actualmente están empezando a aparecer.

Un ejemplo es la herramienta, para el usuario final, Quest de Gupta Tecnologies, con características sofisticadas como consultas, reportes, actividades de tablas, vistas al catalogo de actividades, etc. El usuario consulta la base de datos y recupera la información que requiere, contando con la facilidad de filtrar, ordenar y agrupar los datos , crear complejos reportes, visualizar las tablas existentes, columnas, índices y actualización a las bases de datos.

Por las características mencionadas, Quest puede ser considerada como una herramienta para desarrollo de aplicaciones totales. Sin embargo, Gupta provee una herramienta completamente orientada para el desarrollo de aplicaciones totales, SQLWindows, que más adelante explicaremos en que consiste y que posee todas y más características de las que tiene Quest.

INTERFACES GRÁFICAS		SISTEMAS OPERATIVOS	
Windows	Motif	MS-DOS	AIX
Macintosh	Pen Look	Windows NT	AIUX
Presentation Mananer	NeXTSTEP	Windows 95	HP-UX
Wabi		WIndows NT	Mach
		OS/2	OSF/1
		Workplace OS	SCO-ODT
		Taligent	Solaris
		DR-DOS	Ultrix
		UnixWare	

Tabla 1.3. Software para el Cliente.

[8] VGA (Video Graphics Array) : Monitor estándar adoptado por IBM,que provee una resolución media para gráficas y texto. Este es ahora considerado como el estándar mínimo para una PC.
[9] Memoria CACHE: Memoria en la que se guardan los contenidos activos de la sección de la tabla de páginas en uso.
[10] Client/Server Computing , Dawna Travis D. Pag 37.

INTERFAZ GRÁFICA

La mayoría de las aplicacianes cliente/servidor usan una interfaz gráfica (GUI), en la cual se presenta al usuario la información en ventanas en forma de imágenes llamadas iconas, las cuales pueden ser activados a través de la pantalla (Touch-screen), por el ratón o el teclado. Una entidad en el ambiente GUI, es cualquier área en la pantalla que puede ser seleccionada y que tiene cama resultada una acción algunas funciones que el usuario puede llevar acabo en un ambiente GUI san :

- Activar una o varias tareas (ventanas) cuanda la desee.
- Cambiar el calor, tamaña y posición de una ventana.
- Sabre poner ventanas.
- Minimizar una ventana y esta es representada con un icono.
- Minimizar una ventana, reactivarla y acupar el área de la pantalla en primer plano, cuanda esta halla sida minimizada.
- Cada ventana contiene un texto en la parte superiar que indica que tarea representa.
- Cada ventana contiene barras de desplazamiento en la parte derecha e inferiar, para la información que se despliega.

En ambiente GUI existen manejas de eventos, los cuales incluyen, entre atros:

- Ratón
- Teclado
- Actualización de ventanas
- Cambio de tamaña de ventanas
- Activación y desactivación de ventanas
- Menú
- Inicio y conclusión de aplicaciones.

Cada medio ambiente GUI tiene sus interfaces de programación de aplicaciones (API), que consisten en un conjunto de rutinas programados, que son usadas para praveer servicios y unión de diferentes tipos de software, como funciones de crear y cerrar ventanas, acceso a servidores de base de datos y servicios de red, ver figura 1.12.

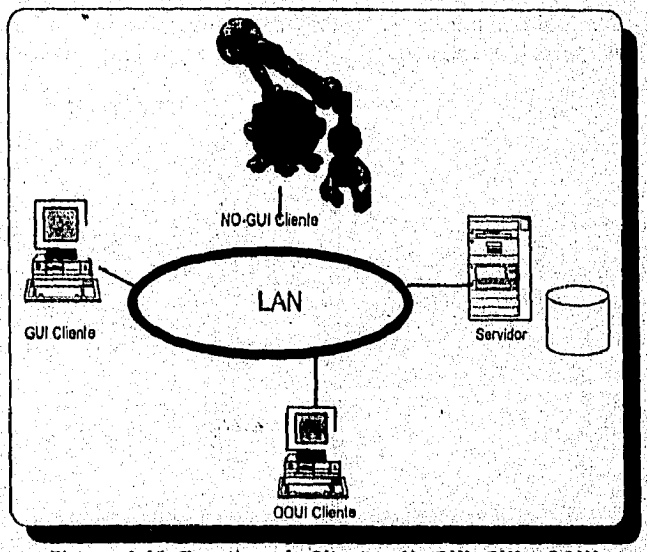


Figura 1.12. Tres tipos de Clientes: No GUI, GUI y OOUI.

1.4 EL SERVIDOR

Es la entidad física que provee un servicio, se encuentra la ejecución de procesamiento de datos, aplicaciones, maneja de la información y recursos. En el reside el BACK-END, que es la parte destinada a recibir las solicitudes del cliente y donde se ejecutan las procesas.

Un servidor es considerado, por muchas, como la conjunción de hardware y software que ejecuta tareas específicas.
Sus principales características son:

- Responde a las peticiones de los clientes.
- Tiene gran capacidad de almacenamiento y rapidez.
- Provee el uso de los recursos y servicios a los clientes.
- Es el administrador de los recursos de la red
- Tiene capacidad de procesamiento transaccional.
- Puede actuar como cliente de otros servidores.
- Contiene los datos, programas, aplicaciones, software e información.
- Realiza procesos como accesar, organizar, almacenar, actualizar y manejar datos o recursos compartidos.

La naturaleza y el grado de servicios esta definida por los objetivos de la implantación de un sistema cliente/servidor, un servicio puede requerir un mínimo de computación en el servidor, como es el caso de servidores de archivos o de impresión, necesitar de procesamiento intensivo, como los servidores de base de datos o de imágenes. Ningún servidor inicia la conversación, con un cliente tampoco atiende directamente interfaces con el usuario final, simplemente actúa como: repositorio de datos servidor de archivos, de conocimientos servidor de base de datos, o como un prestador de servicios de impresión, envío de fax, correo electrónica, etc.

Un servidor si puede iniciar la conversación con otro servidor, el cual es denominado agente, solicitándole un servicio, siendo transparente para el usuario.

Un servidor ideal, hace transparente todo el esquema cliente/servidor al usuario. Un cliente al comunicarse con el servidor, no desea saber que plataforma de hardware y software intenta accesar, así como la tecnología de comunicación que hace posible el enlace. Para ello es deseable y recomendable que un ambiente de servidores múltiples estén comunicados entre si, para proporcionar un servicio al cliente, transparente de múltiple existencia y comunicación entre servidores, ver figura 1.13.

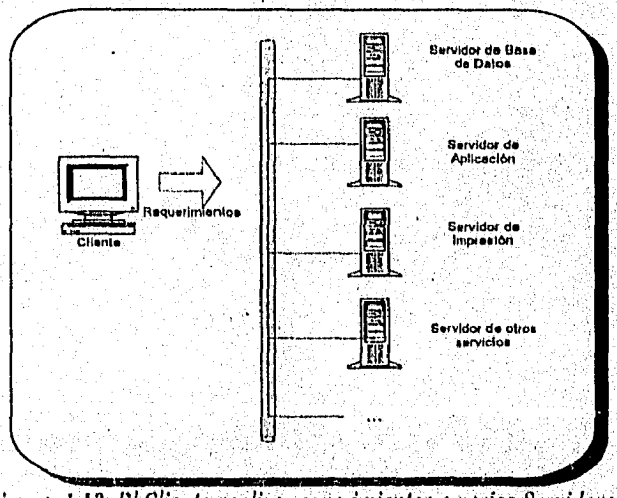


Figura 1.13. El Cliente realiza requerimientos a varios Servidores.

El proceso del servidor es **reactivo**, es decir, realiza una acción en base a una instrucción previa; realiza una función posterior a una petición y/o la ejecución de una transacción requerida por el cliente que también puede ser otro servidor. Además, el hecho de que en el servidor resida la información de la organización, permite incrementar la seguridad, pues establecen mejores controles de acceso a la misma.

Existen elementos que deben ser considerados en la adquisición del hardware del servidor, que son:

1 **Confiabilidad:** ¿ Qué ocurre si falla el sistema ? ¿Que ocurre en el proceso de trabajo de la empresa en el tiempo de solucionar la falla?.

2 **Eficacia:** ¿ Qué tan rápidamente el sistema puede dar servicio después de una falla? Para asegurarse de tener un sistema altamente eficaz, algunos sistemas cuentan con rutinas de autocorrección, procesadores continuos, hardware con fallas tolerantes, alarmas que detectan problemas antes de convertirse en un caos y poder continuar trabajando, aunque más lentamente. Algunos pueden ser inicializados desde lugares remotos.

3 **Flexibilidad y escabilidad:** ¿Qué tan fácilmente puede ser expandido el servidor si el proceso necesita crecer? Debe soportar adición de clientes, como es el caso de Base de Datos distribuidos. Además, procesadores que aumenten la potencialidad del servidor.

Existen diversos tipos de servidores mismos que se clasifican en base a su funcionalidad, estos son denominados ***servidores dedicados*** ya que administran el uso de algún recurso en particular. Por ejemplos:

Servidor de bases de datos: Se encarga del manejo, almacenamiento, conservación y seguridad de los datos; también puede realizar la validación de las claves de acceso (password) para efectuar consultas.

Servidor de impresoras: Es el dispositivo encargado de administrar las colas de impresión direccionando los trabajos a los diferentes dispositivos de salida.

Servidor de aplicaciones: En este medio se almacenan y ejecutan las aplicaciones de software utizadas por los usuarios, evitando así la duplicidad de las mismas, permite un mejor control en la actualización de versiones y productos.

Servidor de respaldos: Este es una innovación propia del ambiente cliente/servidor, de gran utilidad ya que administra la ejecución de los respaldos en línea, asegurando que éstos sean efectuados, ya que si un dispositivo falla o no se encuentra disponible, automáticamente direccionará el proceso a otro para que se ejecute el respaldo correspondiente.

HARDWARE

CLASES DE COMPUTADORAS QUE ACTUAN COMO SERVIDORES.

La adquisición de un servidor para una empresa, depende del grado de confiabilidad que se requiere para el tipo de información que maneja, la complejidad de las aplicaciones, el número de usuarios y aplicaciones que requieren de otra(s) servidor(es). Los servidores son clasificados como:

- Microservidores.
- Super servidores.
- Servidores de base de datos.

- Servidores de medio rango.
- Servidores con fallas tolerantes.

Listar todas las computadoras disponibles en el mercado, sería ostentoso de nuestra parte, mejor optamos por mostrar una tabla comparativa entre microservidores y superservidores. Tabla 1.4.

	MICROSERVIDORES		SUPERSERVIDORES	
	MINIMO	MAXIMO	MINIMO	MAXIMO
Usuarios	2-16	16-64	64-128	128-1000
Costo	5,000 dlls	15,000 dlls	25,000-50,000 dlls	60,000-300,000 dlls
Procesador	386	2 x 386-486	Mips, Intel, Sparc	Mips, Intel, Sparc, RS/6000
RAM	8-16 MB	16-64 MB	32-128 MB	64-256 MB
Disco Duro	8-180 MB	180-840 MB	640 MB - 2 GB	1-32 GB
Sistema Operativo	DOS, OS/2, NETWARE	UNIX, OS/2, NETWARE	UNIX, SCO UNIX	UNIX, SCO UNIX

Tabla 1.4. Comparación de microservidores y superservidores.

SOFTWARE

Mientras que el usuario mira a la computadora que actúa como cliente, como todo el sistema, el servidor es el componente crítico, maneja la red, administra el acceso a los datos que residen en sus dispositivos de almacenamiento, genera datos para otras nodos y transmite los datos a los nodos apropiados. El cliente hace una solicitud y el servidor hace toda coordinación necesaria para cumplir esa solicitud.

Se puede definir ocho categorías de software que se ejecutan en un servidor, ver figura 1.14.

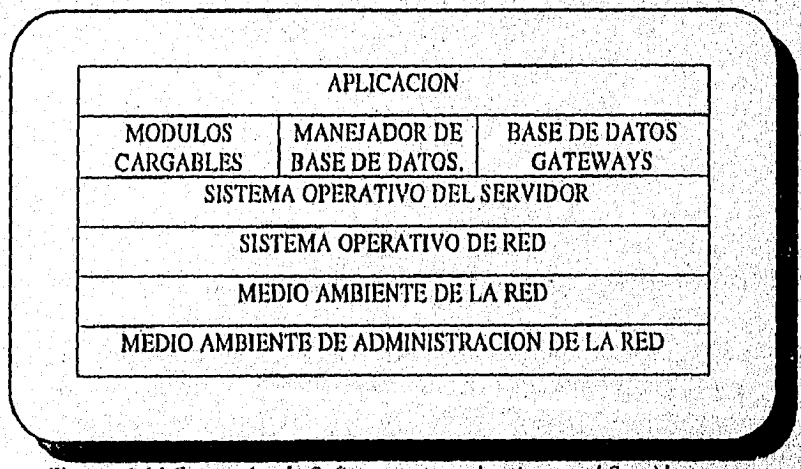


Figura 1.14 Categorías de Software que se ejecutan en el Servidor.

La aplicación desarralla requerimientos de datos los cuales san enviados vía el sistema operotivo de red a 'lo bose de datos gateway[11], al manejador de base de datos ó a los módulos cargables[12]. Estas traducen las solicitudes dentro del código de la computadara y pason al sistemo operativo. El sistema operativo del servidor es guiada por el medioambiente de la red, el cual moneja los requerimientos de servicio para soportar distribución de aplicaciones en medioambientes heterogéneos. La red y el sistema operativo debería ser mirodo como una entidad y la administración del manejo como un proceso único. Este concepto es desarrollada por Unix International Inc, (UI) en el software UI-Atlas.

La tabla 1.5. muestra algunos productos que existen en el mercado para residir en el servidor.

SISTEMAS OPERATIVOS DE RED	
NetWare	Pathworks
Lan Manager	Apple Share
Lan Server	PC-NFS
Vines	

SISTEMAS OPERATIVOS	
Windows NT	Macintosh
Windows 95	AIX
VM	HP-UX
OS/2	Unix
OS/400	OSF/I
MPE/iX	Ultrix
Solaris	

RDBMS	
Oracle	DB2, /2 , /6000
Supra	SQL/DS
Sybase	OS/400
Informix	Microsoft SQL S.
Ingres	Teradata DBC
Paradox	AllBase/SQL
Progress	ADABAS

OODBMS	
ObjetStore	Servio Gemstone
Objetivity/DB	HP OpenODB
Versant	UniSQL/M
Ontos	Ilustra
Matisse	Poet
SQLbase	Delphi
OS	SQLWindows

WORKFLOW	
WorkFlow	Flowlogic
Plexus Floware	FlowareSIS Route B.
IBM Image Plus	Wang OPEN/Image
Folio Views	Reach Workman
Image Enabled Net	View Start
Xorox in Concert	

MANEJADORES DE TRANSACCIONES	
Tuxedo	VIS/TP
Encina	Integris UniKis
TopEnd	CICS
ACMS	

Tabla 1.5. Software del servidor

1.5 SOFTWARE DE COMUNICACIÓN

Algunos profesionales del ámbito informático, utilizan el termino "middlewore", el cual crea más confusión que contribución, yo que es usodo indistintamente. Nosotros utilizaremos el término software de comunicación, para describir las copacidades de comunicación entre el cliente y el servidor.

[11] Base de Datos Gateway : Proveen acceso a Bases de Datos externas, un ejemplo es SQLBase de Gupta Tecnologies.
[12] Modulos cargables : Es software que son cargados en el servidor para apoyo del Sistema Operativo. Ejem NLMs(Netware Loadable Module) Se ocupa para soporte de protocolos de transporte, administración de base de datos o red.

LA RED

La arquitectura de la red define los protocolos, formato de los mensajes y estándares que se usarán. Una arquitectura de red robusta usa una estructura de capas, cada capa consiste en entidades compuestas por hardware y procesos de software, las normas y formatos para comunicación entre capas adyacentes son colectivamente llamadas una interface. Los protocolos son las normas y formatos para comunicación dentro de la misma capa , o través de diferentes dispositivos, incluyen formatos y el orden de los datos para su intercambio y cualquier acción para la transmisión y recepción de datos, como se muestra en la figura 1.15.

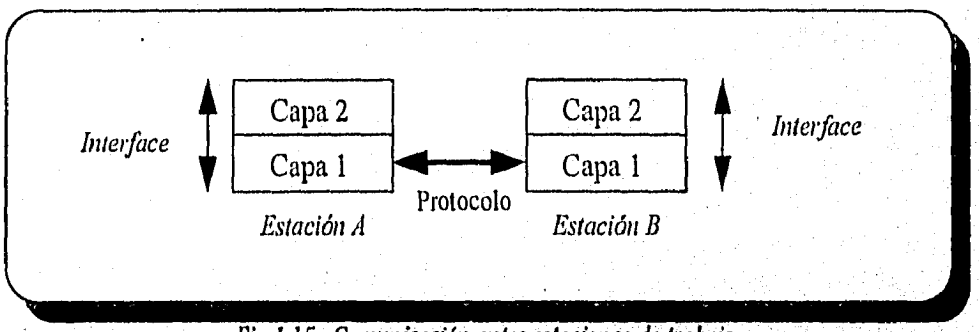


Fig 1.15. Comunicación entre estaciones de trabajo

Si dos estaciones transmiten exactamente al mismo tiempo, se provoca un choque, llamado colisión, los protocolos son usados para prevenir estas problemas por medio de normas de transmisión.

Cambios a un protocolo se pueden hacer sin tener que afectar a otras (funciones) en la pila (los protocolos de todas las capas). Cada capa solo necesita estar consiente de los servicios provistos por la capa inferior inmediata, resumiendo esto , es que diferente hardware y software pueden comunicarse siempre y cuando usen el mismo protocolo y formato de datos.

Conforme las redes locales (LAN,"Local Area Network") van creciendo en tamaño y complejidad, y como las instituciones van confiando en estas redes labores cada día más críticas, surge la necesidad de comunicarlas entre sí, en una misma ciudad, o en ciudades distantes, así se forma lo que comúnmente se denomina redes de área amplia (WAN," Wide Area Network").

En un sentido estricto, una red de área amplia esta compuesta por varias redes, en la que se conectan varias redes locales mediante dispositivos que permiten su conectividad local o remota, a pesar de que tengan diferente topología. Estos dispositivos pueden usar o no líneas telefónicas o servicios públicos de transmisión de datos.

Los puentes, ruteadores, y gateways nos permiten utilizar diferentes topologías y protocolos dentro de un solo sistema heterogéneo, cada uno de estos elementos tiene ventajas y desventajas, así como aplicaciones específicas, las cuales se explican en el punto 3.2 del capítulo III.

SISTEMAS OPERATIVOS DE RED

El sistema operativo es el corazón y alma de la red. El hardware del sistema proporciona los trayectorios de datos y las plataformas en la red, pero el sistema operativo es el encargado de controlar todo lo demás. La funcionalidad, la facilidad de uso, el rendimiento, la administración, la seguridad de los datos y la seguridad de acceso dependen del sistema operativo.

El sistema operativo de red, se engloba en dos componentes básicos. El sistema operativo de red del servidor mismo y el sistema de la estación de trabaja. El sistema operativo del servidor de red se ejecuta dentro del servidor y procesa todas los servicios, los componentes de la estación de trabajo se ejecutan en ésta, establecen la conexión con la red y el servidor, y controla el flujo de la comunicación.

El sistema operativo del servidor de red se puede dividir en cinco subsistemas básicos: el núcleo de control (control kernel), las interfaces de la red, las sistemas de archivo, las extensiones del sistema y los servicios del sistema, como se muestra en la figura 1.16.

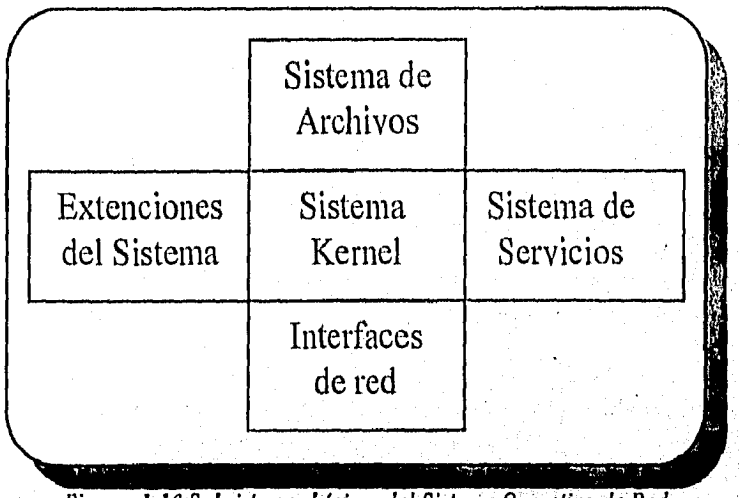


Figura 1.16 Subsistemas básicos del Sistema Operativo de Red

Los componentes principales del sistema operativa son desarrolladas en el punto 2.1. del capítulo II, como lo son: administración de archivos, administración de memoria, multitarea, etc.

Las extensiones que ofrecen los desarrolladores de sistemas cubren la administración de la red, las herramientas del sistema que atienden un margen de apoya de aplicaciones y los servicios de base de datos. Las bases de apoya SQL son un foca actual para las desarrolladores de sistemas, debido a que los sistemas de procesamiento distribuido se benefician de las recursas de datos centralizados

El enfoque alternativo para aquellos sistemas operativos que no están conscientes de la red, es atrapar la entrada/salida de la aplicación antes que esta entrada/salida llegue al sistema operativa local. El software que emplea este método, con frecuencia denominado el redirectar o shell, examina y envía la solicitud al servidar de archivos para su acción.

1.6 BONDADES Y DESVENTAJAS

De lo expuesto en el capítulo se resume que, el ambiente cliente/servidor ofrece las siguientes ventajas:

- ☆ Permite la integración de los recursos y elementos del sistema en red.
- ☆ Incrementa aún mas la productividad y el desempeño de la red.
- ☆ Permite el usa y comparación de las sistemas y recursas interconectadas.
- ☆ Incrementa la velocidad de respuesta al usuario.
- ☆ Evita el sustituir totalmente los equipos instaladas.
- ☆ Permite el ensamble de las aplicaciones heredadas (legancy systems) ya existentes con las aplicaciones y software que se están desarrollanda.
- ☆ Permite la integración abierta en plataformas multivendor.
- ☆ Reduce los costas de mantenimienta de saltware.

- ☆ Agiliza el tráfico en la red y tiempos de respuesta.
- ☆ Mejora el control de los procesos y recursos en red.
- ☆ Propicia un ambiente para la implantación de nuevas tecnologías.
- ☆ Incremento el nivel de calidad y obtención de información.
- ☆ Reduce tangiblemente los costos del sistema una vez implantado la arquitectura cliente/servidor.

Dentro de un ambiente idóneo y "perfecto" como el que se vislumbra bajo ésta arquitectura, también es importante mencionar su lado gris, ya que para migrar los sistemas de las organizaciones actuales a cliente/servidor, se requiere analizar qué tan redituable será dicha inversión en términos monetarios y de tiempo. Así mismo, las principales desventajas son:

- ♦ La migración e implantación es costosa ya que se requiere de servidores poderosos, equipos y recursos.
- ♦ Requiere necesoriamente de una infraestructuro de comunicación en red.
- ♦ Se requiere de software y aplicaciones adicionales a las ya existentes como parte del software de comunicación para conformar el nuevo sistema.
- ♦ Las aplicaciones para cliente/servidor son en ocasiones más complejas que las convencionalmente utilizadas.
- ♦ Aún no existen estándares específicas y reconocidos universalmente que reglamenten esta arquitectura.

Sin embargo, conforme la arquitectura vaya madurando seguramente estas puntos dejarón de ser desventajas, puesto que las expertos ya trabajan sobre ello paro desarrollar las productos que cubran dichas deficiencias.

Por otra parte, el combio hacia cliente/servidor no implica salamente combios técnicos y lo elección de productos entre diferentes morcas, también implica un cambio culturol y organizacional, ya que para que todo proyecto se realice exitasamente, debe antecederle el respaldo gerencial y la aprobación por parte de la dirección. Primeramente la gerencia de sistemos deberá exponer y convencer a los directivos de que cliente/servidor es la arquitectura que les redituará los beneficios empresoriales esperados. También deberán instruir a los usuarios en general para que una vez implantado, efectivamente hagan uso de los recursos y sistemos de información de una manero adecuado.

1.7 CLIENTE/SERVIDOR EN LA DIRECCION GENERAL DE INTERVENTORIA.

La Dirección General de Interventoría inició su equipamiento de hardwore desde 1993, con computadoras personales, 3 computadoras H.P. Vectra y 19 terminales, cuya función actual es proporcionar consultas a los bancos de datos del Centro de Procesamiento Nacional (CPN), ver figura 1.17.

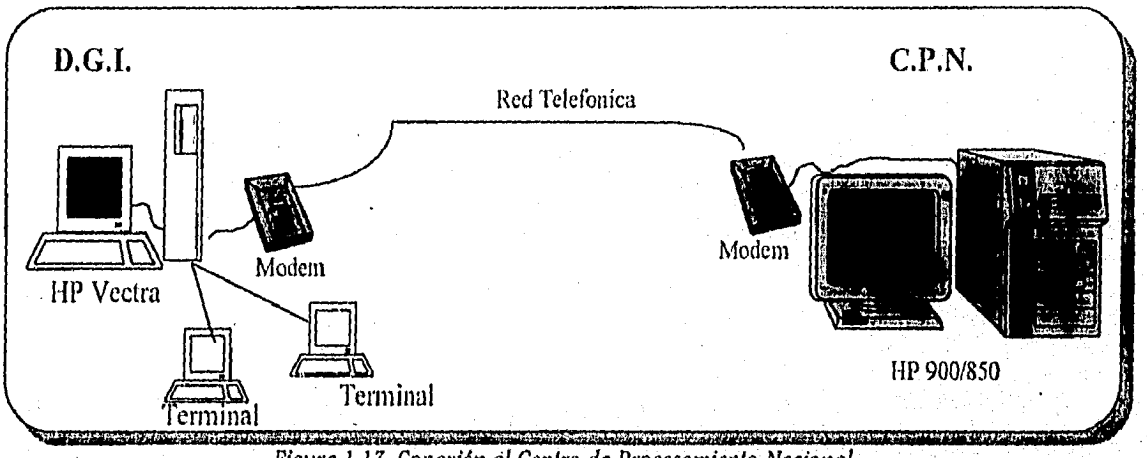


Figura 1.17. Conexión al Centro de Procesamiento Nacional.

En octubre de 1994 se presentó el proyecto "Automatización de Oficinas" que contempla dos etapas. En la primera, denominada "Fase I", se adquirieron: 35 estaciones de trabajo, un servidor Apollo 715/9000, se instaló un cableado de fibra óptica con topología de anillo y concentradores "HUB's" Plexnet. La tabla 1.6 muestra las características de los equipos instalados.

SERVIDOR	
H.P. 715/9000	
MEMORIA RAM	48 M.B.
DISCO DURO	2 G.B.
SIST. OPERATIVO	
UNIX U.X.	
LAN MANAGER	
MANEJADOR DE B.D.	
INFORMIX ONLINE	
INFORMIX STAR	
INFORMIX 4GL	
INFORMIX SQL	

CLIENTES	
H.P. VL 2 486	
MEMORIA RAM	8 M.B.
DISCO DURO	340 M.B.
SISTEMA OPERATIVO	
MS-DOS	
WINDOWS TRABAJO EN GRUPO	
HERRAMIENTA DE DESARROLLO	
SQL WINDOWS	

Tabla 1.6.- Características de los equipos instalados en "Fase I".

La creciente demanda de procesadores más potentes que se instalaron en la fase I, fue un cambio tipo UPSIZING, ya que las computadoras que se tenían se encontraban aisladas y los procesos que se ejecutaban habían rebasado sus límites.

La figura 1.18 muestra la arquitectura cliente/servidor que se instaló en la fase i.

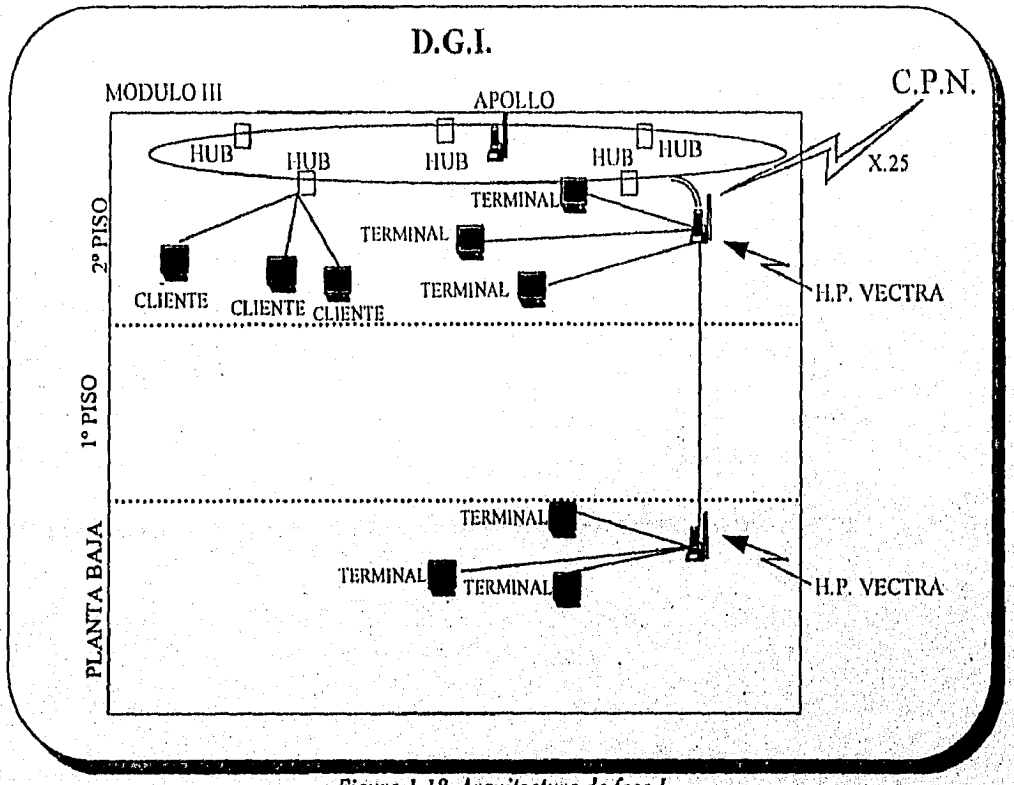


Figura 1.18. Arquitectura de fase I.

En Septiembre de 1995, se lleva a cabo la segunda etapa, denominada "Fase II", se adquieren 3 equipos servidores AT&T Globalyst 630 y 72 AT&T Glabalyst 520, cuyas características se muestran en la tabla 1.7.

SERVIDOR	
AT&T Globalyst 630	
MEMORIA RAM	64 M.B.
DISCO DURO	4 G.B.
SIST. OPERATIVO	
WINDOWS NT	
MANEJADOR DE B.D.	
SQL SERVER	

CLIENTES	
AT&T Globalyst 520	
MEMORIA RAM	8 M.B.
DISCO DURO	512 M.B.
SISTEMA OPERATIVO	
MS-DOS	
WINDOWS TRABAJO EN GRUPO	
HERRAMIENTAS DE DESARROLLO	
ACCESS	
VISUAL BASIC	

Tabla 1.7.- Características de los equipos instalados en "Fase II".

La figura 1.19 muestra la arquitectura en la fase II de Automatización de Oficinas que se encuentra instalada actualmente. Esta plataforma es considerada como el estándar en la Secretaría de Hacienda y Crédito Publico en el área de la Subsecretaría de Ingresos.

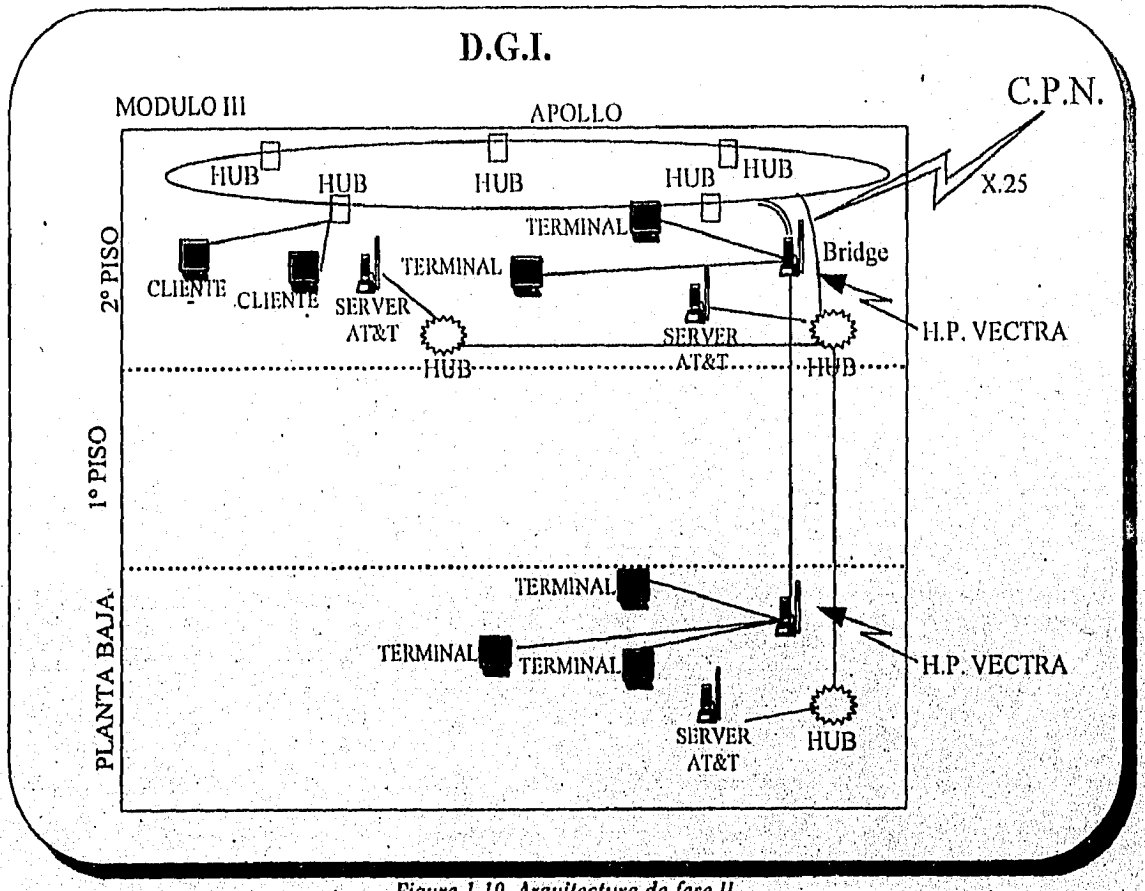
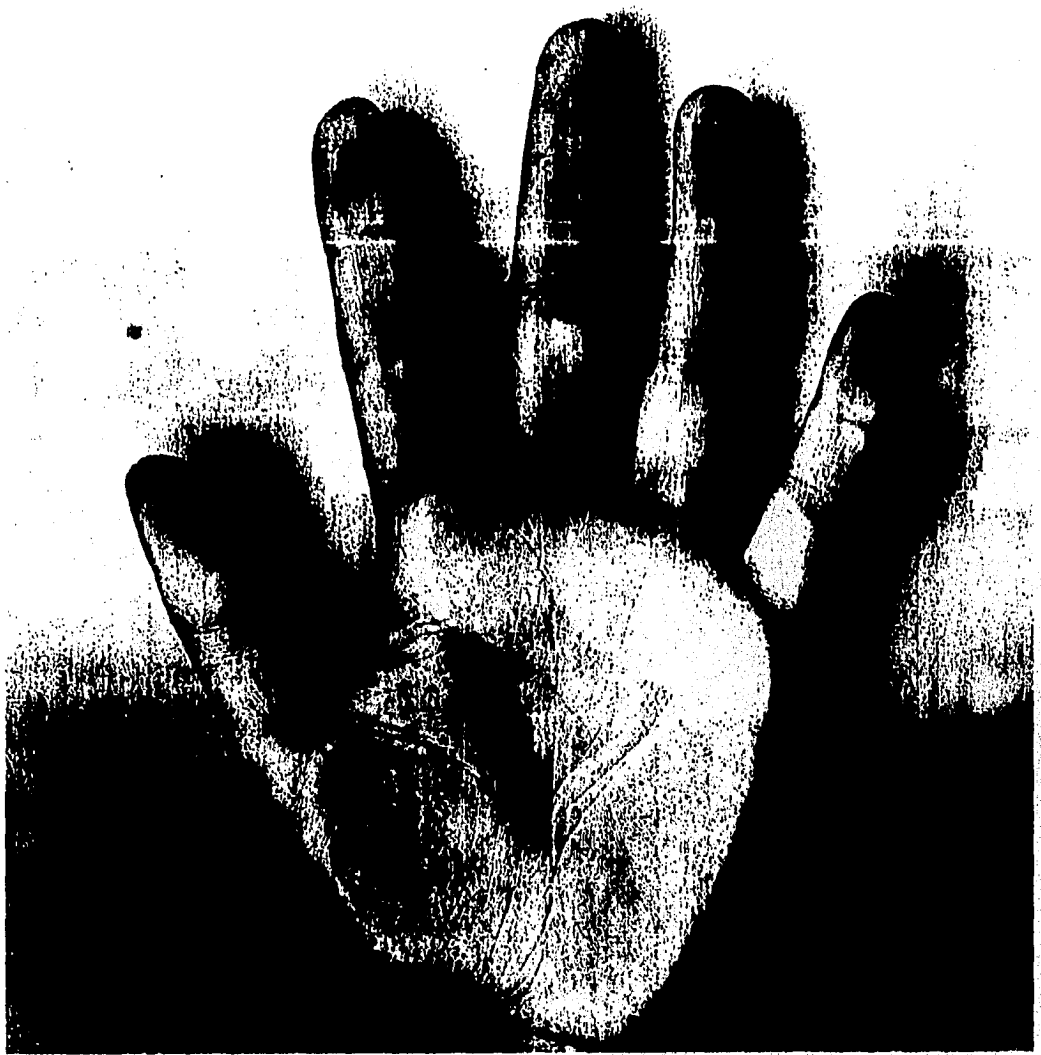


Figura 1.19. Arquitectura de fase II.

CAPITULO II

SISTEMAS OPERATIVOS DESDE UNA PERSPECTIVA DE CLIENTE/SERVIDOR



Estabamos en la escuela a las 8:10 A.M. La clase empezó y todos continuaban hablando. El profesor, tomó la palabra y dijo que tomáramos asiento, que hoy hablaríamos de Sistemas Operativos. Entonces se escuchó una voz que decía, para que Sistemas Operativos si ya lo conocemos.
El profesor dijo que sobre una hoja pusiéramos la mano izquierda, y dibujáramos su contorno, después escondiéramos la mano y trazáramos las líneas de ésta.
Apraximadamente después de 5 minutos, comentó lo siguiente:
Comparemos los trazos dibujados con los de nuestra mano. Y al percatarse que ninguno hablaba, con una sonrisa comento. "Y ESO QUE LA CONOCEN COMO LA PALMA DE SU MANO".

ALFREDO TORALES CHAVEZ

CAPITULO 2

SISTEMAS OPERATIVOS DESDE UNA PERSPECTIVA CLIENTE/SERVIDOR.

La parte fundamental del software de una computadora es el sistema operativo, el cual controla todo los recursos y proporciona las bases sobre las cuales pueden escribirse los programas de aplicación. El Sistema Operativo es una capa de software, que permite interactuar con el hardware de la computadora de una forma lógica, haciendo transparente la orquitectura misma de la computadora. Anteriormente uno de los obstáculos de los programadores o de cualquier persona que utiliza las computadoras era el manejo del hordware, esto se logró gracios a que se ha colocado un nivel de software por arriba del hardware con el fin de controlar todas las partes del sistema y presentar al usuario una Interfaz o máquino virtual que facilite la comprensión del sistema.

Los sistemas operativos se han venido desarrollando de formo paralela con el desarrollo de las computadoras. Uno de los desarrollos más importantes desde la décoda de los 80 ha sido el crecimiento de las redes de computadoras personales con sistemas operativos de red y sistemas operativos distribuidos. En un sistema operativo de red, los usuarios están conscientes de la existencia de varios computadoras que pueden conectarse con computadaras remotas, y acceder a la información y recursos de otras computadoros.

Un sistema operativo distribuido, es aquel que oparece ante sus usuarios como un sistema tradicional de un solo procesador, donde los usuarios no deben de ser consientes del lugar donde su programa se ejecute ó estén los archivos; ésto debe de ser manejado en forma automática y eficaz por el sistema operativo. Así los verdaderos sistemas operativos distribuidos requieren añadir un poco más de código a un sistema operativo de un único procesador. Estos permiten a menudo que un programa se ejecute mediante varios procesadores a la vez, por lo que necesitan algoritmos de osignación de tiempo más complejos, con el fin de optimizar la magnitud de paralelismo logrado.

Los sistemas operativos de red no tienen diferencias fundamentales con los sistemas operativos de un solo procesador. Es obvio que necesitan un controlador de interfaz de la red y algo de software de bajo nivel para dirigirlo, al igual que programas que permitan la conexión y el acceso a un archivo remoto, pero estas características adicionales no modifican la estructura del sistemo operativo.

Hoy en día es muy importante en un equipo que se desempeñe como servidor, el soporte del sistema operativo[1], ver tabla 2.1. ya que involucra los siguientes factores:

- Capacidad del DBMS
- Herramientas del administrador de red
- Cantidad de usuarios

Tipo de Servidor	Tamaño de Base de Datos	Núm. de Usuarios	Sistema Operativo Recomendado
PC/Equipo Portátil	< 500 MB	Simple usuario	DOS, Windows, Macintosh
Servidor LAN	< 1 GB	2-20 usuarios	OS/2, NetWare, Unix, Windows NT
Servidor LAN	< 20 GB	20-100 usuarios	OS/2, Unix, Windows NT
Servidor en Paralelo	> 20 GB	> 100 usuarios	MVS, Tandem Guardian, Unix

Tabla 2.1. Tabla comparativa de servidores VS sistemas operativos.

2.1 SISTEMAS OPERATIVOS

Las partes de un sistema operativo, como se muestra en la figura 2.1. son:

KERNEL O NUCLEO: Es la parte más interna de un sistema operativo, desarrolla tres tareas principales.

- Administrador de interrupciones.
- Asignación de números a los procesos.
- Comunicación entre procesos.

MANEJADOR DE MEMORIA: Proceso encargado de asignar espacio en memoria para los demás procesos que se encuentran ejecutando y de liberar el espacio para aquellos que han sido terminados.

MANEJO DEL PROCESADOR: Este programa del sistema (a veces conocido como despachador de alto nivel o scheduler) se encargá de determinar el orden óptimo de atención a los diversos procesos que están compitiendo por ganar la atención del procesador central. En la sección de procesos se amplíora la información de este punto.

MANEJADOR DE ENTRADAS/SALIDAS: Este se involucra cuando un proceso necesita un dato de entrada o requiere enviar a algún dispositivo. En este caso el propio proceso genera una interrupción y el sistema operativo llama a este manejador. De manera típica cuando este subsistema ha manejado apropiadamente el dato, genera una instrucción de retorno de interrupción y el programa que genero la llamada continua. Estos subsistemas pueden trabajor con el concepto de poner en cola de espera los requerimientos.

[1] DBMS DataBase & Client/Server Solutions - Agosto 1995 Volumen 8 número 9 - Choosing the Right Client/Server Operating System

MANEJO DE INFORMACION: Se encarga de conocer y manejar a los dispositivos de almacenamiento, además de recibir y ejecutar las instrucciones que envían los demás procesos para grabar o leer datos.

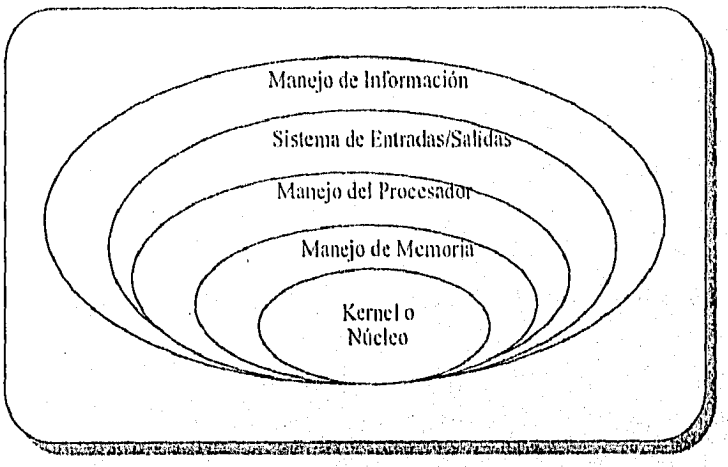


Figura 2.1 Partes de un sistema operativo

MODELO CLIENTE/SERVIDOR

Una tendencia de los sistemas operativos modernos, es la de explotar la idea de mover el código a capas superiores y eliminar la mayor parte posible del sistema operativo para mantener un núcleo mínimo. El punto de vista usual es el de implantar la mayoría de las funciones del sistema operativo en los procesos del usuario. Para solicitar un servicio, como la lectura de un bloque de un archivo, un proceso del usuario (denominado proceso cliente) envía la solicitud a un proceso servidor que realiza entonces el trabajo y regresa la respuesta.

En este modelo, lo único que hace el núcleo es controlar la comunicación entre los clientes y los servidores. Al separar al sistema operativo en partes, cada una de ellos controla una parte del sistema, como el servicio a archivos, servicio de proceso, servicio a terminales o servicio a la memoria, cada parte es pequeña y controlable, ver figura 2.2. Como todos los servicios se ejecutan como procesos en modo usuario y no en modo núcleo, no tiene acceso directo al hardware, en consecuencia si hay un error en el servidor de archivos, éste podría fallar pero no afecta en general a toda la computadora.

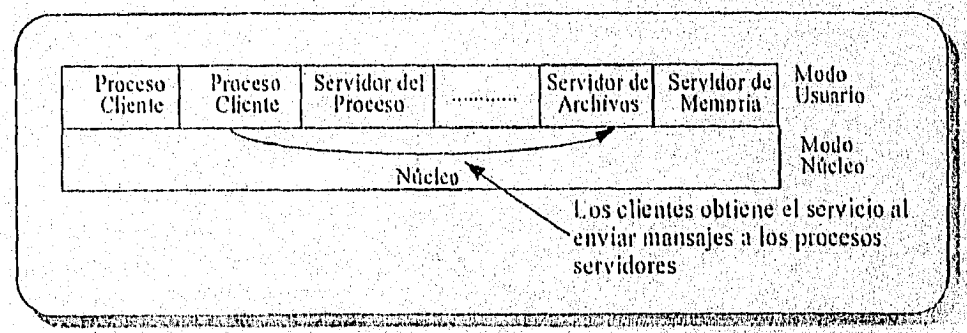


Figura 2.2 El modelo cliente/servidor

Otra de las ventajas del modelo cliente/servidor es la capacidad de adaptación para su uso en los sistemas distribuidos. Si un cliente se comunica con el servidor mediante mensajes, el cliente no necesito saber si el mensaje se maneja en forma local en su computadora o si se envía por medio de una red a un servidor en una equipo remota, ver figura 2.3.

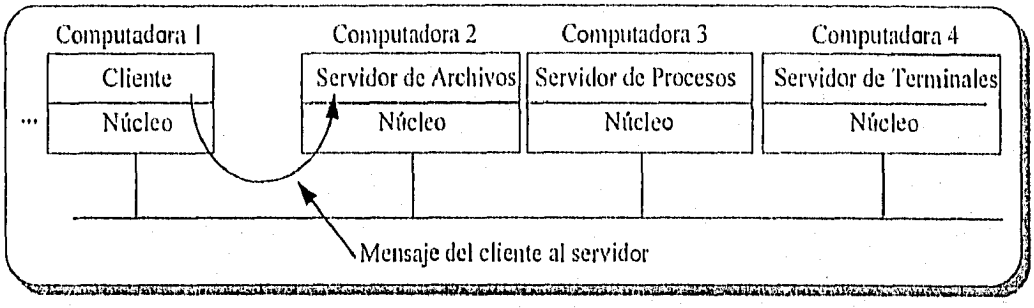


Figura 2.3 El modelo cliente/servidor en un sistema distribuido

La idea anterior de un núcleo que sólo controla el transporte de mensajes de clientes a servidores y viceversa no es totalmente real. Algunas funciones del sistema operativo (como el corgador de comandos en los registros físicas de los dispasitivos de E/S) son difíciles, más no imposibles de realizar a partir de progromas del usuario.

Existen das formas de resolver dicho problema. Uno es hacer que algunos procesos de servidores críticos (por ejemplo, las directivas de dispositivos de E/S) se ejecuten en realidad en modo núcleo, con acceso total al hordware, pero de forma que se comuniquen con los demás procesos mediante el mecanismo normal de mensajes.

La otra forma es construir una cantidad mínima de mecanismos dentro del núcleo, pero manteniendo las decisiones de política relativas a los usuarios dentro de los espacios del usuario. Por ejemplo, el núcleo podría reconocer que cierto mensaje enviado a una dirección especial indica que se tome el contenido de ese mensaje y se cargue en los registros de dispositivos de E/S de algún disco, para iniciar la lectura del disco.

2.1.1 MULTITAREA

El concepto central de cualquier sistema operativa es el proceso. Un proceso es un programa en ejecución. Las computadoras modernas realizan varios tareas al mismo tiempo. En un sistema multiprogramación, el CPU (Unidad de Procesamiento Central) también alterna de programa en programa, ejecutando cada uno de ellos por decenas o cientos de milisegundos. Aunque, en sentido estricto, el CPU ejecuta en cierto instante un solo programa, durante un segundo puede trabajar con varios de ellos, lo que da una apariencia de paralelismo. A veces llamado pseudoparalelismo.

Aunque cada proceso es una entidad independiente, con su propio contador de programa y estado interno, es frecuente que los procesos deban interactuar con otros. Esto nos lleva a lo que se conoce como estado de procesos, la figura 2.4. muestra los tres estados que puede tener un proceso.

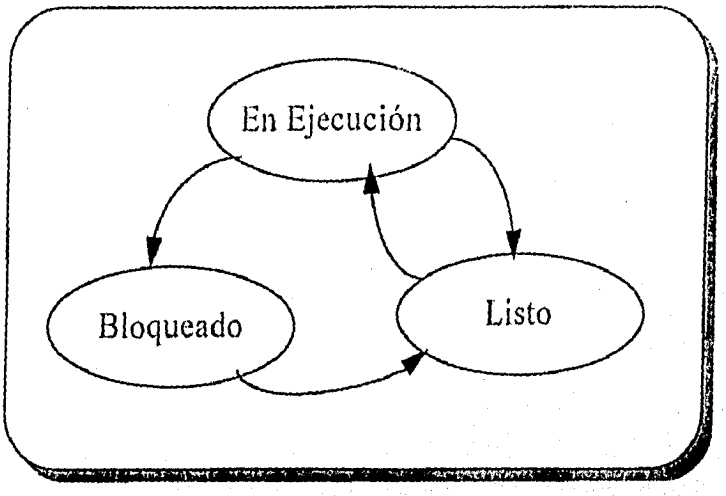


Figura 2.4. Transiciones entre estados

1 EN EJECUCIÓN .- Utiliza el CPU en un instante dado.

2 LISTO .- Ejecutable, se detiene en forma temporal para que se ejecute otro proceso.

3 BLOQUEADO .- No se puede ejecutar debido a la ocurrencia de algún evento externo.

Desde el punto de vista lógico, los dos primeros estados son similares. En ambos el proceso desea ejecutarse, sólo que en el segundo, no existe CPU disponible para él. El tercero es distinto, puesto que el proceso no se puede ejecutar, incluso aunque el CPU no tenga actividad por realizar.

Multitareo, la definiremos como la multiplexión de los recursos del sistema tales como el procesador, la memoria y los dispositivos de E/S entre una serie de programas activos.

El término proceso puede ser en cierto modo confuso cuando se utiliza para describir un entorno de multiprogramación, ya que un sistema de procesamiento es generalmente entendido como un sistema con múltiples procesadores hardware, en lugar de un solo procesador con múltiples procesos que se ejecutan concurrentemente.

Cuando sea necesario distinguirlo de la multiprogramación, utilizamos el término multitarea para designar a un sistema operativo que soporta ejecución concurrente de programas sobre un solo procesador sin soportar necesariamente formas elaboradas de gestión de memoria y de archivos, además de soportar la ejecución concurrente de programas. Por tanto, un sistema operativo de multiprogramación es también un sistema operativo multitarea mientras que la inversa no siempre es cierta.

2.1.2 CONCURRENCIA

Los procesos son un mecanismo esencial para definir y gestionar la ejecución concurrente de los programas bajo control de un sistema operativo. El concepto de proceso aparece implícitamente o explícitamente en todos los sistemas operativos multiprogramados.

Los procesos coma tales no existen aisladamente en un sistemo, de tal forma que se ejecutan concurrentemente con otros procesos, y dependiendo de su interacción podemos clasificarlos como:

1. INDEPENDIENTES
2. COOPERATIVOS
3. COMPETIDORES

1. PROCESOS INDEPENDIENTES : Son aquellos que no se comunican o sincronizan[2] con otros. Si algún error llegara a ocurrir en ellos, los procedimientos de recuperación de errores pueden ser iniciado en un proceso aislada del resta del sistema sin que afecte a los demás.

2. PROCESOS COOPERATIVOS : Son aquellos que son asincronos, pero que en un momento deben camunicarse y sincronizar sus actividades entre ellos para poder realizar algunas aperaciones en común. En consecuencia al ocurrir un error se involucran en la recuperación del mismo todos los procesas cooperativas.

3. PROCESOS COMPETIDORES : Necesitan sincronizarse y camunicarse para la obtención de recursos, en esencia son independiente, normalmente un error en uno de ellos na afectará a los demás, sin embarga cuando el error acurre en la obtención de un recurso, los demás pracesos quedarán involucrados.

HILOS (THREADS)

Los hilos son como minipracesos, cada hilo se ejecuta en forma estrictamente secuencial y tiene su propia contadar de programa y una pila para llevar un registro de su pasición. Los hilas comparten el CPU, de la misma farma que lo hacen las pracesos, primera se ejecuta un hilo y después otra (tiempo campartido). Sálo en un multìpracesador se pueden ejecutar realmente en paralela. Las hilos pueden crear hilas hijos y se pueden blaquear en espera de que se termine las llamadas al sistema, al igual que las pracesas regulares. Mientras un hila está bloqueada, se puede ejecutar otro hila del misma praceso, exactamente en la misma forma cuando se bloquea un procesa, y así se puede ejecutar en la misma equipa otro proceso.

En la mayoría de los sistemas operativas tradicianales, cada proceso tiene un espacio de direccianes y un único hila de control. Sin embarga, con frecuencia existen situacianes en dande se desea tener varios hilas de cantral que compartan un única espacio de direcciones, pera que se ejecutan de manera casi paralela, camo si fuesen de hecho procesas independientes (excepto par el espacia de direcciones campartido).

En la figura 2.5. (a), vemos una equipa con tres procesos. Cada una de ellas tiene su prapia cantadar del programa, su propia pila, su propia conjunto de registros y su propio espacio de direcciones. Los pracesos no tienen nada que ver entre sí, excepta que podrían comunicarse mediante las primitivas de comunicación entre procesos del sistema, camo los semáforos, monitares o mensajes. En la figura 2.5. (b) vemas otra equipa, con un pracesa, sólo que ahora este proceso tiene varios hilos de control, los cuales, por la general, se llaman simplemente hilos a a veces procesos ligeras.

[2] Sincrono.- Circustancia de coincidir fenómenos ó hechos en el tiempo.

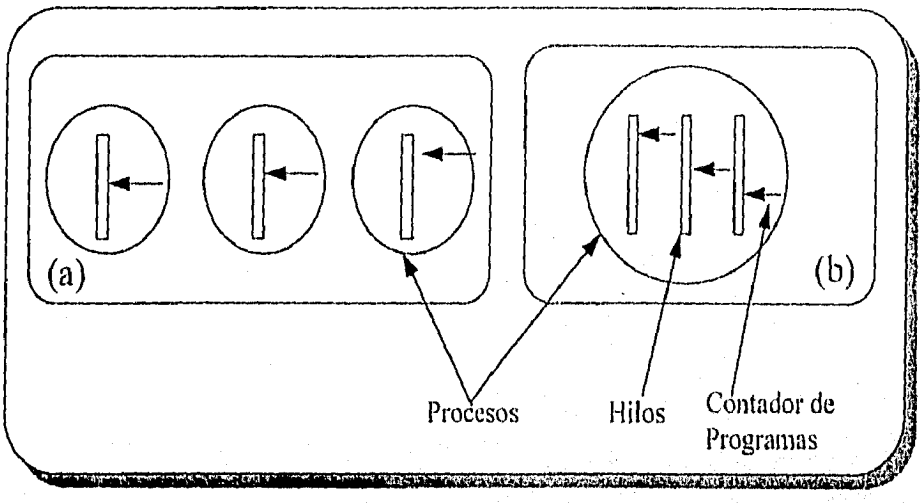


***Figura 2.5.** (a) Tres procesos con un hilo cada uno (b) Un proceso con tres hilos*

Como los procesos tradicionales (es decir, procesos con un único hilo) los hilos pueden tener uno de los siguientes estados: en ejecución, bloqueado, listo o terminado.

USO DE HILOS

Los hilos se inventaron para permitir la combinación del paralelismo con la ejecución secuencial y el bloqueo de las llamadas al sistema. En la figura 2.6.(a) se muestra una posible organización, en la que un hilo, el servidor, lee las solicitudes de trabajo en el buzón del sistema. Después de examinar la solicitud elige a un hilo trabajador inactivo (es decir, bloqueado) y le envía la solicitud, lo cual se realiza con frecuencia al escribir un apuntador al mensaje en una palabra especial asociada a cada hilo.

La estructura del servidor en la figura 2.6.(a) no es la única manera de organizar un proceso de muchos hilos. El modelo de equipo de la figura 2.6. (b) es también una posibilidad. Aquí todos los hilos son iguales y cada uno obtiene y procesa sus propias solicitudes. No hay servidor. A veces llega trabajo que un hilo no puede manejar, en particular si cada hilo se especializa en manejar cierto tipo de trabajo. En este caso, se puede utilizar una cola de trabajo, la cual contiene todos los trabajos pendientes. Con este tipo de organización, un hilo debe verificar primero la cola de trabajo antes de buscar en el buzón del sistema.

Los hilos se pueden organizar también mediante el modelo de entubamiento ver figura 2.6.(c). En este modelo, el primer hilo genera ciertos datos y las transfiere al siguiente para su procesamiento. Los datos pasan de hilo en hilo y en cada etapa se lleva a cabo cierto procesamiento. Las entubamientos se utilizan ampliamente en muchas áreas de los sistemas de cómputo, desde la estructura interna de los CPU RISC hasta las líneas de comandos de UNIX.

Los hilos no tienen nada que ver con RPC ó comunicaciones, los hilos comparten un buffer común y es recomendable por está que se encuentren en procesos comunes.

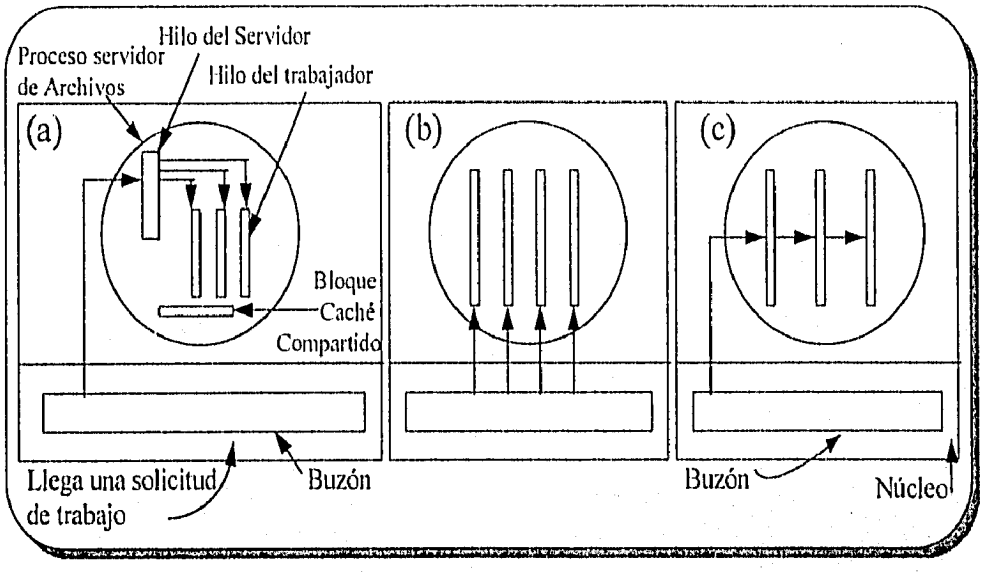


Figura 2.6. Tres organizaciones de hilos en un proceso a) Modelo del servidor/trabajo. b) Modelo de equipo. c)Modelo de entubamiento.

2.1.3 ADMINISTRACION DE ARCHIVOS

Un componente fundamentol de los sistemos distribuidos es el sistema de archivos. Lo tarea del sistema de archivos en las sistemas distribuidos es almacenar programas y datos, poro disponer de ellos cuando sea necesorio.

En el caso de un sistema distribuido, es importante distinguir entre los conceptos de servicio de archivas y el servidor de archivos.

Las servicias de archivos se pueden dividir en dos tipos, según si soportan un modelo corga/descargo o un modelo de acceso remoto. En el modelo cargo/descorga, que se muestra en lo figura 2.7.(a), el servicio de archivo sólo praporciona dos operaciones principales: la lecturo de un orchivo y la escrituro en un archivo. La primera operación transfiere todo un archivo de uno de los servidores de orchivos ol cliente solicitante. La segundo operación transfiere todo un archivo en sentido contrario, del cliente al servidor. Así, el modelo conceptual es el traslado de orchivos completos en olguna de las direcciones. Los archivas se pueden olmocenar en memoria o en un disco local, como sea necesario.

El otro tipo de servicio de archivos es el modelo de acceso remoto, que se muestra en lo figura 2.7.(b). El servicio de archivos proporciona un gran número de operaciones para obrir y cerrar archivas, leer y escribir partes de archivos, moverse a través de un archivo (LSEEK), examinar y modificar los otributos de archivo, etc. Mientras que en el modelo corgo/descargo el servicio de archivos sólo proporciona el almacenamiento físico y la transferencio, en este caso el sistema de orchivos se ejecuto en los servidores y no en los clientes. Su ventoja es que no necesita mucho espacio por parte de los clientes, o la vez que elimino la necesidad de transferir orchivos completos cuando sólo se necesita uno pequeña parte de ellos.

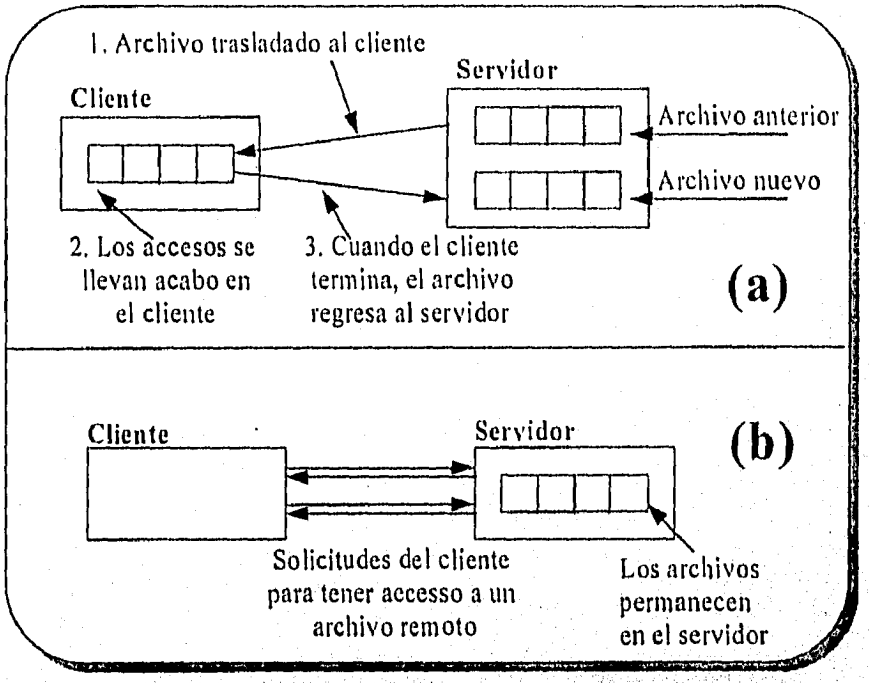


Figura 2.7. (a) Modelo carga/descarga.(b) Modelo de acceso remoto

El sistema de archivos distribuido requiere de seguridad, existe un sistema de archivos NFS *Network File System* que en realidad na tiene ninguna seguridad en el manejo de archivos, además na hay replicación de datos, ID de usuarias no encriptada y si hay reconfiguración en usuarios es necesario realizar cambias en el punto de montaje. El mejor sistema de archivos es AFS *Andrew File System* en el manejo de seguridad, que desarrallaremos más adelante.

LA ARQUITECTURA DE AFS,(Andrew File System," Sistema de Archivos Andrew")

La configuración del sistema se muestra en la figura 2.8. Consta de unidades de asignación, con un servidar de archivas y varias docenas de estaciones de trabajo clientes. La idea es lograr que la mayor parte del tráfico sea local en una unidad de asignación, para reducir la carga en la columna vertebral "RED", ver figura 2.8.

Desde el punta de vista físico, no hay distinción alguna entre las máquinas cliente y servidor; todas ellas ejecutan versiones modificadas (un paco diferentes entre si) del sistema operativo UNIX de Berkeley, con su enorme núcleo monolítico. Sin embargo, por arriba del núcleo, los clientes y servidores ejecutan software completamente distinto. Las máquinas clientes ejecutan manejadores de ventanas, editores y demás software estándar en UNIX, mientras que cada servidor ejecuta un único programa, llamado vice, el cual maneja las solicitudes de archivos de sus clientes. Cada cliente tiene además una parte de código llamada venus, que controla la interfaz entre el cliente y vice.

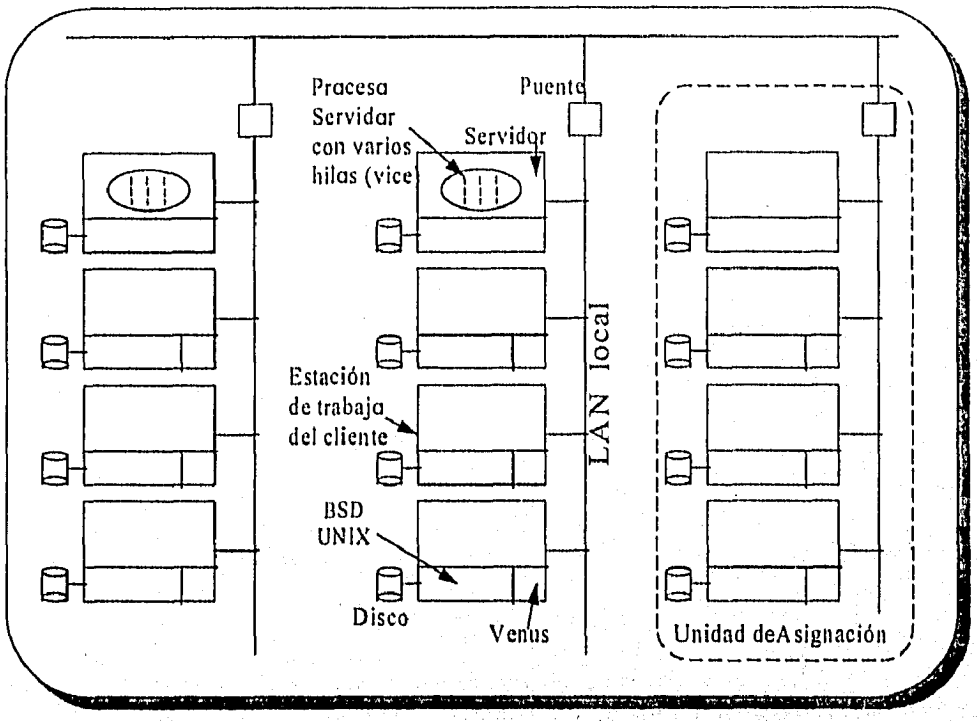


Figura 2.8. Configuración del sistema utilizado por AFS.

El espacio de nombres visible para los programas del usuario se ve como un árbol tradicional en UNIX, pero al cual se añade un directorio /cmu, como se muestra en la figura 2.9. AFS soporta el contenido de /cmu mediante los servidores vice y son idénticos en todas las estaciones de trabajo. Los demás directorios y archivos son estrictomente locales y no se comparten.

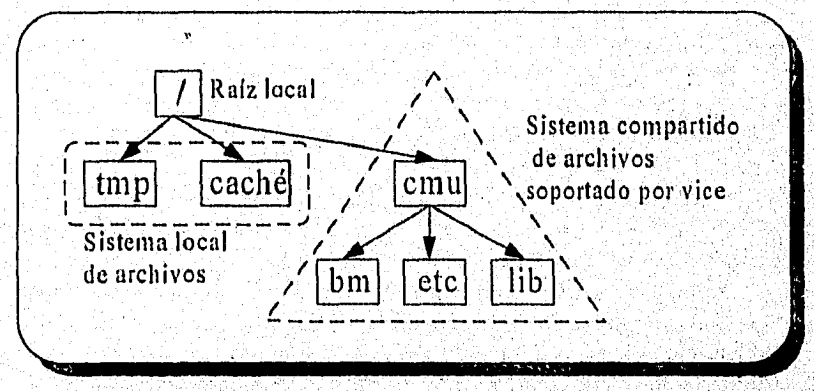


Figura 2.9. Visión del sistema de archivo desde una estación de trabajo del cliente

La idea fundamental detrás de AFS es que cada usuario realice lo más posible en su estación de trabajo y que interactúe con el resto del sistema tan poco como sea pasible. Al abrir un archivo, todo éste (o sí es un orchivo de gran tamaño, una gran parte de él) se carga en disco de la estación de trabajo y se guarda en un caché, sin que el procesa que ejecutó el OPEN se de cuenta de ello. Por esta rozón, cada estación de trabajo cliente tiene un disca duro.

