PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERIA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TITULO DE INGENIERIA DE SISTEMAS

"Diseñar y construir un sistema que permita monitorear y asignar el uso de computadoras en el centro de cómputo de la PUCESA utilizando dispositivos de lectura de códigos de barras."

William Oswaldo Fiallos López

Patricio Gonzalo Lascano Pazmiño

DIRECTOR DE LA DISERTACIÓN: Ing. Santiago Acurio

AMBATO, 18 de Octubre del 2003

PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

SEDE AMBATO

ESCUELA DE INGENIERIA DE SISTEMAS

DISERTACIÓN DE GRADO PREVIA LA OBTENCIÓN DEL TITULO DE INGENIERIA DE SISTEMAS

"Diseñar y construir un sistema que permita monitorear y asignar el uso de computadoras en el centro de cómputo de la PUCESA utilizando dispositivos de lectura de códigos de barras."

DIRECTOR : _____

Ing. Santiago Acurio.

William Oswaldo Fiallos López
Patricio Gonzalo Lascano Pazmiño

Indice

Dedicatoria

A Dios, porque con su infinito amor y sabiduría me ha señalado el camino a seguir para conquistar las metas que me he propuesto.

A mis padres, quienes con su constante comprensión y apoyo me ayudaron a culminar una etapa importante dentro mi vida estudiantil.

A mis hermanos, familiares y amigos, quienes con su colaboración y consejos contribuyeron para la consecución de este objetivo.

Patricio Lascano P.

Dedicatoria

A Dios, porque me hace ser una persona mejor, me hace ser completo y me ayuda a ser lo mejor que puedo; sin él, sería la mitad de lo que soy. Me ha enseñado a mi mismo que cualquier otra persona.

A mis padres, quienes, con su apoyo moral y económico me permitieron alcanzar uno mis primordiales objetivos planteados ,me han ayudado a mantener mi vida en el casillero de cosas importantes.

A mis hermanos, familiares y amigos, quienes, con su desinteresada colaboración me han enseñado cuando hay que parar y saber que siempre hay un mañana. Me han ayudado ha tener una perspectiva de la vida, y a usar todos mis talentos.

William Fiallos L.

Agradecimiento

A la Pontificia Universidad Católica del Ecuador Sede – Ambato, que nos abrió las puertas Y nos brindó la posibilidad de acumular conocimientos, los mismos que han servido de base para la realización de esta disertación.

Al director de nuestra Disertación, que con sus conocimientos y orientación se convirtió en colaborador indispensable para lograr finalizar con éxito este proyecto.

A nuestros profesores, cuyas enseñanzas nos permitieron crecer no solo en el campo intelectual, sino también como seres humanos.

Patricio Lascano P.

William Fiallos L.

CAPÍTULO I

1. EVOLUCIÓN TECNOLÓGICA Y DESARROLLO DEL HARDWARE DE LECTURA DE DATOS.

1.1. INTRODUCCIÓN.

Uno de los inventos de mayor trascendencia e incidencia dentro de las labores cotidianas en el siglo XX sin lugar a dudas, es la computadora. Hoy en día las computadoras permiten simplificar procesos debido a que realiza las operaciones requeridas en un tiempo mínimo; además los resultados obtenidos son altamente confiables.

En la lectura de información los periféricos son unidades o dispositivos a través de los cuales el ordenador se comunica con el mundo exterior, como a los sistemas que almacenan o archivan la información, sirviendo de memoria auxiliar de la memoria principal. La memoria masiva o auxiliar trata de suplir las deficiencias de la memoria central.

El ordenador es una máquina que no tendría sentido si no se comunicase con el exterior, es decir, si careciese de periféricos. Por lo que debe disponer de:

- Unidad(es) de entrada, a través de la(s) cual(es) poderle dar los programas que queramos que ejecute y los datos correspondientes.
- Unidad(es) de salida, con la(s) que la ordenador nos da los resultados de los programas.
- Memoria masiva o auxiliar, que facilite su funcionamiento y utilización.

Los dispositivos de E/S transforman la información externa en señales codificadas, permitiendo su transmisión, detección, interpretación, procesamiento y almacenamiento de forma automática. Los dispositivos de Entrada transforman la información externa (instrucciones o datos tecleados) según alguno de los códigos de entrada/salida (E/S). Así el ordenador recibe dicha información adecuadamente preparada (en binario). En un dispositivo de Salida se efectúa el proceso inverso: la información binaria que llega del ordenador se transforma de acuerdo con el código de E/S en caracteres escritos inteligibles por el usuario.

Hay que distinguir claramente entre periféricos de un ordenador y máquinas auxiliares de un determinado servicio informático. Las máquinas auxiliares no están físicamente conectadas al ordenador (su funcionamiento es totalmente autónomo) y sirven para preparar o ayudar en la confección o utilización de la información que se da a, o produce, el ordenador. Por ejemplo, hace algunos años existían máquinas autónomas para perforar tarjetas, para grabar cintas magnéticas manualmente a través de un teclado, para separar el papel continuo producido por un programa a través de la impresora, etc.

Tampoco hay que confundir periférico con soporte de información. Por soporte de información se entiende aquellos medios físicos sobre los que va la información. Por unidades o dispositivos periféricos se entiende aquellos elementos encargados de transcribir la información al correspondiente soporte.

1.2. DESARROLLO HISTÓRICO DE LOS DISPOSITIVOS DE ENTRADA SALIDA CONVENCIONALES.

A continuación se realiza el estudio de los dispositivos más comunes y más utilizados en la actualidad.

1.2.1. TECLADO.

Los teclados son similares a los de una máquina de escribir, correspondiendo cada tecla a uno o varios caracteres, funciones u órdenes. Para seleccionar uno de los caracteres de una tecla puede ser necesario pulsar simultáneamente dos o más teclas, una de ellas la correspondiente al carácter.

Al pulsar una tecla se cierra un conmutador que hay en el interior del teclado, esto hace que unos circuitos codificadores generen el código de E/S correspondiente al carácter seleccionado, apareciendo éste en la pantalla si no es un carácter de control.

Los teclados contienen los siguientes tipos de teclas:

- **Teclado principal:** Contiene los caracteres alfabéticos, numéricos y especiales, como en una máquina de escribir convencional con alguno adicional. Hay teclados que también incluyen aquí caracteres gráficos.

Teclas de desplazamiento del cursor: Permiten desplazar el cursor a izquierda, derecha, arriba y abajo, borrar un carácter o parte de una línea.

- **Teclado numérico:** Es habitual en los teclados de ordenador que las teclas correspondientes a los caracteres numéricos (cifras decimales), signos de operaciones básicas (+, -, ...) y punto decimal estén repetidas para facilitar al usuario la introducción de datos numéricos.

- **Teclas de funciones:** Son teclas cuyas funciones son definibles por el usuario o están predefinidas en un programa.

- **Teclas de funciones locales:** Controlan funciones propias del terminal, como impresión del contenido de imagen cuando el ordenador esta conectada a una impresora.

En algunos teclados la transmisión no se efectúa pulsación a pulsación sino que se dispone de un almacén de reserva o buffer (tampón) y la transmisión se efectúa a la vez para todo un conjunto de mensajes completos cuando el usuario pulsa una tecla especial destinada a activar dicha transmisión. Esta tecla recibe distintos nombres como Return, Enter, Transmit, Intro, Retorno de carro.

Entre las posibles características técnicas a contemplar a la hora de evaluar la mejor o peor adaptabilidad de un teclado a nuestras necesidades, podemos citar el número de caracteres y símbolos básicos, sensibilidad a la pulsación, tipo de contactos de las teclas (membrana o mecánico), peso, tamaño, transportabilidad. Actualmente se comercializan teclados ergonómicos, con una disposición algo original, aunque se han difundido poco, y hay discusiones sobre si es cierta la ergonomía que propugnan.

Para aplicaciones industriales existen teclados totalmente sellados que soportan ambientes agresivos, como por ejemplo aire, agua y atmósferas de vapores.

Existen varias tecnologías para la construcción de teclados de computadora entre las que se destacan:

- Teclados Mecánicos .

- Teclados Electrónicos.

1.2.1.1. Teclados Mecánicos.

Son más antiguos que los electrónicos y, en algunos casos, menos fiables y caros de construir; por ello, en la actualidad se ha optado por construir casi todos los modelos con tecnología electrónica.

Los teclados mecánicos presentaron un problema debido a que, por su tecnología de construcción, la parte mecánica de la tecla no efectuaba solo un contacto al pulsarla, sino que existía un efecto rebote sobre la superficie del contacto eléctrico que enviaba varias veces la señal al controlador del teclado.

1.2.1.2. Teclados Electrónicos.

Solucionaron ese problema creando un retardo en el controlador para eliminar las señales producidas por el rebote. Sin embargo han creado un curioso problema: el cerebro humano parece que por la costumbre de teclados anteriores, a lo que se denomina efecto QWERTY, "necesita" oír el Click de la tecla al golpear el teclado para poder trabajar más cómodamente y en los últimos modelos de teclados electrónicos se ha tenido que generar este sonido artificialmente.

Casi todos los teclados permiten que sus teclas sean redefinidas por software. Por ejemplo la tecla Ñ no existe en los teclados no españoles pero, por medio de un programa, puede configurarse el sistema informático para que se imprima en la pantalla del sistema informático esta tecla cuando se pulse en un teclado en español.

1.2.2. RATÓN (MOUSE).

El ratón o Mouse informático es un dispositivo señalador o de entrada, recibe esta denominación por su apariencia.

Para poder indicar la trayectoria que recorrió, a medida que se desplaza, el Mouse debe enviar al computador señales eléctricas binarias que permitan reconstruir su trayectoria, con el fin que la misma sea repetida por una flecha en el monitor. Para ello el Mouse debe realizar dos funciones:

En primer lugar debe generar, por cada fracción de milímetro que se mueve, uno o más pulsos eléctricos (CONVERSION ANALOGICA-DIGITAL).

En segundo lugar contar dichos pulsos y enviar hacia la interfaz "port serie", a la cual esta conectado el valor de la cuenta, junto con la información acerca de sí se pulsa alguna de sus tres teclas ubicada en su parte superior.

Existen dos tecnologías principales en fabricación de ratones: Ratones mecánicos y Ratones ópticos.

1.2.2.1. Ratones Mecánicos.

Los ratones mecánicos constan de una bola situada en su parte inferior. La bola, al moverse el ratón, roza unos contactos en forma de rueda que indican el movimiento del cursor en la pantalla del sistema informático.

1.2.2.2. Ratones Opticos.

Los ratones ópticos tienen un pequeño haz de luz láser en lugar de la bola rodante de los mecánicos. Un sensor óptico situado dentro del cuerpo del ratón detecta el movimiento del reflejo al mover el ratón sobre el espejo e indica la posición del cursor en la pantalla de la computadora.

Una limitación de los ratones ópticos es que han de situarse sobre una superficie que refleje el haz de luz. Por ello, los fabricantes generalmente los entregan con una pequeña plantilla en forma de espejo.

1.2.3. SCANNER.

Los escáneres son periféricos diseñados para registrar caracteres escritos, o gráficos en forma de fotografías o dibujos, impresos en una hoja de papel facilitando su introducción la computadora convirtiéndolos en información binaria comprensible para ésta.

El funcionamiento de un escáner es similar al de una fotocopiadora. Se coloca una hoja de papel que contiene una imagen sobre una superficie de cristal transparente, bajo el cristal existe una lente especial que realiza un barrido de la imagen existente en el papel; al realizar el barrido, la información existente en la hoja de papel es convertida en una sucesión de información en forma de unos y ceros que se introducen en la computadora.

Para mejorar el funcionamiento del sistema informático cuando se están registrando textos, los escáneres se asocian a un tipo de software especialmente diseñado para el manejo de

este tipo de información en código binario llamados OCR (Optical Character Recognition o reconocimiento óptico de caracteres), que permiten reconocer e interpretar los caracteres detectados por el escáner en forma de una matriz de puntos e identificar y determinar qué caracteres son los que el subsistema está leyendo.

Un caso particular de la utilización de un scanner, aunque representa una de sus principales ventajas, es la velocidad de lectura e introducción de la información en el sistema informático con respecto al método tradicional de introducción manual de datos por medio del teclado, llegándose a alcanzar los 1.200 caracteres por segundo.

1.2.3.1. Escáner de mano.

Es el menos costoso. Tiene un ancho de escaneado aproximadamente cuatro pulgadas, y es ideal para copiar imágenes pequeñas como firmas, logotipos y fotografías.

1.2.3.2. Escáner hoja por hoja

Un escáner de hoja por hoja produce lecturas mas confiables, es menos costoso y más compacto que uno plano. Este tipo de escáner puede solamente copiar hojas sueltas. Si se desea escanear una página de un libro, se debe arrancar.

1.2.3.3. Escáner Plano

Un escáner plano es el tipo más versátil. Es ideal para escanear páginas de un libro sin tener que desprenderlas

1.3. DISPOSITIVOS NO TRADICIONALES DE LECTURA DE DATOS.

Existen una gran cantidad de dispositivos de lectura de datos, que los vamos a clasificar en las siguientes categorías.

1.3.1. LECTOR DE CODIGOS DE BARRAS.

El lector de códigos de barra esta ampliamente difundido en el comercio y en la industria, siendo que una computadora se conecta a través de la interfaz port serie.

Posibilita la recolección de datos con rapidez, muy baja tasa de errores, facilidad y bajo costo, en comparación con la lectura visual de códigos numéricos seguida de entrada manual por teclado.

Uno de los medios más modernos, y que está tomando cada vez un mayor auge, de introducir información en una computadora es por medio de una codificación de barras verticales.

Cada vez son más los productos que llevan en su etiqueta uno de estos códigos donde, por medio de las barras verticales de color negro, se consigue una identificación para todo tipo de productos, desde libros hasta tarjetas de identificación.

Esta codificación ha sido definida de forma estándar por la Organización de Estándares Internacionales y, en ella, cada una de las líneas tiene un determinado valor dependiendo, en principio, de su presencia o ausencia y también de su grosor.

En general los códigos de barra no son descifrables por las personas. Las lectoras son las encargadas de convertirlos en unos y ceros que irán a la computadora.

Representan caracteres de información mediante barras negras y blancas dispuestas verticalmente. El ancho de las barras y espacios puede ser variable, siendo la más ancha un múltiplo de la mas angosta. En binario las barras significaran unos y los espacios ceros.

1.3.1.1. Clases de lectoras de códigos de barra.

Existen dos clases de lectoras: De haz fijo y de haz móvil. En ambos casos una fuente luminosa ilumina la superficie del código. Siendo las barras oscuras y los espacios claros, estos reflejaran mas luz que las barras. La luz reflejada es detectada por un elemento fotosensor, produciendo los espacios claros una mayor corriente eléctrica en el elemento fotosensor. Para que la lectura progrese debe existir un movimiento relativo del código respecto a la lectora o a la inversa, o bien debe existir un haz láser que se desplaza para explorar el código. Esto hace a la diferencia entre las dos clases de lectoras citadas.

La corriente eléctrica que circula por el fotosensor es proporcional a la intensidad del haz reflejado (que es la magnitud censada), que como el caso del escáner es una señal analógica. Por lo tanto, deberá convertirse en digital (unos y ceros) para ser procesada.

1.3.1.2. Diferentes tipos de lectoras:

1.3.1.2.1. Lectora manual.

Tienen forma de una lapicera, se debe desplazar de toda la longitud del código, para que un haz fijo pueda ser reflejado y censado.

1.3.1.2.2. Lectora de ranura fija.

El operador debe desplazar el código a través de una ranura de la lectora. Es de haz fijo.

1.3.1.2.3. Lectora fija con haz láser móvil.

Un rayo láser rojo anaranjado barre en un sentido a otro el código de barras decenas de veces por segundo. Un rayo láser es dirigido por un espejo móvil, que a su vez dirige el haz hacia otros espejos. Por la ventana de salida parece como si se generan muchos haces láser. Esto permite leer un código de barras que este en distintas ubicaciones espaciales respecto a la ventana citada. Estas lectoras son más exactas que las anteriores.

1.3.2. TABLETA DIGITALIZADORA.

Las tabletas digitalizadoras son unas herramientas que permiten el manejo del cursor a través de la pantalla del sistema informático y facilitan una importante ayuda en el tratamiento de los comandos de órdenes en aplicaciones de CAD/CAM (diseño asistido por computadora).

Las tabletas digitalizadoras convierten una serie de coordenadas espaciales en un código binario que se introduce en la computadora. Estas coordenadas serán manejadas posteriormente por programas de dibujo, ingeniería, etc.

La tableta suele tener impresos en su armazón pulsadores con símbolos dibujados para ejecutar de modo directo comandos que agilizan el trabajo de manejo del software.

Las tabletas digitalizadoras poseen una resolución de alrededor de una décima de milímetro y pueden manejar gráficos en dos y tres dimensiones.

Una posibilidad de manejo muy intuitiva convierte a las tabletas digitalizadoras en unas herramientas muy útiles y polivalentes en los sistemas informáticos de diseño y manejo de gráficos.

Existen diversas tecnologías de construcción de tabletas, pudiendo ser éstas:

- Tabletas mecánicas.
- Tabletas electrónicas.

Las mecánicas, debido al desgaste producido en sus componentes por el uso continuado, son menos precisas y más delicadas de manejar que las electrónicas, siendo éstas, por ello, las más extendidas comercialmente en el mercado.

1.3.3. LECTORES MAGNÉTICOS.

1.3.3.1. Lector de caracteres magnéticos

Los caracteres magnéticos se utilizan en los talones y cheques bancarios, y en las etiquetas de algunos medicamentos en algunos países, pues en España se usa el código EAN . En estos documentos se imprimen, de acuerdo con unos patrones, los caracteres que identifican el cheque o talón. La tinta utilizada es imanable (contiene óxido de hierro) y además es legible directamente por el hombre. La impresión se hace con una máquina auxiliar denominada inscriptora electrónica.

Este dispositivo ofrece una serie de ventajas como:

Permitir la captación directa de datos.

- Los documentos no necesitan cuidados especiales, se pueden doblar, escribir encima con tinta no magnética.
- Se consiguen velocidades de lectura muy apreciables.
- Los caracteres usados son legibles.

Los inconvenientes que presentan son:

- Alto costo.
- Impresión cara y específica.

1.3.3.2. Detector de bandas magneticas

Las bandas magnéticas se emplean en productos como tarjetas de crédito, tarjetas de la Seguridad Social, tarjetas de acceso a edificios y etiquetas de algunos productos. Contienen datos como números de cuenta, códigos de productos, precios, etc.

Las bandas magnéticas se leen mediante dispositivos de lectura manuales, similares a un lápiz, o por detectores situados en los dispositivos en los que se introducen las tarjetas, incluso disponibles en algunos teclados.

La ventaja de este método es que la información es prácticamente imposible de alterar una vez que se ha grabado en la banda, salvo que se le aplique un campo magnético de intensidad suficiente. Esto proporciona un notable grado de seguridad frente a los sistemas convencionales.

1.3.4. LÁPIZ ÓPTICO.

Los lápices ópticos son dispositivos de introducción de datos que trabajan directamente con la pantalla de la computadora, señalando puntos en ella y realizando operaciones de manejo de software.

Para operar con el lápiz óptico se coloca éste sobre la pantalla del sistema informático. En el momento en que el cañón de rayos catódicos de la pantalla barre el punto sobre el que se posiciona el lápiz, éste envía la información a un software especial que la maneja. El microprocesador calcula cuál es la posición sobre la pantalla de la computadora permitiendo manipular la información representada en ella.

Los lápices ópticos permiten la introducción de datos, el manejo del cursor, etc., en la pantalla de la computadora. Son una asistencia para las limitaciones de los teclados en algunas aplicaciones, sobre todo las que no son de gestión pura (creativas, etc.),

1.3.5. SENSORES TÁCTILES.

Son pantallas que pueden detectar las coordenadas (x,y) de la zona de la propia pantalla donde se acerca algo (por ejemplo, con un dedo). Este es un sistema muy sencillo para dar entradas o elegir opciones sin utilizar el teclado.

Se utiliza para la selección de opciones dentro del menú o como ayuda en el uso de editores gráficos. Con frecuencia se ve en los denominados kioscos informativos, cada vez más difundido en grandes empresas, bancos y en puntos de información urbana.

Todo digitalizador consta de tres elementos:

Tabla: Donde se ubica el dibujo a digitalizar (puede ser opaca o transparente).

Mando: Con el que el usuario debe recorrer el dibujo. Este suele tener forma de lápiz o cursor, y está unido al resto del sistema por un cable flexible. En el último caso el cursor tiene una ventana cerrada con una lupa, en cuyo interior se encuentra embebida una retícula en forma de cruz para señalar o apuntar con precisión el punto a digitalizar. El mando puede disponer de uno o varios pulsadores para controlar la modalidad de funcionamiento, forma de transmisión y selección de opciones del programa que gestiona la digitalización.

Circuitos electrónicos: Controlan el funcionamiento de la unidad.

Los digitalizadores, junto con los trazadores de gráficos (plotters) y pantallas gráficas, son elementos fundamentales de los sistemas gráficos, que tienen en la actualidad gran importancia en diversas aplicaciones de la Informática.

1.3.6. CAMARA DIGITAL

Una cámara digital permite tomar fotos que se pueden visualizar e imprimir utilizando una computadora.

La mayoría incluyen una pantalla tipo visualizador de cristal líquido (LCD), que puede utilizar para tener una vista preliminar y visualizar la fotografías.

Incluyen un cable que permite conectar la cámara a un puerto. Permitiendo transferir las fotografías.

Almacenan fotografías hasta que se las transfiera a una computadora. La mayoría tiene una memoria integrada o removible.

Memoria removible: almacenan fotografías en una tarjeta de memoria. Algunas las almacenan en un disquete regular que calza dentro de esta. Se puede reemplazar una tarjeta de memoria o disquete cuando esté llena /o.

Memoria incorporada: almacenan al menos 20 fotografías. Una vez que está llena, se las transfiere a la computadora.

Las filmadoras son unos aparatos periféricos altamente especializados que convierten información, que se les introduce en código binario, en imágenes con una calidad similar a la de una imprenta (1.600 puntos por pulgada como mínimo) o fotogramas similares a los de cinematografía.

Las filmadoras se pueden conectar a una computadora o trabajar con ellas remotamente llevando la información hasta el punto donde están por medio de un soporte magnético.

Se utilizan para grabar conversaciones y otros sonidos, utilizando programas de conferencia para comunicarse a través de Internet. Con los programas de control de voz se puede conversar en un micrófono y emplear los comando de voz para controlar la computadora.

1.3.7. TELEMÁTICA

Definimos comunicación como el proceso por el que se transporta información, la cual es transmitida mediante señales, que viajan por un medio fisico.

El termino TELEMATICA o TELEINFORMATICA conjunción de telecomunicaciones e informática se refiere a la disciplina que trata la comunicación entre equipos de computación distantes.

1.3.7.1. Sistema teleinformática.

Está constituido por:

- Equipos informáticos (computadoras y terminales), para recibir, procesar, visualizar y enviar datos.
- RED DE TELECOMUNICACIONES: Soporte para la comunicación, con medios de transmisión y circuitos apropiados.

1.4. CONCLUSIONES.

Con la evolución de la computadora, los dispositivos de lectura de datos tradicionales fueron cambiando su forma y tamaño dando mayor comodidad y eficiencia a los usuarios, que requerían que sus operaciones se realicen con rapidez y los resultados sean satisfactorios.

Se analizaron, comprendieron y reconocieron los diferentes tipos de dispositivos de entrada de datos

Conocimos la importancia, funcionamiento, beneficio y ventajas de los diferentes tipos de dispositivos de entrada de datos.

CAPÍTULO II

2. SELECCIÓN DEL DISPOSITIVO AUTOMÁTICO DE IDENTIFICACIÓN.

2.1. INTRODUCCIÓN.

En la actualidad todos y cada uno de nosotros hemos tenido contacto con algunos aparatos computarizados que se emplean para identificar objetos y personas. La intencionalidad del presente capítulo es brindar de manera sintética ¿cuáles son estos equipos y sus aplicaciones?

2.2. GENERALIDADES

Los sistemas de identificación se emplean para el manejo de información relativa a las personas y a los objetos. Para tal efecto se utilizan formas de registro magnético, óptico, sonoro e impreso.

Generalmente, estos sistemas requieren de dos componentes fundamentales: un elemento codificado que contiene la información (léase, datos procesados siguiendo alguna norma o patrón preestablecido) y un elemento con capacidad de reconocer la información.

Posteriormente, el equipo lector se comunica con una computadora donde se realizan diversos procesos; en primer lugar, los datos son decodificados, esto es, se transforman en información entendible para la computadora. A continuación, la información es verificada, comparada y aceptada para luego realizar alguna decisión lógica.

De manera cotidiana los sistemas de identificación de personas pueden ser diversos para el acceso a una cuenta en un banco, a un área restringida, a una computadora, a una línea telefónica, a una empresa, a su casa, a los controles remotos, a las tarjetas de crédito, entre otros. Gracias a que los sistemas modernos son automáticos, los procesos se agilizan, se cometen menos errores y en consecuencia se incrementa la confiabilidad y la eficiencia.

Estos sistemas también son empleados para la identificación de objetos (o en inglés se conocen como items -artículos) sobre todo cuando se destinan a usos comerciales. Cuanto mayor es la diversificación, esto es, cuando el número de artículos rebasa la capacidad de clasificación humana, más necesaria es la identificación exacta del producto. De tal manera que el industrial, el comerciante, distribuidor y cliente -conocidos en el argot mercadológico como los elementos integradores de los canales de distribución- puedan

de crédito y de forma inmediata- se imprime el pagaré correspondiente a la transacción comercial realizada.

2.2.3. EL RECONOCIMIENTO MAGNÉTICO DE CARACTERES

Estos equipos aprovechan las características físicas de los caracteres, ya que en su forma, estructura o relieve almacenan la información. Y esta última es leída por medios mecánicos o magnéticos, por lo general los caracteres son números, lo que permite al hombre el lograr su interpretación sin el empleo del equipo lector respectivo.

Algunos ejemplos son los cheques y otros instrumentos comerciales como papel moneda o sistemas mecanizados de correspondencia. Los cheques por lo general presentan estos caracteres o números impresos en la parte inferior del mismo para su procesamiento automático mediante medios magnéticos.

Otrora en Europa se empleaba un código conocido por sus siglas como CMC7, donde la figura de cada número estaba formada por siete líneas verticales y siete espacios. Mediante esta forma de impresión se integraba un código binario de unos y ceros respectivamente; y según su distribución definían los números del 0 al 9, y adicionando algunos símbolos, se podía identificar magnéticamente al banco, sucursal, número de cuenta y número de cheque. Este sistema podría ser catalogado como el antecesor del código de barras.

En los Estados Unidos y México se utiliza con el mismo objeto, el código E13B que recibe su nombre del proceso efectuado por el lector magnético, consistente en leer en sentido vertical líneas de trece milésimas de pulgada que condensan información similar a la citada para el caso europeo.

2.2.4. EL RECONOCIMIENTO ÓPTICO DE CARACTERES (OCR)

Mediante estos sistemas se reconocen caracteres impresos y cuya forma constituye la información que se desea procesar. La lectura se efectúa automáticamente mediante un haz de luz y se interpretan o convierten a través de procedimientos matemáticos en información digital (señal discreta), analógica (señal continúa) o ASCII (American Standard Code for Information Interchange -Norma Americana para la Codificación e Intercambio de Información). Este último código es el que sirve -con algunas variaciones- como medio de comunicación en el mundo de las computadoras.

La lectura de la información se produce por contacto o a distancia, el haz de luz puede ser visible o no (infrarrojo), estático o móvil, la fuente de luz puede ser policromática (incandescente) o coherente; como un láser, de estado sólido (diodos fotoemisores LED) o gaseoso (helio-neón). Estos sistemas están siendo desplazados por el código de barras para su uso comercial.

La asignación y aplicación de un número de código a cada producto no es un sistema automático de identificación, ya que tanto la marcación como la lectura de cada producto es manual y el sistema OCR en particular solamente hace referencia a los sistemas de lectura automática de información.

2.2.5. EL RECONOCIMIENTO DE VOZ

Esta tecnología es de reciente aparición y se integra por un equipo computarizado que se encuentra programado para el reconocimiento e interpretación de palabras, de un cierto vocabulario para su posterior conversión en instrucciones. También posee la capacidad para emitir palabras en forma sintetizada.

El operador del equipo cuenta con un micrófono y un auricular lo que le permite hablar y escuchar a la computadora. Por estas características este sistema se considera útil en los casos en donde los ojos o las manos estén ocupados, por ejemplo, actividades críticas en laboratorios. Es de gran ayuda para personas discapacitadas y el procesamiento de materiales.

Actualmente este tipo de sistemas ha permitido grandes avances en el terreno de las comunicaciones.

2.2.6. EL INFRARROJO

Este tipo de sistemas son de uso cotidiano en casi todos los hogares, al encender el televisor con el control remoto, al abrir la puerta de la cochera, al activar la alarma del automóvil y al utilizar algunos juguetes. En el caso de las cerraduras electrónicas, la información tiene trampas y vericuetos que hacen difícil su alteración. Estos sistemas son de transmisión e identificación simultánea, en su interior la información se traduce e interpreta de diversas formas que después de ser reconocidas permiten el acceso, pasivo o activo, al banco de comandos o a la memoria del computador. La forma común más

2.3.4.1. Lectores tipo pluma y Lectores Láser

Los lectores tipo pluma consiste de una fuente de luz y un foto diodo colocados uno cerca del otro en la punta de una pluma o varilla. Para leer un código de barras se pasa la punta de la pluma a través de todas las barras con una acción firme. El foto diodo mide la intensidad de la luz reflejada atrás de la fuente de luz y genera una forma de onda que es usada para medir los anchos de las banda y los espacios en el código de barras. Las barras oscuras en el código absorben la luz y las blancas la reflejan así que el voltaje generado por el foto diodo es un duplicado exacto del patrón de barras y espacios en el código. Esta forma de onda es decodificada por un scanner en manera similar a como se hace con el código Morse a través de un código de puntos y rayas.

Los lectores láser funcionan de la misma manera como un lector tipo pluma excepto que unos usan un rayo láser como fuente de luz y por lo regular emplean un espejo reflector o un prisma que digitaliza el rayo láser de atrás hacia delante a través del código. De la misma manera que el lector tipo pluma usa un foto diodo también se usa para medir la intensidad de la luz reflejada de la parte de atrás del código. En ambos casos de lectores la luz emitida por el lector es turnada a una frecuencia especifica y el foto diodo es diseñada para detectar solamente la misma frecuencia de luz.

Los scanners tipo pluma y láser se pueden comprar con diferentes resoluciones capaces de soportar diferentes tamaños de códigos. La resolución del scanner mide por el tamaño del punto de luz emitida por el lector. El punto de luz debe ser igual o ligeramente más pequeña que el elemento más angosto (dimensión en "x"). Si el punto es mayor que el ancho de la barra más angosta, entonces no podrá sobreponer dos o más barras al mismo tiempo, lo cual provocará que el scanner no sea capaz de distinguir claramente la transición entre barras y espacios. Si el punto es demasiado pequeño, entonces cada mancha o hueco será mal interpretada como áreas de luz lo cual hará que el código de barras no pueda ser leído. Debido a que la (dimensión "x") es extremadamente pequeña, es importante que el código de barras sea creado en un programa capaz de soportar gráficas en alta resolución.

2.3.5. CONEXIÓN DE UN LECTOR A UNA COMPUTADORA.

2.3.5.1. Lectores con emulación de teclado.

Se conecta a una computadora a través de un puerto llamado interfase de teclado. Cuando un código de barras es digitalizado, la información es transmitida a través de éste al tiempo

que fue capturada en el teclado. Algunas veces se les refiere como lectores con emulación de teclado porque físicamente hay una emulación entre el teclado y la computadora que contiene un segundo teclado. Otra gran ventaja de la emulación de teclado es que la lectura de código de barras puede ser agregada sin que haya cambios en el software; el software piensa que recibe la información como si lo hubiera hecho alguien que teclea muy rápido. Con un lector con emulación cualquier programa que acepta datos tecleados aceptará la información del código de barras sin ningún cambio.

2.3.5.2. Lectores seriales de código de barras.

Otra forma de transmitir datos de un lector a una computadora es conectándola a la computadora a traves de un puerto serial RS-232. La información del código de barras será transmitida a la computadora en un formato ASCII para aparecer como datos tecleados a la computadora. Usando una conexión de puerto serial es ideal para una computadora multiusuario. Con terminales seriales ASCII para cada usuario, el lector de código de barras puede conectarse entre la terminal y la computadora y transmitir datos ASCII justo como una terminal.

2.3.5.3. Lectores portátiles tipo batch.

Los lectores portátiles y los lectores portátiles operados por batería almacenan la información en memoria para actualizarla en la computadora varias veces. Un lector portátil tipo batch contiene un escáner, una pantalla LCD para agilizar al usuario a mejorar una tarea e incluso se pueden agregar variables de teclado como cantidades por ejemplo. Se debe contar con una cuna para actualizar al información a la computadora. Los escaners portátiles tipo batch son ideales cuando la movilidad es una factor a considerar y cuando la información recolectada no es inmediatamente necesaria. Estos escaners vienen en una variedad de estilos incluyendo los portátiles, con cables o montados en algún vehículo. Su aplicación determinará cual estilo es el mejor.

2.3.5.4. Lectores portátiles inalámbricos.

Cuando se requiere recolectar información en un lugar remoto y se necesita contar con la información inmediatamente, una solución inalámbrica es la ideal para este tipo de requerimiento. Un escáner inalámbrico está incluido dentro de una terminal, y actualiza la información hacia la computadora al mismo tiempo que es digitalizado, instantánea y

precisamente. Los productos inalámbricos le permiten al usuario escanear la información en el punto de actividad lo cual es ideal para muchas industrias.

2.3.6. CUÁL LECTOR ES EL INDICADO PARA SU APLICACIÓN ?.

Con todas las opciones disponibles, es importante entender su ambiente de trabajo y la aplicación para poder saber con precisión sus necesidades antes de tomar alguna decisión.

Responda estas preguntas para determinar cual escáner es el adecuado para sus necesidades.

- ¿ En que tipo de ambiente será usado el escáner ?
- ¿ Será trabajo rudo en una fábrica o normal en una tienda ?
- ¿ Es continuo o periódico el escaneo necesario ?
- ¿ Es a manos libres o portátil la capacidad requerida ?
- ¿ El escaneo será aplicado cerca o a distancia del código de barras?
- ¿ Cómo se realizará la conexión ?
- ¿ La información escañada será necesaria en tiempo real ?

Recuerde que hay una variedad de lectores adaptados para cada aplicación. El ultimo y mas costoso lector puede trabajar bien en una aplicación donde el escaneo es frecuente, pero ciertamente no podrá responder en un ambiente de trabajo rudo donde es necesario un lector de uso rudo.

2.3.7. COMPATIBILIDAD CON SISTEMAS.

Escanear y decodificar es una tarea respectiva que manejan el lector del escáner y el decodificador. Al mismo tiempo la información obtenida necesita llevarse a la computadora para poder ser traducida. Sin embargo, mientras su computadora y el software sean capaces de aceptar los datos del código de barras, el software existente podrá imprimir códigos de barras en sus propios sobres codificados. etiquetas, boletos, etc. Esto seria claro, lo más conveniente.

2.3.8. QUÉ APLICACIÓN DE SOFTWARE ES NECESARIA?.

Cuando desee usar código de barras en montacargas, monitores para el cuidado de pacientes, para enviar o recibir paquetes, o en un punto de venta, necesitará una aplicación de software. Es la aplicación software la que acepta los datos del código de barras y

controla el éxito de una aplicación. Es necesario pensar en un software como un socio silencioso de su computadora, organizando datos que llegan en información necesaria para manejo de los negocios.

2.3.9. VENTAJAS DEL USO DE CÓDIGOS DE BARRAS.

- Mayor precisión en la información.
- Mayor productividad de los empleados
- Mayor precisión y control de inventarios
- Rastreo preciso en actividades
- Rastreo preciso de bienes transportados
- Rastreo de documentos durante un proceso
- Mejor control de entradas y salidas
- Rapidez en la captura de datos
- Reducción de errores
- Reducción de las mermas
- Los equipos de lectura e impresión de códigos de barras son flexibles y fáciles de conectar e instalar.

2.3.10. ¿QUIÉN DEBE EMPLEAR LA CAPTURA DE DATOS POR CÓDIGO DE BARRAS?.

Casi cualquier negocio se puede beneficiar con la tecnología de captura de datos por código de barras. No importa si lo fabrica, mueve o comercializa (o todo junto), la captura de datos por código de barras, tiene mucho que ofrecer. Aquí presentamos sólo algunos ejemplos de cómo un sistema de manejo de datos puede mejorar su productividad y rentabilidad.

Manufactura

Los fabricantes pueden acoplar estrechamente las operaciones del almacén y de la planta para apoyar las técnicas actuales de fabricación "justo a tiempo". Su sistema será completamente compatible con su sistema de Planificación de Requisitos de Manufactura, sus Sistemas de manejo de almacén o sus sistemas de ejecución de manufactura.

Transporte

Las compañías de transporte pueden manejar mejor tanto los activos fijos como los móviles. Los sistemas de transporte integran la conectividad, los sistemas de posicionamiento global, las computadoras móviles, lectores de código de barra y el software más novedoso para enlazar todos sus almacenes, distribución y operaciones de transporte. El resultado: costos más bajos y mejores servicios al cliente.

Venta al por menor

Los minoristas pueden controlar el flujo de inventario desde el puerto hasta el almacén y fuera de la tienda. Las aplicaciones de software en la tienda y en el almacén con comunicación inalámbrica, le ayudan a los minoristas a incrementar la productividad. Por ejemplo: pueden aprovechar los sistemas automáticos de disminución y reabastecimiento de existencias; y mejor manejo de precios, control de inventario y movimiento de la mercancía.

2.3.11. ¿PORQUÉ DEBO CONSIDERAR CAMBIAR A LA TECNOLOGÍA DE CAPTURA DE DATOS POR CÓDIGO DE BARRAS?.

Cuando se ponga a trabajar el poder de la captura de datos por código de barras, se verá como disminuyen los costos, se incrementa la precisión y mejora la productividad. Es mucho más rápido capturar los datos a la velocidad de la luz que hacerlo con lápiz y papel. Lo mejor es que la captura de datos por código de barras es extremadamente fácil y rápida de usar. De hecho, puede ser tan fácil como conectar un scanner a la computadora.

2.3.12. ¿CUÁNTO COSTARÁ REALMENTE LA CAPTURA DE DATOS POR CÓDIGO DE BARRAS?.

El aplicar un sistema podrá medir el costo de la tecnología en relación con las horas - hombre que usted ahorrará. Le asombrará lo poco que cuesta reducir los errores de entradas manuales hechas por empleados e incrementar la eficiencia. No olvide que los scanners y las terminales para la captura de datos por código de barras son extremadamente versátiles. El mismo hardware se puede usar para diferentes labores de captura de datos por código de barras en muchos departamentos diferentes.

2.4. CUADRO COMPARATIVO DE LOS DISPOSITIVOS DE LECTURA DE CÓDIGO DE BARRAS.

Lectores Tipo Escáner

Fig. 1 Lectores Tipo Escáner.

Los lectores de códigos de barras captan los datos de entrada al sistema informático al pasar por delante de un sensor óptico la serie de barras verticales. Dispositivos utilizados en el control de almacenes, bibliotecas, etc. Existen diversos modelos y menos costoso que un escáner láser.

Posee importantes características.

Lectores Tipo Escáner Láser.

Fig. 2 Lectores Escáner Láser.

Se diferencian de los anteriores ya que posee un sensor de infrarrojos, dando mayor rapidez para la lectura de los códigos de barras. Por sus grandes facilidades para su utilización su costo es muy alto.

Lectores Especiales

Fig. 3 Lectores Especiales

Los lectores especiales tienen la característica particular de almacenar códigos. Son dispositivos altamente versátiles, algunos de estos permiten el rastreo de productos, transferencia de documentos, inventarios, auditorias, etc.

Lectores Tipo Phasser

Fig. 4 Lectores Tipo Phasser.

Este tipo de lectores de códigos de barras se utilizan especialmente en el sector industrial, debido a que trabaja con cualquier tipo de ambiente donde se encuentre agua, polvo y elementos externos.

Lectores Verticales

Fig. 5 Lectores Verticales.

Dispositivos que permiten mejorar el rendimiento específicamente en aplicaciones verticales. Tienen un alto volumen simétrico de barrido lo que hace que se conviertan en una buena opción para las empresas comerciales.

Lectores de Mesa.

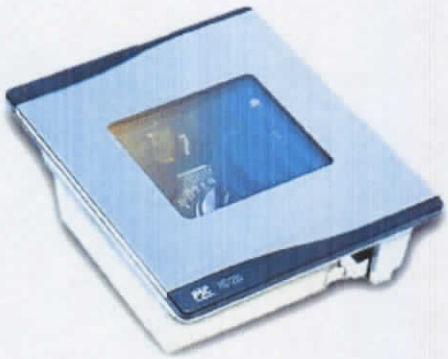


Fig. 6 Lectores de Mesa.

Sus características principales son: Lectura simultánea de 360 sin necesidad de orientar la etiqueta con el código de barras. Compatible con todas las básculas de caja adaptable; estas características lo convierten en uno de los mejores del mercado

Tabla 1. Cuadro comparativo de los dispositivos de lectura

2.5. SELECCIÓN DEL HARDWARE DE LECTURA DE DATOS.

Luego de haber realizado el análisis de los dispositivos automáticos de identificación, hemos optado por los dispositivos de lectura de códigos de barras ya que nos brinda varias ventajas que lo mencionamos a continuación.

- El código de barras almacena datos que pueden ser reunidos de manera rápida y con una gran precisión
- Los códigos de barras ofrecen con un método simple y fácil la codificación de información de texto que puede ser leída por lectores electrónicos de bajo costo.
- El software existente en el medio podrá imprimir códigos de barras de cualquier tipo.

- Mayor precisión en la lectura de información.
- Mayor precisión y control
- Rastreo preciso en actividades
- Rastreo de documentos durante un proceso
- Mejor control de entradas y salidas
- Rapidez en la captura de datos
- Reducción de errores
- Los equipos de lectura e impresión de códigos de barras son flexibles y fáciles de conectar e instalar.

Entre los diferentes tipos de lectores de códigos de barra hemos seleccionado el lector tipo escáner por las siguientes razones:

1. Este dispositivo se adapta perfectamente a los requerimientos de nuestro sistema.
2. No se requiere de un software adicional para su implementación.
3. Es el escáner de alta calidad perfecto para una lectura de contacto, rápida y exacta.
4. Es más liviano, robusto y menos costoso.
5. Diseño ergonómico y operación confortable.
6. Se adapta a la mayoría de estándares de códigos de barras.

Casi cualquier negocio se puede beneficiar con la tecnología de captura de datos por código de barras. No importa si lo fabrica, mueve, comercializa o controla, la captura de datos por códigos de barras, tiene mucho que ofrecer, puede mejorar su productividad, eficiencia, control y rentabilidad.

2.6. CONCLUSIONES.

La selección de dispositivo automático de identificación es de gran relevancia para nuestra disertación, ya que a través del análisis de las características de cada uno, nos ha permitido determinar el que más se ajusta a las necesidades del sistema. Los parámetros que se han considerado son: Operatividad, Mantenimiento, Compatibilidad y Costo.

CAPÍTULO III

3. ESTUDIO DE FUNCIONES Y PROCEDIMIENTOS PARA EL CONTROL AUTOMATICO DE PROCESOS.

3.1. COMUNICACIÓN DE PROCESOS.

3.1.1. CONCEPTOS.

Un proceso se puede definir simplemente como un programa en ejecución. Esto incluye los valores activos del contador, registros y variables del programa. De manera conceptual cada proceso tiene su propia CPU virtual. En la realidad la CPU verdadera alterna entre los procesos.

La diferencia entre un programa y un proceso es sutil pero también crítica. Por ejemplo, cuando una persona está cocinando tiene una receta y una cocina abastecida con los ingredientes necesarios para preparar su platillo. En esta analogía, la receta es el programa, la persona que cocina es la CPU y los ingredientes son los datos de entrada. El proceso es la actividad en la que la persona lee la receta, busca los ingredientes y cocina.

La idea clave es que un proceso es una actividad de cierto tipo. Tiene un programa, entrada, salida y estado. Un solo procesador puede ser compartido entre varios procesos, con cierto algoritmo de planificación, que se utiliza para determinar cuando detener el trabajo en un proceso y dar servicio a otro distinto.

3.1.1.1. Thread.

Windows introduce las hebras (thread), los principales objetos con los que trata el planificador del sistema.

La hebra es la unidad básica de la planificación en Windows. Es el componente real de un proceso que se ejecuta en un momento dado. Se ejecuta en el espacio de direcciones del proceso y utiliza los recursos asignados a dicho proceso.

3.1.1.2. Que es una Hebra?

· Es un camino de ejecución dentro de un proceso.

· Se puede crear mediante cualquier aplicación de 32 bits de windows o VxD que se ejecute en Windows.

· Tiene su propio almacenamiento de pila y contexto de ejecución (principalmente, los registros del procesador) privados.

· Comparte la memoria asignada al proceso padre.

· Puede ser una de las muchas hebras concurrentes creadas por un único proceso.

La propiedad de los recursos corresponde al proceso, no a las hebras. Las hebras utilizan los recursos, pero el proceso conserva la propiedad. Por ejemplo, si una aplicación solicita el uso de un puerto, es el proceso quien controla dicho puerto. Cualquiera de las hebras de ese proceso puede utilizar el puerto, pero una hebra no puede solicitar el uso del mismo.

En un programa de múltiples hebras, el programador tiene la responsabilidad de garantizar que las diferentes hebras no interfieran unas con otras. Esto puede conseguirse utilizando los recursos compartidos de forma que no se entre en conflicto con otra hebra que esté utilizando el mismo recurso.

En particular, el hecho de que las hebras compartan todo el código y los datos globales con su proceso padre significa que establecer una nueva hebra implica hacer cantidades mínimas de asignación de memoria. Cuando Windows carga una aplicación y crea las estructuras de los datos de los procesos asociados, el sistema establece el proceso como una sola hebra. Muchas aplicaciones utilizarán sólo una única hebra durante toda su ejecución, pero una aplicación puede (y muchas lo hacen) utilizar otra hebra para realizar alguna operación no prioritaria de corta duración.

En Windows, los servicios para hebra sólo están disponibles para las aplicaciones de 32 bits y los VxD. Las VM MS-DOS y otras aplicaciones de 16 bits para windows anteriores no pueden llamar API de hebras. Una VM MS-DOS representa una sola hebra: una VM MS-DOS es un proceso que es una hebra. Cada aplicación de 16 bits utiliza una única hebra para su ejecución y con ello se conserva el modelo de multitarea cooperativa de las aplicaciones anteriores de windows.

Cualquier aplicación de 32 bits de windows o VxD puede crear hebras adicionales y windows puede planificarlas con derecho preferente.

3.1.2. ESTADOS DE UN PROCESO.

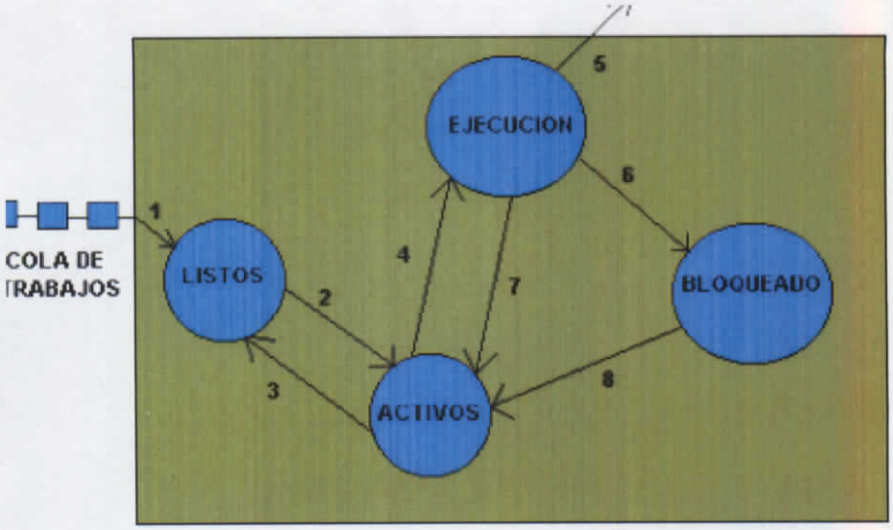


Fig. 7. Estados de un proceso

La figura ilustra los diferentes estados de un proceso asociado al usuario en un sistema con memoria virtual:

Cuando el usuario somete un trabajo al sistema, este debe esperar en una cola hasta que el sistema decide convertirlo en un proceso. Cuando se crea se le asocia al proceso un PCB (Process Control Block) y en particular una PEP, se carga el programa en la memoria y se le asignan los recursos que necesita para su ejecución excepto memoria real y el procesador. Este estado corresponde a la transición (1). El proceso en estado listos.

Una vez un proceso está listo, puede recibir memoria real en cualquier momento, en cuyo caso pasa al estado de activos (transición 2).

Estando en el estado activo, al proceso se le puede retirar la memoria real, en cuyo caso vuelve al estado listos (transición 3). Esta se puede dar porque, por ejemplo llega un proceso más prioritario que él y no hay memoria real para asignarle, en cuyo se ve perjudicado pues el sistema le retira la memoria real asignada y se la asigna al más prioritario.

Estando también en el estado activo, un proceso es candidato para lograr el uso del procesador. Cuando esto ocurre, el planificador lo selecciona y le asigna el procesador. Esto corresponde a la transición 4.

Cuando un proceso está en ejecución, puede ocurrir cualquiera de las tres siguientes situaciones:

El proceso acaba, en cuyo caso se le retira el procesador, se liberan los recursos asignados y se destruye el proceso (transición 5)

El proceso va a realizar una entrada/salida o debe esperar la ocurrencia de un evento, en cuyo caso se le retira el procesador y pasa al estado de bloqueado (transición 6)

Se le acaba el quantum de tiempo asignado para utilizar el procesador y aun no ha concluido su procesamiento, en cuyo caso se le retira el procesador y proceso vuelve al estado activo (transición 7).

Cuando el proceso está bloqueado y ocurre el evento por el que estaba esperando, pasa de nuevo al estado activo (transición 8)

3.1.3. PROCESOS CONCURRENTES.

Se dice que dos o más procesos son concurrentes si están activos simultáneamente. Gráficamente, se tiene:

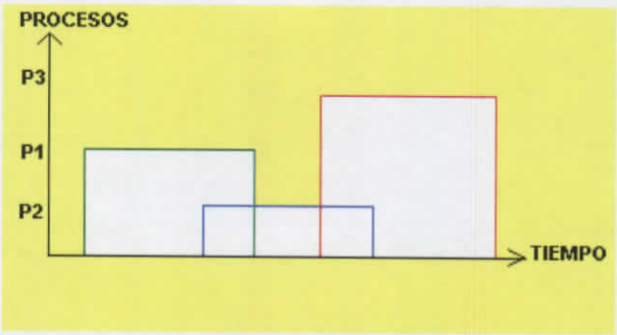


Fig. 8. Procesos Concurrentes

Los procesos P1 y P2, así como P2 y P3 son concurrentes, en cambio P3 y P1 no lo son.

Generalizando se puede decir que cuando hay multiprogramación hay una alta probabilidad de tener concurrencia.

Los procesos no concurrentes no presentan problemas de coordinación, pues se ejecutan en períodos diferentes de tiempo y por lo general no hay comunicación entre ellos.

Los procesos concurrentes pueden ser independientes si no hay interacción directa entre ellos o dependientes si la hay.

Para los procesos concurrentes dependientes deben crearse mecanismos de comunicación y coordinación entre ellos.

3.1.4. EXCLUSIÓN MUTUA.

El problema de la exclusión mutua se debe a que ciertos recursos (denominados "recursos críticos") deben ser utilizados por un solo proceso a la vez. Por ejemplo una impresora.

La exclusión mutua garantiza que si un proceso utiliza un recurso crítico, los demás procesos no puedan utilizarlo.

Para ilustrar el problema considere el siguiente caso: Suponga que existen dos procesos P1 y P2 que desean utilizar la impresora y que no existe spooling.

3.1.5. SINCRONIZACIÓN ENTRE PROCESOS.

La sincronización entre procesos puede definirse como la necesidad que tienen algunos procesos de bloquearse en determinadas circunstancias y ser despertados cuando ocurren ciertos eventos.

Un caso típico en el cual se requiere sincronización ocurre cuando un proceso inicia una lectura y va a utilizar en la siguiente instrucción la información leída. En este caso se debe esperar a que la operación de E/S termine para poder continuar la ejecución.

Uno de los problemas más conocidos es el de los lectores-redactores, dos tipos de procesos, que desean tener acceso a un mismo registro de una base de datos. Los lectores desean consultar el registro y los redactores desean modificarlo. Las lecturas no presentan ningún tipo de consistencia y por lo tanto pueden realizarse simultáneamente. Por otra parte las escrituras deben realizarse en exclusión mutua, ya que si un proceso está actualizando el registro, ningún otro proceso puede tener acceso a él.

Además de ser un problema de exclusión mutua, también es de sincronización pues los procesos que tratan de usar el registro mientras esté ocupado por un redactor, deben bloquearse y los que lo liberan deben despertar a los que están bloqueados.

En este caso, se va a suponer que los lectores tienen prioridad absoluta sobre los redactores, es decir, estos solo pueden tener acceso al registro si no hay lectores esperando tener acceso a él. Ahora, si un redactor utiliza el registro, los lectores deben esperar a que termine.

3.1.6. INTERBLOQUEOS.

Un sistema está interbloqueado cuando dos o más procesos esperan por un evento que no va a ocurrir.

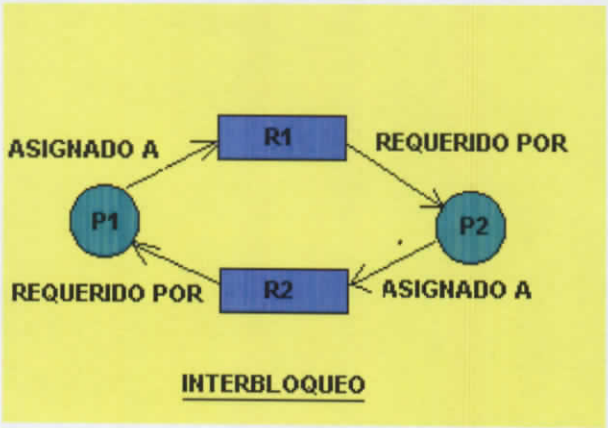


Fig. 9. Interbloqueo

Por ejemplo, suponga que existen dos procesos P1 y P2 que comparten dos recursos R1 y R2. P1 solicita el recurso R2 mientras tiene asignado R1 y simultáneamente P2 solicita R1 mientras tiene asignado R2. En este caso se presenta interbloqueo.

3.1.6.1. Soluciones al Interbloqueo

En general existen dos tipos de soluciones:

- Prevención (evitar que ocurra).

- Tratamiento (permitir que ocurra, detectarlo y corregirlo).