Después de cargar un archivo, se le inserta un directorio local /caché, de modo que parezca un archivo normal para el sistema operativo. El descriptor de archivo que regreso lo llamado al sistema OPEN designa a dicho archivo, de modo que las llamadas READ Y WRITE funcionan de la manera usual, sin utilizar vice o venus. En otras palabras, aunque el código del sistema operativo que maneja la llamada OPEN se ha modificado en gran parte, para que maneje la interacción entre el cliente, el caché y el servidor de archivos, no se ha tocado el código de READ Y WRITE. Sólo se utiliza el archivo local en /caché.

2.1.4. SEMAFOROS

Los procesos requieren con frecuencia la comunicación entre ellos. Por ejemplo, un entubamiento, la salida del primer proceso debe transferirse al segundo proceso y así sucesivamente. Existe así la necesidad de comunicación entre procesos; de preferencia, en una forma estructurada sin utilizar interrupciones.

Existen dos primitivas de comunicación entre procesos que son: dormir (sleep) y despertar (wakeup). Sleep es una llamada al sistema que provoca el bloqueo de quien hizo la llamada, es decir, que sea suspendido hasta que otro proceso lo despierte, la llamada wakeup tiene un parámetro, el proceso por despertar.

En 1965 E. W. Dijkstra sugirió el usa de una variable entera, llamada *semáforo*, para contar el número de despertares almacenados para sus uso posterior. *Un semáforo puede tener un valor 0, lo que indica que no existen despertares almacenados; o bien algún valor positiva si están pendientes uno o más despertares.*

Dijkstra propuso dos operaciones, down y up (generalizaciones de sleep y wakeup, respectivamente). La operación down verifica si el valor de un semáforo es mayor que 0. En este caso, decrementa el valor (es decir, utiliza un despertar almacenado) y continúa. Si el valor es 0, el proceso se va a dormir. La verificación y modificación del valor, así como la posibilidad de irse a dormir se realiza en conjunto, como una sola e indivisible acción atómica. Se garantiza que al iniciar una operación con un semáforo, ningún otro proceso puede tener acceso al semáforo hasta que la operación termine o se bloquee. Esta atomicidad es absolutamente esencial para resolver los problemas de sincronización y evitar condiciones de competencia entre procesos.

La operación up incrementa el valor del semáforo correspondiente. Si uno o más procesos dormían en ese semáforo y no podían completar una operación down anterior, el sistema elige algunos de ellos (por ejemplo, en forma aleatoria) y se le permite terminar down. Así, después de un up en un semáforo con procesos durmiendo, el semáforo seguirá con valor 0, pero habrá un número menor de procesos durmiendo. La operación de incremento del semáforo y despertar de un proceso también es indivisible.

MONITORES

Para facilitar la escritura de programas correctos, Hoare (1974) y Brich Hansen (1975) propusieron una primitiva de sincronización de alto nivel, llamada monitor. Un monitor es una colección de procedimientos, variables y estructuras de datos que se agrupan en cierto tipo particular de módulo o paquete. Los procesos pueden llamar a los procedimientos de un monitor cuando lo deseen, pero no tiene acceso directo a las estructuras de datos internas del monitor desde los procedimientos declarados fuera de él.

Los monitores tienen una propiedad importante que los hace útiles para conseguir la exclusión mutua: sólo uno de los procesos puede estar activo en un monitor en cada momento. Los monitores son construcciones del lenguaje de programación, por lo que el compilador sabe que son especiales y puede controlar las llamadas a los procedimientos del monitor en forma distinta a las llamadas de los demás procedimientos. Por lo general, cuando un proceso llama a un procedimiento monitor, las primeras instrucciones de éste verificarán si hay otro proceso activo dentro del monitor. En caso afirmativo, el proceso que hace la llamada será suspendido hasta que el otro proceso salga del monitor. Si no hay otro proceso que esté utilizando el monitor, el que hace la llamada podrá entrar.

USO DE LOS MENSAJES PARA LA IMPLANTACION DE SEMAFOROS Y MONITORES

Si se dispone de un sistema de mensaje, se pueden implantar los semáforos y los monitores mediante un pequeño truco. Este consiste en introducir un nuevo proceso, el proceso de sincronización. Veamos primero la forma en que se puede utilizar este proceso para implantar los semáforos. El proceso de sincronización mantiene, para cada semáforo, un contador y una lista ligada en procesos en espera. Para llevar a cabo un UP o un DOWN, un proceso llama al correspondiente procedimiento de biblioteca, up o dawn, el cual envía al proceso de sincronización un mensaje que incluye la operación deseada y el semáforo por utilizar. El procedimiento de biblioteca recibe entonces un RECEIVE para obtener la réplica del proceso de sincronización.

Cuando llega el mensaje, el proceso de sincronización verifica el cantador para ver si la operación pedida se puede llevar a cabo. Las UP siempre se pueden llevar a cabo, pero los DOWN se bloquean si el valor del semáforo es 0. Si la operación es permitida, el proceso de sincronización envía de regreso un mensaje vacío, con lo que elimina el bloqueo del proceso que hizo la llamada. Si, por el contrario, la operación es un DOWN y el semáforo es 0, el proceso de sincronización forma el proceso que hizo la llamada en la cola y no envía una réplica. El resultado es el bloqueo del proceso que realiza el DOWN, lo cual es correcto. Más tarde, si se realiza un UP, el proceso de sincronización elige uno de los procesos bloqueados en el semáforo, ya sea por orden de llegada (el primero en llegar es el primero al que se da servicio), prioridad o cualquier otro orden y les envía una réplica. Los condiciones de competencia se evitan en este caso debido a que el proceso de sincronización sólo trabaja con una solicitud a la vez.

Los monitores se pueden implantar por medio de mensajes con el mismo truco. Una forma de lograr ésto es haciendo que el campilador implante los procedimientos del monitor mediante llamadas a los procedimientos de biblioteca up y down para asociarle un semáforo adicional, llamado mutex y semáforos para cada proceso.

2.1.5. ADMINISTRACION DE MEMORIA

Existen varias maneras de manejar la memoria en un sistema de cómputo, por lo que nos enfocaremos a las que permiten la multiprogramación.

MANEJO DE MEMORIA POR PARTICIONES

Es el primer esquema para permitir la multiprogramación (activación de varios procesos simultáneamente) y consiste en subdividir la memoria en varias secciones fijas y asignar cada una de ellas a un usuario o proceso activo. El principal problema por resolver es asegurar que ningún usuario intervenga en el área de memoria asignada a otro. Desde este punto de vista, el

manejo de memoria consiste en contralar cuáles particiones están asignadas a que procesos, para poder liberar particianes cuando las pracesas residentes en ellas terminen o cambien, y poder ofrecer particiones libres a pracesas que soliciten atención por parte del sistema de cómputo.

La ventaja fundamental de este modela es que permite la multipragramación, y su principal desventaja consiste en que deja lugares libres en la memoria que, camo son de tamaño fijo, no pueden ser utilizadas más que por pracesos de longitud menar o igual a la de la partición en cuestión.

PARTICIONES RELOCALIZABLES

La desventaja del esquema anterior llamado fragmentación externa, se padría evitar permitiendo que una partición pueda fusianarse con otra, para lagrar una partición nueva de más capacidad.

Esto da lugar a un nuevo esquema de manejo de memaria llamada particiones relocalizables. La idea consiste en maver celdas de memoria de un lugar a atra para juntar las áreas libres en un misma lugar. Las celdas no se mueven, sino que sus contenidas se copian de un lugar a otro, y aunque con esto se crea un nuevo prablema el de la relocalización, permite mayor flexibilidad que el anterior; sólo que resulta más costoso, puesto que hay que compactar (mover) las pracesas a tiempo de ejecución y realizar algunos cambios en el pracesador central, para evitar que este desplazamiento cause problemas con respecta a las direcciones. El procesador central se encarga de este ajuste a tiempa de ejecución por medio de un componente electrónico adicianal que se conace como registra de relocalización.

SWAPPING (INTERCAMBIO)

En algunos sistemas se recurre a la solución de quitar por completo de la memoria un proceso que está desactivado, copiándalo al disca magnético y liberando así un área significativa en la memaria central. Cuanda llegue el momento de valverlo a ejecutar se corgará nuevamente trayéndolo del disca magnético en el que reside.

PAGINACION

Debido a los castos que representa la compactación (ya que es necesario detener la ejecución del proceso para efectuarla) o el swapping (porque el traslado hacia/desde el disco magnético tama tiempo), se inventó otra esquema, más ágil y eficiente, llamada paginación. Este consiste en dividir los procesos en fragmentos de langitud fija, llamados páginas, que se almacenan en áreas de igual tamaño en memaria, llamadas bloques. Esto es, cada página de cada praceso se guarda en un bloque en memaria. Un proceso común puede constar de quince páginas, residentes en memoria en atros tantos blaques. La ventaja radica en que na es necesario que las páginas de un pracesa estén contiguas en la memoria, quedando automáticamente eliminada el problema de la fragmentación externa. Con la ayuda de una tabla de mapeo de páginas, que controla cuáles páginas de qué procesas residen en cuáles blaques de memoria, se puede implantar un esquema muy ágil de manejo de memaria central, controlado por el sistema aperativa.

De acuerdo can la dicha, si ya na es necesario que todas las páginas de un proceso estén cargadas de farma contigua en la memoria (gracias a la tabla de mapea), entonces tampoca hay necesidad de que todas las páginas de un pracesa determinado estén residentes (contiguas a no) en memaria. Es decir, se padría camenzar a ejecutar un proceso cuando tan sólo una

parte del mismo esté cargada en memoria, e ir cargando a tiempo de ejecución las páginas que se requieran. Esta importante idea recibe el nombre de memoria virtual y es la base sobre la cual descansa el enorme poderío de una computadora grande, y la razón por la que una computadora puede atender a muchos usuarios al mismo tiempo aunque disponga de una memoria limitada.

PAGINACION POR DEMANDA

Cuando un proceso pide una página no residente en la memoria, el sistema operativo lo detecta por medio de una interrupción, que es atendida por el manejador de interrupciones del núcleo. Este determina la causa (interrupción por página) y copia la información solicitada residente en el disco magnético en un bloque libre de la memoria.

Sus ventajas son obvias, pues permite una tremenda flexibilidad en el uso de los recursos del sistema. Su desventaja es, fundamentalmente su enorme complejidad. En efecto, los sistemas operativos de este tipo constan de decenas de miles de instrucciones, y son escritos por grupos enteros de programadores durante meses, además de que se requiere un considerable auxilio por parte de los circuitos electrónicos para que la velocidad de procesamiento no disminuya radicalmente por la gigantesca cantidad de operaciones adicionales que el sistema debe ejecutar. Como la tabla de páginas reside en la memoria central, y es necesaria consultarla para cada acceso, se requiere un ciclo de lectura de memoria adicional (ciclo de fetch) por cada operación sobre una página, lo cual claramente es inaceptable en términos de la reducción de velocidad de proceso resultante. Por tanto, los sistemas de paginación por demanda emplean mecanismos adicionales de hardware para auxiliarse en esta tarea. Uno de ellos es conocido como memoria caché o memoria auxiliar rápida, en la que se guardan los contenidos activos de la sección de la tabla de páginas en uso, reduciendo gradualmente el tiempo extra requerido por cada consulta. Muchos procesadores recientes trabajan en colaboración con otro complejo subsistema electrónico para el manejo de estas tareas de paginación, que recibe el nombre de unidad de manejo de memoria MMU, *Memory Management Unit, "Unidad de Administración de Memoria"*).

SEGMENTACION

En éste, los procesos se dividen en fracciones llamadas segmentos. Un segmento es una unidad lógica autocontenida (un programa completo, una subrutina o un área grande de datos) que se carga en forma independiente en la memoria. La diferencia con respecto a las páginas es de longitud fija, mientras que los segmentos son variables, dependiendo de la cantidad de código que contenga el programa o subprograma que representan. El manejo de segmentos es parecido al de páginas, aunque tiene ciertas ventajas, basta con saber que una computadora con sistema operativo de segmentación es por lo menos tan poderosa y compleja como otra que maneje memoria virtual por paginación.

2.2 DOS, WINDOWS 3.X Y CHICAGO "Windows 95"

El sistema operativo DOS (Disk Operating System, Sistema Operativo residente en Disco), es uno de los más conocidos en el medio informático. El DOS, es un sistema operativo monousuario, desde su inicio existen muchas versiones a la fecha. Uno de los principales

proveedores del DOS es Microsoft[3], el cual se comercializa con el nombre de MS-DOS, y es el que se conserva como estándar.

El manejo de archivos y de directorios es muy fácil de operar, su administración es muy simple pero cuenta con limitaciones, por ejemplo, los nombres de archivos no deben ser mayores a 8 caracteres. Además no existe protección alguna del hardware.

El DOS cumplió su misión, hasta el momento en que los usuarios necesitan compartir recursos, envío y recepción de mensajes a otros equipos. Además, las usuarios requieren de una interfaz gráfica con la ayuda de un mouse, que les permita un ambiente gráfico e imágenes a color para un mejor desempeño de sus actividades.

El DOS se encuentra severamente limitado. Su incapacidad de dar un soporte adecuado o la multitarea, la protección de memoria y un mayor espacio de direcciones han hecho de él una base pobre, para entornos donde el usuario requiere de ejecutar distintas aplicaciones complejas mientras está conectado a una red. Todas estas modificaciones al DOS serian prácticamente como realizar un nuevo sistema operativo.

La desaparición del DOS en el mercado es poco probable, ya que Microsoft realiza versiones nuevas hasta el día de hoy, la versión 6.22 se encuentra actualmente y no tardará en salir la versión 7.0.

Versión	Fecha	Disco	Comentarios
1.0	Ago/81	160 K	Compatible con CP/M; Solo soportaba un directorio.
1.1	Oct/82	320 K	Arregló algunos errores de 1.0
2.0	Mar/83	360 K	Soportaba los discos duros; mas parecidos a Unix
2.1	Nov/83		Venia con la poca afortunada PC Jr.
2.11	Mar/84		Soporte para usuarios internacionales
3.0	Ago/84	1.2 M	Soporte de la PC/AT
3.1	Nov/84		Primera edición en soportar las redes.
3.2	Ene/86	720 K	Soportaba discos de 3.5" y el anillo de elementos de IBM
3.3	Mar/87	1.44 M	Incluido en la PS
4.0	Jul/88		Soporte de discos mayores de 32 M; shell de DOS
5.0	Mar/91	2.88 M	Limpieza general; mejor uso de la memoria extendida.

Tabla 2.2 Versiones de MS-DOS

El MS-DOS tiene una estructura de tres copas : BIOS, NUCLEO y el SHELL.

- BIOS: es una colección de manejadores de dispositivos de bajo nivel que sirven para aislar a MS-DOS de los detalles de Hardware.

- NUCLEO: está contenido en los archivos io.sys y msdos.sys los cuales contienen la parte independiente de la maquina del sistema operativo.

- SHELL: no es un propiamente un programa del sistema operativo y además puede ser remplazada por el usuario. Consta de dos partes una residente siempre en memoria y otra transitoria que está localizada en la parte superior de la memoria.

3 Microsoft compro el 86-DOS y contrato a Tim Patterson para adecuar el MS-DOS, en abril de 1981. Sistemas Operativos Modernos .- Andrew S. Tanenbaum, Página 357.

WINDOWS 3.x

Windows inicia en 1985, pero hasta 1988 con *Windows 386 v2.0* entra en el mercado y es en 1990 cuando cambia a *windows 3.0.* Windows 3.x, se presenta como un sistema operativo monousuario. Así windows 3.x necesita compartir archivos, impresoras y envío de mensajes por una red, Microsoft proporciona *Windows para Trabajo en Grupo* (Windows for Workgrups) *WFW 3.11* a principios de 1994. Windows para trabajo en grupo es red *par a par ó peer-to-peer*, lo que significa que no tiene un servidor dedicado para compartir o accesar información de otro cliente. Cuenta con las características de compartir información, envió de mensajes de correo electrónico a otros clientes, planear reuniones de grupo, compartir archivos e impresoras, gestionar calendarios y trabajar en conjunto en proyectos.

Un sistema conectado a la red, que esté ejecutando Windows para trabajo en grupo puede usar aquellos recursos compartidos como un cliente. Se puede ejecutar el programa a un sistema que ya este conectado a una red, o se puede construir una nueva red mediante la adición de tarjetas de interfaz con la red (NICs, Network Interface Cards "Tarjetas de Interfaz de Red") y uniéndolas con un cable de red (Coaxial ó UTP).

> **Bill Gates, presidente y CEO de Microsoft, realizó en siguiente comentario con relación a el trabajo en grupo** ***"La integración de características de red y grupos de trabajo en Windows es enorme y novedoso paso. Sea grande o pequeña la compañía, la gente necesita trabajar en grupo, compartir información, involucrarse uno con otro de forma rápida en la toma de decisiones, y coordinar actividades de planificación y negocios. Los servicios integrados en Windows para trabajo en grupo soportarán todas estas actividades, tanto en las aplicaciones existentes como en las nuevas, se beneficiarán de la extensión para grupos de trabajo del sistema operativo. Windows para trabajo en grupo harán más fácil la informática en grupos."***

Ventajas de Windows para trabajo en grupo:

- Transferencia de archivos rápidas y cómodas entre computadoras de la red.
- Almacenamiento de archivos y copias de seguridad centralizados.
- Soporte de Vinculación e Incrustación de Objetos (OLE) a través de la red, con lo que se consigue la actualización automática de la información de los documentos enlazados.
- Correo electrónico y mensajería
- Compartición de impresoras, discos duros, unidades de CD-ROM y otras dispositivos periféricos.
- Aplicaciones para grupo de trabajo.

SEGURIDAD. Windows para Trabajo en Grupo cuenta con características de seguridad que pueden impedir que usuarios no autorizados accedan a los recursos compartidos de la red. Los usuarios del sistema pueden compartir archivos e impresoras con otros usuarios. Los otros usuarios de la red pueden tener acceso completo al directorio, o permisos concedidos para leer archivos, pero no para modificarlos de ningún modo. El acceso se otorga dando a estos usuarios claves específicas, relacionadas con el nivel de seguridad que deberían tener. *Si se requiere una seguridad más sofisticada, se recomienda usar Windows NT o Novell Netware.*

CHICAGO (WINDOWS 95)

Windows 95 es el mejor producto de Microsoft hasta el momento. En junio de 1994 tenía el nombre de *Chicago incluso hasta Windows 4*, pero en julio de 1994, Microsoft decidió llamarlo *Windows 95.*

La misión de Windows 95 era desarrollar un sistema operativo, que cubrirá una amplia plataforma de hardware, software y fácil para el usuario.

Windows 95 es más que una extensión de MS-DOS, presenta un buen nivel de rendimiento, seguridad, robustez en la administración de red y una transparencia en la distribución de datos. Además, trabaja con aplicaciones de 32 Bits, ya separtadas por Windows NT es un salto y una completa discontinuidad con el pasado.

El diagrama de bloques de Windows 95, ver figura 2.10.

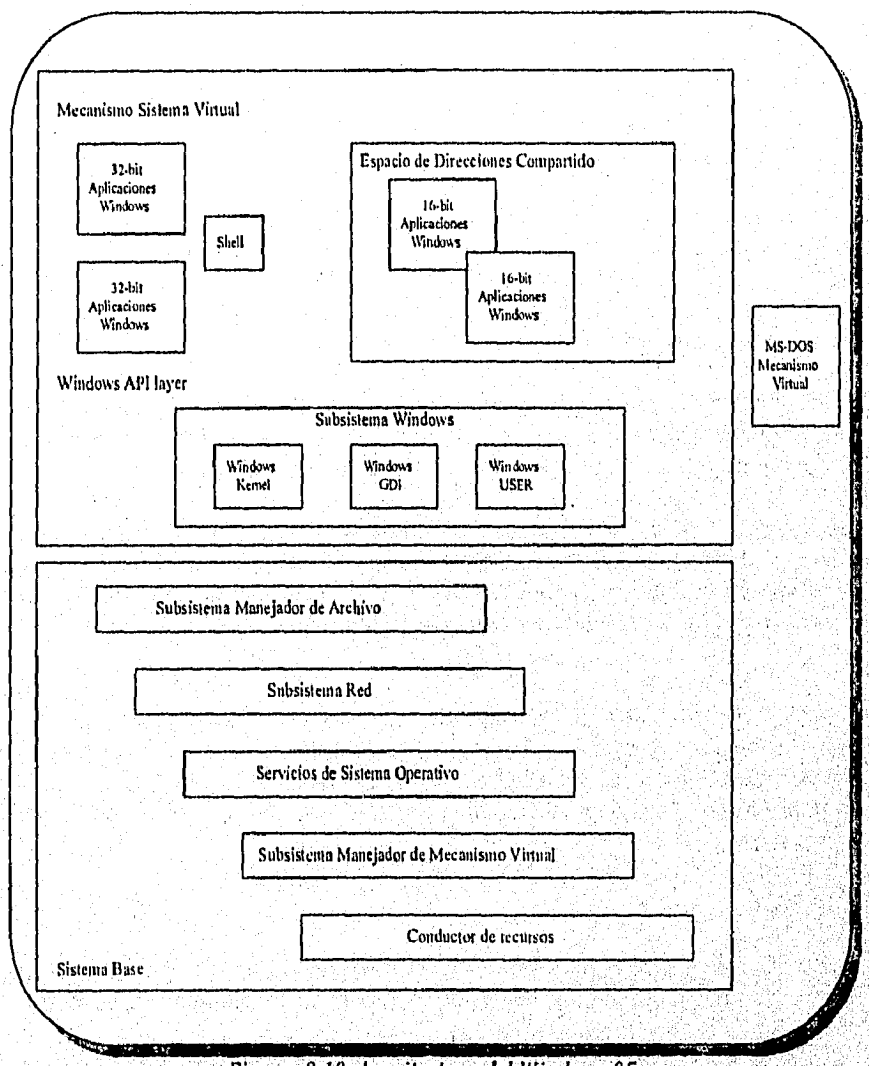


Figura 2.10. Arquitectura del Windows 95.

La Máquina virtual del sistema (a sencillamente VM *del sistema*) es el nombre dado en Windows 95 al entorno que soporta todas las aplicaciones y las componentes de subsistema Windows, como, por ejemplo, la Interfaz de Dispositivo Gráfico (GDI).

Las aplicaciones Windows de 32 bits son las nuevas aplicaciones Windows que usan el modelo de memoria de 32 bits del procesador 80386 y posteriores como Pentium, además un

subconjunto de la interfaz de programación de aplicaciones (API) Win32 de Microsoft. En Windows 95, cada una de las denominadas aplicaciones Win32 tiene un espacio de direcciones privado que es inaccesible a otras aplicaciones. Windows 95 puede planificar con derecho preferente las aplicaciones de 32 bits.

La Interfaz de órdenes es una aplicación Windows de 32 bits que proporciona al sistema la interfaz de usuario esencial. La Interfaz de órdenes en Windows 95 concentra las funciones de las utilidades Administrador de programas, Administrador de archivos y Administrador de tareas de Windows 3.1 en una única aplicación.

Las aplicaciones Windows de 16 bits son las aplicaciones Windows "anteriores", (las que se usan hoy día en Windows 3.1.). Estas aplicaciones usan el modelo de memoria segmentada de la familia de procesadores Intel en realidad, el modelo de memoria de un 80286. Como en Windows 3.1, las aplicaciones de 16 bits que se ejecutan en Windows 95 comparten un espacio de direcciones único y no pueden ser planificados con derecho preferente. Microsoft a éstas las denomino aplicaciones Win16.

La capa de la interfaz de programación de aplicacianes proporciona en Windows 95 una compatibilidad completa con la actual API de Windows 3.1, así como el soporte para la nueva API de 32 bits accesible sólo para aplicacianes Windows de 32 bits. La API de 32 bits es un subconjunto de la completa API Win32 de Microsoft vista por primera vez en Windows NT y en la Win32s añadida en Windows 3.1.

El Núcleo de Windows proporciona soporte a los servicios del nivel más bajo que requieren las aplicaciones Windows, tales como la asignación dinámica de memoria. Para Windows 95, el Núcleo proporciona estos servicios a las aplicaciones de 16 y 32 bits.

La GDI es el corazón de las capacidades gráficos de Windows, gestionando los tipos de letras, las primitivas de dibujo y el color tanto para los dispositivos de visualización como para los de impresión. Aunque la GDI en Windows 95 continua dando soporte a las aplicaciones de 16 bits existentes, incluye nuevas utilidades importantes disponibles sólo para los programas de 32 bits.

Los Máquinas virtuales MS-DOS dan soporte a la ejecución de aplicaciones MS-DOS en Windows. Como en Windows 3.1, el usuario puede ejecutar múltiples VM MS-DOS concurrentemente. Windows 95 incluye varias utilidades nuevas diseñados para mejorar la gestión por parte del usuario de esas VM, pero el diseño básica de la VM MSDOS no ha cambiada mucha.

EL SISTEMA BASE

Los módulos restantes implementan diversos aspectos del sistema operativo subyacente en Windows 95. Normalmente, al grupo de estos componentes se le denomina sistema base.

La administración de archivos ha cambiado de forma marcada en Windows 95. En Windows 3.1 es MS-DOS quien controla el sistema de archivos local del disco fijo. Este control del MS-DOS perjudicaba las prestaciones de Windows y la oportunidad de mejorar el soporte del sistema de archivos resultaba imposible mientras MS-DOS siguiera con el control. Bajo Windows 95 la situación es totalmente diferente. Sobre todo, ya no se utiliza MS-DOS para la administración de archivos en los discos locales. El nuevo subsistema de gestión de archivos proporciona varias interfaces que admiten la coexistencia de los sistemas de archivos de todos

los discos locales (incluida el sistema de archivos CD ROM) y los múltiples sistemas de archivos de red.

El subsistema de red es la versión más reciente de la red igualitaria de Micrasoft; vista por primera vez en 1992 en el producto Windows para trabajo en grupa y después en Windows NT. El subsistema de red utiliza el nuevo subsistema de administración de archivos para coordinar su acceso a los archivos remotos. Otros distribuidores de redes también pueden conectar sus productos a los nuevos servicios de administración de archivas, permitiendo que el usuario acceda simultáneamente a más de un tipo de red. Windows incorpora soporte para los protocolos Novell y TCP/IP.

Los servicios del sistema operativo incluyen en Windows 95 componentes importantes como el subsistema de configuración hardware conectar y listo, además de un grupo de funciones diversas entre las que se incluyen aquellas que satisfacen solicitudes de fecha y hora.

El Administrador de máquina virtual es el carazán del sistema aperativa Windows 95. Incluye el software que implementa todas las primitivas básicas del sistema para la planificación de tareas, operaciones de memoria virtual, carga y finalización de programas, y comunicación entre tareas.

Las controladores de dispositivos pueden tener formas diferentes en Windows 95 entre otras, controladores de modo real y los denominados controladores virtuales. Algunos sistemas aún pueden necesitar el uso de los antiguos controladores de dispositivos de MS-DOS en modo real para dar soporte a dispositivas hardware concretos, pero una de las metas del desarrollo de Windows 95 ha sido desarrollar controladores de modo protegido para tantos dispositivos populares como fuera posible, incluyendo nuevos controladores de modo protegido para el ratón, los dispositivos CD ROM y muchos dispositivos de disco fijo.

Los controladores de dispositivos virtuales (VxD), asumen el papel de campartición de un único dispositivo hardware entre diversas aplicaciones. Por ejemplo, la ejecución de dos aplicaciones MS-DOS en distintas ventanas de la pantalla necesita que el sistema cree dos VM MS-DOS, cada una de las cuales requiere el acceso a una única pantalla física. El VxD del controlador de pantalla tiene que aceptar estos requisitos de comparición VxD, también se usa como un descriptor general para otros módulos del sistema operativo de 32 bits.

2.3 WINDOWS NT

El diseño de Windows NT comenzá realmente a principlos de 1988. El equipo de desarrollo encabezada por Dave Cutler, incluía expertos ingenieros de saltware can experiencia en el diseño de Windows, UNIX, VMS y OS/2.

CARACTERISTICAS TECNICAS DE WINDOWS NT

- Permite tatal compatibilidad con los actuales paquetes de software de Microsoft, esto incluye interfaces con usuario final y API's (Aplication Program Interface).
- Proceso distribuido, está pensada para ambientes de red. Es capaz de distribuir tareas a otras computadaras en la red.
- Escalabilidad y multiproceso, las aplicaciones san capaces de aprovecharse al máximo el rango de plataformas de cómputo existentes.(Los usuarios deben de ser capaces de ejecutar sus aplicaciones en máquinas con uno a más procesadores).

- Robustez, el sistema se protege a si misma tanto de mal funcionamiento interno como externo. Cuidando que ninguna aplicación afecte negativamente la operación del sistema.
- Portabilidad, el código puede viajar fácilmente de un procesador a otro. (actualmente ya abarca la familia Intel de 80x86 y algunos procesadores Risc de MIPS y Digital).
- Tiene una arquitectura simétrica capaz de soportar multiprocesos, en procesadores de 32 Bits.
- Accesar hasta 4GB de RAM y varios Terabits de almacenamiento con direccionamiento de 64 Bits, cada aplicación podría direccionar hasta 2 GB de memoria virtual.
- Autoajuste al máximo de la memoria disponible, esto dinámicamente balanceando páginas de RAM y cache de archivos.
- Soporta drives para discos como: ST-506, ESDI, SCSI, CD-ROM, para videos: VGA, Super VGA, XGA, 8514 y TIGA, todas las tarjetas de red que actualmente soporta LM inclusive las de 32 Bits, Drives para cintas tales como: SCSI 4mm, 8mm y ¼ "; todas las impresoras que actualmente soporta Windows 3.1.
- Diseño portable, gracias a su Windows NT Ejecutivo, Micro-Kernel de 50K's.
- Alta seguridad según el estándar del US Government C2[4].
- Una sola máquina puede ser utilizada por varios usuarios secuenciales con distintos ambientes y privilegios.
- Las actuales aplicaciones de MS-DOS, Windows, OS/2 y POSIX, podrán ejecutarse en Windows NT sin ningún problema.

ARQUITECTURA DE WINDOWS NT

Este se divide en 2 partes; la parte del usuario del sistema, la cual consiste de los Procesos Servidores Windows NT, y la porción Kernel, llamada Ejecutivo, ver figura 2.11, los servidores Windows NT también son llamados subsistemas protegidos, por que cada uno cuenta con su propia memoria protegida de otros procesos por medio del manejador de memoria virtual del Ejecutiva.

En vista de que los servidores no comparten memoria la comunicación entre ellas se realiza por medio del paso de mensajes, representados por flechas en la figura 2.11. El ejecutivo esta diseñada para soportar cualquier número de procesos servidores que a su vez presenten distintas interfaces de programación (API's) para distintos ambientes de aplicación: OS/2, POSIX, MS-DOS, etc.

Existen 2 tipos de subsistemos protegidas: las subsistemas ambientales y integrados. Un subsistema ambiental es un modo usuario que provee API's (ej. POSIX a Windows 16 Bits). El mas importante subsistema ambiental es el de WIN32, el cual provee sus API's para aplicaciones Windows de 32 Bits, estas interfaces incluyen funciones de interfaz gráfica para el usuario y todas los controladores de E/S necesarios.

Los subsistemas integrados realizan importantes tareas del sistema operativo, como el subsistema de seguridad, el cual tiene privilegios especiales que no se encuentran en otro servidor en modo usuario.

---

4 Seguridad C-2.- Desarrollador por el Departamento de Defensa de los Estados Unidos, son una serie de reglas de seguridad e inspección de sucesos no solo para usuarios de la red tambien para usuarios locales.

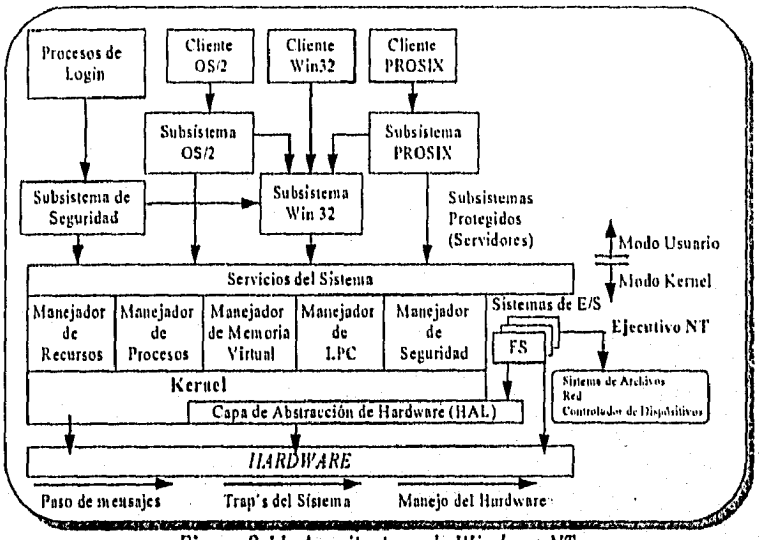


Figura 2.11. Arquitectura de Windows NT

En el modo kernel, se realizan operaciones tales como administración de memoria virtual, administración de recursos (objetos), sistemas de archivos y de entrada/salida, comunicaciones interprocesos y una parte de la seguridad del sistema. La mayor parte de estos componentes interactúan con otras a través de una estructura modular de capas, realizándose llamadas por medio de un conjunto de funcianes internas. El modelo por capas aparece también en las partes bajas del ejecutivo en el kernel y el HAL (Hardware Abstraction Layer). El kernel desorrolla las operaciones de bajo nivel del sistema aperativa, muchas de ellas encontradas en un micra-kernel, tales coma asignación de CPU, interrupciones y sincronización del multipraceso, así mismo, el Kernel da una serie de recursos básicos para implementar instrucciones de alto nivel.

Windows NT utiliza el modelo cliente/servidar primordialmente para proveer los servicios de API's y los servicias comunes de un sistema operativo. En la figura 2.12, el servidor Win32 provee al usuario una interfaz fundamental hacia la operación del sistema (los términas de subsistema y servidor pueden ser utilizados indistintamente). Los otros servidores pueden ser cargados si se desea sobre el NT ejecutivo. La comunicación entre estos sùbsistemas es a través de paso de mensajes usanda los servicios con las que cuenta el ejecutivo.

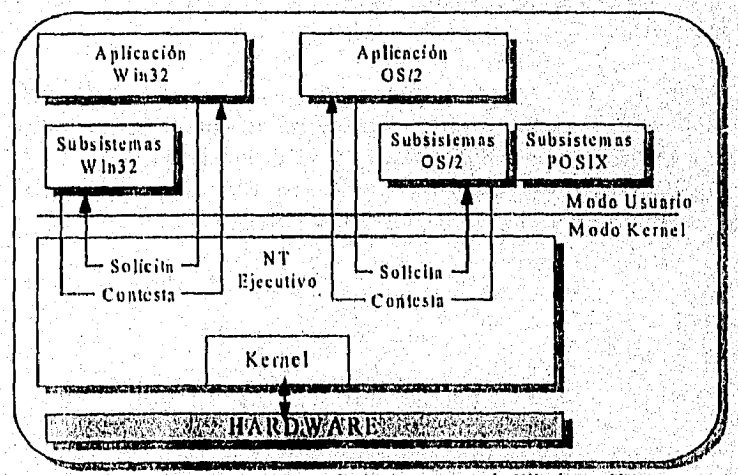


Figura 2.12. Diagrama de Aplicaciones de Windows NT.

La utilización del modelo cliente/servidor tiene varias ventajas, con el Windows NT:

- Simplifica la operación de la bose del sistema operativo. Coloca cada conjunto de API's (Win32 para aplicaciones Windows de 32 bits, MS-DOS, Windows de 16 bits, POSIX Portable Operating System Interface for UNIX) en servidores separados, libera de conflictas y duplicacianes para la base del NOS y hace mas fácil agregar nuevas API's.
- Permite implementarlo can efectividad, ya que cada procesa-servidor al correr en un nodo usuario tendrá su propia proporción de memoria protegida de otros procesos y coma ya se comento estos modos no pueden accesar directamente el hardware o modificar la memoria.
- Par si misma tienen un esquema de proceso distribuida, esta facilita que se incarpore a un esquema de red ya que el manejo de mensajes con máquinas rematas es natural y el cliente na requiere el saber que un X servicia es remoto o local en un dada caso.

Para que exista una compatibilidad de software depende de varias cosas, la principal es la arquitectura del nueva procesador, si este usa el misma conjunta de instrucciones y el mismo esquema de direccionamienta de memaria que el anteriar, entonces existe campatibilidad binaria.

La compatibilidad binario entre procesadores es diferente dependienda de cada fabricante, en caso de incompatibilidad, se puede realizar can un programa emuladar que canvierta las instrucciones que van hacia la máquina que tenemas realmente. Sin este emulàdor todas las aplicacianes que se deseen emigrar se deberán recompilar y religar para que funcione.

CARACTERISTICAS ESPECIALES DE WINDOWS NT

Windaws NT brinda soporte para Win32API, que es la interfaz de programacián principal. Para equipos con procesadores Intel, un subsistema protegida afrece compatibilidad binaria para aplicaciones MS-DOS, Windaws 3.x (Windows de 16 Bits), OS/2 y Lan Manager. En un pracesador RISC, la compatibilidad binaria para aplicacianes de MS-DOS y Windows de 16 Bits se realizo a trovés de un emuladar, la compotibilidad de código fuente es ofrecida para OS/2 e Interfaces de programación Lan Manager.

Windows na solo incluye compatibilidad a nivel de API's, si na también a nivel de archivos, para lo cual Windows NT incluye soporte para las sistemas de archivos de MS-DOS (FAT), el de OS/2 (HPFS), el sistema d archivos de CD-ROMS (CDFS), y el nuevo, recuperable sistema de archivos Windows NT (NTFS).

La multitarea comparte en un procesador varios procesas, pero cuando se tienen varios procesadores se pueden compartir los procesos entre ellos. Para tal efecta existen dos modelos, el simétrico y el asimétrica. El modela simétrico de multiprocesamiento (ASMP), tipicamente selecciona una de los procesadores paro cargar el sistema operativo, mientras los otros procesadares corren otras tareas, este se muestra en la figura 2.13.

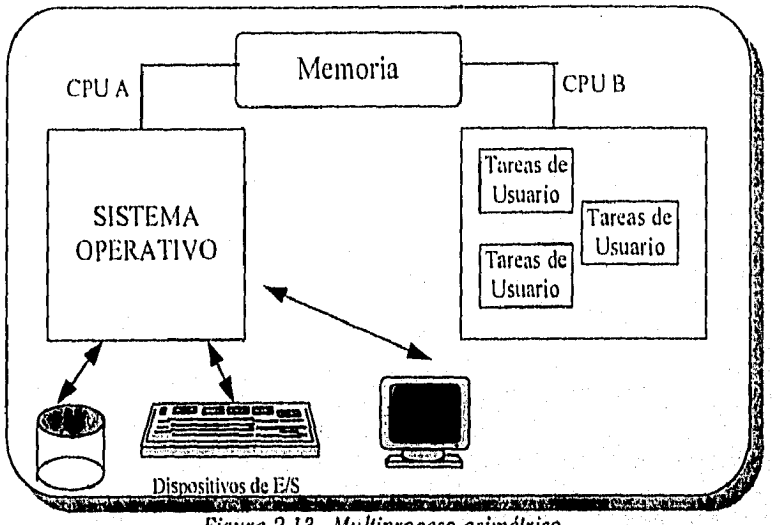


Figura 2.13. Multiproceso asimétrico

Los sistemas operativos de multiproceso simétrico (SMP) como Windows NT, permiten que corra en cualquier procesador disponible o en varios o la vez compartiendo memoria entre ellos. Esto permite un aprovechamiento ol máximo de los recursos disponibles en el equipo.

La forma en que Windows NT implementa el multiprocesamiento simétrico, es distribuyendo partes del código del sistema operativo en varios procesadores, o varias partes en uno, a excepción del kernel el cual debe estar en un solo procesador, ya que maneja las interrupciones y el scheduler, ver figura 2.14.

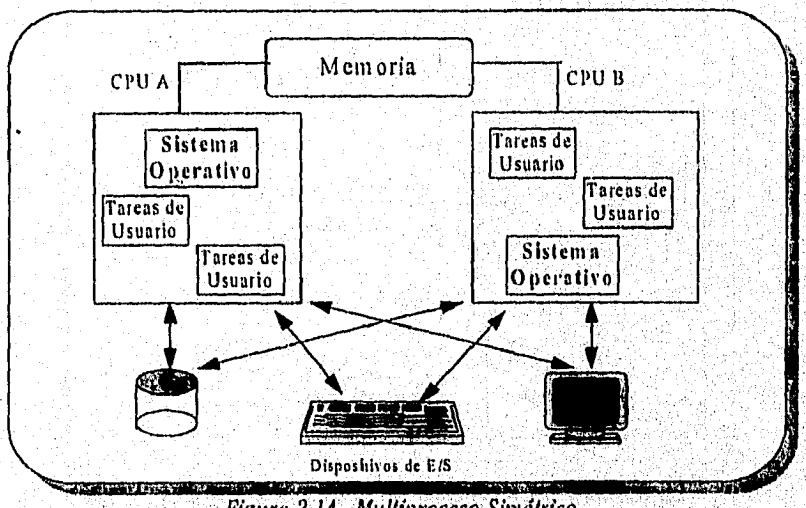


Figura 2.14, Multiproceso Simétrico

Para el control de errores Windows NT tiene una estructura manejadora de excepciones que captura las condiciones del error y responde a ellas de forma uniforme sin afectar a los procesos no involucrados. Windows NT controla excepciones provocadas tanto por software como por hardware como podrían ser llamadas a segmentos de memoria protegidos o divisiones entre cero; para estos existe un filtro que decide que tan grave es la falta o si el programa se culmina, continúa o se realiza otra operación.

Windows esta escrito en lenguaje C, el lenguaje ensamblador solo es utilizado en las partes de el sistema que están en la comunicación directa con el hardware y para aquellos componentes que requieren de una velocidad óptima. Por estas características se dice que es un diseño para ser portable.

Las secciones no portables fueron cuidadosamente aisladas con los componentes que los utiliza. El Kernel de Windows NT se desarrollo para cada tipa de procesador, el cual se utiliza para todas las plataformas basadas en ese procesador. Adicionalmente a la dependencia del procesador, la dependencia de plataforma es otra concepta que Windows NT encapsula en un código de dependencias de plataforma dentro de un DLL conocido HAL o Hardware Abstraction Layer,"Capa de Abstracción de Hardware".

Windows NT utiliza un mecanismo de paso de mensajes de alta velocidad en las llamadas al sistema, llamado Local Procedure Call (LPC), que se encuentra incluido como parte del sistema.

FUNCIONAMIENTO DE WINDOWS NT

Windows NT Advence Server: Este provee extensiones para entrar en un ambiente de redes controlando por dominios (redes grandes orientadas a manejar muchos servidores en un solo ambiente), la plataforma necesaria para interactuar con otros ambientes y mejorar la tolerancia a fallas. Windows NT Advence Server reemplaza al proyecto inicial de vender Windows NT y LAN Manager NT.

Para entender camo Windows NT implementa sus servicios de red se hará referencia a la figura 2.15. correspondiente al modelo OSI.

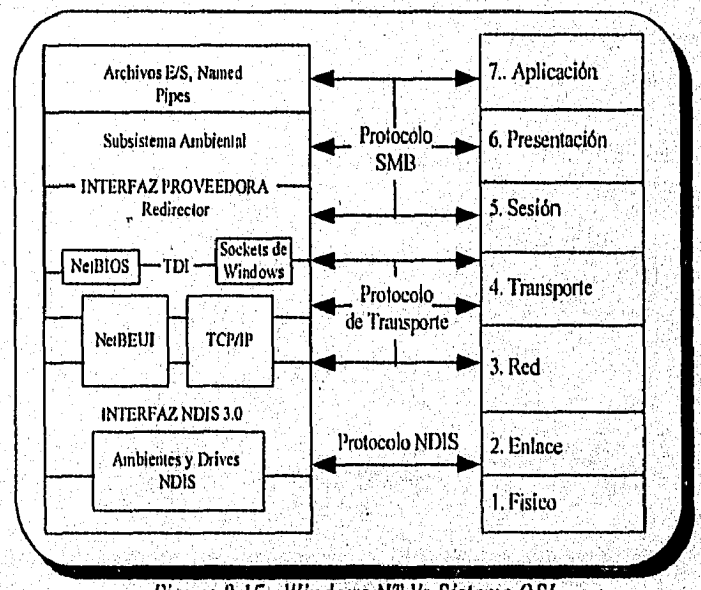


Figura 2.15. Windows NT Vs Sistema OSI

Un software que se encuentra corriendo en moda usuario (user mode) hace una llamada de E/S remota a a través de una llamada a los servicios de E/S nativos de Windows NT, el administrador de E/S de Windows NT crea un paquete de solicitud o IRP (I/O Request Packet) pasando la llamada a un drive del sistema de archivos registrando, que en el caso de Windows NT recibe el nombre de redirector. El redirector pasa el IRP a los drives inferiores (de la capa de transporte)los cuales pondrán el paquete en la red finalmente. Cuando el paquete llega a

otra máquina con Windows NT es recibido por los drives de red localizados en el modo núcleo (Kernel mode), de ahí ascenderá al sistema de archivos del servidor. Este sistema de archivos servidor o simplemente el server pasará al paquete al sistema de archivos local y finalmente al dispositivo físico, tal como se muestra en la figura 2.16.

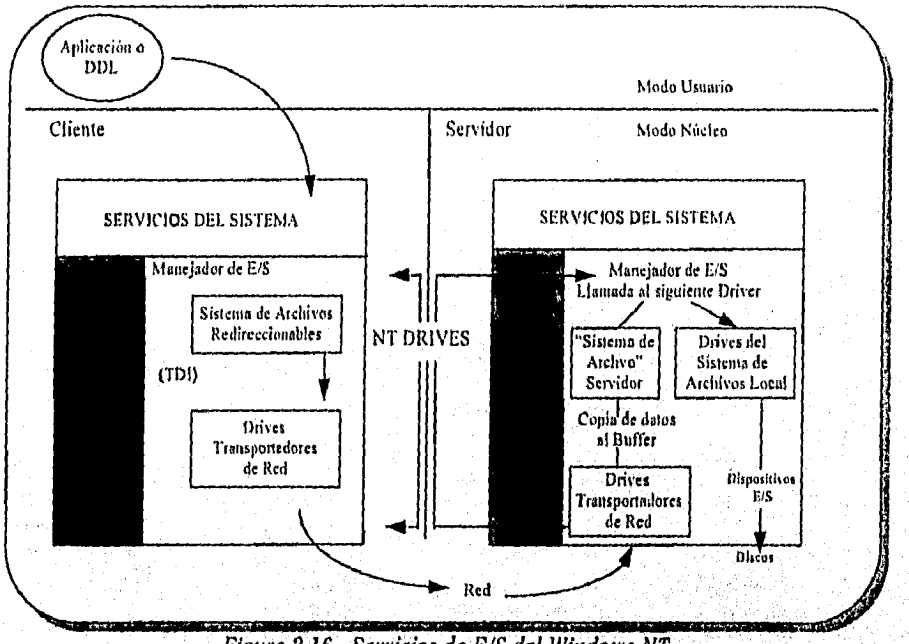


Figura 2.16. Servicios de E/S del Windows NT

La principal ventaja del Driver Server de Windows NT es que esta dentro del ejecutivo de Windows NT y que puede hacer llamadas al administrador de caché directamente, optimizando la transferencia de datos, lo que mejora el tiempo de acceso.

Tanto el redirector como el Server utilizan una interfaz llamada TDI (Transport Driver Interfce) para enviar y recibir las SMB's (Server Massage Block) involucrados en una transacción. El TDI es un conjunto de rutinas colocados como otro driver en el sistema, que a su vez puede platicar con los drives de transporte cargados dentro de la máquina con Windows NT.

Windows NT soporta el acceso a distintos sistemas de archivos a través de una librería llamada Wnet API, permite que diferentes protocolos sean cargados al mismo tiempo (Por ejemplo, TCP/IP, NetBEUI,etc.) y que estos llamen a la misma tarjeta de red. El hecho de que el acceso a tarjetas de red sea a través de drives NDIS (Network Driver Interface Specification) garantiza que las tarjetas podrán ser accesadas por varios protocolos, permitiendo a los fabricantes tener un estándar de referencia para crear sus drives.

El proceso distribuido se lleva cuando una serie de tareas son realizadas por varias computadoras comunicadas entre sí, de tal manera que juntos realizan un proceso determinado de forma transparente para el usuario. Windows NT provee la plataforma necesaria para crear y correr aplicaciones distribuidas.

Para la implantación de esta tecnología, Windows NT incluye el servicio de RPC, este servicio permite a los programadores desarrollar aplicaciones que tengan llamadas a procesos locales y procesos remotos.

En comunicaciones remotas Windows NT Advanced Server provee un servicio incluido llamado RAS (Remote Access Server), por medio del cual se pueden utilizar enlaces de X.25, líneas telefónicas conmutadas o privadas para comunicarse.

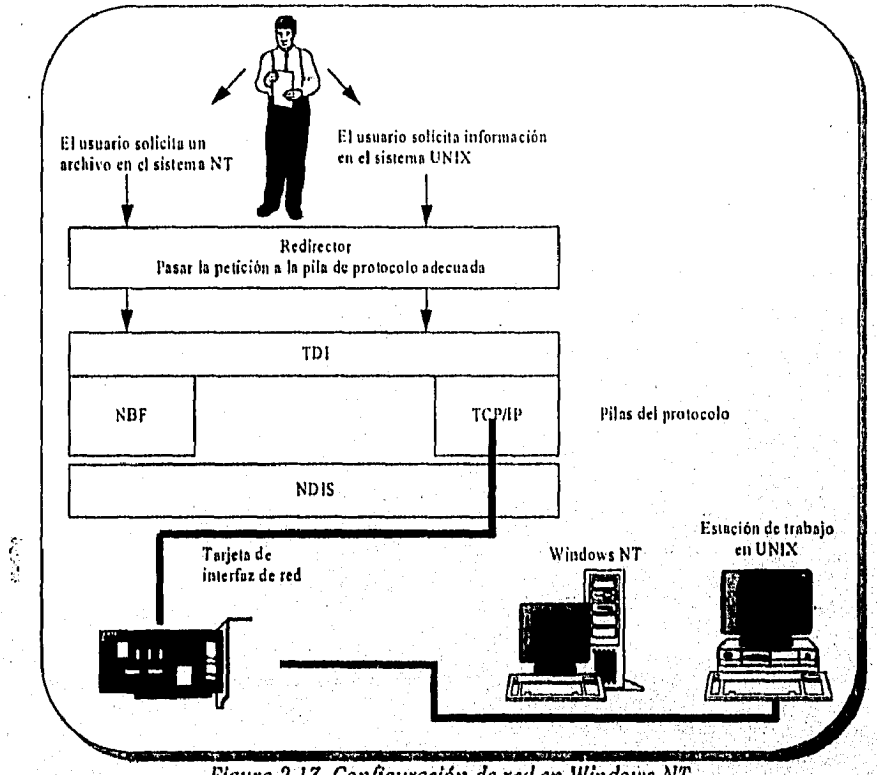


Figura 2.17. Configuración de red en Windows NT.

2.4 NETWARE "NOVELL"

Novell ofrece una serie de sistemas operativas de red bajo el nombre de NetWare, desde el básico y económico NetWare Lite hasta el carporativo. Las lines de productos que ofrece son :

- NetWare Lite. Sistema operativo de red par a par de 2 a 25 usuarios. Se ejecuta sobre el sistema operativo DOS y es compatible con Windows de Microsoft. Los usuarios con poco conocimiento sobre conexión de redes, pueden establecer una compartición de archivas, aplicaciones e impresoras.

- NetWare 2.x. Diseñado para grupos de trabajo y oficinas de pequeño a medio tamaño dentro de grandes campañías. El sistema operativo se ejecuta tanto en modo dedicado como no dedicado en computadoras basadas en 80286, 80386 y 80486 de Intel. Proporciona soporte para la interconexión de red local y remota, como las herramientas para los administradores de red.

- NetWare 3.x. Esta diseñado para dar saporte a cientos de usuarios en un único servidor dedicada. Ofrece muchas de las utilidades avanzadas tratadas en esta sección, inclusa el diseño modular y la capacidad de integrar distintos sistemas.

- NetWare 4.x. Sistema operativo corporativo de Novell que hereda todas las capacidades de NetWare 3.x y añade nuevas utilidades propias para crea un entorno multiservidor distribuida.

El sistema operativo NetWare reside en un servidor de red que es normalmente una computadora de Intel, proporcionondo servicios y conexiones de red a las estaciones de trabajo. En la figura 2.18. se ilustra la relación entre el servidor y el cliente. Esta relación ofrece un sistema de comunicación que distribuye los servicios de red a los clientes. Una de los componentes más importantes del cliente es el software de redirección, que se carga generalmente cuando se arrancan las computadoras. El redireccionador intercepta las órdenes para los servidores de NetWare y las envía a través de la red. Las órdenes que no vienen de la red se envían al sistema operativo local.

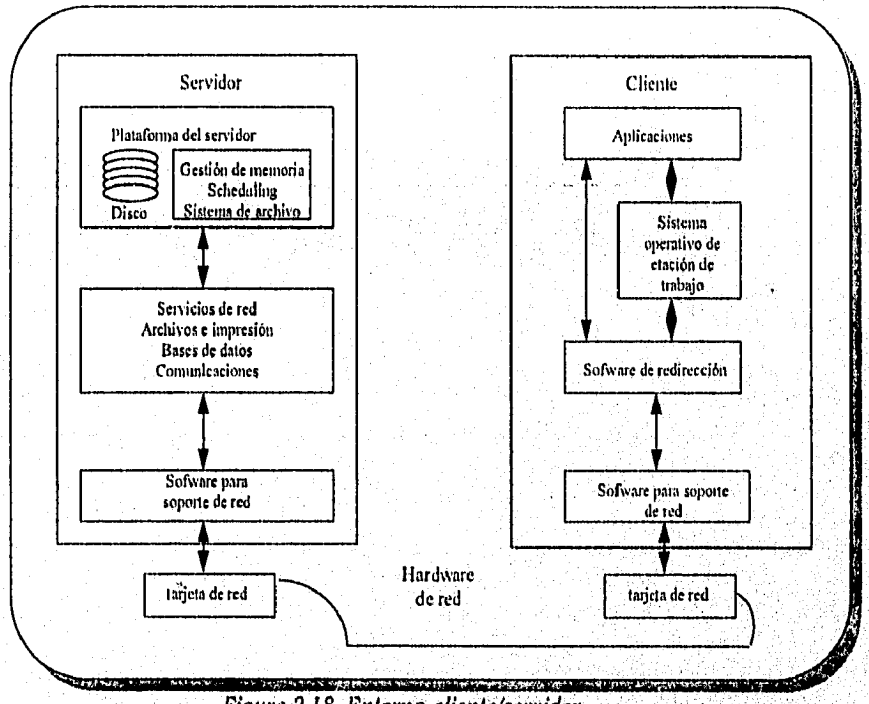


Figura 2.18. Entorno cliente/servidor

Las principales funciones proparcionadas por el servidar de NetWare son la gestión del sistema de archivos, la de memoria y la planificación de las tareas de procesamiento. En la figura anterior se observa la relación entre el servidor y la estación de trabajo que esta basada en la *arquitectura cliente/servidor*, donde las estaciones de trabajo realizan la carga de procesamiento, permitiendo al servidor realizar sus propias tareas más eficientemente.

ARQUITECTURA DE NETWARE

NetWare 3.x y 4.x son sistemas operativos completos de 32 bits que utilizan un único espacio de direcciones sin segmentación. Además, son modulares y ampliables, permiten los cambios, actualizaciones y adiciones de la red. Se puede incluir un Módulo Cargable de NetWare (NLM, Netware Loadable Modules), ejemplos de NLMs son :

- Soporte que permite el almacenamiento de archivos no DOS.

- Servicios de comunicación
- Servicios de bases de datos.
- Servicios de mensajería.
- Servicios de archivodo y copias de seguridad.
- Servicios de gestión de red.

Estos servicios se añaden al sistema operativo como se muestra en la figura 2.19.

Se puede cargar o descargar cualquier módula de la consola del servidor cuando se quiera sin desconectarlo. Cada módulo utiliza memoria adicional, así que es necesario asegurarse de que el servidor tiene suficiente memoria para gestionar los NLMs que se intente cargar. Debido a que los módulos se localizan en el servidor junto con el sistema operativo, se adhieren a él y posibilitan el acceso a los servicios de forma instantánea.

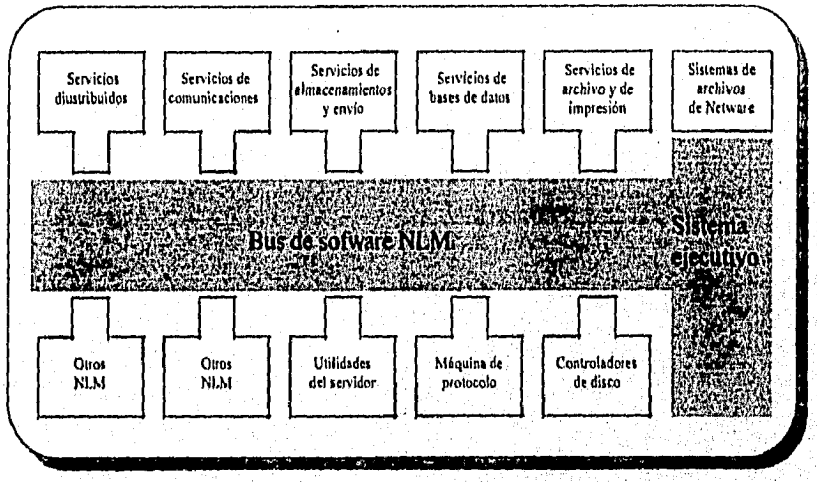


Figura 2.19. Netware Modulos cargables (NLMs)

Una desventaja es que pueden monopolizor los recursos del servidor y este acaparamiento es controlado por NLMs que no liberen su proceso de memoria. Generalmente estos NLMs se encuentran en C, pero pueden escribirse en lenguaje ensamblador.

La ISO, define cinco áreas de administración de redes: *Configuración, rendimiento, fallas, contabilidad y seguridad.* Los NLMs están concentrados en rendimiento, fallas y seguridad.

INDEPENDENCIA DE PROTOCOLOS

Una de las características más importantes de NetWare es su soporte para otros sistemas operativos. Puede unir estaciones de trabajo que ejecuten en DOS, Windows, OS/2 y UNIX. El software suministrado de NetWare para estaciones de trabajo de OS/2 proporciona el soporte que se necesita para comunicarse con un servidor NetWare. NetWare permite los atributos extendidos y los nombres de archivos largos de OS/2, y que las aplicaciones del servidor de OS/2 se ejecuten en la red. Se pueden añadir a la red productos opcionales como soporte para Macintosh de Apple, NFS basado en UNIX, Acceso y gestión en la transferencia de archivos (FTAM, File Transfer Access and Management) de OSI.

NetWare utiliza una estructura de protocolo independiente conocida como la Interfaz abierta de enlace de datos (ODI, Open Data-link Interface), que proporciona soporte simultáneo para los distintos protocalos en la red. En la figura 2.20. se muestra la interfaz ODI que ofrece soporte

multiprotocolo al servidor, observándose que se permite diversas tarjetas de interfaces. Los controladores de estas tarjetas se enlazan al nivel de lo Interfaz abierta de enlace de datos. Los paquetes se dirigen a la pila de protocolos apropiada por encima del nivel ODI, como los sistemas operativos IPX, TCP/IP ó Apple Talk.

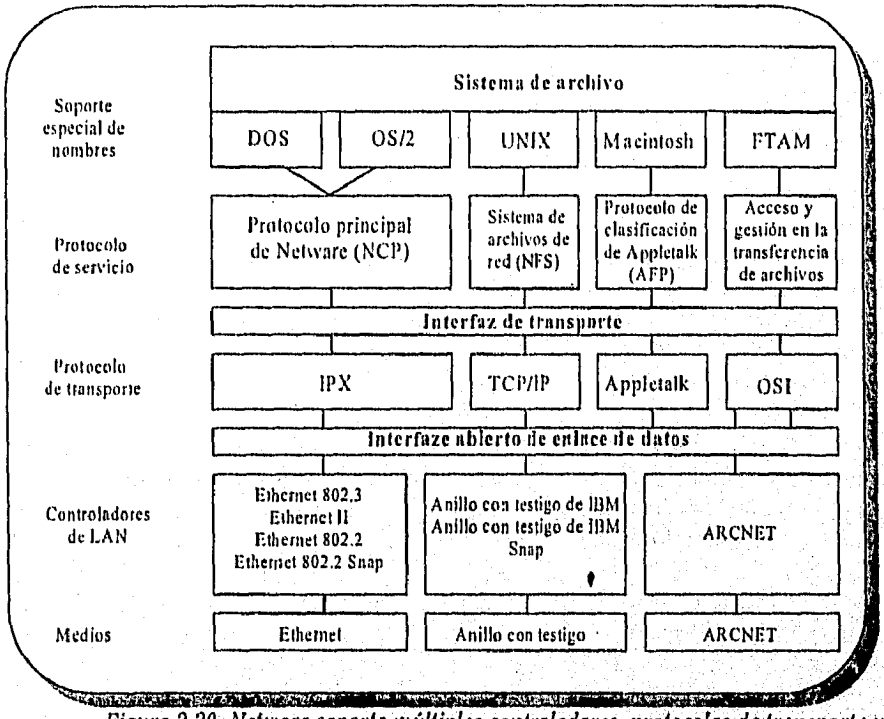


Figura 2.20. Netware soporta múltiples controladores, protocolos de transporte y protocolos de servicio LAN.

Se utilizo un esquema parecido en las estaciones de trabajo para permitir a los usuarios conectarse a redes que utilizan diferentes protocolos de comunicación como TCP/IP.

Si se necesito conectar estaciones de trabajo o LANs de NetWare tanto como a otros tipos de redes, como los redes LAN Manager de Microsoft, LAN Server de IBM y 3Com 3+Share, se puede instalar el controlador para soporte de red de la interfaz abierta de enlace de datos (ODINSUP, Open Data-link Interface Network Support) suministrado con el paquete de NetWare. ODINSUP permite que coexistan la interfaz del controlador de red de ODI y la especificación de la interfaz del controlador de red (NDIS, Network Driver Interface Specification), que es una especificación de Microsoft para el soporte de múltiples controladores.

GESTIÓN DE MEMORIA

NetWare soporta hasta 4 GB (gigabytes) de RAM en el servidor. La gestión de memoria en NetWare 4.x se diseño para incrementar su eficiencia. NetWare v3.11 asigna memoria para diferentes usos en cinco o más zonas. Esto produce que algunas aplicaciones se queden sin ella debido a que, cuando un proceso utiliza la memoria, las rutinas de gestión no la reasignan para otros usos. NetWare 4.x gestiona la memoria como si se tratase de una única zona y es más efectiva en la asignación de memoria entre operaciones.

EL SISTEMA DE ARCHIVOS

El Sistema universal de archivos en NetWare ofrece muchas utilidades que mejoran la eficiencia:

- Método del ascensor (elevatorseeking). Esta utilidad del sistema da prioridad a las peticiones de lectura que entran, proporcionando su localización actual. El funcionamiento del método del ascensor es análogo al del elevador de un edificio. No sube y baja pisos al azar, simplemente se basa en quién solicitó el servicio primero, se para en los pisos intermedios hacia arriba o hacia abajo para recoger pasajeros que necesitan montarse. El método del ascensor minimiza el movimiento de la cabeza del disco, de esta manera mejora el tiempo de acceso y reduce la degradación del hardware.

- Escrituras en paralelo. Las escrituras en disco se gestionan independientemente de las lecturas. Esta separación permite que el sistema operativo escriba datos en el disco cuando disminuyen las peticiones de los usuarios al mismo. Las escrituras en paralelo dan mayor prioridad a los usuarios que necesitan leer datos de la unidad, lo cual mejora las prestaciones desde su punto de vista.

- Búsquedas solapadas. Esta utilidad de NetWare está disponible si hay dos o más discos rígidos y cada uno se conecta a su propio controlador (canal de disco). NetWare accede a todos los controladores simultáneamente. Si se enlazan dos discos a un drive, sólo se accede a uno de estos discos en un momento dado.

- Turbo FAT. Esta utilidad también se conoce como tabla de asignación de archivos indexada. La turbo FAT indexa esta tabla de archivos sobre 2 MB, así el sistema operativo localiza los segmentos de la FAT de forma inmediata sin necesidad de leerla.

- Compresión de archivos. NetWare 4.x puede incrementar el espacio del disco hasta un 63 por ciento con su utilidad de compresión de archivos. NetWare gestiona la compresión en paralelo. Los administradores y usuarios pueden marcar los archivos que serán comprimidos después de utilizarlos.

- Se permiten archivos con un tamaño de hasta 4 GB y el sistema de archivos soporta más de 2 millones de directorios y archivos por volumen, y 100,000 archivos abiertos. Los volúmenes se pueden extender sobre múltiples unidades de disco y se puede incrementar dinámicamente su tamaño si se añaden nuevas unidades.

SEGURIDAD

Las utilidades de seguridad de NetWare son críticas para grandes entornos de la red corporativa. Los sistemas de archivos de NetWare y de DOS son bastante diferentes; un usuario no puede acceder al sistema de archivos de NetWare si arranca el servidor con un disco DOS o simplemente si inicializa el disco. Por supuesto, esto no evita que la gente robe o destruya un disco; aun se necesita realizar una copia de seguridad para protegerse, pero un ladrón no puede acceder a los datos sin los derechos y claves de acceso.

- Seguridad de inicio de sesión/palabra clave. Los usuarios escriben la orden LOGIN para tener acceso al sistema de archivos. Introducen su usuario y después una palabra clave. No se permite el acceso sin, la clave.

- Restricciones de cuenta. En NetWare, el administrador de la red gestiona la cuenta de cada usuario. Se pueden, aplicar restricciones a las cuentas para controlar, cuando iniciar la sesión los usuarios, las estaciones en las que pueden iniciarla y cuando finalizan sus cuentas. También es posible forzar a los usuarios a cambiar sus palabras clave regularmente.

- Seguridad de archivos y objetos. En NetWare4.x, el administrador de la red otorga a los usuarios derechos de administrador a objetos tanto como a directorios y archivos. Estos derechos determinan exactamente cómo pueden acceder los usuarios o los recursos del sistema.

- Seguridad inter-red. Los Servicios de directorios en NetWare (NDS, NetWare Directory Services) siguen o todos los objetos en una inter-red, entre los que se cuentan los objetos de usuario y sus derechos de acceso. Los administradores de la red utilizan NDS para la creación y gestión de cuentas de usuarios, seguimiento de los recursos de la red y asignación de acceso a estos recursos para los usuarios.

SERVICIOS DE COMUNICACIÓN

Novell vende paquetes completos para dar soporte a las líneas de comunicación, conocidos como Servicios de comunicación de NetWare y se ejecutan en los servidores de NetWare V.3.11 y 4. Estos productos permiten conexiones LAN o anfitrión "host", LAN a LAN y remota o LAN. Los Servicios de comunicación de NetWare se centran en las necesidades de gestión de red y de LAN a anfitrión para organizaciones con grandes redes de arquitectura de sistemas en red (SNA, Systems Network Architecture).

2.5 UNIX

Unix fue creado, en Bell Labs por Brian Kernighan y lo llamo UNICS (Sistema de Información y Computo con uniplexión), poco tiempo después lo llamo como todas lo conocemos Unix.

Ken Thompson reconstruyo el Unix para un equipo PDP-11 y desde ese momento y durante la década de los 70 en casi todos los Universidades encontramos un equipo PDP con Unix.

UNIX PORTABLE

Una vez escrito Unix en C, su traslado a otra máquina era relativamente fácil, si se cuenta con un compilador en C. Steve Jhonson "Bell Labs", diseño e implementa el compilador portable de C. En 1984, la compañía AT&T lanzó su producto comercial Unix el Sistema II; No fué bien recibido hasta que solió la versión Sistema V, de esto formo el Sistema V, es uno de los más conocidas.

UNIX BERKELEY

La Universidad de Berkeley, disponía del código fuente del Unix y con apoyo de DARPA (Agencia de Proyectos de Investigación Especiales de la Defensa), produjo y distribuyó una versión para la PDP-11, llamada 1BSD; Pero más importante fueron las versiones 3BSD y 4BSD

para el equipo VAX. Además introdujo el sistema de redes lo que hizo que el protocolo de redes BSD, *TCP/IP*, se convirtiera en un estándar.

UNIX ESTANDAR

A fines de la década de los 80, se utilizaban dos versiones y casi totalmente incompatibles, Unix:4.3BSD y la versión 3 del sistema V.

Los primeros intentos por generar un estándar de unix, se iniciaron con la IEEE Standards Board, bajo un proyecto llamado POSIX Portable Operating System y las letras IX, de la referencia Unix.

Pero de un grupo de vendedores como IBM, DEC, Hewlett-Packard y otros forman un consorcio con el nombre de OSF, Open Software Foundation, para generar un estándar similar al de IEEE.

La reacción de AT&T fue de creor su propio consorcio denominado UI, Unix International, para hacer lo mismo. IBM tiene su propia variante, AIX y así podemos hablar de otros compañíos y lo que antes solo existían dos versiones, después fue más difícil el generar un estándar de Unix.

POSIX genero varios estándares, que se resumen en la siguiente tabla:

Estándar	Descripción	Estándar	Descripción
1003.0	Guía y panorama general	1003.5	Enlace con ADA
1003.1	Funciones de Bibliotecas (llamadas al sistema)	1003.6	Extensiones de seguridad
		1003.7	Administración del Sistema
1003.2	Shell y utilerías	1003.8	Acceso transparente a archivos
1003.3	Métodos de prueba y conformación	1003.9	Enlace con Fortran 77
1003.4	Extensión a tiempo real	1003.10	Supercómputo

Unix es un sistema interactivo de tiempo compartido, fue diseñado por progromadores para progromadores y cuyo uso existen usuarios relativamente sofisticados.

El sistema Unix puede representarse como una pirámide ver en la figuro 2.21.

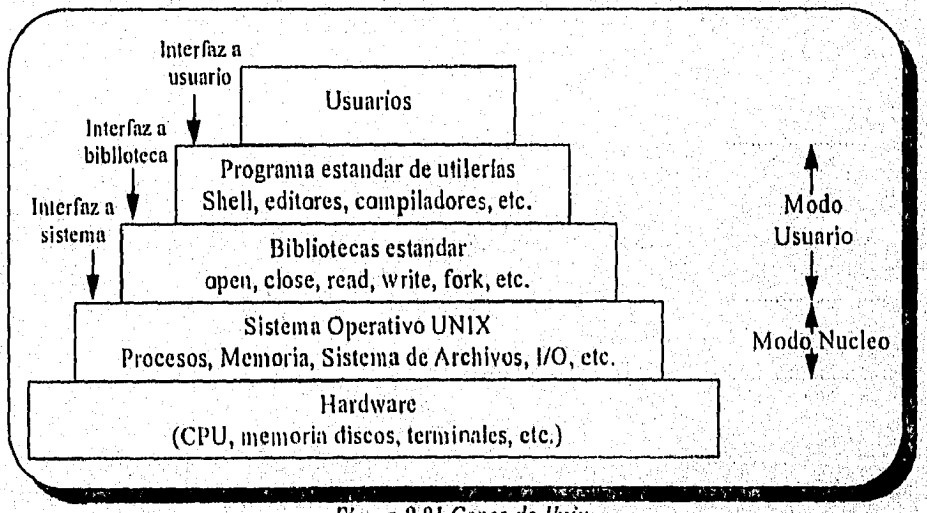


Figura 2.21 Capas de Unix

Varios características de Unix han contribuido a su extensa difusión, entre ellas:

- Flexibilidad para adaptarse a aplicaciones diversas.
- Potencia y capacidad.
- Portabilidad que permite usarse en casi todos los ordenadores a partir de un cierto tamaño.
- Compartición de sus Archivos con MS-DOS.

Los requerimientos que necesita Unix son de 2.5 megabytes de memoria RAM y 40 Mb de espacio en disco duro, maneja más de 400 órdenes y 2500 Archivos esto lo hace poco accesible a usuarios pequeños.

ARCHIVOS

El sistema de archivos en Unix, se presenta al usuario como una estructura de árbol que se construye a partir de un directorio raíz que identificamos con el carácter "/". Para localizar un determinado archivo, se necesita una ruta de acceso que si se parte de la raíz se llama absoluto y si partimos del directorio donde nos encontramos en instancia se llama camino de acceso relativo.

Los archivos especiales sirven para designar a los periféricos. Se agrupan en un directorio llamado "/dev".

Cada archivo independiente tiene asociado un conjunto de indicadores que describen los derechos de acceso al archivo para el propietario, los miembros del mismo grupo y los demás usuarios. Los nombres de los derechos para cada categoría son R (lectura) W (escritura) y X (ejecución). Al obtener la información que afecta a los archivos podemos mencionar las siguientes características:

El tipo de archivo ("-" para archivos normales, "d" para directorios, y "b" para archivos especial). Los derechos de acceso (r,w,x) para propietario, grupo y demás usuarios.

El sistema de archivos se encarga de establecer la correspondencia entre un nombre del árbol y el lugar físico donde se graba la información de dicho archivo. Los bloques de información se agrupan formando volúmenes que quedan descritos en los siguientes bloques:

1. El primer bloque: bloque 0 no pertenece propiamente al sistema de archivos ya que contiene, usualmente, un programa que se utiliza, si es preciso en la iniciolización del sistema.
2. El segundo bloque: bloque 1 también conocido como superbloque está destinado a contener la información relativa a:

- El número de bloques ocupados por los nodo-i (descriptor de archivos)
- El número de bloques ocupados por el sistema de archivos.
- Un apuntador que señala al comienzo de la tabla de bloques disponibles.
- Los bloques ocupados por los nodos-i contienen la tabla de nodos i.

Entendemos por archivos del sistema, aquellos que están incluidos en el directorio raíz.

ARCHIVO BOOT: Este archivo contiene el programa que cubre la segunda etapa de la inicialización tras el proceso de carga desde el bloque 0 del sistema del archivo inicial, del programa que cubre la primera etapa de dicha inicialización.

ARCHIVO UNIX: Este archivo contiene los programas correspondientes al núcleo (Kernel) del sistema operativo y es cargado en la memoria interna cubriendo así la tercera parte de la inicialización. A partir de este momento se desencadena un proceso llamada inicialización del sistema.

DIRECTORIOS DEL SISTEMA

Son directorios del sistema aquellos que dependen directamente del directorio raíz. Los nombres de estos directorios son:

/bin
/tmp
/lib
/usr
/etc
/dev

/bin :Este directorio y su alternativa /usr/bin incluye los Archivos que contienen los programas que respaldan las órdenes que están a disposición de la generalidad de los usuarios del sistema.

Los programas que respaldan estas órdenes pueden provenir de la compilación de un programa fuente en C, Pascal, etc, o de un guión del caparazón (shell script).

En todo caso situados a nivel de operador y tras el prompt ($) el usuario dará la orden a ejecutar por el sistema o, dicha de otro modo, tecleará el nombre del archivo que contiene el programa que respalda a la arden en cuestión. Al pulsar "Intro" el caparazón busca tal orden, secuencialmente a través de los directorios fijados en la variante del caparazón PATH.

Los valores por defecto de PATH son establecidos por el administrador del sistema, siendo estas normalmente.

:/bin :/usr/bin

Lo que indica, que el primer directorio de búsqueda de la orden dada es el directorio actual del usuario (representado por los dos primeros puntos), si no lo encuentra ahí, continuará la búsqueda en el directorio /bin para finalizar en el directorio /usr/bin.

Una vez localizado el Archiva correspondiente a la orden en cuestión, el caparazán concluirá su labor provocando la ejecución del programa contenido en él y se ejecutará la orden, con el servicio que fue requerido.

/tmp : Este directorio y su alternativo /usr/tmp incluye aquéllos archivos cuya existencia es tan breve como la que requiere una determinada transacción, de aquí que se les asigne el calificativo de temporales (tpm). Normalmente, estos archivos desaparecen durante el proceso de reinicilización.

/lib : Este directorio y su alternativo /usr/lib incluye los archivos que contiene las rutinas enlazables con la compilación de programas, y cuyo conjunto es conocido como librería (lib) o biblioteca.

/etc : Este directoria incluye, generalmente aquéllas archivos cuya contenido está relacionado can la administración del sistema y su administradar y del sistema. ejempla :

El Archivo /etc/passwd almacena la infarmación que el sistema requiera para cantralar a tados y cada uno de las usuarios recanacidas.

/dev : Este directario incluye todas las archivos especiales y por tanto sus respectivas contenidos están íntimamente vinculadas con las dispositivos periféricos. Por ejemplo, referencia /dev/lp indicará al Archivo la asociación a la impresara.

/usr : Este directorio es padre de otros directarios cuyos archivos almacenan información, relacionada con las usuarios a es de utilidad para ellos. Camo ejemplo de está se puede citar los directorios:

/usr/man	Dacumentas de Manuales del Sistema de Administración.
/usr/doc	Documentos personales
/usr/games	Almacena los programas de juegos.

COMUNICACION EN UNIX

Existen dos populares interfaces de programas de aplicación de comunicación de redes en ambiente UNIX: *sockets y streams*. Los sockets son introducidas desde 1983 en la Universidad de Berkeley en el Unix 4.2 BSD y los streams por la compañía AT&T en 1986 en su producto Sistema V versión 3. Los sockets y streams son dos ejemplos claros de la clase de procesas llamados IPC (Interproces Comunicotions, "Comunicación de Interprocesos").

SOCKETS

Los sackets se pueden crear y destruir de manera dinámica. La creación de un socket regresa un descriptor de archivo, el cual es necesario para establecer una conexión, leer datos, escribir datos y liberar la conexión. Cada sacket soporta un tipo particular de red, el cual se determina al crear el socket. Los tipos más comunes son:

- Orientado a conexiones por flujo confiable de bytes .
- Orientado a conexianes por flujo confiable de paquetes.
- Transmisión no confiable de paquetes.

El primer tipo de socket permite que das procesas en distintas máquinas establezcan de manera efectiva el equivalente de un tubo entre ellos. Los bytes se envían en un extremo y llegan en el misma orden al extremo contrario.

El segundo tipo es similar al primero, excepto que preserva las fronteras de los paquetes. Si el emisor hace cuatro llamadas independientes a WRITE, cada una con 512 bytes y el receptor pide 2560 bytes, con un socket de tipa 1, los 2560 bytes regresarán. Con un socket de tipo 2, sálo regresarán 512 bytes. Se necesitan cuatro, llamadas más para el resto.

El tercer tipo de socket se utiliza para que el usuario tenga acceso directo a la red. Este tipa es particularmente útil para los aplicaciones de tiempo real y en aquellas situaciones en que el usuario desea implantar un esquema especializado de maneja de errores.

Durante la creación de un socket, una de los parámetros determina el protocolo a utilizar. Para los flujos de paquetes y bytes, el protocolo más popular es TCP/IP. Para una transmisión no confiable orientada a paquetes, es UDP.

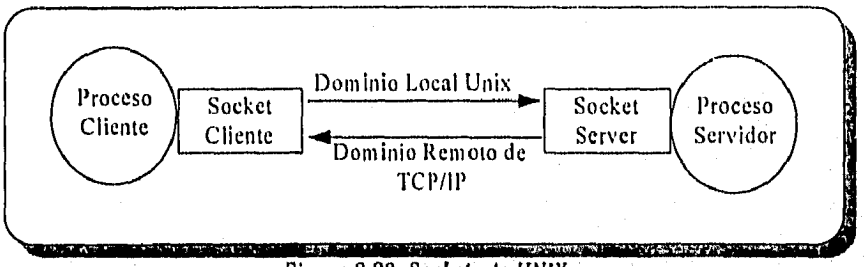


Figura 2.22. Sockets de UNIX.

STREAMS

Los STREAMS se relacionan más a drivers de dispositivos. El kernel del UNIX es responsable de buscar las direcciones de los drivers y módulos de los streams en una tabla llamada *"CHARACTER DEVICE SWITCH"*.

Los STREAMS es una parte importante del sistema operativo UNIX. Permite que las aplicaciones que se ejecutan en niveles superiores accedan a los controladores de dispositivos que se ejecutan en los niveles inferiores. En el entorno de red, STREAMS sirve de interfaz entre las aplicaciones y los controladores de dispositivo de los adaptadores de red. El objetivo de STREAMS es ofrecer la posibilidad de añadir y eliminar módulos del flujo (stream). De esta forma, si por ejemplo se necesita algún tipo de cifrado, se puede añadir un módulo de cifrado al flujo sin necesidad de recompilar el sistema operativo.

La parte superior del flujo se denomina cabeza del flujo y es el lugar donde las aplicaciones interactúan con los módulos del flujo. Los datos y la información de control se transfieren entre el flujo y las aplicaciones por medio de mensajes. Según se muestra en la Figura 2.23, los datos se sitúan en la cabeza del flujo, que es quien se encarga de transmitirla hasta el controlador de la red. En el flujo se pueden insertar nuevos módulos, como, por ejemplo, los protocolos de red OSI y TCP/IP.

Los tipos de mensajes que pueden utilizar los usuarios para interactuar con el flujo son los que define la interfaz proveedora de transporte. La interfaz del nivel de transporte (TLI, Transport Layer Interface) es una biblioteca de llamadas y módulos de tiempo de ejecución, definida inicialmente por AT&T, que simplifica el trabajo de los programadores. El programador no tiene más que utilizar las llamadas de la biblioteca TLI en sus aplicaciones, en lugar de escribir las aplicaciones para que utilicen los protocolos de transporte específicos. Se pueden cambiar o actualizar los protocolos sin que los programas se vean afectados, puesto que el código dependiente del protocolo no está incluido en el programa.

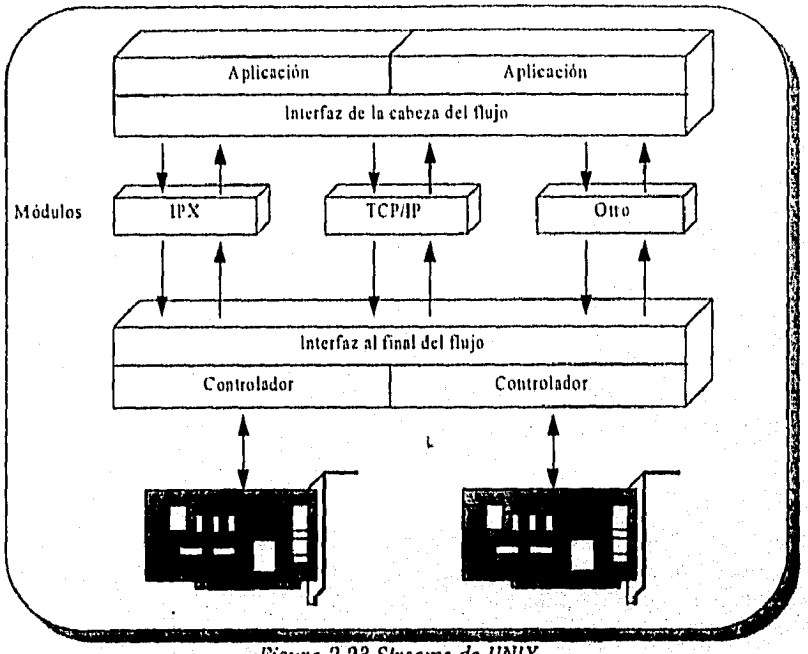


Figura 2.23 Streams de UNIX

Con las funciones que se incluyen en TLI las aplicaciones pueden solicitar servicios de transporte, orientados o no a la conexión, a los protocolos de transporte de niveles inferiores. La aplicación no necesita conocer los detalles de la comunicación a través de la red. Simplemente hace una solicitud. En el nivel TLI hay otras muchas funciones estándar a disposición de las aplicaciones.

2.6 WINDOWS NT COMO SISTEMA OPERATIVO DE D.G.I.

En un inicio la S.H.C.P. tenía como estándar el Sistema Operativo UNIX, en lo que respecta al servicio de comunicaciones. El manejo de bases de datos se realizaba en INFORMIX. Ambos servicios se efectuaban con bastante eficiencia, por tal motivo las oficinas regionales cuentan con equipos H.P. con sistema operativo UNIX y manejadores de base de datos INFORMIX.

Como un proyecto a nivel nacional, se presentó "Automatización de Oficinas" la primera plataforma de cliente/servidor en la Secretaría de Hacienda y Crédito Publico. La fase II de Automatización de Oficinas se llevó a cabo con servidores Windows NT; porque es un sistema operativo bastante robusto y con la seguridad que se requiere para una institución como es la S.H.C.P.

Windows NT cuenta con el nivel de seguridad C-2, en un ambiente cliente servidor es importante la integridad de los datos, seguridad en las bases de datos y seguridad en redes. Los estándares en seguridad analizados para el proyecto fueron:

Modelo de seguridad en redes:	
ISO 10181-5	Seguridad en redes, Parte 5 : Integridad
Integridad:	
IEEE 802.10B	Seguridad en intercambio de datos

Windows NT fue configurado de un servidor primario y varios servidores de backup, de esta forma sólo existe un servidor que tiene dominio de la red, por lo que las relaciones de confianza están normadas por la cuenta del administrador del servidor primario. La copia de la base de datos de usuarios se encuentra en todos los servidores de backup a efecto de que si se presenta algún problema o el servidor primario se encuentra en mantenimiento, cualquier servidor de backup pueda desempeñarse como servidor primario "replicación"[5].

El acceso en Windows NT, es controlado por cuentas de usuarios que son creadas por el administrador, si un usuario desea iniciar una sesión necesita conocer la cuenta y el password, este proceso es conocido como "Discovery". Además los recursos compartidos como archivos e impresoras cuentan con permisos y passwords asignados algunos por el administrador y otros por los usuarios.

Las herramientas administrativas para el control de usuarios son bastante aceptables. Con ellas se tiene el control de la fecha de expiración de una cuenta, desde qué estación de trabajo iniciará una sesión, las horas en que esta permitido accesar al servidor y los perfiles del usuario, como pueden ser que al iniciar se ejecute un archivo de arranque "archivo con extensión .bat".

---

[5] Replicación:- Es la habilidad de un servidor duplicar y sincronizar sus bases de datos con otros servidores.

Para ilustrar lo anterior se muestra la pantalla de administración de usuarios, figura 2.24, que al intentar ser accesada por un usuario que no cuenta con los permisos para crear o modificar los perfiles de un usuario, se le niega el acceso.

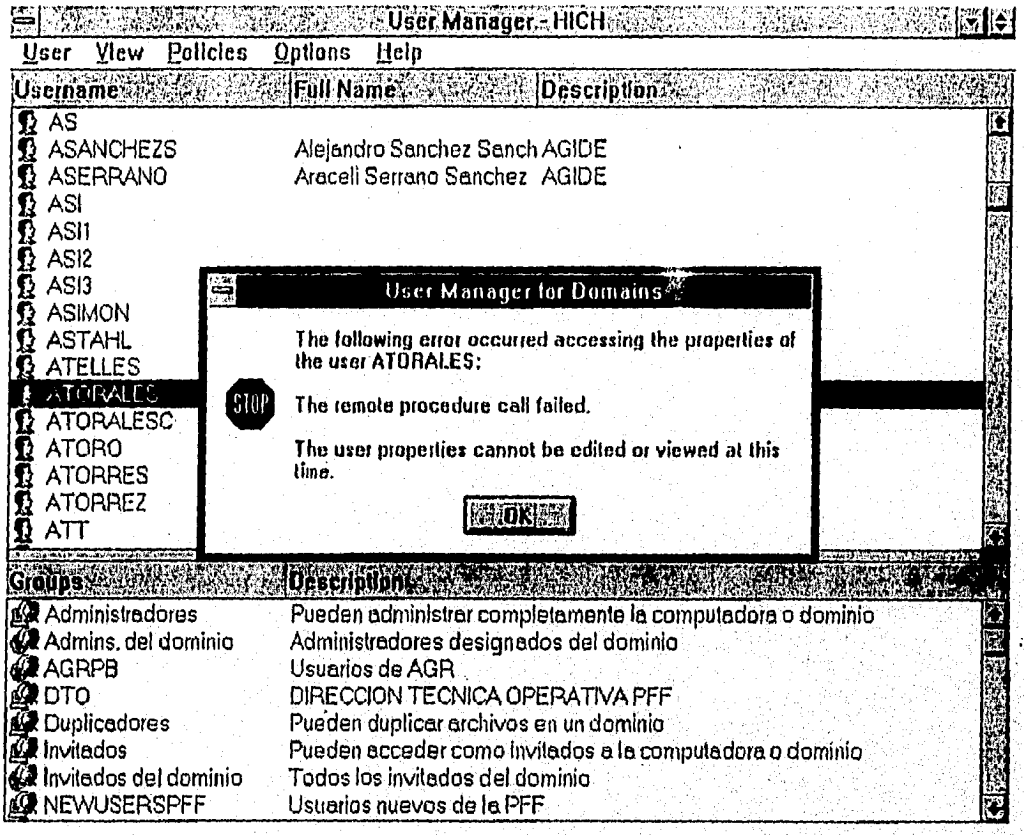
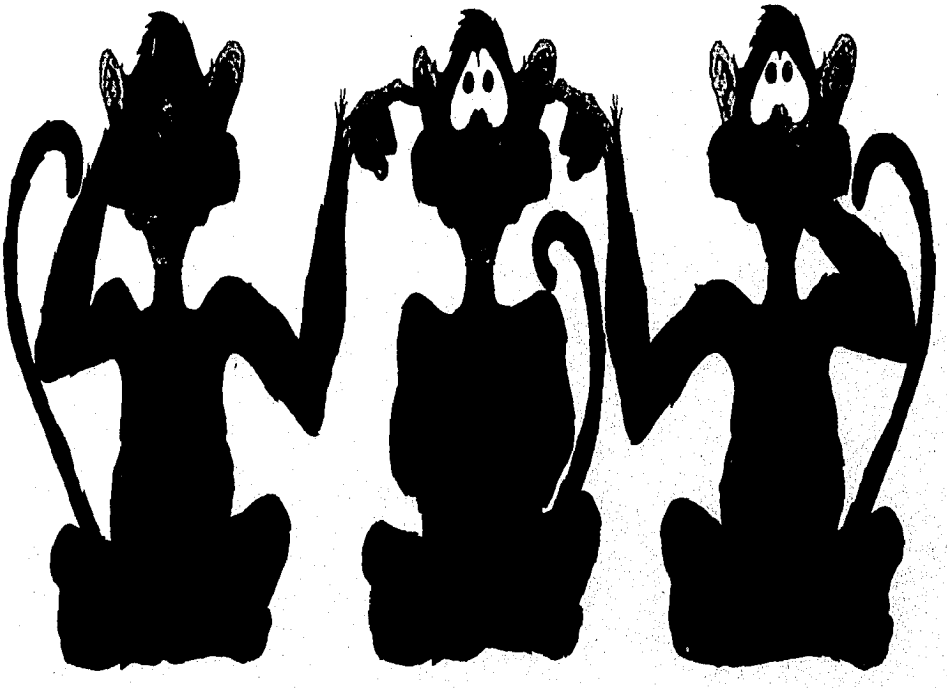


Figura 2.24 .- Acceso denegado en Windows NT.

Actualmente en las estaciones de trabajo que se desempeñan como clientes, se tiene instalado el sistema operativo Windows para Trabajo en Grupo 3.11, aunque en algunas de ellas se está experimentando con Windows NT Workstation 3.51 ó Windows 95.

CAPITULO III

COMUNICACION



HELLO, GOODBYE

YOU SAY YES, I SAY NO,
YOU SAY STOP AND I SAY GO, GO, GO.

OH NO
YOU SAY GOODBYE AND I SAY HELLO,
HELLO, HELLO.
I DON'T KNOW WHY YOU SAY GOODBYE,
I SAY HELLO, HELLO, HELLO.
I DON'T KNOW WHY YOU SAY GOODBYE,
I SAY HELLO.

I SAY HIGH, YOU SAY LOW,
YOU SAY WHY AND I SAY I DON'T KNOW, OH NO
YOU SAY GOODBYE, AND I SAY HELLO,
HELLO, HELLO.
I DON'T KNOW WHY YOU SAY GOODBYE,
I SAY HELLO.

HELLO, HELLO
I DON'T KNOW WHY YOU SAY GOODBYE,
I SAY HELLO.

LENNON & MCCARTNEY

CAPITULO 3

COMUNICACION

En el ambiente cliente/servidor existen varias computadoras autónomas, pero interconectadas, capaces de intercambiar información, la conexión puede hacerse a través de hilos de cobre, láser, fibra óptica, microondas y satélites, utilizando dispositivos que permiten la conectividad local y remota de las computadoras (puentes, ruteadores y gateways), tabla 3.1.

Toda esta conexión entre computadoras es transparente para el usuario y ve a todo el sistema como la computadora que está utilizando (como si estuviera local).

Distancia entre procesadores	Procesadores ubicados en:	Nombre genérico del sistema
0.1 m.	tarjeta de circuito	Máquina de flujo de datos
1 m.	sistema	Multiprocesador
10 m. 100 m. 1 km.	cuarto edificio terreno de una universidad	Red de Area Local (LAN)
10 km.	ciudad	Red de Area Metropolitana (MAN)
100 km. 1000 km. 10,000 km.	país continente planeta	Red de Area Amplia (WAN)

Tabla 3.1 Comparativa de diferentes medios de comunicación.

Para facilitar la comunicación entre ambientes heterogéneos la Organización Internacional de Estándares, ISO (International Standards Organization), definió un modelo al que llamó OSI (Open System Interconect), el cual tiene una estructura lógica y está dividido en 7 niveles de protocolos (se refiere a la manera como los datos viajan de una estación a otra).

La clave del éxito en un medioambiente cliente/servidor es la capacidad de accesa de datos de cualquier modo en el sistema. Las bases de datos que residen en el servidor, son accesadas por herramientas que residen en la computadora que actúa como cliente. Idealmente el servidor de base de datos, debe proveer integridad, seguridad y funcionalidad. La mayoría de las bases de datos actualmente cuentan con un lenguaje de acceso de datos SQL (Lenguaje Estructurado de Consultas). Es importante mencionar que están empezando a utilizarse bases

de datos orientadas a objetos que cuentan con la habilidad de manejar diferentes tipos de datos, incluyendo complejas tipos que se almacenan en la estructura de la base de datos, como son los BLOBs (Objetos Binarios de Gran Tamaño), incluyen entidades como hojas de cálculo, imágenes digitalizadas y documentos realizados en procesadores de palabras.

3.1 MODELO OSI

En 1978, la ISO elaboró un modelo para la interconexión de computadoras en red denominado modelo de referencia para la Interconexión de sistemas abiertos (OSI). Este modelo describe el flujo de datos dentro de una red, desde las conexiones físicas hasta el nivel superior, es decir, las aplicaciones que utilizan los usuarios finales.

El modelo de referencia OSI consta de siete niveles, ver tabla 3.2. El nivel más bajo, conocido como nivel físico, es donde los bits de datos se transfieren físicamente al cable. El nivel más alto es el de aplicación, que corresponde a la presentación de las aplicaciones a los usuarios. Los principios aplicados para el establecimiento de división fueron:

1 Una capa se creará en situaciones en donde se necesita un nivel diferente de abstracción.

2 Cada capa deberá efectuar una función bien definida.

3 La función que realizará cada capa deberá seleccionarse con la intención de definir protocolos normalizados internacionalmente.

4 Los límites de las capas deberán seleccionarse tomando en cuenta la minimización del flujo de información a través de las interfaces.

5 El número de capas deberá ser lo suficientemente grande para que funciones diferentes no tengan que ponerse juntas en la misma capa y, por otra parte, también deberá ser lo suficientemente pequeño para que su arquitectura no llegue a ser difícil de manejar.

Capa 7	**Nivel de aplicación**
Capa 6	**Nivel de presentación**
Capa 5	**Nivel de sesiones**
Capa 4	**Nivel de transporte**
Capa 3	**Nivel de redes**
Capa 2	**Nivel de enlace**
Capa 1	**Nivel de control físico**

Tabla 3.2 Capas del modelo OSI

El nivel físico es responsable de la transferencia de bits de una computadora a otra. Se encarga de regular la transmisión de una secuencia de bits a través de un medio físico. Este nivel define el modo en que se conecta el cable a la tarjeta adaptadora de red y la técnica de transmisión empleada para enviar los datos a través del cable. También define la sincronización de bits y las operaciones de comprobación.

El nivel de enlace empaqueta los bits sin procesar procedentes del nivel físico, agrupándolos en tramas. Una trama es un paquete lógica estructurado en el cual pueden colocarse datos. El

nivel de enlace es responsable de transferir tramas de una computadora a otra, sin errores. Una vez que el nivel de enlace envía una trama, queda esperando una aceptación o acuse de recibo procedente de la computadora destinataria. Las tramas para las cuales no se recibe una aceptación vuelven a enviarse.

El nivel de redes direcciona los mensajes y traduce las direccianes y nombres lágicos, convirtiéndolos en direcciones físicas. También determina la ruta a través de la red entre la computadora de arigen y la de destino, y administra aspectos asociadas al tráfico en la red como la conmutación, el encaminamiento y el control de la congestión de paquetes de datos.

El nivel de transparte es responsable de detectar y resolver errores, can el fin de garantizar la fiabilidad en la entrega de los mensajes. También se encarga de volver a empaquetar las mensajes cuando es necesario, dividiendo los mensajes de gran longitud en mensajes más cortos para su transmisión y reconstruyendo posteriormente, en el extremo receptor, el mensaje original a partir de los paquetes más pequeños. El nivel de transporte del extremo receptar se ocupa también de enviar la aceptación o acuse de recibo.

El nivel de sesiones permite a dos aplicaciones situadas en distintos computadoras establecer, utilizar y terminor una sesión. Este nivel establece el control del diálogo entre las dos computadoras que participan en una sesión, regulando cuál de los dos extremos transmite, cuándo lo hace y durante cuánto tiempo.

El nivel de presentación traduce los datos desde el nivel de aplicación hasta un formato intermedio. Este nivel se encarga también de cuestiones relacianadas con la seguridad, proporcianando servicios como encriptado de datas o campresión de la información, para que se transmitan menos bits a través de la red.

El nivel de aplicación es el que permite a las aplicaciones de usuaria final acceder a las servicios de la red.

Cuando dos computadoras se comunicon a través de una red, el software de cada uno de los niveles asume que se está camunicando con el nivel homólogo de la otra computadora. Por ejemplo, el nivel de transparte de una de las computadora se comunicará con el nivel de transporte de la otra. El nivel de transporte de la primera computadora no necesita preocuparse del modo en que la comunicación atraviesa realmente los niveles inferiores de la primera computadora, recorre el medio físico y, por último, vuelve a ascender a través de los niveles inferiores de la segunda computadara.

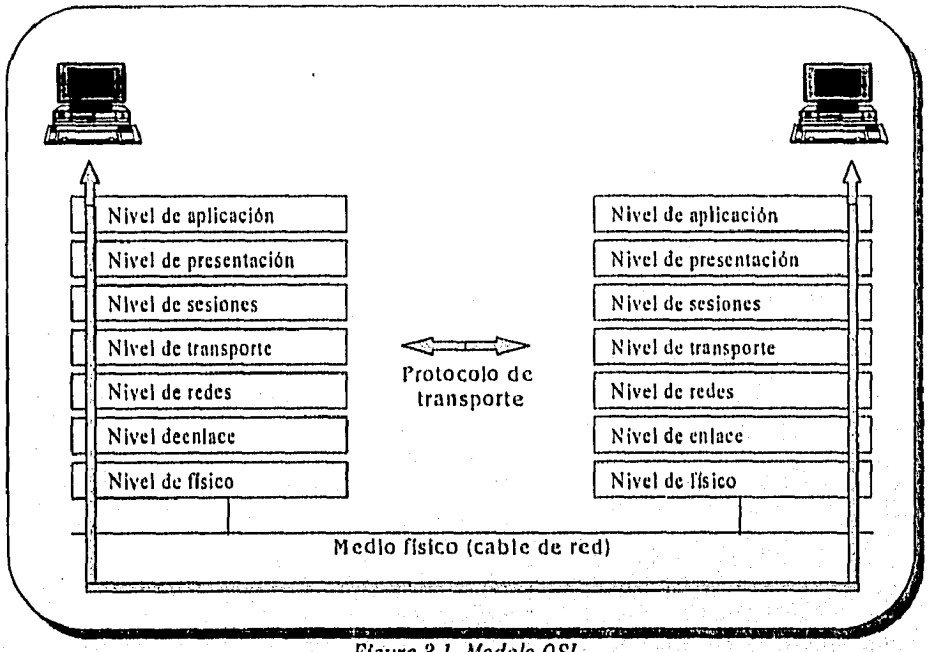


Figura 3.1 .Modelo OSI

La importoncio del modelo OSI se debe o que ho conseguido presentar una visión global y estructurado del problema de la interconexión de sistemas abiertos, aunque en la práctica aún no ha tenido mucho éxito, debido principolmente a su inmadurez y complejidad, hoy en día, es un indiscutible punto de referencia. Es importante recordar que el término sistema abierto se uso para designar a un sistema capaz de interconectarse con otro, de acuerdo con unas normas establecidas.

3.2 *BRIDGE, ROUTER Y GATEWAY*

PUENTES "BRIDGE"

El puente (bridge) es el dispositivo que intercanecta las redes y proporciona un camino de comunicación entre dos o más segmentos de red o subredes. Un segmento de red o subred tiene la misma dirección de red y el mismo tipo de tecnología de conexión de red. Por ejemplo, la figuro 3.2 ilustra como un servidor praporciona puenteado entre dos odaptadores de red. El puente proporciono un camino o la estación de una red para que difunda mensajes a las estaciones de otras redes. Por tanto es un dispositivo de dos puertas (o más) que une segmentos de red. Por otro lodo, un puente se puede usor para segmentar uno red muy activa en dos segmentos, así se reduce lo cantidad de tráfico existente en codo una y oumentan sus prestaciones.

Los puentes filtran las emisiones entre las redes, lo que permite que sólo el tráfico esencial de inter-red cruce el puente.

Los siguientes razones determinan el uso de puentes:

- Para ampliar lo extensión de la red o el número de nodos que la constituyen.
- Para reducir el cuello de botella del tráfico causado por un número excesivo de nodos unidos.
- Para unir redes distintas como Ethernet y anilla con testigo y enviar paquetes entre ellas, asume que ejecutan el mismo protocolo de red.

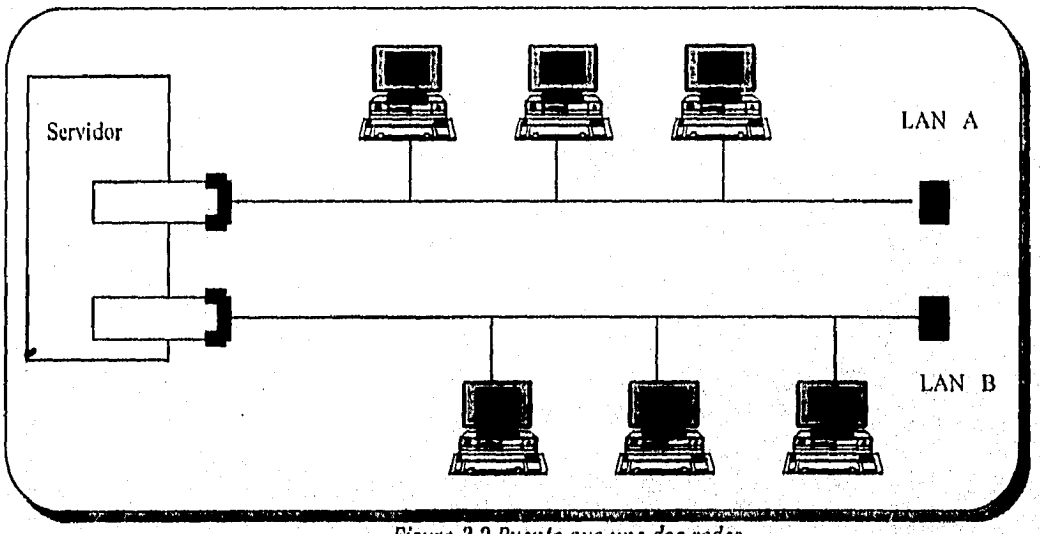


Figura 3.2 Puente que une dos redes

Un puente lo constituye un dispositivo autónomo o se crea en los servidores mediante la instalación de una o más tarjetas de interfaz de red, que suponen que el sistema operativo del servidor soporta puenteado. Cada segmento de Red de Area Local (LAN, Local Area Network) conectado por un puente tiene un número de red distinto.

Los puentes realizan funciones de filtrado mediante el envío de la dirección en la trama Ethernet o en la de anillo de testigo, y así determinan a qué segmento de LAN pertenece el paquete de datos. Sin embargo, los puentes no tienen acceso a la información del protocolo de nivel de red, por este motivo no pueden encaminar por el camino más adecuado. Los routers se pueden programar (o aprenderán) para que encaminen los paquetes por los caminos específicos que reducen los costos o eviten la congestión del tráfico; se pueden usar routers multiprotocolo para el control del tráfico de red con múltiples protocolos de comunicaciones.

Cuando una red crece, el número de conexiones puenteadas también crece, se crea la posibilidad de que aparezcan bucles o caminos ineficientes. Además de ser incapaces de determinar el camino óptimo para enviar los datos, también los puentes carecen de control de la congestión, esta se produce cuando muchas estaciones necesitan transmitir al mismo tiempo. En una red puenteada, se relega el control del flujo al sistema final. Realmente, los puentes pueden añadir problemas a la congestión cuando transmiten excesivos paquetes en un intento de recuperarse de estos problemas.

TIPOS DE PUENTES

Generalmente hay dos tipos de puentes: locales y remotos. Un puente local proporciona puntos de conexión para LANs y se usa para la interconexión de segmentos LAN dentro del mismo edificio o área como muestra la parte inferior de la figura 3.3. Los puentes remotos tienen puertos para los enlaces analógicos y digitales de telecomunicaciones y de esa modo conectan las redes a otros lugares, como muestra la parte superior de la figura 3.3. Los

conexiones entre puentes remotos se hacen sobre las líneas analógicas con el uso de módems o sobre líneas alquiladas digitales como T1 que proporcionan un rendimiento de 1,544 Mbits/seg.

Fundamentalmente una línea analógica es una línea de grado de voz y de enlace telefónico que praporciona un circuito para la conexión de puentes en función de un enlace telefónico temporal o de forma permanente.

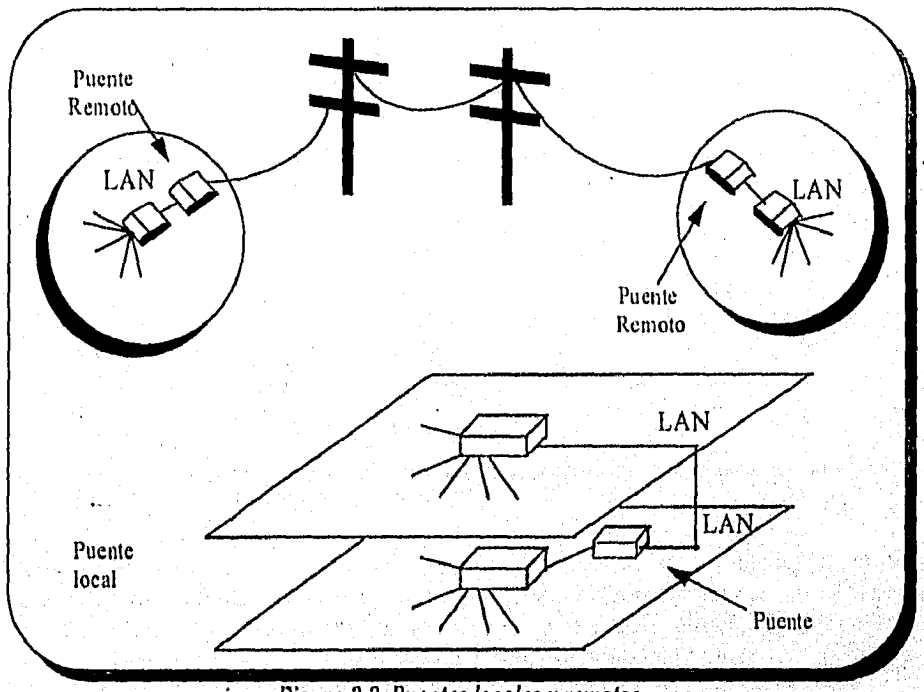


Figura 3.3. Puentes locales y remotos

FUNCIONALIDAD DEL PUENTE

Un puente une dos segmentos de LAN semejantes o no. Se puede consideror un puente como un clasificador de correo que exomina las direcciones de los paquetes y las coloco en los segmentos apropiados de lo red. El puenteado se realiza en el nivel de vinculación de datos, con relación al modelo de protocolo Interconexión de Sistemas Abiertos (OSI Open System Interconnection). Cualquier dispositiva que se ajuste a las especificaciones del control de acceso al medio (MAC, Medio Access Control) de la norma 802 de IEEE puede tender un puente a otros dispositivos MAC de IEEE. Ethernet, onillo con testigo e Interfaz de datos distribuidos por fibra (FDDI, Fiber Distributed Datos Interface) son ejemplos de redes que se ajuston a dicha normo paro el puenteado del nivel MAC.

El nivel de enlace de datas se fragmento en el subnivel superior de control de enlace lógico (LLC, Logical Link Contral) y el subnivel inferior del MAC. Los dispositivos que soporton lo norma 802 de IEEE tienen un subnivel MAC modular que, como se muestro en la figura 3.4., puede gobernar múltiples tipas de red, tales como Ethernet o anillo can testigo. Por tanto el subnivel superior LLC funciono como un "central" que tronsfiere paquetes de datos entre los módulos de red en el subnivel MAC. En este ejemplo, los datos se eliminan de la trama Ethernet y se reempoquetan con los formatos de la troma de anillo con testigo. Este proceso extra intraduce algún retordo, así que generalmente, se miden los puentes en función del número de paquetes que pueden procesor por segundo.

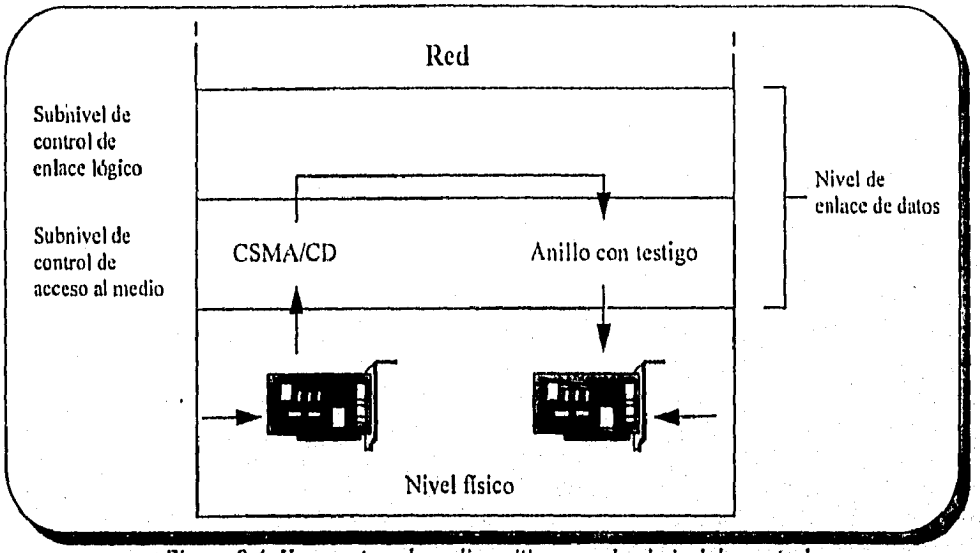


Figura 3.4. Un puente enlaza dispositivos en el subnivel de control

Los puentes proporcionan las siguientes funciones:

Reenvío de tramas. El reenvío de tramas (farwarding) constituye una forma de filtrado. Un puente enviará paquetes a otros segmentos de LAN si su dirección coincide con las direcciones de los segmentos. Esta evita que los paquetes con direcciones locales crucen un puente, sin filtrado, los paquetes se enviarían por toda la red. Cuando un paquete llega a un puente, éste lee la dirección destino contenido en él y determina si debe atravesar ese puente.

Resolución de bucle. Las grandes LANs puenteadas pueden contener bucles que causen que un paquete viaje continuamente. Algunos puentes detectan tales paquetes en los bucles y los interceptan.

Técnicas de aprendizaje. Los puentes construyen tablas de dirección que describen las rutas, bien sea mediante el exámen del flujo de los paquetes o bien con la obtención de la información de los «paquetes exploradores» que han aprendido durante sus viajes la topología de la red. El primer método se llama *puenteado transparente* y el segundo *encaminamiento fuerte*. A continuación se discuten estas técnicas de aprendizaje.

Los primeros puentes requerían que los gestores de la red introdujeran a mano las tablas de dirección. Esto era una tarea tediosa, además las tablas se debían actualizar periódicamente por si una estación de trabajo a un usuario se traslada a otro lugar. Los puentes avanzados actuales aprenden la dirección de otra estación de la red con el uso de las técnicas aquí tratadas. Nótese que frecuentemente se llama a los puentes transparentes, puentes de aprendizaje, y usan el algoritmo de expansión de árboles que es la norma 802.1 de IEEE. En el entorno Ethernet se utiliza el puenteado transparente mientras que en el de anillo con testigo se usa el encaminamiento fuerte.

PASARELA, "GATEWAY"

Un *gateway* ó *pasarela* consiste en una computadora u otro dispositivo que actúa como traductor entre dos sistemas que no utilizan los mismo protocolos de comunicaciones, formatos

de estructura de datos, lenguajes y/o arquitectura. Un Gateway no es un puente, que simplemente transfiere información entre dos sistemas sin realizar conversión. Un gateway modifica el empaquetamiento de la información a su sintaxis para acamodarse al sistema destino.

Hay que tener en cuenta que las pasarelas trabajen el nivel más alto del modelo de referencia OSI, el de aplicación. Tombién hay que considerar que debido a que los gateway realizan conversaciones de protocolos.

GATEWAY IBM SISTEMAS ANFITRIONES (Hosts).

Un gateway IBM a un sistema anfitrión conecta estociones de trabajo de una red de área local (LAN) con sistemas que en el pasado no eran capaces de reconocer sistemas inteligentes conectadas a las LANs. Con un gateway, las estaciones de trabajo de la LAN aparecen coma terminales 3270 de IBM ante anfitrión. Un teclado de PC se hace corresponder con el formato del teclado 3270. Sin embargo, normalmente es posible la conmutación entre sesión 3270 y el PC mediante la pulsación de ALT-ESC u otra secuencia de teclas. Otras funcianes de gateway más sofisticadas permiten a las PCs conectarse a ésta para transferir archivos hacia o desde el anfitrión o para ejecutar aplicaciones cliente/servidar que permiten a las PCs acceder a servicios de base de datos posteriores en dicho anfitrión. La conexión de red avanzada par a par (APPN, Advanced Peer-to-Peer Networking) de IBM proporciona servicios de red entre pares en el entorno IBM, de modo que los sistemas anfitriones de IBM se convierten sencillamente en parte integrante de la red.

GATEWAY DEC

En los sistemas DEC se utilizan los gateway para los sistemas anfitriones de IBM. Los productos de PATHWORKS de Digital permiten el acceso de una cierta variedad de computadoras personales a los sistemas VMS (Virtual Memory System) de DEC.

GATEWAY LAN

Un gateway LAN proparciona un trayecto para los datos que fluyen de una LAN a otro a través de una LAN intermedia que actúa como modo de interconexión. Esta LAN intermedia utiliza normalmente un protocolo diferente, de modo que hay que realizar una conversión de datos en el transporte. Un gateway puede realizar estos servicios, por ejemplo, muchos de los gateway proporcionan conexión Ethernet y FDDI (Fiber Distributed Interface). Los paquetes que fluyen de la red Ethernet a la FDDI pueden traducirse (mediante una función de gateway) y distribuirse hacia un nodo de la red FDDI, o bien encaminarse hacia otra red FDDI. Esta última opción constituye una forma de encapsulación, y la red FDDI actúa como una red saporte para las redes Ethernet. Existen además gateways entre protocolos Apple Talk y TCP/IP, IPX y TCP/IP.

GATEWAY INTERNET

En el mundo Internet, lo que habitualmente se conocía camo gateway ahora se denomina router, y los gateway actuales hacen referencia a sistemas de conversión de protocolos, como por ejemplo retransmisores de correo electránico. El dispositivo Internet ahora denominado router enlaza generalmente dos redes o sirve de punto de enlace entre una red interna y otra externa. Este dispositivo captura los paquetes procedentes de los sistemas anfitriones, examino sus direcciones y encamina los paquetes hacia otro router o sistema anfitrión, mientras que un gateway hace referencia a los dispositivos que realizan una traducción entre una terminal virtual OSI y el programa telnet en Internet.

ENCAMINADORES "ROUTERS"

Los routers son conmutadores de paquetes (o retransmisores a nivel de red) que operan al nivel de red del modelo de protocolo de Interconexión de Sistemas Abiertos (OSI, Open Systems Interconnection). Los routers interconectan redes en áreas locales como extensas, y cuando existe más de una ruta entre dos puntos finales de la red, proporcionan control del tráfico y filtrado de funcianes, como se muestra en la figura 3.5. Los routers son críticos en las redes interconectadas grandes y de área extensa que usan enlaces de telecomunicación. Dirigen los paquetes a través de las rutas más eficientes o económicas dentro de la malla de redes, que tiene caminos redundantes a un destino.

COMO TRABAJAN LOS ROUTERS

Un routers examina la información de dirección de los paquetes y los envía hacia su destino a través de una ruta predeterminada. Los routers mantienen tablas de los routers adyacentes y de las redes de área local (LAN's, Local Area Network) que hay dentro de la red. Cuando un router recibe un paquete, consulta dichas tablas para ver si puede enviarlo directamente a su destino. En caso contrario, determina la posición de otro rauter que pueda hacerlo avanzar hacia su destino

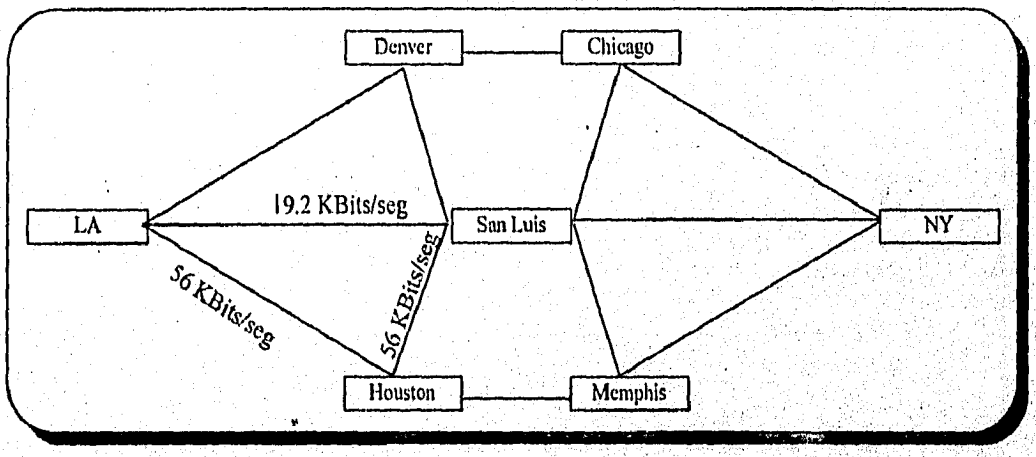


Figura 3.5. Los routers dirigen los paquetes a través de una malla de la red.

El proceso de avance requiere la realización de un procesamiento. Cuando el router ha recibido, la totalidad de un paquete, consulta la información de dirección HS y a continuación lo reenvía . Como consecuencia, el rendimiento se verá influido por las diferencias en los componentes del router y en la arquitectura. Algunos sistemas operativos de red como Novell Netware, soportan el encaminamiento en el servidor. Esto se logra mediante la instalación de dos o más tarjetas de red. Sin embargo, las tareas de encaminamiento las puede realizar el servidor. En ese caso, es necesario los rauters externos, para que liberen ol servidor de dichas tareas y se ocupen sólo de los archivos.

Los routers trabajan bien con un protocolo único camo el protacolo de control de transmisión/Protocala Internet (TCP/IP, transmisión Control Protocal/Internet Protocal), o bien can múltiples protocolos como intercambio secuencial de paquetes/Intercambio de paquetes entre redes (SPX/IPX, Sequenced Paqued Exchange/Internetwork Packet Exchange) y TCP/IP. Recuérdese que no se soportan todas los protocolos y que algunos de ellos no pueden ser encaminados. Sin embargo, estos últimos pueden transmitiese a través de las redes interconectadas usando técnicas de encapsulación.

Los routers permiten segmentar una red en otros para que se direccionen por separado. Los segmentos son más fáciles de gestionar, cada segmento LAN tiene su propio número de LAN específico y, cada estación de trobojo del segmento tiene su propia dirección. Esta es la información que empaquetan las protocolos del nivel de red.

EL PROCESAMIENTO DE PAQUETES REALIZADO POR LOS ROUTERS

Los routers manipulan paquetes que tienen la misma dirección de red. Cuando un router recibe un paquete, comienza un procedimiento que lo desempaqueta y determina dónde se debe enviar. El contenido de un paquete y la información añadida por cada uno de los protocolos del nivel de red se tratan en el epígrafe "PAQUETES". A continuación se dan los procedimientos que sigue el router, cuando trabaja con un paquete:

1.- Se, comprueba si el paquete tiene algún error con el uso del valor del código de paridad contenido en el paquete.

2.- Se descarta la parte de la información del paquete que le añadieron los protocolos de nivel físico y de enlace de datos, tal como se muestra en la figura 3.6.

3.- Se evalúa la información que añadieron, en la computadora fuente, los protocolos de la red.

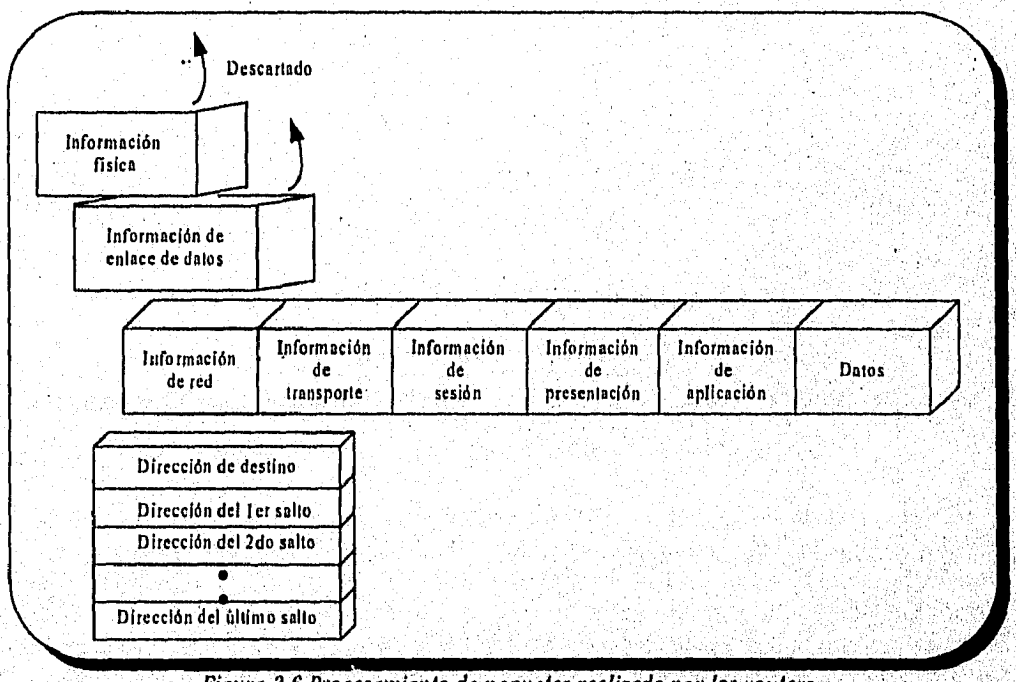


Figura 3.6 Procesamiento de paquetes realizado por los routers

La información de los protocolos del nivel de red contiene la dirección de destino, y en el caso de las redes que, al igual que TCP/IP, tengan un encominamiento fuente también contienen uno listo de *"saltos"* que definen la *"ruta optima"*, previamente determinada, para cruzar la red. El router podría hacer una de las siguientes cosas:

- El paquete podría estar dirigido al propio router así que el router evalúa cual es la información remanente en el paquete.

- Si un paquete tiene destino en la propia red, el router simplemente lo envía.
- Si la lista de filtros está disponible, el router compara la dirección del paquete con los valores de la lista y lo descarta si es necesario. Esto hace que un paquete quede dentro o fuera de la red, en base a razones de seguridad.
- Si el paquete contiene información procedente del router fuente en la que se contenga el nombre del próximo router que está en la ruta hacia su destino simplemente dirige el paquete hacia él.
- Un router mantiene una tabla de rutas que pueden emplear los paquetes para cruzar la inter-red.
- Si un router no conoce una ruta o no puede encontrar la dirección de destino del paquete en su tabla de caminos descarta el paquete y podría devolver un mensaje de error en la fuente.
- Algunos paquetes (del tipo de los TCP/IP) contienen información acerca del número de saltos que han hecho en la red, si un paquete sobrepasa un cierto número, el router lo descarta ya que asume que está en un bucle. El router podría devolver un mensaje de error en la fuente.

LAS ESPECIFICACIONES DE LOS ROUTERS

Normalmente, cuando una red es pequeña o está en un único edificio, se hace útil el uso de puentes. Estos pueden facilitar el tráfico entre los distintos segmentos de una red local muy ocupada. Nótese a la derecha de la figura 3.7., que se han conectado varias subredes a través de puentes, y que el conjunto de los mismas está conectado al soporte FDDI mediante un router.

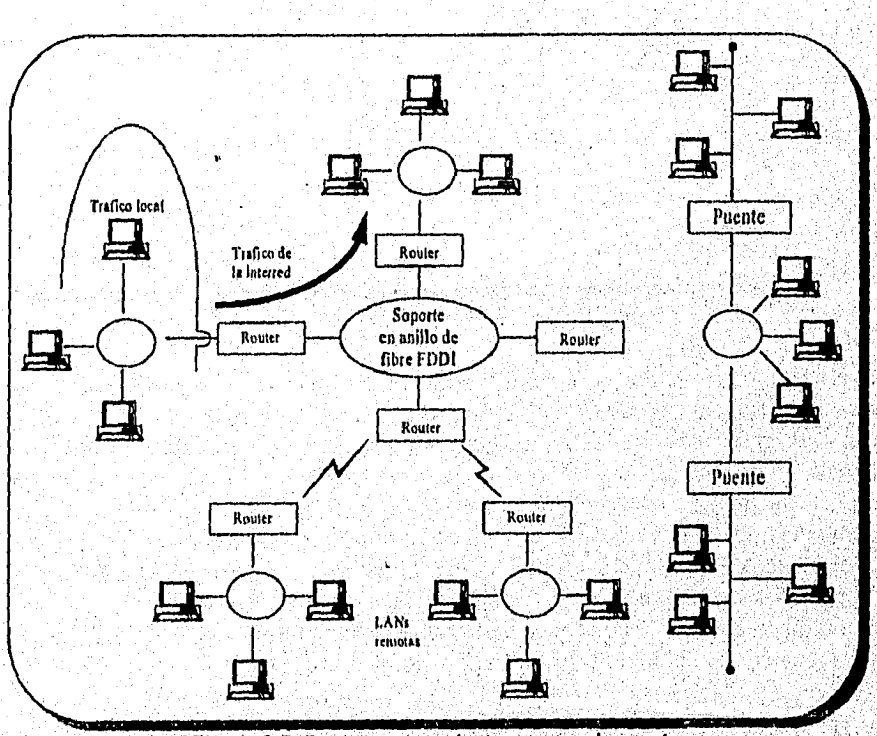


Figura 3.7. Router perteneciente a una red soporte.

Cuando se evalúan y se compran routers hay que cerciorarse de que todos los de la inter-red usan los mismos métodos para el encaminamiento y trabajan con los mismos protocolos. Algunos routers utilizan técnicas de compresión de datas para el incremento del rendimiento de los paquetes, para evitar problemos se debe intentar usar siempre el mismo router en todos los puntos.

3.3 RPC "REMOTE PROCEDURE CALL " Y PEER-TO-PEER

Un RPC es un procedimiento que se ejecuta en otra computadora diferente a la que hizo la invocación del procedimiento, es decir, el cliente no ejecuta el procedimiento, sólo lo invoca en el servidor. Cuando un proceso ejecuta un RPC y se suspende los parámetras de la llamada son enviados a donde se encuentra el procedimiento remoto el proceso es ejecutado ahí, cuando el procedimiento remota se completa, sus resultodos son enviados de regreso a través de la red y el proceso llamado reanuda el procesamiento como si estuviera regresando una llamada a un procedimiento local. Esto se ilustra en la figura 3.8.

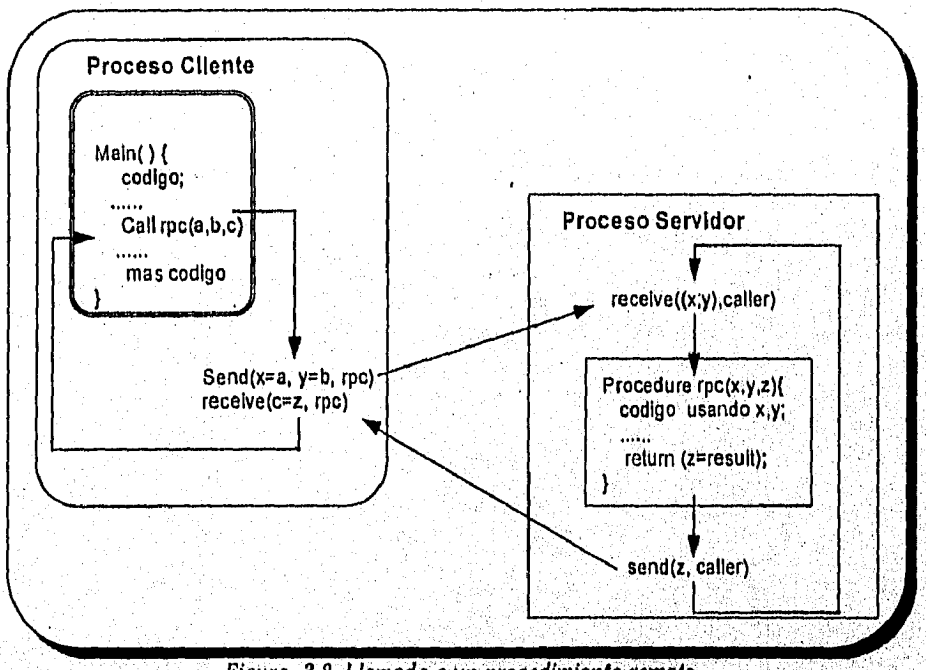


Figura. 3.8 Llamada a un procedimiento remoto

Se hace notar que al ejecutar un procedimiento remota tiene la misma semántica que ejecutar el mismo procedimiento localmente. Para realizar ésto, el ambiente de la llamada (pila de procesos y variables globales) debe ser transferida al procesador destino, a menos que haya restricciones en el tipo de procedimiento llamado, por ejemplo, cuando un procedimiento remoto pasa un puntero a una estructura de datos grande o tiene referencia a una variable global, a menos que el ambiente del proceso de llamada este disponible para el procedimiento remoto, el puntero o la referencia a la variable global estando fuera de uso. En la práctica, La mayoría de RPCs restringen el uso de variables globales, para evitar la necesidad enviar el contexto del proceso completo a través de la red. En lugar de enviar un puntero, la mayoría de RPCs no referenciará el puntero y enviará el objeto al cual apunta.

Los RPCs son inherentemente sincronizados, ya que raramente tiene sentido llamar un procedimiento sin resultados de regreso. El par emisor/receptor que es ejecutado por el cliente

debe bloquear las operacianes para osegurar que el resultado este siempre regresondo de el RPC. Esto es fácilmente lograda ya que el emisor y el receptor están fuera del ámbito de el pragrama de aplicación. Las programadores de aplicociones simplemente enlazan su programa a la librería del RPC en el nodo que llama y su procedimiento llamado a la librería del RPC en el nodo destino.

Esto es mostrado en mayor detalle en la figura 3.9. La librería del RPC sobre el equipo cliente proporciona un módulo de código para el procedimiento remato. Este módulo de código contiene un par emisor/receptor y es enlazado al código de la aplicación para formar el proceso del cliente. La librería del RPC en el servidar praparciana una infraestructura para el pracedimiento remoto. Esta infraestructura contiene un par emisor/receptar con el código necesario para invocar al pracedimiento remota. Cuando se enlaza con el código de la aplicación constituye el proceso del servidar. Para el programador de aplicaciones es casi tan trivial como llamar a procedimientos locales, ya que el protocola es transparente.

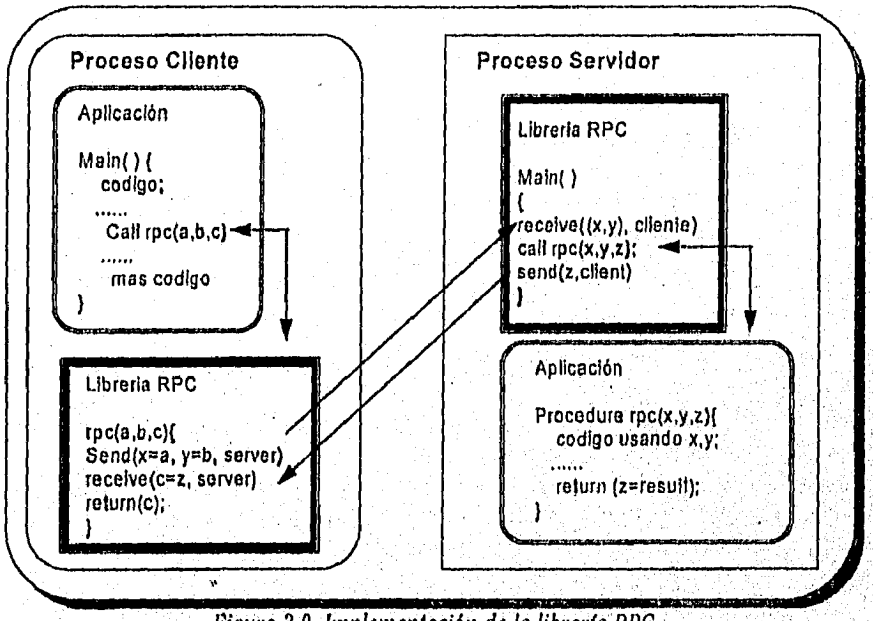


Figura 3.9. Implementación de la librería RPC

PRINCIPALES COMPONENTES DE UN RPC

Una buena ilustración de un RPC es el seleccionado por la Fundación de Software Abierto (OSF, Open Software Faundatlan) basado en el RPC de Hewlett-Packard.

Librería de tiempo de corrida (run-time library) implementa el protocolo de RPC. A estas funcianes corresponde en gran parte las facilidades de intercambio de datos de extremo a extremo en estructuras previas, la librería de tiempo de corrida del RPC tiene un conjunto de interfaces (run-time API) para usarse por el módulo de código para la llamada, este módulo es utilizada para buscar información en un llamado al servidor o un directario de sistema, establecer conexión a el servidar, enviar llamadas de solicitudes y respuestas, etc. Una responsabilidad clave de la librería de tiempo de carrida es la de mantener el control de las condicianes de estado en nombre del cliente y servidor durante una conexión. Finalmente, la librería del tiempo de corrida construye los encabezados apropiados para las paquetes del RPC, como es requerida por el suministrador del servicia de transporte e invoca al suministrador del servicio de transporte el nombre del cliente y el servidor.

El Lenguaje para Definición de Interface en la Red (Network Interface Defition Lenguage NIDL), para definir la interface de RPC y producir el módula de código para la llamada, es un lenguaje estrictamente declarativo, contiene sólo construcciones para definir constantes, tipos y operaciones de una interface; no tiene construcciones ejecutables (el NIDL esta considerado para ser una función de la capa 7 de OSI). El NIDL fuente esta compuesta con declaracianes de un lenguaje de alto nivel, aumentado con definiciones de la interface e indicaciones para el uso de cada parámetra. Otra información en el NIDL donde es hecha la canversión de datas, tipo de parámetras ordenados para ser usadas, soparte para la verificación del servidor y si la librería de tiempo de carrida sapartará protocolos de recuperación de dos fases.

El compilador NIDL (algunas veces conocido como el procesadar NIDL o generador del módula de código para la llamada), checa la sintaxis, traduce las definiciones NIDL a declaraciones en lenguajes como C y genera el módulo de código para la llamada para usarse en el cliente y en el servidar.

El módulo de código para la llamado, (Stub)en acuerdo con la librería del tiempo de carrida del RPC, conecta los programas de llamada al ambiente de tiempo de carrida (fig. 3.10). EL módulo de código para la llamada del cliente primero emitirá un comando RpcBindTolnterface, incluyendo la sintaxis de transferencia propuesta y el esquema de transparte, para intentar la vinculación. La información retornada puede indicar una opción una sintaxis de transferencia y/o direcciones de transporte para ser usadas. El módulo de código para la llamada buscará seleccionar una de cada una y continuará con el actual proceso de llamada. Un comando RpcSend manda la transmisión de una solicitud de llamada a un servidor. Un comando RpcReceive obtendrá las resultados del servidor. Un comanda RpcUbind terminara la vinculación.

La Representación de Datos en Red (Network Data Representation NDR) es un protocolo que define como los valores estructurados que son pasados en el RPC están codificados para la transmisión de la red. El propósito de NDR es habilitar máquinas con diferente representación local de datos para comunicar un tipo de valores de una a otra. El protocolo NDR especifica un conjunto de tipo de datos que pueden ser usados para especificar un conjunto de tipo de valores, además define como representar formatos escalares. NDR representa una secuencia de valores con un formato de etiqueta más una cadena de bits.