3.1.6.1.1. Prevención

Existen dos tipos de prevenciones: una estática, cuya solución se basa en un conjunto de reglas y otra dinámica, en la cual se evita que ocurra, no asignando recursos cuando ello puede conducir a interbloqueos.

La prevención estática se basa en el hecho que para que ocurra un interbloqueo se deben dar las siguientes cuatro condiciones en forma simultánea:

c1) Los recursos se utilizan en exclusión mutua.

c2) Los recursos, una vez asignados, no pueden ser retirados a los procesos.

c3) Los procesos esperan por la asignación de recursos mientras tienen asignados otros.

c4) Existe una cadena circular de procesos en la cual cada uno tiene asignado uno o más recursos que son solicitados por otros procesos.

Si se desea evitar los interbloqueos, se debe impedir que se cumpla al menos una de las cuatro condiciones.

La condición c1) no se puede evitar, ya que como se mencionó previamente, los recursos se deben utilizar en exclusión mutua.

Para evitar la condición c2) se puede actuar de la siguiente forma: Si un proceso tiene asignado un recurso R1 y le es negado otro recurso R2, entonces se le retira el recurso R1. Sin embargo, esta solución puede llevar a que la ejecución de un proceso se prolongue indefinidamente, debido a que nadie puede garantizar que termine en un tiempo determinado.

Para evitar la condición c3), se pueden asignar todos los recursos necesitados por el proceso en el momento de su creación, sin embargo, esto conlleva a un manejo poco eficiente de los mismos.

Para eludir la condición c4) se puede hacer una asignación de recursos por niveles. En este caso se le asigna a cada tipo de recurso un nivel y se establece que los recursos deben ser asignados en orden ascendente de nivel.

La prevención dinámica consiste en analizar cada vez que se va a asignar un recurso, si tal asignación puede conducir a un interbloqueo (situación no-segura), en cuyo caso no se asigna. En caso contrario (situación segura) el recurso se asigna.

Por ejemplo suponga que en el sistema existen múltiples recursos de un solo tipo y que existen cuatro procesos que quieren usarlos.

PROCESO	RECURSOS ASIGNADOS	MAXIMO RECURSOS NECESITADOS
P1	4	8
P2	3	7
P3	3	5
P4	0	5

Tabla 2. Procesos

Suponga además que la cantidad de recursos disponibles es 2. Si el proceso P3 solicita estos recursos se le pueden asignar, ya que tal asignación conduce a una situación segura, puesto que P3 puede terminar y liberar los 5 recursos asignados y con ellos se garantiza que todos los demás procesos pueden terminar. Sin embargo, si estos 2 recursos son solicitados por P2, la situación no es segura si se le asignan, ya que no se puede garantizar que los procesos terminen satisfactoriamente.

El hecho de que la situación no se segura no implica necesariamente que se vaya a presentar interbloqueo, sino que puede ocurrir.

3.1.6.1.2. Tratamiento

El tratamiento se basa en el hecho de detectar el interbloqueo y corregirlo. La corrección se limita en la mayoría de los casos a seleccionar uno o más procesos de baja prioridad y destruirlos.

En el peor de los casos, se debe reiniciar el sistema. Como se puede suponer, este tipo de solución puede conducir a inconsistencias en el sistema, además de ser una solución demasiado costosa.

3.1.7. PLANIFICACIÓN.

3.1.7.1. Definiciones

Cuando dos o más procesos son ejecutables desde el punto de vista lógico, el sistema operativo debe decidir cuál de ellos debe ejecutarse en primer término. Esta labor la lleva a cabo el planificador.

Un buen algoritmo de planificación debe cumplir con ciertos criterios:

1. Equidad: garantizar que cada proceso obtiene una proporción justa de CPU.

2. Eficacia: mantener ocupada la CPU al 100%.

3. Tiempo de respuesta: minimizar el tiempo de respuesta para los usuarios interactivos.

4. Tiempo de regreso: minimizar el tiempo que deben esperar los usuarios por lotes para obtener sus resultados.

5. Rendimiento: maximizar el número de tareas procesadas por unidad de tiempo (hora).

Un buen planificador además debe tener en cuenta que el comportamiento de cada proceso es impredecible. Algunos utilizan una gran cantidad de tiempo en espera de operaciones de E/S, mientras otros utilizan varias horas la CPU en forma ininterrumpida, si tienen la posibilidad.

Cuando el planificador deja que un proceso se comience a ejecutar, no está en capacidad de saber cuánto tiempo transcurrirá hasta que el proceso se bloquee, ya sea para E/S, por un semáforo, o por alguna otra razón. Para garantizar que ningún proceso se ejecute un tiempo excesivo, los computadores recurren a interrupciones generadas por el reloj y en cada una de ellas el sistema operativo decide si el proceso que se ejecuta en ese momento tiene permiso para continuar o si debe suspenderlo para dar el control a otro proceso.

Asignación del procesador con derecho preferente. Esta planificación pone a disposición del sistema operativo un control completo sobre qué proceso se ejecuta a continuación y por cuánto tiempo. En cualquier momento, el planificador puede despojar de la CPU al proceso en curso y asignársela a otro. Este acto de apropiación se producirá como respuesta directa a un suceso que demande un cambio de atención. El planificador asocia una prioridad con cada proceso en ejecución. Si se produce un suceso de interés para un

proceso de alta prioridad, el planificador relega al proceso actual y da preferencia de ejecución al proceso de alta prioridad. El planificador consigue el control del sistema tanto cuando un proceso libera la CPU como cuando se produce una interrupción de reloj. La mayoría de los sistemas programan el reloj para que pulse de 20 a 50 veces por segundo y al pulso final es cuando el planificador consigue el control y puede relegar al proceso en ejecución.

Las prioridades de los procesos se recalculan frecuentemente. En el cálculo de prioridades se tiene en cuenta la duración del lapso de tiempo. No tendría sentido asignar continuamente la CPU a un proceso y luego expulsar el proceso cuando tan sólo haya ejecutado unas pocas instrucciones.

Asignación cooperativa. Esta depende de la ayuda de los programadores de aplicaciones para mantener el sistema ejecutándose sin sobresaltos.

Con la técnica cooperativa el planificador puede conmutar entre procesos sólo cuando el proceso actualmente en ejecución entrega la CPU. Según la buena práctica de programación para sistemas de multitarea cooperativa las aplicaciones deben devolver regularmente la CPU al sistema operativo.

La cesión es una técnica que por parte de una aplicación permite que el planificador ejecute un proceso de mayor prioridad en caso de que haya uno preparado.

3.1.7.2. Algoritmos de Planificación

Algunos de los algoritmos de planificación más usados son los siguientes:

1. **SPT (Shortest Processing Time).-** Elige al proceso con menor tiempo de procesamiento. Como no es posible conocer este valor de antemano, se hace necesario realizar aproximaciones a él.

2. **Round Robin. (Reciclaje).-** En este método cada proceso tiene asignado un intervalo de tiempo de ejecución, llamado quantum. Si el proceso continúa su ejecución al final de su quantum, otro proceso se apropia de la CPU. Si el proceso está bloqueado o ha terminado antes de consumir la totalidad de su quantum, se alterna el uso de la CPU (claro, si el proceso se ha bloqueado). Cuando el quantum de un proceso se agota y no ha

terminado su procesamiento, vuelve al estado activos, a competir de nuevo por el uso del procesador.

Se denomina tiempo de alternancia de procesos o de contexto, al tiempo que emplea el sistema en realizar las labores inherentes a guardar el contexto de ejecución de un proceso y montar el de uno nuevo. Si este tiempo es mayor que el quantum asignado a cada proceso, la eficacia del sistema se ve seriamente afectada. Ahora bien, si el quantum dado a cada proceso es muy grande, se pueden presentar demoras en la respuesta a los usuarios interactivos. Un quantum cercano a los 100 ms. es con frecuencia el elegido.

3. Reciclaje de varias colas.- Es una modificación del anterior. La idea es definir diferentes colas para cada tipo de programa (largos o cortos).

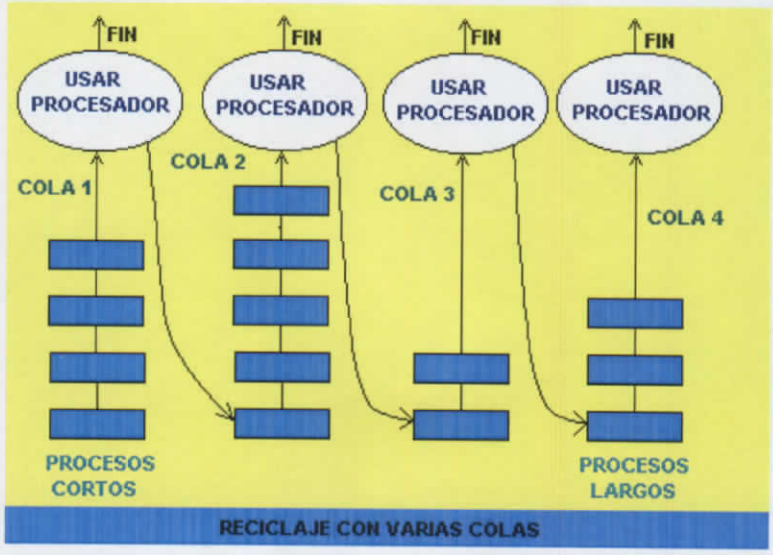


Fig. 10. Reciclaje con varias colas

Todo proceso que entra al sistema va a la cola 1 (asociada a procesos cortos) en donde permanece un tiempo preestablecido. Si transcurrido ese tiempo no ha terminado su ejecución, ni realizado ninguna operación de E/S, pasa a la cola 2, y así sucesivamente para las demás colas. Cuando el proceso realiza una E/S sale de la cola y cuando termine la operación vuelve al final de la misma. De esta forma los procesos largos o intensivos en E/S van a la última cola.

El procesador se asigna primero a los procesos de la cola 1, luego a los de la cola 2 y así sucesivamente. El quantum va aumentando en la medida que asciende en el número de cola. Esto con el fin de que los programas largos (los de la última cola), puedan recibir

suficiente tiempo de procesador antes de que este les sea retirado, pues sino su ejecución sería muy lenta. Sin embargo esto tiene el inconveniente de que le da prioridad a los procesos largos. Para solucionar esto, entonces se le dan varias vueltas a la cola 1, luego varias vueltas a la cola 2, pero menos veces que a la cola 1 y así sucesivamente.

3.1.7.3. Planificación en Windows

El componente del sistema operativo que gestiona la multitarea en Windows es el planificador.

El planificador trata principalmente con el tiempo y los sucesos. Un proceso de Windows consigue un lapso de tiempo que determina durante cuánto tiempo puede usar la CPU. Al final del lapso del proceso, el planificador decide si permite que un proceso diferente utilice la CPU. Los sucesos influyen en las decisiones del planificador. Para éste, una pulsación del ratón es un suceso que puede significar que debe asignar la CPU al proceso al que pertenece la ventana sobre la que se produjo la pulsación del ratón, o el planificador puede considerar que la conclusión de una transferencia de datos de red realizada de forma simultánea es un suceso que debe recibir una mayor atención que la pulsación del ratón. En ese caso, el proceso que gestiona la red conseguiría la CPU y el otro proceso tendría que esperar.

Windows utiliza sistema de multitarea cooperativa y multitarea con derecho preferente o asignación prioritaria, para mantener la compatibilidad con los programas existentes.

La planificación es el proceso por el cual se determina qué hebra debe utilizar el procesador. Este proceso se basa en una unidad de tiempo predeterminada. En la práctica, el lapso de tiempo real depende de la configuración del sistema.

Planificación de la máquina virtual. El proceso de planificación en Windows está tan relacionado con las máquinas virtuales que es apropiado examinar la planificación como parte del estudio del VMM.

Dentro del VMM de Windows existen dos planificadores:

· **Planificador principal:** Que es el responsable de calcular las prioridades de las hebras.

· **Planificador de lapsos de tiempo:** Que se responsabiliza de la asignación de los lapsos. Decide qué porcentaje del tiempo disponible del procesador se asigna a cada diferente

hebra. Si una hebra no recibe tiempo de ejecución, quedará suspendida y no podrá volver a ejecutarse hasta que el planificador vuelva a evaluar la situación.

Los planificadores de Windows. Cuando hay varios procesos activos en el sistema, es necesario disponer de algún método para determinar el orden en que se ejecutan las hebras.

Cada hebra tiene una prioridad base. La prioridad determina cuando se ejecuta una hebra en relación con otras hebras del sistema. Se otorga el uso del procesador a la hebra que tenga la mayor prioridad. La prioridad base de las hebras de un proceso puede alterarse en dos niveles ascendentes o descendentes.

Hay 32 niveles de prioridad, que abarcan desde el nivel de prioridad más bajo, de 0 hasta el 31. La prioridad base de las hebras de un proceso se puede cambiar en un máximo de dos niveles hacia arriba o hacia abajo. Uno de los métodos para cambiar la prioridad base de un programa es que el planificador modifique la prioridad de las hebras.

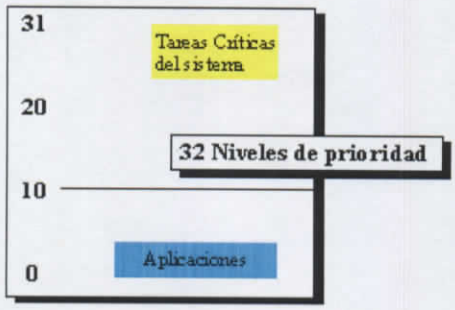


Fig. 11. Prioridades de una hebra

A continuación se explica cómo trabaja el proceso de planificación:

1. El planificador principal examina cada hebra presente en el sistema y calcula un valor de prioridad de ejecución para la hebra, un entero entre 0 y 31.

2. El planificador principal suspende cualquier hebra que tenga un valor de prioridad de ejecución inferior al valor más alto. (Si dos hebras tienen el valor de prioridad de ejecución igual a 20 y las demás tienen un valor menor de 20, entonces 20 es el valor más alto hasta que se vuelva a calcular la prioridad). Una vez que suspende una hebra, el planificador principal no le presta más atención en el cálculo de prioridad durante el lapso en cuestión.

3. Después, el planificador de lapsos calcula el porcentaje de lapso a asignar a cada hebra, usando los valores de prioridad y conociendo el estado actual de la VM.

4. Se ejecutan las hebras. Por omisión, el planificador principal volverá a evaluar las prioridades cada 20 milisegundos.

También entran en juego en este proceso tres señalizadores de control que mantiene cada VM:

VM Stat_Exclusive. Indica al planificador que la VM en cuestión debe recibir el 100 por 100 del lapso siguiente; ninguno de los otros dos señalizadores restantes estará activado.

Alguno de los dos indicadores restantes: VM Stat_Background y VM Stat_High_Pri_Background debe estar activado para que el planificador pueda garantizar una VM de segundo plano alguna asignación en el lapso siguiente; de otro modo, la VM de primer plano obtendrá toda la asignación.

Planificación dentro de la máquina virtual del sistema. Todas las hebras de las aplicaciones windows se ejecutan dentro del contexto de la VM del sistema. La VM del sistema es la única VM que admite múltiples hebras: una por cada aplicación windows de 16 bits, y al menos una por cada aplicación windows de 32 bits. A partir del algoritmo de planificación, la VM del sistema frecuentemente contendrá múltiples hebras no ociosas con prioridades igual de altas.

Para la gestión de esta situación, el planificador de lapsos adopta una política de planificación igualitaria para asegurar una asignación justa de tiempo de ejecución entre las hebras que tienen igual prioridad.

Una vez que una hebra dentro de la VM del sistema consume su tiempo asignado de ejecución, el planificador la coloca al final de la cola que contiene las hebras de igual prioridad.

Esta técnica asegura que todas las hebras con el nivel más alto de prioridad tienen una misma oportunidad de consumir su tiempo de procesador.

Control del planificador. Dos influencias diferentes controlan el planificador.

Una de ellas es la de sus propios algoritmos internos que intentan proporcionar un entorno de multitarea uniforme donde cada hebra comparte equitativamente el tiempo del procesador.

La otra influencia en el planificador son las llamadas directas a los servicios del sistema que podrían hacer los VxD.

Internamente, el planificador utiliza tres técnicas como ayuda para lograr su objetivo de distribución equitativa del tiempo del procesador, con el fin de dar una impresión de respuesta rápida y uniforme:

Prioridad dinámica estimulada. Permite al planificador principal subir o bajar la prioridad de una hebra. Por ejemplo, una pulsación del teclado o del ratón indica que hay que estimular la prioridad de la hebra receptora.

Decaimiento de tiempo. Hace que la prioridad estimulada de una hebra vuelva, gradualmente, a su valor normal.

Herencia de prioridad. Convierte rápidamente una hebra de prioridad baja en una hebra de prioridad más alta. Normalmente se invierte la prioridad de una hebra para permitir que una hebra con una prioridad baja termine rápidamente su utilización de un recurso exclusivo por el que esperan hebras de prioridad más alta.

3.1.8. PASO DE MENSAJES

3.1.8.1. Paso de Mensajes en Windows

Windows utiliza un modelo de "paso de mensaje", para controlar las aplicaciones.

Los mensajes se generan siempre que se produce un evento. Por ejemplo, si el mouse (ratón) inicia o detiene su movimiento, si se presiona una tecla o si un búfer de hardware obtiene datos, se produce una interrupción. Estas interrupciones se convierten en mensajes. Las aplicaciones Windows también crean mensajes para solicitar al sistema operativo que realice un servicio o que transmita información. Los mensajes se colocan en la cola de mensajes adecuada.

Este procesamiento asíncrono de mensajes significa que cada cola se procesa independientemente. Si una aplicación se queda bloqueada (y ya no lee sus mensajes) ello no impide que las demás colas procesen sus mensajes.

Las aplicaciones basadas en Windows de 32 bits disponen de una cola de mensajes para cada tarea (o thread) que esté ejecutando la aplicación.

Todas las aplicaciones basadas en Windows de 16 bits comparten una cola de mensajes común. Si una aplicación basada en Windows de 32 bits se queda bloqueada, todas las aplicaciones basadas en Windows de 16 bits que estén en ejecución tendrán sus mensajes bloqueados hasta que se quite la aplicación colgada.

Puesto que las aplicaciones basadas en MS-DOS no utilizan el diseño de paso de mensajes, no disponen de ninguna cola de mensajes.

Un problema, con windows 3.1 es que cada aplicación windows recupera mensajes de una sola cola de mensajes del sistema. Esta cola de mensajes contiene los mensajes brutos recibidos en una forma procesada que resulta conveniente para ser consumida por la aplicación, además de otros mensajes que fluyen por el sistema.

Siempre que un proceso solicita un mensaje, el sistema simplemente distribuye el mensaje a la cabeza de la cola. Hasta que el proceso cede el control de la CPU, el sistema no intenta distribuir más mensajes. Puesto que no hay expulsión en windows 3.1, si falla una aplicación, se interrumpe el flujo de mensajes y consecuentemente el sistema.

Aunque windows 3.1 proporcionase un entorno de multitarea con derecho preferente, una única cola de mensajes también causaría el mismo problema.

Para evitar esta clase de situaciones, windows incluye múltiples colas de mensajes. Dado que el flujo eficiente de mensajes es vital para un buen tiempo de respuesta y una multitarea uniforme. Asegura que una sola aplicación errante no puede bloquear todo el sistema. La técnica de colas múltiples se denomina desincronización de entrada y la Figura muestra cómo funciona.

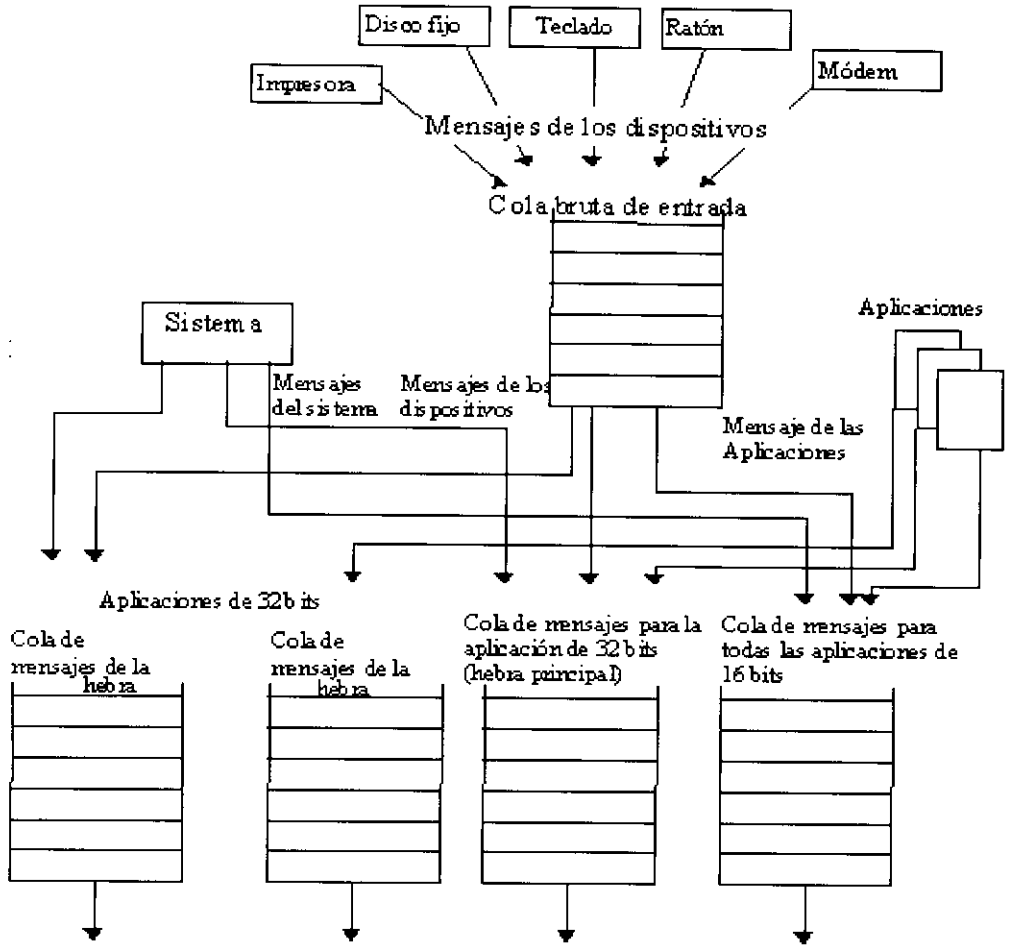


Fig. 12. Entrega de mensajes con múltiples colas.

En Windows, los nuevos mensajes se colocan en la cola bruta de entrada tan sólo brevemente, una hebra de ejecución del propio sistema vacía esta cola regularmente y transfiere los mensajes a una de las otras colas:

· Una cola única para todas las aplicaciones de 16 bits -igual que en windows 3.1.

· Una cola privada por hebra para aplicaciones de 32 bits.

Los mensajes generados por el propio sistema y otros procesos se transfieren directamente a las colas privadas. Si el sistema está extremadamente ocupado habrá algún tratamiento adicional sobre búferes, pero la mayor parte del tiempo éste no es necesario.

Cuando se ejecuta por primera vez un proceso de 32 bits, éste tiene una sóla cola de mensajes asociada con su hebra principal. Si el proceso genera otra hebra, el sistema no crea inmediatamente otra cola de mensajes. Cuando la segunda hebra hace su primera llamada relacionada con la cola, entonces es cuando el sistema genera otra cola de mensajes. Si una hebra no necesita una cola, el sistema no perderá recursos generándola.

3.2. LOS SOCKETS.

La Interface de sockets es una API para redes TCP/IP , se compone de funciones o rutinas

Originalmente se construyó para el sistema operativo UNIX , aunque hoy en día también la utilizan otros sistemas operativos como Microsoft Windows.

Entrada / Salida de Red :

Las llamadas al sistema de E/S en UNIX se basan en el proceso de open-read-write-close (abrir-leer-escribir-cerrar), y esto se utiliza tanto con archivos como con dispositivos hardware. En este tipo de comunicación se basó el diseño de la interface de sockets, con lo que para comunicarse con una red TCP/IP, se abre primero una conexión con la red , se leen y escriben datos a través de ella y una vez terminados los procesos se cierra la conexión.

3.2.1. QUÉ ES UN SOCKET

Un socket es una representación abstracta del extremo (endpoint) en un proceso de comunicación. Para que se dé la comunicación en una red , el programa requiere un socket en cada extremo del proceso de comunicación.

Para crear un socket , se utiliza la función socket , que devuelve un identificador de socket. Se deben especificar tres parametros :

socket = socket(protocol_familiy, socket_type,protocol)

3.2.2. TIPOS DE SOCKETS.

Define las propiedades de las comunicaciones en las que se ve envuelto un socket, esto es, el tipo de comunicación que se puede dar entre cliente y servidor. Estas pueden ser:

- Fiabilidad de transmisión.

- Mantenimiento del orden de los datos.
- No duplicación de los datos.
- El "Modo Conectado" en la comunicación.
- Envío de mensajes urgentes.

Los tipos disponibles son los siguientes:

- Tipo SOCK_DGRAM: sockets para comunicaciones en modo no conectado, con envío de datagramas de tamaño limitado (tipo telegrama). En dominios Internet como la que nos ocupa el protocolo del nivel de transporte sobre el que se basa es el UDP.
- Tipo SOCK_STREAM: para comunicaciones fiables en modo conectado, de dos vías y con tamaño variable de los mensajes de datos. Por debajo, en dominios Internet, subyace el protocolo TCP.
- Tipo SOCK_RAW: permite el acceso a protocolos de más bajo nivel como el IP (nivel de red)
- Tipo SOCK_SEQPACKET: tiene las características del SOCK_STREAM pero además el tamaño de los mensajes es fijo.

3.2.3. ESTRUCTURA DE DATOS DEL SOCKET

Cuando se llama a la función socket, la implementación del socket lo crea y devuelve un identificador de socket que identifica a un registro en la tabla de descripción. El registro muestra la estructura de datos del socket.

Estructura de Datos del Socket
Familia de Protocolos
Tipo de Servicio
Dirección IP Local

Dirección IP Remota
Puerto de Protocolo Local
Puerto de Protocolo Remoto

Tabla 3. Estructura de datos del socket

Cada vez que la aplicación llama a la función socket, la implementación de este reserva memoria para una nueva estructura de datos y almacena la dirección de la familia, el tipo de socket y el protocolo.

Entonces una conexión de red entre dos procesos se compone de cinco elementos :

1. Un puerto de protocolo local que especifica donde recibe mensajes un proceso.
2. La dirección del anfitrión o host local, la cual identifica al anfitrión que recibirá los paquetes de datos.
3. Un puerto de protocolo remoto que identifica al proceso destino.
4. La dirección del anfitrión remoto, que identifica al anfitrión destino.
5. Un protocolo, que define como los procesos transfieren la información a través de la red.

Cuando un proceso desea establecer una comunicación con otro, el proceso emisor transmite la información al socket y la API de sockets maneja la interface con la pila de protocolos TCP/IP.

3.2.4. CONFIGURACIÓN DEL SOCKET

Una vez creado el socket , utilizando la función socket, se pueden utilizar las funciones de configuración dependiendo del uso que se le vaya a dar al socket :

Si se trata de un cliente orientado a conexión, se deberá llamar a la función connect que se encargará de almacenar toda la información local y remota en la estructura de datos del socket.

Si se trata de un cliente sin conexión las funciones llamadas son:

Maquina Local: bind Maquina remota: sendto

En el caso de un servidor orientado a conexión:

Maquina Local: bind Maquina Remota: listen y accept

En el caso de un servidor sin conexión:

Maquina Local: bind Maquina Remota: recvfrom

3.2.5. CONEXIÓN DE UN SOCKET

Un programa cliente orientado a conexión utiliza la función connect para configurar un socket, y requiere como parámetros, el identificador de socket, que es el valor del descriptor del socket que devolvió la función socket. Como segundo parámetro, utiliza la dirección del socket remoto, es decir la dirección IP del host remoto y el puerto de protocolo. El tercer parámetro se refiere a la longitud de la dirección, el tamaño en bytes de la dirección del socket remoto.

result = connect(socket_handle, remote_socket_address, address_length);

En la mayoría de los casos un programa cliente orientado a conexión no especifica puerto de protocolo local, ya que puede recibir datos en cualquier puerto de protocolo. Es decir, no es necesario almacenar las direcciones IP locales, la implementación del socket lo hace automáticamente y selecciona por si misma un puerto de protocolo local.