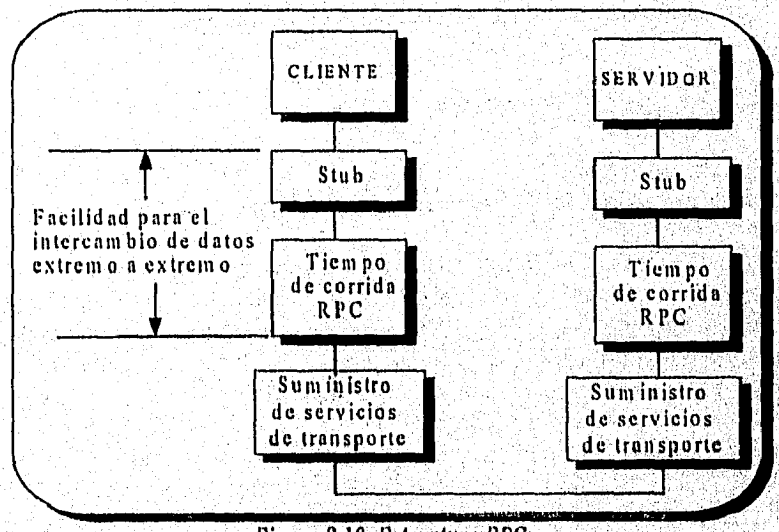


Figura 3.10. Estructura RPC

MODELOS DE INTERACCION

La mayoría de los protocolos de red requieren que la librería del RPC en el servidor esté corriendo antes de que el primer RPC use sus servicios. Protocolos más avanzados son capaces de iniciar un proceso con el servidor aunque la librería no esté ya ejecutándose, esto es hecho considerando el nombre del servidor en un archivo de configuración basado en el tipo de mensaje recibido. Si la librería del servidor no está corriendo el archivo de configuración indica como arrancar. Una vez arrancado este bloque en el servidor con una recepción, espera solicitudes del cliente, cuando una solicitud llega, el servidor responde, llamando el procedimiento de la aplicación, que regresa sus resultados y cierra el ciclo con un receive para la próxima solicitud.

Otro solución es suministrar un procesador maestro que contiene la operación receive para invoca al procedimiento remoto como un procesador esclavo. El maestro está entonces inmediatamente disponible para servir a la próxima llamada RPC. Esto es ilustrado en la figura 3.11.

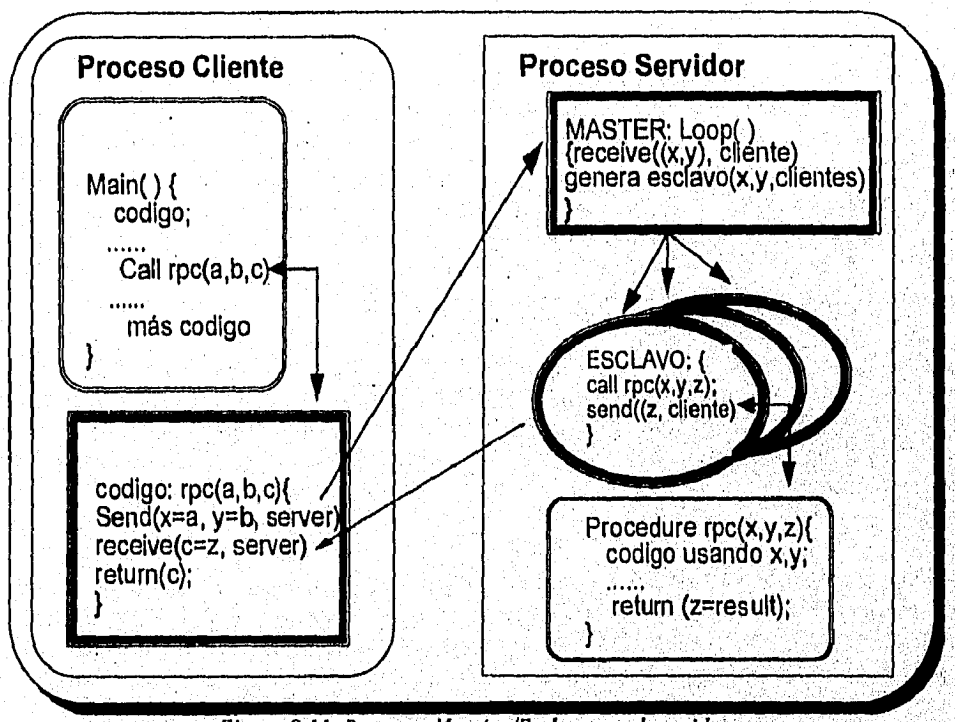


Figura 3.11. Procesos Maestro/Esclavo en el servidor

Hasta aquí se ha descrito un estricto modelo RPC en el cual el servidor esta ocioso de que entren solicitudes de los clientes. Con el modelo de reunión (rendezvous) ambas partes del proceso se ejecutan independientemente pero acuerdan reunirse periódicamente para intercambiar mensajes. En una reunión cliente/servidor las solicitudes del cliente se ejecutan en un punto de entrada específico del servidor. Cuando el servidor esta listo para ejecutar este punto de entrada, emite un comando accept (punto-entrada,mensaje), el comando accept es lo mismo que el receive (cualquier mensaje) excepto que el primero sólo acepta mensajes para un punto de entrada específica.

Otra desviación del modelo estricto RPC involucra el uso de llamadas devueltas (callbacks). Usando llamadas devueltas, el cliente proporciona el nombre de un procedimiento que el

servidor puede llamar si ocurre una excepción durante el procesamienta del RPC. Las llamadas devueltas son frecuentemente usadas para dar notificacianes al cliente de eventas significantes ocurriendo en el servidor.

TIPOS DE FALLAS

Hay cuatro tipos de fallas para un RPC:

- Caída del cliente después de llamar un procedimiento remoto.
- El mensaje llamando al pracedimiento remoto puede perderse.
- Caída del servidor mientras ejecuta el procedimiento remoto.
- El mensaje regresando los resultados puede perderse.

Si el cliente se cae después de llamar un procedimiento remoto, el servidor hará el trabajo infructuosamente. El mensaje al regresar las resultados deberá ser descartado par el subsista de comunicaciones del cliente, porque seria un proceso inexistente. Aun si el cliente reinicia, el protocolo debe distinguir entre un proceso actual y un proceso previo y descartar el mensaje anterior.

Si los mensajes llamando o regresanda desde el procedimiento remoto están perdidos, el cliente puede potencialmente esperar para siempre. La mayoría de protocolos RPC dependen de reconocimientos (ACK's) para asegurar que estos mensajes estén correctamente recibidos. Ya que el servidor no tiene forma de saber si su ACK fue recibido, debe estar atento y ejecutar la primera llamada, por lo tanto, el servidar debe tener algún registro de llamadas previas para no procesar la misma llamada dos veces. La semántica de un RPC es generalmente implementada agregando un número secuencial a cada solicitud, si una llamada es reenviada, su número secuencial deberá ser el misma que antes.

Si el cliente no recibe el ACK como respuesta del mensaje, el servidor puede esperar cierto tiempo y usar su memoria de llamadas procesadas para reenviar la respuesta al cliente, si el ACK de respuesta al mensaje esta perdido, el cliente recibirá dos respuestas a la misma llamada y deberá saber como descartar la segunda. La figura 3.12. ilustra como un protocolo de tres paquetes (solicitud, respuesta y reconocimiento) puede ser usado para procesamiento de fallas.

Si el servidor cae, sus clientes esperarán cierta tiempo y reenviaran sus llamadas, si el servidor está fuera por mucha tiempa, las intentos repetidos de estos clientes pueden causar una congestión en la red. Después de algún número de reintentos, los clientes deben desistir dependiendo si las llamadas devueltas o el usuario saben cuando reiniciará el servidor.

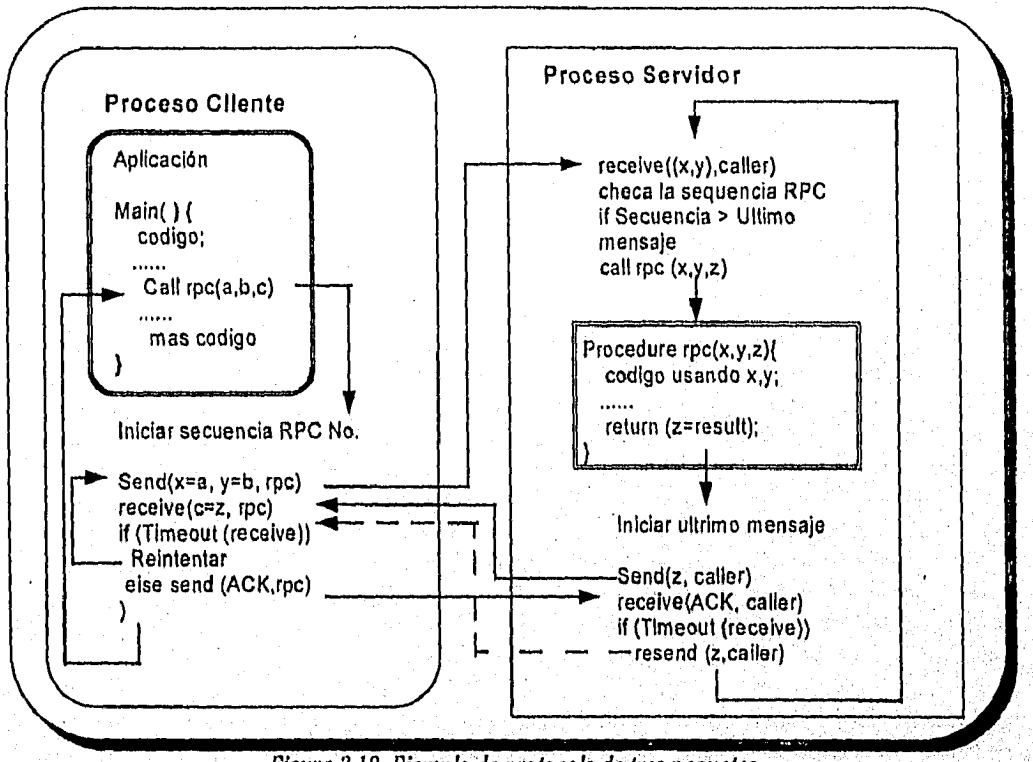


Figura 3.12. Ejemplo de protocolo de tres paquetes

En la práctica, un servicio de RPC orientado a conexión garantiza una entrega confiable de mensajes en el orden en que fueron enviados. Un servicio de RPC no orientado a conexión es usualmente construido en base a un servicio de transporte con datagramas y no garantiza que todos los mensajes enviados sean recibidos.

PEER-TO-PEER

Una sesión de comunicación par a par es aquella donde cada estación tiene un control similar sobre la sesión. Las estaciones de trabajo tienen su propia potencia de procesamiento y pueden controlar sus propias actividades. Esto contrasta con los viejos modelos de computadora central (mainframe) y minicomputadora, en los cuales una computadora central proporcionaba toda la potencia de procesamiento para muchos terminales diferentes, éstos simplemente mostraban pantallas de información y proporcionaban un lugar para que los usuarios introdujeran órdenes mediante un teclado, incluso cuando se añadieron las computadoras inteligentes a estos sistemas el control permanecía todavía en su mayor parte en manos de la computadora central.

Cuando se instalaron computadoras con su propia potencia de procesamiento dentro de las organizaciones se hizo evidente que había muchas ventajas para la descentralización de las computadoras centrales y la proporción de procesamiento cooperativo en el cual un usuario en una computadora podría acceder y usar los recursos de otras computadoras así como sus archivos o impresoras.

IBM modificó su orquitectura de sistemas en red (SNA. Systems Network Architecture) para dar soporte a un tipo de conexión de red par a par, a principios de los 80, pero el anfitrión (host)

todavía establecía v mantenía las sesianes. Actualmente, IBM adapta una disposición en la cual las estaciones de trabajo pueden establecer sus propias sesiones par a par. Las estrategias de conexión de red avanzada par a par (APPN, Advanced Peer-to-Peer Networking) proporciona entornos de informática cooperativa mediante la cual las computadoras grandes y pequeñas se pueden comunicar unas con otras coma entidades pares sobre Redes de área local (LAN, Local Area Networks) y Redes de área extensa (WANS, Wide Area Networks). Los usuarios de estaciones de trabajo o de sistemas finales pueden iniciar una comunicación en la red con otros sistemas finales sin ninguna ayuda del anfitrión. APPN cambia el papel de los sistemas anfitriones. Muchos se han convertido en servidores pasteriores (back-end) que proporcionan servicios par a par a las usuarios de la red.

La conexión de red par a par y el modelo cliente/servidor están relacionados estrechamente. En el modelo cliente/servidor las estaciones de trabajo actúan como clientes y acceden a recursos de otras computadaras, que actúan camo servidores. En una disposición par a par, cualquier computadora de la red puede hacer que sus recursos estén disponibles para otros usuarios, sin embargo. dentro del entorno NetWare se limita esta relación cliente/servidor en algún grado, las estaciones de trabajo acceden a recursos en servidores centralizados que disponen de entornos controlados estrictamente. En un entorno típico de NetWare los estaciones de trabajo son clientes/servidores y no pares con otras estaciones de trabajo, aunque existe disponible un software de terceros para permitir a los clientes dialogar unos con otros, aunque en un entorno estricto de NetWare, las estaciones de trabajo acceden a los archivas localmente o desde un servidor.

Aunque existen muchas ventajas en el enfoque de conexión de red par a par, muchos gestores de red se resisten a su uso porque proporciona poca gestión y control de seguridad. En una disposición par a par, se almacenan los datos en muchos lugares de la red, con la creación de problemas administrativos y en el seguimiento de la localización de los datos, de las copias de seguridad y verificaciones de seguridad. Los sistemas de servidor de archivo dedicodo, como NetWare de Novell, proporcionan un almacén de datos centralizada, seguridad avanzada que incluye autentificación y autorización a múltiples servidores y, en algunos casos, funcionamiento mejarado al utilizar los superservidores.

A pesar de estas inconvenientes, la conexión de red entre pares encuentra su camino dentro de las organizaciones a través de los usuarios de grupos de trabajo o de usuarios que adquieren paquetes como Windows for Workgroups, que incluye características de conexión de red entre pares. Los siguientes sistemas operativos de red proporcionan funcionalidad par a par:

- Windows para trabajo en grupos de Microsoft.
- Personal NetWare de Novell
- LANtastic de Artisoft.
- LAN Server y OS/2 de IBM

3.4 TCP/IP, NETBIOS Y NETBEUI

En un sistema cliente/servidor un protocolo permitirá fundamentalmente iniciar, mantener y terminar un diálogo entre elementos del sistema, asimismo, un protocolo regulará la forma en que deberán generarse e interpretarse los elementos orientados al control de errores y la forma de recuperar la información perdida, igualmente estarán previstas en un protocolo la forma de identificar el camino que se utiliza para el intercambio de la información y la identificación del tipo de mensaje. Los elementos del diálogo de un protocolo serán los mensajes, otros elementas importantes de un protocalo son:

- *Sintaxis*: Incluye formato y codificación de datos, nivel de señales.
- *Semántica*: Incluye la información de control necesaria para la coordinación y detección de errores.
- *Temporización*: Incluye temporización y secuencialización de eventos.

TCP/IP

A mediadas de los 70 el Departamento de Defensa (DOD) de las Estados Unidos decidió estandarizar las comunicaciones entre sus computadoras para facilitar la conexión entre equipos de diferentes marcas, modelos y de diferentes proveedores, con este propósito pone en marcha un proyecto orientado al desarrollo de un conjunto de protocolos estándar que más tarde se conocerían como los protocolos TCP/IP, la figura 3.15, muestra la estructura de este conjunto de protocolos o protocolos estándar del Departamento de Defensa de los Estados Unidos, los protocolos desarrollados son:

MIM-STD	1777	Internet Protcol (IP)
MIM-STD	1778	Transmision Control Protocol (TCP)
MIL-STD	1780	File Transfer Portocol (FTP)
MIM-STD	1781	Simple Mail Transfer Protocol (SMTP)
MIL-STD	1782	TELNET Protocol

Muy pronto este conjunto de protocolos tuvo mucha popularidad gracias a la aceptación de algunos de los principales centros científicos, ocodémicos y estatales de los Estados Unidos:

- El proyecto DARPA decide usar TCP/IP para la red Internet.
- La U. de California implementa las protocolos TCP/IP bajo Unix en su versión Unix 4.2BSD.
- La Fundación Nacional de Ciencias (National Science Foundation) de los Estados Unidos decide usar TCP/IP.

La función que realiza cada uno de los niveles, figura 3.15, es la siguiente:

Nivel de proceso FFP,SMTP,TELNET	(Process layer)
Nivel Nodo-a-Nodo TCP	(Host-to-Host layer)
Nivel de Inter-Red IP	(Internet layer)
Nivel acceso a red	(Network access layer)

Figura 3.13. Estructura de los niveles de los protocolos TCP/IP.

Nivel de Acceso a Red (Network Access Layer) Intercambio de datos entre la computadora y la red; este nivel es el encargado de direccionar físicamente los nodos de la red.

Nivel de Inter-Red (Internet Layer) en caso de que las computadoras que establecen comunicación residan en redes diferentes, este nivel se encarga de la transmisión de datos entre las dos redes (ruteo); no establece conexión lógica, envía datagramas, no garantiza recepción libre de errores.

Nivel Nodo-a-Nodo (Host-to-Host Layer) establece conexión lógica (activa o pasiva) entre procesos remotos a través de sockets, garantiza recepción libre de errores.

Nivel de Aplicación (Application Layer) ofrece servicios al usuario (sesiones remotas, transferencia de archivos y correo).

Actualmente, los protocolos TCP/IP han sido implementados en diferentes tipos de computadoras que van desde las PC's, hasta la supercomputadora Cray X/MPx. Las grandes ventajas de TCP/IP son: popularidad, eficiencia y confiabilidad. Las principal desventaja de este protocolo es su falta de compatibilidad con el modelo OSI, como lo ilustra Figura la 3.14.

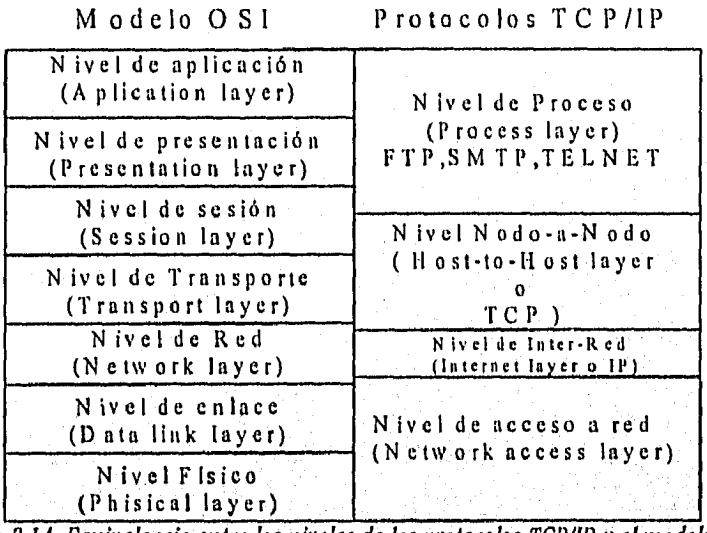


Figura 3.14. Equivalencia entre los niveles de los protocolos TCP/IP y el modelo OSI.

COMUNICACION A TRAVES DE TCP/IP

La transmisión de un mensaje en TCP/IP, como se oprecia en la figura 3.15., es similar o la transmisión de un mensaje en el modelo OSI: conexión entre niveles iguales usando los servicios que ofrece el nivel inmedioto inferior, excepto en el nivel de enlace en donde la conexión es físico y directa (sin niveles intermedios).

Así, si el usuario necesita un envío libre de errores, debe pasar por TCP, pero si considera que no necesita de los servicios de TCP puede ignorarlo y dirigirse a IP por sus servicios. La figura 3.16., muestra el esquemo de una comunicación entre dos nodos usando TCP/IP: si un proceso deseo comunicarse con otro, debe reservar un puerto en su nodo, los puertos son únicos dentro de los nodos y ayudan al protocolo TCP a identificar al proceso que lo reservó; como cada nodo tiene una dirección única dentro de Internet, el proceso quedo identificado dentro de la red Internet o través del nodo y el puerto reservado, o la combinación dirección de noda, número de puerto, se le conoce como socket

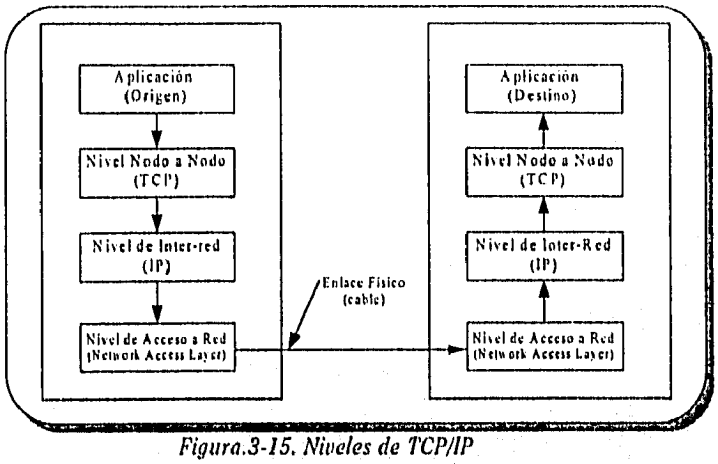


Figura.3-15. Niveles de TCP/IP

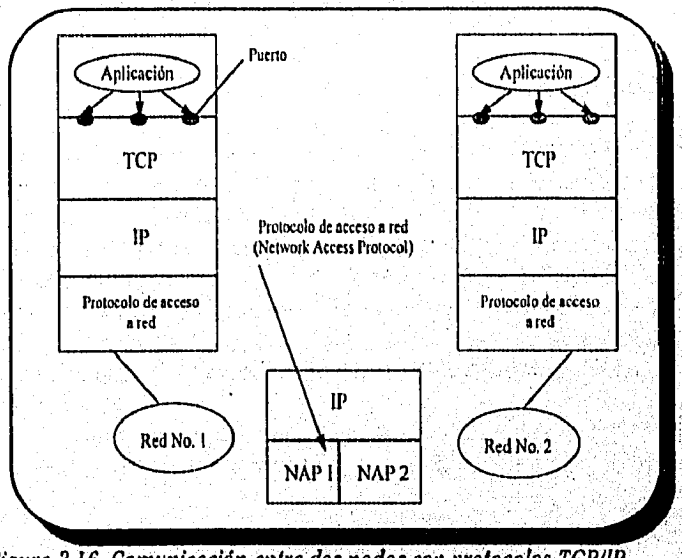


Figura 3.16. Comunicación entre dos nodos con protocolos TCP/IP.

EL PROTOCOLO DATAGRAM

Los cinco protocolos mencionados anteriormente son la base del conjunto de protocolos TCP/IP, pero además de esos existen otros que complementan el grupo, el Protocolo Datagram (User Datagram Protocol, UDP) es uno de ellos. Como se ve en la figura 3.17., este protocolo se encuentra al mismo nivel que TCP y nos ofrece una transmisión sin garantía de recepción, se limita a enviar/recibir mensajes sin esperar ajuste de recepción y sin cuidar que los mensajes lleguen en el orden enviado.

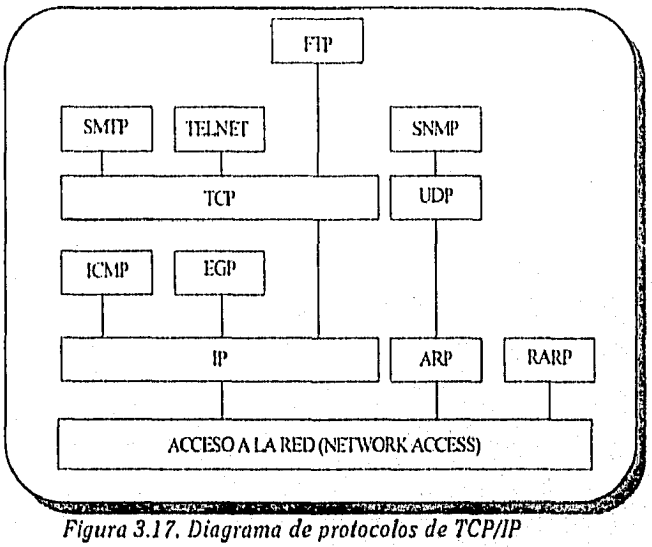


Figura 3.17. Diagrama de protocolos de TCP/IP

SEGMENTACION Y REENSAMBLADO

El objetivo de un protocolo de comunicaciones es el intercambio de flujo de datos entre dos entidades, usualmente, la transferencia puede ser caracterizada como una secuencio de bloques de datos de tamaño limitado. En el nivel de procesos, podemos referimos a uno unidad lógica de transferencia de datos como un mensoje, ahora, ya sea que lo aplicación (por ejemplo TELNET,FTP,SMTP) envíe los datos vía mensajes o en un flujo continuo, los protocolos de nivel inferiores requieren romper los datos en bloques de menor tamaño, esta función es llomada *segmentación*. Por conveniencia, nos referiremos a un bloque de datos intercambiados por dos entidades vía un protocolo como PDU (Protocal Data Unit).

La contraparte de la segmentación es el reensamblado, eventualmente los datos segmentados deben ser reensamblados en mensajes apropiados a la oplicación.

ENCAPSULADO

Cada PDU contiene no sólo datos sino también contienen información de control. En algunos casos, los PDU's consisten solo de información de control. La adición de la información de control a los datos es referida como encapsulado, la información es recibida o generado por una entidad y encapsulada en un PDU conteniendo más información de control.

CONTROL DE CONEXION

Una entidad puede transmitir datos a otra entidad de una manera no planeada, sin coordinación previa, ésto se conoce como transmisión de datos sin-conexión (connectionless data transfer). Aunque este modo puede ser útil, es menos común que la transferencia de datos orientada-a-conexión (connection-oriented data transfer). La transferencia de datos orientada es preferida (hasta requerida) si la estación anticipa la longitud del intercambio de datos y/o ciertos detalles del protocolo deben ser enviados fuera dinámicamente, tres fases ocurren:

1. Establecimiento de la conexión
2. Transferencia de datos
3. Terminación de la conexión

El protocolo internet es del tipo sin-conexión; esto es deseable para proveer la capacidad de interconexión de redes. TCP es orientado-a-conexión para soportar confiablemente la transferencia de datos. Como FTP,SMTP y TELNET hacen uso de TCP también presentan la característica de orientados a conexión.

DIRECCIONAMIENTO

La arquitectura de comunicaciones TCP/IP intenta soportar un ambiente en el cual existan múltiples redes, múltiples nodos en cada red, y múltiples procesos en cada nodo. Esto requiere un esquema de direccionamiento complejo. Las direcciones de los nodos TCP/IP consisten de 32 bits, tradicionalmente divididos en cuatro partes de 8 bits donde cada uno representa un número entre el 1 y el 255, separados por un punto, existen tres clases de direccionamiento:

Clase A	Considera siete bits del primer byte para definir la dirección de la red con un rango que va del 1.0.0.0 al 127.0.0.0, los tres bytes restantes son utilizados para definir la dirección de los nodos con un rango que va del 1 al 255. Esta clase es utilizada en empresas con pocas redes, pero con muchos nodos en cada sus redes. Con la clase A se pueden tener 128 redes con hasta 16,777,216 nodos cada una.
Clase B	Considera seis bits del primer byte y ocho bits del segundo para definir la dirección de la red con un rango que va del 128. 1.0.0 al 19 1.254.0.0, los dos bytes restantes son utilizados para definir la dirección de los nodos. Esta clase es utilizada por la mayoría de las empresas debido a que es capaz de soportar una gran cantidad de redes y nodos. Con la clase B se pueden tener hasta 16,384 redes con 65,536 nodos cada una.
Clase C	Considera cinco bits del primer byte y 16 bits del segundo y tercer byte para definir la dirección de la red con un rango que va del 192. l. 1 0.0 al 254.254.254.0, el último byte es utilizado para la dirección de los nodos. Esta clase es utilizada en empresas que requieren de una gran cantidad de redes pequeña hasta 2,097,152 con hasta 256 nodos en cada una.

La siguiente tabla muestra el esquema de direccionamiento TCP/IP

Clase	Modo de Direccionamiento	Núm. Redes en la Clase	Nodos Direccionados
A	0nnnnnnn hhhhhhhh hhhhhhhh hhhhhhhh	128	16,777,216
B	10nnnnnn nnnnnnnn hhhhhhhh hhhhhhhh	16,384	65,536
C	110nnnnn nnnnnnnn nnnnnnnn hhhhhhhh	2' 097,152	256

Tabla 3.3 Esquema de direccionamiento TCP/IP

n: bit de dirección de red 0: fijo
h: bit de dirección ID del nodo 1: fijo

Los nombres y direcciones de los elementos o nodos de una red son muy importantes en un ambiente de cómputo cooperativo y distribuido que soporte la arquitectura cliente/servidor. Las direcciones de los nodos deben ser globalmente únicos en toda la red.

La clase es seleccionada en el diseño de la red y no puede ser cambiada arbitrariamente.

La definición de la dirección de la red se vuelve importante cuando la red que se diseña va a integrarse con otras redes que estén inscritos en Internet. En ese caso se deberá solicitar la dirección de la red a Network Information Center (NIC).

Los estándares TCP/IP están definidos en las Peticiones de Comentarios (RFC), publicadas en el Grupo de Ingeniería de Internet (IETF) y otros grupos de trabajo, en la siguiente tabla 3.4. se describe las RFC más importantes.

RFC	DESCRIPCION
768	Protocolo de datagramas de usuario (UDP)
783	Protocolo de transferencia de archivos trivial (TFTP)
791	Protocolo Internet (IP)
792	Protocolo de mensajes de control de Internet (ICMP)
793	Protocolo de control de transmisión (TCP)
826	Protocolo de resolución de direcciones (ARP)
854	Protocolo Telnet (TELNET)
862	Protocolo Eco (ECHO)
863	Protocolo Descartar (DISCARD)
864	Protocolo Generador de caracteres (CHARGEN)
865	Protocolo Cita del día (QUOTE)
867	Protocolo Día (DAYTIME)
894	IP sobre Ethernet
919,922	Datagramas de difusión IP (difusión con subredes)
959	Protocolo de transferencia de archivos (FFP)
1001,1002	Protocolos de servicio NetBIOS
1034,1035	Sistema de nombres de dominio (DOMAIN)
1042	IP sobre Token Ring
1055	Transmisión de IP sobre líneas serie (IP-SLIP)
1112	Protocolo multitransmisión de pasarela Internet (IGMP)
1122,1123	Requisitos de host (comunicaciones y aplicaciones)
1134	Protocolo punto a punto (PPP)
1144	Compresión de cabeceras TCP/IP para enlaces serie de baja velocidad
1157	Protocolo de administración de red simple (SNMP)
1179	Protocolo daemon de impresora de líneas
1188	IP sobre FDDI
1191,	Detector de ruta NM
1201	IP sobre ARCNET
1231	MIB Token Ring IEEE 802.5 (MIB-H)
1332	Protocolo de control de protocolo Internet PPP (IPCP)
1334	Protocolos de autentificación PPP
1533	Opciones de DHCP y extensiones de fabricantes BOOTP
1534	Interoperación entre DHCP y BOOTP
1541	Protocolo de configuración dinámica de host (DHCP)
1542	Clarificaciones y extensiones del protocolo de inicio
1547	Requisitos para el Protocolo punto a punto (PPP)
1548	Protocolo punto a punto (PPP)
1549	PPP en tramas de Control de vínculo de datos de alto nivel (HDLC)
1552	Protocolo de control de intercambio de paquetes de inter-red PPP (IPXCP)
1553	Compresión de cabecera IPX
1570	Extensiones del Protocolo de control de vínculo (LCP)
RFC en borrador	Protocolo de control de trama NetBIOS (NBFCP), PPP sobre ISDN, PPP sobre X.25, Protocolo de control de compresión.

Tabla 3.4 RFC de TCP/IP

Todas las rfc se encuentran en internet a través de *ds.internic.net*.

NETBIOS/NETBEUI

IBM y Microsoft diseñaron los protocolos del Sistema básico de entrada salida en red (NetBIOS, Network Basic Input Output System) y de la Interfaz extendida de usuario de NetBIOS (NetBEUI, NetBIOS Extended User Interface) para dar soporte a los comunicaciones en entornos de red de área local (LAN, Local Area Network) de pequeño y medio tamaño. En la figuro 3.18. se representa el entorno complejo del protocolo.

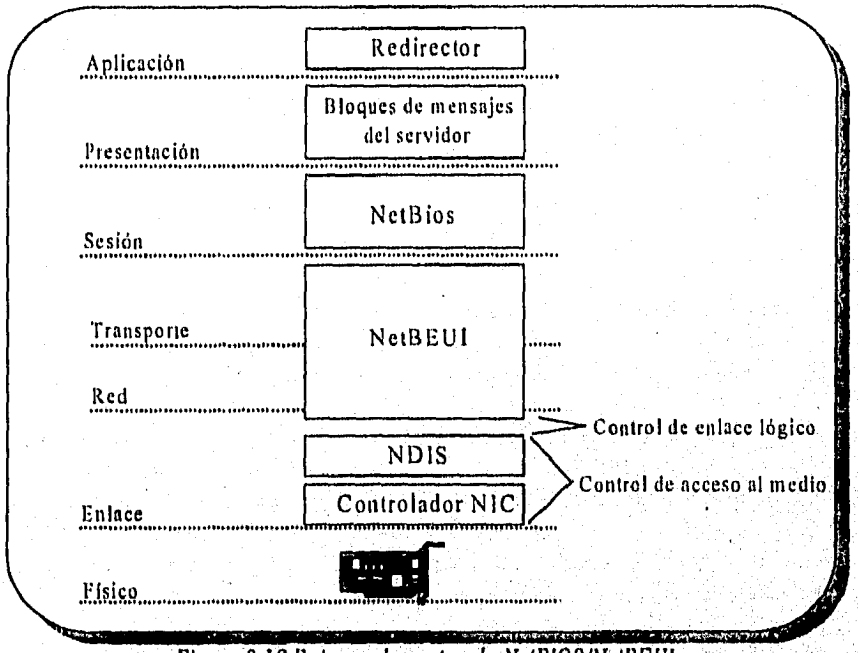


Figura 3.18 Entorno de protocolo NetBIOS/NetBEUI.

Redireccionador. Dirige las peticiones de red a los servidores de la misma y las órdenes locales al sistema operativo local.

Bloques de mensajes del servidor. Proporciona el lenguaje par a par y los formatos necesarios para que las computadoras se comuniquen unas con otras.

NetBIOS. Un protocolo del nivel de sesión relacionado con el modelo de protocolos OSI. Establece y mantiene las sesiones de comunicación entre computadoras.

NetBEUI. Proporciona los servicios subyacentes para el transporte de los datos.

Especificación de la interfaz del controlador de red (NDIS, Network Driver, Interface Specification). Una especificación de Microsoft desarrollada recientemente que proporciona una forma de dar soporte a otros protocolos, como TCP/IP, con una única interfaz de red.

NetBIOS y NetBEUI se implementon en diversos sistemas operativos de equipos de escritorio y de red, incluidos OS/2, Windows para trabajo en grupo, Windows NT, LAN Manager de Microsoft y LAN Server de IBM. Obsérvese, sin embargo, que Microsoft también da soporte a la pila de protocolos del Protocolo de control de transmisión/Protocolo Internet TCP/IP, en todos sus productos de red actuales y futuros. Una razón para esto es que NetBIOS no es un protocolo encaminable y no es adecuado en un entorno de red de área extensa.

NETBEUI

IBM desarrolló NetBEUI en 1985 como un protocolo de tronsporte de red para LANs de pequeño a medio tamaño. Microsoft en sus productos de conexión de red, incluidos Windows para trabajo en grupa y Windows NT dan soporte a NetBEUI. Las redes de IBM y Microsoft pueden comunicarse debido a la implementación de NetBEUI y NetBIOS.

NetBEUI es un protocolo de los niveles de transporte y de red del modelo de protacolas OSI. Se integra con NetBIOS para ofrecer un sistema de comunicaciones eficiente en el entorno LAN de grupos de trabajo. NetBEUI proporciona los servicios de transparte de datas que NetBIOS necesita. Tómese coma ejemplo una llamada telefánica. NetBIOS es como la persona que hace la llamada y NetBEUI es cama el programa de control que trabaja can el sistema subyacente de conmutación para completar la llamada.

NETBIOS

NetBIOS se diseñó can la premisa de que las PC's en una LAN sólo necesitan comunicarse con otros PC's en la misma LAN. Es un protocolo del nivel de sesión del modelo de pratacolos OSI, que en un principio Sytek Inc, desarralló para que IBM lo utilizase en sus redes.

NetBIOS es una interfaz de programación de aplicaciones (API, Application Program Interfoce) que las programadores utilizan para crear aplicaciones LAN para los entornos LAN Server de IBM, LAN Manager de Microsoft y OS/2. Los conductos nominados san una extensión de red hacia OS/2 que proporciona características parecidas pero más avanzadas. NetBIOS y los conductos nominados en el entorno LAN san las protocalos donde se canstruyen distintas aplicaciones, NetBIOS ofrece las servicios siguientes:

NetBIOS establece nombres lógicas únicos para los nodos de la red, así simplifica la tarea de referenciar otros sistemas. El nombre tiene una longitud de 1 a 15 caracteres e identifica una estación de trabajo en concreto, tanto para las computadoras como para los usuarios. Este nombre se especifica cuando una computadora se asigna o incluye por primera vez en la red, a las grupos de nombres se les puede enviar mensajes.

NetBIOS establece una sesión orientada a la conexión sobre la cual los nodos se comunican entre ellos, la sesión tiene lugar sobre un circuito o conexión lógico. NetBIOS establece, mantiene y finaliza una sesión. Las estaciones de trabajo se comunican en tiempo real, con el envío de datos garantizado, a través de la confirmación del envío de los mensajes.

NetBIOS también puede proporcionar servicios de datagramas no orientados a la conexión en las cuales los mensajes se dirigen a otros sistemas sin el establecimiento de una conexión o la supervisión del flujo de paquetes previos.

NetBIOS transmite información sobre la ubicación de servidores y los nombres de estos servidores, esta transmisión puede sobrecargar la red con exceso de paquetes y causar problemas en las inter-redes, sin embargo, el filtrado en puentes (bridges) y encaminadores (routers) puede resolver estos problemas.

NetBIOS no es ruteable, en parte debido a su sistema de nombres de 15 caracteres. NetBIOS se debe encapsular (empaquetar) en las paquetes de otras protocolos para su distribución sobre las inter-redes. El protocolo TCP/ IP consta de un procedimiento para el encapsulamiento de NetBIOS.

NetBIOS está bien arraigado y todavía se le da soporte en muchos entornos, incluso está disponible para nuevos protocolos como los conductos nominados. El funcionamiento de NetBIOS es fácil de entender y muchas aplicaciones la utilizan, por lo que se seguirá utilizando durante algún tiempo.

3.5 DCE

La Fundación de software abierta (OSF, Open Software Foundation) es una organización que desempeña un papel clave en el desarrollo de las normas para productos interoperables. El Entorno de informática distribuida (DCE, Distributed Computing Environment) de OSF es un conjunto de software "de permisos" que oculta las diferencias entre los productos, las tecnologías y los estándares de múltiples vendedores que permite a los programadores la creación de aplicaciones para los entornos cliente/servidor distribuidos. DCE proporciona la independencia de los sistemas operativos y de las redes con un entorno abierto de desarrollo.

Un entorno distribuido de corporación extensa consta normalmente de computadoras, sistemas operativos y aplicaciones de múltiples vendedores, todas enlazadas a una plataforma de red común. La figura 3.19 ilustra un entorno heterogéneo, el truco es hacer que todas los recursos disponibles sean transparentes a los usuarios, tales redes pueden tener un enorme alcance y estar sujetas a variaciones en las velocidades de acceso de datos, a menudo dictadas por las conexiones de redes de área extensa. Pueden surgir la sincronización de datos y otros problemas, eso complica la tarea de desarrollo de aplicaciones distribuidas, además, los usuarios deberían poder acceder a la base de datos de cualquier computadora con la aplicación frontal (front-end) que ellos elijan. Los problemas de interoperatividad se deberían manejar en segundo plano y ocultar a los usuarios, una forma de conseguir esto es construir una infraestructura genérica. Después los fabricantes parten de una plataforma común sobre la cual pueden construir productos que interoperen con los de otros fabricantes. El objetivo de OSF con DCE es crear esta infraestructura genérica.

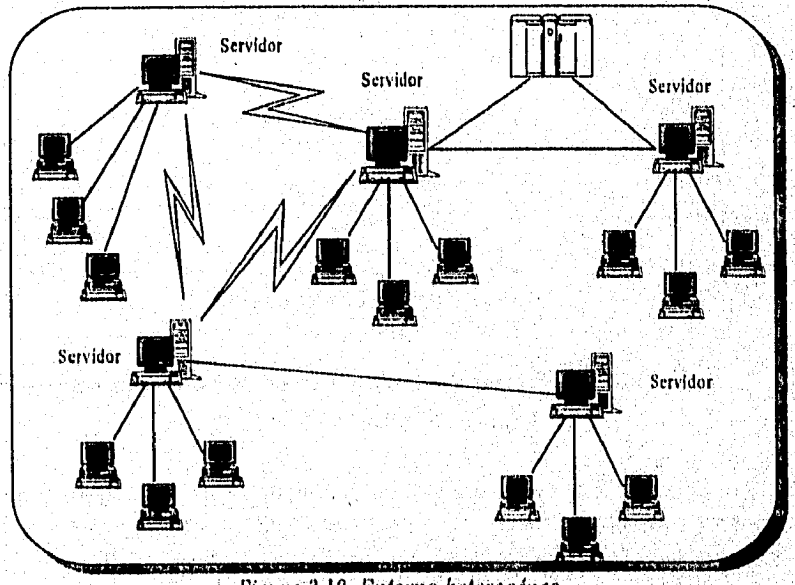


Figura 3.19. Entorno heterogéneo

DCE es un sistema operativo y una red independientes. En la mayoría de los casos es compatible con entornos ya en vigor, los principales vendedores como IBM, DEC, Hewlett-Packard y otros proporcionan productos basados en DCE.

DCE proporciona las bases de la interoperatividad para Aplicaciones de entornos genéricos (CAE, Common Applications Environment) de X/Open, varias especificaciones internacionales para los sistemas abiertos. El movimiento hacia los sistemas abiertos guía a DCE, mediante la integración y la interoperatividad de diversos sistemas y productos de redes de múltiples vendedores.

OFS obtuvo su tecnalagía a través de un proceso llamado petición de tecnología (Request for Tecnology). Los principales vendedores de hardware y software abandonaron sus técnicas y tomaron lo mejor de OSF, algunas ejemplas son :

- Hewlett-Packard y la llamada a procedimiento remota (RPC) de DEC.
- Servicios de nombra en el directoria X.500 de Software AG.
- Servicias de seguridad Kerberos de MIT con extensión de Hewlett-Packard.
- Sistema de archivos Andrew de Transarc.
- Tecnalagía integrada LM/X PC Micrasoft.
- Arquitectura coordinada de múltiples hilos DEC.

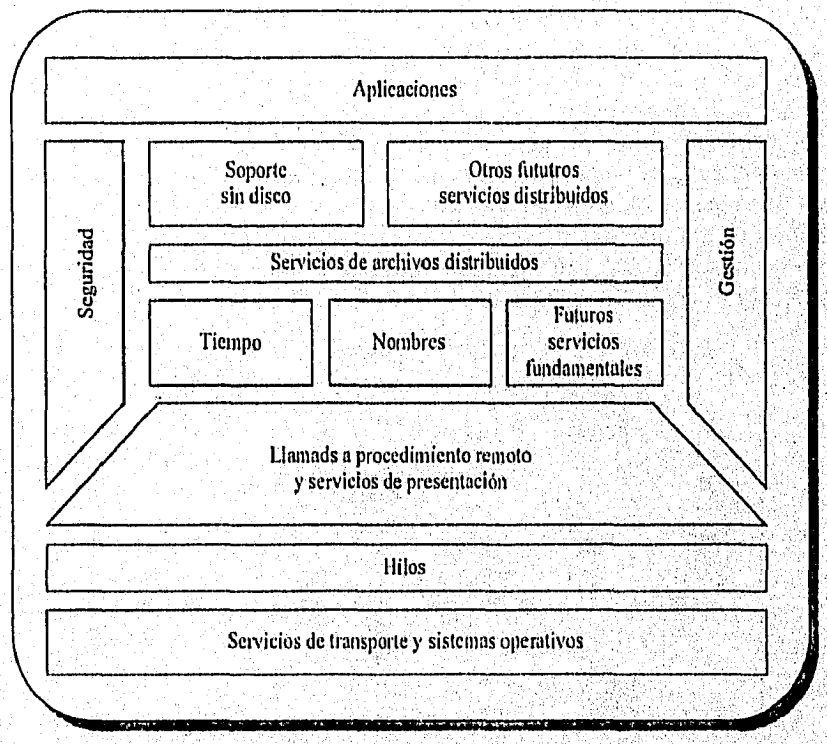


Figura 3.20. Arquitectura DCE.

La figura 3.20. representa la arquitectura DCE, es un madelo de niveles que integra varias de las tecnalagías que se describen en el resto de esta sección. Debajo están los servicios básicos (como los sistemas operativas) y arriba las aplicacianes. Los servicias ofrecidos por DCE se diseñan para enmascarar la complejidad de las entornos de red de múltiples vendedores y permitir que la información fluya fácilmente a donde se necesite.

COMPONENTES DE DCE

Los servicios DCE se agrupan en dos categorías: herramientas de desarrollo y servicios de comparación de datos. Las herramientas de desarrollo ayudan a los programadores de software en la creación de servicios de usuario final que la informática distribuida necesita. Entre ellos, se incluyen los siguientes:

- Llamadas a procedimientos remotos.
- Servicios de directorio.
- Servicios de tiempo.
- Servicios de seguridad.
- Servicios de hilos.

Los servicios de comparación de datos proporcionan a los usuarios finales capacidades para añadir herramienta de desarrollo para que accedan fácilmente a la informoción, y son los siguientes:

- Sistemas de archivos distribuidos.
- Soporte sin disco.

DCE incluye el modelo OSI en el nivel 5 (sesión), nivel 6 (presentación) y el nivel 7 (aplicación). El modelo de Interconexión de Sistemos Abiertos (OSI, Open System Interconnect) de ISO define los protocolos normalizados para la interoperatividad de múltiples vendedores. DCE permite que los programadores utilicen los servicios ISO normalizados al mismo tiempo que los protege de la complejidad de tales servicios.

3.6 PROCESO DE TRANSACCION

Una transacción constituye una unidad discreta de trabajo que normalmente es parte de una transacción de negocios. Un sistema de Procesamiento de transacción interactiva (OL TP, Online Transaction Processing) opera en tiempo real para recoger y procesar los datos afines a la transacción y enviar las cambios a las bases de datos compartidas y otras archivas. La mayoría de los procesos por lotes, tales como los envíos a cuentas, se ejecutan durante las horas nocturnas, los resultados de un OLTP están disponibles inmediatamente en la base de datos, suponiendo que se completen las transacciones, los ejemplos más comunes de OLTP son los sistemas de reservas de líneas áreas y las de transacciones bancarias.

Los sistemas gestores de bases de datos ejecutan transacciones con la utilización de sentencias en lenguajes, como el Lenguaje de Consulta Estructurado SQL, (Structured Query Language). La ejecución de transacciones en entornos de un único usuario y una única base de datos es simple, debido a que no existen problemas de contención o necesidad de sincronización entre base de datos. Los problemas de contención se producen cuando múltiples usuarios intentan cambiar el mismo bloque de datos simultáneamente. Además, se debe sincronizar la escritura en múltiples bases de datos y debe haber una garantía de que realmente se hizo la escritura, en todas las bases de datos. Se necesita un programa de supervisión para asegurar la integridad de la base de datos. Hay cuatro requisitos, colectivamente llamados «ACID», para el procesamiento de transacciones en entornos distribuidos.

1. Atomicidad. Define las unidades de trabajo. Si se distribuye una transacción, todas las subtransacciones que afecten a los datos en lugares separados deben ejecutarse juntas como una única transacción, bien para completarla o para volverse atrás si no se completa.

Paro mantener consistentes los datos en múltiples sitios, se utiliza el procedimiento de compromiso en dos fases.

II. Consistencia. Fundamentalmente consiste en un requisito para que las bases de datos cambien de un estado a otro en coordinación. El monitor de transacción debe verificar que todos los datos afectados son consistentes.

III. Aislamiento. Los transacciones deben ejecutarse oisladamente hasta que se completen, sin influencia de otras transacciones.

IV. Durabilidad. Esta propiedad tiene que ver con el compromiso final de una transacción, una vez que se ha verificado que la transacción es correcta en todos los sistemas afectados, se la confirma y no se puede deshacer.

PROCESAMIENTO DE TRANSACCIONES INTERACTIVAS (OLTP)

Una transacción es una unidad de trabajo discreta en un sistema de bases de datos, por ejemplo, una transacción en una base de datos es una escritura que cambia el balance de cuentas de un usuario o actualiza un artículo del inventario. El procesamiento de transacciones interactivas (OLTP, On-Line Transaction Processing) tiene lugar en tiempo real. El sistema de reservas de las líneas aéreas y los máquinas ATM de los bancos son ejemplos de sistemas de procesamiento de transacciones interactivas.

La mayoría de los sistemas de transacciones interactivas han sido tradicionalmente implementados en los sistemas de computadora central (mainframe), debido a la complejidad de las operaciones y a la necesidad de entradas/salidas, auditorías y gestión rápidas, algunos sistemas procesan de 400 a 500 transacciones por segundo o más. Si una transacción debe actualizar datos en múltiples localizaciones se necesita un mecanismo de gestión para evitar reescribir los datos y ofrecer sincronización. Otras necesidades incluyen la capacidad de retirar las transacciones que han fallado, proporcionar utilidades de seguridad y, si es necesario, la recuperación de datos. Un supervisor para el procesamiento de transacciones gestiona esto. El supervisor asegura que, o bien se completan totalmente las transacciones, o bien se retiran, de modo que la base de datos adopta el estado que tenía antes de la transacción.

En un entorno distribuido, a menudo la escritura tiene lugar en paralelo en más de un servidor de bases de datos. Tal procesamiento de transacciones concurrentes necesita un mecanismo de «vuelta atrás» que asegure la integridad de la base de datos, en caso de que un sistema falle durante una escritura. Las transacciones se comprometen a la vez o se devuelven. Una transacción se devuelve si uno o más de los sistemas involucrados en la transacción no responde con un compromiso, lo cual implica que puede haber fallado el sistema o la comunicación.

A continuación se enumeran algunos monitores genéricos para el procesamiento de transacciones (TP, Transaction Processing):

- El Sistema de Control de Información al Cliente (CICS, Customer Information Control System) de IBM es un monitor TP que se ejecuta en sistemas anfitriones (hosts) de IBM.

- Tuxedo es un monitor TP distribuido desarrollado por AT&T y comercializado por UNIX System Group de Novell. Se ejecuta en muchos sistemas de computadoras diferentes y da soporte a varios clientes, entre los que se encuentran DOS, OS/2 y Windows.

- El monitor TP Encina se basa en el software del Entorno de Informática Distribuida (DCE, Distributed Computing Environment) de la Fundación de software abierto. Los principales vendedores como IBM y Hewlett-Packard planean la utilización de Encina.

3.7 DBMS

Un DBMS (Data Base Management System, Sistema Gestor de Bases de Datos) es un programa software que normalmente opera en un servidor de bases de datos o en un sistema de computadoro central, el cual entre otras funciones gestiona los datos, acepta y responde las consultas de los usuarios, presenta las siguientes características:

Proporciona una forma de estructurar los datos como registros, tablos u objetos.

Acepta que los operadores introduzcan los datos, que almacena para su posterior recuperación.

Ofrece lenguajes de consulta para búsqueda, clasificación, infarmación y otras actividades de "Soporte de Decisiones", que ayuda a los usuarios a correlacianar y a valarar los datas recopiladas.

Permite acceso multiusuaria a los datos, junto con utilidades de seguridad que evitan que algunos usuarios vean y/o cambien cierto tipo de informacián.

Proporciona utilidades para la integridad de los datas, para evitar que más de un usuario acceda y cambie la misma información simultáneamente.

Un DBMS se sitúa en un sistema de computadara central y de terminales «tontas» acceden a él. Sin embargo, en el madelo cliente/servidar las clientes frontales (front-end) inteligentes que acceden al DBMS realizan algunos de los procesamientos y dedican el DBMS a las tareas de procesamienta de datas.

Una base de datos sencilla es una colección de registras con campos de información. Una agenda telefónica es un ejemplo de base de datos impresa, cada línea representa un registro can "campos" de información como el nombre, la dirección y el número de teléfono. Una base de datos real de computadora can esta información seria una base de datos de archivo plano, ya que toda la información podría almacenarse en un archivo.

ARQUITECTURA DE BASES DE DATOS RELACIONALES DISTRIBUIDAS (DRDA)

DRDA (Distributed Relatianal Database Architecture) es una norma de IBM para el acceso a la información de las bases de datas a través de cualquier tipo de plataformas, tanta IBM como otras. DRDA sigue las normas SQL y es un componente clave en el marco en el que trabaja el almacén de información de IBM. Se interesa en proporcionar una forma de interconexión de sistemas de bases de datos basadas en LAN can sistemas de bases de datos basadas en camputadara central de IBM como DB2, DBM, SQL/DS y SQL/400.

BASES DE DATOS DISTRIBUIDAS

En un sistema de información distribuida los datos se ubican en múltiples emplazamientos, las usuarios podrán acceder a estos datos sin tener en cuenta su localización, después de todo, a los usuarios les interesan las resultadas, no los detalles de la red de computadaras. Aquí, se enumeran las directrices generales para el desarrollo de un sistema de base de datos

distribuidas que Cris J. Date, uno de los diseñadores de los sistemas de base de datos relaciónales.

Autonomía local permite que cada zona mantenga una naturaleza independiente, de este modo las autoridades locales pueden asegurar, proteger y gestionar los datos y recursos.

No centralización, elimina zonas de datos centrales que representan un único punto de falla.

Transparencia oculta la localización de los datos a los usuarios, de modo que no necesitan preocuporse de donde están o de como conseguirlos.

Fragmentación (particionamiento) proporciona una forma de dividir la base de datos y almacenarla en multiples lugares.

Replicación posibilita la copia de múltiples fragmentos de la base de datos a múltiples emplazamientos.

Procesamiento de consultas distribuidas permite que los usuarios consulten zonas remota, utilizando el mejor camino hacia ese lugar y los mejores recursos que puedan realizar correctamente la consulta.
Procesamiento de transacciones distribuidas asegura que los datos se escriben correctamente en todas los bases de datos o se retiran si se produce un fallo en cualquier parte.

Independencia de hardware implica que se de soporte a los sistemas de computadoras y plataformas de múltiples vendedores.

Independencia de sistemas operativos ofrece soporte a diversos sistemas operativos.

Independencia de redes acepta los protocolos de comunicación y las topologías de múltiples redes.

Independencia de DBMS permite que los usuarios accedan a cualquier sistema gestor de bases de datos desde su aplicación de cliente.

Una vez que se distribuyen los datos, se instalan las medidas de procesamiento de transacciones, fragmentación y replicación para asegurar la fiabilidad, disponibilidad y protección de los datos como se describe en las secciones siguientes.

CONEXIONES CLIENTE/SERVIDOR

Los métodos siguientes se utilizan para proporcionar los conexiones entre clientes y servidores, e intercambiar peticiones y solicitudes de información. Uno de los siguientes métodos de conexión se usa para el intercambio de mensajes.

"Circuitos" orientados a la conexión se utilizan para establecer un canal de comunicación sobre una red, así los dos sistemas intercambian información en tiempo real o mantienen una conexión continua hasta que se completen las transacciones.

Servicios de datagramas no orientados a la conexión se utilizan cuando el intercambio de información no es en tiempo crítico. No se crea un "circuito". En su lugar, la información se empaqueta en datagramas y se transmiten por el camino más adecuado a su destino.

Los sistemas de almacenamiento y reenvío de mensajes han suavizado las restricciones de tiempo, una aplicación de usuario envía una solicitud a un servidor en un mensaje de correo

electrónico. Cuando el servidor reciba el mensaje, procesará la solicitud y enviará una respuesta al usuario o a su buzón. Este método supone que el servidor tiene varios procedimientos almacenados que los usuarios pueden ejecutar y son prácticos para los usuarios móviles que necesitan acceder a la información de las bases de datos de la compañía.

En un entorno heterogéneo, hacer estas conexiones no es siempre fácil. Existen diversos protocolos de comunicación, interfaces de aplicación y necesidad de componentes que hacen difícil la integración. Existen entornos y herramientas de desarrollo para la informático distribuido. El Entorno de informática distribuida (DCE), de la Fundación de Software Abierto (OSF, Open Software Foundation) es uno. Otro es el entorno de Informático abierto en red (ONC, Open Network Computing) de SunSoft.

En la conexión de bases de datos, también llamados interfaces normalizadas, existen sutiles diferencias que evitan que la aplicación cliente de un fabricante acceda a los datos de un DBMS de otro vendedor. Distintas fabricantes y grupos de estándares trabajan para solucionar el problema.

ALMACEN DE DATOS (DATA WAREHOUSE)

La razón principal de crear una base de datos es el almacenamiento de información de datos operacionales con el fin de consultarlos para toma de decisiones y ejecutar acciones instantáneas en algunos casos, pero qué pasa cuando la información que hay que consultar se mide en gigabytes e inclusive en terabytes, aunado a esto, si se necesita accesar información de varias fuentes, el objetivo perseguida tal vez no se llegue a cumplir. Actualmente se utilizan los almacenes de datos (Dato Warehouse) para acelerar y simplificar significativamente estas actividades. La depuración y herramientas de fusión de datos pueden transformar los datos operacionales e información libre de errores y formateada consistentemente, para después depositarlas en servidores especializados con la intención de que la extracción de datos revele información oculta para una rápida toma de decisiones y acciones.

Este proceso para crear un almacén de datos puede ser lento y muy laborioso ya que se debe determinar qué formato utilizar y cómo los datos deberán representarse antes de almacenarlos y que tal vez sean emigradas eventualmente, Ver figuro 3.21.

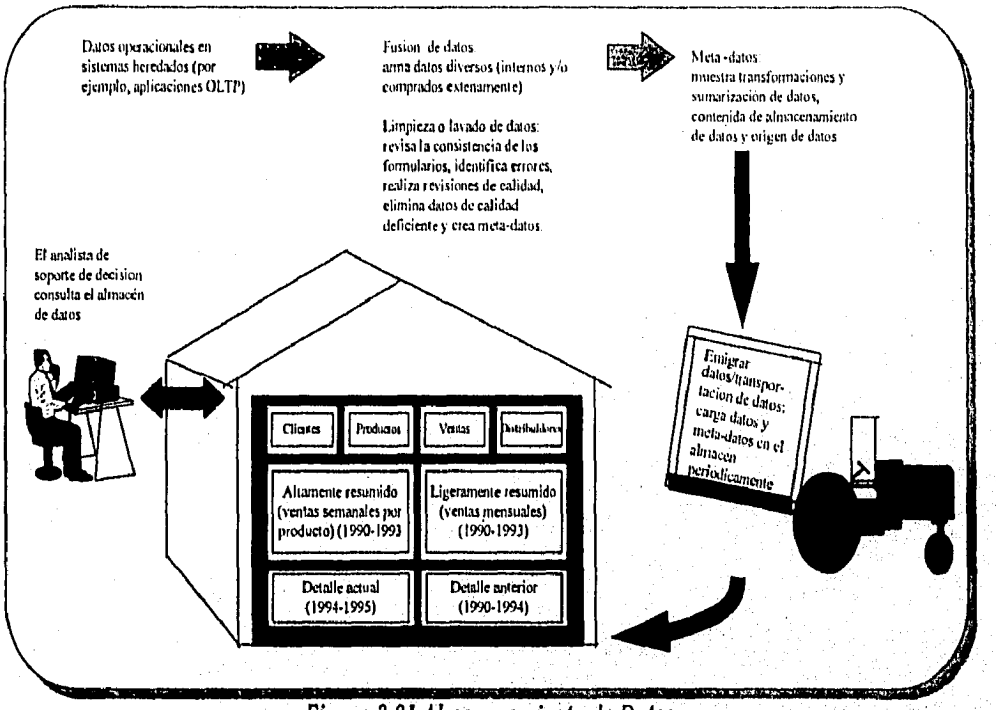


Figura 3.21 Almacenamiento de Datos

El objetivo que se persigue al crear un almacén de datos es tener un fácil acceso a los datos con tiempo de respuesta instantáneo y reducir el volumen de información almacenada con respecto a las fuentes.

Las características de un almacén de datos son:

- Orientado a temas: Los datos deben tener un límite de redundancia la cual debe estar representada en grupos lógicos fácilmente entendibles para el usuario. En otras palabras, se deben organizar los datos de forma que representen los procesos de la organización (temas), más que como procesos computacionales , ver figura 3.22.

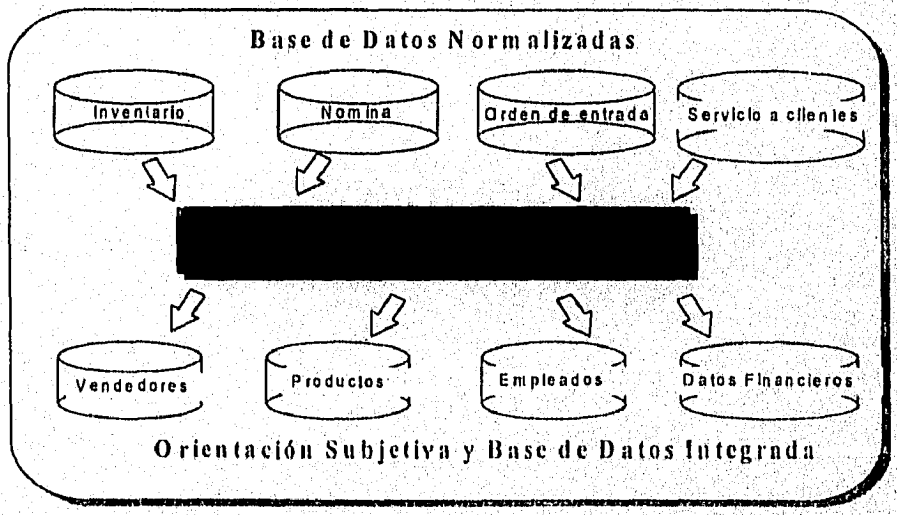


Figura 3.22. Data warehouse compuesta de orientación subjetiva y Base de Datos Integradas

- Integral: Integración de datos de varias fuentes ya sea de la organización y/o externas. Estandarización de códigos de medidas (cm, m, yarda), de formatos (Fecha juliana, europea o americana).

- Varía con el tiempo y no es volátil: Información actual e histórica, las llaves de acceso contienen la variable de tiempo. La información es almacenada periódicamente y generalmente los procesos de actualización y borrado de registros no son implementados, la información se encuentra en detalle, sumarizada y altamente sumarizada.

La clave para poblar y mantener actualizado el almacén de datos en forma automatizada es la introducción de meta datos que describen las fuentes, el destino y la relación entre éstos. Esta abstracción permite a la estructura de los datos o los procesos llegar a cambiar de acuerdo a las necesidades de la organización, sin éstos, el proceso de poblar y mantener actualizado el almacén de datos sería infinito, con ésto también se encuentra definido la forma en que el usuario buscará datos en el almacén.

Los meta datos describen el contenido del almacén de datos, donde se originaron los datos almacenados, las traslaciones, agregados, búsqueda de tablas y otras transformaciones que ocurrieron en el proceso de almacenamiento. Toda esta información se vuelve crítica después de que la migración se ha completado, cuando más detalle se necesito sobre un grupo de datos en particular o cuando los errores en los datos almacenados se vuelven evidentes.

El poder del procesamiento en paralelo que acelera el trabajo de los sistemas para el soporte de toma de decisiones tales como extracción de datos, divide una consulta complicada en partes múltiples y asigna cada parte en un procesador por separado, así cómo la tecnología de almacenamiento RAID (Asignaciones Redundantes de Discos Independientes "Redundant Arrays of Independent Disks"), están dando un auge fenomenal a la creación de almacenes de datos en la presente década y algunos expertos predicen que el 90% de todas las organizaciones en los Estados Unidos tendrán esta estrategia para finales de 1996. Ya sea en forma personal de organización o una que esté disponible a través de redes.

3.8 OODBMS

SISTEMAS ORIENTADOS A OBJETOS DISTRIBUIDOS

Los sistemas orientados a objetos ofrecen una única solución para el almacenamiento de datos y la creación de aplicaciones en entornos corporativos, los sistemas orientados a objetos presentan las características siguientes:

Los *objetos* son abstracciones de entidades del mundo real como personas en una base de datos cliente, facturas en un sistema de contabilidad, o impresoras y servidores en una base de datos de los servicios de directorios en red.

Un objeto mantiene datos e incluye un conjunto de *procedimientos* que se soliciten para manipular o investigar los datos del objeto.

Hay *clases* y *subclases* de objetos. Primero se define una clase y se utiliza como plantilla para crear objetos en esa clase, por ejemplo, un inventario para un almacén de computadoras tendría una clase llamada *computadora*.

Una subclase es la especialización de una clase en una estructura jerárquica. Una subclase denominada *computadora portátil* se debería definir baja la clase *computadora* en el inventario del almacén.

La herencia es un aspecto importante de la jerarquía de clases. Cualquier subclase creada en una clase hereda las características de ésta y puede tener alguna especial propia, la herencia facilita el desarrollo mediante el fomento de objetos reutilizables.

Los objetos interactúan entre ellas con el envío de mensajes que solicitan los procedimientos del objetos.

Los objetos son polimorfos en el sentido de que diferentes objetos podrían invocar de forma indistinta un mensaje. Por ejemplo, al ejecutar la orden de imprimir un objeto cliente se imprimirá su nombre y dirección, en cambio para un objeto factura se imprimirá la factura.

La información de un objeto se encapsula y sólo se puede cambiar con la salicitud de los procedimientos que pertenecen a ese objeto. Una entidad externa no puede evitar estos procedimientos, ni cambiar los datas internos. Esto crea un entorno altamente controlado que facilita el mantenimienta y la construcción de aplicaciones.

Debido a que los objetos mantienen los datos en las entradas de tipo campo, se podría comparar un objeto con un registro de base de datos, pero aquí es dande termina el parecido. Los objetos tienen sus propios procedimientos internos para trabajar con las datos que contienen, mientras que los procedimientos externos gestionan cualquier manipulación de una base de datos relacional. Esto da al objeto una cierta independencia. Si un objeto se traslada, los procedimientos necesitan extraer su información para moverla con él.

APIS DE CONECTIVIDAD EN BASES DE DATOS Y MIDDLEWARE

Una organización que quiera crear una red corporativa a partir de sus redes departamentales, se encontrará normalmente can diversos sistemas de bases de datos autónomos estructurados sobre modelos diferentes. Estas sistemas de bases de datos pueden ser relaciones arientados a objetos o jerárquicos. Pueden ser de diferentes vendedores y tener interfaces incompatibles. Esta situación se produce porque en el pasado, los distintos departamentos construyeron sus propias sistemas gestores de bases de datos. Puesta que los usuarios dentro de estos departamentos tenían las apropiadas aplicaciones frontales (front-end) para acceder a las bases de datos, no existían problemas en el acceso a los datos. Sin embargo, las diferencias en los protocolos, procedimientos, interfaces y plataformas hacen que las datos corporativos sean más difíciles de obtener y manipular.

La interfaz entre el cliente y el servidor consta de un lenguaje de acceso a los datos que normalmente es un *Lenguaje de consulta estructurado SQL, una interfaz de programación de aplicaciones (API, Application Program Interface)* para que interactúen cliente y servidor, que permite intercambios SQL *formatos y protocolos (FAP. y Formats and Protocols)*, que definen los procedimientos y formatos de comunicación entre cliente y servidar para el envío de mensajes a través de una red. En la actualidad, SQL no es un estándar, aunque existen una serie de normas en desarrollo. Normalmente los vendedores han desorrollada el lenguaje para adaptarla sus propias necesidades.

El problema radica en conseguir que muchas aplicaciones cliente puedan camunicarse con el servidor y éstos su vez puedan comunicarse entre ellos. Una solución sería esperar que quienes desarrollan aplicaciones frontales incluyan interfaces para cualquier servidar posterior (back-

end). Otra que quienes desarrollan aplicaciones posteriores incluyan interfaces para cualquier aplicación frontal. Ambas soluciones están lejas de hacer realidad a corto plazo.

La Organización internacional de normalización ISO y una organización llamada Grupo de acceso a SQL (SAG, SQL Access Gruap) trabajan en la creación de normas que permitirán la interoperatividad entre los sistemos frontales y los sistemas posteriares. El objetiva de SAG es promaver y desarrollar especificaciones para API's de SQL y FAP's que se basen en la especificación del Accesa a base de datos remotas (RDA, Remote Database Access) de ISO, que define cómo los clientes basados en SQL acceden a los servidores de base de datas remotas. El grupo ha desarrollado tres especificaciones técnicas :

- *Lenguaje de consulta estructurada.* Una especificación para realizar el lenguaje SQL que cumple las especificaciones internacionales.

- *Accesso a base de datos remotas de SQL. Esta especificación define las comunicaciones* entre un cliente basodo en SQL y un servidor de base de datos remota.

- *Interfaz del nivel de llamada. Un conjunto de APIs que interactúan con productos basados* en SQL.

La Interfaz del nivel de llamada de acceso a SQL es la base de la norma para conectividad abiertas en bases de datos (ODBC, Open Database Conectivity) de Micrasoft y para la interfaz de programación de aplicaciones para bases de datos independientes (IDAPI, Independent Database Application Program Interface), especificación desarrollada por Borland International, IBM y Novell. Tanto ODBC como IDAPI afrecen una interfaz genérica entre clientes y servidores de las bases de datos multiplataforma de múltiples vendedores. La Interfaz tiene controladares que interceptaran las peticiones de los clientes y las traducen en declaraciones o accesos adecuados para el servidor al que necesita acceder el cliente. Por esta razón, ODBC e IDAPI se consideran productos middleware.