En el caso de clientes no orientados a conexión o servidores en general tienen que atender a un puerto de protocolo las solicitudes que les pueden llegar.

La función de asignación de nombres, bind, en la API de sockets permite a un programa asociar una dirección local con un socket :

Result = bind(socket_handle, local_socket_address, address_lenght);

De esta manera se le comunica a la implementación del socket , que puerto de protocolo utilizar para la entrega de datos.

3.2.6. TRANSMISIÓN DE DATOS

Se proporcionan cinco funciones para transmitir datos a través de un socket y se dividen en dos grupos. Existen tres funciones que requieren una dirección de destino como parámetro, las restantes que son las utilizadas en los procesos orientados a conexión, no lo precisan.

FUNCION	DESCRIPCIÓN
Send	Transmite datos a través de un socket de conexión.
Write	Transmite datos a través de un socket de conexión utilizando un bufer de datos simple.
Writev	Transmite datos a través de un socket de conexión utilizando bloques de memoria no contiguos.
Sendto	Transmite datos a través de un socket sin conexión utilizando un bufer de mensajes simple.
Sendmsg	Transmite datos a través de un socket sin conexión, utilizando una estructura de mensajes flexible como bufer de mensajes

Tabla 4. Funciones para la transmisión de datos (sockets)

Las funciones del API de sockets que hacen transmisiones de datos orientadas a conexión no requieren que el programa especifique una dirección destino como parámetro. Las funciones send, write y writev sólo trabajan con sockets conectados, y tiene la siguiente estructura:

Result= write(socket_handle, message_buffer, buffer_length);

El primer parámetro es el identificador de socket, el segundo es el búfer de mensajes, que apunta al búfer de datos que contiene la información a transmitir. El tercer parámetro es el tamaño del búfer de datos.

Por otro lado la función writev no requiere que los datos ocupen bloques de memoria contiguos como si lo hace la función write. Así writev permite que se especifique una tabla de direcciones que contenga los datos.

Result= writev(socket_handle, io_vector, vector_length);

También requiere como primer parámetro , un identificador de socket , el segundo parámetro especifica la dirección de una tabla que contiene una secuencia de apuntadores. Cuando la función writev transmita los datos, enviará la información contenida en cada localidad de memoria especificada por el array de apuntadores, transmitiéndolos en el mismo orden en que aparecen en el array.

La función send es otra función del interface de sockets para utilizar con sockets orientados a conexión:

Result = send(socket(socket_handle, message_buffer, buffer_length, special_flags));

Con send se pueden especificar banderas opcionales para controlar la transmisión, como la gestión de datos urgentes (fuera de banda).

Las tres funciones write, writev y send , devuelven un valor entero, que es el numero de bytes transmitidos por el socket. En caso de que exista un error devuelven : -1 .

3.2.7. SOCKETS SIN CONEXIÓN

Para enviar datos a través de un socket sin conexión se tienen las funciones , sendto y sendmsg :

Result = sendto(socket_handle, message_buffer, buffer_length, special_flags, socket_address_structure, address_structure_length);

Requiere 6 parámetros, entre los nuevos que aparecen ahora la dirección de destino (socket_address_structure, y la longitud de la misma en bytes (address_structure_length).

La función sendmsg nos permite utilizar para la transmisión una estructura de mensaje, en lugar de un simple buffer de datos. Requiere como parámetros un identificador de socket, un puntero a la estructura del mensaje (message_structure) y banderas (special_flags).

Result= sendmsg(socket_handle, message_structure, special_flags);

3.2.8. RECEPCIÓN DE DATOS POR UN SOCKET

La Interface de Sockets , incluye 5 funciones para la recepción de datos, que se corresponden con las 5 funciones anteriores vistas en la transmisión de datos :

Send	Recv
Write	Read
Writev	Readv
Sendto	Recvfrom
Sendmsg	Recvmsg

Tabla 5. Funciones para la recepción de datos (Sockets)

A parte de estas correspondencias, hay que tener en cuenta que no es indispensable utilizarlas unas con otras , ya que una vez se han enviado datos , para recibirlos se puede realizar con cualquiera de las funciones correspondientes al tipo de servicio , orientado o no a conexión.

3.2.8.1. Descripción Del Proceso

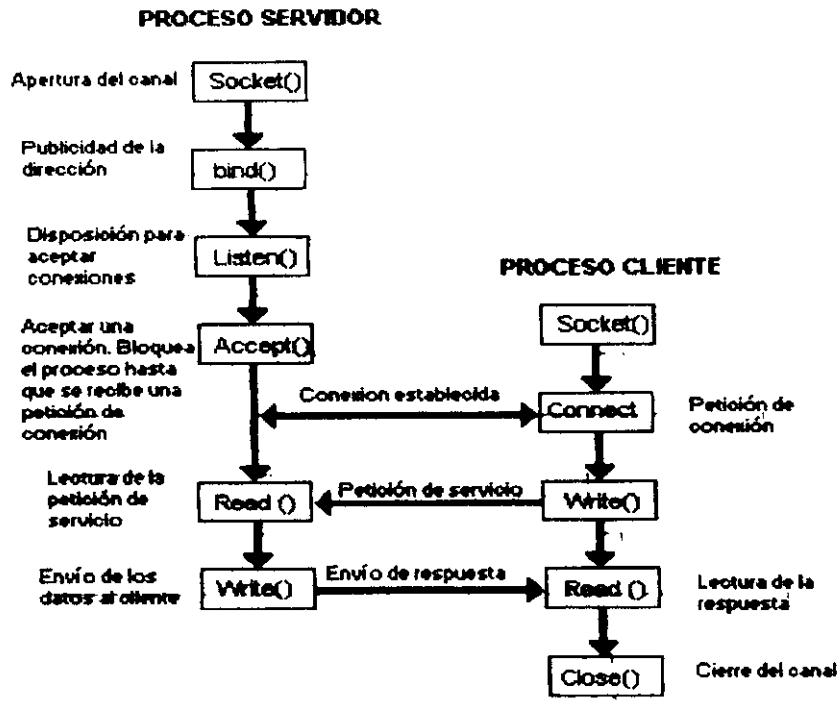


Fig. 13 . Descripción del proceso

La parte izquierda muestra las llamadas a funciones del servidor , mientras que la derecha muestra las llamadas a funciones del cliente.

El programa servidor solicita a la implementación del socket que le asigne una estructura de datos para el socket y que le devuelva un descriptor de sockets para utilizarlo en las siguientes llamadas a funciones de la interfaz de sockets.

Después el servidor une el socket a un puerto de protocolo local. La función listen indica al socket que atienda las conexiones entrantes y que confirme las solicitudes de conexión, se encarga de poner al socket en modo de atención pasiva.

Result = listen(socket_handle, queue_length);

Donde el segundo parámetro (queue_length) , nos permite especificar el número máximo de solicitudes que pueden acumularse en la cola.

Después de configurar una cola de datos entrantes, el programa servidor llamará a la función accept. , cesa su actividad y espera una solicitud de conexión de un programa cliente.

Result = accept (socket_handle, socket_address, address_length);

El programa cliente también crea un socket , pero no necesita ocuparse de que dirección local usará el protocolo ya que utiliza un protocolo orientado a conexión, por lo tanto no llama a la función bind. Lo que hace es iniciar la conversación en red llamando a la función connect.

Después de que el cliente y el servidor establecen la conexión , pueden ocurrir comunicaciones adicionales a través de las funciones write y read.

3.2.9. LA API WINDOWS SOCKETS

El software llamado Windows Sockets o Winsock es una API (Application Program Interface) , Interface de programas de aplicación para redes TCP/IP, y específica de la familia de sistemas operativos de Microsoft Windows en todas sus versiones.

Windows Sockets implementa la interface de sockets como una biblioteca de enlace dinámico, una DLL (Dynamic Lynk Library), que no es más que un módulo ejecutable que el sistema puede cargar en cualquier momento.

La API Winsock proporciona una biblioteca de funciones que se divide en tres grupos :

Funciones de los Berkeley Sockets.

Funciones de Bases de Datos que permiten obtener información relacionada con el DNS, servicios de comunicaciones y protocolos.

Las extensiones específicas de Windows a las rutinas de los Sockets de Berkeley.

Distinguimos entre las funciones de bloqueo, que son aquellas que evitan que se llame a cualquier otra función hasta que esta termine sus propias operaciones de red. Y las de no bloqueo que terminan de inmediato o emiten un mensaje de error.

3.2.10. FUNCIONES SOCKET

Función	Descripción
Accept	Confirma una conexión de entrada. Crea un socket nuevo y lo conecta al host remoto que pidió la conexión. Devuelve el socket original a su estado de atender.
Closesocket	Cierra un extremo de una conexión por sockets.
Connect	Inicia una conexión en el socket especificado.
Recv	Recibe información de un socket conectado.
Recvfrom	Recibe información de un socket conectado o de uno no conectado.
Select	Ejecuta multiplexaje sincrono de E/S al monitorear el estado de múltiples sockets.
Send	Envía información a un socket conectado.
Sendto	Envía información a un socket conectado o a uno no conectado.

Tabla 6. Funciones Socket (Bloqueo)

Las funciones vistas en la tabla anterior, se pueden clasificar como de bloqueo porque requieren comunicación con el anfitrión remoto. Mientras que las que aparecen en la tabla de abajo sólo utilizan información almacenada de manera local, o bien sólo trabajan con el extremo de una conexión del socket de su computadora.

Función	Descripción
Bind	Asigna un nombre local a un socket sin nombre.
Getpeername	Obtiene el nombre del par conectado al socket especificado.
Getsockname	Obtiene el nombre local para el socket especificado.
Getsockopt	Obtiene opciones asociadas con el socket especificado.
Htonl	Convierte un número de 32 bits de ordenamiento de byte de anfitrión a ordenamiento de byte de red.
Htons	Convierte un número de 16 bits de ordenamiento de byte de anfitrión a ordenamiento de byte de red.
Inet_addr	Convierte una cadena de caracteres que representa una dirección IP en notación decimal al valor binario de 32 bits.
Inet_ntoa	Convierte una dirección IP a notación decimal.
Ioctlsocket	Controla varios parámetros relacionados con la forma de operar del socket y el manejo de la E/S de red.
Listen	Indica a un socket específico que atienda las conexiones entrantes.
Ntohl	Convierte un número de 32 bits de ordenamiento de byte de red a ordenamiento de byte de anfitrión.
Ntohs	Convierte un número de 16 bits de ordenamiento de byte de red a ordenamiento de byte de anfitrión.
Setsockopt	Almacena opciones asociadas con el socket especificado.

Shutdown	Cierra parte de una conexión full duplex.
Socket	Crea un extremo para la comunicación y devuelve un identificador de socket.

Tabla 7. Funciones Socket (Almacenamiento)

3.2.11. FUNCIONES DE BASE DE DATOS

Las funciones de base de datos de Winsock , permiten obtener información sobre nombres de dominio, servicios de comunicación y protocolos .

Función	Descripción
Gethostbyaddr	Obtiene nombre de dominio y dirección IP correspondiente a una dirección de red.
Gethostbyname	Obtiene nombre de dominio y dirección IP correspondiente a un nombre de anfitrión.
Gethostname	Obtiene el nombre de dominio del anfitrión local.
Getprotobyname	Obtiene un protocolo por nombre y devuelve el nombre oficial y el número definido para representar al protocolo.
Getprotobynumber	Obtiene el nombre y número de protocolo representado por un número específico.
Getservbyname	Obtiene un nombre y el puerto protocolo correspondiente al nombre del servicio.
Getservbyport	Obtiene el nombre y puerto correspondiente a un puerto de protocolo especifico.

Tabla 8. Funciones Socket (Bases de Datos)

Algunas de las funciones de bases de datos de Winsock devuelven estructuras de datos volátiles, ya que sólo guardan los resultados hasta la próxima llamada a otra función Winsock , por lo tanto estos resultados debe guardarse en un lugar diferente de la memoria. Para ello tenemos las versiones asíncronas de estas funciones que nos permiten aprovechar el despliegue de mensajes dentro de Windows.

WSAAsyncSelect	Ofrece una versión asincrona de la función Select.
WSACancelAsyncRequest	Cancela una instancia de la función WSAAsyncGetXByY
WSACancelBlockingCall	Cancela una llamada de bloqueo de la API
WSACleanup	Termina sesión desde los DLL subyacentes de Winsock
WSAGetLastError	Obtiene detalles sobre el último error de la API Winsock.
WSAIsBlocking	Determina si el DLL de Winsock subyacente está bloqueado.
WSASetLastError	Establece el regreso de error para WSAGetLastError subsecuente.
WSAStartup	Inicia el DLL de Winsock subyacente.
WSAUnhookBlockingHook	Restaura la función original de bloqueo.

Tabla 9. Funciones Sockets (Winsock)

3.2.12. PROGRAMACIÓN CON WINDOWS SOCKETS

Los programas de red basados en la API Windows Sockets deben incluir un encabezado de archivo llamado winsock.h :

#include<winsock.h>

La API de Windows Sockets requiere dos funciones especificas de windows, que son:

.WSAStartup : Se debe llamar a esta función antes de llamar a cualquier otra de Winsock. Nos permite especificar que versión de la API Windows Sockets va a necesitar nuestro programa. Se establece una negociación entre la aplicación y Winsock.dll .

.WSACleanup : Se debe colocar por cada llamada que se haga a WSAStartup. Cuando se llama a la función WSACleanup por última vez, winsock desconecta cualquiera de los sockets de flujo de bytes existentes. Aunque respeta la información pendiente.

El descriptor de Socket : Winsock define SOCKET como un tipo de datos sin signo y emplea la constante INVALID_SOCKET para identificar a los no válidos.

Un socket válido está en el rango desde 0.....INVALID SOCKET-1 .

Para el caso de que ocurra un error de socket , winsock nos proporciona la constante SOCKET_ERROR , que define como –1, y para identificar la condición concreta del error nuestra aplicación deberá llamar a la función WSAGetLastError, que se encarga de obtener el último error que ocurrió en la red.

La función Select : Esta función deja que un solo proceso monitoree o determine el estado de múltiples sockets. Un conjunto es una lista específica de sockets que Winsock monitorea para revisar los cambios de estado. Para manipular estos conjuntos Winsock se vale de las siguientes macros :

MACRO	FUNCION
FD_CLR	Borra un identificador de socket de la lista.
FD_ISSET	Devuelve un valor diferente de cero (TRUE) si el identificador de socket está establecido y cero (FALSE) si no lo está.
FD_SET	Agrega un identificador de socket a una lista.
FD_ZERO	Inicializa un conjunto de identificadores.

Tabla 10. Funciones Socket (Función Select)

Winsock proporciona una versión asíncrona de SELECT , WSAAsyncSelect para las operaciones de socket de no bloqueo en Winsock.

Nuestra aplicación puede llamar a la función WSAAsyncSelect para cambiar la operación del socket de bloqueo a no bloqueo. Teniendo en cuenta que WSAAsyncSelect sólo acepta un identificador de socket a la vez como parámetro.

WSAAsyncSelect (SOCKET s, HWND hWnd, unsigned int wMsg, long lEvent);

El parámetro lEvent es una máscara de bits que especifica que combinación de eventos de la red se desean revisar.

Sus valores pueden ser :

Constante	Significado
FD_READ	Solicita notificación de que está listo para leer.
FD_WRITE	Solicita notificación de que está listo para escribir.
FD_OOB	Solicita un aviso de la llegada de información fuera de banda.
FD_ACCEPT	Solicita un aviso de las conexiones entrantes.
FD_CONNECT	Solicita un aviso de las conexiones terminadas.
FD_CLOSE	Solicita un aviso de la terminación de socket.

Tabla 11. Funciones Socket (Función Event)

El segundo parámetro hWnd especifica el identificador de la ventana que recibirá el mensaje. (parámetro wMsg)

El tercer parámetro wMsg , define el mensaje que usted quiere que envíe Windows cuando el evento especificado ocurra en la red.

Cuando se llama a la función WSAAsyncSelect, Winsock designa al socket que se especifica como de no bloqueo. Por ejemplo si se quiere realizar una operación usando la función recv , evitando operaciones de bloqueo. Se llamará a la función WSAAsyncSelect, que se encargará a su vez de decirle a Windows que notifique a la aplicación en cuestión cuando el socket está listo para leer. Cuando el socket recibe la información, Windows enviará el identificador de mensaje correspondiente a la ventana que especifique la función WSAAsyncSelect . Cuando el procedimiento de manejo del mensaje para esa ventana reciba el mensaje, la aplicación podrá ejecutar un código que llame a la función recv.

3.2.12.1. Hook De Bloqueo.

Winsock nunca permite que una operación de bloqueo ocurra dentro de Windows. Cuando un programa llama a una función que causaría una operación de este tipo, Winsock entra en un ciclo y llama de forma repetitiva a un identificador de bloqueo o rutina hook, cuyo propósito es interceptar las llamadas a funciones que causan una operación de bloqueo.

El identificador de bloqueo estándar de Winsock incluye un código análogo a las siguientes instrucciones:

```
If (peekMessage(&msg, NULL, 0, 0, PM_REMOVE))
{TranslateMessage(&msg);
DispatchMessage(&msg);}
```

La función PeekMessage revisa la cola de mensajes de la aplicación, en caso de que existan mensajes, lo coloca en una estructura MSG y devuelve un valor que no sea cero.

De todas formas el hook de bloqueo no resuelve el problema en su totalidad, ya que cuando se ejecuta es posible que aparezca un mensaje de Windows para la tarea en curso, que puede causar que la aplicación llame a otra función de Winsock. Con lo cual se tiene

que siempre que una operación de bloqueo de Winsock esté en progreso, no se podrá llamar desde la aplicación a otras funciones de Winsock.

Winsock proporciona dos funciones para detectar y manejar operaciones de bloqueo:

Ninguna de las funciones requiere parámetros.

1. WSAIsBlocking : Se llama a esta función para determinar si está en progreso una operación de bloqueo. Devolverá TRUE si es así y FALSE, en caso contrario.

2. WSACancelBlockingCall : Se llamará a esta función para cancelar la operación de bloqueo que esté en curso. La llamada a la función que inició la operación de bloqueo coge el valor de error WSAEINTR.

La API de Winsock incluye dos funciones avanzadas para manejar las operaciones de bloqueo:

. WSASetBlockingHook : Nos permite definir nuestra propia rutina de bloqueo.

. WSAUnhookBlockinhook : Se encarga de restaurar el hook de bloqueo predeterminado.

Para los sistemas operativos Windows, el bloqueo ocurre en base a cada tarea. Winsock no incluye un hook de bloqueo para las versiones multitarea de Windows, cuando sucede una operación de bloqueo todas las demás actividades en la tarea se detienen hasta que esta concluye. De todas formas Winsock permite utilizar en tales versiones de Windows la función WSASetBlockingHook para implementar nuestro propio hook de bloqueo.

3.3. FILOSOFÍA CLIENTE-SERVIDOR:

3.3.1. EL SERVIDOR.

Vamos a explicar el proceso de comunicación servidor-cliente en modo conectado, modo utilizado por las aplicaciones estándar de Internet (telnet, ftp). El servidor es el proceso que crea el socket no nombrado y acepta las conexiones a él. El orden de las llamadas al sistema para la realización de esta función es:

1°) int socket (int dominio, int tipo, int protocolo)

crea un socket sin nombre de un dominio, tipo y protocolo específico

dominio : AF_INET, AF_UNIX

tipo : SOCK__DGRAM, SOCK__STREAM

protocolo : 0 (protocolo por defecto)

2°) int bind (int dfServer, struct sockaddr* direccServer, int longDirecc)

nombra un socket: asocia el socket no nombrado de descriptor dfServer con la dirección del socket almacenado en direccServer. La dirección depende de si estamos en un dominio AF_UNIX o AF_INET.

3°) int listen (int dfServer, int longCola)

especifica el máximo número de peticiones de conexión pendientes.

4°) int accept (int dfServer, struct sockaddr* direccCliente, int* longDireccCli)

escucha al socket nombrado servidor dfServer y espera hasta que se reciba la petición de la conexión de un cliente. Al ocurrir esta incidencia, crea un socket sin nombre con las mismas características que el socket servidor original, lo conecta al socket cliente y devuelve

un descriptor de fichero que puede ser utilizado para la comunicación con el cliente.

3.3.2. EL CLIENTE.

Es el proceso encargado de crear un socket sin nombre y posteriormente enlazarlo con el socker servidor nombrado. O sea, es el proceso que demanda una conexión al servidor. La secuencia de llamadas al sistema es:

1°) int socket (int dominio, int tipo, int protocolo)

crea un socket sin nombre de un dominio, tipo y protocolo específico dominio : AF_INET, AF_UNIX

tipo : SOCK__DGRAM, SOCK__STREAM

protocolo : 0 (protocolo por defecto)

2°) int connect (int dfCliente, struct sockaddr* direccServer, int longDirecc)

intenta conectar con un socket servidor cuya dirección se encuentra incluida en la estructura apuntada por direccServer. El descriptor dfCliente se utilizará para comunicar con el socket servidor. El tipo de estructura dependerá del dominio en que nos encontremos.

Una vez establecida la comunicación, los descriptores de ficheros serán utilizados para almacenar la información a leer o escribir.

SERVIDOR .	**CLIENTE**
descrServer = socket (dominio, SOCK_STREAM,PROTOCOLO)	descrClient = socket (dominio, SOCK_STREAM,PROTOCOLO)
Bind (descrServer, PuntSockServer,longServer)	
	do {
Listen (descrServer, longCola)	
DescrClient = accept (descrServer,PuntSockClient,longClient)	result = connect (descrClient, PuntSockServer,longserver)
[close (descrServer)]	} while (result == -1)
< DIALOGO >	< DIALOGO >
Close (descrClient)	close (descrClient)

3.3.3. FUNCIONAMIENTO DEL SERVIDOR

El proceso intérprete de órdenes deberá esperar a que algún proceso cliente le solicite sus servicios. Mientras esto no ocurra permanecerá bloqueado. Cuando se le demande un servicio, saldrá del estado de bloqueo y actuará. Básicamente el esquema de funcionamiento de este proceso es el siguiente:

Generar el proceso demonio.

Establecer mecanismos de comunicación.

```
for (;;)
{
        recibe_petición (&mi_peticion);
        if ( orden de finalizar ) break;
        Procesar línea de órdenes.
        Ejecutar orden.
        Esperar por la finalización de la orden.
        envia_respuesta (codigo_de_respuesta);
}
```

Eliminar los mecanismos de comunicación.

Terminar

Observe cómo se trata de un proceso que entra en un bucle infinito, del cual se sale cuando algún proceso cliente le ordene al servidor su finalización. Esta es una estructura típica para un proceso servidor, y aparece con frecuencia en los lenguajes de paso de mensajes o los sistemas dirigidos por eventos (por ejemplo, al programar en un entorno de ventanas). Consiste en un bucle sin fin en el que cada iteración consiste en recibir un mensaje y atenderlo, así de simple.

Seguidamente se detallan los aspectos más importantes de cada punto de la ejecución.

3.3.3.1. Generación de un proceso demonio

Consiste en la creación de un nuevo proceso que se deberá ejecutar en segundo plano (proceso demonio). Una vez creado el nuevo proceso, el proceso deberá finalizar. De esta forma, el servidor no acapara la terminal donde se ha lanzado.

3.3.3.2. Establecimiento de los mecanismos de comunicación

Consiste en la creación de los recursos que permitirán la comunicación entre el servidor y los clientes. Para esta práctica se podrán utilizar dos clases de herramientas de

intercomunicación: colas de mensajes, o bien usar conjuntamente semáforos y memoria compartida.

3.3.3.3. Espera por una nueva petición

El servidor habrá de bloquearse mientras no tenga ninguna solicitud de servicio. Así pues, se tendrá que diseñar un mecanismo de sincronización y comunicación que garantice que mientras no haya solicitud, el servidor permanezca en estado de bloqueo (sin consumo de CPU). Además, este mecanismo deberá garantizar la integridad de las peticiones; esto es, una petición que se está recogiendo no podrá ser destruida por la llegada de una nueva.

3.3.3.4. Comprobación del servicio solicitado

En este paso se comprueba si el servicio requerido se corresponde con una orden de finalización del servidor; o, por contra, se solicita la ejecución de un programa. Como se ha dicho, esto viene expresado por el valor del campo orden[0] de la petición. Si vale FINALIZA, el servidor sale del bucle, elimina los recursos IPC que haya creado y termina su ejecución.

3.3.3.5. Procesamiento de la orden

En este paso se produce el análisis de la cadena de caracteres que contiene la orden a ejecutar. Este tratamiento tiene la finalidad de obtener: a) el nombre del fichero ejecutable y b) cada uno de los argumentos del programa. Las funciones de rastreo de cadenas, como strchr, pueden ser útiles para trocear la línea de órdenes.

Cada uno de los datos extraídos se empleará para invocar posteriormente al programa solicitado, haciendo uso de alguna función exec... del UNIX.

3.3.3.6. Ejecución de la orden

El programa servidor creará un nuevo proceso hijo mediante la función fork. El proceso padre deberá esperar a que el hijo recién lanzado finalice su ejecución; la espera se hará mediante la función wait.

Por su parte, el proceso hijo deberá cargar y ejecutar el programa solicitado con los argumentos especificados en la línea de órdenes mediante el uso de alguna de las funciones exec.

La salida estándar de la orden habrá de redirigirse al archivo especificado en el campo canal de la petición recibida. Más adelante se explica cómo se realiza esto.

Algunas de las situaciones de error que pueden surgir, y que deberán ser notificadas al cliente en la respuesta, son:

1. No se pudo crear el proceso hijo con el fork (devolver ERR_HIJO)

2. No se pudo lanzar el programa con exec (devolver ERR_CARGA)

3. La orden dio un resultado erróneo (devolver ERR_EJEC)

Si no ocurre nada anómalo, se le entrega al cliente el valor ERR_OK

3.3.3.7. Espera por la finalización del servicio solicitado

Tal y como se ha indicado, esto lo realizará el servidor mediante el uso de la llamada al sistema wait. Lo único a reseñar es que no se deberá llegar a este punto si se produce un error en la invocación a la función fork.

Si el error se produce en la carga o en la ejecución del programa solicitado, entonces, se podrá detectar consultando el argumento utilizado en la llamada a la función wait.

3.3.3.8. Notificación del resultado al cliente

El servidor deberá comunicar al cliente que le solicitó el servicio si la orden demandada se ha realizado o no. Para ello se utilizará la función envia_respuesta(), que ustedes habrán de implementar.

3.3.4. ESTRUCTURA DE UN PROCESO CLIENTE

Los pasos básicos de ejecución de un cliente son:

Construir petición en mi_peticion

envia_peticion(&mi_peticion);

recibe_respuesta(&resultado);

Comprobar resultado de la ejecución.

Finalizar.

Seguidamente se describe lo que hace cada uno de estos pasos:

3.3.4.1. Construcción de la petición para el servidor

Las primeras instrucciones del servidor consisten en elaborar la petición que se enviará al servidor. Se tendrá que leer los parámetros argc y argv[] de la función main en el programa cliente, y convertirlos en un struct peticion.

Por otra parte, el cliente deberá rellenar el campo canal de la estructura petición, con la ruta de la terminal o el archivo donde se desea visualizar el resultado. Para conocer la terminal actual, se invoca a la función funcion().

3.3.4.2. Solicitud de servicio y espera por la notificación de servicio realizado

Una vez que el cliente ha construido la petición, deberá solicitar el servicio al proceso servidor. Deberá hacer uso del mecanismo de sincronización y comunicación diseñado por ustedes, basado en algún IPC de UNIX. La implementación deberá garantizar que el cliente se mantenga bloqueado hasta que el servidor le notifique que se ha completado su petición, o bien hasta que el servidor finalice.

Observe que el servidor puede encontrarse ocupado con peticiones de otros clientes, y que no se deben interferir unas peticiones con otras.

3.3.4.3. Comprobación del resultado del servicio realizado.

Una vez que se le ha notificado que el servicio ha sido realizado, el cliente deberá consultar el campo resultado de la estructura struct Respuesta, para determinar si la solicitud se tramitó correctamente o no.

CAPÍTULO IV

4. PROTOCOLOS DE COMUNICACIÓN.

4.1. MODELO OSI.

Las redes de computadoras surgieron para hacer viable el comportamiento eficiente de recursos computacionales entre usuarios cuando pertenecen a sistemas heterogéneos en cuanto a aplicaciones y fabricantes se refiere, lo cual dificulta su interconexión.

Los grandes fabricantes desarrollaron soluciones para la interconexión de sus propios equipos mediante el uso de una Arquitectura de Red propia, la cual estaba constituida por un conjunto de convenciones para la interconexión de sus equipos.

En 1977 para solucionar este problema, la Organización Internacional de Estandarización (ISO) vió la necesidad de normas para la interconexión de sistemas diferentes y creó el subcomité SC16 para estudiar el problema. Este desarrolló el modelo de Arquitectura llamado "Modelo de Referencia para la Intercomunicación de Sistemas Abiertos" (OSI), el cual fue aprobado por la ISO en 1983 a través del documento ISO7494. Este modelo es estandarizado y se estructura en siete niveles, de los cuales los tres inferiores constituyen un estándar muy difundido que se conoce como X.25.

En el concepto de OSI, un sistema es un conjunto de una o más computadoras; el software asociado, los periféricos, las terminales, los procesos físicos, los medios de transferencia de información, etc., forman un ente autónomo con capacidad de realizar el procesamiento de la información

OSI pone atención al intercambio de información entre sistemas y no al funcionamiento interno de cada sistema en particular o sea, el modelo de referencia OSI constituye el marco de trabajo para el desarrollo de protocolos estándares para la comunicación entre dos niveles homónimos ubicados en equipos separados. El objetivo a largo plazo de OSI es desarrollar una compatibilidad total inter-sistemas, entre los muchos transportadores alrededor del mundo.

4.1.1. CAPAS.

El modelo OSI tiene siete capas. Los principios aplicados para el establecimiento de siete capas fueron los siguientes.

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.

El modelo OSI, por sí mismo, no es una arquitectura de red, dado que no especifica, en forma exacta, los servicios y protocolos que se utiliza, dado que no especifica, en forma exacta, los servicios y protocolos que se utilizarán en cada una de las capas. Sólo indica lo que cada capa deberá hacer. Sin embargo, la ISO también ha generado normas para todas las capas, aunque éstas, estrictamente hablando, no forman parte del modelo.

4.1.1.1. Capa Física

Comprende el conjunto de recursos físicos de reglas lógicas, que permiten la transmisión de bits entre nodos de comunicación que conforman una Red de Computadoras. Este provee las caracteristicas mecánicas, eléctricas, funcionales y procedimientos necesarios para establecer, mantener y liberar conexiones físicas entre el dispositivo terminal (DTE) y el punto de conexión de la RED (DCE), o entre dos (DTE's).

En general el proceso de comunicación es bidireccional o sea, nodos interconectados transmiten y reciben bits simultáneamente o alternadamente. Cuando la transmisión es realizada en forma simultánea a los dos sentidos del sistema, se conoce como Full Duplex, en caso de la transmisión típica de bits en redes locales de computadoras, es del tipo Semiduplex.

4.1.1.2. Capa de Enlace.

La finalidad de este nivel es implementar mecanismo de detección y recuperación de errores, ofreciendo de esta forma un servicio más confiable a los niveles superiores. Otras funciones incluyen el establecimiento de la conexión y procedimientos que permitan el uso eficiente del medio de transmisión. Este nivel provee la conexión lógica a través de la línea, el direccionamiento, el secuenciamiento y la recuperación de errores.

Existe una dirección de enlace en el nivel DLC (Data Link Control). En este nivel se determina el uso de una disciplina de comunicaciones conocida como HDLC (High Level Data Link Control). Este es el protocolo de línea conocido como un Estándar Universal, al cual muchos toman como modelo, siendo los datos organizados en tramas.

Al juntar las funciones de los niveles 1 y 2, se tiene la forma de conectar físicamente dos nodos adyacentes y transferir un mensaje entre ellos, manejando direccionamiento, control de errores, etc.

4.1.1.3. Capa de Red

Este nivel suministra los medios para establecer, mantener y liberar las conexiones de Red, añade a los anteriores los medios para transportar información por medio de red. Las facilidades del nivel de Red están orientados al control de los nodos de conmutación de la Red y proporciona un cambio lógico entre dos extremos de la Red, bien como circuito virtual para toda la comunicación, o bien como unidades independientes o Datagrama.

Las funciones proporcionadas por este nivel incluye el ruteo de los mensajes, las notificaciones de errores y opcionalmente la segmentación y el bloqueo. La utilidad de este nivel puede ser vista como de "Dirección de Control entre los puntos de conmutación", más que como proveedora de ayuda para la transferencia de datos entre éstos puntos. En este nivel se determina el formato del campo de información de la trama HDLC. A esto se le llama "Paquete" y es un término que se ha vuelto muy popular, a raíz de la difusión del uso de redes X.25 o de Conmutación de Paquetes (Packets Switching). Estos tres primeros niveles recomiendan procedimientos para solucionar los requerimientos de conexión entre DTE y un DCE, para efectos de realizar la transmisión de mensaje con propósito y con un buen grado de confiabilidad.

4.1.1.4. Capa de Transporte

Este nivel debe asegurar la fiabilidad de la conexión y conseguir la transferencia de datos desde su origen a su destino, además de proporcionar el control entre nodos de usuarios a través de la Red.

Los niveles de 1 al 4 de OSI forman el subsistema de transporte. El nivel 4 revela a las secciones de cualquier consideración de detalle referente a la forma en la cual se realiza la transferencia de los datos.

Una conexión de transporte se identifica por un "Identificador de Punto Final de Transporte" y una o más conexiones de transporte pueden ubicarse dentro de la misma conexión de Red.

4.1.1.5. Capa de Sesión.

Proporciona la función necesaria para mantener un diálogo entre los procesos del nivel de aplicación (nivel 7) incluyendo las funciones necesarias para el establecimiento y terminación de la sesión. Además, provee el soporte de interacciones entre entidades que cooperan en el nivel de presentación.

Las funciones del nivel de sesión se pueden dividir categorías: Determinación y cancelación de contrato entre dos entidades, comprendiendo sincronización, delimitación y recuperación de operaciones con los datos (esto se llama servicio de dialogo de sesión). Una sesión se identifica por "Identificadores de Destino Final". Se han definido tres tipos de interacciones:

1- Dos vías simultáneas

2- Dos vías alternadas

3- Dos vías

4.1.1.6. Capa de Presentación.

Este nivel suministra las transformaciones requeridas de la información intercambiada entre los procesos del nivel de aplicación, incluyendo las funciones necesarias para el establecimiento y terminación de una sesión. Este nivel proporciona un conjunto de servicios de conversión y descifrado, que el nivel de aplicación puede seleccionar para poder interpretar el significado de los datos intercambiados.

El modelo identifica tres ejemplos de protocolo en este nivel:

- Protocolo de Terminal Virtual

- Protocolo de Archivo Virtual y

- Protocolo de transferencia de trabajo y manipulación

4.1.1.7. Capa de Aplicación.

Este nivel se refiere a la aplicación específica de los usuarios de la red de transporte y realiza las actividades del sistema o la aplicación necesaria para suministrar o soportar las funciones específicas de procesar la información. Todos los otros niveles existen en función de brindar soporte a éste. Una aplicación se compone de procesos cooperantes que se intercomunican mediante el uso de los protocolos definidos en este nivel.

4.2. PROTOCOLO TCP/IP

4.2.1. DEFINICIÓN TCP/IP.

Se han desarrollado diferentes familias de protocolos para comunicación por red de datos para los sistemas UNIX. El más ampliamente utilizado es el Internet Protocol Suite, comúnmente conocido como TCP / IP.

Es un protocolo DARPA que proporciona transmisión fiable de paquetes de datos sobre redes. El nombre TCP / IP Proviene de dos protocolos importantes de la familia, el Transmission Contorl Protocol (TCP) y el Internet Protocol (IP). Todos juntos llegan a ser más de 100 protocolos diferentes definidos en este conjunto.

El TCP / IP es la base del Internet que sirve para enlazar computadoras que utilizan diferentes sistemas operativos, incluyendo PC, minicomputadoras y computadoras centrales sobre redes de área local y área extensa. TCP / IP fue desarrollado y demostrado por primera vez en 1972 por el departamento de defensa de los Estados Unidos, ejecutándolo en el ARPANET una red de área extensa del departamento de defensa.

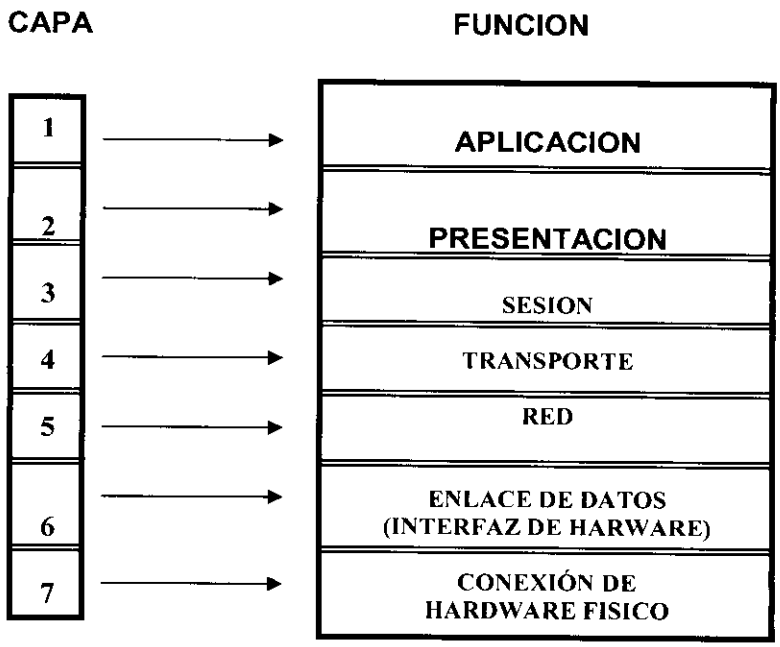
4.2.2. FUNCIONALIDAD DE LAS CAPAS

Una vez que se toma la decisión de subdividir los problemas de comunicación en cuatro subproblemas y organizar el software de protocolo en módulos, de manera que cada uno maneja un problema, surge la pregunta. "¿Qué tipo de funciones debe instalar en cada

modulo?". La pregunta no es fácil de responder por varias razones. En primer lugar, un grupo de objetivos y condiciones determinan un problema de comunicación en particular, es posible elegir una organización que optimice un software de protocolos para ese problema. Segundo, incluso cuando se consideran los servicios generales al nivel de red, como un transporte confiable es posible seleccionar entre distintas maneras de resolver el problema. Tercero, el diseño de una arquitectura de red y la organización del software de protocolo esta interrelacionado; no se puede diseñar a uno sin considera al otro.

4.2.3. MODELO DE REFERENCIA ISO DE 7 CAPAS

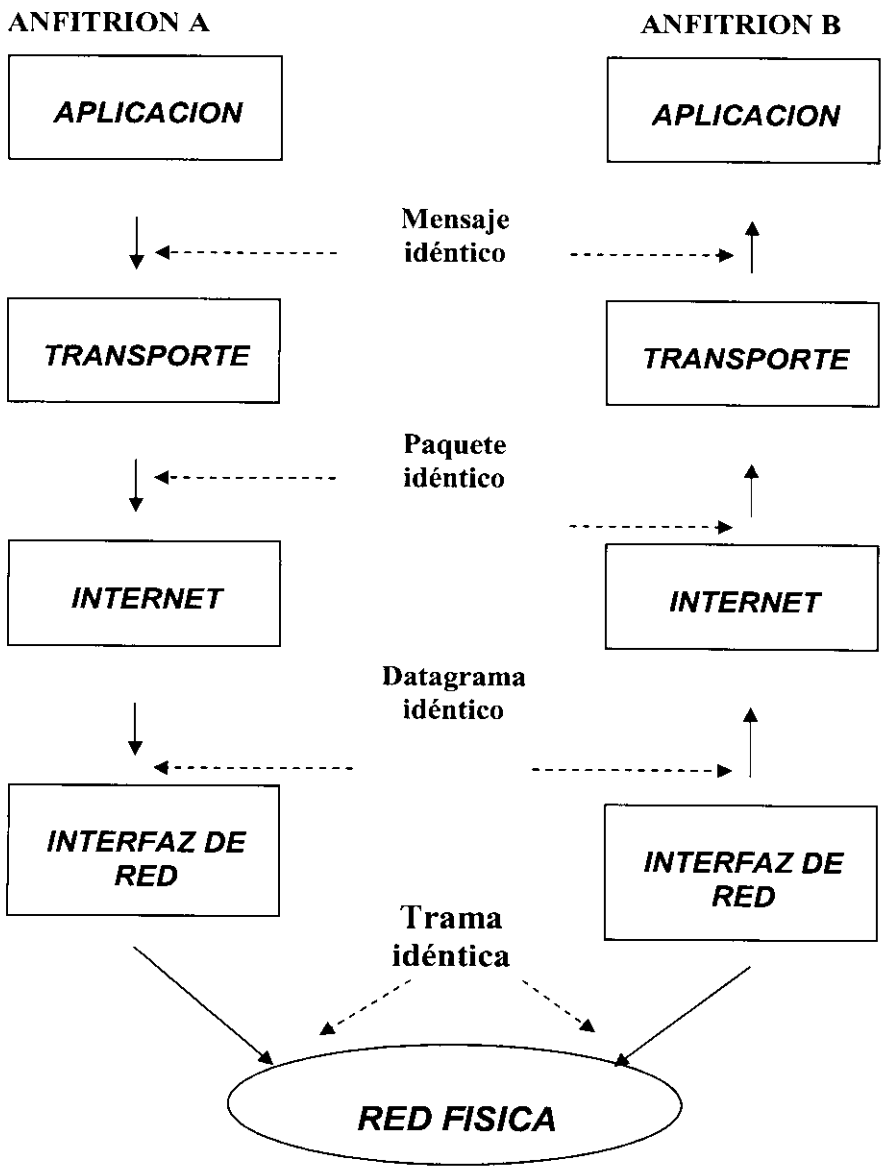
Existen dos modelos dominantes sobre la estratificación por capas de protocolo. La primera, basada en el trabajo realizado por la International Organization for Standarization (Organización para la Estandarización o ISO, por sus siglas en inglés), conocida como Referencia Model of Open System Interconnection Modelo de referencia de interconexión de sistemas abiertos) de ISO, denominada frecuentemente modelo ISO. El modelo ISO contiene 7 capas conceptuales organizadas como se muestra a continuación:



El modelo ISO, elaborado para describir protocolos para una sola red, no contiene un nivel especifico para el ruteo en el enlace de redes, como sucede con el protocolo TCP/IP.

4.2.4. EL PRINCIPIO DE LA ESTRATIFICACIÓN POR CAPAS DE PROTOCOLOS.

Independientemente del esquema de estratificación por capas que se utilice o de las funciones de las capas, la operación de los protocolos estratificados por capas se basa en una idea fundamental. La idea, conocida como principio de estratificación por capas puede resumirse de la siguiente forma:



Los protocolos estratificados por capas están diseñados de modo que una capa n en el receptor de destino reciba exactamente el mismo objeto enviado por la correspondiente capa n de la fuente.

El principio de estratificación por capas explica por que la estratificación por capas es una idea poderosa. Esta permite que el diseñador de protocolos enfoque su atención hacia una capa a la vez, sin preocuparse acerca del desempeño de las capas inferiores. Por ejemplo, cuando se construye una aplicación para transferencia de archivos, el diseñador piensa solo en dos copias del programa de aplicación que se correrá en dos máquinas y se concentrará en los mensajes que se necesitan intercambiar para la transferencia de archivos. El diseñador asume que la aplicación en el anfitrión receptor es exactamente la misma que en el anfitrión emisor.

4.2.4.1. Estratificación Por Capas En Presencia De Una Subestructura De Red.

Cuando un ruteador recibe un datagrama, este puede entregar el datagrama en su destino o en la red local, o transferir el datagrama a través de una línea serial hacia otro ruteador. La cuestión es la siguiente: "¿cómo se ajusta el protocolo utilizado en una línea serial con respecto al esquema de estratificación por capas del TCP/IP?" La respuesta depende de como considera el diseñador la interconexión con la línea serial.

Desde la perspectiva del IP, el conjunto de conexiones punto a punto entre ruteadores puede funcionar como un conjunto de redes físicas independientes o funcionar colectivamente como una sola red física. En el primer caso, cada enlace físico es tratado exactamente como cualquier otra red en una red de redes. A esta se le asigna un numero único de red (por lo general de clase C) y los dos anfitriones que comparten el enlace tiene cada uno una dirección única IP asignada para su conexión. Los ruteadores se añaden a la tabla de ruteo IP como lo harían para cualquier otra red.

Un nuevo modulo de software se añade en la capa de interfaz de red para controlar el nuevo enlace de hardware, pero no se realizan cambios sustanciales en el esquema de estratificación por capas. La principal desventaja del enfoque de redes independientes es la proliferación de números de redes (uno por cada conexión entre dos maquinas), lo que ocasiona que las tablas de ruteo sean tan grandes como sea necesario. Tanto la línea serial

IP (Serial Line IP o SLIP) como el protocolo punto a punto (Point to Point Protocol o PPP) tratan a cada enlace serial como una red separada.

El segundo método para ajustar las conexiones punto a punto evita asignar múltiples direcciones IP al cableado físico. En lugar de ello, se tratan a todas las conexiones colectivamente como una sola red independiente IP con su propio formato de trama, esquema de direccionamiento de hardware y protocolos de enlace de datos. Los ruteadores que emplean el segundo método necesitan solo un numero de red IP para todas las conexiones punto a punto.

Usar el enfoque de una sola red significa extender el esquema de estratificación por capas de protocolos para añadir una nueva capa de ruteo dentro de la red, entre la capa de interfaz de red y los dispositivos de hardware. Para las máquinas con una sola conexión punto a punto, una capa adicional parece innecesaria.

El programador que diseña software de ruteo dentro de la red determina exactamente como selecciona el software un enlace físico. Por lo general, el algoritmo conduce a una tabla de ruteo dentro de la red. La tabla de ruteo dentro de la red es análoga a una tabla de ruteo de una red de redes en la que se especifica una transformación de la dirección de destino hacia la ruta. La tabla contiene pares de enteros, (D, L), donde D es una dirección de destino de un anfitrión y L especifica una de las líneas físicas utilizadas para llegar al destino.

Las diferencias entre una tabla de ruteo de red de redes y una tabla de ruteo dentro de la red son que esta ultima, es mucho más pequeña. Contiene solamente información de ruteo para los anfitriones conectados directamente a la red punto a punto. La razón es simple: la capa Internet realiza la transformación de una dirección de destino arbitraria hacia una ruta de dirección especifica antes de pasar el datagrama hacia una interfaz de red. De esta manera, la capa dentro de la red solo debe distinguir entre máquinas en una sola red punto a punto.

4.2.4.2. La Desventaja De La Estratificación Por Capas.

La estratificación por capas es una idea fundamental que proporciona las bases para el diseño de protocolos. Permite al diseñador dividir un problema complicado en subproblemas y resolver cada parte de manera independiente. Por desgracia, el software resultante de una estratificación por capas estrictas puede ser muy ineficaz. Si se considera el trabajo de la capa de transporte, debe aceptar un flujo de octetos desde un programa de

aplicación, dividir el flujo en paquetes y enviar cada paquete a través de la red de redes. Para optimizar la transferencia, la capa de transporte debe seleccionar el tamaño de paquete más grande posible que le permita a un paquete viajar en una trama de red. En particular, si la máquina de destino está conectada a una máquina de la misma red de la fuente, solo la red física se verá involucrada en la transferencia, así, el emisor puede optimizar el tamaño del paquete para esta red. Si el software preserva una estricta estratificación por capas, sin embargo, la capa de transporte no podrá saber como ruteará él modulo de Internet él trafico o que redes están conectadas directamente. Mas aún, la capa de transporte no comprenderá el datagrama o el formato de trama ni será capaz de determinar como deben ser añadidos muchos octetos de encabezado a un paquete. Así, una estratificación por capas estricta impedirá que la capa de transporte optimice la transferencia.

Por lo general, las implantaciones atenúan el esquema estricto de la estratificación por capas cuando construyen software de protocolo. Permiten que información como la selección de ruta y la MTU de red se propaguen hacia arriba. Cuando los buffers realizan el proceso de asignación, generalmente dejan espacio para encabezados que serán añadidos por los protocolos de las capas de bajo nivel y pueden retener encabezados de las tramas entrantes cuando pasan hacia protocolos de capas superiores. Tal optimización puede producir mejoras notables en la eficiencia siempre y cuando conserve la estructura básica en capas.

4.2.5. COMANDOS TCP/IP

TCP/IP incluye dos grupos de comandos utilizados para suministrar servicios de red:

- Los comandos remotos BERKELEY
- Los comandos DARPA

Los comandos remotos BERKELEY, que fueron desarrollados en la Universidad Berkeley (California), incluyen órdenes para comunicaciones entre sistemas operativos UNIX, como copia remota de archivos, conexión remota, ejecución de shell remoto, etc.

Permiten utilizar recursos con otros hosts, pudiendo tratar distintas redes como si fueran una sola.

En la versión 4 para UNIX Sistema V, se pueden distinguir los siguientes comandos más comunes:

RCP	Realiza una copia de archivos al mismo o a otro servidor
RLOGINGL-RLOGINVT	Se utiliza para hacer una conexión al mismo o a otro servidor
REXEC-RSH	Permite ejecutar comandos del sistema operativo en el mismo o en otro servidor.

Los comandos DARPA incluyen facilidades para emulación de terminales, transferencia de archivos, correo y obtención de información sobre usuarios. Pueden ser utilizadas para comunicación con computadores que ejecutan distintos sistemas operativos.

En la versión 2.05 para DOS, dependiendo de las funciones que realizan, se pueden distinguir los siguientes grupos de comandos:

Kernel PC/TCP y herramientas asociadas

Se utilizan para cargar el núcleo TCP/IP en la memoria del computador.

BOOTP	Asigna la dirección IP de la estación de trabajo
INET	Descarga el núcleo PC/TCP de la memoria y/o realiza estadísticas de red
KERNEL	Carga el núcleo TCP/IP en la memoria y lo deja residente

Configuración de la red

Permiten configurar TCP/IP con determinados parámetros.

IFCONFIG	Configura el hardware para TCP/IP
IPCONFIG	Configura el software TCP/IP y la dirección IP

Transferencia de archivos

Se utilizan para transferir archivos entre distintos computadores.

DDATES	Muestra las fechas y horas guardadas en un archivo creado con el comando TAR
FTP	Transfiere archivos entre una estación de trabajo y un servidor
FRPSRV	Convierte una estación de trabajo en un servidor
FTP	
PASSWD	Se utiliza para poner contraseñas en las estaciones de trabajo a los usuarios para poder utilizar él comando FTPSRV
RMT	Permite realizar copia de archivos en una unidad de cinta
TAR	Realiza una copia de archivos creando un único archivo de BACKUP
TFTP	Transfiere archivos entre una estación de trabajo un servidor o a otra estación de trabajo sin necesidad de validar al usuario

Impresión

Permiten el control de la impresión en las impresoras conectadas al servidor.

DOPREDIR	Imprime un trabajo de impresión que aún no ha sido impreso
IPRINT	Envía un texto o un archivo a un servidor de impresoras de imagen
LPQ	Indica el estado de la cola de impresión indicada
PR	Envía un texto o un archivo a una impresora local o de red.
LPRM	Elimina trabajos pendientes de la cola de impresión
ONPREDIR	Realiza tareas de configuración para el comando PREDIR
PREDIR	Carga o descarga el programa que permite la impresión remota y lo deja residente.

PRINIT Se usa con los comandos PREDIR y ONPREDIR

PRSTART Indica a la estación de trabajo remota que imprima un archivo usando la configuración por defecto

Conexión a servidores

Permiten la conexión de los computadores a servidores de nuestra red.

SUPDUP Permite conectarse a otro servidor de la red

TELNET - TN Es el método normal de conectarse a un servidor de la red

Información sobre los usuarios

Muestran información sobre los usuarios conectados a la red.

FINGER Muestra información sobre un usuario conectado a otra estación de trabajo

NICNAME Muestra información sobre un usuario o sobre un servidor solicitada al centro de información de redes

WHOIS Muestra información sobre un usuario registrado que esté conectado a otra estación de trabajo

Envío y recepción de correo

Estos comandos permiten el envío y/o recepción de correo entre los usuarios de la red.

MAIL Permite enviar y recibir correo en la red

PCMAIL Permite leer correo. Se ha de usar con el comando VMAIL

POP2 - POP3 Se utiliza para leer correo. Se han de usar con VMAIL Y SMTP

SMTP Se utiliza para enviar correo en la red

SMTPSRV Permite leer el correo recibido

VMAIL Es un comando que muestra una pantalla preparada para leer el correo recibido. Se utiliza en conjunción con los comandos PCMAIL, POP2 0 POP3

Chequeo de la red

Permiten chequear la red cuando aparecen problemas de comunicaciones.

HOST Indica el nombre y la dirección IP de una estación de trabajo determinada

ING Envía una Llamada a una estación de trabajo e informa si se puede establecer conexión o no con ella

SETCLOCK Muestra la fecha y la hora que tiene la red

4.2.6. COMO FUNCIONA TCP/IP

Una red TCP/IP transfiere datos mediante el ensamblaje de bloques de datos en paquetes, cada paquete comienza con una cabecera que contiene información de control; tal como la dirección del destino, seguido de los datos. Cuando se envía un archivo por la red TCP/IP, su contenido se envía utilizando una serie de paquetes diferentes. El Internet protocol (IP), un protocolo de la capa de red, permite a las aplicaciones ejecutarse transparentemente sobre redes interconectadas. Cuando se utiliza IP, no es necesario conocer que hardware se utiliza, por tanto ésta corre en una red de área local.

El Transmissión Control Protocol (TCP); un protocolo de la capa de transporte, asegura que los datos sean entregados, que lo que se recibe, sea lo que se pretendía enviar y que los paquetes que sean recibidos en el orden en que fueron enviados. TCP terminará una conexión si ocurre un error que haga la transmisión fiable imposible.

4.2.7. ADMINISTRACION TCP/IP

TCP/IP es una de las redes más comunes utilizadas para conectar computadoras con sistema UNIX. Las utilidades de red TCP/IP forman parte de la versión 4, muchas facilidades de red como un sistema UUCP, el sistema de correo, RFS y NFS, pueden utilizar una red TCP/CP para comunicarse con otras máquinas.

Para que la red TCP/IP esté activa y funcionando será necesario:

- Obtener una dirección Internet.
- Instalar las utilidades Internet en el sistema
- Configurar la red para TCP/IP
- Configurar los guiones de arranque TCP/IP
- Identificar otras máquinas ante el sistema
- Configurar la base de datos del o y ente de STREAMS
- Comenzar a ejecutar TCP/IP.