ODBC DE MICROSOFT.

La mayoría de los sistemas de bases de datos posteriores realizan Las funciones comunes proporcionadas por ODBC. Por ese motivo, se escriben las aplicaciones frontales para ODBC y luego se aprovechan las ventajas de estas funciones. Este planteamiento puede que no tenga la utilidad de las caracterísiticas especiales disponibles en todo sistema posterior, pero es un paso hacia la interoperatividad que permitirá a las usuarios beneficiarse del accesa a los datos de sistemas posteriores antes inaccesibles. Microsoft distribuye ODBC coma un conjunto de contraladores que proporcionan acceso desde aplicaciones como Access y Excel de Microsoft, FoxPro, Btreve, DBASE y Paradox a IBM, Oracle, Paradox y otros sistemas de bases de datos posteriores. Los controladores están disponibles para ser distribuidos a los vendedores de software y para ser incluidos en las aplicaciones. El propósito de ODBC, desde el punto de vista de Microsoft, es hacer de Windaws la plataforma cliente más popular.

IDAPI

En un principia, Borland desarrolla IDAPI y después otros vendedores dieron el soporte a la norma. IDAPI es similar a ODBC en su funcionalidad y también se diseñó alrededor de la Interfaz del nivel de llamada. Según Borland, IDAPI soporta más servidores que ODBC.

3.9 RED DE DATOS DE LA D.G.I.

Un sistema de cableado estructurado[1], se realizá en la Subsecretaría de Ingresos; está diseñada para datar un sistema de cableado integrada y transparente para las necesidades de comunicacianes. El cable de par trenzado sin blindar y la fibra óptica comparten el misma media de transmisián "canaleta " y el atra es reservado para el eléctrico "par las características físicas del edificia".

Para una mayar referencia listaremas las características de cableada:

CONCEPTO	DESCRIPCION
Cable	UTP Nivel 5
Fibra áptica	4,8 y 12 Hilas
Jumper de fibra óptica	2 Hilos
Jumper de cable	RJ-45

Las concentradares "Hub 's" utilizadas son estándar Ethernet y sus características son:

- Madelos 5DN002, 5DN003, 5DN308P, 5DN378P-F y 2814-05 de marca Bay Netwarks.
- Puerta Interconexión 10BASE-FL.
- Soparte Unshield Twisted Pair "UTP" .
- Protocolos de red IEEE 802.3 CSMS/CD 10 Mb/s.

El protocolo de comunicación estándar es TCP/IP, par lo que se definieron los siguientes estándares: .

DOMINIO	:	HICH
NOMBRE DE SERVIDOR	:	HICHDGIGINB1
MASCARA	:	255.255.252.0
DIRECCION IP "D.G.I"	:	99.96.176. ???
DIR. IP ROUTERS	:	99.96.176.1 hasta 99.96.176.50
DIR. IP IMPRESORAS RED	:	99.96.176.51 hasta 99.96.176.99
DIR. IP SERVIDORES	:	99.96.176.100 hasta 99.96.176.120
DIR. IP CLIENTES	:	99.96.176.121 hasta 99.96.176.254

El TCP/IP utilizada es de Microsoft versión de 32 bits, en las estaciones de trabaja y Windows NT para impresoras (algunas impresoras ocupan el protacolo DLC). Hasta el mamento se cuenta con 5 servidores y 138 estaciones de trabajo.

La conexián de la red de fase I con fase II, se realizó a través de un bridge como se muestra en la figuro 3.23.

1 Cableado estructurado : Sistema de cableado preplanificado que está pensado para hacer frente a la reconfiguraciones y el crecimiento. Conforma una infraestructura para las partes criticas de la red, incluye cables, conectores de comunicación, enchufes, conectores, adaptadores, balum, sistema de paneles de parches y componentes electrónicos. Proporciona un medio para transmisión de datos, video, voz y otros tipos de información.

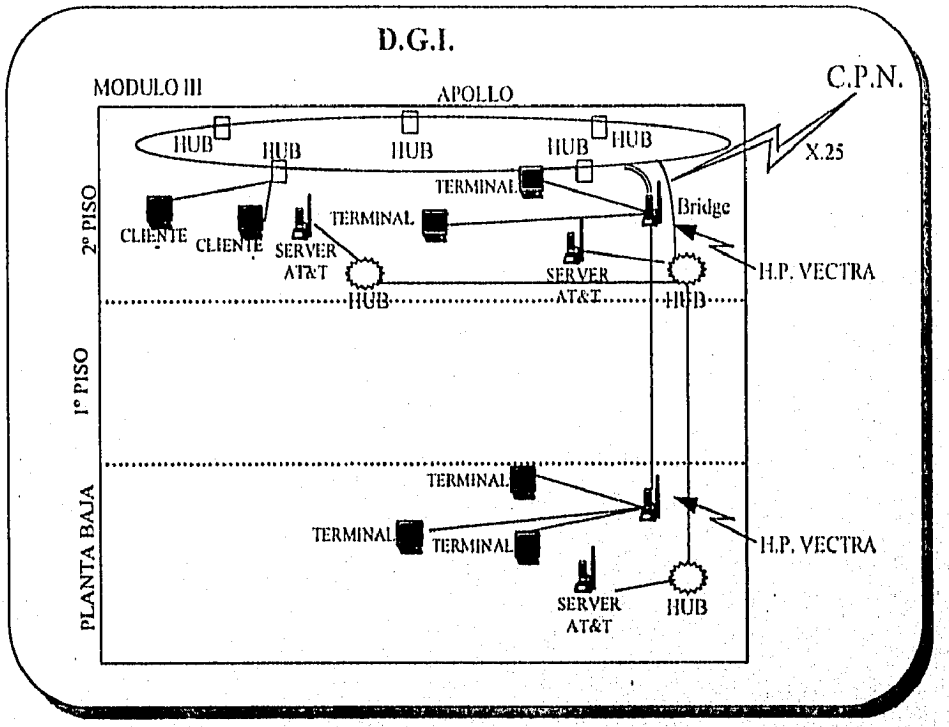


Figura 3.23 .- Arquitectura cliente/servidor en la D.G.I.

El correo electrónico interno es Mail Server 3.2, y se está evaluando Microsoft Exchange como el correo electrónico institucional. Este ha reducido el uso de papel en el envío de documentos internos con bastante confidencialidad.

Se esta definiendo el modo de transferencia de datos en la red interna, y se piensa en ATM (Asyncronous Transfer Mode) que es una tecnología de banda ancha para transmitir voz, vídeo y datos sobre LAN's ó WAN's. Es una tecnología de retransmisión de celdas (paquetes de datos fijos), que al contrario de Frame Relay donde los paquetes tienen diferente tamaño, esto hace posible una predicción y garantía en el ancho de banda en las aplicaciones que la necesitan. Lo más relevante es el dispositivo de conmutación, donde muchos nodos pueden trasmitir simultáneamente, ver figura 3.24.

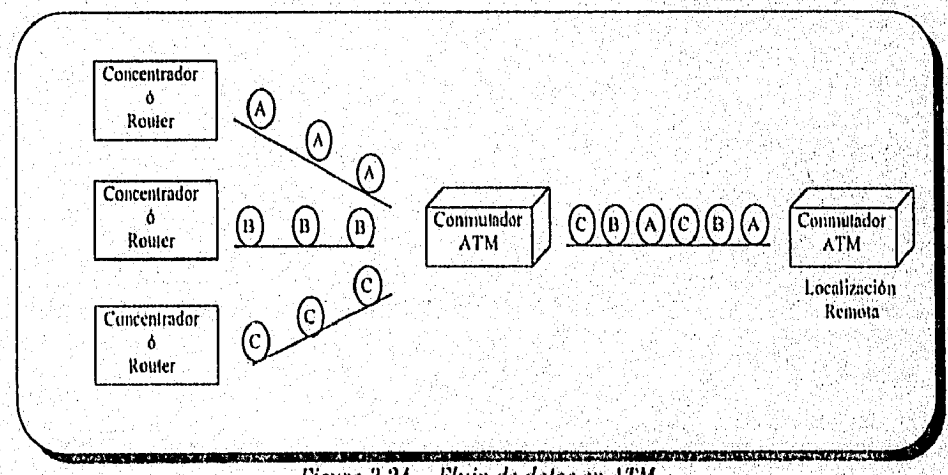
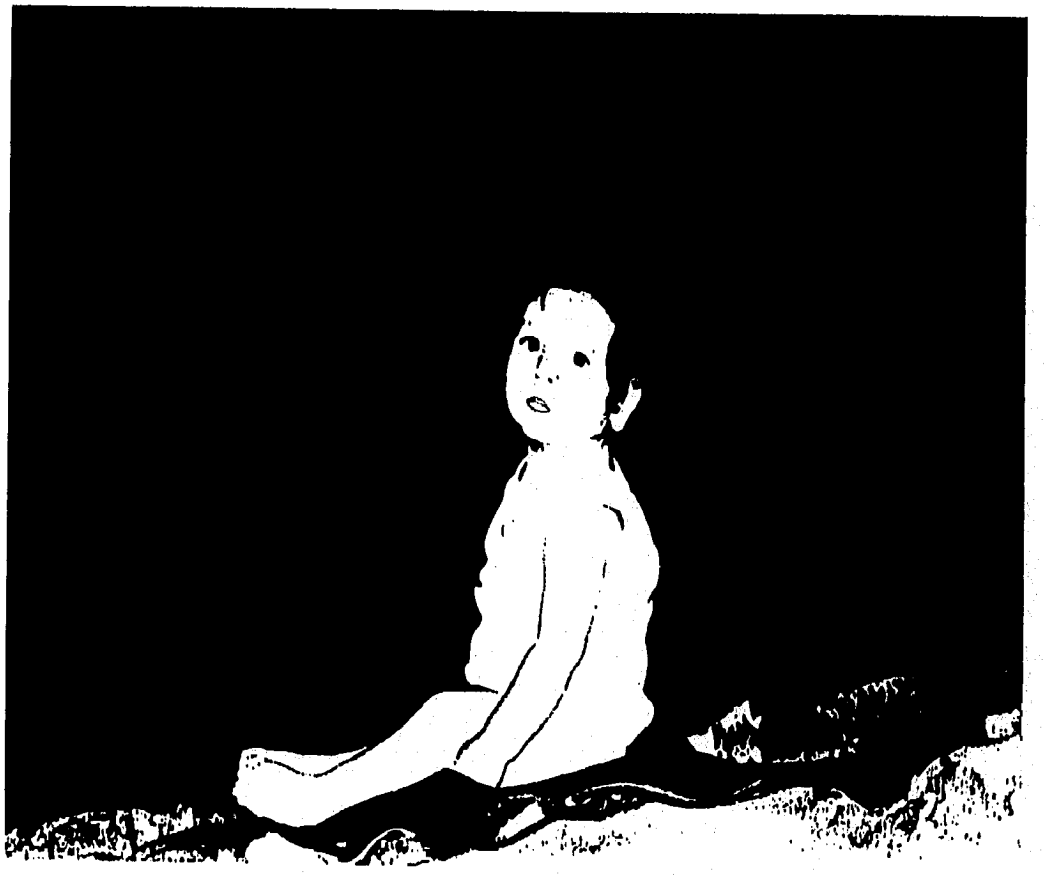


Figura 3.24 .- Flujo de datos en ATM.

CAPITULO IV

SQL



La potencionalidad existe, sólo hay que saber explotarla.

JOSE G. VARGAS.

CAPITULO 4

SQL (LENGUAJE ESTRUCTURADO DE CONSULTAS, " STRUCTURED QUERY LANGUAGE")

Antes de exponer los siguientes incisos de SQL desarrollaremos una idea global del funcionamiento de SQL, ilustrando sus características y funcianes más importantes.

SQL esta basado en el modelo relacional que organiza los datos en una base como una colección de tablas.

- Cada tabla tiene un nombre que la identifica unívocamente.
- Cada tabla tiene una o más columnas dispuestas en un orden específico de izquierda a derecha.
- Cada tabla tiene cero o más filas, conteniendo cada una un única valor en cada columna. Las filas están desordenadas.
- Todos los valores de una columna determinada tienen el mismo tipo de datos, y éstos están extraídos de un conjunto de valores legales llamado el dominio de la columna.

Las tablas están relacionadas unas con otras par los datos que contienen. El modelo de datos relacional utiliza claves primarias y foráneas para representar estas relaciones entre las tablas.

- Una clave primaria es una columna o combinación de columnas dentro de una tabla(s) cuyo(s) valor(es) identifica(n) unívocamente a cada fila de la tabla.

- Una clave foránea es una columna o combinación de columnas en una tabla, cuyo(s) valor(es) es(son) un valor de la clave primaria .

- Una cambinación de claves primarias/foráneas crea una relación padre/hijo entre las tablas que las contienen.

Todas las sentencias en SQL comienzan con un verbo, una palabra clave que describe lo que la sentencia hace (CREATE, INSERT, DELETE, COMMIT), la sentencia continúa con una o más cláusulas (Ver fig. 4.1). Una cláusula puede especificar los datos sobre los que debe actuar la sentencia, o proporcionar más detalles acerca de lo que ésta hace, también comienza con una palabra clave (WHERE, FROM, INTO, HAVING, etc.), algunas son opcionales y otras son necesarias. La estructura y contenido varían de una cláusula a otra.

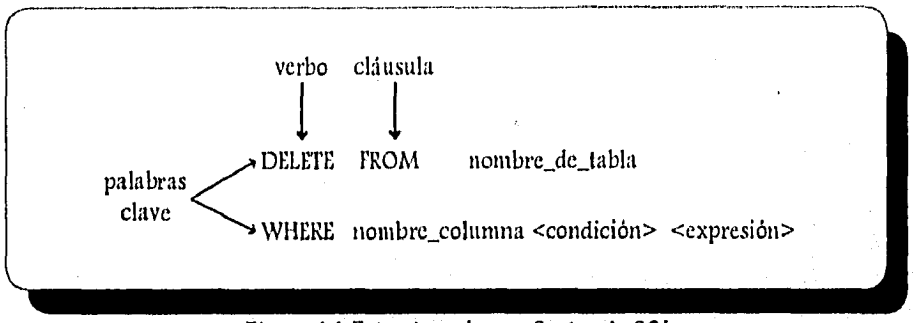


Figura 4.1 Estructura de una Sentencia SQL.

El estándar SQL ANSI/ISO (American National Standards Institute/International Standards Organization) especifica las palabras clave que se utilizan como verbos y cláusulas de sentencias. De acuerdo al estándar, estas palabras clave no pueden ser utilizadas para designar objetos de la base, tales como tablas, columnas y usuarios. Los objetos de una base de datos basada en SQL se identifican asignándoles nombres únicos, utilizándose en las sentencias para identificar el objeto en la base sobre la que la sentencia debe actuar. El estándar ANSI/ISO especifica palabras permisibles para identificar tablas, columnas y usuarios; algunas implementaciones soportan objetos como procedimientos almacenados, relaciones clave primaria/foránea y formularios de entrada de datos, entre otros.

El estándar ANSI/ISO especifica varios tipos de datos que pueden ser almacenados y manipulados por el lenguaje SQL y la mayoría de los productos los soporta, además de ofrecer un conjunto más extenso que éste.

4.1 HISTORIA Y PERSPECTIVAS DE SQL.

Una de las principales tareas de un sistema computacional es almacenar y gestionar datos, por tal motivo, programas especializados en computadoras conocidos como sistemas de gestión de bases de datos comenzaron a aparecer a finales de los sesenta y comienzo de los setenta. Anteriormente un sistema de gestión de base datos o DBMS (*Sistema Manejador de Base de Datos, "Database Management System"*), ayudaba a los usuarios a organizar y estructurar sus datos, y permitía al sistema computacional jugar un papel más activo en la gestión de los datos. Aunque los sistemas de bases de datos se desarrollaron inicialmente en mainframes, su popularidad se ha extendido a minicomputadoras, computadoras personales y estaciones de trabajo.

Durante los últimos años se ha popularizado un tipo específico de DBMS, llamado sistema de gestión de base de datos *relacional* (RDBMS). Las bases de datos relacionales organizan los datos en una forma tabular sencilla y proporcionan muchas ventajas sobre los tipos anteriores de bases de datos. SQL (*Lenguaje Estructurado de Consultas " Structured Query Language"*) es un lenguaje relacional utilizado para trabajar con bases de datos relacionales.

BREVE HISTORIA DE SQL

La historia del lenguaje SQL está íntimamente relacionada con el desarrollo de las bases de datos relaciónales. La Tabla 4.1 muestra algunos de los acontecimientos en sus veinte años de historia.

El concepto de base de datos relacional fue desarrollado originalmente por el Dr. E. F. Ted Codd, quien publicó un artículo titulado «*Un modelo relacional de datos para grandes bancos de datos compartidos*» que esquematizaba una teoría matemática de como los datos podrían ser almacenados y manipulados utilizando una estructura tabular.

Fecha	Acontecimiento
1970	Codd define el modelo de base de datos relacional.
1974	Comienza el proyecto System/R de IBM.
1974	Primer artículo que describe el lenguaje SEQUEL.
1978	Test de clientes del System/R.
1979	Oracle introduce el primer RDBMS comercial.
1981	Relational Technology introduce Ingres.
1981	IBM anuncia SQL/DS.
1982	ANSI forma el comité de estándares SQL.
1983	IBM anuncia DB2.
1986	Se ratifica el estándar ANSI SQL.
1986	Sybase introduce RDBMS para procesamiento de transacciones.
1987	Se ratifica el estándar ISO SQL
1988	Ashton-Tate y Microsoft anuncian SQL Server para OS/2
1988	IBM anuncia la versión 2 de DB2
1989	Primera entrega de servidores de bases de datos SQL para OS/2

Tabla 4.1 Acontecimientos en el desarrollo de SQL.

ACEPTACIÓN COMERCIAL

La publicación del estándar ANSI/ISO para SQL en 1986, dio carácter oficial a SQL como estándar. La tabla 4.2 muestra algunos de los sistemas de gestión de base de datos en SQL más populares sobre diferentes tipos de sistemas de computadoras.

Al inicio de los noventa, SQL estaba claramente establecido como el lenguaje estándar de base de datos. Los vendedores de base de datos que no soportaban SQL se preocuparon de hacerlo sobre todo por que cualquier nuevo producto de base de datos requería a SQL como reclamo para ser tomado en serio; SQL se convirtió en el estándar oficial para las bases de datos relacionales.

DBMS	Sistemas informáticos
DB2	Mainframes IBM bajo MVS.
SQL/DS	Mainframes IBM bajo VM y DOS/VSE.
Rdb/VMS	Mainframes VAX/VMS de Digital.
Oracle	Mainframes, minicomputadoras y PC's.
Ingres	Mainframes y PC's.
Sybase	Mainframes y LANs.
Informix-SQL	Mainframes y PC's basados en UNIX.
Unify	Mainframes basados en UNIX.
OS/2 Extended Edition	Sistemas PS/2 de IBM bajo OS/2.
SQL Server	PC LANs basados en OS/2.
SQLBase	PC LANs basados en DOS y OS/2.
DBASE IV	PC's y PC LANs.

Tabla 4.2 Principales sistemas de gestión de base de datos basados en SQL.

Uno de las desarrollos más importantes en la aceptación del mercado de SQL es la creación de las estándares SQL. Las referencias al «estándar SQL» significa generalmente el estándar aficial adoptado por la ANSI e ISO.

La mayaría de los desarrolladores de saftware están de acuerda en que un estándar para SQL es esencial para el cómputo cliente/servidar. Sin embargo, no menos de ocho proyectos se están llevando a cabo para crear un estándar SQL. Sólo el ANSI es responsable de tres estándares ya sean publicados a en proceso. Otras esfuerzos involucran a consorcios de la industria, tales camo SAG (el grupa de accesa a SQL), X/Open y a compañías como IBM, Microsaft y Borland. Si además consideramos que cualquier Base de Datos "habla su propio dialecta de SQL" y tiene su propio conjunta de extensiones de este lenguaje, no es fácil la tarea para su estandarización .

Las razones para la proliferación de estándares de SQL son numerosas; una de ellas es que los estándares están diseñados salamente para servir como guías, esta deja mucha lugar para la interpretación cuando un proveedor toma una especificación del lenguaje y la pone en práctica en saftware. Sin embargo, la mayaría de las veces los proveedores intentan hacer mucha más y el problema es que aunque cada proveedor tome decisiones excelentes, éstas son diferentes acerca de como extender el estándar de SQL.

A falta de un estándar aceptado, las proveedores de bases de datos están realizando sus propias trabajos de estandarización. A continuación mencionamos los más significativos para la creación de estándares.

- **SQL'89 de ANSI.** El estándar SQL del ANSI ratificado en 1989 fue el primer nivel de estandarización de SQL.

- **SQL'92 de ANSI.** Una extensión al estándar de 1989 que se convirtió en un estándar importante en las siguientes años, canforme los proveedores de bases de datos y herramientas hicieron madificaciones para tener compatibilidad con él.

- **SQL'93 de ANSI.** Un esfuerzo para expandir sustancialmente el lenguaje, proporciona más estructura y estándares para tener acceso a bases de datos orientadas a objetos.

- **SAG (Grupo de acceso a SQL) y X/Open.** SAG es un cansorcio de proveedores de bases de datos y herramientas. X/Open es un cansorcio de la industria dedicada a publicar y reforzar el uso de estándares para sistemas abiertos. Estas dos agrupaciones están trabajando conjuntamente para

establecer un estándar de SQL y un API basados en el SQL'89 de ANSI, y en los últimos avances en el ámbito cliente/servidor.

- **IDAPI (Interface Integrada para la Programación de Aplicaciones de Bases de Datos).** IDAPI está basada en el trabajo de SAG y de X/Open, refleja el intento de Borland por definir un API para SQL.

- **DRDA (Acceso Distribuido a Bases de Datos Relacionales).** El estándar emergente de IBM para acceso a bases de datos a través de todas sus plataformas, incluye otra incursión en SQL.

La creciente popularidad de la conexión de computadoras par red durante los últimos años ha tenido un fuerte impacto en la gestión de base de datos y ha dado a SQL una mayor relevancia. Conforme las redes posan a ser más comunes, las aplicaciones que han corrido tradicianalmente en una minicomputadora o mainframe central se están transfiriendo a redes de área local con estaciones de trabajo de sobremesa y servidores. En estas redes SQL juega un papel crucial como vínculo entre una aplicación que corre en una estación de trabajo y el DBMS que gestiona los datos compartidos en el servidor, en los siguientes incisos mostramos como actúa el DBMS en tres arquitecturas.

ARQUITECTURA CENTRALIZADA

La arquitectura de base de datos tradicional utilizada por DB2, SQL/DS y las bases de datos sobre minicomputadoras tales como Oracle e Ingres se muestra en la figura 4.2. En esta arquitectura el DBMS y los datos físicos residen en un sistema mainframe o minicomputadora central, junto con el programa de aplicación que acepta entradas desde la terminal de usuario y muestra los datos en la pantalla del usuario.

Supongamos que el usuario teclea una consulta que requiere una búsqueda secuencial en la base de datos; por ejemplo, hallar la cantidad media de mercancías de todos los pedidos; el DBMS recibe la consulta, explora la base de datos para acceder a cada uno de los registros de datos del disco, calcula el promedio y muestra el resultado en la pantalla de la terminal.

Tanto el procesamiento de la aplicación como el procesamiento de la base de datos se producen en la computadora central, *y como el sistema es compartido por muchos usuarios, cada usuario experimenta una degradación del rendimiento cuando el sistema tiene una carga fuerte.*

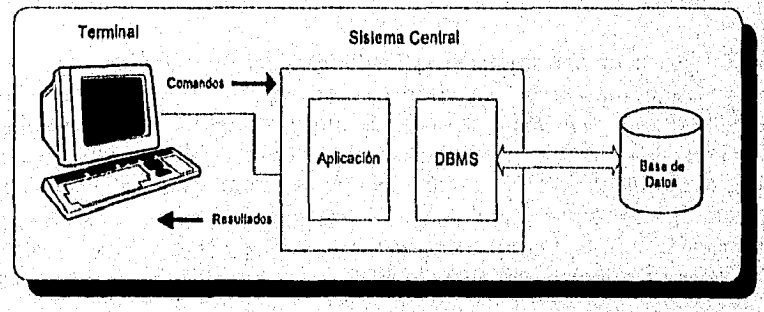


Figura 4.2. Gestión de Base Datos en una arquitectura centralizada.

ARQUITECTURA DE SERVIDOR DE ARCHIVOS

La introducción de las computadoras personales y de las redes de área local condujo al desarrollo de la arquitectura *servidora de archivos*, mostrada en la figura 4.3. En esta arquitectura, una aplicación que corre en un computadora personal puede acceder de forma transparente a datos localizados en un servidor de archivos que almacena los archivos compartidos. Cuando una aplicación en PC solicita datos de un archivo compartido, el software de red recupera automáticamente el bloque solicitado del archivo en el servidor. Varias bases de datos PC populares, entre las que se incluye Dbase, R:BASE y Paradox, soportan esta estrategia servidora de archivos, donde cada computadora personal ejecuta su propia copia del software DBMS.

Para consultas típicas esta arquitectura proporciona un rendimiento excelente, ya que cada usuario dispone de la potencia completa de un computadora personal ejecutando su propia copia del DBMS. Sin embargo, consideremos una consulta que requiere una exploración secuencial de la base de datos, el DBMS solicita repentinamente bloques de datos de la base de datos, la cual está localizada físicamente a través de la red en el servidor, eventualmente *todos los bloques del archivo están solicitados y enviados a través de la red. Obviamente esta arquitectura produce un fuerte tráfico de red y un bajo rendimiento para consultas de este tipo.*

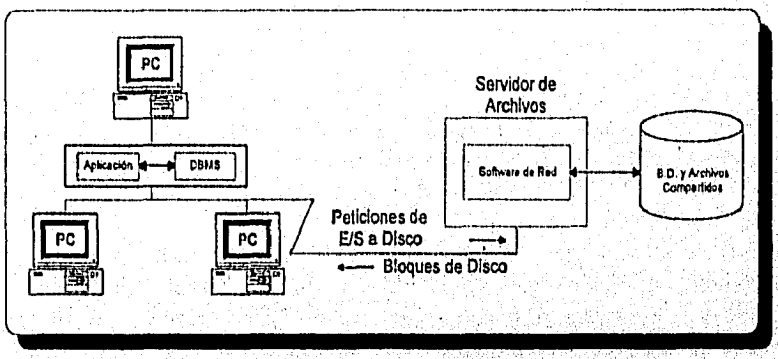


Figura 4.3. Gestión de base de datos en una arquitectura Servidor de Archivos.

ARQUITECTURA CLIENTE /SERVIDOR

La figura 4.4 muestra la emergente arquitectura *cliente/servidor* para gestión de base de datos. En esta arquitectura, las computadoras personales están combinadas en una red de área local junto con un *servidor de base de datos* que almacena las bases de datos compartidas. Las funciones del DBMS están divididas en dos partes: los « frontales » (*front-ends*) de base de datos, tales como herramientas de consulta interactiva, escritores de informe y programas de oplicación, que se ejecutan en la computadora personal y la máquina de « soporte » (*back-end*) de la base de datos que almacena y gestiona los datos se ejecutan en el servidor. SQL se ha convertido en el lenguaje de base de datos estándar para comunicación entre las herramientas frontales y la máquina de soporte en esta arquitectura.

Consideremos una vez más la consulta secuencial. En la arquitectura cliente/servidor, la consulta viaja a través de la red hasta el servidor de base de datos como una petición SQL, la máquina de base de datos en el servidor procesa la petición y explora la base de datos, que también reside en el servidor. *Cuando calcula el resultado, la máquina de base de datos envía de regreso a través de la red una única contestación a la petición y la aplicación frontal la muestra en pantalla de la PC.*

La arquitectura cliente/servidor reduce el tráfico de red y divide la carga de la base de datos. Las funciones de intensiva relación con el usuario, tales como el manejo de la entrada y la visualización de los datos, se concentran en la PC. Las funciones intensivas en proceso de datos, tales como la entrada/salida de archivos y el procesamiento de consultas, se concentran en el servidor de datos. Lo que es más importante, el lenguaje SQL proporciona una interfaz bien definida entre los sistemas frontales y de soporte, comunicando las peticiones de acceso a la base de datos de una manera eficiente.

La arquitectura cliente/servidor ha recibido gran atención con la introducción de redes de PC basados en OS/2. SQL Server, el Servidor Oracle para SQLBase de Gupta Technologies utilizan esta estrategia.

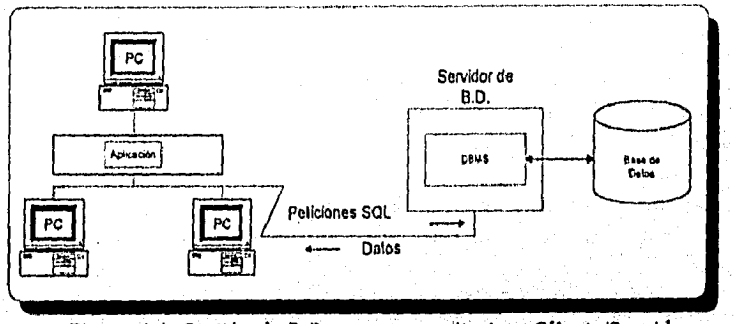


Figura 4.4. Gestión de B.D. en una arquitectura Cliente/Servidor.

SQL es un vehículo natural para implementar aplicaciones utilizando una arquitectura cliente/servidor distribuida. En este papel, sirve como enlace entre los sistemas informáticos «*frontales*» (*front-end*) optimizadas para interactuar con el usuario y los sistemas «de apoyo» (*back-end*) especializados para gestión de bases de datos, permitiendo que cada sistema rinda lo mejor posible. También permite que los computadoras personales funcionen como frontales de bases de datos mayores dispuestas en minicomputadoras y mainframes, proporcionando acceso a datos corporativos desde aplicaciones informáticos personales.

SQL Y EL PROCESAMIENTO DE TRANSACCIONES

SQL y las bases de datos relacionales habían tenido históricamente muy poco impacto en las aplicaciones de procesamiento de transacciones en línea (OLTP). Al hacer énfasis en las consultas, las bases de datos relacionales se confinaron, además, al soporte de decisiones y aplicaciones en línea de baja volumen, en las cuales su rendimiento más lento era una desventaja. Para aplicaciones OLTP, donde cientos de usuarios necesitan acceso en línea a los datos y tiempos de respuesta por debajo del segundo, el Information Management System (IMS) no relacional de IBM predominaba como DBMS.

En 1986 un nuevo vendedor DBMS, Sybase, introdujo una nueva base de datos basada en SQL diseñada especialmente para aplicaciones OLTP. El DBMS Sybase corría en minicomputadoras VAX/VMS y en estaciones de trabajo Sun, y se centraba en obtener un rendimiento en línea máximo. Oracle Corporation y Relational Technology siguieron en breve con anuncios de que también ellos ofrecerían versiones OLTP de sus populares sistemas de base de datos Oracle e Ingres. En el mercado UNIX, Informix anunció una versión OLTP de su DBMS, llamada Informix-Turbo.

En abril de 1988 IBM supo subirse al tren del OLTP relacional con DB2 Versión 2, cuyos programas de prueba mostraban que la nueva versión operaba por encima de 250 transacciones por segundo en mainframes; IBM proclamó que el rendimiento de DB2 era ahora adecuado para casi todas las aplicaciones OLTP excepto las más exigentes, y animó a los clientes a considerarla como una serie

alternativa a IMS. Los bancas de prueba de OLTP se han convertida ahora en una herramienta estándar de ventas para base de datos relaciónales, a pesar de serias cuestiones acerca de lo adecuado de esos programas para medir efectivamente el rendimiento de aplicaciones reales.

La conveniencia de SQL para OLTP continúa mejorando, debida a las avances en la tecnología relacianal y a la mayor potencia del hardware que conducen a tasas de transaccianes cada vez más altas. Las clientes parecen considerar ahora seriamente a DB2 y a las bases de datas relaciónales para su utilización en aplicaciones OLTP.

SQL EN COMPUTADORAS PERSONALES

SQL tuvo paca impacto en las computadaras personales hasta finales de los ochenta, para entances, ya eran camunes las PC patentes que sapartaban decenas a centenares de megabytes de almacenamienta en disco. Los usuarios se canectaban, además, mediante redes de área local y deseaban campartir bases de datas. En resumen, los PC camenzaran a necesitar las característicos que SQL y las boses de datos relaciónales podían proparcianarles.

Las primeras bases de datos basadas en SQL para computadoras personales fueran versiones de las productos populares para minicamputadaras que difícilmente se ajustaban a las computadoros personales. Profesional Oracle, anunciado en 1984, requería 2 Mb de memoria en un IBM PC, y Oracle Macintash, onunciada en 1988, tenía exigencias análagas; uno versión PC de Ingres anunciada en 1987, entraba (aunque muy justamente) dentro del límite de 640 KB de MS-DOS; Informix-SQL para MS-DOS fue anunciado en 1986, proporcionando una versión PC de la popular base de datos UNIX. Tambián en 1986, Gupta Technologies, una empresa fundada par un ex-diretor de Orocle, onunció SQLBase, una base de datos para redes de áreo local PC. SQLBose estuvo entre los primeros productos PC en ofertar una arquitecturo cliente/servidor, un ovance de los anuncios de bases de datos OS/2 por llegar.

SERVIDORES DE BASE DE DATOS BASADOS EN SQL

A finales de 1989 oparecieron una gran cantidad de anuncias de servidores de bases de datos bajo OS/2 y cuatra de los productos emergieron coma líderes en el mercado de servidores de boses de datos camo :

- ***OS/2 Extended Edition***. Aunque su versión servidora aún no estaba comercializada, todos esperaban que el gestor de base de datos OS/2 de IBM jugara un papel importante en el mercado de servicios as/2.

- ***SQL Server de Microsoft***. Respaldado por dos de los tres principales desarrolladares de software para PC, este servidor aparecía como el jugadar independiente más fuerte. Tenía la ventaja de una buena tecnología (pracedente de Sybase), publicidad excelente y una distribución establecida.

- ***Oracle para OS/2 de Oracle***. La versión OS/2 del DBMS relacional independiente más importante reforzaba la buena conexión con las bases de datos Oracle sobre minicamputadaras y mainframes, con la potencia de la fuerza de ventas directas de Oracle.

- ***SQL Base de Gupta***. El primer vendedor con una base de datos en PC LAN fue el más pequeño de los cuatro y un cantendiente oscura, pera el servidar era real, tenía buenos herramientas frontales gráficas, estaba camercializándase (la que no era necesariamente cierta de los otros tres) y funcionaba.

Existían además, productos servidores de bases de datos de Novell (NetWare SQL), XDB System (XDB Server) y de otras compañías más pequeñas.

La arquitectura cliente/servidor también estimuló el desarrollo de herramientas de bases de datos frontales que podían funcionar con los nuevos servidores, quizá el producto frontal más significativo fue Dbase IV, que Ashtan-Tate anunció funcionaría como frontal de SQL Server y de OS/2 Extended Edition. Paradox y DataEasy otras dos bases de datos en PC, también anunciaron frontales para tres o cuatro de los principales servidores. Cada vendedor de servidor también ofertó sus propios productos frontales, incluyendo herramientas de desarrollo de aplicaciones, consultas orientadas a ventanas y facilidades de entradas de datos, herramientas de inspección de la base de datos y otras.

El modelo cliente/servidor de OS/2 también significó un nuevo incentivo para las bases de datos basadas en MS-DOS. Aunque los servidores requerían típicamente la potencia del sistema OS/2, un PC basado en MS-DOS era adecuado como sistema frontal para muchas aplicaciones, tales como entrada de datos, elaboración de informes y consultas interactivas.

Conforme los años ochenta finalizaban, el papel de los servidores de base de datos basadas en SQL y OS/2 se convertía en uno de los temas más novedosos en la prensa informática. Sin embargo, el tamaño último del mercado, la aceptación por parte de los clientes de los sistemas y el relativo éxito de los fabricantes, siguen siendo cuestiones abiertas a resolver en los noventa.

4.1.1 CONCEPTOS Y CONSULTAS

SQL se divide en cuatro grandes categorías de servicios, los cuales son :

I. ***Lenguaje de definición de Datos*** (DDL: Data Definition Language). Consiste en comandos para definir objetos en la base como tablas, vistas, índices e integridad de datos; mediante estos comandos, el DBMS verifica que las acciones que el usuario aplica a los objetos se realicen correctamente.

II. ***Lenguaje de manipulación de datos*** (DML: Data Manipulation Language). Son comandos que le permiten al usuario seleccionar, actualizar, insertar o borrar información en la Base de Datos. Este acceso a los datos y funciones de manipulación es la razón principal de usar un DBMS.

III. ***Lenguaje de control de Datos*** (DCL: Data Control Language). Consiste en comandos que controlan la ejecución de otros comandos. Existen comandos para el manejo de concurrencia, seguridad y consistencia en la Base; por ejemplo, existen comandos que controlan la ejecución de un transacción y restauración de datos si ésta falla.

IV. ***Program Language Constructs***. Son comandos que no pueden ser usados en un modo interactivo con SQL, éstos sólo pueden ser usados desde un lenguaje de programación externo al del DBMS. Su propósito es facilitar la interface entre SQL y los lenguajes de programación, como "C".

La tabla 4.3 muestra los cuatro servicios que ofrece SQL, así como algunos comandos de cada una de estos grupos.

SERVICIO	COMANDOS RELACIONADOS
DDL	CREATE, DROP, ALTER TABLE
DML	DELETE, INSERT, SELECT, UPDATE
DCL	COMMIT, ROLLBACK, GRANT, REVOKE, LOCK TABLE
Program Lenguage Constructs	BEGIN, DECLARE SECTION, EXECUTE, DECLARE CURSOR, OPEN, CLOSE, FETCH

Tabla 4.3 Ejemplos de comandos de servicios de SQL.

ARQUITECTURAS DE BASES DE DATOS

Las figuras 4.5, 4.6 Y 4.7 muestran tres arquitecturas para manejar en forma remota bases de datos; tales como: procesa por cliente (process-per-client), multihilos e híbrida.

- La arquitectura de ***proceso por cliente*** proporciana un máximo de protección de los procesos, dondo a coda base de datos cliente, su propio espacia de dirección para procesos. La base de datos corre en uno o más pracesos formados por separado. La ventaja de esta orquitectura es que protege a un usuario de las otros y también protege al administrador de la base de datos de los usuarios. Además, el proceso puede ser fácilmente cambiado a otro procesador o una máquina multiprocesador SNMP (Protocolo para la administración de red simple, "Simple Network Management Protocol"), ya que la arquitectura depende del sistema operativo local para los servicios de multitarea. La desventaja de proceso por cliente es que utilizan más memoria y recursos del CPU que los esquemas alternativos e incluso puede ser lento ya que los procesos se intercambian y usan IPC. Sin embargo, estos prablemas pueden ser superados fácilmente con el uso de monitor de procesamiento de transacciones (TP monitor) que maneje una ronda de procesos reusables.

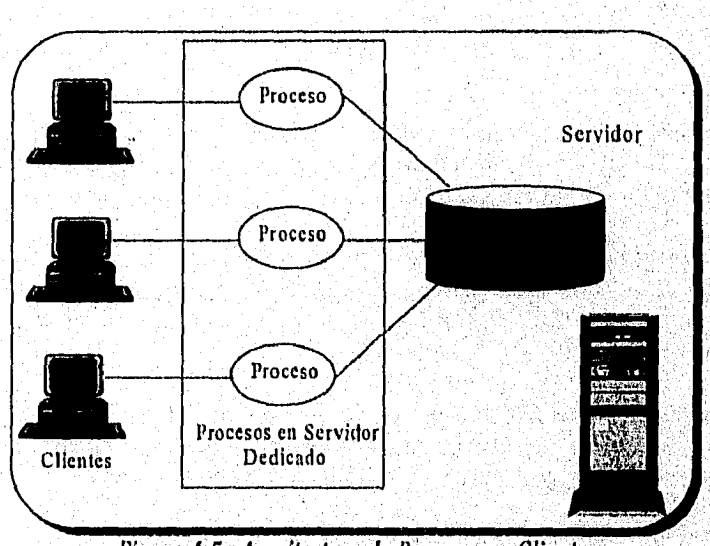


Figura 4.5. Arquitectura de Proceso por Cliente.

- La arquitectura ***multihilos*** ofrece la mejor ejecución corriendo todos los usuarios conectados, las aplicaciones y la base de datos en el mismo espacio de direcciones. Esta arquitectura proporciona su propio programador de tiempos que conserva memoria y capacidad de trabajo del CPU pues no requiere de intercambios frecuentes de contexto.

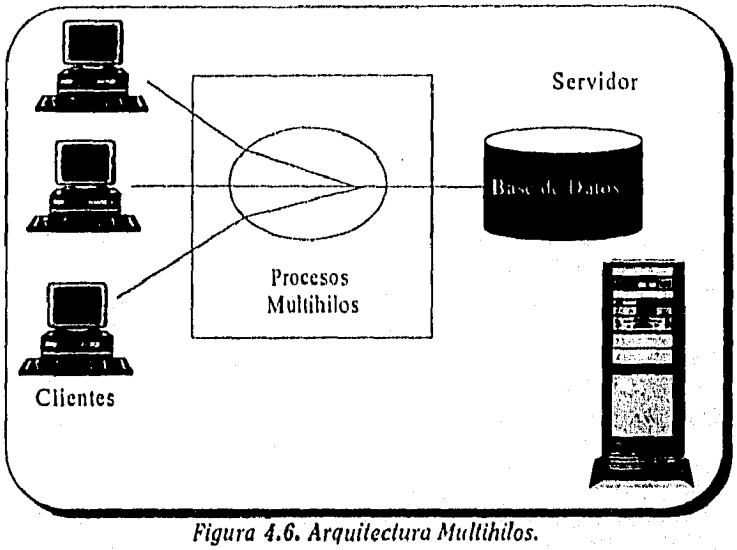


Figura 4.6. Arquitectura Multihilos.

Además, las implemetaciones en el servidor tienden a ser más portables entre diferentes plataformas ya que no requiere de muchos servicios del SO local. La desventaja es que un mal comportamiento de las aplicaciones del usuario puede tirar toda el servidor de la base de datos y sus tareas. Inclusa, las programas de usuario que tienen una duración larga puede consumir todos los recursos del servidor. Finalmente la programación previa de tareas ofrecida por el servidor tiende a ser inferiar que la del SO.

- La arquitectura ***híbrida*** consiste de tres componentes: 1) red multihilos atenta a escuchar que participe en la conexión inicial de tarea al asignar el cliente al despachador; 2) despachadores son tareas que ponen mensajes en una cola de mensajes internos y entonces toma la respuesta y la envía de regreso al cliente; y 3) El servidor de procesos reusables y compartidos que remueve los trabajos para liberarlos de la cola, los ejecuta y coloca la respuesta sobre una cola de salida. La ventaja de esta arquitectura es que ofrece un ambiente protegido para correr las tareas de usuario sin asignar un proceso permanente a cada usuario. Las desventajas son colas latentes. Esta arquitectura es en apariencia buena, su carga de balance no es tan buena como en un monitor TP; de hecho, las colas pueden estar en forma de mönitores TP teniendo sus propios algoritmos de programación de tareas. El primer servidor de base de datos que implementa esta arquitectura es Oracle.

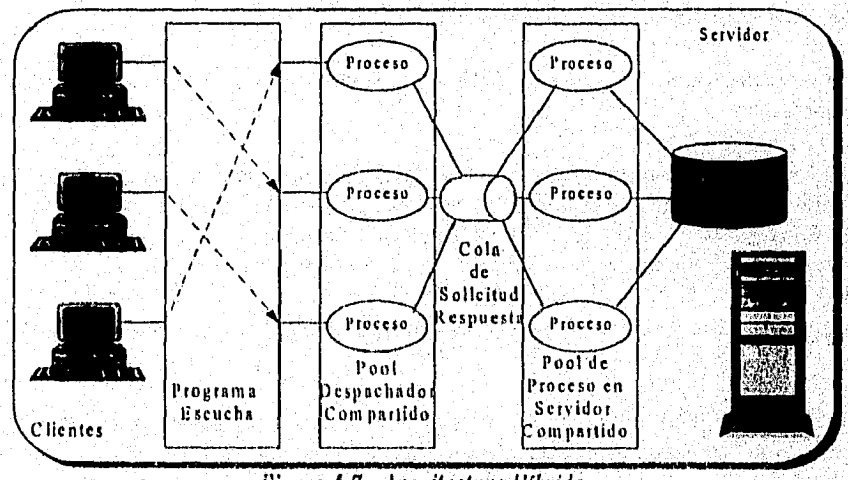


Figura 4.7. Arquitectura Híbrida.

4.1.2 INTEGRIDAD Y TRANSACCIONES

INTEGRIDAD DE DATOS

El término integridad de datos se refiere a mantener la consistencia y validación de los datos almacenados en una base, el contenido de ésta puede perderse de muchas formas al momento de realizar modificaciones. Ante esta, SQL cuenta con características que ayudan al DBMS en la tarea de preservar la integridad de las datos con el objeto de restringir los valores creados o insertados por actualizaciones en la base, como son:

- ***Datos requeridos***: Impide la existencia de valores nulos en campos donde sólo se permite un valor de dato válido.
- ***Comprobación de validez***: Sólo permite insertar valores que se encuentren definidos en el rango especificado para el campo.
- ***Integridad de entidad***: Verifica la unicidad para claves primarias.
- ***Integridad referencial***: Comprueba la relación que para cada llave foránea exista una llave primaria.
- ***Consistencia***: Comprobación de que un conjunto de sentencias tengan efecto en su totalidad en la base de datos y en caso de falla (sea por Software o por Hardware) ninguna de éstas sea realizada, a estos controles se les denomina transacciones, que son tema de este inciso.
- ***Reglas comerciales***: La actualización de datos pueden estar restringidos por reglas establecidos por las empresas. Los transacciones que se realizan en el mundo real están representadas por las actualizaciones a las bases de datos. El DBMS puede comprobar cada nuevo dato introducido, asegurándose que sus valores no violan las reglas comerciales, estos procesos pueden ser implantados en la base a través de los eventos llamados disparadores que se explicarán más adelante.

TRANSACCIONES

SQL posibilita que la secuencia de actualizaciones a una base de datos pueda ser proposiciones "todo o nada" mediante sus características de procesamiento de transacciones.

Una transacción es un bloque secuencial de trabajo formada por una o más sentencias SQL relacionadas e interdependientes.

El DBMS basado en SQL comprueba que la secuencia de sentencias entera se efectúe asegurando la integridad de la base de datos. ***"O todas las sentencias son ejecutadas con éxito, o ninguna de las sentencias es ejecutada"***. El manejador tiene este compromiso incluso si el programa de aplicación aborta o se produce un fallo hardware a mitad de la transacción.

A efecto de soportar las transacciones de base de datos, SQL utiliza las sentencias COMMIT y ROLLBACK (Figura 4.8).

COMMIT: Señala el final con éxito de una transacción; informa al DBMS que todas las sentencias que forman la transacción han sido ejecutadas y la base de datos es autoconsistente.

ROLLBACK: Señala el final sin éxito de una transacción, informando que un usuario no desea completar ésta, ante esto, el DBMS debe deshacer los cambios realizados en la base a fin de restaurar a la base de datos a su estado previo al inicio de la transacción.

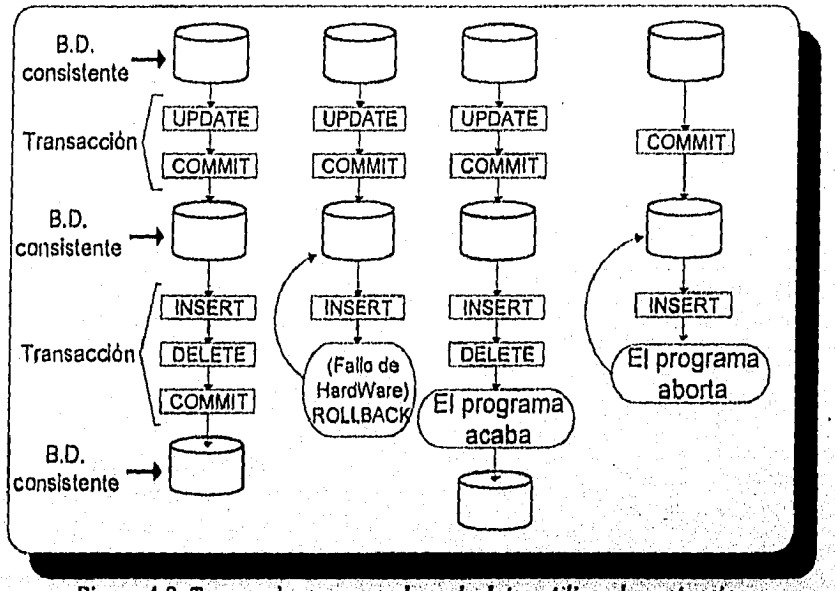


Figura 4.8. Transacciones en una base de datos utilizando sentencias COMMIT y ROLLBACK

EL MODELO DE TRANSACCIÓN ANSI/ISO.

El modelo de transacción SQL (que está basado en DB2), está definida bajo el estándar ANSI/ISO, así como la función de COMMIT y ROLLBACK. El estándar especifica que una transacción SQL comienza automáticamente con la primera sentencia SQL y continúa en forma secuencial con las sentencias SQL subsiguientes hasta finalizar de uno de los cuatro siguientes modos:

Para SQL programado	**Para SQL interactivo**
COMMIT finaliza la transacción con éxito haciendo los cambios en la base permanentes. Se inicia inmediatamente una nueva transacción después de esa sentencia COMMIT.	La terminación de un programa con éxito también finaliza la transacción correctamente; sin embargo, puesto que el programa está finalizado no hay ninguna transacción que comenzar.
ROLLBACK aborta la transacción, deshaciendo las modificaciones hechas a la base. Se inicia inmediatamente una nueva transacción después de la sentencia ROLLBACK.	La terminación anormal de un programa también aborta la transacción; pero puesto que el programa está finalizado no hay ninguna transacción que comenzar.

4.1.3 VISTAS

Una vista es una tabla virtual generada por una consulta a partir de una o varias tablas, en la vista, SQL permite el acceso y modificación de los datos como si fuera una tabla real.

Una vista no existe en la base de datos, es la definición de la vista la que se almacena permanentemente en la base, al efectuar una consulta los datos visibles son el resultado de esta, ver figura 4.9

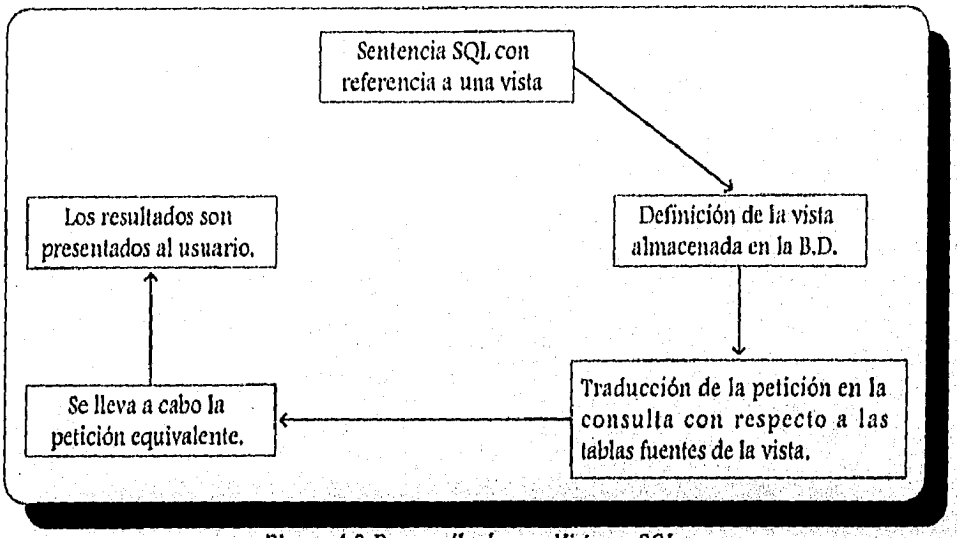


Figura 4.9 Desarrollo de una Vista en SQL.

Al usar las vistas se obtiene los siguientes beneficios:

- Seguridad al mostrar solo los datos autorizados al usuario.
- Simplicidad de consultas multitabla, presentadas como consultas de una sola tabla.
- La estructurado de la BD no se altera ante las consultas
- Integridad de datos. Comprobación de las restricciones de integridad de los datos introducidos a través de la vista.

Al crear una vista, se debe considerar que el DBMS debe traducir las consultas, con respectos a las tablas fuentes relacionadas. Para consultas complejas multitabla, la vista se convierte en una combinación complicada y puede tardar mucho tiempo en completarse. Si el usuario trata de actualizar la información en la vista, el DBMS debe traducir la petición de actualización a las tablas relacionadas en la vista, dado la complejo de este procesa este tipo de vista son creadas de sólo lectura.

El estándar SQL ANSI/ISO, especifica las vistas que pueden ser actualizadas, en base a las consultas que las define, las cuales deben cumplir con las siguientes condiciones:

- Las filas duplicadas no deben de ser suprimidas.
- Sólo una tabla fuente relacionada en la vista puede ser actualizadas siempre que el usuario tenga los privilegios requeridas.
- Cada elemento de selección, debe de ser una referencia de columna simple, la lista de selección no puede contener expresiones, columnas calculadas o funciones de columna
- No se puede definir una subconsulta en la cláusula WHERE.
- La consulta no debe incluir GROUP BY o HAVING

Estas restricciones se pueden resumir en el siguiente concepto:

"Para que una vista sea actualizable el DBMS debe ser capaz de relacionar cualquier fila de la vista con la fila fuente. Análogamente, el DBMS debe ser capaz de relacionar cada columna individual a actualizar con su columna fuente."

Si la vista satisface esta comprobación es posible definir operaciones INSERT, DELETE y UPDATE, y tener efecto sobre los datos fuente.

Tipos de Vistas:

- ***Horizontales***: Muestran sólo las filas seleccionadas sobre una tabla.
- ***Verticales***: Permiten accesar sólo a ciertas columnas seleccionadas de una tabla, cada fila de la tabla fuente está representada en la vista.
- ***Con subconjuntos Fila/Columna***: Los datos visibles son un subconjunto de fila/columna de una tabla, contiene las columnas designadas y las filas que satisfacen una condición de búsqueda.
- ***Agrupadas***: Agrupación de filas relacionadas, generando una fila de resultados de consulta por cada grupo y sumarizando los datos de ese grupo, en estas vistas se requiere mucho tiempo de procesamiento a efecto de mantener la tabla virtual, no se puede actualizar debido, a que no hay modo de trasladar la petición de actualización de la vista a las filas de la tabla fuente, este tipo de vistas son de sólo lectura.
- ***Compuestas***: Extracción de datos de dos a más tablas diferentes, en la que cada fila es la combinación de una fila de cada tabla fuente, en estas vistas el tiempo de procesamiento es considerable, si la consulta llega a ser demasiado compleja, la vista se vuelve de sólo lectura.

Se pretende que todas las vistas lleguen a ser actualizadas en un tiempo mínimo de proceso, pero esto aún se encuentra en proceso de investigación.

4.1.4 SEGURIDAD

La seguridad de la información almacenada en la base de datos es especialmente importante en un DBMS basado en SQL. La implementación de un esquema de seguridad y el reforzamiento de las restricciones de seguridad son responsabilidad del Software del DBMS. El lenguaje SQL define un panorama general para la seguridad de la Base de Datos, y la sentencias SQL son utilizadas para especificar restricciones de seguridad, el cual se basa en tres conceptos principales:

- Los usuarios son los principales demandantes de la información almacenada en la Base de Datos. Cada vez que el DBMS accesa a la base de datos, para recuperar, insertar, suprimir o actualizar datos, lo hace a cuenta de algún usuario, por lo que el DBMS permitirá o prohibirá la acción dependiendo de qué usuario esté efectuando la petición.
- Los objetos de la base de datos son los elementos a los cuales se puede aplicar la protección de seguridad. La seguridad se aplica generalmente a tablas y vistas, pero otros objetos tales como formularios, programas de aplicación y bases de datos enteras también pueden ser protegidos.
- Los privilegios son las acciones que un usuario tiene permitido efectuar para un determinado objeto de la base de datos, por ejemplo, un usuario puede tener permiso para recuperar o insertar datos en una tabla determinada, pero se le puede negar el permiso de borrar o actualizar datos. A cada usuario se le pueden dar diferentes permisos para acceder a la base de datos.

Para establecer un esquema de seguridad en una base de datos, es utilizada la sentencia SQL, GRANT y REVOKE para negar el acceso, como lo muestra el siguiente ejemplo:

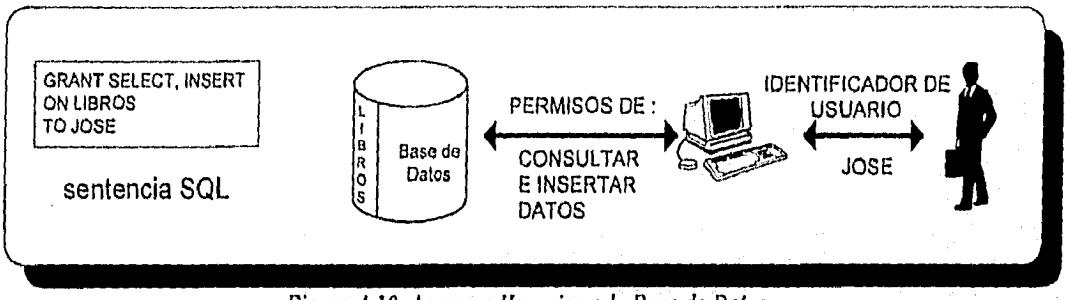


Figura 4.10 Acceso a Usuarios a la Base de Datos.

La sentencia GRANT especifica una combinación de un identificador de usuario (ID-usuario José), un objeto (La tabla Libros) y privilegios concedidos para recuperar e insertar datos. De forma análoga, se pueden negar los accesos permitidos si se cambia la palabra GRANT por REVOKE en el ejemplo anterior.

A cada usuario de la base de datos basada en SQL se le asigna un ID-usuario, que es un nombre breve que lo identifica dentro del DBMS. Todos las sentencias SQL ejecutadas por el DBMS se llevan a cabo a cuenta de un ID-usuario específico, el DBMS, entonces, determina si el ID-usuario se le han concedido privilegios para la sentencia que ha solicitado. El estándar SQL ANSI/ISO utiliza el término ID-autorización que, técnicamente es más adecuado, la razón de que se utilice este término, es que algunos DBMS utilizan el ID-usuario que es asignado para ingresar al sistema.

Las ID-usuarios están asociados con programas o grupos de programas, más que con personas.

En ciertas bases de datos con frecuencia existen grupos de usuarios con necesidades similares. Bajo el esquema de seguridad de SQL ANSI/ISO, se pueden manejar grupos de usuarios con similares necesidades de dos formas diferentes:

1 Se puede asignar un ID-usuario única a todas las personas del grupo, esto simplifica la administración de la base de datos, ya que los privilegios de acceso a datos se asignan al única ID-usuario del grupo, pero la desventaja consiste en que las personas que comparten el ID-usuario no podrán distinguirse en las visualizaciones del operador del DBMS, ni en los informes del DBMS.

2 Se puede asignar un ID-usuario diferente a cada persona del grupo, dando la ventaja de asignar diferentes privilegios de acceso a los datos a cada usuario y distinguirlos en los informes producidos por el DBMS. Sin embargo, esto hace que la administración de la seguridad de la base de datos sea tediosa y propensa a errores.

Por la tanto, el esquema de seguridad que se elija, dependerá particularmente de los compromisos de cada base de datos y aplicaciones para lo que hayan sido creados.

Las protecciones de seguridad en el estándar SQL ANSI/ISO se aplican a dos objetos específicos, tablas y vistas. Así cada tabla y vista en la base de datos puede ser individualmente protegida. El acceso a una tabla o vista puede ser permitido para ciertos ID-usuarios y prohibido para otros. El conjunto de acciones que un usuario puede efectuar sobre un objeto en la base de datos se denomina: privilegios que el usuario tiene sobre éste.

El estándar SQL ANSI/ISO especifica cuatro privilegios sobre los objetos:

1 Recuperación de datas
2 Inserción de datos
3 Supresión de datas
4 Actualización de datos. Posibilita la restricción de columnas específicas, permitiendo actualizaciones y desautorizando a cualquier otra columna.

Estos cuatro privilegios son soportados por casi todos los productos SQL comerciales, sin embargo, la mayoría soporta más de lo que el estándar ANSI/ISO ha definido.

Otro privilegio que es importante mencionar es el de propiedad y éste es otorgado al usuario al instante que crea un objeto (tabla o vista) en la base de datos. Quien crea un objeto en la base de datos se convierte en su propietario y recibe totales privilegios sobre éste, como son: recuperación, inserción, supresión y actualización de datos. Sin embargo, hay que tomar en cuenta que para crear una vista, hay que contar con los privilegios de recuperación de datos y si se tienen los demás privilegios el DBMS los concederá.

Al crear un usurario un objeto, éste es el única que puede otorgar privilegios a otras usuarios para autorizar el acceso a este objeto. Por lo que los privilegios otorgados no podrán ser transmitidos por quien(es) lo(s) haya(n) recibido. De este modo, el propietario de un objeto mantiene un control muy estricto sobre quien tiene permiso para autorizar el objeto y sobre qué formas de acceso se permiten. Sin embargo, la mayoría de los productos SQL, cuentan con opciones de transmisión de privilegios en cascada, dando mayor flexibilidad del control del objeto al propietario.

4.1.5 TRIGGERS "DISPARADORES"

Antes de iniciar el tema de disparadores, es necesario mencionar algunos conceptos.

El estándar SQL ANSI/ISO utiliza la sentencia CREATE TABLE para especificar restricciones de unicidad en columnas o combinaciones de columnas. Sin embargo, los valores NULL presentan un problema cuando aparecen en la clave primaria de una tabla o en una columna que está especificada en una restricción de unicidad.

Por ejemplo, supongamos que se intentase insertar una fila con una clave primaria que fuera NULL (o parcialmente NULL, si la clave primaria está compuesta por más de una columna), debido al valor NULL, el DBMS no puede decidir concluyentemente si la clave primaria está o no duplicada con respecto a otra que ya existe. La respuesta debe ser "quizás", dependiendo del valor "real" del dato que falta (NULL).

Por esta razón, SQL requiere que toda columna que forma parte de una clave primaria y toda columna designada en una restricción de unicidad debe ser declarada NOT NULL.

SQL también cuenta con la restricción de integridad referencial, la cual es una parte esencial del modelo relacional desde que fue propuesto por el Dr. Codd, donde se asegura la integridad de las relaciones padre/hija, creadas mediante claves foráneas y claves primarias.

Los siguientes puntos resúmen los problemas de actualización a la base de datos que pueden corromper la integridad referencial:

- ***Inserción o actualización de una fila "hijo"***: Cuando se inserta o actualiza una fila en la tabla hijo, su valor de clave foránea debe coincidir con uno de los valores de la clave primaria en la tabla padre.

- ***Supresión o actualización de una fila padre***: El DBMS detecta error si se intenta borrar o actualizar una fila de la tabla "padre" a la cual se está haciendo referencia por otra fila de la tabla "hijo".

Aunque el estándar SQL ANSI/ISO no proporciona en su totalidad la solución de los puntos anteriores, existen algunos DBMS como DB2, SQL Windows y SyBASE que incluyen el manejo de integridad referencial más completo.

Citando algunos ejemplos tenemos las siguientes cláusulas:

- **RESTRICT:** Impide suprimir una fila de la tabla padre si la fila tiene alguna fila hijo.
- **CASCADE:** Si una fila padre es suprimida, también son suprimidas todas las filas hijo.
- **SET NULL:** Si una fila padre es suprimida, los valores de la clave foránea en todas las filas hija deben ser actualizadas al valor NULL.

El concepto de disparador se puede definir como: "Para cualquier evento que provoca un cambio en el contenido de una tabla, se puede especificar una acción asociada al DBMS que debe efectuar".

Las sentencias que pueden disparar una acción son:

- INSERT
- DELETE
- UPDATE

La acción disparada por estas cláusulas son especificadas mediante una secuencia de sentencias SQL, como lo muestra el siguiente ejemplo:

```
CREATE TRIGGER    NUEVO PEDIDO

ON    PEDIDOS
FOR   INSERT
AS    UPDATE    REPVENTAS
        SET     ventas = ventas  + Inserted.importe
       FROM     Repventas , Inserted
      WHERE     Repventas.num_empl = Inserted.rep

      UPDATE    Productos
        SET     existencias = existencias - Inserted.cant
       FROM     Productos , Inserted
      WHERE     Productos.id_dir = Inserted.fab
        AND     Productos.id_producto = Inserted.producto
```

Otras extensiones no mostradas en el ejemplo incluyen sentencias IF/THEN/ELSE, iteraciones, llamadas de procedimiento, e incluso sentencias PRINT que visualizan mensajes de usuario.

Los disparadores proporcionan un modo alternativo de implementar las restricciones de integridad referencial, proporcionados por claves foráneas y primarias. De hecho los defensores de esta característica, señalan que el mecanismo disparador es mucho más flexible que la integridad referencial proporcionado por algunos DBMS e incluso por el estándar SQL ANSI/ISO.

La principal ventaja de los disparadores, es que las reglas comerciales pueden almacenarse en las bases de datos y ser forzadas consistentemente con cada actualización que tengan. Esto puede reducir substancialmente la complejidad de los programas de aplicación que acceden a la base de datos. Sin embargo, algunas desventajas son:

- ***Complejidad de la Base de Datos***: Cuando las reglas se trasladan al interior de la base, prepararla, pasa a ser una tarea más compleja. Las usuarias que razonablemente podían esperar crear pequeñas aplicaciones propias con SQL, encontrarán que la lógica del programa de los disparadores hace la tarea mucho más difícil.

- ***Reglas Ocultas***: Con las reglas ocultas dentro de la base de datos, programas que parecen efectuar sencillas actualizaciones, pueden de hecho, generar una enorme actividad en la base. El programador ya no controla totalmente lo que sucede en la base de datos.

Las disparadores están ausentes del estándar SQL ANSI/ISO, aunque algunos vendedores de DBMS han proclamado públicamente planes para añadir las características de los disparadores a las versiones futuras de sus productos, la importancia futura es un poco nebulosa y solamente el tiempo dirá si se convierten en parte importante del entorno SQL.

4.2 SQL WINDOWS

Al elegir el software de desarrollo se debe verificar que el funcionamiento (tanto de las herramientas de desarrollo como del manejador de BD) sea óptimo en la plataforma de hardware en que se implantó. Dentro de un ambiente cliente/servidor se incluye el software para el desarrollo de aplicaciones SQL Windows de Gupta Tecnologies, como una alternativa muy recomendable, la cual ofrece un ambiente de desarrollo rápido y poderoso en Windows de Microdsoft, que incluye la arquitectura Quick Object, para el rápido desarrollo de aplicaciones gráficas para PC. Dentro de este producto de desarrollo se incluye un compilador de lenguaje de cuarta generación, que convierte el lenguaje en código "C" , además de contar con características de programación en grupo.

Las herramientas de desarrollo visuales con que cuenta proporcionan un ambiente poderoso y de mayor velocidad, tanto para desarrollos individuales como en grupo. Su base de datos es compacta y con un alto desempeño, esta proporciona una fácil escalabilidad en comunicaciones, desde avanzados dispositivos hasta la integración de servidores dedicados de redes LAN e incluso WAN.

Sus principales herramientas son:

- **SQL Windows**: Ambiente de desarrollo rápido y poderoso que incluye la arquitectura Quick Object, para el desarrollo de aplicaciones gráficos para PC. Este producto de desarrollo incluye un compilador de lenguaje de cuarta generación, que convierte el lenguaje de cuarta generación en código "C" y cuenta con altas capacidades de programación en grupo.

- **SQLBase:** Es el servidor de base de datos

- **Quest:** Herramienta de consulta a bases de datos, contiene un generador de informes, que por su fácil manejo se encuentra orientada a hacia usuarias finales.

- **SQL Talk**: Herramienta gráfica de administración y monitoreo de base de datos. Accesa datos en forma gráfica, permitiendo manipular la información desde simples query's hasta sofisticados reportes, teniéndo accesa a diferentes plataformas como mainframes, minicomputadoras, servidores. Permitiendo utilizar otras DBMS como DB2.

- **Report Windows**: Permite la creación de reportes de manera rápida y sencilla interactuando con el software de desarrollo SQL Windows y Quest.

- **SQL Network**: Software de conectividad nativa que permite el intercambio de datos entre los datos cliente/servidor de la PC en red y las aplicaciones y las base de datos de las minicamputadoras y mainframes basadas en huéspedes.

SQLWindows cuenta con un completo medioambiente visual de desarrollo de aplicaciones, la herramienta visual de programación esta integrada al código fuente de la aplicación. Ver figura 4.11.

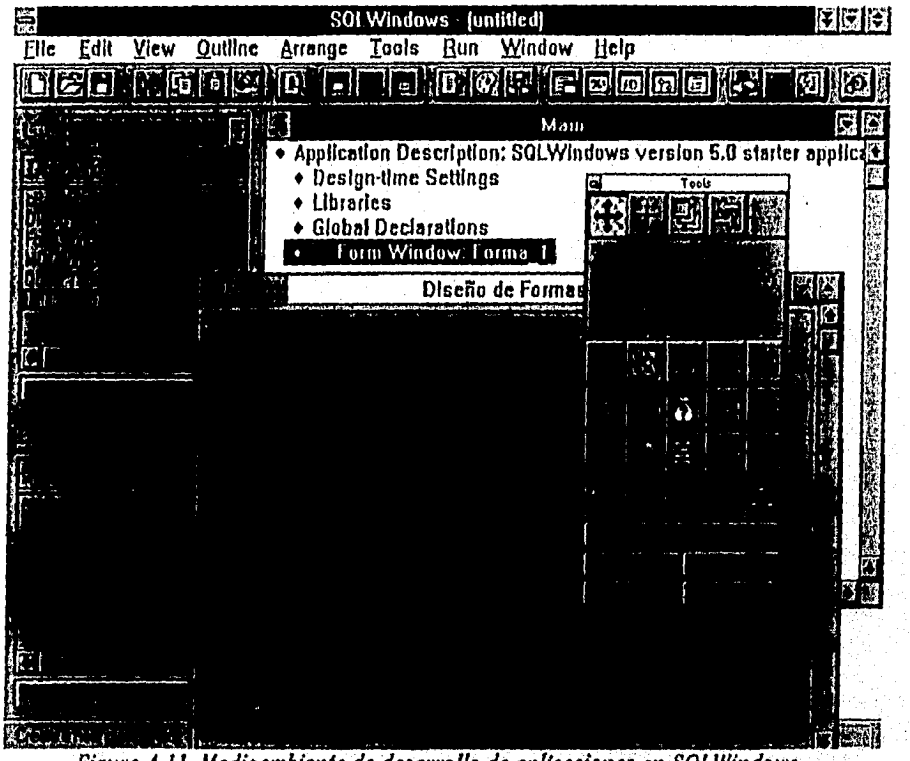


Figura 4.11. Medioambiente de desarrollo de aplicaciones en SQLWindows.

La figura 4.11 muestra la relación entre la herramienta visual de objetos (tool) y el código fuente. El programador sólo tiene que elegir el objeto en la paleta de herramientas y depositarlo en el diseñador de formas; SQLWindows automáticamente adiciona el código en la aplicación que representa al objeto seleccionado.

La paleta de herramientas contiene la mayoría de objetos que el desarrollador necesita, y brinda la facilidad de poder crear otras clases de objetos. Ver figura 4.12.

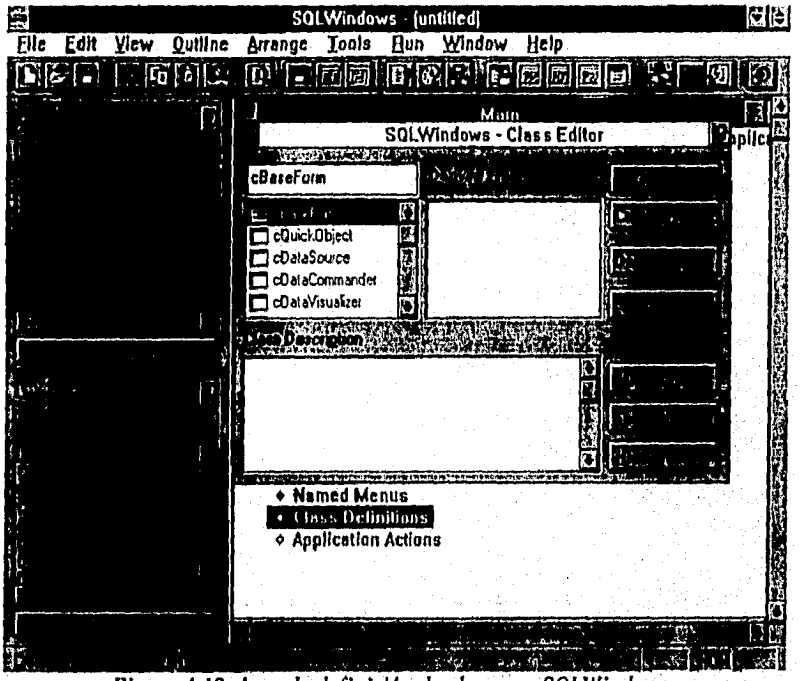


Figura 4.12. Area de definición de clases en SQLWindows.

En el archivo que contiene el código fuente de la aplicación; SQLWindows despliega la paleta de instrucciones con el código y los objetos que sólo pueden adicionados en la instrucción subsecuente a la línea de código activa, aunque esta no impide anexar líneas con código erróneo.

El código de la aplicación pude ser comprimido hasta una sola línea o expandirse hasta un nivel total para visualizar detalles como: posición del objeto, tipo de letra que contiene el objeto y otros. Brindando la facilidad al desarrollador de que únicamente desplegar el código del objeto en que esté interesado.

SQLWindows también cuenta con ayuda en línea y sensitiva a la instrucción que desea conocer, sólo basta elegir la instrucción en la aplicación y solicitar la ayuda para que se despliegue el texto que explica ésta, así como instrucciones relacionadas a esta. Ver figura 4.13.

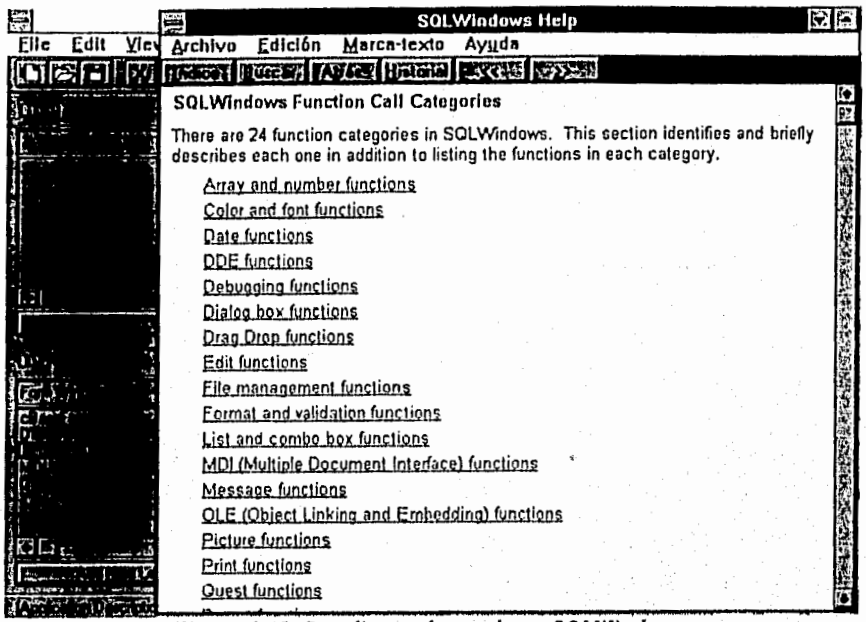


Figura 4.13. Despliegue de ayuda en SQLWindows.

4.2.1 SQL BASE

El campo de los nuevos datos corporativos requiere un nuevo acercamiento hoy en día, el mainframe ya no es el depósito exclusivo, los datos son dispersados a todas las partes de la corporación, y más allá, a computadoras de escritorio y portátiles, facilitando a los usuarios la toma de decisiones más confiables, respondiendo más rápidamente a los cambios y mejorando el entendimiento de los clientes.

La descentralización está modificando el terreno de los datos corporativos y demandando estrategias de dirección más dinámicas y acertadas en la actualidad.

Las capacidades de escalabilidad, sofisticación y monitoreo propio de datos, demandadas hoy en día, responsabilizan al DBMS como el vehículo perfecto para el manejo de nuevos y complejos datos.

Actualmente, las corporaciones requieren de un DBMS que puede ejecutarse como un servidor central para un grupo de nodos en la red o en un dispositivo lo bastante pequeño para adaptarse en un estilo de manejador de bolsillo. El modelo cliente servidor ayuda a la descentralización de los datos al moverlos fuera de los mainframes y minicomputadoras a poderosas computadoras personales (servidores).

Dentro de la línea de productos DBMS que existen en el mercado, SQLBase de Gupta Corporation responde a las expectativas de las corporaciones. Este producto está diseñado para satisfacer las necesidades de un amplio expectro que va desde usuarios en grupos de trabajo hasta usuarios móviles. Asimismo, SQLBase ofrece utilización sencilla de opciones para trabajo en grupo, características mejoradas para desarrollo de aplicaciones, escalabilidad y capacidad extendidas. Es de fácil instalación y configuración, operación inteligente de auto-administración facilidades de monitoreo remoto y administración (incluyendo soporte SNMP), interface gráfica de usuario para la configuración y administración de todos los componentes de la BD, procedimientos almacenados, disparadores y "Two-phase commit", nueva arquitectura para cargar datos y capacidad expandida para BD muy extensas.

La familia de productos incluye :

- **SQLBase Server**: Servidor para grupos de trabajo robusto basado en redes, diseñado para correr aplicaciones para grupos de trabajo.

- **SQLBase Desktop**: Base de datos de grupos de trabajo que aprovechan la habilidad de los actuales sistemas operativos (Windows NT, Windows OS/2 y Windows 95) para eliminar la distinción entre cliente/servidor. Su flexibilidad soporta configuraciones diversas de grupos de trabajo incluyendo cliente a servidor, aplicaciones móviles "stand-alone" y procesamientos en computadoras de escritorio cliente-a-cliente en los cuales la información se comparte directamente entre las computadoras de escritorio del grupo de trabajo.

- **SQLBase Ranger**: Es un movilizador de datos programable para aplicaciones descentralizadas que permite a los desarrolladores construir aplicaciones que movilizan la información de bases de datos corporativas o computadoras de escritorio a cualquier parte de la organización en una base de datos SQL compacta de alto desempeño.

- **SQLBase Runtime**: Es un vehículo económico para la utilización de aplicaciones en computadoras de escritorio, con él, los desarrolladores pueden crear versiones "runtime" de una base de datos SQLBase 5.2. La información puede insertarse, actualizarse o borrarse libremente de la aplicación del usuario. Sin embargo, las estructuras de las bases de datos son establecidas durante el desarrollo y no pueden ser alteradas durante el tiempo en el que están corriendo.

Es importante señalar que Gupta Corporation cuenta con un software (SQLNetwork) para la conectividad con otros DBMS como :

- IBM DB2, AS/400 y OS/2
- Oracle
- Sybase SQL Server
- Microsoft SQL Server
- Informix
- Ingres
- Cincom Supra
- HP ALLBASE/SQL
- dBase
- Paradox y otras.

4.2.2 SQL TALK

Es una interface interactiva para SQLBase, que permite administrar las BD a través de comandos SQL permitiendo:

- Definir la estructura.
- Adicionar, suprimir y cambiar datos.
- Consultar la información.
- Controlar la seguridad y acceso .
- Generar Reportes.
- Funciones de administración.

Al activar SQLTalk se despliega la pantalla que se muestra en la figura 4.14, la cual está compuesta por un menú principal, la ventana de introducción de comandos (ventana superior), y la ventana de resultados a los comandos introducidos (ventana inferior).

- **SQLBase Server:** Servidor para grupos de trabajo robusta basado en redes, diseñada para correr aplicaciones para grupas de trabajo.

- **SQLBase Desktop:** Base de datos de grupas de trabajo que aprovechan la habilidad de los actuales sistemas operativos (Windows NT, Windows OS/2 y Windows 95) para eliminar la distinción entre cliente/servidor. Su flexibilidad soporta canfiguraciones diversas de grupos de trabajo incluyenda cliente o servidor, aplicaciones móviles "stand-alone" y procesamientos en computadoras de escritario cliente-a-cliente en los cuales la información se comparte directamente entre las computadoras de escritorio del grupo de trabajo.

- **SQLBase Ranger:** Es un mavilizador de datos programable para aplicaciones descentralizadas que permite a las desarrolladores construir aplicacianes que movilizan la infarmación de bases de datos carporativas a computadoras de escritoria o cualquier parte de la organización en una base de datos SQL compacta de alta desempeño.

- **SQLBase Runtime:** Es un vehícula ecanómico para la utilización de aplicaciones en camputadaras de escritario, can él, las desarralladores pueden crear versianes "runtime" de una base de datos SQLBase 5.2. La infarmación puede insertarse, actualizarse a barrarse libremente de la aplicación del usuario. Sin emborga, las estructuras de las bases de datos son establecidas durante el desarrollo y no pueden ser alteradas durante el tiempo en el que están corrienda.

Es importante señalar que Gupta Carporatian cuenta can un saftware (SQLNetwark) para la conectividad con otros DBMS cama :

- IBM DB2, AS/400 y OS/2
- Oracle
- Sybase SQL Server
- Microsoft SQL Server
- Informix
- Ingres
- Cincom Supra
- HP ALLBASE/SQL
- dBase
- Paradox y atras.

4.2.2 SQL TALK

Es una interface interactiva para SQLBase, que permite administrar las BD a través de camandos SQL permitiendo:

- Definir la estructura.
- Adicionar, suprimir y cambiar datas.
- Consultar la información.
- Cantralar la seguridad y accesa .
- Generar Reportes.
- Funciones de administración.

Al activar SQLTalk se despliega la pantalla que se muestra en la figura 4.14, la cual está campuesta por un menú principal, la ventana de introducción de comandos (ventana superior), y la ventana de resultados a los camandos introducidos (ventana inferiar).

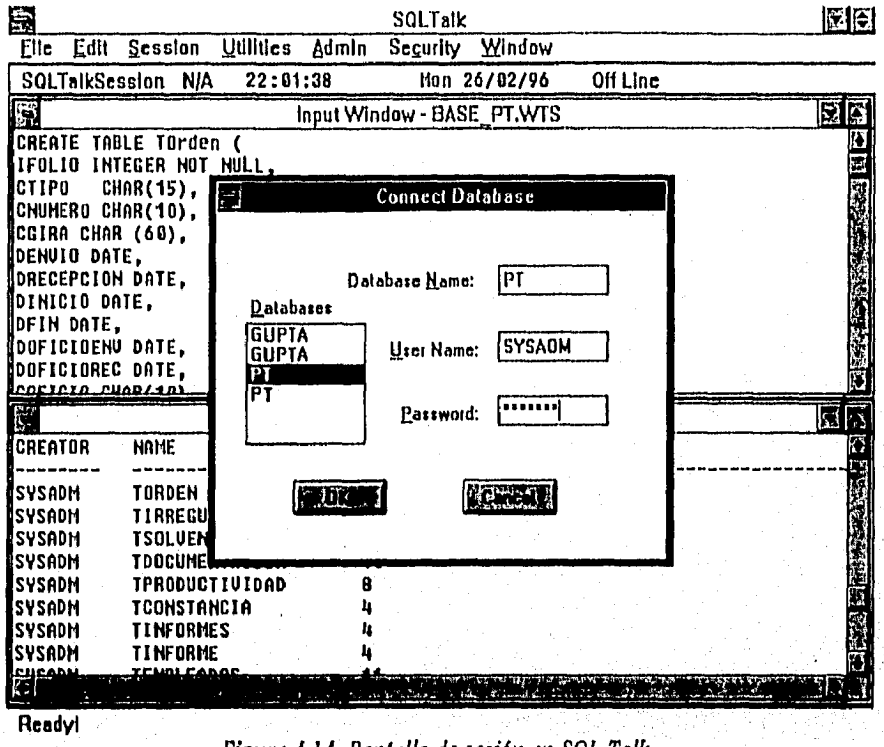


Figura 4.14. Pantalla de sesión en SQL Talk.

Los comandos que pueden ser introducidos en SQLTalk se pueden agrupar en las siguientes categorías :

- **Control de Sesión** — Comandos que conectan o desconectan a la(s) base(s) de dato(s); CONNECT,DISCONNET,EXIT,USE, etc.
- **Administración a la B.D.** — Comandos de las funciones de administración de la B.D. ; BACKUP, COPY, LOAD, RESTORE, SET SERVER, etc.
- **Reporteador** — Estos comandos formatean, despliegan e imprimen el resultado de sentencias SQL en un reporte; BRAEK, BTITLE, COLUMNCOMPUTE, PRINT, TTITLE, etc.
- **Procedimientos almacenados** — Comandos que almacenan, ejecutan y borran sentencias SQL; ERASE, EXECUTE y STORE.
- **Archivos de Comandos** — Archivos que contienen los comandos introducidos en una sesión y que pueden ser ejecutados; PAUSE, REMARK, RUN,SAVE.
- **Control de Configuración** — El comando SET, permite controlar el medioambiente de la sesión en SQLTalk. El comando SHOW despliega el medioambiente en que se encuentra la sesión activa en SQLTalk.

4.2.3 SAL "SYSTEM APLICATION LANGUAGE "

El código de las aplicaciones creadas en SQL Windows es escrito en SAL, un lenguaje especialmente diseñado para construir aplicaciones gráficas. Estas aplicaciones son creadas particularmente para la presentación de batas en el Front-end.

Este lenguaje ofrece un alto nivel de funciones para la creación de aplicaciones y maneja a las bases de datas que se encuentran en el Back-End.

SAL proporciana un rango de funciones para soportar características de ventanas como: DDE (Dynamic Data Exchange), OLE (Objet Linked and Embebeding), MDI (Multiple Data Interface) y Drag and Drop (Arrastrar y saltar). Con las funciones que contiene la manipulación de base de datos es muy accesible.

Las funciones se encuentran divididas en las siguientes categorías de funciones:

- **Matemáticas y de arreglos.**
- **Para colores y tipos de letra.**
- **De fecha/hora .**
- **DDE (Intercambio dinámico de datos), soportadas por protocolos de Microsoft.**
- **Debugging (Depuración de código).**
- **Para cajas de diálogo.**
- **Para mover y arrastrar objetos (drag y drop).**
- **De edición (campos de datos).**
- **Para la administración de archivos.**
- **De formatos y validación.**
- **Para caja de lista y caja de tipo combo.**
- **MDI (Interfaz de documentos múltiples).**
- **De mensajes.**
- **OLE (Vinculación e incrustación de objetos).**
- **De imagen.**
- **De impresión.**
- **Quest (Funciones de interface con la herramienta Quest).**
- **De reportes.**
- **Para la barra desplazadora.**
- **De sentencias en SQL.**
- **Para conversión de datos de tipo string a tipo de fecha/hora.**
- **Del sistema.**
- **De ventanas de tablas.**
- **Para la administración de Windows.**

SAL es un lenguaje por medio del cual se puede realizar una verdadera programación orientada a objetos (definir clases, herencia múltiple y simple, polimorfismo), dentro de este lenguaje todas las operaciones que se llevan a cabo en la aplicación son eventos que manejan objetos. En una explicación básica, los objetos son ventanas de varios tipos, de nivel principal (padre) y nivel secundario (hijo), donde se les asigna un identificador para hacer referencia a estos. Las ventanas pueden contener otros objetos como campos de datos, cajas desplegables, más una gran variedad de objetos.

Un concepto fundamental en la programación orientada a objetos es el manejo de eventos y mensajes. Un evento es invocado a través de un mensaje, en el evento existe una sección especial para ejecutar las operaciones para lo que el objeto fue diseñado, si en esta sección no se definieron operaciones que

realizor, el objeto no harán nada, sólo será invocado por el mensaje. Existen cerca de cuarenta mensajes y citaremos algunas para dar una idea más concreta de lo que hacen estos:

- ***SAM_AppStarup***: Mensaje que recibe la parte definida como acción de la aplicación (Application Action) antes de que cualquier objeto sea creado dentro de ésta.

- ***SAM_AnyEdit***: Mensaje que se recibe a un campo de edición, campo texto, o tabla de ventana, cuando el usuario realiza cualquier cambio sobre éste.

- ***SAM_Click***: Mensaje que recibe un objeto cuando es activada por el ratón o el teclado.

- ***SAM_SqlError***: Mensaje que es enviada al objeto que invocó la instrucción SQL que produjo un error, si el objeto no contiene una sección del manejo de error, el mensaje es enviado a la acción de la aplicación, y de no existir un maneja del error producido se envía un mensaje al ambiente que la aplicación ha producida un error en Windows con lo cual, la aplicación es abortada y se retorna el control o Windows.

4.2.4 REPORT WINDOWS

Es el software que permite crear reportes gráficos de manera sencilla en SQLWindows, cuenta con las siguientes características:

- Diseñar e Imprimir varios tipos de reportes.
- Formatear cada elemento del reporte con acceso fácil al formato.
- Usar agrupamientos de campos para emitir operaciones de cálculos
- Importar y desplegar datos y gráficos desde diferentes lugares.

Report Windows es una herramienta flexible que permite generar y ejecutar reportes de diferentes orígenes, éste puede ser utilizado desde SQL Windows y Quest.

SQL Windows trabaja con Report Windows por medio de llamadas a funciones SAL y mensajes. Report Windows pregunta por la línea de datos y SQL Windows regresa los datos a éste, después del procedimiento se puede desplegar, cambiar, interactuar o imprimir el reporte (figura 4.15).

Básicamente un reporte en Report Windows está constituída por grupos de bloques :

- Report Header: Aparece una sola vez en el reporte, y es utilizado para asignar el título general al reporte .

- Page Header: Contiene inscripciones/rótulos como: número de página, referencias, totales de cada página, etc. Estos desplegados aparecen una sola vez por página, excepto en la primera donde aparece el Report Header.

- Group Header: Son desplegadas antes de cada línea del grupo de registros y es impreso cada vez que el registro cambia.

- Footers: Están constituidas por tres tipos, que son:

  - Report Footers: Contiene totales del reporte e información sumaria.
  - Page Footers: Está localizada en el margen inferior de cada página y típicamente contiene número de página, totales de página y fecha.
  - Group Footers: Son desplegados después de la última línea de registros de cada grupo definido, y son impresas cada vez que un nuevo grupo cambia.

- Break Groups: Pueden ser usados para subtotalizar o paginar. Cada grupo debe tener elementos de entrada que son llamados al reporte cuando cada grupo definido cambia.

- Detail Blocks: Contiene información de cada fila (registro de datos) que es enviado al reporte.

- Input Item: Son elementos individuales en el reporte como campos o datos originados en la aplicación, las cuales constituyen los datos de entrada hacia el reporte.

Además, dispone de una gran variedad de elementos para realzar o dar una mejor presentación al reporte como:

- Imágenes que pueden contener objetos OLE, DDE, etc.
- Líneas, recuadros, sombreados, etc.
- Texto, con características de tipos de letra, tamaño, etc.

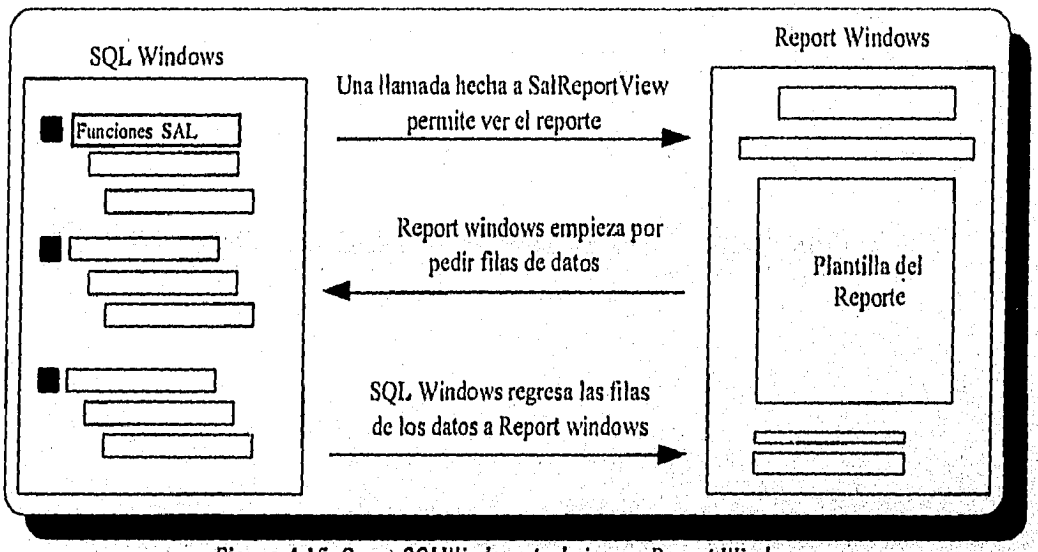


Figura 4.15. Como SQLWindows trabaja con Report Windows.

Como se observa en la figura 4.16, el reporteador cuenta con una paleta de diseño, que permite al desarrollador insertar imágenes, texto (personalizado), recuadros, líneas, etc., para dar una excelente presentación a la creación de reportes.

Como se mencionó anteriormente, ReportWindows trabaja a través de bloques, con el fin de proporcionar un fácil manejo de datos y tener un mejor control sobre el reporte; cuenta con la opción de presentación preliminar del reporte que se este diseñando, para dar una idea exacta de lo que se desea crear, esta opción está disponible aunque no se disponga de datos fuente en el reporte.

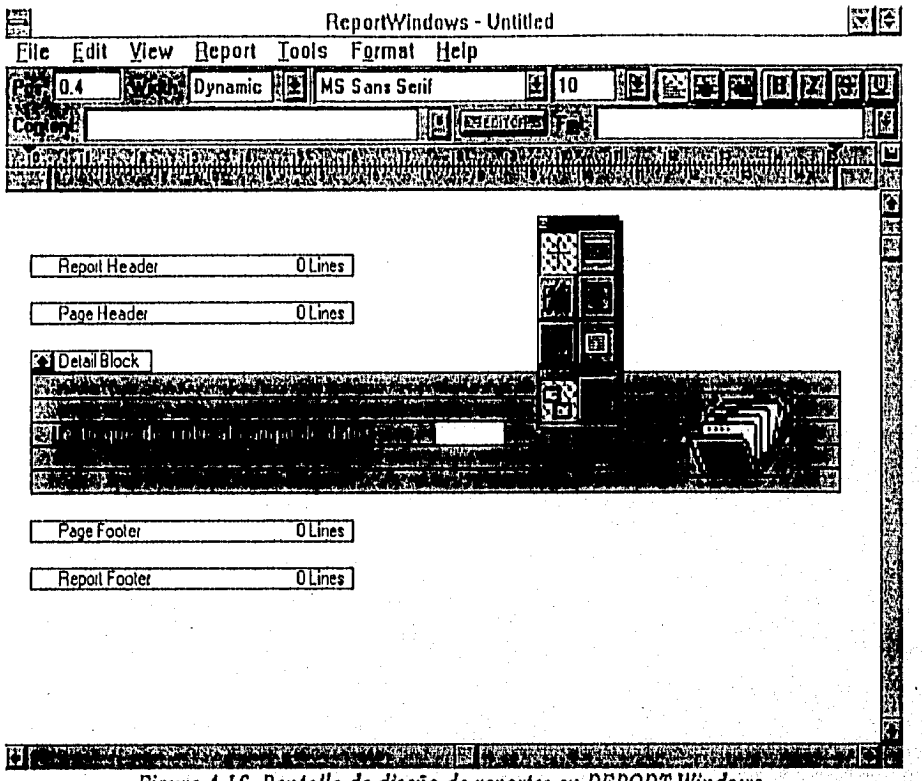
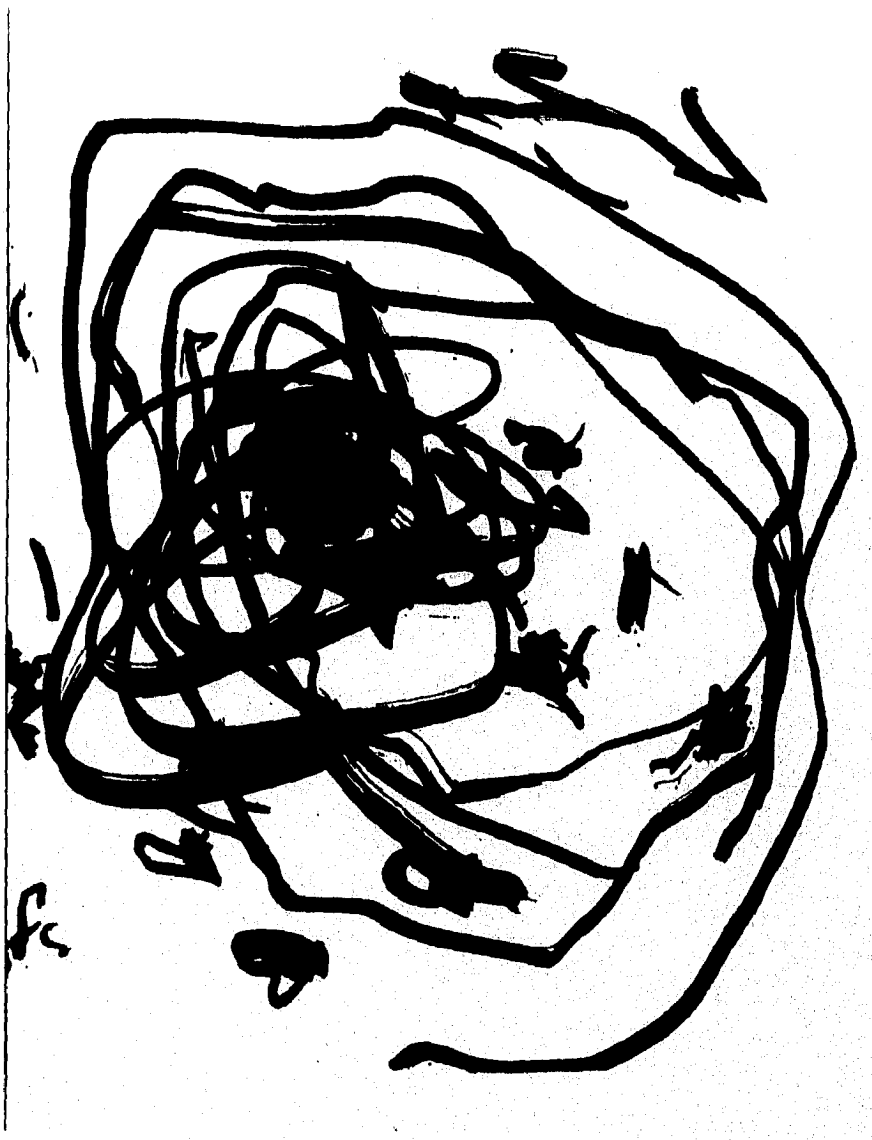


Figura 4.16. Pantalla de diseño de reportes en REPORT Windows.

CAPITULO V

DESARROLLO DEL SISTEMA



El arte de modelar naturalmente.

JOSE ALBERTO.

CAPITULO 5

DESARROLLO DEL SISTEMA

Con el análisis Orientado a Objetos (OO) la forma de modelar la realidad difiere del análisis convencional, modelamos el mundo en términos de tipos de objetos y lo que ocurre a éstos. El modelo representa un aspecto de la realidad y se construye de modo que nos ayude a comprender a ésta. Después de toda, deseamos capturar el punto de vista de los usuarios respecto al mundo y traducirlo en software de la manera más automática y confiable posible. En el análisis OO, se construyen dos tipos de modelos estrechamente relacionadas que se representan mediante diagramas llamadas esquemas.

I **Modelo de los tipos de objetos y sus estructuras:** se refiere a los tipos de objetos, clases, relaciones entre los objetos y herencia AEO (Análisis de la Estructura de Objetos) y DEO (Diseño de la Estructura de Objetos).

II **Modelo de lo que le ocurre a los objetos:** se refiere al comportamiento de los objetos y lo que les ocurre al paso del tiempo. ACO (Análisis del Comportamiento de Objetos) y DCO (Diseño del Comportamiento de Objetos).

En las metodologías tradicionales para la creación de sistemas, los modelos conceptuales utilizados para el análisis difieren de los que se emplean para el diseño. La programación tiene también un tercer punto de vista del mundo. Los analistas utilizan los modelos de relación entre entes, la descomposición funcional y las matrices. Los diseñadores utilizan diagramas de flujo de datos, tablas de estructura y diagramas de acción.

En las técnicas OO, todos utilizan el mismo modelo conceptual; analistas, diseñadores, programadores y, de modo particularmente importante, los usuarios finales, figura 5.1.

La transición del análisis al diseño es tan natural, que a veces es difícil especificar el punto final del análisis y el punto inicial del diseño. Cuando el desarrollo tradicional cruza los muros que se muestran en la figura 5.1, a veces se pierde información y aparecen las incompensaciones. El empleo de un modelo conceptual único en las técnicas OO, conlleva a un concepto única por parte de los involucrados (analistas, diseñadores, programadores e, incluso, usuarios), siendo esta un claro ejemplo, el paradigma de la creación del software se puede resolver a través de las metodologías OO.

El análisis de la estructura de objetos se ocupa de definir las características de objetos en que los asociamos, donde hay que responder las siguientes preguntas:

- ¿ Qué tipos de objetos hay ?
- ¿ Cuáles son sus relaciones y funciones ?

- ¿ Qué subtipos y supertipos son útiles ?
- ¿ Hay algún tipo de objeto compuesto por otros objetos ?

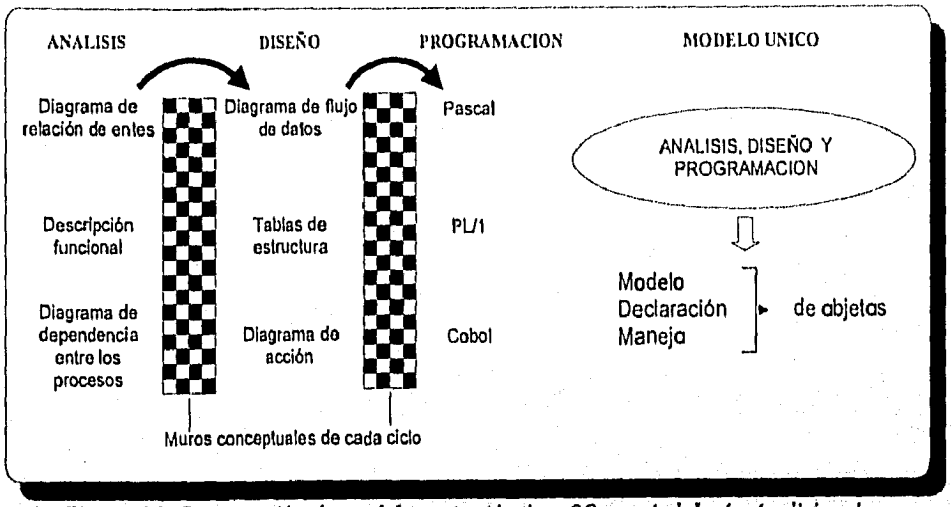


Figura 5.1. Comparación de modelos entre técnicas OO y metodologías tradicionales.

En la etapa del diseña de la estructura de objetos, se identifican las clases (la implantación de tipos de objetos). Se definen las superclases, subclases, rutas de herencia y los métodos a utilizar y se realiza el diseño detallado de las estructuras de datos, en la figura 5.2 se definen las etapas del análisis y diseño OO.

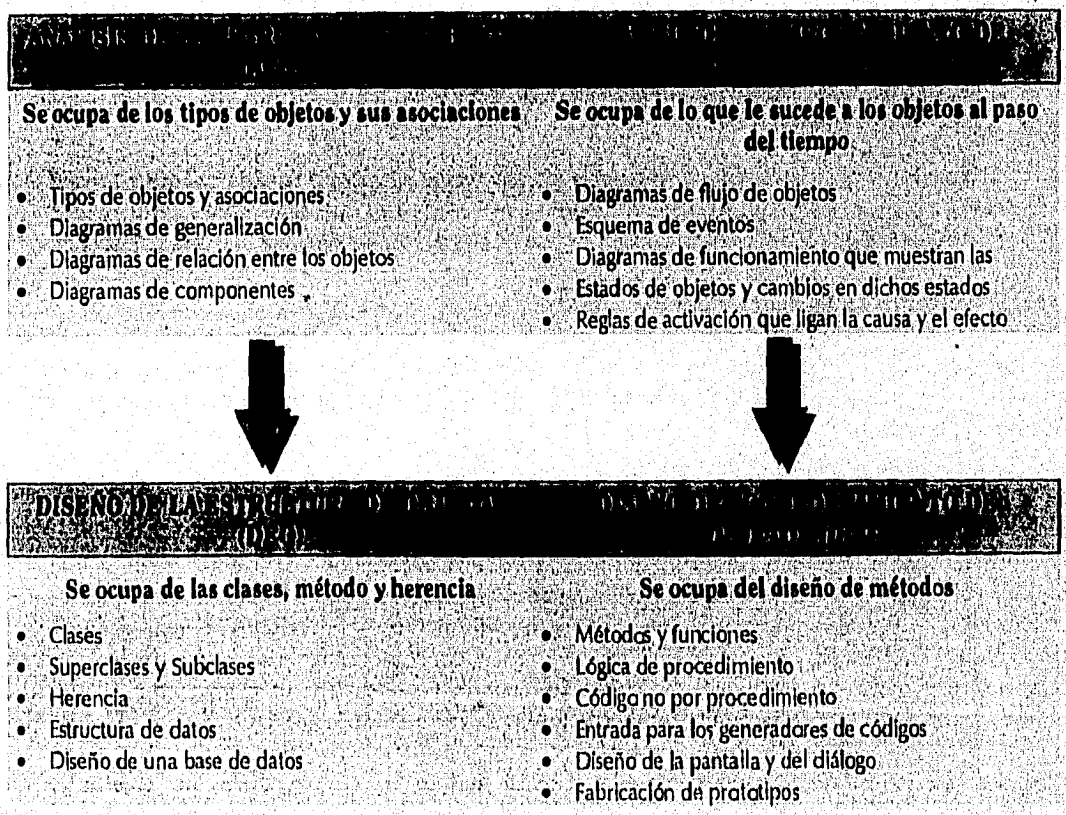


Figura 5.2 Etapas del análisis y diseño OO.

5.1 DEFINICION DEL PROBLEMA.

La Dirección General de Interventoría (DGI), a través de la Dirección de Interventoría en Programas de Trabajo (DIPT), tiene como objetivo principal verificar y garantizar el buen desempeño y actuación de los servidores públicos involucrados en los programas de trabajo dentro de las áreas administrativas, en los rubros de recursos humanos, recursos financieros, recursos materiales y capacitación, en el ámbito de la Subsecretaría de Ingresos de la Secretaría de Hacienda y Crédito Público (S.H.C.P.), teniendo camo objetivos generales:

1 Verificor el buen desarrollo y seguimiento de los progromas de trabajo de las Unidades Administrativas.
2 Verificar el cumplimiento y opego o la ley y normatividad en materia administrativa.
3 Detectar irregularidades que obstoculicen lo consecución de las metas y programas.

El proceso de Interventoría ver figuro 5.3, se realiza en las dependencias adscritas a la Subsecretarío de Ingresos, en el ámbito central, regional y local, onolizondo la información del desarrollo de actividades y consecución de metas de los programas de trabaja comprometidos por la administración en los áreas de :

- Recursos Humanos.
- Recursos Finoncieros.
- Recursos Materiales.
- Capacitoción.

El problema que se tiene con el sistemo manuol es que se pierde el cantrol de las interventorías concluídas, así como el nivel de avance de aquellas que están en proceso, dificultándase la detección de los irregularidodes que se encuentran en los límites del plazo de solventoción.

El control de interventorías concluidos y en proceso, conociendo el grado de avance en bose a los rubros o revisor en coda uno de estos, el levontomiento de constancias de hechos y el control de solventación a los irreguloridades detectadas, es lo que se contemplo automatizar en el presente proyecto.

PROCESO GENERAL

Lo DIPT, llevo a cabo una interventoría, ya sea por solicitud de lo unidad odministrotiva o revisar, par indicación del Director Generol de Interventoría o por cumplimiento al progroma calendarizado.

Se eloboro un oficio dirigida al titular de la unidad administrativo notificando la realización de la Interventorío al área, determinanda. El personol designado paro realizar lo Interventoría acude o las instalaciones de lo unidad administrativa, entrega el oficio de Interventoría al titular de lo dependencia, con el fin de que les sean presentodos los responsables de las áreas administrativas de :

- Recursos Humanos
- Recursos Financieros
- Recursos Materioles
- Capacitación

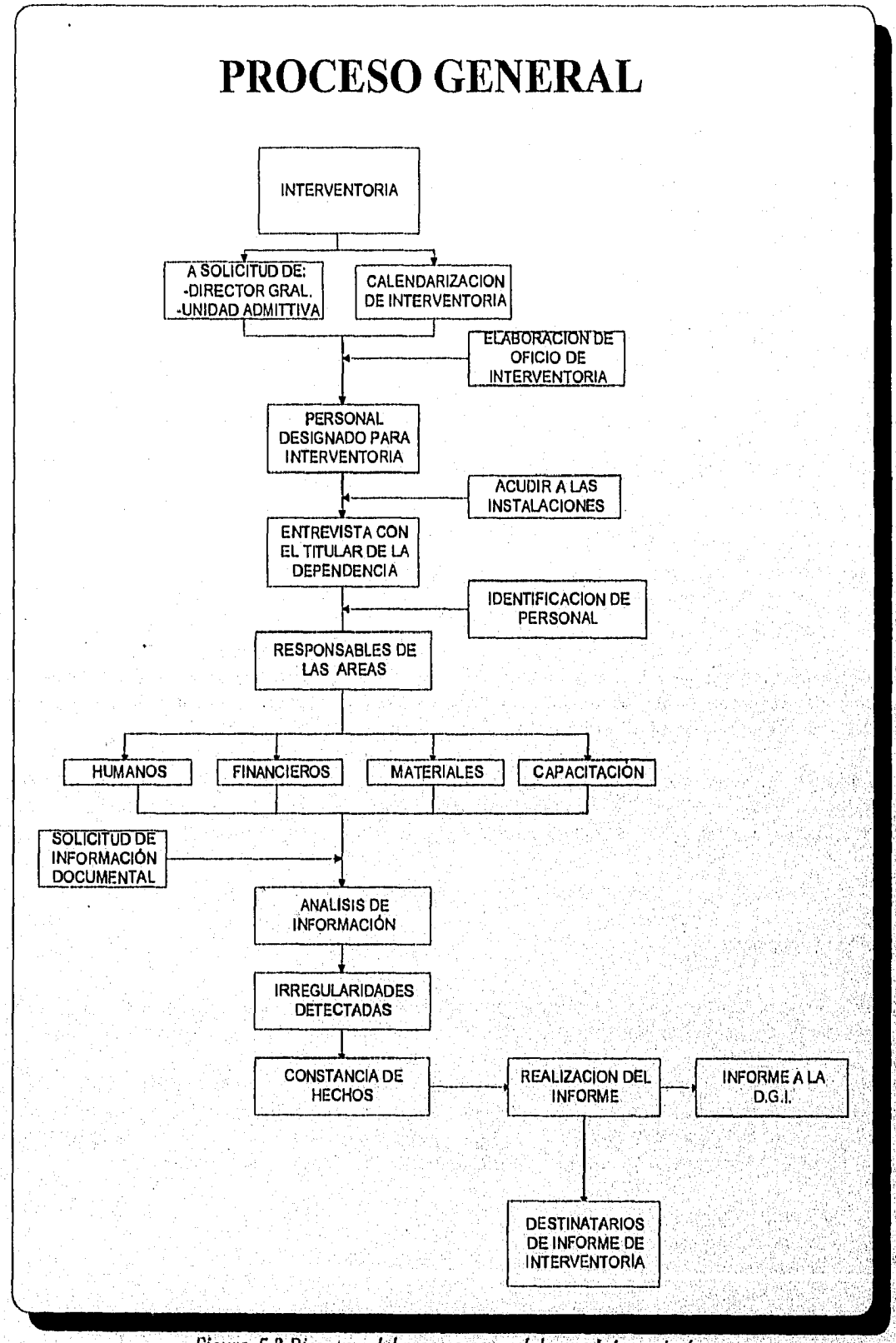


Figura 5.3 Diagrama del proceso general de una Interventoría.

Los revisores solicitan toda la documentación necesaria para verificar el apego y cumplimiento de los programas de trabajo a que se hallan comprometido las áreas administrativas. A los titulares de cada área administrativa se les fincarán las responsabilidades de las irregularidades que sean encontradas y que serán asentadas en una constancia de hechos.

La constancia de hechos es el documento oficial donde se asientan todos los datos de la Interventoría realizada; estado en que se encuentra la unidad administrativa, responsables de cada área administrativa, irregularidades encontradas, las razones y la documentación soporte que exista para justificar dichas irregularidades.

Cada irregularidad registrada tiene un máximo de tres meses para ser solventada; de rebasar el tiempo límite, la Unidad de Contraloría Interna dictamina las sanciones a los servidores públicos que no hallan solventado éstas.

La solventación de cada irregularidad es llevada a cabo por el responsable del área donde se detectaron las irregularidades, quien tienen la obligación de hacer llegar la documentación de las solventaciones a la DIPT.

Los resultados obtenidos en la interventoría son presentados por la DIPT mediante un informe ejecutivo, que junto con la información soporte se envía a los siguientes entidades:

- Subsecretaría de Ingresos.
- INCAFI (Instituto Nacional de Capacitación Fiscal).
- Administradores Regionales.
- Administradores Locales.
- Director General de Interventoría.
- Unidad de Contraloría Interna.

La Unidad de Contraloría Interna determina las sanciones correspondientes a cada irregularidad que no fue solventada dentro del plazo de tres meses.

5.2 FASE DE ANALISIS

5.2.1 ESTRUCTURADO.

La metodología de análisis y diseño estructurado se basa en las técnicas propuestas por Gane Yurdan y Jacksan Sarson.

El análisis y diseño estructurado es una actividad de diseño y construcción de modelos (mapas, diagramas), con una estandarización de notaciones para representarlas; permitiendo reflejar el flujo y contenido de información.

El análisis y diseño estructurado se basa en tener un control de lo que se está realizando, con ayuda de la definición de modelos de diagramación se puede visualizar la información. La utilización de las herramientas CASE logra que resulte más dinámica y con un mejor mantenimiento de la consistencia y la corrección de la información.

En la siguiente tabla se muestra los requerimientos mínimos para las tres etapas críticas del ciclo de vida del software.

REQUERIMIENTOS POR ETAPA

ANALISIS	DISEÑO	CONSTRUCCION
DATOS - DATOS IMPORTANTES DE LA EMPRESA - MODELO DE DATOS	- DISEÑO DE REGISTROS	- VISTA DE DATOS POR PROGRAMA APLICATIVO
ACTIVIDADES - PANORAMA ESTRATEGICO DE SUS METAS. - PROCESOS NECESARIOS Y COMO SE RELACIONAN	- DISEÑO DE PROCESOS	- CODIFICACION DETALLADA DE PROCESOS.
TECNOLOGIA - ESTRATEGIA TECNOLOGICA DEL NEGOCIO - IDENTIFICACION DE LA TECNOLOGIA	- DEFINICION DE HARDWARE Y SOFTWARE BASE.	- APLICACION DEL HARDWARE Y SOFTWARE.

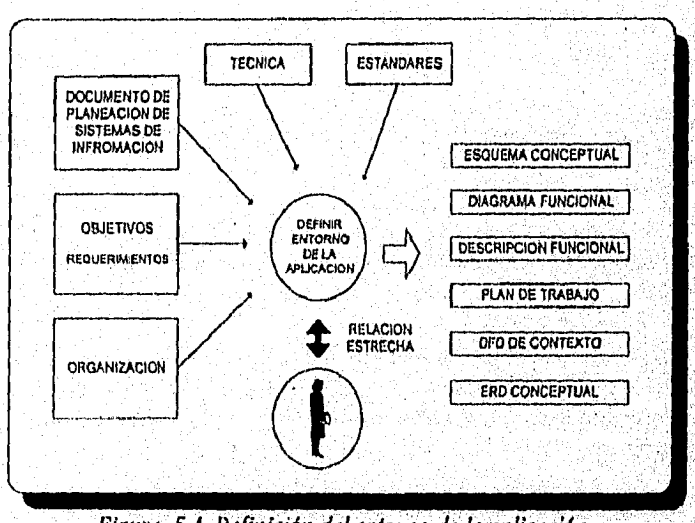


Figura 5.4. Definición del entorno de la aplicación.

ANALISIS DEL AREA

- Identifica los requerimientos de información en el área de negocio.
- Crea un modelo de información que refleje la interrelación entre los datos y las actividades de la empresa.
- Identifica los cambios deseables por el usuario en el modelo existente de información.
- Evalúa la construcción del sistema de infarmación.

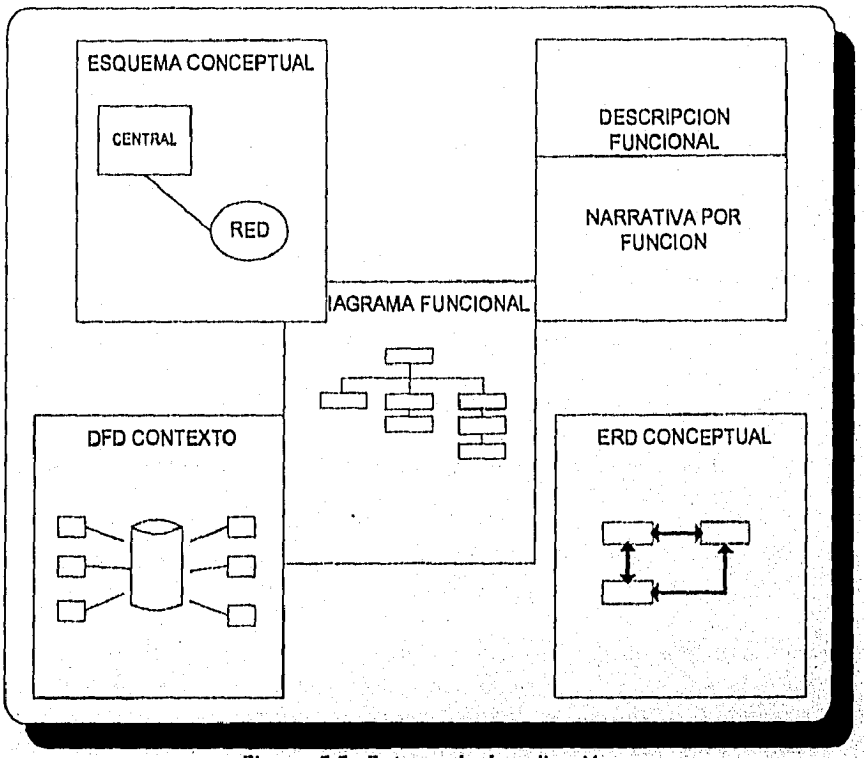


Figura 5.5. Entorno de la aplicación.

Como se puede observor en la figura 5.6, se hoce un recuento de la informoción que se obtiene con la metodalogía de análisis y diseño estructurado.

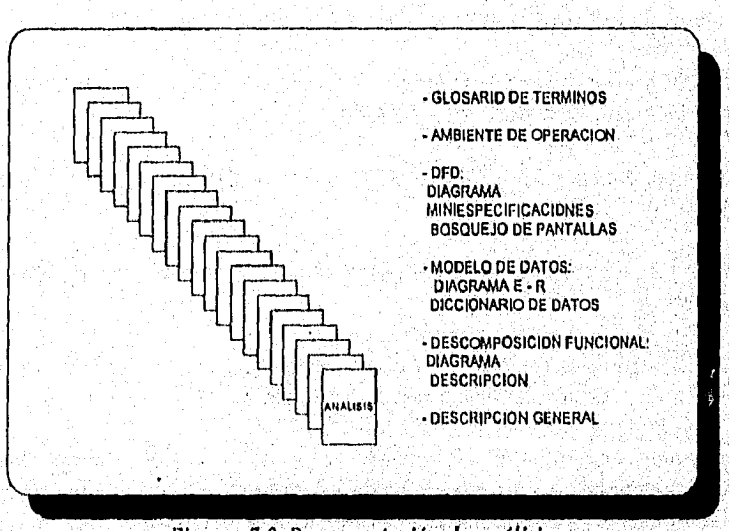


Figura 5.6. Documentación de análisis.

5.2.2 ORIENTADO A OBJETOS.

Como se mostró en la figura 5.2, la fase de análisis OO está compuesta por dos etapas:

I **Análisis de la Estructura de Objetos (AEO).**

II **Análisis del Comportamiento de Objetos (ACO).**

I. ANALISIS DE LA ESTRUCTURA DE OBJETOS (AEO).

Antes de iniciar el análisis OO, partamos de las siguientes definiciones:

OBJETO: Es cualquier cosa (real o abstracta) acerca de la cual almacenamos datos y los métodos que controlan dichos datos.

METODO: Los métodos especifican la forma en que se controlan los datos de un objeto.

TIPO DE OBJETO: Es una categoría de objeto, donde un objeto es una instancia de un tipo de objeto.

Citaremos dos ejemplos para aclarar el concepto:

1 Un tipo de objeto es un servidor público y un objeto podría ser Alberto, Leticio o José que son servidores públicos.

2 Un tipo de objeto es una constancia de hechos y un objeto podría ser la constancia de hechos DGI-21343.

ENCAPSULADO: Es el empacamiento de datos y métodos a los cuales tendrá acceso exclusivamente el mismo objeto. Por lo que ningún objeto podrá tener acceso a los datos de otro objeto. Para utilizar la estructura de otro objeto se tendrá que solicitar a través de mensajes.

El encapsulado es importante, ya que al crear el objeto lo declaramos como una cosa independiente, ya que sólo él puede tener acceso a sus datos y si deseamos modificarlo o probarlo no alteraremos los datos de los demás objetos.

A partir de los conceptos anteriores fundamos el siguiente razonamiento:

"Un objeto es entonces una cosa cuyas propiedades están representadas por tipos de datos y su comportamiento por métodos, que están encapsulados".

MENSAJE: Es una solicitud para que se lleve a cabo la operación indicada por el objeto y se produzca el resultado.

Los objetos pueden estar compuestos por otros objetos muy complejos. Al implantar los tipos de objetos, se les denomina clase.

CLASE: Es una implantación de un tipo de objeto. Especifica una estructura de datos y los métodos operativos permisibles que se aplican a cada uno de sus objetos.

El método se especifica una sola vez en la clase y todos los objetos de la clase lo comparten. Si analizamos las definiciones de las características de los tipos de objetos, tenemos que una clase puede tener subclases, las subclases pueden tener sub-subclases y así sucesivamente. Por tanto, una clase

implanto el tipo de objeto, una subclase hereda propiedades de la clase y así sucesivamente. Una subclase puede heredar la estructura de datos y los métodos o algunos de éstos, de su superclase (padre). También tiene sus métodos e incluso tipos de datos propios.

Una clase puede estar definida con sus estructuras de datos y métodos y, o su vez, heredarle algunos o todos de otra, de su superclase.

En el AEO se definen las categorías de los objetos que percibimos y las formas en que los asociamos, creando los modelos del AEO, los cuales son una guía para la definición de clases y sus estructuras de datos en el diseño de la estructura y comportamiento del objeto. En esta etapa se examina tres tipos básicos de información.

I ¿Qué son los tipos de objetos y cómo se asocian? La identificación de los objetos y sus asociaciones se representan mediante esquemas de objetos. Esta información guía al diseñador en la definición de clases y estructuras de datos.

II ¿Cómo se organizan los tipos de objetos en supertipos y tipos? Las jerarquías de generalización se pueden organizar en diagramas e indicar al diseñador las direcciones de herencia.

III ¿Cuál es la composición de los objetos complejos? Se pueden elaborar diagramas de jerarquías compuestas. La composición guía al diseñador en la definición de mecanismos que controlen adecuadamente a los objetos dentro de otros objetos.

OBJETOS Y TIPOS DE OBJETOS.

En la etapa AEO, el grupo de analistas se preocupa más por identificar los tipos de objetos que por identificar los objetos individuales en un sistema. Los tipos de objetos son importantes, puesto que crean los bloques conceptuales de construcción para el diseño de sistemas. En la programación orientada a objetos, estos bloques de construcción guían al diseñador en la definición de las clases y sus estructuras de datos. Además los tipos de objetos son un índice para los procesos del sistema, por ejemplo, actividades como crear, avance de solventación, concluir o archivar están íntimamente ligadas con el tipo de objeto constancia de hechos, puesto que combian su estado. En otros palabras, un objeto sólo debe ser controlado por medio de las funciones asociadas con su tipo. Así, las operaciones no se pueden definir de manera adecuada sin los tipos de objetos.

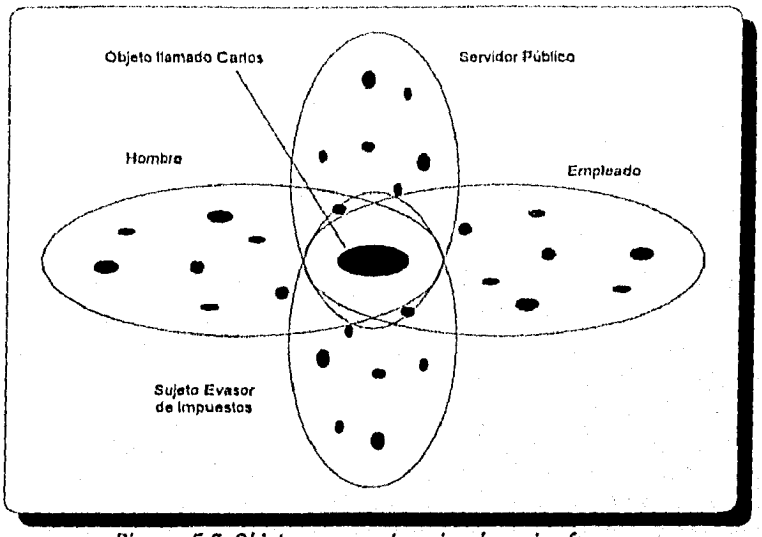


Figura 5.7. Objeto que se categoriza de varias formas.

Los tipos de objetos que definimos y utilizamos son variados, puesto que los elegimos con base en la compresión de nuestro mundo, por lo tanto un objeto se puede categorizar en más de una forma, por ejemplo, una persona puede considerar al objeto Carlos como hombre, la Dirección de programas de trabajo puede considerar al objeto como un servidor público. Su jefe como un empleado. La S.H.C.P determina que Carlos es una instancia del tipo de objeto llamado "sujeto evasor de impuestos", etc. Ver figura 5.7., donde cada elipse indica la colección de objetos a los que se aplica el tipo de objeto hombre.

ASOCIACIONES DE OBJETOS

Es importante modelar la forma como los objetos se asocian entre sí, por ejemplo, la figura 5.8 (a) muestra como los objetos, constancia de hechos, se asocian con los responsables de anomalías (servidores públicos). La figura 5.8 (b), representa otra forma de expresar la asociación entre estos dos tipos de objetos.

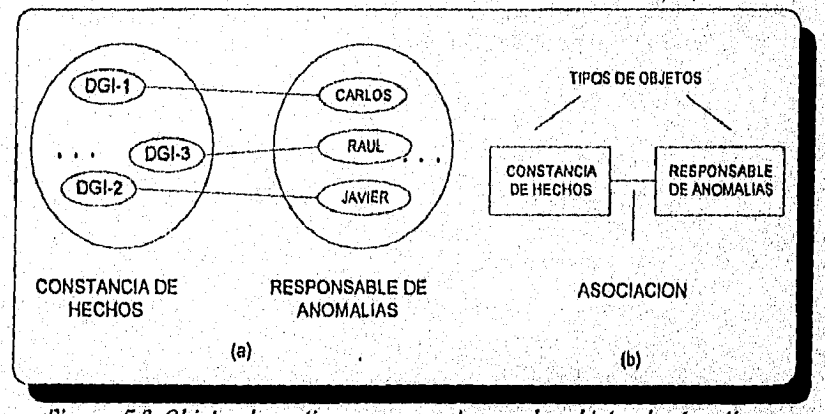


Figura 5.8. Objetos de un tipo que se asocian con los objetos de otros tipos.

Aunque la figura anterior muestra las asociaciones entre dos tipos de objetos, no se indica el significado de la asociación, tampoco se indica la cantidad de objetos con los que un objeto dado puede y debe asociarse. En el análisis, es útil nombrar de alguna forma a las asociaciones e indicar la cantidad de

objetos de un tipo dada que se deben asocior con los objetas de otro tipo, puesto que esta le da significado y aumenta lo comprensión de la asaciación, figura. 5.9.

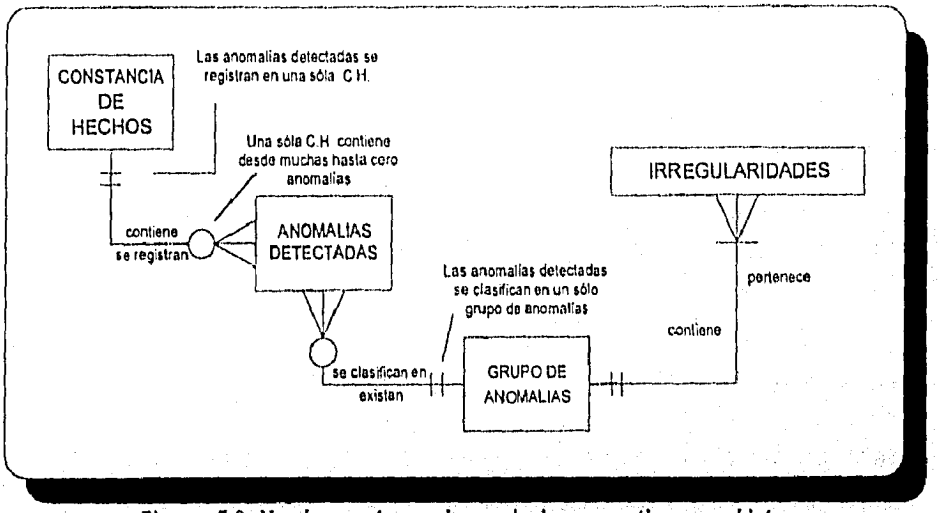


Figura 5.9. Nombre y número de asociaciones que tiene un objeto.

JERARQUIAS DE GENERALIZACION

La generalización es el resultado (o el acto) de distinguir un tipo de objeto como más generol o, incluso, que es más que otro. Todo lo que se oplique o un tipo de objeto tombién se oplico o sus subtipos. Cado instoncio de un tipo de objeto es también uno instoncia de sus supertipos.

Los jerarquíos de generalizoción son importantes poro el desarrollodor OO, porque el uso de supertipos y subtipos proporciona uno herramienta útil poro describir el mundo del sistema de oplicación e indico los direcciones de herencio entre los closes en los lenguajes orientados o objetos, figuro 5.10.

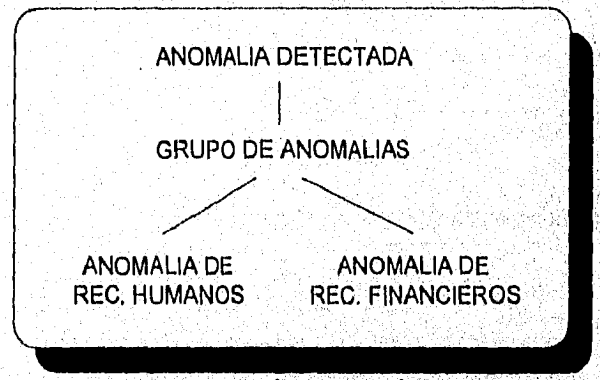


Figura 5.10. Diagrama de jerarquía de generalización.

JERARQUIA COMPUESTA

Codo objeto se puede controlor como un objeto único que conste de otros, los cuoles o su vez, se pueden controlor de formo independiente en coso necesorio. A este tipo de objetos se les llomo objetos complejos.

DIAGRAMAS DE RELACION ENTRE LOS OBJETOS

Los tipos de objetos están relocionodos con otros tipos de objetos. Lo formo de diogromor lo reloción entre objetos es esenciolmente iguol o un diogromo de reloción entre entes utilizado en el onólisis estructurodo, figuro 5.11.

ESQUEMA DE OBJETOS

Lo comprensión de un modelo suele ser más sencillo si los tipos de objetos y relociones se presenton mediante un diagramo de relación entre objetos; los supertipos y subtipos se presenton en un diogromo de jerorquío de generolización y los estructuras compuestas en un diogramo compuesto. Sin embargo, puede ser más útil si se considero presentorlo todo en el mismo diogromo, ol cuol se le llomo Esquema de Objetos.

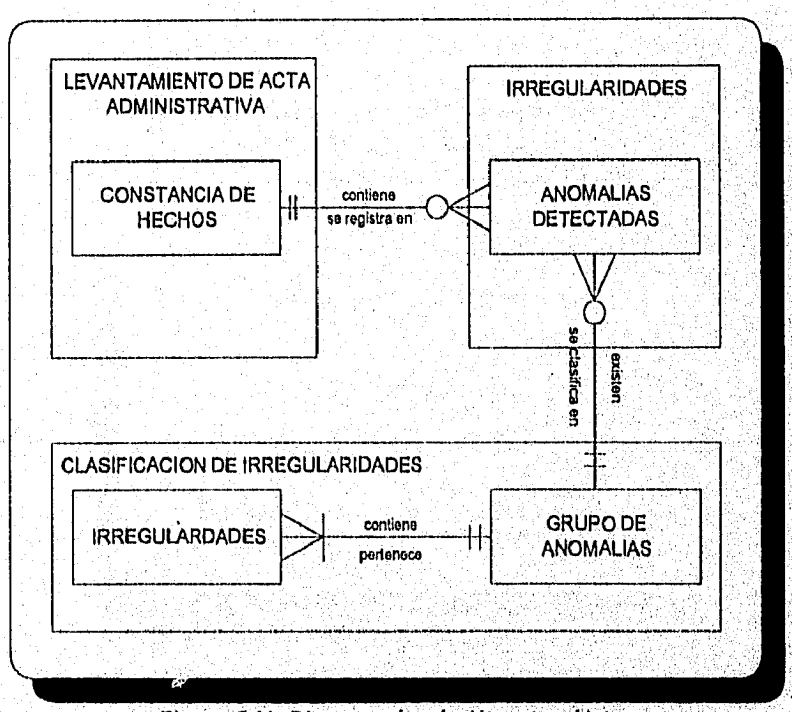


Figura 5.11. Diagrama de relación entre objetos.

II ANALISIS DEL COMPORTAMIENTO DE OBJETOS (ACO).

En el onálisis del comportamiento de objetos, se onalizon eventos que muestron: lo secuencio en que ocurren y cómo los eventos combian el estodo de los objetos. Así, los esquemos de eventos se deben expresor en términos de esquemos de objetos, puesto que los eventos combion el estodo de determinodos tipos de objetos.

En lo fose del ACO se identifico lo siguiente informoción:

- ¿En qué estodos puede estor un objeto? Un objeto puede tener vorios conjuntos de estodos.
- ¿Qué tronsiciones de estodos se pueden dor? Estos se determinon en el diogromo de tronsición de estodo.
- ¿Qué eventos ocurren? Un evento es el combio de estodo de un objeto.
- ¿Qué operociones se llevon o cobo? Se reoliza un esquemo de eventos, poro mostror lo secuencio de operociones y eventos.
- ¿Qué interocciones ocurren entre objetos? Se reoliza un diogromo poro mostror los mensojes tronsferidos entre los clases.
- ¿Cuóles son los reglos de activoción que se reolizon poro reoccionor onte el evento?
- ¿Cómo se representon los operaciones en los métodos? Se utilizon los diogromos, enunciodos declorotivos u otros medios poro determinor los métodos con detolles suficientes como poro generor código.

Estados de un objeto. Un objeto puede existir en varios estodos; por ejemplo, una irreguloridod detectodo puede ser una instancio de olgunos de los siguientes tipos de objetos:

Irregularidad:

- Sin iniciar solventoción (en tiempo permitido)
- En proceso de solventoción (en tiempo permitido)
- En tiempo límite de solventoción (en proceso)
- En tiempo límite de solventoción (sin inicior)
- Sobrepasoron tiempo límite de solventoción
- Solventodos.

Por ejemplo el mismo objeto, irreguloridod detectodo, también puede tener los siguientes estodos relocionodos con su closificoción:

- Irregularidad de Recursos Humanos
- Irreguloridod de Recursos Finoncieros
- Irreguloridod de Recursos Moteriales
- Copocitoción

Entonces, un objeto puede ser ol mismo tiempo una instancio de varios tipos de objeto, por lo que definimos:

El estado de un objeto es la colección de los tipos de objeto que se aplican a él. En otras palabras, es la colección de asociaciones que tiene un objeto.

Eventos: El evento es un cambio en el estado de un objeto. En el análisis OO, el mundo se describe en términos de los objetos y sus estados, así como de los eventos que modifican esos estados. Ver figura 5.12.

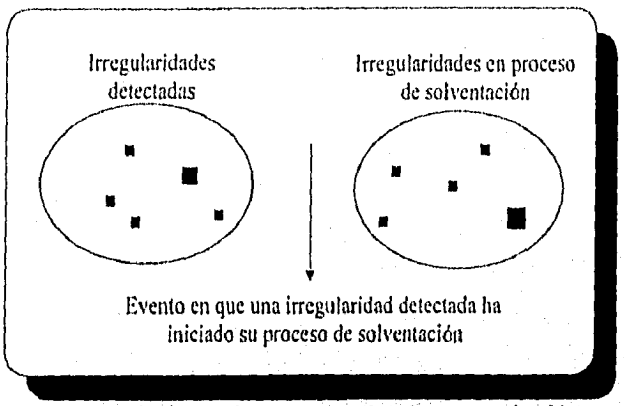


Figura 5.12. Estado de un objeto que cambia de un tipo de objeto a otro.

Tipos de eventos: En el análisis OO no se desea conocer cada evento que ocurra en una organización: sólo los tipos de eventos. Así como hablamos de tipos de objetos e instancias de tipos de objetos, podemos hablar de tipos de eventos e instancias de tipos de eventos. Los tipos de eventos indican los cambios sencillos en el estado de un objeto. Básicamente, los tipos de eventos describen las siguientes formas de cambios de estado:

- **Un objeto se crea:** se crea una constancia de hechos.
- **Un objeto se termina:** Una interventoría a una administración se finaliza.
- **Un objeto se clasifica como una instancia de un tipo de objeto:** Una irregularidad se clasifica en el grupo de irregularidades de Recursos Financieros.
- **Un objeto se clasifica como una instancia de un tipo de objeto:** Un tipo de irregularidad sale del catálogo de irregularidades.
- **Un objeto cambia de clasificación:** Una irregularidad detectada pasa al proceso de solventación.
- **El atributo de un objeto se cambia.**

Una operación hace que los eventos ocurran. Dibujamos la operación como un cuadro con esquemas redondeados, puesto que los eventos se representan como triángulos negros llenos, generalmente unidos a la caja de operación. Ver figura 5.13.