4.3. PROTOCOLO TCP (TRANSFER CONTROL PROTOCOL).

4.3.1. INTRODUCCIÓN

A finales de los años 60 y principios de los 70, las redes no estaban diseñadas de forma que fuera posible compartir recursos entre redes diferentes.

Desde entonces se ha hecho cada vez más necesario que las aplicaciones de usuario compartan recursos. Pero para que puedan hacerlo, los administradores de las redes deben acordar primero un conjunto de tecnologías y normas comunes para que las redes puedan comunicarse. Es por eso que las aplicaciones como la transferencia de archivos y el correo electrónico se deberían estandarizar también para permitir la interacción entre aplicaciones de usuario.

El Protocolo de Control de Transmisión se desarrolló con esos objetivos.

El propósito primordial de TCP es proporcionar circuitos lógicos confiables o servicios de conexión entre parejas de procesos. Esto no implica confiabilidad desde protocolos de más bajo nivel (como IP) así que TCP debe garantizar esto por sí mismo.

4.3.2. PROTOCOLO DE CONTROL DE TRANSMISIÓN (TCP)

TCP reside en el nivel de transporte del modelo de niveles convencional. Está situado entre IP y los niveles superiores. TCP no está cargado en las pasarelas. Está diseñado para residir en los computadores o en las máquinas que se ocupan de conservar la integridad de la

transferencia de datos entre extremos. Lo más común es que TCP resida en los computadores de usuario.

Como IP es una red no orientada a conexión, es TCP quien se debe encargar de las tareas de fiabilidad, control de flujo, secuenciamiento, aperturas y cierres. Aunque TCP e IP estén tan relacionados que incluso se les denomine juntos "TCP/IP", TCP puede soportar otros protocolos. Por ejemplo, otro protocolo no orientado a conexión como el ISO 8473 (Protocolo de Redes No Orientado a Conexión o CNLP) podría funcionar con TCP (si se realizan algunos ajustes de los interfaces entre módulos). Además, los protocolos de aplicación, como el Protocolo de Transferencia de Correo Simple (de siglas en inglés, SMTP) se apoyan en muchos servicios que proporciona TCP

4.3.3. PRINCIPALES CARACTERÍSTICAS DE TCP

TCP suministra una serie de servicios a los niveles superiores. Esta sección presenta brevemente esos servicios.

TCP es un protocolo orientado a conexión. Esto quiere decir que TCP mantiene información del estado de cada cadena de datos de usuario que circula por él. El término utilizado en este contexto significa también que TCP es responsable de la transferencia de datos entre extremos por la red o redes hasta la aplicación de usuario receptora (o el protocolo de nivel superior). TCP debe asegurar que los datos se transmiten y reciben correctamente por los computadores atravesando las correspondientes redes.

Como TCP es un protocolo orientado a conexión, es responsable de la transferencia fiable de cada uno de los caracteres (bytes u octetos) que reciben del nivel superior correspondiente. En consecuencia, utiliza números de secuencia y aceptaciones/rechazos.

El término asociado con estos aspectos de los protocolos orientados a conexión es el de circuito virtual.

Cada octeto transmitido lleva asignado un número de secuencia. El módulo TCP receptor utiliza una rutina de checksum para comprobar la posible existencia de daños en los datos producidos en el proceso de transmisión. Si los datos son aceptables, TCP envía una acepción positiva (ACK) al módulo TCP remitente. Si los datos han resultado dañados, el TCP receptor los descarta y utiliza un número de secuencia para informar al TCP remitente del problema. Como muchos otros protocolos orientados a conexión, TCP emplea

temporizadores para garantizar que no transcurre un lapso de tiempo demasiado grande antes de la transmisión de aceptaciones desde el nodo receptor y/o de la transmisión de datos desde el nodo transmisor.

TCP recibe datos de un protocolo de nivel superior de forma orientada a cadenas. Esto es diferente a muchos otros protocolos empleados en la industria. Los protocolos orientados a cadenas se diseñan para enviar caracteres separados y no bloques, tramas, datagramas, etc. Los datos son enviados por un protocolo de nivel superior en forma de cadenas, byte a byte. Cuando llegan al nivel TCP, los bytes son agrupados para formar segmentos TCP. Dichos segmentos se transfieren a IP (o a otro protocolo de nivel inferior) para su transmisión al siguiente destino. La longitud de los segmentos de la determina TCP, aunque el realizador de un determinado sistema puede determinar la forma en que TCP toma su decisión. Los realizadores de TCP que han trabajado con sistemas orientados a bloques, como los sistemas operativos de IBM, puede que tengas que modificar ligeramente su forma de pensar acerca de las prestaciones de TCP. TCP admite el uso de segmentos de longitud variable, debido a su diseño orientado a cadenas. Por tanto, las aplicaciones que trabajan normalmente con bloques de datos de longitud fija (una aplicación de gestión de personal que envíe registros de empleados de longitud fija o una aplicación de gestión de nóminas con registros de pago también longitud fija) no pueden utilizar TCP para transmitir bloques fijos al receptor. El nivel de aplicación debe ocuparse de configurar los bloques dentro de las cadenas de TCP.

TCP comprueba también la duplicidad de los datos. En el caso de que el TCP remitente decida retransmitir los datos, el TCP descarta los datos redundantes. Estos datos redundantes podrían aparecer en la interred, por ejemplo cuando el TCP receptor no acepta el tráfico de manera temporizada, en cuyo caso el TCP remitente decidirá retransmitir los datos.

Además de la capacidad de transmisión de cadenas, TCP soporta también el concepto de función push. Esta función se utiliza cuando una aplicación desea asegurarse de que todos los datos que han pasado al nivel inferior se han transmitido. Para hacer eso, gobierna la gestión del buffer de TCP. Para obtener esta función, el protocolo de nivel superior envía una orden a TCP con un identificador de parámetro de push a 1. Esta operación implica que TCP envía todo el tráfico almacenado en forma de segmento o segmentos hacia su destino.

Además de utilizar los números de secuencia para aceptaciones, TCP los utiliza para la reordenación de los segmentos que llegan a su destino fuera de orden. Como TCP descansa sobre un protocolo no orientado a conexión, es bastante posible que en la interred se creen datagramas duplicados. TCP también elimina los segmentos duplicados.

TCP emplea un esquema de aceptación inclusiva. El número de aceptación acepta todos los octetos hasta (e incluyendo) el del número de aceptación menos uno. Este esquema es un método muy sencillo y eficiente de aceptar tráfico, pero presenta una desventaja. Por ejemplo, supongamos que se han transmitido diez segmentos y debido a las operaciones realizadas durante el proceso de encaminamiento llegan desordenados. TCP está obligado aceptar sólo el mayor número de bytes contiguos recibidos sin error. No está permitido aceptar el byte de mayor número recibido hasta que hayan llegado todos los bytes intermedios. Por tanto, como en cualquier otro protocolo orientado a conexión, podría transcurrir el periodo de temporización de aceptaciones y TCP transmisora retransmitiría el tráfico no aceptado todavía. Esas retransmisiones podrían introducir una considerable sobrecarga en la red.

El módulo TCP receptor se ocupa también de controlar el flujo de los datos del transmisor, lo que es muy útil para evitar el desbordamiento de los dispositivos de almacenamiento y la saturación de la máquina receptora. La idea que utiliza TCP es algo poco usual en protocolos de comunicación. Se basa en enviar el dispositivo transmisor un valor de "ventana". Se permite que el transmisor envíe un número máximo de bytes igual al valor de su ventana. Cuando se ha llegado a ese valor, la ventana se cierra y el transmisor debe interrumpir el envío de datos.

Además, TCP posee una facilidad muy útil que permite multiplexar varias sesiones de usuario en un mismo computador. Esta operación se realiza definiendo algunas convenciones para compartir puertos y sockets entre usuarios.

TCP proporciona transmisión en modo dúplex integral entre las entidades que se comunican. De esta forma, la transmisión se puede efectuar en ambos sentidos sin necesidad de esperar a la señal de indicación de cambio de sentido, necesaria en las transmisiones semidúplex. Además, TCP permite a los usuarios especificar niveles de seguridad y prioridades de las conexiones. Aunque esas opciones no están incluidas en todos los protocolos TCP, están definidas en el estándar TCP.

TCP proporciona el cierre seguro de los circuitos virtuales (la conexión lógica entre dos usuarios). El cierre seguro se ocupa de que todo el tráfico sea reconocido antes de la desactivación del circuito virtual.

4.3.4. APERTURAS ACTIVA Y PASIVA

Los puertos TCP pueden establecer dos tipos de conexiones. El modo de apertura pasiva permite que el protocolo de nivel superior (por ejemplo, un servidor) indique al TCP y al sistema operativo del computador que va a esperar la llegada de solicitudes de conexión procedentes del sistema remoto, en lugar de enviar una apertura activa. Tras recibir esta solicitud, el sistema operativo asigna un número de puerto a este extremo. Esta utilidad se puede usar para realizar comunicaciones con usuarios remotos sin tener el retardo de la apertura activa.

Los procesos de aplicaciones que solicitan la apertura pasiva pueden aceptar una solicitud de cualquier usuario (supuesto que se cumplen algunos requisitos de compatibilidad). Si se puede aceptar cualquier llamada (sin requisitos de compatibilidad) el número de socket exterior se pone a ceros. Los números de socket exterior no especificados sólo se permiten en aperturas pasivas.

La segunda forma de establecimiento de conexión es el modo de apertura activa. En esta situación, el protocolo de nivel superior designa específicamente otro socket por el que establecer la conexión. Típicamente, se envía la apertura activa a un puerto con apertura pasiva para establecer un circuito virtual.

TCP admite un escenario en el que se envían dos aperturas activas de un sistema a otro a la vez. TCP realizará la conexión. Esta característica permite que las aplicaciones envíen una apertura en cualquier momento, sin preocuparse de si la otra aplicación ha enviado otra apertura o no.

TCP establece convenciones estrictas sobre cómo se deben utilizar conjuntamente las aperturas activas y pasivas. En primer lugar, una apertura activa identifica un socket específico, así como sus niveles de prioridad y de seguridad. TCP garantiza una apertura si el socket remoto tiene una apertura pasiva compatible, o si ha enviado una apertura activa compatible.

4.3.5. EL BLOQUE DE CONTROL DE TRANSMISIÓN (TCB)

Como TCP debe recordar varias cosas de cada conexión virtual, almacena esa información en un Bloque de Control de Transmisión (TCB). Entre la información que se almacena en la TCB destacamos los números de socket local y remoto, los punteros a los buffers de transmisión y recepción, los punteros a la cola de retransmisión, los valores de seguridad y prioridad de la conexión y el segmento en curso. La TCB también contiene varias variables asociadas a los números de secuencia de envío y recepción.

4.3.6. EL SEGMENTO TCP (PDU)

Las PDU que se intercambian entre dos módulos TCP se denominan segmentos. El segmento se divide en dos partes, la parte de cabecera y la parte de datos. La parte de datos sigue a la parte de cabecera. Los primeros dos campos del segmento se denominan puerto de fuente y puerto de destino. Esos campos de 16 bits identifican a los programas de aplicación de nivel superior que utilizan la conexión TCP.

El siguiente campo se denomina número de secuencia. Este campo contiene el número de secuencia del primer octeto del campo de datos de usuario. Su valor especifica la posición de la cadena de bits del módulo transmisor. Dentro del segmento especifica el primer octeto de datos de usuario.

El número de secuencia se utiliza también durante la operación de gestión de la conexión. Si dos entidades TCP utilizan el segmento de solicitud de conexión, entonces el número de secuencia especifica el número de secuencia de envío inicial (ISS) que se utilizará para la numeración subsiguiente de los datos de usuario.

El valor del número de aceptación permite aceptar los datos previamente recibidos. Este campo contiene el valor del número de secuencia del siguiente octeto que se espera recibir del transmisor. Con esa definición permite la aceptación inclusiva, en el sentido de que permite la aceptación de todos los octetos hasta, e incluyendo, el valor de este número menos 1.

El campo de desplazamiento de datos especifica el número de palabras alineadas de 32 bits de que consta la cabecera de TCP. Este campo se utiliza para determinar dónde comienza el campo de datos.

Como puede esperarse, el campo reservado está reservado. Consta de 6 bits que deben valer cero. Estos bits están reservados para usos futuros.

Los seis bits siguientes se denominan indicadores (flags). Son bits de control de TCP y se utilizan para especificar ciertos servicios o utilidades que se pueden emplear durante la sesión. El valor de algunos de esos bits indica cómo interpretar otros campos de la cabecera. Los seis bits mencionados llevan la siguiente información.

URG indica que el campo de puntero de urgencia es significativo.

ACK indica si el campo de aceptación es significativo.

PSH significa que el módulo va a utilizar la función push.

RST indica que la conexión se va a inicializar.

SYN indica que se van a sincronizar los números de secuencia; se utiliza en los segmentos de establecimiento de conexión como indicación de que se van a realizar algunas operaciones de preparación.

FIN indica que el remitente no tiene más datos para enviar. Es comparable a la señal de fin de transmisión (EOT) en otros protocolos.

El campo siguiente, denominado ventana, se pone a un valor que indica cuántos octetos desea aceptar el receptor. Este valor se establece teniendo en cuenta el valor del campo de aceptación (número de aceptación). La ventana se establece sumando los valores del campo de ventana y del campo de número de aceptación.

El campo de checksum contiene el complemento de 1 a 16 bits del complemento a 1 de la suma de todas las palabras de 16 bits del segmento, incluyendo la cabecera del texto. El propósito de este cálculo es determinar si el segmento procedente del transmisor ha llegado libre de errores.

El siguiente campo del segmento, denominado puntero de urgente, se utiliza sólo si el indicador de URG está a 1. El objeto de este puntero es identificar el octeto de datos al que siguen datos urgentes. Los datos urgentes se denominan datos fuera de bada. TCP no dice lo que hay que hacer con los datos urgentes. Depende de la implementación. Dicho de otro modo, sólo se indica el lugar donde empiezan los datos urgentes, no lo que hay que hacer

con ellos. El valor de este campo es un desplazamiento del número de secuencia y apunta al octeto a partir de cual siguen los datos urgentes.El campo de opciones está concebido para posibilitar futuras mejoras de TCP. Está diseñado de forma semejante al campo de opción de los datagramas de IP, en el sentido de que cada opción se especifica mediante un byte que especifica el número de opción, un campo que contiene la longitud de la opción y finalmente, los valores de la opción propiamente dichos.

Actualmente el campo de opción tiene un uso bastante limitado, y el estándar TCP sólo especifica tres opciones:

0: fin de lista de opciones

1: no operación

2: tamaño máximo de segmento

Finalmente, el campo de relleno asegura que la cabecera TCP ocupa un múltiplo par de 32 bits. Finalmente, como se muestra en la figura, siguen los datos de usuario.

4.4. PROTOCOLO IP (INTERNET PROTOCOL).

4.4.1. INTRODUCCION

El protocolo IP es el software que implementa el mecanismo de entrega de paquetes sin conexión y no confiable (técnica del mejor esfuerzo).

El protocolo IP cubre tres aspectos importantes:

1. Define la unidad básica para la transferencia de datos en una interred, especificando el formato exacto de un Datagrama IP.
2. Realiza las funciones de enrutamiento.
3. Define las reglas para que los Host y Routers procesen paquetes, los descarten o generen mensajes de error.

4.4.2. EL DATAGRAMA IP

El esquema de envío de IP es similar al que se emplea en la capa Acceso a red. En esta ultima se envían Tramas formadas por un Encabezado y los Datos. En el Encabezado se incluye la dirección física del origen y del destino.

En el caso de IP se envían Datagramas, estos también incluyen un Encabezado y Datos, pero las direcciones empleadas son Direcciones IP.

Encabezado	Datos

4.4.2.1. Formato del Datagrama IP

Los Datagramas IP están formados por Palabras de 32 bits. Cada Datagrama tiene un mínimo (y tamaño más frecuente) de cinco palabras y un máximo de quince.

Ver	Hlen	TOS	Longitud Total	
Identificación			Flags	Desp. De Fragmento
TTL		Protocolo	Checksum	
Dirección IP de la Fuente				
Dirección IP del Destino				
Opciones IP (Opcional)				Relleno
DATOS				

Tabla 12. Formato del datagrama IP

- Ver: Versión de IP que se emplea para construir el Datagrama. Se requiere para que quien lo reciba lo interprete correctamente. La actual versión IP es la 4.
- Hlen: Tamaño de la cabecera en palabras.
- TOS: Tipo de servicio. La gran mayoría de los Host y Routers ignoran este campo. Su estructura es:

Prioridad	D	T	R	Sin Uso

La prioridad (0 = Normal, 7 = Control de red) permite implementar algoritmos de control de congestión más eficientes. Los tipos D, T y R solicitan un tipo de transporte dado: D = Procesamiento con retardos cortos, T = Alto Desempeño y R = Alta confiabilidad. Nótese que estos bits son solo "sugerencias", no es obligatorio para la red cumplirlo.

- Longitud Total: Mide en bytes la longitud de doto el Datagrama. Permite calcular el tamaño del campo de datos: Datos = Longitud Total – 4 * Hlen.

Antes de continuar con la segunda palabra del Datagrama IP, hace falta introducir conceptos relacionados con la fragmentación.

4.4.2.2. Fragmentación

En primer lugar, De qué tamaño es un Datagrama?. El tamaño para un Datagrama debe ser tal que permita la encapsulación, esto es, enviar un Datagrama completo en una trama física. El problema está en que el Datagrama debe transitar por diferentes redes físicas, con diferentes tecnologías y diferentes capacidades de transferencia. A la capacidad máxima de transferencia de datos de una red física se le llama MTU (el MTU de ethernet es 1500 bytes por trama, la de FDDI es 4497 bytes por trama). Cuando un Datagrama pasa de una red a otra con un MTU menor a su tamaño es necesaria la fragmentación. A las diferentes partes de un Datagrama se les llama fragmento. Al proceso de reconstrucción del Datagrama a partir de sus fragmentos se le llama Reensamblado de fragmentos.

El control de la fragmentación de un Datagrama IP se realiza con los campos de la segunda palabra de su cabecera:

- **Identificación:** Numero de 16 bits que identifica al Datagrama, que permite implementar números de secuencias y que permite reconocer los diferentes fragmentos de un mismo Datagrama, pues todos ellos comparten este numero.

- **Banderas:** Un campo de tres bits donde el primero está reservado. El segundo, llamado bit de No - Fragmentación significa: 0 = Puede fragmentarse el Datagrama o 1 = No puede fragmentarse el Datagrama. El tercer bit es llamado Más – Fragmentos y significa: 0 = Unico fragmento o Ultimo fragmento, 1 = aun hay más fragmentos. Cuando hay un 0 en más – fragmentos, debe evaluarse el

campo desp. De Fragmento: si este es cero, el Datagrama no esta fragmentado, si es diferente de cero, el Datagrama es un ultimo fragmento.

- **Desp. De Fragmento:** A un trozo de datos se le llama Bloque de Fragmento. Este campo indica el tamaño del desplazamiento en bloques de fragmento con respecto al Datagrama original, empezando por el cero.

Para finalizar con el tema de fragmentación, hay que mencionar el Plazo de Reensamblado, que es un time out que el Host destino establece como máximo para esperar por todos los fragmentos de un Datagrama. Si se vence y aun no llegan TODOS, entonces se descartan los que ya han llegado y se solicita el reenvío del Datagrama completo.

4.4.2.3. Formato del Datagrama IP.

- TTL: Tiempo de Vida del Datagrama, especifica el numero de segundos que se permite al Datagrama circular por la red antes de ser descartado.

- Protocolo: Especifica que protocolo de alto nivel se empleó para construir el mensaje transportado en el campo datos de Datagrama IP. Algunos valores posibles son: 1 = ICMP, 6 = TCP, 17 = UDP, 88 = IGRP (Protocolo de Enrutamiento de Pasarela Interior de CISCO).

- Checksum: Es un campo de 16 bits que se calcula haciendo el complemento a uno de cada palabra de 16 bits del encabezado, sumándolas y haciendo su complemento a uno. Esta suma hay que recalcularla en cada nodo intermedio debido a cambios en el TTL o por fragmentación.

- Dirección IP de la Fuente:

- Dirección IP del Destino:

- Opciones IP: Existen hasta 40 bytes extra en la cabecera del Datagrama IP que pueden llevar una o más opciones. Su uso es bastante raro.

- Uso de Ruta Estricta (Camino Obligatorio)

- Ruta de Origen Desconectada (Nodos Obligatorios)

- Crear registro de Ruta
- Marcas de Tiempo
- Seguridad Básica del Departamento de Defensa
- Seguridad Extendida del Departamento de Defensa

4.4.2.4. Enrutamiento IP.

Enrutar es el proceso de selección de un camino para el envío de paquetes. La computadora que hace esto es llamada Router.

En general se puede dividir el enrutamiento en Entrega Directa y Entrega Indirecta. La Entrega Directa es la transmisión de un Datagrama de una maquina a otra dentro de la misma red física. La Entrega Indirecta ocurre cuando el destino no esta en la red local, lo que obliga al Host a enviar el Datagrama a algún Router intermedio. Es necesario el uso de mascaras de subred para saber si el Host destino de un Datagrama esta o no dentro de la misma red física.

4.4.2.5. Encaminamiento con Salto al Siguiente.

La forma más común de enrutamiento requiere el uso de una Tabla de Enrutamiento IP, presente tanto en los Host como en los Routers. Estas tablas no pueden tener información sobre cada posible destino, de hecho, esto no es deseable. En ves de ello se aprovecha el esquema de direccionamiento IP para ocultar detalles acerca de los Host individuales, además, las tablas no contienen rutas completas, sino solos la dirección del siguiente paso en esa ruta.

En general una tabla de encaminamiento IP tiene pares (Destino, Router), donde destino es la dirección IP de un destino particular y Router la dirección del siguiente Router en el camino hacia destino. Nótese que Router debe ser accesible directamente desde la maquina actual.

Este tipo de encaminamiento trae varias consecuencias, consecuencia directa de su naturaleza estática:

Todo trafico hacia una red particular toma el mismo camino, desaprovechando caminos alternativos y el tipo de trafico.

Solo el Router con conexión directa al destino sabe si este existe o esta activo.

Es necesario que los Routers cooperen para hacer posible la comunicación bidireccional.

4.4.2.5.1. Algoritmo de Enrutamiento IP

```
Ruta Datagrama(Datagrama) {
  Extrae de la Cabecera de Datagrama la dirección de destino D;
  Extrae de D el prefijo de Red N;
  Si N corresponde a cualquier dirección directamente conectada Entonces
    Envía el Datagrama a D sobre la Red N;
  Sino
    Si en la tabla hay una ruta especifica para D Entonces
      Envía Datagrama al salto siguiente especificado;
    Sino
      Si En la tabla hay una ruta para la red N Entonces
        Envía Datagrama al salto siguiente especificado;
      Sino
        Si En la tabla hay una ruta por defecto Entonces
          Envía el Datagrama a la dirección por defecto;
        Sino
          Declarar Fallo de Enrutamiento;
        Fsi
      Fsi
    Fsi
  Fsi
}
```

4.4.2.6. Manejo de Datagramas Entrantes.

Cuando un Datagrama llega a un Host, el software de red lo entrega a IP. IP verifica la dirección de destino y si esta concuerda con la de la maquina local, entonces acepta el Datagrama y lo entrega a las capas superiores. De no coincidir la dirección de destino, el Datagrama es descartado.

Por otra parte, un Router que reciba un Datagrama compara la dirección de destino con la suya propia. Si coinciden, el Datagrama pasa a las capas superiores, sino, se le aplica el algoritmo de encaminamiento y se reenvía el Datagrama.

4.4.2.7. Direccionamiento sin Clase

Durante la introducción a TCP/IP (Juan Carlos Guevara), se explicaba como mediante el empleo de Mascaras de subred, se lograba convertir una única red (generalmente una Clase B) en múltiples redes lógicas interconectadas y administradas por la organización propietaria. El problema se presenta cuando el crecimiento explosivo de las redes locales produce el fenómeno ROADS (Running Out of Address Space), que consiste simplemente en el agotamiento del espacio de direcciones útil, causado por la gran demanda de las direcciones Clase B, de las cuales solo hay 16.384, mientras que las Clases C permanecían sin Asignar (pues aunque hay 2.097.152 de ellas, nadie las quiere por ser muy pequeñas).

Para enfrentar este problema se desarrollo el esquema de Direcciones sin Clase, que consiste en asignar a una misma organización un bloque continuo de direcciones de Clase C. De esta manera, una organización que requiera conectar a Internet un numero moderado de Hosts (digamos 3.800) puede recibir un bloque de 16 redes continuas de Clase C (por ejemplo, de la red Clase C 199.40.72.0 a la 199.40.87.0), con lo cual dispone de 4.096 direcciones IP validas para administrar.

4.4.2.8. CIDR Enrutamiento Inter – Dominio Sin Clases (Classless Inter – Domain Routing)

El esquema de direcciones sin clase genera el problema de aumentar la información que debe incluirse en las tablas de enrutamiento. En el caso del ejemplo, se tendría que incluir 16 nuevas entradas en cada tabla de enrutamiento de cada Host y Router. CIDR resuelve el problema al incluir en las tablas información acerca del tamaño de los bloques y el numero de bloques, así, en las tablas de enrutamiento IP se tienen pares (Destino, Router), donde destino no es una dirección de Host o Red tradicional, sino que incluye información acerca del numero de redes que incluye el bloque (en nuestro ejemplo, 16) y el tamaño de cada una de esas redes (en el ejemplo, son Clases C, 256 direcciones cada una).

El Direccionamiento sin clase modifica la estructura de una dirección IP, de esta manera:

Prefijo de Red	Identificador de Host

Así, CIDR debe incluir en las tablas de enrutamiento cual es la primera red que compone el bloque, cuantos bits se emplean como Prefijo de Red y la mascara de subred que se emplea. En nuestro ejemplo, las tablas de enrutamiento IP contendrían esta información:

199.40.72.0/20 255.255.240.0

Refiriéndose a un bloque que se inicia con la red 199.40.72.0 y que tiene 20 bits en el prefijo de red. La mascara 255.255.240.0 (11111111.11111111.11110000.00000000) nos indica que se están usando 4 bits extra (los que se han resaltado) para identificar a las redes que componen al bloque. Nótese que cuatro bits permites agrupar precisamente 16 redes Clase C.

Un aspecto importante que hay que subrayar es que en ningún momento cambia el algoritmo básico de enrutamiento IP, lo que cambia es el contenido de las tablas. Además, las nuevas tablas contienen información resumida, por lo que buscar una dirección destino en la tabla se hace de otra manera, pero el algoritmo permanece inalterado.