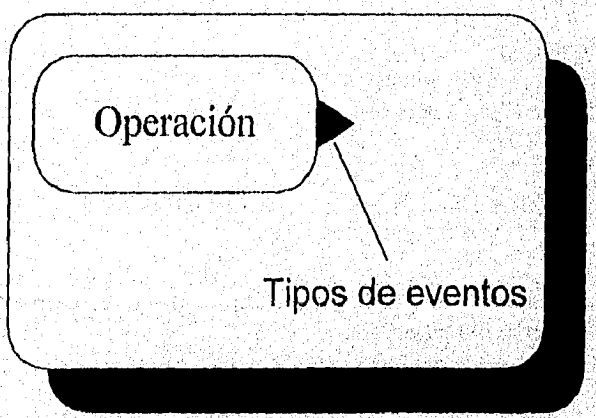


Figura 5.13. Esquema de eventos.

El ciclo de un objeto: La mayoría de los objetos tienen un ciclo vital en el que una sucesión de eventos puede ocurrirle y cada uno de éstos modifica su estado.

En el Análisis Orientado a Objetos se dibuja un diagrama de transición entre los estados, que muestre el ciclo vital de un objeto. Ver figura 5.14.

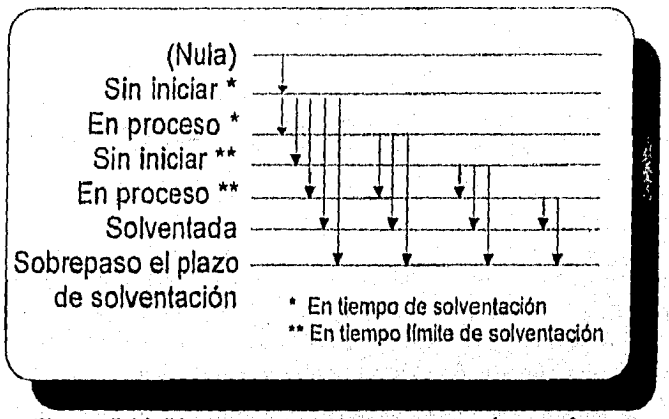


Figura 5.14. Diagrama transición que muestra los estados del objeto irregularidad.

INTERACCIONES ENTRE TIPOS DE OBJETOS

Las diagramas de transición de estados son útiles para expresar el ciclo vital de un objeto particular. Sin embargo, la mayoría de objetos cambian de estado y pueden solicitar a otros objetos que cambien su estado en el proceso, figura 5.15.

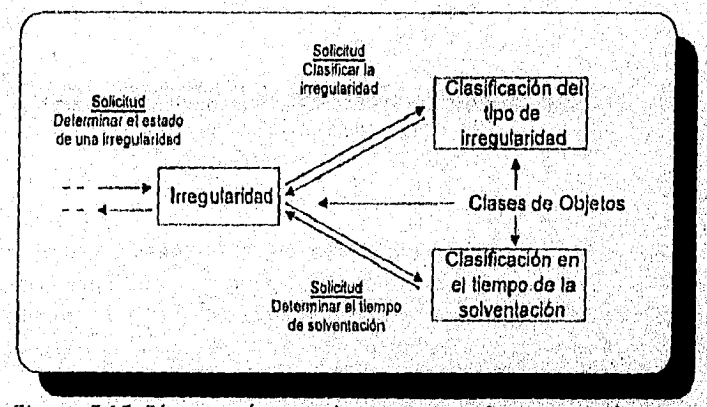


Figura 5.15. Diagrama de mensajes que se transfieren entre clases.

Los eventos son cambios de estado a los que un sistema debe conocer y reaccionar ante ellos de algún modo. Muchas de las operaciones suelen ser externas al sistema; en estos casos, el símbolo es como lo muestra la figura 5.16 (a).

Un reloj externo es una forma particular de fuente externa, indica que un proceso externo emitirá señales de reloj con cierta frecuencia predeterminada. La figura 5.16 (b), muestra el símbolo que lo representa.

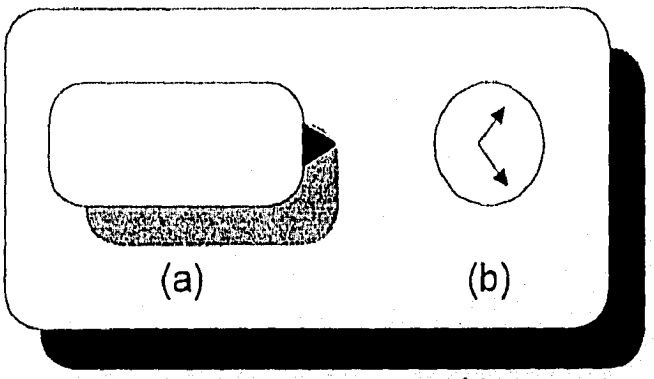


Figura 5.16. Operaciones externas y reloj externo.

Las reglas de activación definen la relación entre la causa y el efecto. Siempre que acurra un evento de cierto tipo, la regla de activación invoca una operación ya definida. Sin embargo, antes de invocar de hecho a la operación, se verifica su condición de control. Si los resultados de evaluación de la condición son verdaderos, se invoca su operación si son falsas no se invoca, ver figura 5.17.

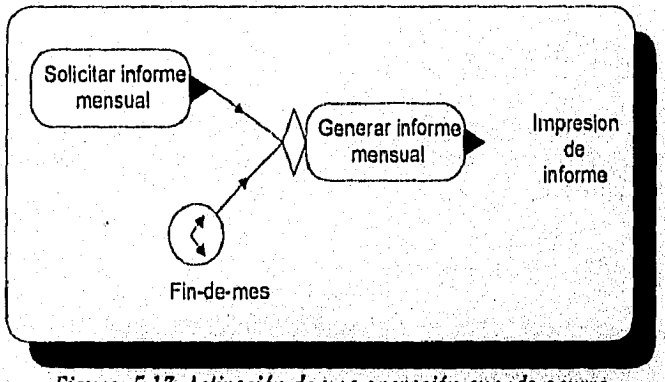


Figura 5.17. Activación de una operación cuando ocurre cualquier evento que lo solicite.

La palabra partición implica que alga se divide en subconjuntos ajenos. En la técnicas OO, ese "alga" se llama supertipo, las particiones (subtipos) no son operaciones independientes que caardinan las condiciones de bifurcación para las formas de activación ajenas, sino que indican los objetivos y distintos subjetivas de las operaciones a las que estón asociadas. La figura 5.18. incisa (a), combina las formas de representarse en los incisos (b) y (c).

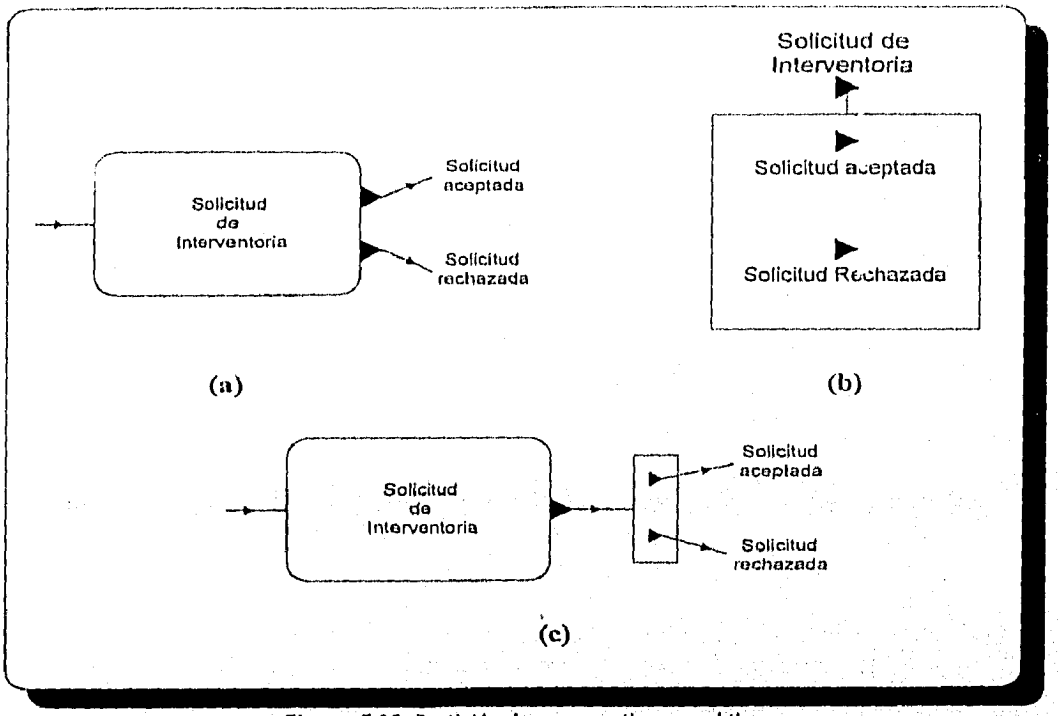


Figura 5.18. Partición de un supertipo en subtipos.

DIAGRAMAS DE FLUJO DE OBJETOS.

Los esquemas de eventos son adecuados para la descripción de procesos en términos de eventos, de reglas de activación, de condiciones o de operaciones. Sin embargo, podría no ser adecuado expresar así procesos grandes y complejos. En ocasiones una área del sistema es demasiado vasta e intrincada como para expresar la dinámica de los eventos y las formas de la activación. En situaciones como esta es útil un Diagrama de Flujo de Objetos (DFO).

Los DFO son parecidos a los Diagramas de Flujo de Datos (DFD) puesto que muestran las actividades que interactúan con otras. En los DFD, una interfaz transfiere datos. En las técnicas OO, no se limitan a esto, sino que el diagrama debe representar cualquier tipo de cosa que se transfiera de una actividad a otra, ya sean pedidos, partes, artículos terminados, diseños, servicios, hardwore, software o datos. En síntesis, el DFO indica los objetos que se producen y las actividades que los producen e intercambian.

Con el fin de ejemplificar los diagramas DFO, se analizará una situación diferente en lo que se refiere al proyecto planteado, ver figura 5.19, a efecto de comparar los diagramas analizados en el presente sección, ver figura 5.20.

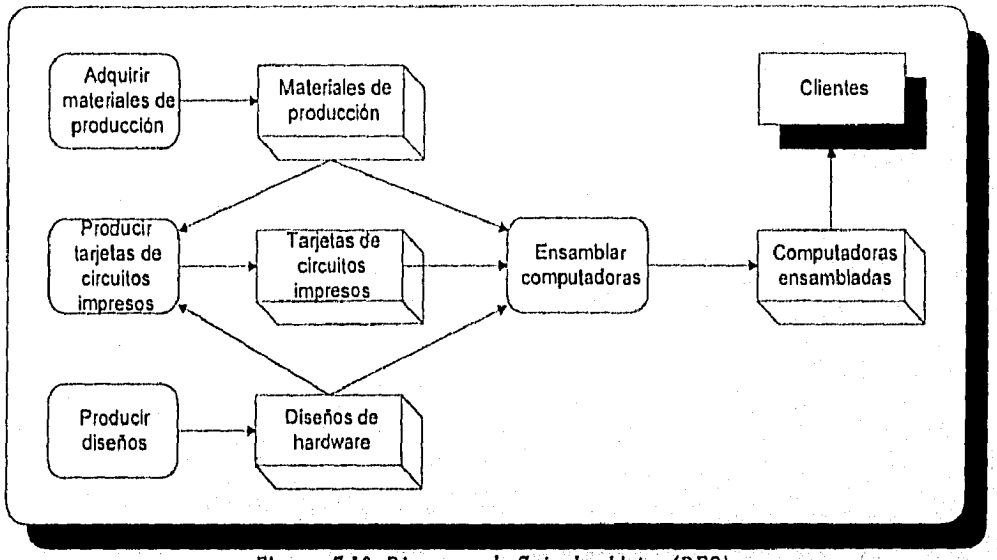


Figura 5.19. Diagrama de flujo de objetos (DFO).

Al observar un Diagrama de Flujo de Datos (DFD), se reconoce enseguida las cajas de actividades, con esquinas redandeadas; las cajas de agentes externos con sombreado y la dirección como líneas de flujo. Sin embargo, no está presente el símbolo de almacenamiento de datos, en su lugar se utiliza una caja tridimensional que indica que el DFO representa el hecho de que los objetos de la vida real fluyen entre las actividades.

Para expresar un proceso de manera más rigurosa y poder generar un código adecuado es necesario un esquema de eventos. Un diagrama de flujo de objetos es útil para representar las estructuras básicas de control y el flujo de procesamiento, cuando no se entiende totalmente la dinámica de los eventos y las claves de activación.

En resumen, el onálisis del comportamiento de objetos puede utilizar una gran voriedad de puntos de vista de diagramación como se muestran en la figuro 5.20.

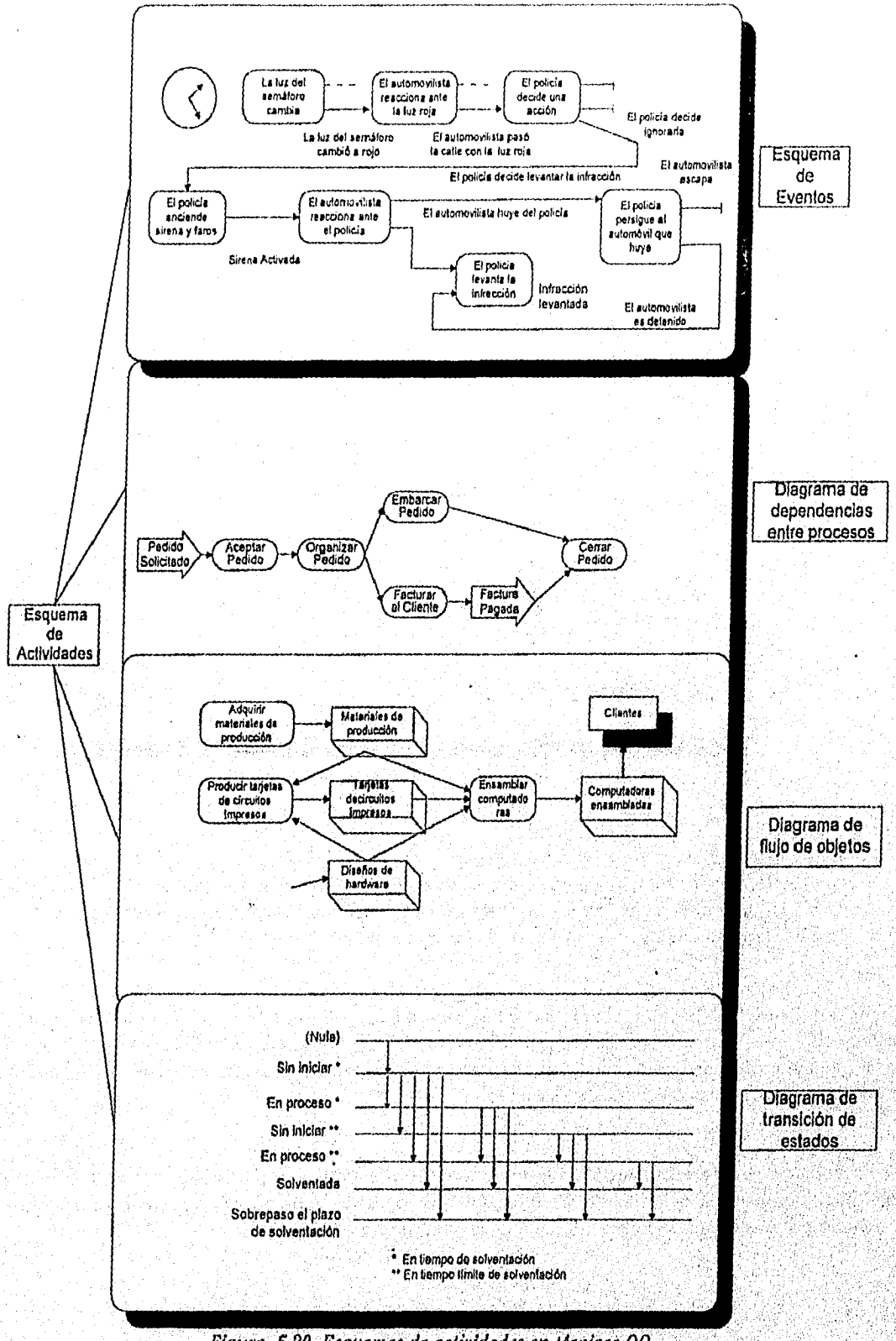


Figura 5.20. Esquemas de actividades en técnicas OO.

5.3 FASE DE DISEÑO

5.3.1 ESTRUCTURADO

Las técnicas de la metodología del diseño estructurado se basan en los pasos ilustrados en la siguiente figura.

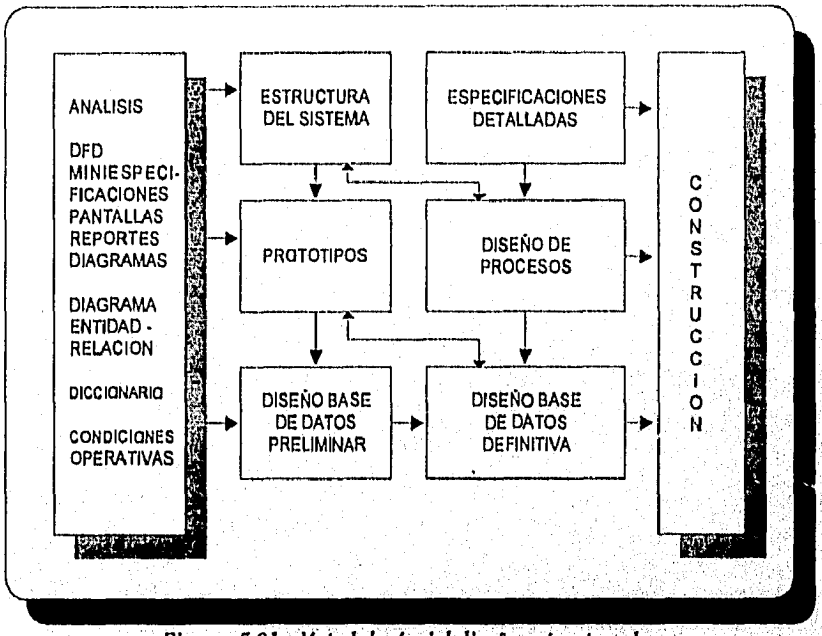


Figura 5.21. Metodología del diseño estructurado.

- Define como los procesos y datos seleccionados en el análisis, pueden ser automatizados.
- Transforman los requerimientos del análisis en especificaciones detalladas de la aplicación.
- El diseño deberá satisfacer los requerimientos del usuario apoyada en prototipos.
- Se determina el ambiente operativo de las aplicaciones.

Como se observa en la figura 5.21, las datos resultantes del análisis son la entrada para el diseño, información que al conjuntarse con una herramienta CASE resulta más tangible para el usuario, al decir más tangible es porque en la actualidad éstas producen prototipos desde la etapa de diseño y el usuario empieza a visualizar los resultados más rápido.

En la definición de la estructura del sistema, se debe procurar que el usuario realmente participe con nosotros, lograr una relación fuerte. Es en esta etapa donde el usuario es quien diseña su sistema y al ser diseñada por él se evitará la etapa de mantenimiento, figura 5.22.

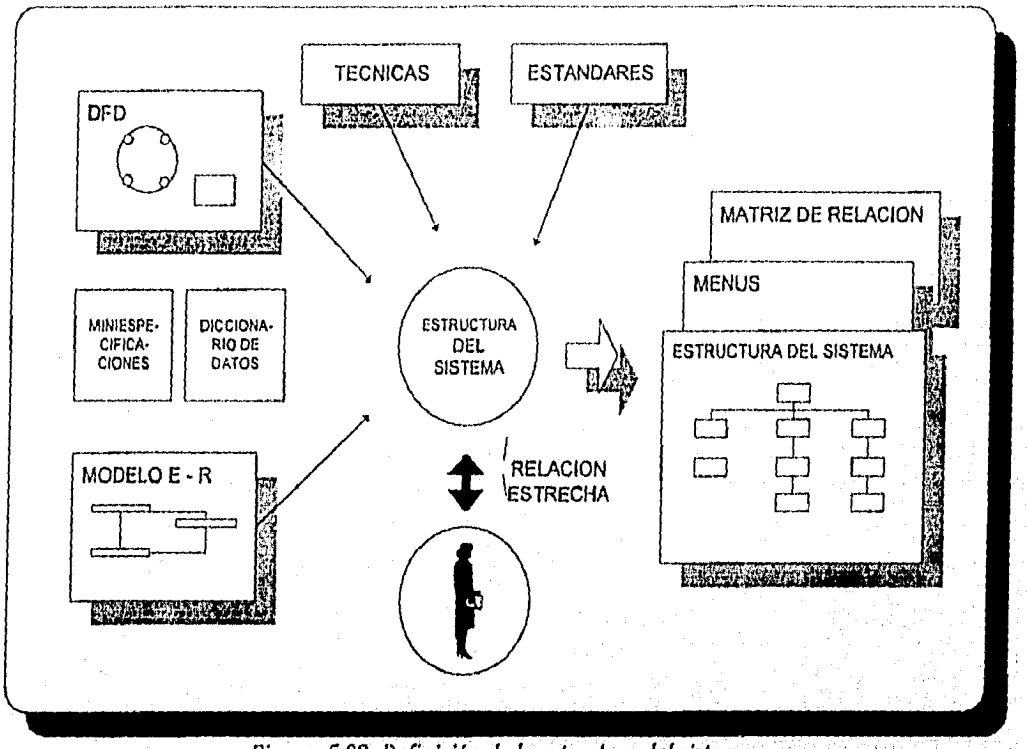


Figura 5.22. Definición de la estructura del sistema.

En cuanto al diseño de la base de datos la participación del usuario es esporádica ya que sólo se le consultará para la definición de algunas entidades, figura 5.23.

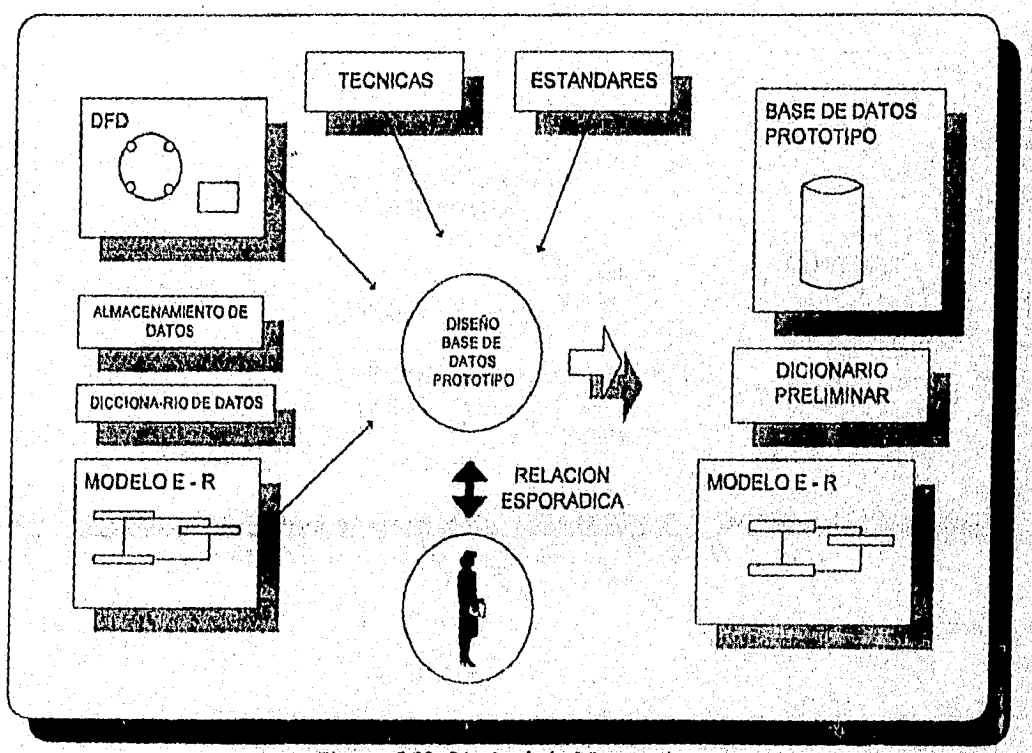


Figura 5.23. Diseño de la BD prototipo.

PROTOTIPOS

Propósitos:

- Tener mayor iteración con el usuario
- Examinar alternativas más realistas
- Desarrollo completo del sistema

Herramientas:

- Lenguajes de 4a. generación
- Sistema administrador de BD.
- Generador de código y aplicaciones.

Con los prototipos, el usuario puede ver realmente como nace su sistema y no como tradicionalmente se hacía, donde el usuario participa en el inicio y final del ciclo de vida de su sistema, figura 5.24.

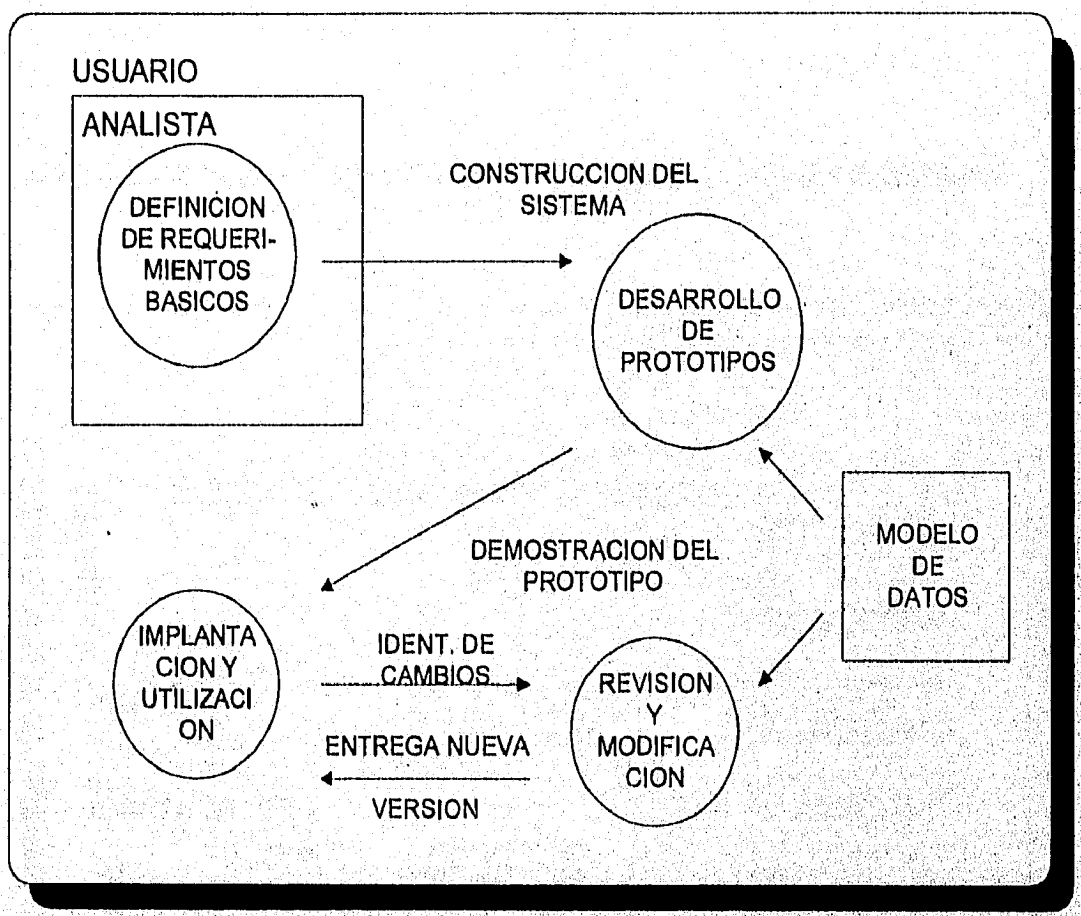


Figura 5.24. Relación de prototipos en el ciclo de vida del software.

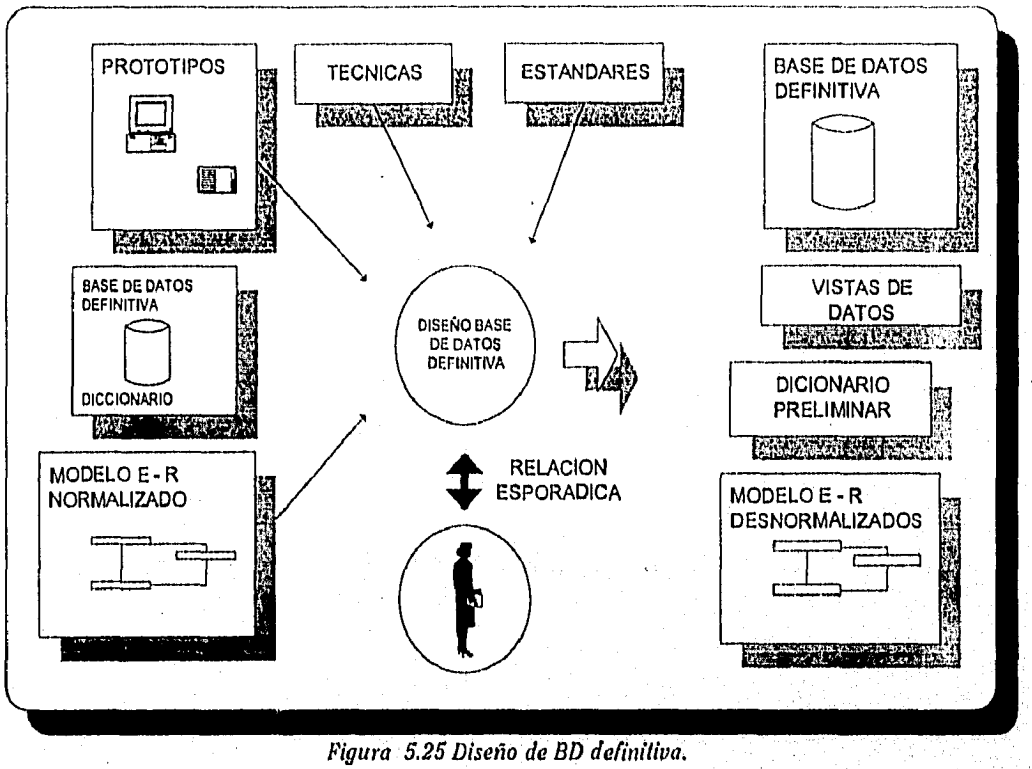


Figura 5.25 Diseño de BD definitiva.

Como se muestra en la figura 5.25, ya se tiene definida la BD definitiva. Con la ayuda de los prototipos se logra definir rápidamente la construcción de vistas donde el usuario logra visualizar su sistema.

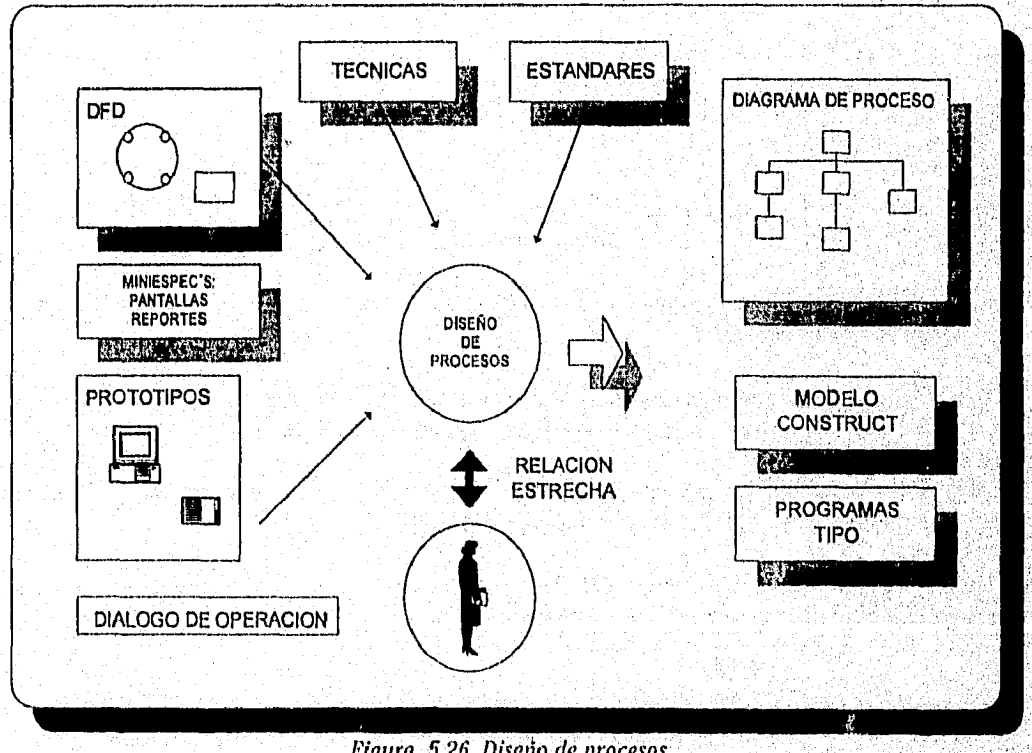


Figura 5.26. Diseño de procesos.

Para el diseño de procesos se necesita la opinión del usuario constantemente y pasar a la construcción física del sistema. Como puede observarse en la figura 5.26, los datos resultantes de esta fase son para la generación de código de programación.

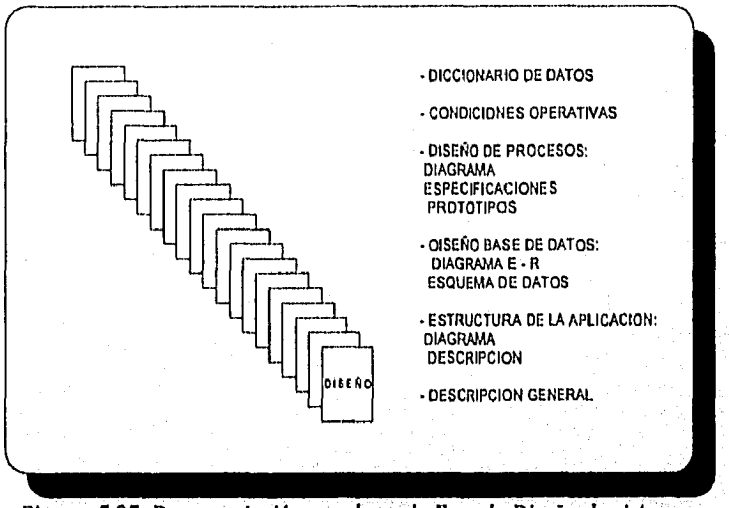


Figura 5.27. Documentación creada en la Fase de Diseño de sistemas.

5.3.2 ORIENTADO A OBJETOS

En el diseño OO se contemplan dos aspectos: el diseño de la estructura de objetos (DEO) y el diseño del comportamiento de objetos (DCO). Los lenguajes de programación OO tienen estructuras de datos y métodos, ambos sujetos a herencia y combinados en unidades llamadas clases. Por esto el DEO y el DCO están entrelazados. En el diseño de la estructura y comportamiento de objetos se identifican los siguientes componentes:

1 ¿Qué clases se implementarán? Los tipos de objetos en el AEO serán la guía en esta decisión.

2 ¿Qué estructura de datos utilizará cada clase? Se puede hacer un diagrama para representar la estructura de datos.

3 ¿Qué operaciones ofrecerá cada clase y cuáles serán sus métodos? Se enumeran las operaciones y se especifican sus métodos en determinado momento.

4 ¿Cómo se implantará la herencia de clases y como afectará ésta las especificaciones de los datos y las operaciones?

5 ¿Cuáles son las variantes? Se identifican las probables variantes de las clases ("IGUAL QUE, EXCEPTO"... se aplica a la mayoría de los componentes reutilizables)

CLASE: Es la implantación de un tipo de objeto. Especifica la estructura de datos y los métodos operativos permitidos que se aplican a cada uno de sus objetos.

En el análisis de estructura de objetos (AEO), identificamos los tipos de objetos; en el DEO nos centramos en la implantación de estos tipos de objetos.

La clase especifica la estructura de datos de cada uno de sus objetos y las operaciones que se utilizan para tener acceso a los objetos. La especificación de cómo se lleva a cabo las especificaciones de una clase se llama método. Los objetos se pueden utilizar exclusivamente con métodos específicos.

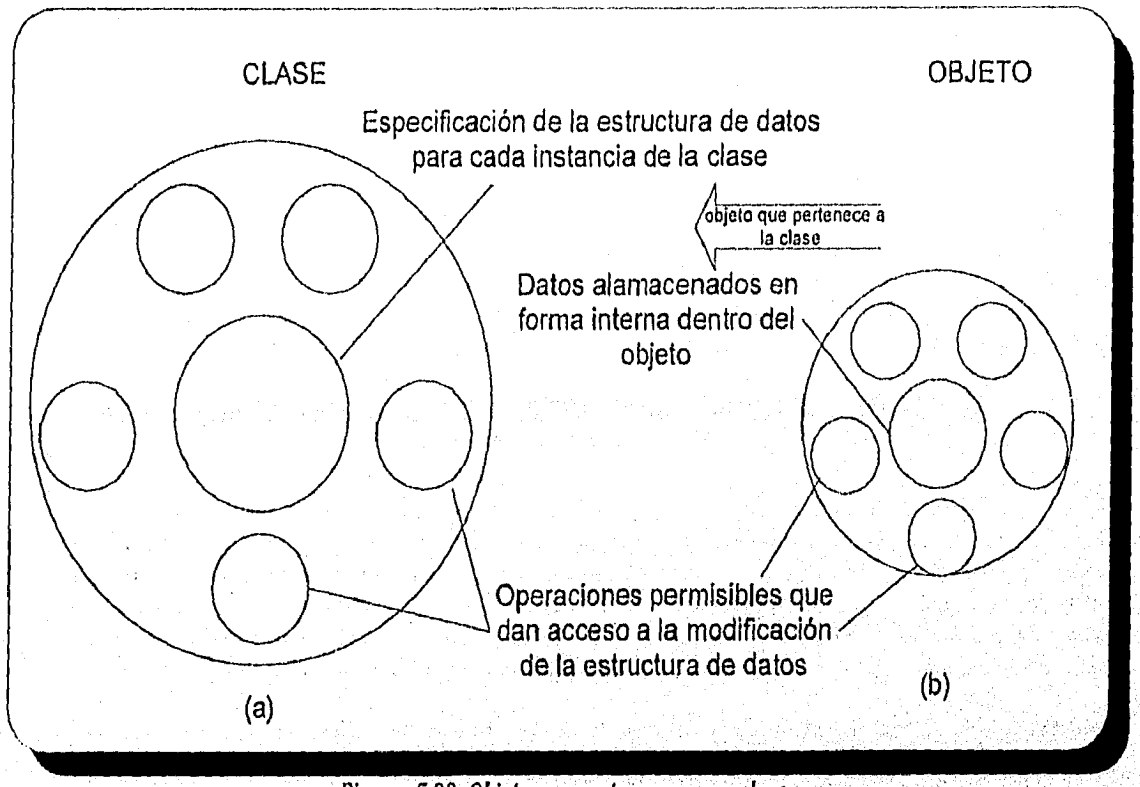


Figura 5.28. Objeto que pertenece a una clase.

La figura 5.28 (b), muestra un objeto que es instancia de la clase de la figura 5.28 (a). Los datos y operaciones que encapsula quedan especificados por su clase. Los datos del objeto se almacenan dentro de él y se tiene acceso a ellos, y se les modifica sólo mediante las operaciones permisibles. Esta restricción al acceso se debe al encapsulado. El encapsulado protege los datos del uso arbitrario o no pretendido. El acceso a la actualización directa de los datos de un objeto por parte del usuario violaría el encapsulado.

Los usuarios observan el "comportamiento" del objeto en términos de las operaciones que se pueden aplicar a los objetos, así como los resultados de tales operaciones. Estas operaciones forman la interfaz del objeto con sus usuarios.

La diferencia entre operación y método, radica en que una operación es un proceso que se puede solicitar como unidad y un método es la especialización de una operación, es decir, la operación es el tipo de servicio solicitado, y el método es su código de programación.

Por ejemplo, una operación asociada con la clase constancia de hechos sería aquella que determine el estado en que se encuentran las irregularidades asentadas (en tiempo de inicio, medio, máximo, de caducidad o solventada). El método especificaría la forma de determinar el estado de cada irregularidad. Para esto, el método podría determinar el estado de cada irregularidad al enviar una solicitud a los objetos estado de irregularidad asociado.

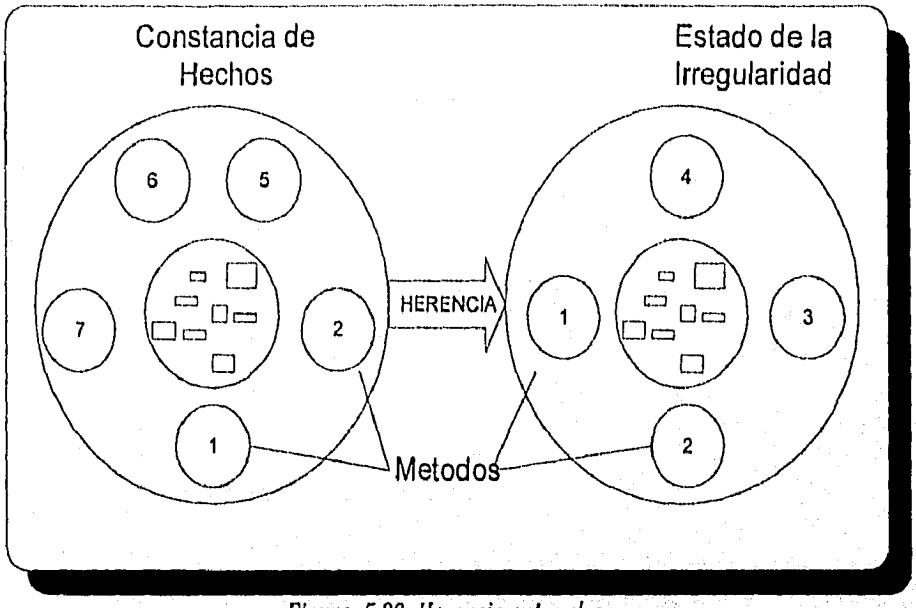


Figura 5.29. Herencia entre clases.

Como se muestra en la figura 5.29, la clase Estado de irregularidad, hereda los métodos 1 y 2 de la clase constancia de hechos, además de tener sus propios métodos 3 y 4. Los métodos de una clase controlan solamente a los objetos de esa clase. No pueden tener acceso directo a las estructuras de datos de un objeto en una clase distinta. Para utilizar las estructuras de datos de una clase diferente, se debe enviar una solicitud a ese objeto, por lo que recordemos que, encapsulada significa que los métodos de una clase protegen a todos los objetos que se encuentran dentro de esa clase.

HERENCIA DE CLASE: La generalización es una noción conceptual. La herencia de clase (que se conoce por herencia) es una implantación de la generalización. La generalización establece que las propiedades de un tipo se aplican a sus subtipos. La herencia de clase hace que la estructura de datos u operaciones sean disponibles para su reutilización por parte de sus subclases. La herencia de las operaciones de una superclase permite que las clases compartan el código (en lugar de volver a definirlo). La herencia de estructura de datos permite la reutilización de la estructura.

HERENCIA MULTIPLE: Se da cuando una clase puede heredar la estructura de datos y operacianes de más de una superclase.

SELECCION DEL METODO: Cuando se envía una solicitud a un objeto, el software selecciona los métodos por utilizar. Ya hemos comentado que el método no se almacena "en el objeto", puesto que esto causaría réplica múltiple. En vez de esto, el método se asocia con la clase. El método no puede estar en la clase de la el objeto es una instancia, sino en una superclase. De esta forma, los usuarios sólo deben especificar lo que se debe hacer, dejando que sea el mecanismo de selección el que determine la forma de localizar y la ejecute.

POLIMORFISMO. Uno de los objetivos principales de las técnicas OO es utilizar otra vez el código. Sin embargo, algunas de las operaciones requieren adaptación para resolver necesidades particulares. Este fenómena se conoce como polimorfismo. La palabra polimorfismo se aplica a una operación que adopta varias formas de implementación. Una de las ventajas del polimorfismo es que se puede hacer una solicitud de una operación sin conocer el método que debe ser llamado. Estos detalles de la implementación quedan ocultos para el usuario; la responsabilidad descansa en el mecanismo de selección de la implantación OO.

IGUAL QUE, EXCEPTO...:La reutilización práctica requiere en su mayoría que el implantador modifique el componente reutilizable. Un arquitecto toma el diseño estándar de un baño y lo modifica para incluir mármol rosa y añade los muebles que lo compondrán. La frase "Igual que, excepto..." describe la mayor parte de la reutilización.

Las técnicas OO deben permitir la adaptación de las clases, se debe permitir tomar una clase y adaptarla a las necesidades particulares; sin embargo, siempre hay que tener presente que las clases se pueden volver muy complejas, por lo que se deben diseñar de un modo que se puedan adaptar con facilidad.

La figura 5.30 muestra un ejemplo sencillo que permite al diseñador redefinir el uso de las propiedades heredadas de un objeto.

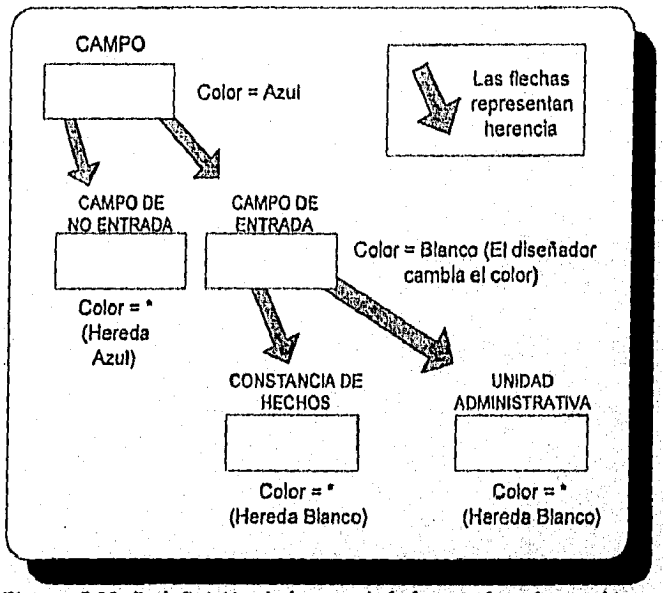


Figura 5.30. Redefinición de las propiedades que hereda un objeto.

Este tipo de adaptación es sencilla para el usuario. Las aplicaciones adaptadas se pueden construir con rapidez, sin preocuparse por el código de una clase. Los diseñadores deben prever los aspectos de un diseño que los usuarios desearán modificar, así como las medias sencillas para la adaptación. El hecho de lograr una máxima reutilización es parte esencial de las metodologías OO en el desarrollo de sistemas.

El analista o el diseñador que genere una clase debe preguntarse: ¿Cómo se utilizará esta clase en el futuro? Debe crear la clase de forma que se pueda adaptar con facilidad a las necesidades futuras. En un ambiente OO bien administrado, todo se construye a partir de clases ya existentes o se crean nuevas clases que serán utilizadas de nuevo en el futuro. Todo se relaciona con el reluso en el pasado o en el futuro.

ESTANDARES DE DIAGRAMACION: El análisis OO requiere diagramas precisos, en el presente trabajo se utiliza un conjunto estándar de diagramación, la figura 5.31 resume los símbolos de uso común en los diagramas que utilizan técnicas estructuradas. Estos símbolos se utilizan en muy diferentes tipos de diagramas, entre los que se encuentran los diagramas de relación entre entes, de descomposición, de dependencia, de flujo de datos, árboles de decisión, diagramas de transición de estado, de diseño de diálogo, de análisis de los datos y diagramas de acción.

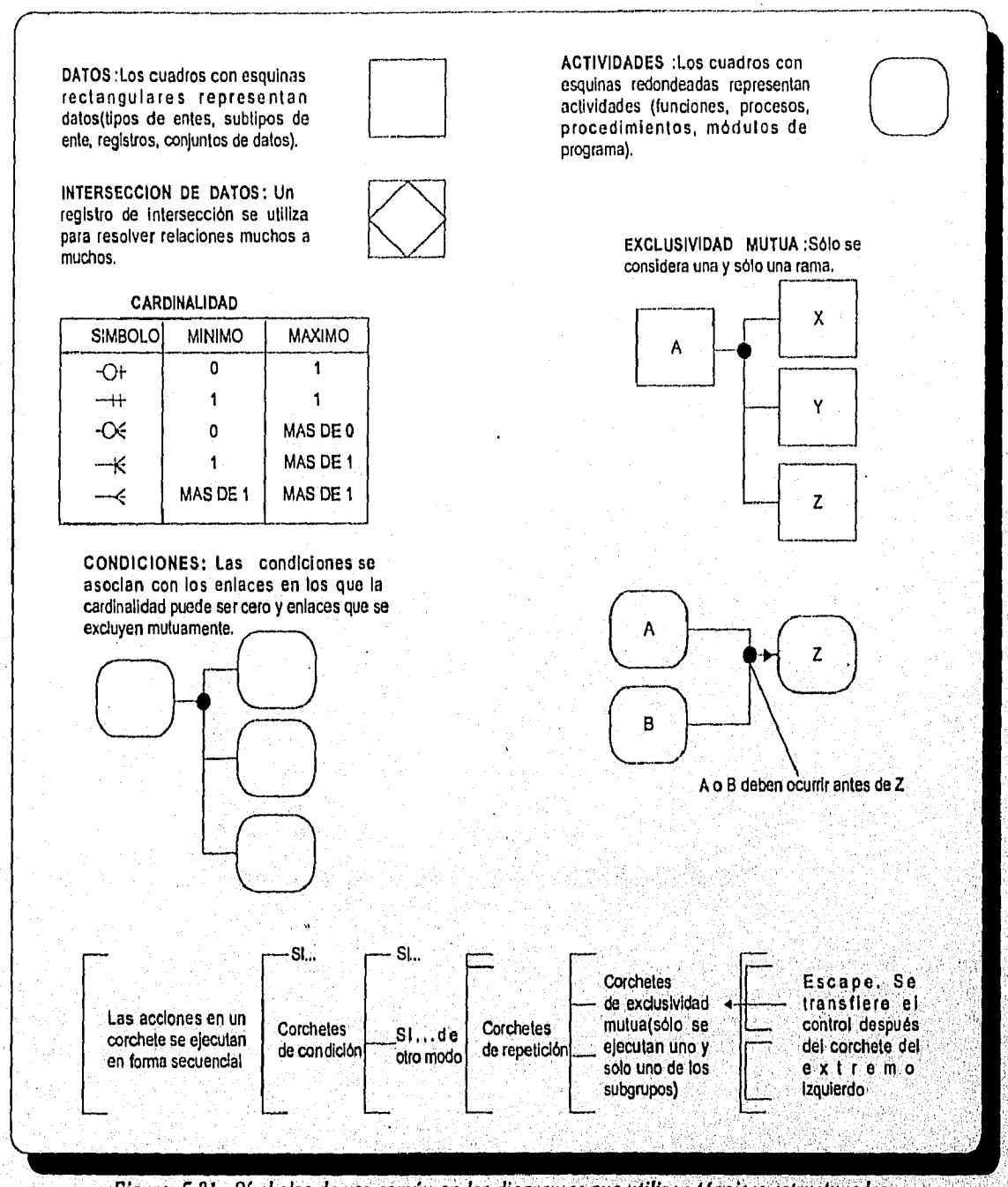


Figura 5.31. Símbolos de uso común en los diagramas que utilizan técnicas estructuradas.

La figura 5.32 resume los símbolos que se utilizan para el análisis y diseño OO y que amplía el conjunto de símbolos de la figura 5.31.

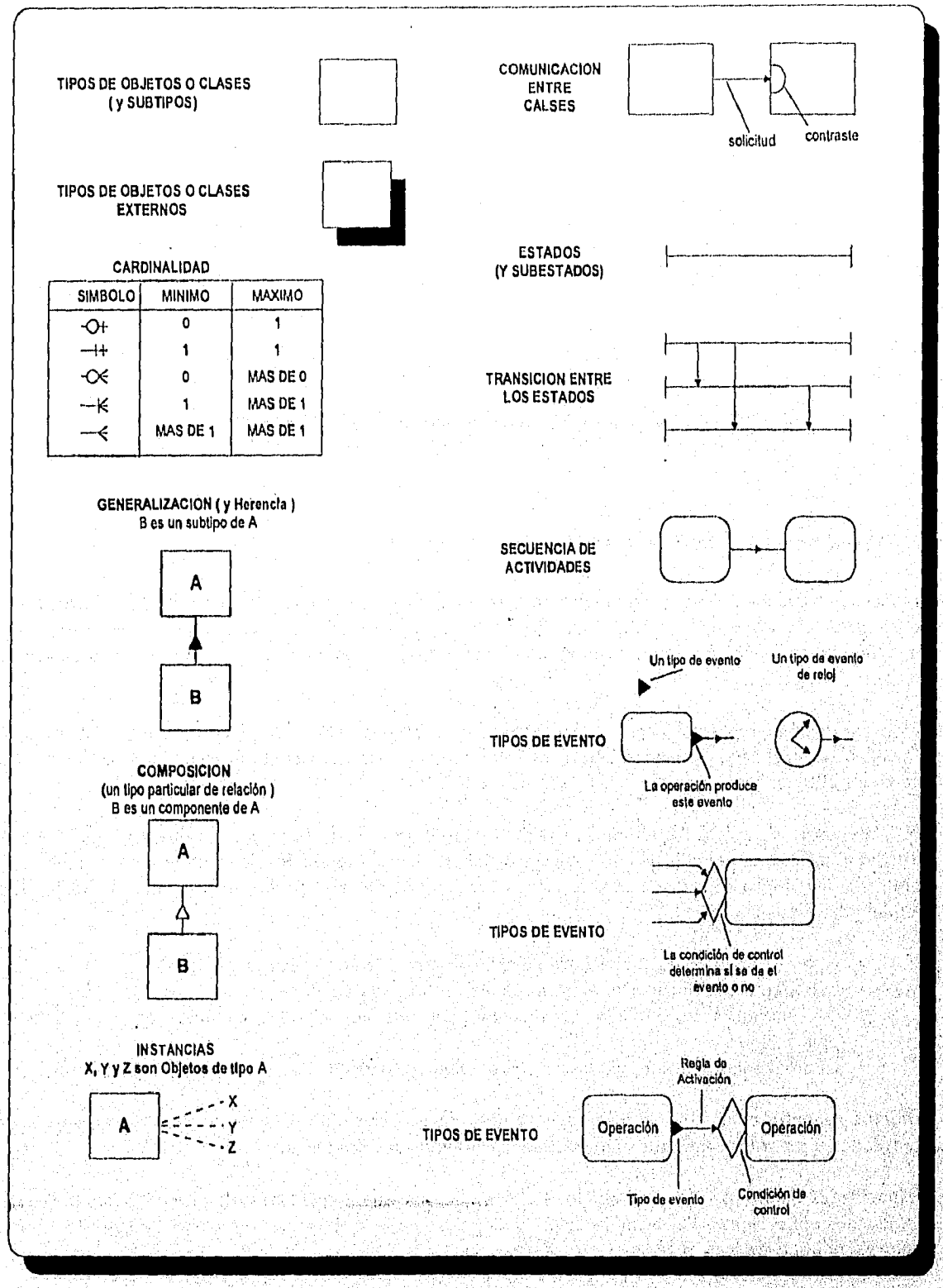


Figura 5.32. Símbolos de uso común en los diagramas que utilizan técnicas OO (Continuación).

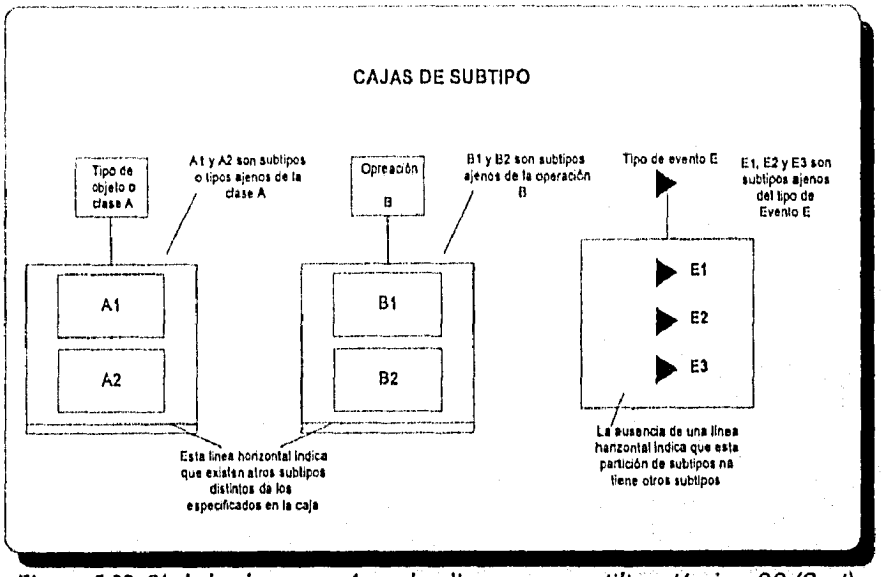


Figura 5.33. Símbolos de uso común en los diagramas que utilizan técnicas OO (Cont).

ESTANDARES PARA LA INTERACCION DE OBJETOS.

Las clases se pueden ejecutar en máquinas conectadas en red, procedentes de diversos fabricantes, con distintos sistemas operativos, sistemas de BD e interfaces de usuario. Los estándares internacionales que permiten la intercomunicación son tan importantes como los estándares internacionales abiertos para las redes.

La organización para establecer estándares en la industria es el OMG (Grupo de Administración de Objetos, "Object Management Group"), que es una asociación comercial internacional, integrada por cerca de 200 compañías de hardware y software.

El OMG tiene un modelo de referencia para la arquitectura de administración de objetos. El objetivo de esta arquitectura es permitir que el software de diversos proveedores funcione de manera conjunta. Se pretende que influya en el diseño de los componentes sólo en la medida de lograr la interoperobilidad. El modelo de referencia indica:

- La forma en que los objetos hacen y reciben solicitudes y respuestas.
- Las operaciones básicos que deben ofrecerse por cada objeto.
- Las interfaces de los objetos que ofrezcan las capacidades comunes útiles en muchos aplicaciones.
- La arquitectura para la administración de objetos consta de cuatro partes principales:

Objetos de Aplicación (OA). Son las aplicaciones de uso final que pueden ser construidas por varios proveedores o por las organizaciones internas de sistemas de información.

Capacidades Comunes (CC). Son objetos y claves que ofrecen características de propósito general, útiles en muchas aplicaciones. Por ejemplo, imprimir, formar una cola de impresión, correo, administración de enlaces, editores de texto, ayuda.

Servicios a Objetos (SO). Son una colección que ofrece funciones básicas para crear y mantener a los objetos. Entre ellas se cuentan los sistemas para el manejo de archivos o bases de datos, administradores de transacciones, servicios o directorios, etc.

Agente de Solicitud de Objetos (ORB, "Object Request Broker"). El ORB es el centro de la arquitectura para la administración de objetos, permite que los OO se comuniquen de manera independiente de las plataformas y técnicas específicas. El objetivo del ORB es garantizar que los objetos puedan actuar entre sí, sin importan que estén en la misma máquina, canectados en una ambiente cliente/servidor o en diferentes redes de sistemas heterogéneos. Un objeto hace una solicitud de manera estándar y el ORB dispane el procedimiento de ésta. El ORB llama a cierto método y envía el resultado al solicitante.

DISEÑO DE LA BASE DE DATOS.

Una Base de Datos tradicional sólo almacena datas, sin procesarlos, de modo que resultan independientes de las procedimientas, los datos son accesibles a diferentes usuarios, con diversos propósitos. Por el cantrario, una BD OO almacena objetos. Los datos se almacenan junto con los métodos que procesan dichos datos. Toma la idea de las bases inteligentes de datos a su conclusión lógica; no se tienen acceso a dato alguna si no es a través de los métodos almacenados en la base de datos. Estos métodos están listos para entrar en acción al momento que reciben una solicitud. Los datos de todas los objetos quedan entonces encapsulados. En general, los datos son activos, más que pasivos. En el desarrollo estructurado de sistemas, los modelos conceptuales para el análisis, el diseño y las modelas para detección de acceso a la BD eran diferentes. Por el contrario, las técnicas OO utilizan los mismos modelos conceptuales para el análisis, diseño y canstrucción.

El madelo conceptual de la BD es igual al las diferentes modelos existentes, en lugar de utilizar palabras independientes como SQL. El uso de un modela conceptual para todos los aspectos de desarrollo, simplifica éste; mejora la comunicación entre usuarios, analistas y programadores, además de que reduce las posibilidades de error.

El diseño de una BD relacional es de algún modo independiente del diseño del programa. La BD es un espacio ajeno al espacio del pragrama. El pragramador debe de idear la forma de extraer o introducir datos en las tablas. Con las BDOO el pragramadar trabaja con objetos temporales y persistentes de manera uniforme. Los objetos persistentes están en la BDOO, con lo que se derriban los muros conceptuales entre la programación y la BD. Una persona que utiliza el mismo modelo conceptual puede hacer de manera iterativa el análisis, diseño, generación de código y la generación de la BD. Esta puede aumentar en gran medida la razón con la que una persona creativa idea y refina los sistemas.

Las BDOO se desarrollan al escribir en primer lugar, los tipos de objetos importantes del dominio de aplicación y los comportamientos asociados con aquellos tipos de objetos. Estos tipos de objetos determinan las clases que conformarán en la definición de la BDOO. Por ejemplo; una base de datos diseñada para almacenar la geometría de partes mecánicas podrá incluir clases como: cilindro, esfera y cubo. El comportamiento del cilindro podría incluir información relativa a sus dimensiones, volumen y área superficial:

```
CLASE CILINDRO {
        FLOAT ALTURA();
        FLOAT RADIO();
        FLOAT VOLUME();
        FLOAT AREA_SUPERFICIE();
};
```

Se pueden llegar a definiciones similares para el cubo y la esfera. En la definición anterior de la clase, altura, radio, volumen y área_superficie representan los mensajes que se pueden enviar a un objeto cilindro. Se observa la ausencia de detalles de la implantación, aún si se almacena a calcula la información. La implantación se lleva a cabo en el mismo lenguaje, escribiendo funciones correspondientes a las solicitudes OO:

```
CILINDRO :: ALTURA() {RETURN CILINDRO_ALTURA;}
CILINDRO :: VOLUMEN() {RETURN PI*RADIO()*RADIO()*ALTURA();}
```

En este caso, ALTURA se almacena coma un elemento de las datas, mientras que VOLUMEN se calcula mediante la formula aprapiada. Obsérvese que la implantación interna de VOLUMEN utiliza salicitudes para obtener ALTURA y RADIO. Sin embarga, el aspecta más impartante de la sencillez y uniformidad que experimentan los usuarios de la clase cilindra. Sálo necesitan canacer la farma de enviar una solicitud y las salicitudes dispanibles.

Este enfaque también le da flexibilidad a la BDOO, puesta que toda la aplicación se escribe mediante el envió de salicitudes, las implantacianes se pueden alterar en cualquier lado sin afectar a las aplicaciones. En este sentido, las BDOO afrecen una separación aún mayar entre las especificacianes de un sistema y su implantación.

Las BDOO se pueden canstruir mediante alguna de las tres enfaques siguientes:

- El primera utiliza un sistema de administración canvencianal de una BD y añade un nivel para procesamienta de las salicitudes y métada de almacenamienta OO. Este enfaque tiene un mérita: se puede utilizar el código actual altamente campleja de las sistemas de administración de BD, de modo que una BDOO se implante más rápida sin partir de cera.

- El segunda enfaque considera a la BDOO camo una extensión de las terminalagías de BD relaciónales. De este modo, las herramientas, técnicas y vasta experiencia de la tecnalagía relacional se utiliza para construir un nueva sistema administrador de BD. Se pueden añadir apuntadores a las tablas de relación para ligarlas con abjetas binarias de gran tamaño (BLOB).

- El tercer enfaque reflexiona sobre la arquitectura de las sistemas de BD y produce una nueva arquitectura aptimizada, que cumple las necesidades de la tecnalagía OO. Las praveedores que utilizan este enfaque afirman que la tecnalagía relacional es un subconjunta de una capacidad más general. Las estructuras relaciónales de datas se padrían utilizar en las casas adecuadas, pera la usual es que para las abjetas camplejas sean mejares atras estructuras de datas; éstas permiten el acceso a las datas de un abjeta campleja sin desplazar el meconismo de acceso. Las pramatares de las BDOO afirman que las BDOO na de relación son aproximadamente das veces más rápidas que las BD relacianales para almacenar y recuperar información compleja[1].

Las abjetas están campuestas par abjetas que, a su vez, están compuestas por más abjetas, etc. Debido a esto, es frecuente que la estructura de la clase sea muy compleja. Al actualizar un abjeta, sus datas deben paderse leer, sin tener que maver el meconismo de acceso al disco. Las datas de cada abjeta deben estar reunidas, esto no acurre si se utiliza una base de datas relacional El abjeta padría utilizar datas de abjetas de varias relaciones y, par tanta, necesitar varias mecanisma de acceso.

Las sistemas BDOO, evitan de manera deliberada el modelo de relación para mejarar el rendimiento de la máquina. Permiten que las datas de una clase se liguen entre sí de una manera eficiente; can frecuencia recurren a la estructura de apuntadores.
Si una clase utiliza con frecuencia determinados métadas, tiene sentida aptimizar la estructura de datas para abtener un buen rendimiento de la máquina mediante tales métodas[2].

Can las bases de datas relacionales, las datas se normalizan para evitar la redundancia[3] y las anamalías provacadas par la redundancia de las datas. Esta evita la redundancia en las datas pero na el código de aplicación.

---

[1] *Database Design Wiederhold Mc Graw Hill 1993.*
[2] *Object-Oriented Database Systems, ZD NONIK Stanley B. IBM System Journal, Dic 1990*

La tecnología OO utiliza la herencia para reducir el desarrollo redundante de los métodos. También crea clases diseñadas para utilizarse otra vez en muchas aplicaciones. El encapsulado y la herencia reducen la cantidad de código y datos redundantes, de dos maneras: mediante la herencia y la reutilización de las clases.

5.4 FASE DEL PROTOTIPO

El desarrollo de prototipos es una metodología valiosa para identificar con rapidez las necesidades particulares de información del usuario. La figura 5.34 muestra los cuatro tipos de información que se buscan con el desarrollo de prototipos.

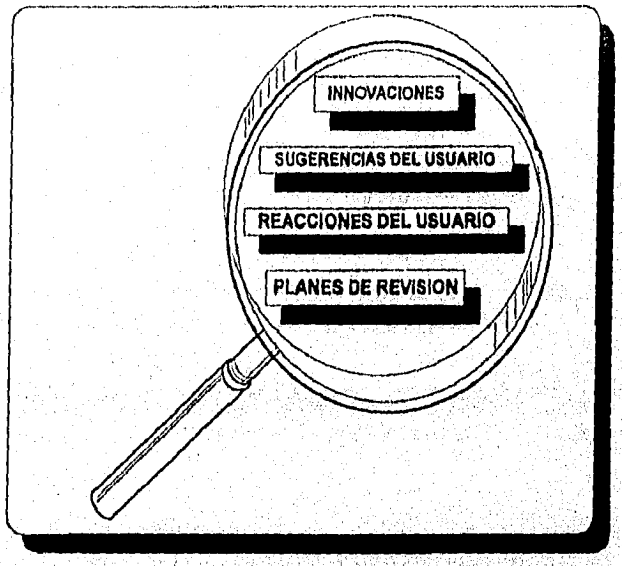


Figura 5.34. Información que se busca al desarrollar un prototipo.

Algunos usuarios no saben en realidad lo que quieren, sino hasta que ven algo tangible. Cuando los prototipos se desarrollan, se reduce efectivamente el periodo que transcurre entre la indicación de los requerimientos de información y la entrega de un sistema funcional.

Con un prototipo, el usuario puede ver en realidad lo que llegará a ser posible, y asimismo, cómo se traducen sus necesidades en hardware y software.

Entre los inconvenientes que implica el desarrollo de prototipos se tiene que el sistema de información puede conformarse de manera prematura aún antes de entender cabalmente el problema que se aborda. El uso de prototipos como alternativa puede inducir a que ciertos grupos de usuarios lo acepten, a pesar de ser inadecuado para las necesidades globales del sistema.

Se considera al desarrollo de prototipos como un método complementario, especializado para solucionar ciertos requerimientos de información del usuario, permite hacer planteamientos de necesidades de información más allá de los verbales entregados por los usuarios.

[9] *Martin James Managing The Database Environment Prentice Hall 1983.*

La determinación de los requerimientos de información puede satisfacerse con el desarrollo de protatipos, permite que se incorpore de manera activa al usuario en la determinación de sus requerimientos; de esta manera, el usuario expresa necesidades que sin el protatipo no llegarían a plantearse.

Los prototipos se consideran dentro del contexto de la última definición (esto es, que el prototipo incluye algunas, pero no todas las características), y que eventualmente, el prototipo funcional llegará a formar parte del sistema principal que se entregue al final.

El protatipo es sólo una parte del sistema que eventualmente se instalará. No es un sistema completo, ya que al desarrollarlo con rapidez, puede quedar limitado; contanda con sólo ciertas funciones elementales. Sin embargo, es importante imaginar y luego construir el prototipo como parte de un sistema actual, con el cual interactuará el usuario. Debe incorporar suficientes funciones representativas para que el usuario acepte que interactúa can un sistema real. El uso de protatipos permite reducir retroalimentación sobre el sistema propuesto, y además qué tan bien satisface las necesidades de información.

Existen cuatro lineamientos básicos para el desarrollo de prototipos, ver figura 5.35.

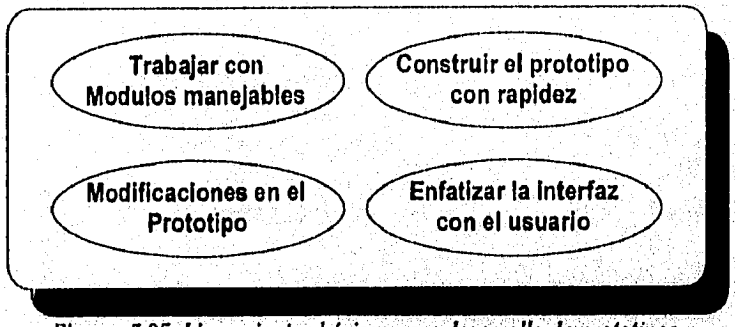


Figura 5.35. Lineamientos básicos para desarrollo de prototipos.

- **Trabajar con módulos manejables**: Una ventaja distintiva de los prototipos es que no necesariamente se construye un sistema funcional entero como prototipo. Un módulo manejable es aquél que nos permite relacionarnos con sus características, y además su construcción es independiente de otros módulos del sistema. Aquellas características que se consideren poco importantes quedarán fuera del protatipo.

- **Construir el prototipo con rapidez**: La esencia del desorrolla del prototipo de un sistema de información con éxito es la rapidez de su realización. Uno de los inconvenientes en el desorrollo de sistemas es, que el intervalo que transcurre entre el planteamiento de los requerimientos y la entrega del sistema completo es tan largo que, no llega a satisfacer de monera efectiva las necesidades evolutivas del usuario.

- **Modificaciones en el prototipo**: El prototipo debe tolerar modificaciones. Para ello, el protatipo requiere contar con módulos que tengan entre sí una baja dependencio. El prototipo se modifica en repetidas ocasiones, tales cambios deben acercar el prototipo al sistema que el usuario considera como relevante. Se debe tener la idea que el protatipo requerirá modificaciones, para transmitir al usuario que será necesaria su retroalimentación si desea que mejore el sistema.

- **Enfatizar la interfaz con el usuario**: El fin de crear un prototipo es que los usuarios planteen más allá sus requerimientas de información. Para ello deben ser capaces de interactuar, sin complicaciones, can el prototipo. Paro muchos usuarios lo interfoz, se contempla como el

sistema y no debería ser un obstáculo. La interfaz debe permitir al usuario interactuar con el mínimo de adiestromiento, y además contar con el máximo control sobre las funcianes presentadas.

Otras recomendaciones para el desarrollo de prototipos, con el fin de que su creación sea más eficiente y represente lo más cercano al sistema de información que se desee implantar son:

- Se debe promover y crear un ambiente de confianza y el grupo de trabajo debe tenerlo en cuenta.
- Si el usuario percibe cierto enfrentamiento al comentar sus opiniones, es poco probable que se planteen las reacciones y que éstas llegarán a ser fructíferas.
- Se debe contemplar la realización de sesiones privadas (con un mínimo de supervisión) donde el usuario interactúe y responda al prototipo.
- Realizar una bitácora de sesiones donde mínimamente se registre:

  - Fecha de la sesión
  - Nombre del usuario
  - Periodo observado
  - Sugerencias del usuario
  - Innovaciones
  - Planes de revisión

En los siguientes párrafos describiremos en forma general el comportamiento del sistema y los conceptos orientados a objetos que se utilizaron.

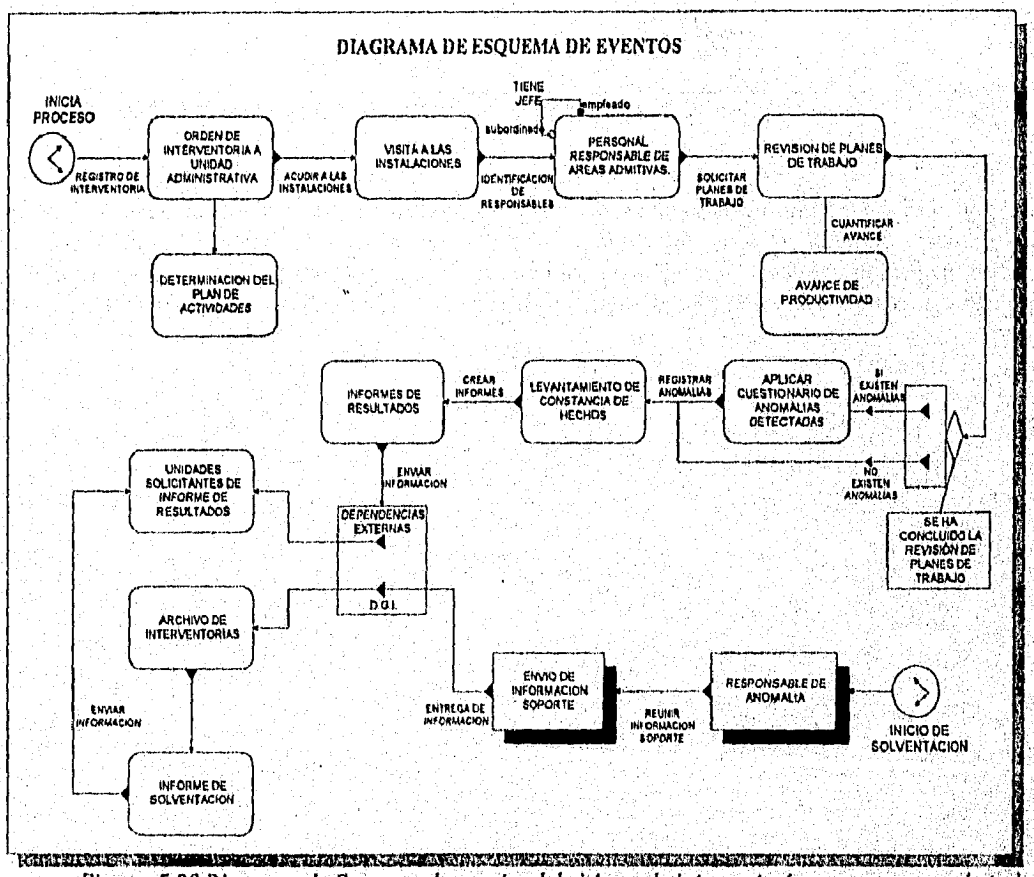


Figura 5.36 Diagrama de Esquema de eventos del sistema de interventorías en programas de trabajo

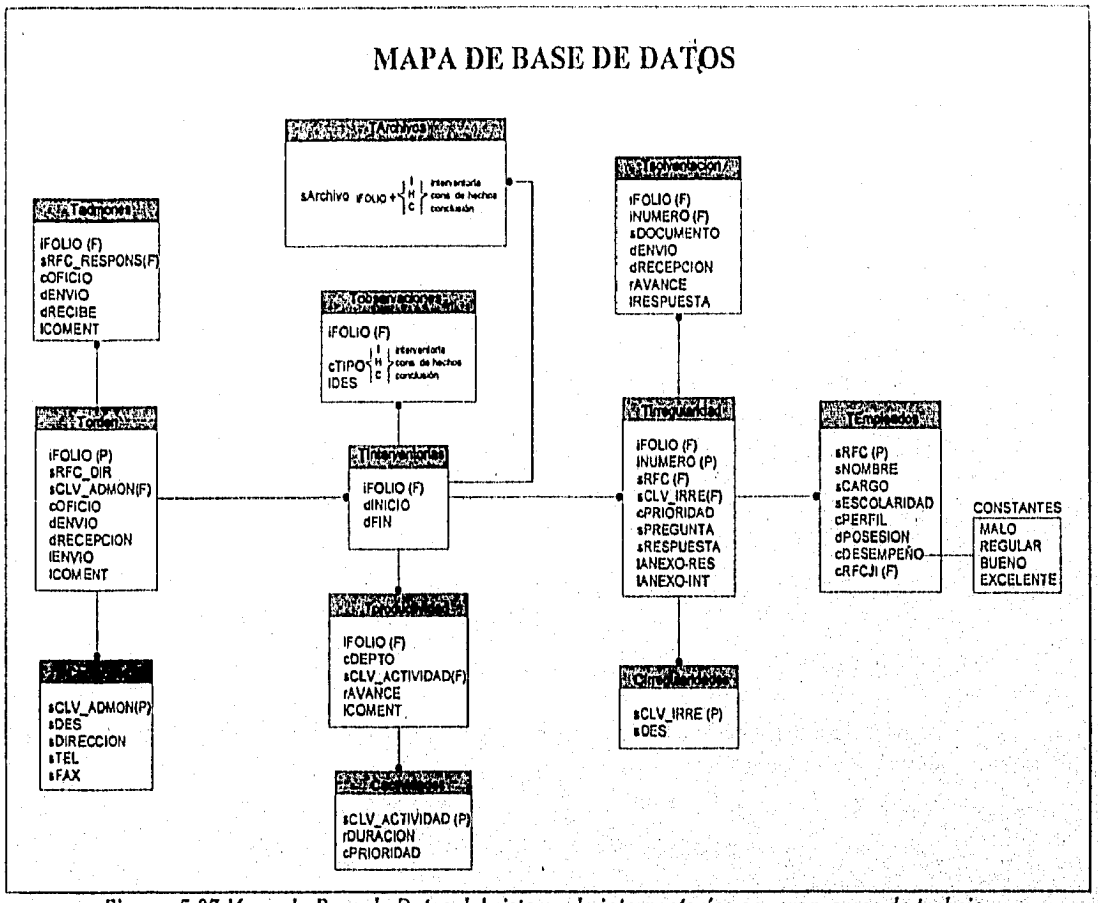


Figura 5.37 Mapa de Base de Datos del sistema de interventorías en programas de trabajo.

El objetivo principal del sistema es determinar las irregularidades que deben ser solventadas. La pantalla de la figura 5.38, contiene los posibles estados en que pueden estar las irreguloridades, como se representaron en la figura 5.14 "Transición de estados de una irregularidad". Los botones asocian la pantalla 5.39 de acuerdo a las leyendas del lado izquierdo, para desplegar las solventaciones de cada irregularidad.

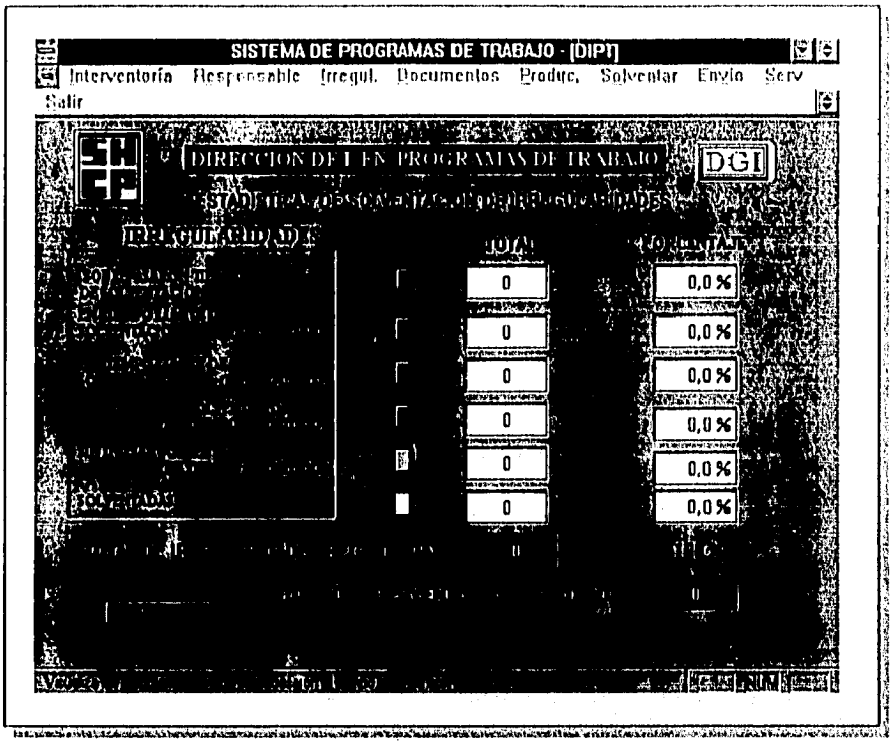


Figura 5.38 Pantalla que visualiza las irregularidades en sus diferentes etapas.

La pantalla 5.39 representa la aplicación del polimorfismo; la característica de que un objeto actúe de una forma de las diferentes que existen en su clase. Cabe señalar que los comportamientos están determinados por instrucciones en SQL.

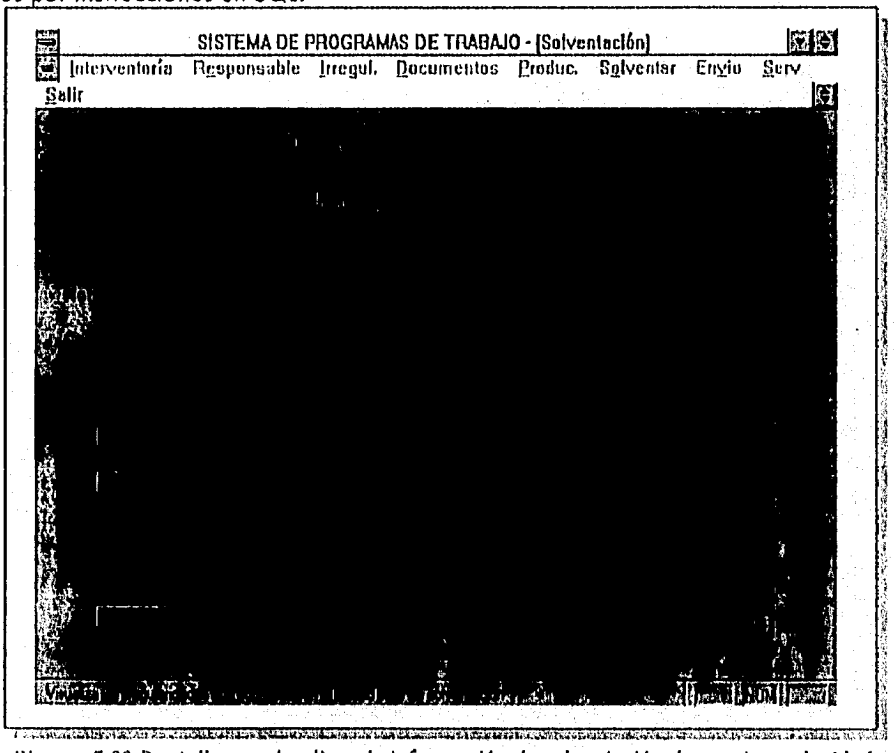


Figura 5.39 Pantalla que despliega la información de solventación de una irregularidad.

La selección de las diferentes opciones que contiene el sistema, ver figura 5.38, son de tipo lista desplegable.

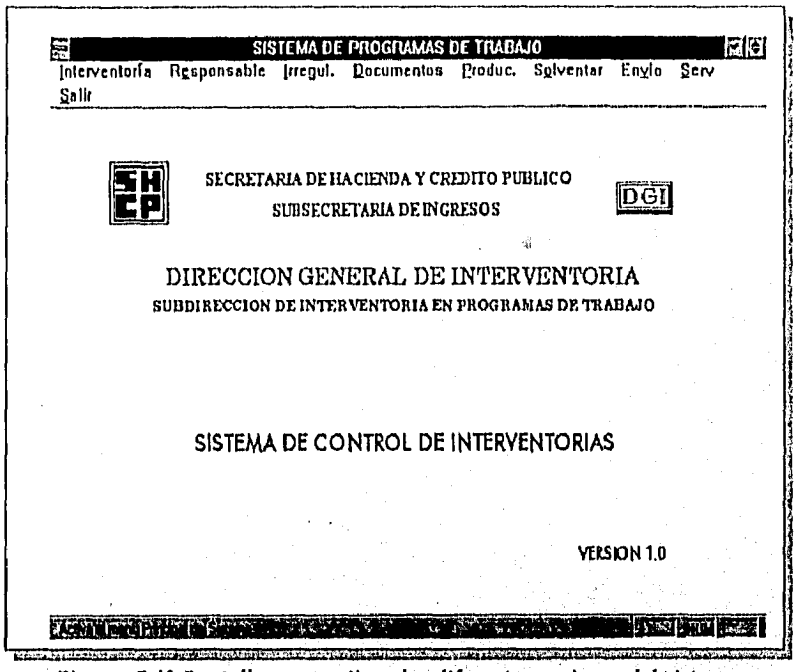


Figura 5.40 Pantalla que contiene las diferentes opciones del sistema.

Lo pantallo de lo figura 5.41 contiene el registro de coda interventoría, donde se aplicó el concepto de objeto, en el que los datos y los métodos fueron encapsulados. Si un objeto requiere información acerca del registro de interventorías se lo solicita a través de un mensaje, entonces el objeto de registro de interventorías retorno los datos de lo interventoría. El mismo concepto se aplico con el registro de irregularidades, cuya pantallo se muestra en la figura 5.42

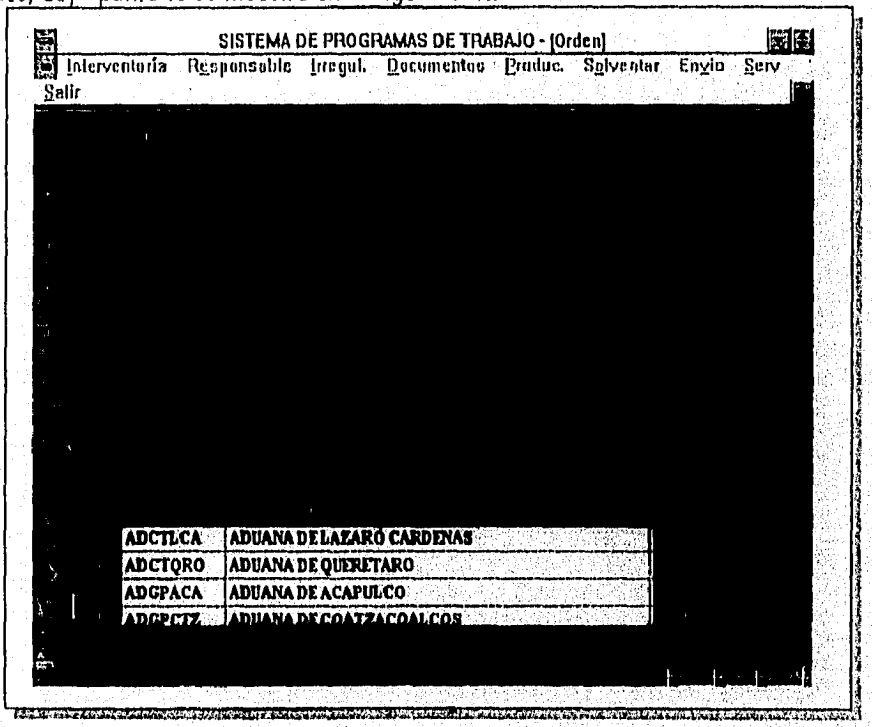


Figura 5.41 Pantalla que registra una interventoría.

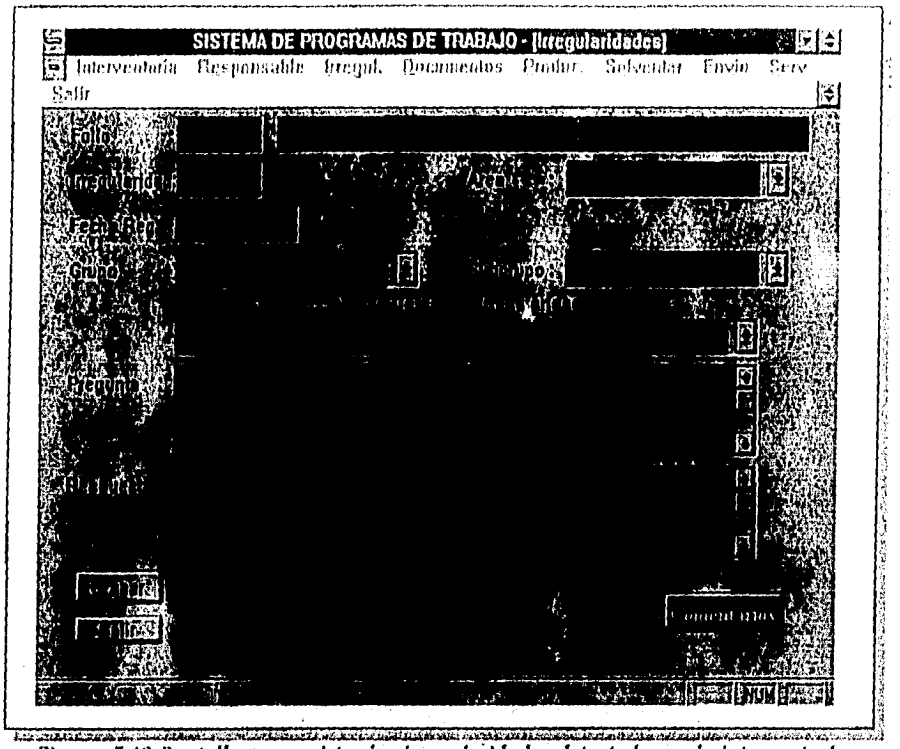


Figura 5.42 Pantalla que registra las irregularidades detectadas en la interventoría.

Uno de los aspectos importantes fue el de enlazar al sistema con el procesador de palabras Word de Microsoft. SQLWindows utiliza API's de Windows, por lo que para la generación de los informes nos permitió enlazarlos a través de estos, como se observa en la pantalla de la figura 5.43.

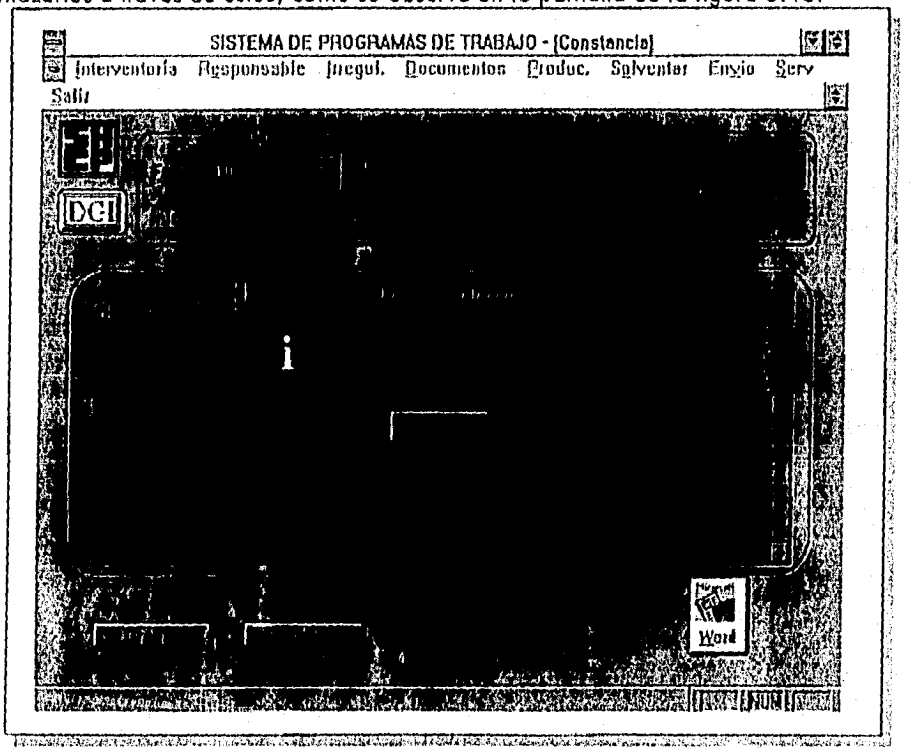


Figura 5.43 Pantalla de registro de informes.

El aspecto de herencia se aplico en la generación de los reportes; una clase puede heredar los métodos de otra clase y contener sus propios métodos. Al generar los reportes, como se muestra en la pantalla de la figura 5.44, los datos que contienen algunos reportes son generados a través de otros. Un ejemplo es el reporte de irregularidades y solventaciones que es generado a partir de los métodos individuales de cada uno de estos.

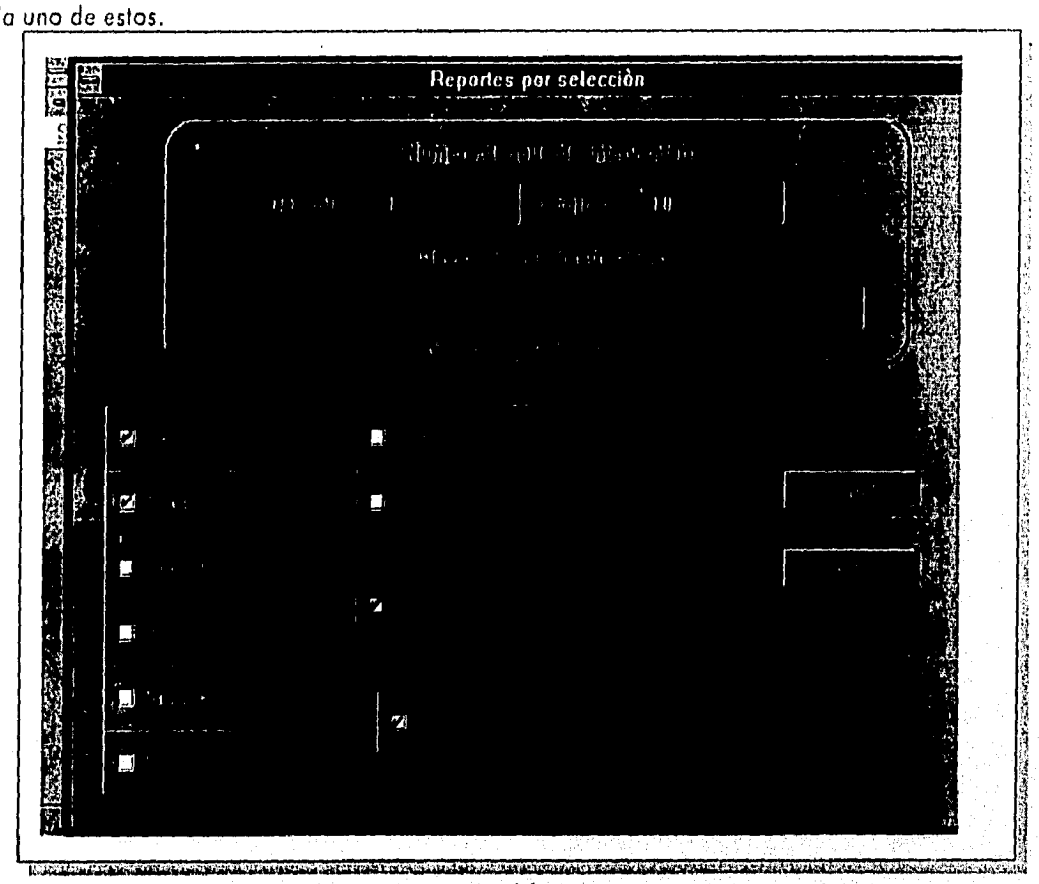


Figura 5.44 Pantalla que contiene los diferentes reportes que genera el sistema.

CONCLUSIONES

La arquitectura computacional cliente/servidor es una solución en tecnología informática que satisface las demandas actuales de costo-rendimiento, ofreciendo la optimización de las tareas y recursos existentes. Siendo sus ventajas, que la capacidad de cámputo está determinada por todas las equipos interconectados con que se cuenta, agregación de nuevas funcionalidades y una considerable reducción de trafico en la red, ya que las operaciones de procesamienta se jerarquizan y la red da prioridad a las necesidades del usuario, transmitiéndole información realmente útil. Esta constituye un estímulo al uso de sistemas abiertas dada que tanto clientes camo servidores aperan en diferentes plataformas de hardware y software lo cual permite comprar productas de diferentes proveedares sin interferir en el desempeño de las demás aplicaciones y equipos instalados.

Las cualidades que afrece esta arquitectura son interesantes y muy convincentes, pero esto no significa que sea perfecta, sabre todo cuanda la que se persigue es la integración plena de ambientes heteragéneas, por lo cual se enfrenta a :

- Muchas aplicaciones no son capaces de separar tareas por lo que el compartamiento en la red será el de una arquitectura tradicional.

- Se requiere de una evaluación y planeación completas de la arquitectura cliente/servidor para minimizar el tráfico y evitar la saturación del media de comunicación. Aspecto que una aplicación cliente/servidor bien diseñada si prevé.

- Como la administración de este ambiente es más complicada, las pracesos y herramientas al conectar equipa pueden camplicar ampliamente las labores de administración y contral.

- El personal que se requiere para dar soporte y administrar el ambiente seguramente tendrá que incrementarse. Primero parque al ser un medio abierto se requiere más personal, segundo porque al tratarse de medios abiertas se requieren más especialistas can conocimientos en diversas tecnologías. Además los costos de mantenimiento también se elevan, pues cada versión de una aplicación cliente/servidor se deberá prabar en diferentes plataformas.

Todos estos obstáculos se pueden contrarrestar y quizá las grandes ventajas que afrece un ambiente abierto sean de mayor pesa para emigrar a esta arquitectura.

Es importante recordar que la Tecnalagía cliente/servidor siempre tiene que ver con redes, lo cantraria sería falso, es decir no todas las redes trabajan baja la arquitectura cliente/servidor. Esta pequeña pero importante diferencia abre una brecha enorme en el tipo de productas o servicios que el usuario puede adquirir para apravechar su equipo. El conacer el verdadera significado del concepto es de trascendental impartancia para quien desea tener una solución productiva.

Por otra parte, el análisis y diseño orientado a objetas, cambia la forma en que pensamos sabre el desarrollo de sistemas en una forma más natural que las anteriores técnicas, tanta las usuarios cama los desarrolladares de software piensan en términos de objetas, eventos y mecanismos de activación, que son términas más familiares, por la que nas permite utilizar sólo un modelo conceptual para todos. La complejidad de los objetos llega a ser transparente para crear nuevos objetas a partir de las ya existentes permitiendo la reutilización y mantenimiento más eficiente de los objetos. Ya que si modificamas un objeto los demás na se verán afectadas par la razón de que san independientes.

La tecnalogía arientada a objetos puede atacar los problemas de diseño de base de datas y aplicaciones de forma unificada. Así las aplicaciones cliente/servidar san distribuidas, par lo tanto son más complejas

que las no distribuidas, este riesgo se debe eliminar mediante el uso de una buena planeación de la arquitectura y el praceso del desarrollo orientado a objetos.

La Subsecretaría de Ingresos ha realizado una fuerte inversión con lo que respeta a equipo de cómputo, software y una infraestructura de red (cableado, concentradores, routers, fibra óptica, etc) para implantar una plataforma cliente/servidor.

El proyecto que hoy se realiza en la Subsecretaría de Ingresos es muy ambicioso, ya que los requerimientos que se solicitan son de alcances muy altos , camo lo son las bases de datos en línea sumarizadas (Data warehouse).

Los beneficios de la arquitectura cliente/servidor, no se reflejan de inmediato y no es por cuestión de seguimienta, se puede considerar que debe existir todo un cambio de canciencia en los usuarios y reconocer que los avances tecnológicos en ámbito informático ya son un hecho. La arquitectura cliente/servidor no es un fin y debe considerarse más bien un medio que nos facilita realizar nuestras actividades asignadas con mayor calidad y rapidez.

🕮 SQL Windows Programming
William Gietz
Gupta Corporation

🕮 SQL Windows 5 Developer's Guide
Sunnet Shah & Chris Griswold
Sams Publishing

🕮 Microsoft Proyect Ver. 3.0
User Reference and Feature Guide
Microsoft

🕮 Análisis Y Diseño de Sistemas
Kendall y Kendall
Prentice Hall

🕮 Redes de Ordenadores
Andrew S. Tanenbaum
Prentice Hall

🕮 Enciclopedia de Redes
Tom Sheldon
McGraw Hill

🕮 Sistemas operativos modernos
Andrew S. Tanenbaum
Prentice Hall

🕮 C++ Un enfoque Orientado a Objetos
Luis Joyanes Aguilar
Mc Graw Hill

🕮 El entorno de programación UNIX
Brian W. Kernighan
Rob Pike
Prentice Hall

🕮 SCO UNIX Operating System
Ver. 4.0
Santa Cruz Operation

🕮 Metodología de Desarrollo: Programación Automática de Software con CASE
Lopez Fuensalida Antonio
MacrObit

🕮 CASE "La automatización de Software"
Carma Mc Clure
Addison -Wesley Iberoamerica

🕮 Introducción a la programación en Visual Basic
Kenyon Brown
Grupo Noriega Editores

BIBLIOGRAFIA

Implementing Production-Quality Client/Server System
Barbara Bochenski
Wiley Professional Computing

Client/Server Computing
Patrick N. Smith
Steven L. Guengerich
Prentice Hall

Client/Server Computing
Dawna Travis Dewire
McGraw Hill

Client/Server Survival Guide
Robert Orfali Harkey
Dan Harkey
Van Nostrans Reinhold

Análisis y diseño Orientado a Objetos
James Marti & James J. Odell
Prentice Hall

Aplique SQL
Gorff Weinberg
Mc Graw Hill

DB2/SQL
Tim Martin
Tim Hartley
Mc Graw Hill

Programación Orientada a Objetos una Introducción
Greg Voss
Mc graw Hill

Análisis y diseño Orientado a Objetos con aplicaciones
Grady Booch
Addison Wesley

Objet Oriented and Design
James Rumbough
Prentice Hall

Análisis estructurado Moderno
Eduard Yourdon
Prentice Hall

APENDICES

VISUAL BASIC y HERRAMIENTAS CASE

LA VERDAD

Un buscador de la verdad salió a los caminos del mundo y en el cruce de caminos de un desierto interrrogó a un grupo de personas que en ese momento pasaba, preguntando que ¿cuál era la verdad ?

- *Busca en la filosofía -respondieron los filósofos.*
- *No, argumentaron los políticos -La verdad está en el servicio.*
- *Entra en las catedrales - le aseguraron los clérigos.*
- *Sin duda, la verdad es;- la sabiduría -terciaron los sabios.*
- *Renuncia a todo -esgrimieron los ascetas.*
- *Contempla y ensalza las maravillas del Señor -le anunciaron los místicos.*
- *Acata y cumple las leyes señalaron los gobernantes.*
- *Cónocete a tí mismo cantaron los guardianes del esoterismo.*
- *La verdad está en los numeros sagrados -dedujeron loscabalistas.*
- *Vive los placeres del mundo- aseguraron los epicureas.*
- *Unete a nosotros- le gritaron los revoluctonarios.*
- *Vive y deja vivir - clamaron los existencialistas.*
- *La verdad es un mito- respondieron los escépticos.*
- *El pasado : es la unica verdad - lamentaron los nostálgicos.*

Confundido, aquella persona se dejó caer sobre el polvo del camino, mientras aquella multitud se alejaba, cantando y reinvidicando su verdad; en eso, acertó a pasar un anciano que portaba un refulgente diamante.

¿ Quien eres? preguntó el derrotado buscador de la verdad. Y el anciano, mastrándole el diamante, respondió:
Soy el guardián de la verdad.
¿La verdad? ¿ Es que existe?

El anciano sonrió y aproximando la gema al rostro replico:
La verdad como este tesoro, tiene mil caras; a cada uno le corresponde averiguar cúal le toca.

APENDICE A

VISUAL BASIC

Visual Basic de Microsoft es un lenguaje de programación que ha sido muy aceptado como herramienta de desarrollo para aplicaciones cliente/servidor. Su popularidad se debe al gran número de programadores familiarizados con Basic y la integración al ambiente Windows, puede residir en el Front-End (su lugar más común) o en el Back-End.

Visual Basic no contiene todos los objetos como SQLWindows, muchos de los objetos (controles) se adquieren por separado como extensiones a Visual Basic "VBX", (Extensiones de Visual Basic, "Visiual Basic eXtension"). No soporta totalmente la POO (como herencia múltiple entre clases). SQLWindows provee acceso a otros DBMS a través de ruteadores, que en comparación con ODBC (Canectividad Abierta a Base de Datos, "Open DataBase Connectivity") son más rápidos.

Visual Basic es una versión estructurada de Basic, cada programa consiste de formas y módulos.

- Las formas contienen los objetos que se despliegan en las ventanas (código y controles que maneja las ventanas).
- Los módulos son archivos que contienen solo código y variables que componen funciones o subrutinas y que son accesadas por otros módulos.

Es una herramienta de desarrollo de aplicaciones en Windows con atractivas interfaces gráficas. Es un hecho que Microsoft proporciona una herramienta de interfaz gráfica de usuario (GUI). Esta se crea con solo dibujar objetos en una aplicación dentro del mismo modo gráfico y definir las propiedades de esos objetos, así como manejar su comportamiento escribiendo código que responda a los eventos que ocurren en la interface.

Las características principales de Visual Basic son :

- Interfaz de documentos múltiples MDI (Multiple-document interface)
- OLE (Object Linking & Embeddung)
- DDE (Dinamic data exchange)
- Extensiones añadiendo Custom Controls y llamadas a procedimientos "Dynamic data Exchange" DLLs

- Genera archivos ejecutables (.EXE), que utilizan en tiempo de ejecución DLLs que permiten libremente ser distribuidos.

La idea de crear un proyecto en Visual Basic es unir todo lo necesario para que una aplicación se ejecute; formas, módulos y librerías. Para esto es necesario disponer de componentes VBX, OCX, OLE Custom Controls, DLL (Dinamically Linker Library) y ODBC para conexión a DBMS de otros proveedores.

Estos componentes son llamados PLUG-IN, están integradas en código precompilado que usualmente realizan una tarea en particular, consisten en interfaces como TCP/IP, API's y adiciones de interfaces para usuarios como VSVBX de Videosoft.

Los componentes DLL son programas escritos en C, C++ ó en otros lenguajes. Contienen rutinas para cálculos matemáticos ó tareas especificas de interface.

Los componentes VBX y OCX son muy fáciles de usar; por ejemplo, algunos se adicionan a la paleta de herramientas para el diseño de formas y de ahí se toman, como lo muestra la figura A.1.

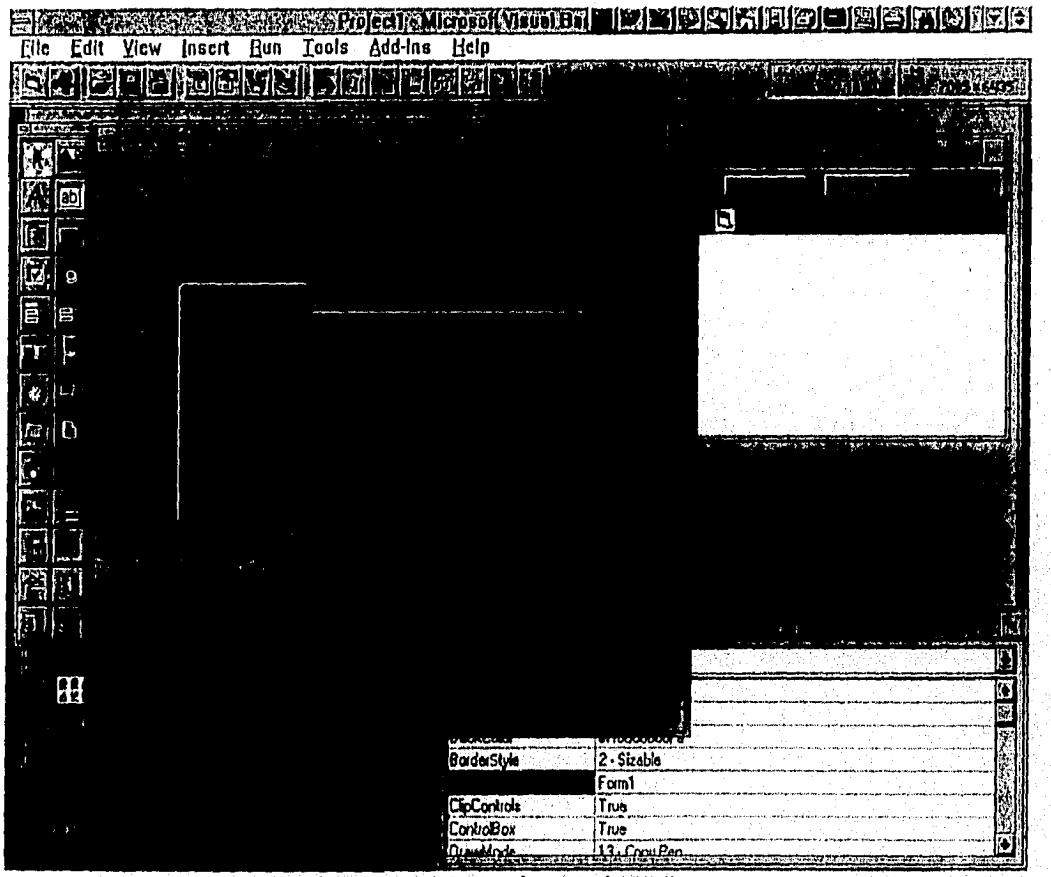


Figura A.1. Uso de controles VBX

Los componentes PLUG-IN son numerosos y existen para casi todas las tareas que se desean en una aplicación como por ejemplo; realizar una vista previa de un reporte , sólo es necesario comprar el componente anexarlo y literalmente ejecutar el comando "vista previa" para que este haga todo. Bajo este concepto Visual Basic crea la idea de componentes disponibles para el usuario propio y de terceros vendedores.

ODBC (Conectividad Abierta a Bases de Datos, "Open DataBase Connectivity") son funciones API's que permiten accesar datos de diferentes DBMS como ; Foxpro, Btrieve, lenguajes xBase, Paradox, SQLServer, SQLBase e Informix entre otras. Estos API's consisten de llamadas para estoblecer canexión , sesión de mantenimiento, comandos, etc. a las Bases de Datos externas y que consisten en más de 100 llamadas, ver figura A.2.

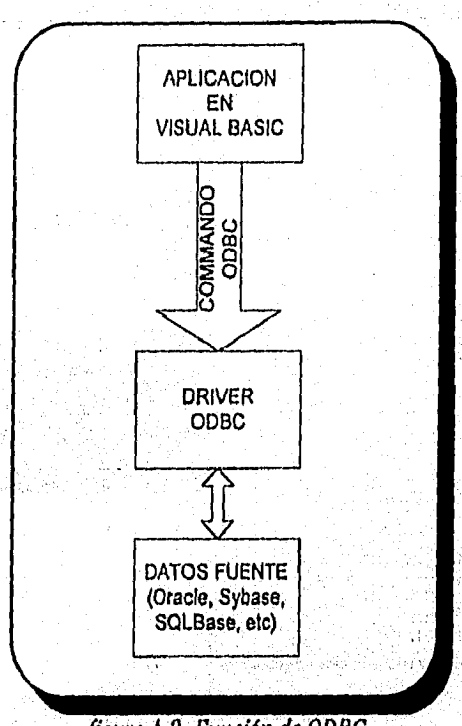


figura A.2. Función de ODBC.

ODBC tiene capacidades que hacen transparente el acceso a los datos y que están basadas en SQL como: cursores de resultados de consultas, transacciones, concurrencia y procedimientos almacenados entre otros.

Al iniciar Visual Basic presenta una pantalla nombrada Forma y utiliza una caja de herramientas similar a la de los programas de dibujo, donde se añaden con estos: botones de comandos, textos, dibujos entre otros objetos. Cada programa puede tener tantos controles como requiera el usuario, figura A.3.

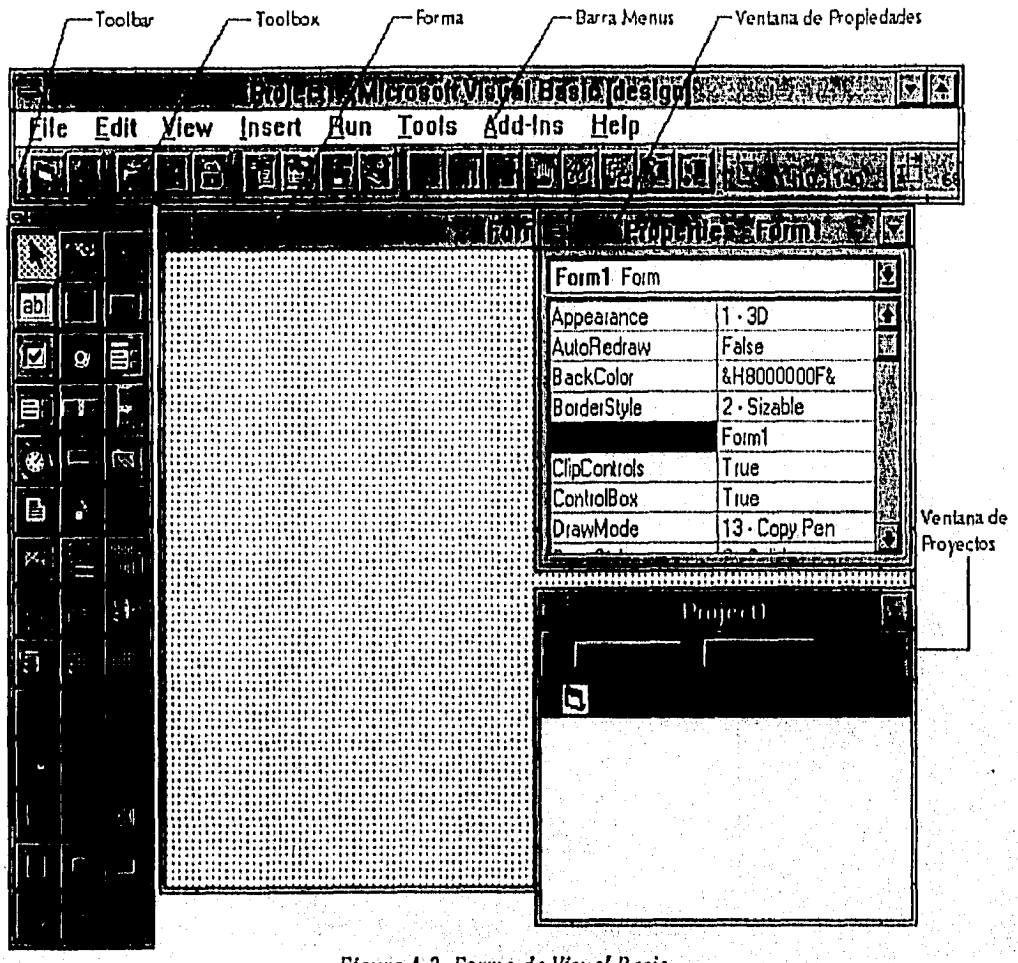


Figura A.3. Forma de Visual Basic.

A continuoción se describen cada uno de estos elementos.

Toolbar — Provee un rápido occeso a los comondo más usodos en el ambiente de progromación. La descripción generol de los comondos contenidos en esta barra, se muestro en la siguiente figura A.4.

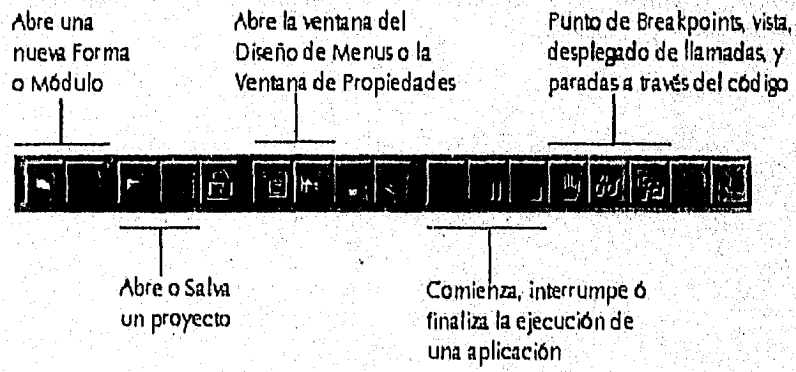


Figura A.4. Botones de Barra de Herramientas

ToolBox — Provee determinadas herramientas de control, representadas por iconos; que son utilizadas al tiempo de diseño dentro de las formas. El Toolbox se extiende cada vez que se adiciona un control. Cada archivo con una extensión VBX, dentro de la Ventana de Proyectos; provee uno o más nuevos controles para el Toolbox.

Barra Menús — Permite desplegar los comandos que se usan para el desarrollo de la aplicación.

Forma — Es la interface de desarrollo de la aplicación. Funciona como una ventana elemental de Windows. En ella se puede añadir controles, gráficas, dibujos, entre otros elementos. Cuando se crea un nuevo proyecto, Visual Basic crea una Forma vacía, con el título como Form1.

Ventana de Proyectos (Project) Lista las formas, módulos de código y los archivos de controles (Custom Control) que pertenecen al proyecto. Un proyecto es una colección de archivos que se utilizan para construir una aplicación.

Ventana de Propiedades. (Properties) Lista las propiedades de la Forma ó control seleccionado. Una propiedad es un mecanismo formal para describir los atributos de un objeto, como el tamoño, título ó color.

Los principales pasos para desarrollar una aplicación en *Visual Basic* son los siguientes:

1. CREAR LA INTERFACE
2. DETERMINACIÓN DE PROPIEDADES
3. ESCRIBIR CÓDIGO.

Para ver como se realiza esto, se crea una simple aplicación que consiste en una caja de textos y un botón de comandos. Se realizará de la siguiente forma al dar click en el botón, el mensaje "Hola" aparece en la caja de textos.

1.- CREACIÓN DE LA INTERFACE.

El primer paso es dibujar el objeto. Utilizaremos las consolas del *ToolBox*. Figura A.5.

Control	Nombre
	Text Box (Caja de Textos)
	Command Button (Botón de Comandos)

Figura A.5. Botones de Toolbox

Para dibujar un control usando el *Toolbox*.

1. Hacer click en el control que se desea. Utilizar en este caso, el de Text Box.
2. Mover el apuntador dentro de la *Forma*. Ver figura A.6.

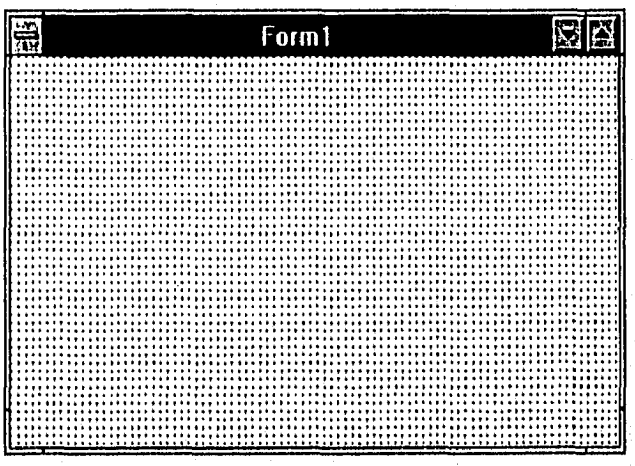


Figura A.6. Pantalla de Forma.

3. Hacer click en la Forma con el botón izquierdo del mouse. Sin soltar el botón, arrastrar el objeto.
4. Soltar el botón del mouse

De esta manera, el control aparece en la *Forma*.

Un simple camino para añadir un control dentro de la *Forma*, es hacer doble-click en el icono del control del ToolBox: entonces se crea un control localizado en el centro de la *Forma*, con tamaño normal.

2.- COLOCACIÓN DE PROPIEDADES

Como colocar las Propiedades para los objetos que se crearon. La *Ventana de Propiedades (Properties)* provee un fácil camino, para la determinación de las *Propiedades* de todos los objetos en la *Forma*. Para abrir la *Ventana de Propiedades*, se puede: seleccionar desde el menú de Windows; hacer click en el botón de Propiedades en el *Toolbar* ó bien, presionar la tecla F4.

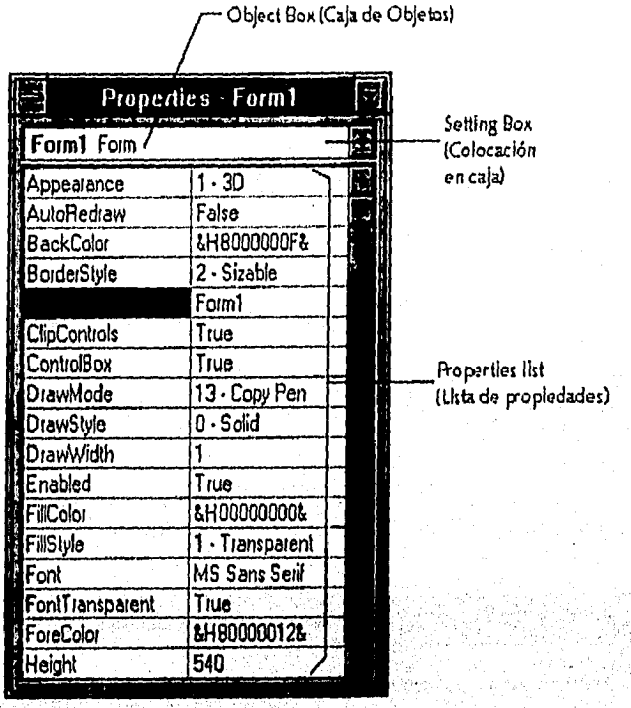


Figura A.7. Ventana de propiedades

La Ventana de Propiedades consiste de los siguientes elementos:

Object Box . - Despliega el nombre del objeto al cual se determinan las propiedades. Al hacer click en la flecha a la derecha de la caja, se puede seleccionar una Forma ó cualquier control que se encuentre dentro de la Forma actual.

Setting Box .- Permite editar la colocación, para la propiedad seleccionada en la lista de propiedades. Alguna colocación puede ser cambiada, haciendo click en la flecha a la derecha de la Caja: ésta despliega una lista de opciones, y puede seleccionarse alguna.

Properties list.- La columna izquierda despliega la lista de propiedades que posee el objeto. La columna derecha determine la colocación de cada propiedad.

Para completar el ejemplo anterior, donde se despliega el mensaje "Hola"; se necesita cambiar a los objetos, las siguientes propiedades:

Objetos	Propiedades	Setting
Form	Caption	Hola
TextBox	Text	(Vacio)
Text Box	FontName	Arial
Text Box	FontSize	16
Command1 Button	Caption	OK

Para continuar con el ejemplo, añadir un segundo Command Button (Botón de comando) y colocar la siguiente propiedad:

Command2 Button	Caption	Exit

3.- Escribiendo Código.

La *Ventana de código*, es donde se escribe el código de *Visual Basic* para la aplicación. El código consiste del lenguaje de declaraciones y constantes. Usando la *Ventana de código*, se puede rápidamente ver y editar el código en la aplicación.

Para abrir la Ventana de código.

Hacer doble-click en la *Forma* ó en el control que está dentro de la *Forma* al que se desea escribir código.

También se puede dentro de la *Ventana* de proyectos, seleccionando el nombre de la *Forma* y seleccionando el botón *View Code*.

La figura A.8 muestra la Ventana de código *(Code Windows)* que aparece, cuando se hace doble-click en el *command button control* (botón de comandos)

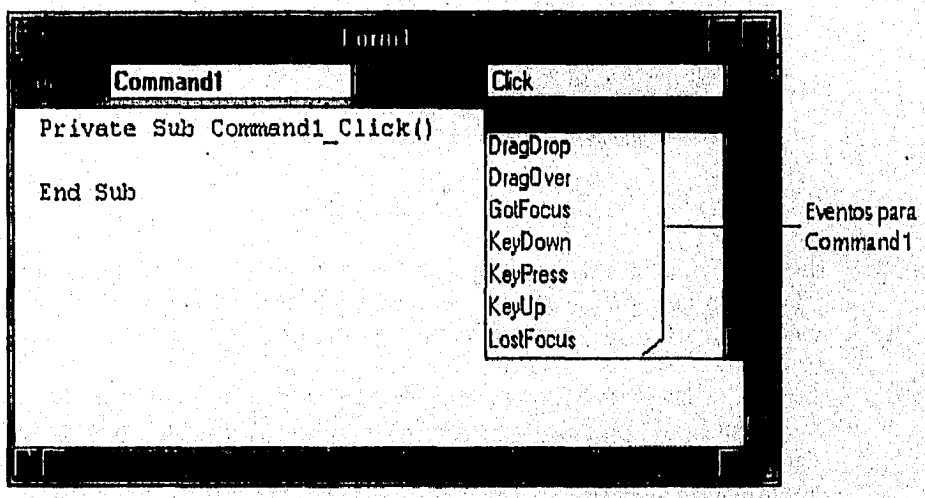


Figura A.8. Ventana de Código

La *Ventana de código* incluye las siguientes elementos:

> *Objet Box .- Despliega el nombre del objeto seleccionado. Haciendo click en la flecha que se encuentra al lado derecho de la caja, se despliega una lista de todos los objetos asociados con esta Forma.*
>
> *Procedure List Box. - Lista los procedimientos para un objeto. La caja despliega el nombre del procedimiento seleccionado; en este caso, Click. Seleccionando la flecha a la derecha de la caja, se despliega la lista de todos los procedimientos para el objeto.*

El código, en una aplicación de *Visual Basic;* es dividido en pequeños bloques llamados *procedimientos (procedures)* . Un *event procedure* , como el que se creó anteriormente; contiene *código* que es ejecutado cuando ocurre un evento (como hacer click en el botón).

Para crear un event procedure.

1. En el Object Box, se selecciana el nombre de un objeto en la Forma que se encuentra activada, para continuar con el ejemplo anterior y seleccionar el *command button*, Command1.

2. En el Procedure List bax, se selecciana el nambre de un evento para el objeta seleccianada.

   Aquí, el procedimienta Click ya fue seleccionada. Este es el pracedimienta default para los *command buttans.*

3. Escribir el siguiente código entre la declaración Sub y End Sub:

   Para el Command Button, Command1 escribir:

```
Form1.BackColor = QBColor(Rnd * 15)
Text1.Text = "Hola"
Text1.BackColor = QBColor(Rnd * 15)
Text1.FontSize = 16
Text1.FontName = "Arial"
```

   El evento procedure de Command1, se ve así:

```
Sub Command1_Click( )
    Form1.BackColor = QBColor(Rnd * 15)
    Text1.Text = "Hola"
    Text1.BackColor = QBCalor(Rnd * 15)
    Text1.FontSize = 16
    Text1.FontName = "Arial"
End Sub
```

   Para el Command Buttan, Command2 escribir: End

```
Sub Command2-click()
    End
End Sub
```

La sintaxis para este ejemplo, es tomada desde *object.property= setting*. Se puede usar esta sintaxis para *Formas* y controles en la ocurrencia de eventas, mientras la aplicación está corriendo.

APENDICE B

HERRAMIENTAS CASE

La historia de la tecnología CASE comienza a principios de los años ochenta con la introducción de la documentación asistida por computadora y de las herramientas de diagramación las cuales representaban los primeros intentos para automatizar el análisis y diseño de tareas. Su propósito era producir automáticamente la documentación estructurada requerida por las distintas metodologías de desarrollo.

Este cambio en las herramientas de software es parte de una nueva tecnología denominada Computer Aided Software Engineering ó CASE (ingeniería de software asistida por computadora). Es una tecnología para automatizar el desarrollo y mantenimiento de software siendo además una respuesta muy práctica en los últimos 25 años de la crisis del software. Realmente CASE no es totalmente nueva, ya que está construida sobre técnicas estructuradas que se desarrollaron en la década de 1970.

HISTORIA DE CASE

1a. GENERACION (84 - 86)	Herramienta para diagramación simple sobre PC/XT
2a. GENERACION (87 - 89)	Herramienta de diagramación apoyada en métodos, generación de documentación y generación de código sobre PC/AT y workstation nivel proyecto.
3a. GENERACION (90 - ?)	Herramientas múltiples integradas a través de un repositorio sobre PC/386 y workstation a nivel empresa.

EVOLUCION DE LA TECNOLOGIA CASE

Principios de los 80	Ayuda de computadora para documentación Ayuda de computadora para diagramación. Herramientas de análisis y diseño.
Mediados de los 80	Chequeo automático de análisis y diseño.

	Almacenamiento de diagramas estructurados en bibliotecas automatizadas (repositorio)
Finales de los 80	Generación automática de código. Ligar el diseño automatizado a la programación automatizada
Principios de los 90	Manejo de metodología inteligentes Habilitación de interfases con el usuario Reutilización como metodología de desarrollo.

En sus inicios las herramientas CASE se enfocaron principalmente a la automatización de la documentación para mejorar la productividad del software. A mediados de los años ochenta, las herramientas CASE se mejoraron para proporcionar dos funciones muy importantes

- Comprobación automática de diagramas estructurados.
- Almacenamiento de diagramas estructurados en librerías de diseño automático llamadas diccionario, depósitos o enciclopedias.

La base de CASE está en el diseño automático, porque el análisis y el diseño se consideran como los fases iniciales del ciclo de vida del software.

El paso siguiente en la evalución de la tecnalogía CASE fue la unión del diseño autamático con la programación automática. Mientras que el diseño automático cubre las tareas de análisis y diseño, la programación automática significa generadores automáticos de código. La unión implica que del 80 al 90 por cienta del sistema del saftware puede generarse a partir del diseño de diagramas estructurados.

El diseño automático lo soportan las herramientas CASE basadas en computadoras personales desarrolladas durante la primera mitad de la década de los años ochenta. La programación autamática la praparcionan los generadores de aplicacianes y de código de la cuarta generación, la mayoría de los cuales están basados en mainframes. Para unir el diseña y la programación automática se requiere de un puente entre dos entornos hardware, (camputadoras personales - mainframe) y dos tecnalogías de saftware (CASE y lenguajes de cuarta generación). Una parte muy importante para esta unión será añadir un repositorio al entorno CASE.

El repositorio CASE es un mecanisma para almacenar y organizar toda la información sabre un sistema de saftware. Esta información es necesaria para la gestión del proyecto, y para la generación automática de código incluye, información del problema que se va a resolver, el dominio del problema, los procesos del saftware que están siendo utilizados, las modelos de procesos, prototipos, recursos del proyecto, y del contexto organizativo.

Además de la unión entre el diseño automático y la programación automática, el repositorio CASE convierte por primera vez el concepto del saftware reutilizable en algo practico. Los componentes del saftware reutilizables son la clave del vertiginaso aumento de la productividad del saftware. En lugar de ir creando cada vez nueva saftware o de mejorar el dudoso, se pueden utilizar los componentes del saftware reutilizable almacenada en el repositorio CASE. No solamente se podrán reutilizar las módulos de código fuente sino que también podrán ser usados los planes de proyectas, modelos de prototipas, madelas de datos y especificaciones de diseña.

CASE ha cambiado radicalmente la farma de construir las sistemas de saftware al proporcionar tres áreas aveces principales:

1 Un entorno de desarrollo interactivo con un tiempo de respuesta rápido, recursos dedicados y uno comprobación de errores desde el principio.

2 La automatización de muchas toreas de desarrollo y mantenimiento del software.

3 Una programación visual proporcionada por potentes estándares gráficos.

Lo meta de la tecnología CASE es automatizar todo el ciclo de vida del software mediante un conjunto de herromientas de software integradas. No obstante para el estado tecnológico actual se trata de una meta o conseguir, no de una realizada.

La tendencio de CASE es que el desorrollo y mantenimiento del software debería verse como una actividad formal y disciplinada capaz de una comprobación mejor de la exactitud y automatización del proceso. La única forma real para incrementar significativamente la calidad y la productividad de software es a través de la automatización .

CAMBIOS HACIA EL CICLO DE VIDA CASE

Los cambios y mejoras de la tecnología CASE es que el desarrollo y mantenimienta del saltware son variados. En la siguiente tabla se muestran las diferencias entre el desarrollo de saltware tradicional y el de CASE a través de la automatizoción.

DESARROLLO TRADICIONAL	AUTOMATIZACIÓN DEL SOFTWARE.
• Énfasis en codificación y pruebas	• Énfasis en análisis y diseño
• Especificación basada en papel	• Uso de prototipos
• Codificación manual	• Generación automática de código
• Documentación manual	• Generación automática
• Pruebas de software	• Chequeo automático de diseño
• Mantenimiento de código	• Mantener especificaciones de diseño

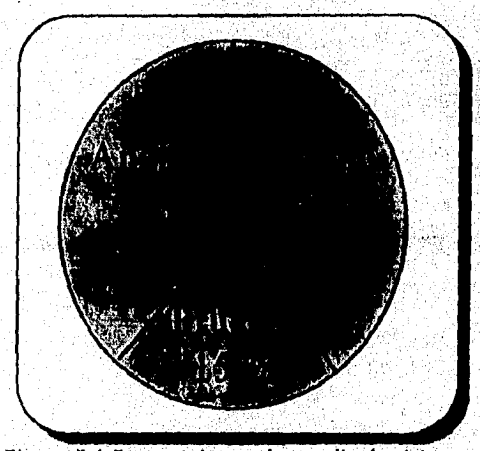


Figura B.1 Porcentajes en desarrollo de sistemas.

Como se muestra en la figura B.1., los desarrolladores se enfocan más a las etapas de análisis y diseño que a las de codificación o prueba. Lo codificación manual es sustituida por los generadores de código CASE que producen de un 80 a 100 por ciento del código del sistema

en base a especificaciones de diseño, las cuales son estructuras de datos, entidades de datos y procedimientos.

Estas especificaciones son creadas interactivamente en la computadora por analistas de sistemas y son introducidas en librerías automáticas CASE para poder ser utilizadas después si es necesario.

La tecnología CASE cambia la metodología de análisis de sistemas tradicional. Se crea un prototipo que reemplaza a los métodos basados en papel de usuario. Las herramientas CASE tales como el diseño de pantallas y reportes o lenguajes de especificación ejecutables pueden ser usados para construir modelos prototipo de sistemas, permitiendo así determinar los requerimientos del usuario.

CASE reduce enormemente la prueba de software, el análisis y chequeo automática, y por ende elimina errores. A su vez el mantenimiento se simplifica con librerías de diseña y los generadores de código.

Finalmente, entra el concepto de reutilizable en el desarrollo de software; esto es, código fuente, modelos de datos y especificaciones de diseño, los cuales pueden ser reutilizados por los desarrolladores.

COMPONENTES DEL SISTEMA CASE

Podemos dividir un sistema CASE en tres componentes básicos figura B.2.

1) Front-end (Upper Case - o - Case de alto nivel)
2) Repositorio
3) Back-end (Lower Case- o - Case de bajo nivel)

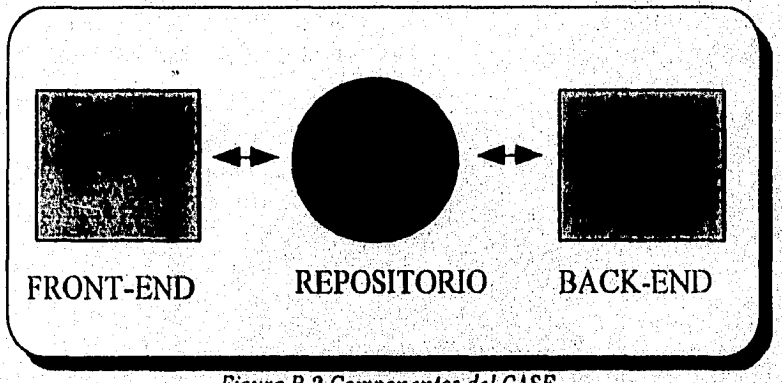


Figura B.2.Componentes del CASE

El componente front-end corresponde a las fases primarias del ciclo de vida del software; es decir, el análisis y el diseño. El front-end puede corresponder a la parte de la computadora o estación de trabajo de la plataforma de hardware Case. Las herramientas Case front-end proporcionan funciones para soportar las actividades de análisis y diseño, como diagramación, prototipos y comprobación de especificaciones. Esta actividades se realizan mucho mejor en una computadora personal dedicada con tiempo de respuesta rápida, mapa de bits gráficos de alta resolución y mouse.

El componente Back-end corresponde a las últimas fases del ciclo de vida del software; es decir la implantación y el mantenimiento del programa. El back-end también corresponde a la porción de la computadora principal en la plataforma de hardware Case. Las herramientas del Back-end Case automatizan el código, las comprobaciones, la generación de las bases de datos, la normalización de los datos y el análisis del impacto en el sistema existente. Estas tareas requieren la potencia y capacidad de almacenamiento de minicomputadoras o Mainframe.

El repositorio es el enlace entre los componentes front-end y back-end de un sistema CASE. Es el vehículo de la comunicación por el cual toda la información del sistema reunido durante el ciclo de vido del software se gestiona y comparte. Por medio del repositorio el trabajo del sistema en los diferentes equipos se puede combinar, analizar y consolidar en una representación del sistema consistente, en progreso y completo.

Lógicamente, el repositorio CASE está dividido en librerías de proyectos y en modelos del sistema. Físicamente, está dividido en niveles que no corresponden con las Plataformas de hardware del sistema CASE. A nivel computadora personal existe un repositorio local para soportar el desarrollo individual. A nivel computadora principal existe un repositorio para mantener toda la información corporativa. A nivel de departamento o de proyecto hay un repositorio intermedio para guardar la información.

Debe existir un enlace entre todos los repositorios anteriores para que los datos puedan cargarse y descargarse entre los niveles.

Esto es, considerar en la conectividad entre los sistemas hardware lo siguiente :

- Selección de los datos a descargar
- Cargo y consolidación de los datos
- Necesita de reconstrucción del formato de los datos
- Control y seguridad de los datos
- Velocidad
- Eficiencia
- Acceso multiusuarios a los archivos
- Copias de seguridad (backup)

LAS PRINCIPALES HERRAMIENTAS CASE

Las diferentes herramientas CASE, se enfocan hacia el soporte de diferentes fases del ciclo de vido del software al desarrollo de diferentes tipos del sistemas software. Para distinguir hay tres categorías básicas de herramientas CASE:

1) Juegos de herramientas CASE.
2) Bancos de trabajo CASE.
3) Asistentes de metodologías CASE.