El problema de buscar direcciones de destino en una tabla, consiste en que cualquier dirección cuya mascara de destino tenga menos bits, incluye a la que tiene mas bits. Con esto quiero decir que una mascara de subred como 255.255.0.0 (11111111.11111111.00000000.00000000, es decir, 16 bits de prefijo de red) incluye dentro de si a la mascaras de subred 255.255.128.0 (11111111.11111111.10000000.00000000, 17 bits de prefijo de red) y esta a su ves incluye a la mascara 255.255.192.0 (11111111.11111111.11000000.00000000) y en general, entre menos bits tiene el prefijo de red, mas direcciones Host abarca. Por esta razón cuando se explora la tabla de enrutamiento IP en busca de una dirección de destino, se hace una búsqueda que inicia con las mascaras de más bits y termina en la de menos bits. Es decir, se inicia con mascaras como

255.255.255.255 (todo en uno) y se continua con la 255.255.255.254 (31 unos y un cero) y así sucesivamente. Esto quiere decir que tendrían que hacerse 32 recorridos secuenciales a la tabla, lo cual es muy ineficiente en cuanto a tiempo, pues además de ser un procedimiento demorado, se sabe ya que direcciones normales de Clase B (255.255.0.0) requieren 16 barridos a la tabla, además, hacen falta 32 barridos para notar que no hay una entrada en la tabla para esas dirección. Por esta razón se emplean otros métodos para hacer estas búsquedas en las tablas de enrutamiento IP. Un esquema muy popular emplea un Arbol Binario, en el cual cada bit representa una nueva rama en el árbol. Así, en nuestro ejemplo, podrían dividirse las direcciones asignadas a la organización (4.096) en subredes de esta forma: dos subredes de 1.024 direcciones cada una, tres de 512 y dos de 256 direcciones. De esta forma, el árbol resultante tendría esta forma:

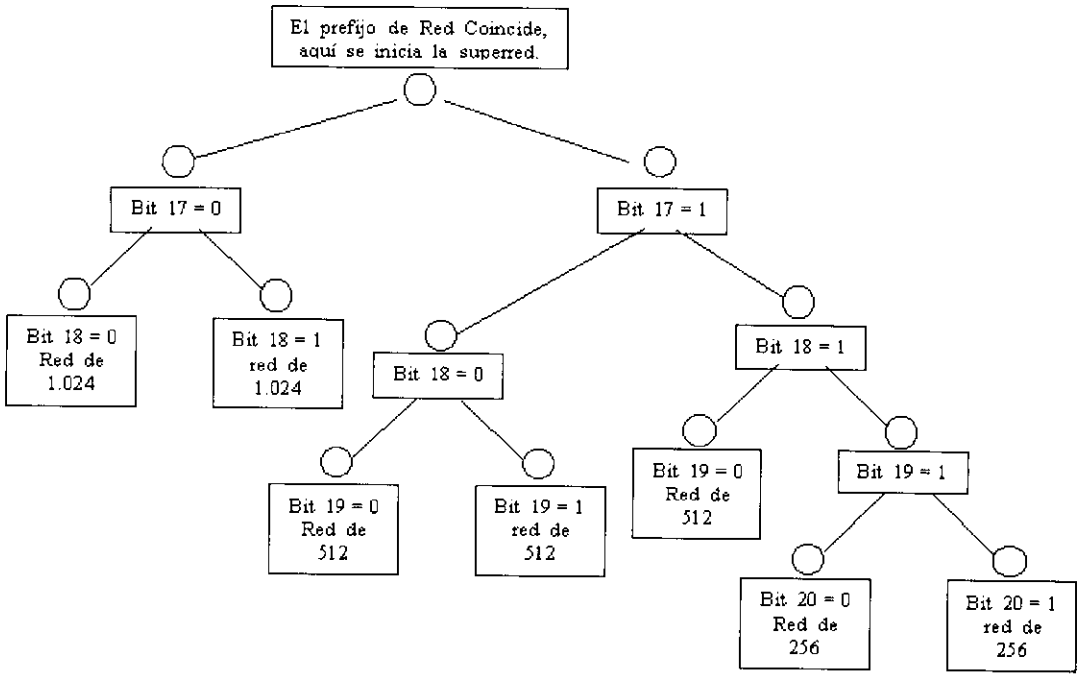


Fig. 14. Árbol Binario

4.5. CONCLUSIONES

Los protocolos de Internet se han convertido en una de las familias de protocolos más ampliamente utilizada en el mundo. Están diseñados para facilitar la intercomunicación de redes de computadores.

Los protocolos Internet constan de muchos protocolos diseñados para dar soporte a las operaciones de comunicación entre redes.

Además, los protocolos interred contienen una gran cantidad de protocolos de nivel de aplicación, como TELNET, el protocolo de transferencia de archivos (FTP) y el protocolo simple de transferencia de correo (SMTP)

CAPÍTULO V

5. ESTRUCTURACION DE LA BASE DE DATOS EN SQL_SERVER.

5.1. FUNDAMENTOS DE SQL SERVER .

Microsoft® SQL Server™ es una base de datos relacional cliente-servidor basada en el Lenguaje de consulta estructurado (SQL, *Structured Query Language*). Cada uno de estos términos describe una parte fundamental de la arquitectura de SQL Server.

5.1.1. BASE DE DATOS

Una base de datos es similar a un archivo de datos en cuanto a que ambos son un almacenamiento de datos. Como en un archivo de datos, una base de datos presenta información directamente al usuario; el usuario ejecuta una aplicación que tiene acceso a los datos de la base de datos y los presenta al usuario en un formato inteligible.

Los sistemas de bases de datos son más útiles que los archivos de datos. Los datos están mucho mejor organizados. En una base de datos bien diseñada, no hay elementos de datos duplicados que el usuario o la aplicación tengan que actualizar al mismo tiempo. Los elementos de datos están agrupados en una única estructura o registro, y se pueden definir relaciones entre dichas estructuras y registros.

Una base de datos suele tener dos componentes: los archivos que almacenan la base de datos física, el software del sistema de administración de la base de datos (DBMS, Database Management System), que las aplicaciones utilizan para tener acceso a los datos. El DBMS es el responsable de mantener la estructura de la base de datos, lo que incluye:

- El mantenimiento de las relaciones entre los datos de la base de datos.
- La garantía de que los datos estén correctamente almacenados y de que no se infrinjan las reglas que definen las relaciones entre los datos.
- La recuperación de todos los datos hasta un punto coherente en caso de fallos del sistema.

5.1.2. BASE DE DATOS RELACIONAL

Hay varias formas de organizar los datos en las bases de datos, pero las bases de datos relacionales son una de las formas más efectivas. Los sistemas de bases de datos relacionales son una aplicación de la teoría matemática de los conjuntos al problema de la organización de los datos. En una base de datos relacional, los datos están organizados en tablas (llamadas relaciones en la teoría relacional).

Al organizar los datos en tablas, se pueden encontrar varias formas de definirlas. La teoría de las bases de datos relacionales define un proceso, la normalización, que asegura que el conjunto de tablas definido organizará los datos de manera eficaz.

5.1.3. CLIENTE-SERVIDOR

En los sistemas cliente-servidor, el servidor es un equipo relativamente grande situado en una ubicación central que administra recursos utilizados por varios individuos. Cuando los individuos tienen que utilizar un recurso, se conectan con el servidor desde sus equipos, o clientes, a través de la red.

En la arquitectura cliente-servidor de las bases de datos, los archivos de la base de datos y software DBMS residen en el servidor. Se proporcionan componentes de comunicaciones para que las aplicaciones se puedan ejecutar en equipos cliente y se comuniquen con el servidor de bases de datos a través de la red. El componente de comunicación de SQL Server también permite la comunicación entre una aplicación que se ejecute en el servidor y SQL Server.

Las aplicaciones del servidor suelen poder trabajar con varios clientes al mismo tiempo. SQL Server puede operar con miles de aplicaciones cliente simultáneas. El servidor tiene funciones que impiden que se produzcan problemas de lógica si un usuario intenta leer o modificar los datos actualmente utilizados por otros usuarios.

Aunque SQL Server ha sido diseñado como servidor para redes cliente-servidor, también puede funcionar directamente como base de datos independiente en el cliente. Esta escalabilidad y la facilidad de uso de las funciones de SQL Server le permiten funcionar eficazmente en los clientes, sin consumir demasiados recursos.

5.1.4. LENGUAJE DE CONSULTA ESTRUCTURADO (SQL)

Para trabajar con los datos de una base de datos, tiene que utilizar un conjunto de comandos e instrucciones (lenguaje) definidos por el software del DBMS. En las bases de datos relacionales se pueden utilizar distintos lenguajes, el más común es SQL. Los estándares de SQL han sido definidos por el American National Standards Institute (ANSI) y la International Standards Organization (ISO). La mayor parte de los productos DBMS modernos aceptan el nivel básico de SQL-92, el último estándar de SQL (publicado en 1992).

5.2. QUE ES MICROSOFT SQL SERVER?

5.2.1. CARACTERÍSTICAS DE SQL SERVER

Microsoft® SQL Server™ proporciona soporte para un conjunto de características que aportan las siguientes ventajas:

- Facilidad de instalación, distribución y utilización

SQL Server incluye un conjunto de herramientas administrativas y de desarrollo que mejoran la capacidad para instalar, distribuir, administrar y utilizar SQL Server entre varios sitios.

- Escalabilidad

Puede utilizarse el mismo motor de base de datos a través de plataformas que van desde equipos portátiles que ejecutan Microsoft Windows® 95 ó 98 hasta grandes servidores con varios procesadores que ejecutan Microsoft Windows NT®, Enterprise Edition.

- Almacenamiento de datos

SQL Server incluye herramientas para extraer y analizar datos resumidos para el proceso analítico en línea (OLAP, Online Analytical Processing). SQL Server incluye también herramientas para diseñar gráficamente las bases de datos y analizar los datos mediante preguntas en lenguaje normal.

- Integración del sistema con otro software de servidor

SQL Server se integra con el correo electrónico, Internet y Windows.

5.3. ARQUITECTURA SQL SERVER.

5.3.1. ARQUITECTURA CLIENTE-SERVIDOR

Microsoft® SQL Server™ está diseñado para operar de forma eficiente en varios entornos:

- Como sistema de base de datos cliente-servidor de dos estratos o de varios estratos
- Como sistema de base de datos de escritorio

5.3.1.1. Sistemas de bases de datos Cliente-Servidor.

Los sistemas cliente-servidor están construidos de tal modo que la base de datos puede residir en un equipo central, llamado servidor y ser compartida entre varios usuarios. Los usuarios tienen acceso al servidor a través de una aplicación de cliente o de servidor:

- En un sistema cliente-servidor de dos estratos, los usuarios ejecutan una aplicación en su equipo local, llamado cliente, que se conecta a través de la red con el servidor que ejecuta SQL Server. La aplicación de cliente ejecuta las reglas de la compañía y el código necesario para presentar el resultado al usuario; también se conoce como cliente amplio.
- En un sistema cliente-servidor de varios componentes, la lógica de la aplicación de cliente se ejecuta en dos ubicaciones:
    - El cliente reducido se ejecuta en el equipo local del usuario y se encarga de presentar resultados al usuario.
    - La lógica de la compañía se encuentra en aplicaciones de servidor que se ejecutan en un servidor. Los clientes reducidos solicitan funciones a la aplicación de servidor, que, a su vez, es una aplicación multiproceso capaz de operar con varios usuarios simultáneos. La aplicación de servidor es la que abre las conexiones con el servidor de la base de datos y se puede ejecutar en el mismo servidor que la base de datos, o se puede conectar a través de la red con otro servidor que opere como servidor de base de datos. Éste es el escenario típico de las aplicaciones de Internet.

El tener los datos almacenados y administrados en una ubicación central ofrece varias ventajas:

- Todos los elementos de datos están almacenados en una ubicación central en donde todos los usuarios pueden trabajar con ellos.

  No se almacenan copias separadas del elemento en cada cliente, lo que elimina los problemas de hacer que todos los usuarios trabajen con la misma información.

- Las reglas de la organización y las reglas de seguridad se pueden definir una sola vez en el servidor para todos los usuarios.

  Esto se puede hacer en una base de datos mediante el uso de restricciones, procedimientos almacenados y desencadenadores. También se puede hacer en una aplicación de servidor.

- Los servidores de base de datos relacionales optimizan el tráfico de la red al devolver sólo los datos que la aplicación necesita.

- Las gastos en hardware se pueden minimizar.

  Como los datos no están almacenados en los clientes, éstos no tienen que dedicar espacio de disco a almacenarlos. Los clientes tampoco necesitan la capacidad de proceso para administrar los datos localmente y el servidor no tiene que dedicar capacidad de proceso para presentar los datos.

  El servidor se puede configurar para optimizar la capacidad de E/S de disco necesaria para obtener los datos y los clientes se pueden configurar para optimizar el formato y presentación de los datos obtenidos desde el servidor.

  El servidor puede estar situado en una ubicación relativamente segura y estar equipado con dispositivos como Sistemas de alimentación ininterrumpida (SAI), lo que resulta más económico que si se protegieran todos los clientes.

- Las tareas de mantenimiento como las copias de seguridad y restauración de los datos son más sencillas porque están concentradas en el servidor central .

En los sistemas cliente-servidor grandes, miles de usuarios pueden estar conectados con una instalación de SQL Server al mismo tiempo. SQL Server tiene una protección completa para dichos entornos, con barreras de seguridad que impiden problemas como tener varios usuarios intentando actualizar el mismo elemento de datos a la vez. SQL

Server también asigna eficazmente los recursos disponibles entre los distintos usuarios, como la memoria, el ancho de banda de la red y la E/S de disco.

Las aplicaciones SQL Server se pueden ejecutar en el mismo equipo que SQL Server. La aplicación conecta con SQL Server utilizando componentes de comunicación entre procesos (IPC, *Interprocess Communications*) de Windows, como la memoria compartida, en lugar de la red. Esto permite que SQL Server se utilice en sistemas pequeños en los que las aplicaciones tienen que almacenar los datos localmente.

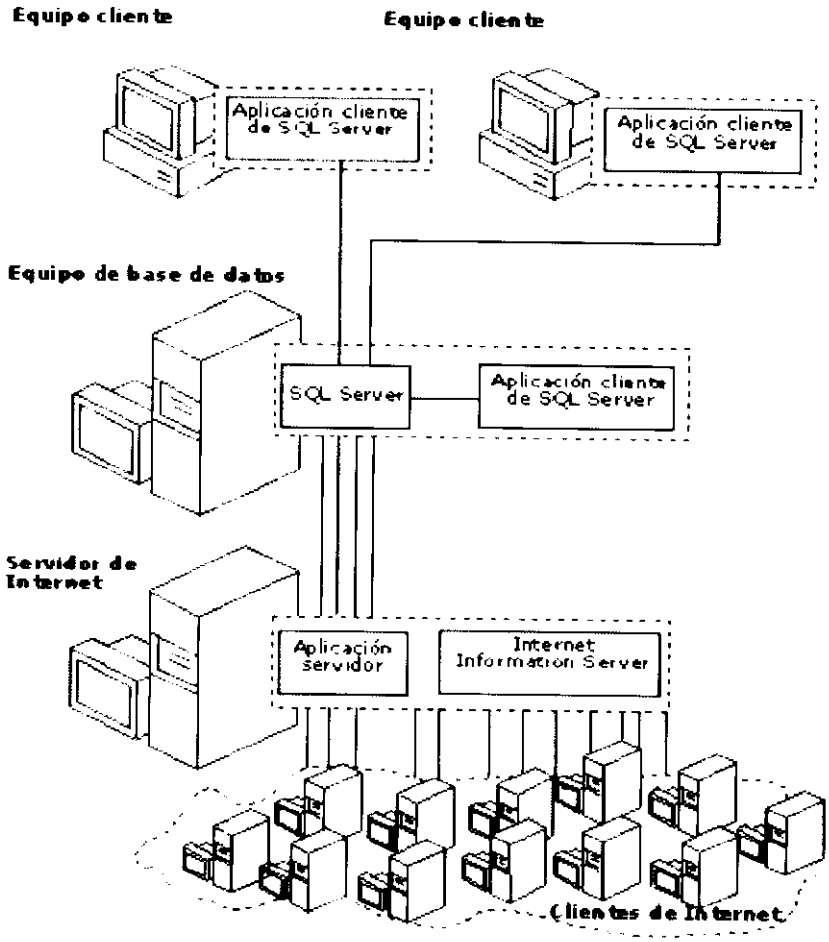


Fig. 15. Bases de datos Cliente - Servidor

5.3.1.2. Sistemas de bases de datos de escritorio.

Aunque SQL Server funciona muy eficientemente como servidor, también se puede utilizar en aplicaciones que necesiten bases de datos independientes almacenadas de forma local en el cliente. SQL Server se puede autoconfigurar dinámicamente para que se ejecute más eficientemente con los recursos disponibles en el cliente, sin tener que dedicar un administrador de bases de datos a cada cliente. Los fabricantes de aplicaciones también pueden incrustar SQL Server como componente de almacenamiento de datos en sus aplicaciones.

Cuando los clientes utilizan bases de datos SQL Server locales, una copia del motor de bases de datos de SQL Server se ejecuta en el cliente y administra todas las bases de datos de SQL Server de dicho cliente. Las aplicaciones conectan con el motor de la base de datos casi de la misma forma en que se conectan a través de la red con un motor de base de datos que se ejecuta en un servidor remoto.

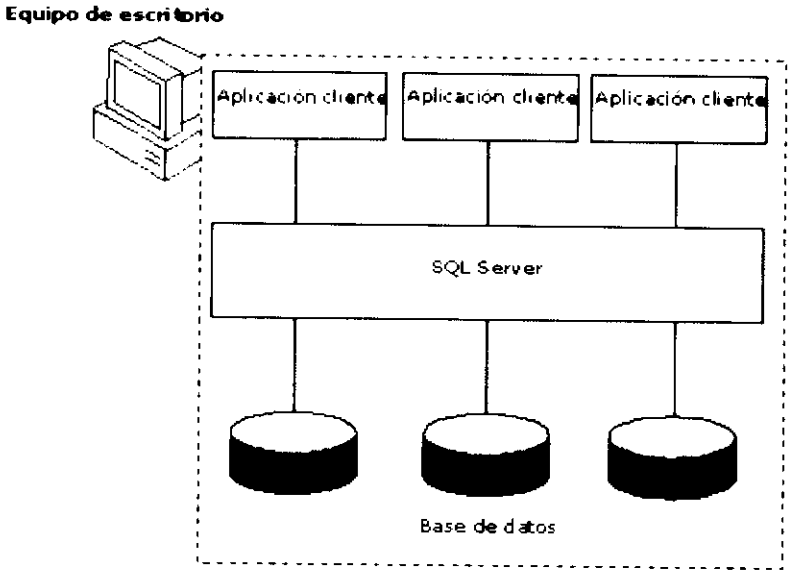


Fig. 16. Bases de datos de escritorio

5.4. SEGURIDAD EN SQL SERVER.

5.4.1. REGLAS PARA LOS NOMBRES DE INICIO DE SESIÓN, USUARIOS, FUNCIONES Y CONTRASEÑAS DE SQL SERVER.

Los nombres de inicio de sesión, usuarios, funciones y contraseñas de Microsoft® SQL Server™ pueden contener de 1 a 128 caracteres y pueden incluir letras, símbolos y números (por ejemplo **Agustina Rivera**, **Federico Couto** o **13&#57abc)**. Por lo tanto, los nombres de usuario de Microsoft Windows NT® y Microsoft Windows® 95 o 98 pueden usarse como nombres de inicio de sesión de SQL Server. Sin embargo, en las instrucciones de Transact-SQL algunos símbolos sólo se pueden utilizar si el nombre de inicio de sesión, usuario, función o contraseña están delimitados por comillas dobles (") o corchetes ([ ]). Debe utilizar delimitadores en las instrucciones de Transact-SQL cuando el nombre de inicio de sesión, usuario, función o contraseña de SQL Server:

- Contenga o comience por un espacio.
- Comience por el carácter **$** o **@**.

Nota No es necesario especificar los delimitadores al escribir los nombres de inicio de sesión, usuarios, funciones y contraseñas en los cuadros de texto de las herramientas gráficas de cliente de SQL Server como, por ejemplo, el Administrador corporativo de SQL Server.

Además, un nombre de inicio de sesión, usuario o función de SQL Server no puede:

- Contener el carácter barra diagonal inversa (\), salvo que se refiera a un usuario o grupo existente de Windows NT. La barra diagonal inversa separa el nombre del equipo o dominio de Windows NT del nombre del usuario.
- Existir ya en la base de datos actual (o en **master**, en el caso de los nombres de inicio de sesión).
- Ser NULL, o una cadena vacía ("").

5.4.2. DISEÑAR LA SEGURIDAD.

El propósito de un diseño de la seguridad es identificar qué usuarios pueden ver qué datos y qué actividades pueden realizar en la base de datos. Se siguen una serie de pasos dependiendo de las condiciones presentar los requisitos de seguridad:

1. Enumerar todos los elementos y actividades de la base de datos que deban controlarse con un plan de seguridad.

2. Identificar los individuos y grupos de la organización.

Establecer las referencias cruzadas entre las dos listas para identificar los datos que puede ver cada usuario y las actividades que puede realizar en la base de

5.4.3. SEGURIDAD DE LOS DATOS.

Una de las funciones de una base de datos consiste en proteger los datos; para ello, se impide que determinados usuarios vean o modifiquen datos importantes y se evita que los usuarios en general cometan errores graves. El sistema de seguridad de Microsoft® SQL Server™ controla qué usuarios pueden trabajar con qué tipo de datos y qué usuarios pueden realizar determinadas actividades en la base de datos.

5.4.4. SEGURIDAD DE PAQUETE.

Los paquetes DTS proporcionan dos niveles de seguridad: propietario y usuario. Cuando un paquete se guarda, se le puede asignar una contraseña de propietario, de usuario o ambas. Los usuarios que desean modificar el paquete deben especificar la contraseña de propietario y los usuarios que desean ejecutarlo deben especificar la contraseña de propietario o la de usuario.

Cuando se ha asignado a un paquete la contraseña de propietario, el paquete se almacena en formato cifrado. Cuando un paquete o versión DTS se cifra, también se cifran todas sus colecciones y propiedades, excepto las propiedades **Name**, **Description**, **ID**, **VersionID** y **CreationDate**.

Los usuarios también deben tener permiso para obtener acceso al medio de almacenamiento que contiene el paquete que desean utilizar. Los usuarios deben tener permisos de archivo para abrir el archivo DTS de almacenamiento con estructura COM que contiene los paquetes. Los usuarios deben tener permisos para iniciar una sesión en la base de datos **msdb** que contiene los paquetes del almacenamiento de Microsoft® SQL Server™ o de Microsoft Repository, y el inicio de sesión también debe tener permisos para

obtener acceso a las tablas del sistema que contienen la definición de los paquetes. Los usuarios deben tener permisos de lectura para ejecutar un paquete y permisos de actualización para modificarlo.

5.4.5. ADMINISTRAR LA SEGURIDAD.

Para garantizar que sólo tengan acceso a los datos y a los objetos almacenados en Microsoft® SQL Server™ los usuarios autorizados, la seguridad debe configurarse correctamente. Conocer cómo se configura correctamente la seguridad puede ayudarle a simplificar la administración continua. Entre los elementos de seguridad que puede ser necesario configurar están los modos de autenticación, los inicios de sesión, los usuarios, las funciones, otorgar, revocar y denegar permisos de objetos e instrucciones Transact-SQL, y el cifrado de datos.

Parte de la información puede compartirse con todos los miembros de la organización o, incluso, con el público en general. La información más reservada y confidencial se guarda bajo llave en archivadores, o en cámaras subterráneas. Para tener acceso a esta información, es necesario tener las credenciales adecuadas.

Aunque el término "seguridad" se utiliza habitualmente en el contexto de la prevención de robos, también puede evitar que las personas cometan costosos errores. . En cuanto a las operaciones de una organización, una base de datos ofrece mecanismos para almacenar, administrar y controlar la información

Los errores no intencionados y el acceso no autorizado a los datos pueden suponer un costo muy alto para una organización.

Es fundamental que el programador de una base de datos no haga ninguna suposición acerca de lo que harán los usuarios con ella. Además de los controles integrados en las aplicaciones, la propia base de datos debe tener un sistema de seguridad sólido para controlar las actividades que pueden realizarse y la información que se puede visualizar o cambiar. Este enfoque de la seguridad garantiza la protección de los datos, independientemente de la forma en que los usuarios tengan acceso a la base de datos y de lo que hagan en ella.

5.4.6. ARQUITECTURA DE SEGURIDAD.

La arquitectura de un sistema de seguridad se basa en los usuarios y grupos de usuarios, a los que se hace referencia como las Principales de Seguridad. Esta ilustración muestra el modo en que los usuarios y los grupos locales y globales de Microsoft® Windows NT® se pueden asignar a cuentas de seguridad de Microsoft SQL Server™, y cómo SQL Server puede controlar las cuentas de seguridad de forma independiente al sistema de cuentas de Windows NT.

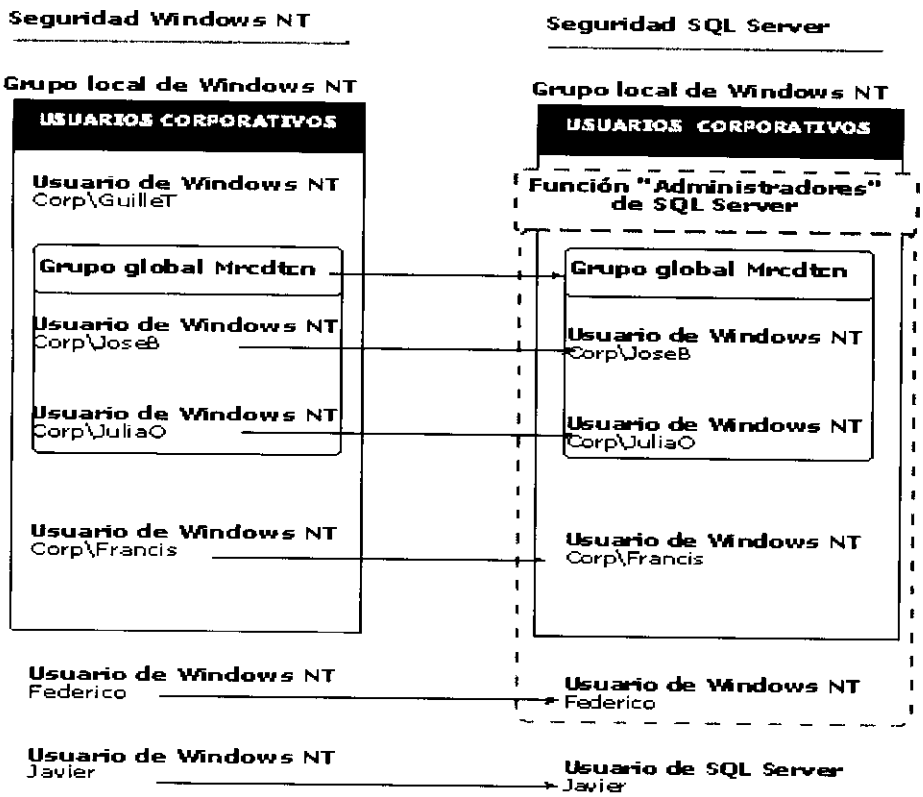


Fig. 17. Arquitectura de seguridad

El grupo local **USUARIOSCORP** contiene dos usuarios y un grupo global, **Mrcdtcn**, que también contiene dos usuarios. SQL Server permite utilizar directamente los grupos locales y globales de Windows NT para organizar sus cuentas de usuario. Además, los usuarios de Windows NT **Pedro** y **Javier**, que no pertenecen a un grupo de Windows NT, pueden agregarse a SQL Server directamente como usuarios de Windows NT (por ejemplo, **Pedro**), o como usuarios de SQL Server (**Javier**).

SQL Server amplía aún más el modelo anterior con el uso de las funciones. Las funciones

son grupos de usuarios organizados con fines administrativos, de modo similar a lo que ocurre con los grupos de Windows NT. Se puede utilizar las funciones para organizar usuarios cuando no exista un grupo equivalente en Windows NT. Por ejemplo, la función **Administradores** contiene el grupo global **Mredtcn** de Windows NT y los usuarios de Windows NT **Francisco** y **Pedro**.

SQL Server ofrece también seguridad en las aplicaciones, mediante el uso de funciones de aplicación de base de datos individuales.

5.4.6.1. Seguridad de aplicaciones y funciones de aplicación.

El sistema de seguridad de Microsoft® SQL Server™ se implementa en el ámbito más bajo, el de la propia base de datos. Éste es el método mejor y más eficaz para controlar las actividades de los usuarios, independientemente de la aplicación que utilicen para comunicar con SQL Server.

Además, puede ser conveniente que el acceso de los usuarios a los datos se restrinja únicamente a establecerse a través de una aplicación específica, sin que se pueda producir un acceso directo, por ejemplo, con el Analizador de consultas de SQL Server o con Microsoft Excel. Así se evita que un usuario que se conecte con SQL Server y utilice una aplicación como el Analizador de consultas de SQL Server ejecute una consulta poco eficiente y afecte al rendimiento de todo el servidor.

SQL Server contempla esta situación mediante el uso de funciones de aplicación. Las diferencias básicas entre las funciones ordinarias y las funciones de aplicación son las siguientes:

- Las funciones de aplicación no contienen miembros. No se puede agregar usuarios, grupos de Microsoft Windows NT® ni funciones a las funciones de aplicación, y los permisos de la función de aplicación se obtienen cuando ésta se activa para la conexión del usuario a través de una o varias aplicaciones específicas. La asociación de un usuario a una función de aplicación se debe a su capacidad de ejecutar una aplicación que activa esa función, y no a ser miembro de ella.

- De forma predeterminada, las funciones de aplicación están desactivadas, y es necesaria una contraseña para activarlas con el procedimiento almacenado del sistema sp_setapprole. La contraseña puede ser especificada por el usuario, por

ejemplo, a través de una petición de la aplicación, pero es más habitual que esté integrada en la aplicación. La contraseña puede cifrarse al enviarla a SQL Server.

- Cuando una aplicación activa una función de aplicación para una conexión, la conexión pierde, de forma permanente, todos los permisos aplicados al nombre de inicio de sesión, a la cuenta de usuario o a otros grupos o funciones de base de datos de todas las bases de datos mientras se mantenga la conexión. La conexión obtiene los permisos asociados a la función de aplicación para la base de datos en la que la función de aplicación existe. Las funciones de aplicación sólo pueden aplicarse en las bases de datos en las que existen, por lo que la conexión sólo podrá tener acceso a otra base de datos con los permisos asignados a la cuenta de usuario invitado de esa base de datos. Por tanto, si en una base de datos no existe la cuenta invitado, la conexión no tendrá acceso a la base de datos. Si en la base de datos existe la cuenta de usuario invitado, pero no tiene asignados explícitamente los permisos de acceso a un objeto, la conexión no tendrá acceso a ese objeto, independientemente de quién lo haya creado. Los permisos que el usuario obtenga de la función de aplicación se mantienen hasta el término de la sesión en SQL Server.

Las funciones de aplicación permiten a la aplicación asumir la responsabilidad de la autenticación de los usuarios, y relevan a SQL Server de esa tarea. Sin embargo, SQL Server precisa autenticar la aplicación cuando tiene acceso a las bases de datos. La aplicación debe entonces especificar una contraseña, ya que no hay otro modo de autenticarla.

Nota Las funciones de aplicación funcionan con los dos modos de autenticación.

5.4.7. CONFIGURAR CUENTAS DE SEGURIDAD.

Cada usuario debe obtener acceso a Microsoft® SQL Server™ a través de una cuenta que establece su capacidad de conectarse (autenticación). A continuación, esta cuenta de inicio de sesión se asigna a una cuenta de usuario de SQL Server que se utiliza para controlar las actividades realizadas en la base de datos (validación de permisos). Por lo tanto, se asigna un mismo nombre de inicio de sesión a cada cuenta creada en cada base de datos a la que el usuario precise acceso. Si en una base de datos no hay ninguna cuenta de usuario, el

usuario no podrá tener acceso a ella, aunque pueda conectarse con SQL Server.

La cuenta de inicio de sesión se crea en Microsoft Windows NT®, en lugar de en SQL Server. A continuación, se concede a esta cuenta (una cuenta de usuario o de grupo de Windows NT) el permiso para conectarse con SQL Server.

La cuenta de inicio de sesión se crea en SQL Server.

Las cuentas de usuario de SQL Server asignadas a los nombres de inicio de sesión (de Windows NT o de SQL Server), y que permiten el acceso a la base de datos, siempre se crean en cada base de datos de SQL Server.

5.4.8. ADMINISTRAR LAS CUENTAS DE SEGURIDAD.

Cuando se hayan agregado cuentas de seguridad a Microsoft® SQL Server™, puede modificarlas de acuerdo a las variaciones de las necesidades de su organización. Normalmente, esto implica visualizar, modificar y quitar las cuentas de seguridad de la base de datos para adaptarlas a las necesidades de cada momento.

5.4.9. ADMINISTRAR PERMISOS.

Cuando los usuarios se conectan con Microsoft® SQL Server™, las actividades que pueden llevar a cabo se determinan con los permisos otorgados a sus cuentas de seguridad, a los grupos de Microsoft Windows NT® o a las jerarquías de funciones a las que pertenecen sus cuentas de seguridad. El usuario debe tener los permisos adecuados para realizar cualquier actividad que implique cambiar la definición de una base de datos o el acceso a su contenido. Algunas actividades como, por ejemplo, las que no cambien ningún elemento de la base de datos ni supongan acceso a su contenido, no necesitan permisos.

5.4.9.1. Validación de permisos.

Una vez que se ha autenticado un usuario y se le ha permitido iniciar una sesión en Microsoft® SQL Server™ mediante su nombre de inicio de sesión, es necesario que tenga una cuenta en cada base de datos a la que precise tener acceso. El hecho de requerir una cuenta de usuario en cada base de datos evita que el usuario pueda conectarse a SQL Server y tener acceso a todas las bases de datos de un servidor.

La cuenta de usuario de cada base de datos se utiliza para aplicar los permisos de seguridad en los objetos (tablas, vistas, procedimientos almacenados, etcétera) que contiene la base

de datos. Esta cuenta de usuario puede asignarse desde cuentas de usuario de Microsoft Windows NT®, grupos de Windows NT de los que el usuario sea miembro o cuentas de inicio de sesión de SQL Server. Si no hay ninguna cuenta asignada directamente, el usuario puede trabajar en la base de datos con la cuenta **invitado**, si existe. Las actividades que se permite realizar al usuario se controlan con los permisos aplicados a la cuenta de usuario utilizada para el acceso a la base de datos.

SQL Server sólo acepta comandos después de que el usuario haya obtenido acceso a la base de datos. El usuario puede especificar comandos "ad hoc" o elegir opciones de menú de una aplicación. Todas las actividades que realiza el usuario en una base de datos se comunican a SQL Server a través de instrucciones de Transact-SQL. Cuando SQL Server recibe una instrucción de Transact-SQL, comprueba que el usuario tenga permiso para ejecutar esta instrucción en la base de datos. Si el usuario no tiene los permisos adecuados, ya sea para ejecutar una instrucción, o para tener acceso a un objeto de la base de datos que utilice la instrucción, SQL Server devolverá un error de permisos.

5.4.9.2. Cifrado.

El cifrado es un método para mantener la confidencialidad de la información reservada, y consiste en alterar los datos para darles un formato ilegible. El cifrado garantiza la seguridad de los datos, pues mantiene la información oculta a todos, incluso cuando los datos cifrados se visualicen directamente. El descifrado es el proceso de devolver a los datos cifrados su formato original, de modo que los usuarios autorizados puedan leerlos.

Microsoft® SQL Server™ cifra o puede cifrar:

- Las contraseñas de inicio de sesión y de funciones de aplicación almacenadas en SQL Server.
- Cualquier dato enviado entre el cliente y el servidor en paquetes de red.
- Las definiciones de los procedimientos almacenados.
- Las definiciones de las vistas.
- Las definiciones de los desencadenadores.

5.4.9.3. Autenticación.

Microsoft® SQL Server™ puede funcionar en uno de dos modos de seguridad (autenticación):

- Modo de autenticación de Windows NT (autenticación de Windows NT)
- Modo mixto (autenticación de Windows NT y autenticación de SQL Server)

El Modo mixto permite a los usuarios conectarse mediante la autenticación de Windows NT o la autenticación de SQL Server. Los usuarios que se conectan a través de una cuenta de usuario de Microsoft Windows NT® pueden utilizar las conexiones en las que se confía (conexiones validadas por Windows NT), ya sea en el Modo de autenticación de Windows NT o en el Modo mixto. Una vez establecida la conexión con el servidor SQL Server, el mecanismo de seguridad es el mismo en los dos modos.

Los sistemas de seguridad basados en nombres de inicio de sesión y contraseñas de SQL Server (autenticación de SQL Server) pueden resultar más fáciles de administrar que los basados en las cuentas de usuario y de grupo de Windows NT, especialmente en el caso de bases de datos que no sean fundamentales o aplicaciones sin información reservada o confidencial. Por ejemplo, es posible crear un único nombre de inicio de sesión y una contraseña para todos los usuarios de una aplicación, en lugar de crear todas las cuentas de usuario y de grupo de Windows NT necesarias. Sin embargo, con este método se elimina la posibilidad de controlar y hacer un seguimiento individualizado de las actividades de cada usuario y, por ello, no es recomendable.

5.4.10. LOS PROCEDIMIENTOS ALMACENADOS COMO MECANISMOS DE SEGURIDAD.

Los procedimientos almacenados pueden servir para personalizar los permisos de seguridad de forma muy similar a como lo hacen las vistas, y se usan habitualmente como interfaz sencilla para realizar actividades complejas. En una situación de almacenamiento de datos de archivo, los datos anteriores a un intervalo especificado se copian en una tabla de archivo y, a continuación, se eliminan de la tabla principal. Es posible utilizar permisos para evitar que los usuarios eliminen directamente las filas de la tabla principal, o que inserten filas en la tabla de archivo sin eliminarlas de la tabla principal. Puede crear un procedimiento que asegure que ambas actividades se realicen juntas, y entonces conceder a los usuarios permisos para ejecutar el procedimiento.

5.4.11. LAS VISTAS COMO MECANISMOS DE SEGURIDAD.

Las vistas pueden actuar como mecanismos de seguridad. Mediante una vista, los usuarios sólo pueden consultar y modificar los datos que ven. El resto de la tabla o base de datos no es visible, ni se tiene acceso a ella. El permiso de acceso al subconjunto de datos de la vista debe otorgarse, denegarse o revocarse, independientemente de los permisos aplicados en las tablas subyacentes.

Por ejemplo, suponga que la columna **salario** de una tabla contiene información confidencial acerca de los empleados, pero que el resto de las columnas contiene información que debe estar disponible para todos los usuarios. Puede definir una vista que incluya todas las columnas de la tabla, excepto **salario**. Si la tabla y la vista tienen el mismo propietario, al otorgar a un usuario el permiso SELECT para la vista, el usuario podrá ver las columnas no confidenciales de la vista sin tener ningún permiso en la tabla.

Al definir vistas diferentes y conceder selectivamente permisos en ellas, es posible restringir subconjuntos de datos distintos a determinados usuarios, grupos o funciones. Los ejemplos siguientes ilustran el uso de las vistas con fines de seguridad:

- El acceso puede restringirse a un subconjunto de las filas de la tabla base. Por ejemplo, podría definir una vista que sólo contuviera las filas que correspondan a libros de negocios y psicología, y mantener oculta al usuario la información acerca de libros de otros tipos.

- Es posible restringir el acceso a un subconjunto de las columnas de la tabla base. Por ejemplo, podría definir una vista que contuviera todas las filas de la tabla **titles** (títulos), pero que omitiera las columnas **royalty** (regalías) y **advance** (anticipo), ya que esta información es confidencial.

- Se puede restringir el acceso a un subconjunto de filas y columnas de la tabla base.

- Se puede restringir el acceso a las filas que cumplan una condición de combinación de más de una tabla base. Por ejemplo, podría definir una vista que combinara las tablas **title**, **authors** (autores) y **titleAuthor** (tituloAutor) para mostrar los nombres de los autores y los libros que han escrito. Esta vista ocultaría los datos personales de los autores y la información financiera de los libros.

- También puede restringirse el acceso a un resumen estadístico de los datos de la tabla base. Por ejemplo, podría definir una vista que sólo contuviera el precio medio de cada tipo de libro.

- Es posible restringir el acceso a un subconjunto de otra vista o de alguna combinación de vistas y tablas base.

5.4.12. REALIZAR COPIAS DE SEGURIDAD Y RESTAURAR BASES DE DATOS.

El componente para la realización de copias de seguridad y restauración de Microsoft® SQL Server™ proporciona una importante protección para los datos decisivos almacenados en bases de datos de SQL Server. La realización de copias de seguridad y la restauración de las bases de datos permite la restauración total de los datos en una amplia gama de problemas posibles en el sistema:

- Errores de medios

  Si se produce un error en una o más de las unidades de disco que contienen una base de datos, se enfrenta a la pérdida total de los datos a menos que pueda restaurar una copia anterior de los mismos.

- Errores de usuarios

  Si un usuario o una aplicación, ya sea por equivocación o intencionadamente, realiza un gran número de modificaciones no válidas en los datos, la mejor manera de solucionar el problema puede ser la restauración de los mismos con los de un momento dado anterior a aquel en que se realizaron las modificaciones.

- Pérdida permanente de servidores

  Si un servidor se deshabilita de manera permanente o se pierde un sitio debido a una catástrofe natural, es posible que deba activar un servidor de reserva activo o restaurar una copia de una base de datos en otro servidor.

Además, la realización de copias de seguridad y la restauración de bases de datos resulta útil para resolver problemas que no son del sistema, como mover o copiar una base de datos de un servidor a otro. Puede hacerse una copia de una base de datos rápida y fácilmente mediante la realización de una copia de seguridad de la base de datos de un

equipo y su restauración en otro.

5.4.12.1. Realizar una copia de seguridad de una base de datos.

La copia de seguridad de una base de datos consiste en una copia que se puede utilizar para restaurar la base de datos si se pierde. Al realizar una copia de seguridad de una base de datos, se copia todo el contenido de la base de datos, incluidas las partes necesarias del registro de transacciones.

El registro de transacciones es un registro de serie que mantiene todas las modificaciones realizadas en una base de datos y la transacción que ha realizado cada modificación. El registro de transacciones se utiliza durante las operaciones de recuperación para rehacer transacciones completadas y deshacer transacciones no completadas.

Al realizar una copia de seguridad de un registro de transacciones, se copian sólo los cambios producidos en el registro desde que se realizó la última copia de seguridad.

La copia de seguridad funciona como una instantánea aproximada de una base de datos o un registro de transacciones:

- La copia de seguridad de una base de datos registra el estado completo de la información de ésta en el momento en el que se completa la operación de copia de seguridad.

- La copia de seguridad de un registro de transacciones registra el estado del registro en el momento en el que se inicia la operación de copia de seguridad.

5.4.12.2. Restaurar una base de datos.

La restauración de una copia de seguridad de una base de datos devuelve la base de datos al mismo estado en el que se encontraba cuando se creó la copia. Las transacciones incompletas de la copia de seguridad de la base de datos (transacciones que no estaban completas cuando finalizó la operación de copia de seguridad) se deshacen para asegurar la coherencia de la base de datos.

La restauración de una copia de seguridad de un registro de transacciones vuelve a aplicar todas las transacciones completadas del registro a la base de datos. Cuando se aplica una copia de seguridad de un registro de transacciones, SQL Server va leyendo el registro de

transacciones y rehace cada una de las transacciones del registro. Cuando SQL Server llega al final del registro de transacciones, ha creado de nuevo el estado exacto de la base de datos en el momento en el que se inició la operación de copia de seguridad.

Nota Al realizar una copia de seguridad de una base de datos, no se realiza una copia de seguridad de los datos de índices de texto de los catálogos de texto. No obstante, si se han definido índices de texto para tablas, los metadatos de las definiciones de los índices de texto se almacenan en las tablas del sistema, en la base de datos que contiene los índices de texto. Por tanto, se realiza una copia de seguridad de los metadatos de los índices de texto cuando se crea una copia de seguridad de la base de datos. Una vez restaurada una copia de seguridad de la base de datos, se pueden volver a crear y llenar los catálogos de índices de texto.

5.5. BASES DE DATOS EN SQL-SERVER.

5.5.1. INTRODUCCIÓN A BASES DE DATOS.

Una base de datos de Microsoft® SQL Server™ consta de una colección de tablas con datos y otros objetos como vistas, índices, procedimientos almacenados y desencadenadores, que se definen para poder llevar a cabo distintas operaciones con datos. Los datos almacenados en una base de datos suelen estar relacionados con un tema o un proceso determinados como, por ejemplo, la información de inventario para el almacén de una fábrica.

SQL Server puede trabajar con un gran número de bases de datos y cada una de ellas puede almacenar datos interrelacionados o datos no relacionados con los de las otras bases de datos. Por ejemplo, un servidor podría tener una base de datos en la que se almacenen datos del personal y otra en la que se almacenen datos relacionados con los productos. Por otra parte, puede utilizarse una base de datos para almacenar datos acerca de pedidos actuales de los clientes y otra base de datos relacionada puede almacenar pedidos anteriores de los clientes que se utilicen para la elaboración de los informes anuales.

Antes de crear una base de datos, es importante entender las partes que la componen y cómo diseñarlas para asegurar que la base de datos funcione correctamente una vez implementada.

5.5.2. CREACION DE BASES DE DATOS.

Para crear una base de datos, debe determinar su nombre, su propietario (el usuario que crea la base de datos), su tamaño, los archivos y los grupos de archivos que se utilizarán para almacenarla.

Antes de crear una base de datos, considere lo siguiente:

- De forma predeterminada, tienen permiso para crear una base de datos las funciones fijas del servidor **sysadmin** y **dbcreator,** aunque se puede otorgar permisos a otros usuarios.
- El usuario que crea la base de datos se convierte en su propietario.
- En un servidor pueden crearse hasta 32.767 bases de datos.
- El nombre de la base de datos debe ajustarse a las reglas establecidas para los identificadores.

Se utilizan tres tipos de archivos para almacenar una base de datos:

- Principal

  El archivo principal contiene la información de inicio para la base de datos. Este archivo se utiliza también para almacenar datos. Cada base de datos tiene un único archivo principal.

- Secundario

  Los archivos secundarios contienen los datos que no se almacenan en el archivo de datos principal. No es necesario que las bases de datos tengan archivos de datos secundarios si el archivo principal es lo suficientemente grande como para contener todos los datos. Algunas bases de datos pueden ser muy grandes y necesitar varios archivos de datos secundarios o utilizar archivos secundarios en unidades de disco distintas, de modo que los datos estén distribuidos en varios discos.

- Registro de transacciones

  Contiene la información de registro que se utiliza para recuperar la base de datos. Debe haber al menos un archivo de registro de transacciones para cada base de datos, aunque puede haber más de uno. El tamaño mínimo para un archivo de registro es 512 kilobytes (KB).

Importante Los archivos de datos y de registro de transacciones de Microsoft® SQL Server™ no deben colocarse en sistemas de archivos comprimidos ni en unidades de red remotas como, por ejemplo, un directorio de red compartido.

Cuando se crea una base de datos, todos los archivos que la componen se llenan con ceros que sobrescriben los datos de los archivos ya eliminados que hubieran quedado en el disco. Aunque esto provoca que el proceso de creación de los archivos sea más largo, así se evita que el sistema operativo tenga que llenar los archivos con ceros cuando se escriban por primera vez datos en los archivos durante las operaciones habituales con la base de datos. Así se mejora el rendimiento de las operaciones cotidianas.

Es recomendable especificar el tamaño máximo de crecimiento del archivo. De ese modo se evita que se agote el espacio disponible en el disco al agregar datos. Para especificar un tamaño máximo para el archivo, utilice el parámetro MAXSIZE de la instrucción CREATE DATABASE o bien la opción **Restringir crecimiento (MB)** cuando utilice la página de propiedades del Administrador corporativo de SQL Server para crear la base de datos.

Puede crear bases de datos mediante Transact-SQL, el Administrador corporativo de SQL Server, el Asistente para creación de bases de datos o mediante programación con SQL-DMO.

Después de crear una base de datos, es recomendable que realice una copia de seguridad de la base de datos **master**.

5.5.2.1. Creación de Tablas.

Después de determinar el diseño de la base de datos, se puede crear las tablas en las que se vayan a almacenar los datos. Los datos suelen almacenarse en tablas permanentes. Las

tablas se almacenan en los archivos de la base de datos hasta que son eliminadas y están disponibles para cualquier usuario que cuente con los permisos correspondientes.

5.5.2.1.1. Tablas temporales

También puede crear tablas temporales. Las tablas temporales son similares a las permanentes, salvo por el hecho de que las tablas temporales se almacenan en tempdb y se eliminan automáticamente cuando ya no se utilizan.

Los dos tipos de tablas temporales, las locales y las globales, difieren en cuanto a sus nombres, visibilidad y vida útil. Las tablas temporales locales presentan un solo signo de número (#) como primer carácter del nombre; son visibles únicamente para la conexión actual del usuario y se eliminan cuando el usuario se desconecta de los equipos en los que se ejecuta Microsoft® SQL Server™. Las tablas temporales globales presentan dos signos de número (##) antes del nombre, son visibles para cualquier usuario después de su creación y se eliminan cuando todos los usuarios que hacen referencia a la tabla se desconectan de SQL Server.

Por ejemplo, si crea una tabla denominada empleados, puede ser utilizada por cualquier persona que cuente con los correspondientes permisos de seguridad establecidos para la base de datos hasta que la tabla sea eliminada. Si crea una tabla temporal local denominada #empleados, usted es la única persona que puede trabajar con la tabla, y se elimina cuando se desconecta. Si crea una tabla temporal global denominada ##empleados, cualquier usuario de la base de datos puede trabajar con esta tabla. Si ningún otro usuario trabaja con esta tabla después de que la cree, la tabla se elimina cuando se desconecte. Si otro usuario trabaja con la tabla después de que la cree, SQL Server la elimina cuando los dos se desconectan.

5.5.2.1.2. Propiedades de la tabla

Puede definir hasta 1.024 columnas por tabla. Los nombres de las tablas y de las columnas deben seguir las reglas para los identificadores; tienen que ser únicos dentro de una tabla determinada, pero puede utilizar el mismo número de columna en distintas tablas de la misma base de datos. También debe definir un tipo de datos para cada columna.

Aunque los nombres de las tablas tienen que ser únicos para cada propietario de una base de datos, puede crear varias tablas con el mismo nombre si especifica distintos propietarios

para cada tabla. Puede crear dos tablas denominadas empleados y designar a Juan como propietario de una y a Sara como propietaria de otra. Cuando necesite trabajar con una de las tablas de empleados, puede distinguir las tablas si especifica el propietario con el nombre de la tabla.

5.5.2.2. Modificación de Tablas.

Después de crear una tabla, puede cambiar muchas de las opciones que fueron definidas para la tabla cuando se creó originalmente; por ejemplo, es posible:

- Agregar, modificar o eliminar columnas. Por ejemplo, se puede cambiar el nombre, la longitud, el tipo de datos, la precisión, la escala y la aceptación de valores NULL de la columna, aunque hay algunas restricciones. Para obtener más información, consulte Modificar las propiedades de las columnas.

- Agregar o eliminar restricciones PRIMARY KEY y FOREIGN KEY.

- Agregar o eliminar restricciones UNIQUE y CHECK, así como definiciones (y objetos) DEFAULT.

- Agregar o eliminar una columna de identificadores mediante las propiedades IDENTITY o ROWGUIDCOL. Asimismo, es posible agregar o quitar la propiedad ROWGUIDCOL de una columna existente, aunque en una tabla sólo puede haber una columna que tenga la propiedad ROWGUIDCOL.

- Registrar una tabla y las columnas seleccionadas de una tabla para la indización de texto.

Para obtener más información acerca de las modificaciones que pueden realizarse en una tabla, consulte ALTER TABLE.

Asimismo, puede cambiar el nombre o el propietario de una tabla. Cuando lo haga, también deberá cambiar el nombre de la tabla en todos los desencadenadores, procedimientos almacenados, secuencias de comandos de Transact-SQL u otros códigos de programación que utilicen el nombre o propietario anterior de la tabla

5.5.2.3. Creación de Usuarios.

La colección Users contiene objetos User que hacen referencia a las definiciones de usuario de la base de datos de Microsoft® SQL Server™.

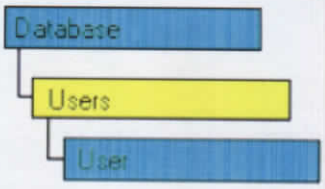


Fig. 18. Creación de usuaros

Observaciones

Un usuario de SQL Server constituye una parte de la implementación de la seguridad de SQL Server. Un usuario representa un inicio de sesión de SQL Server o una cuenta de seguridad de Microsoft Windows NT® con privilegios de acceso a datos de una base de datos de SQL Server.

Con la colección Users, puede:

- Crear un usuario de base de datos de SQL Server.
- Quitar un usuario de base de datos de SQL Server.

Para obtener más información acerca de cómo crear y quitar usuarios de una base de datos de SQL Server mediante el objeto User y la colección Users, consulte Objeto User.

Cuando se utiliza el método Item o Remove, la colección Users admite la identificación de los miembros utilizando bien el nombre o la sintaxis de referencia ordinal. Por ejemplo:

Set oUser = oDatabase.Users("anned")

O bien:

Set oUser = oDatabase.Users(2)

Nota Para crear o quitar usuarios de base de datos de SQL Server utilizando la colección Users son necesarios los privilegios apropiados. El inicio de sesión de SQL Server que se

usa para la conexión del objeto SQLServer debe ser un miembro de la función fija db_accessadmin o de una función con mayores privilegios.

5.5.3. MODIFICANDO BASES DE DATOS.

Después de crear una base de datos, es posible cambiar su definición original. Entre otros cambios, pueden realizarse los siguientes:

- Ampliar el espacio asignado en la base de datos para el registro de transacciones o para los datos.
- Reducir el espacio asignado en la base de datos para el registro de transacciones y los datos.
- Agregar o quitar archivos de datos y de registro de transacciones.
- Crear grupos de archivos.
- Cambiar el grupo de archivos predeterminado.
- Cambiar la configuración de la base de datos.
- Desconectar bases de datos.
- Adjuntar nuevas bases de datos y separar bases de datos no utilizadas.
- Cambiar el nombre de la base de datos.
- Cambiar el propietario de la base de datos.

CAPÍTULO VI

6. DESARROLLO DEL SISTEMA BAJO PLATAFORMA WINDOWS.

6.1. INTRODUCCION.

La gente ha estado creando y usando tarjetas de identificación desde finales del siglo pasado. a finales de los ochenta, el método más común de producir una credencial era conocido como el compuesto o enmicado. Este simplemente involucraba el tomar la foto de la persona, cortarla, pegarla a un pedazo de papel que contenía el nombre de la persona, su número y algunos otros datos, y finalmente, era enmicada con calor todo el conjunto.

A pesar de que la inversión inicial para un sistema de enmicado era relativamente baja, el tiempo, trabajo y el costo individual por tarjeta eran altos. Además, estas tarjetas eran fácilmente falsificadas. Como resultado, un nuevo método llamado impresión digital empezó a dispararse a principios de los noventa.

Hoy en día, la Impresión Digital se ha convertido en la elección tecnológica de la mayoría de las organizaciones que requieren de tarjetas de identificación y por una buena razón. Los sistemas de Identificación digital están mejorando cada día, son muy fáciles de usar, y ofrecen una gama de beneficios.

Por esta razón, el eficientar la captura de datos es un reto, en la actualidad, existen tecnologías de identificación automática tales como: Códigos de barras, bandas magnéticas, y reconocimiento óptico de caracteres, son algunas tecnologías para captura que podemos utilizar

Existen diversas soluciones prácticas para sus necesidades de control de acceso con sistemas caracterizados por la innovación tecnológica y respaldados por los más reconocidos fabricantes.

Los sistemas de control de acceso ofrecen a sus usuarios una manera sencilla de implementar el acceso a lugares críticos, de personas, o de bienes e información.

El campo de aplicación de los sistemas de control de acceso están directamente relacionadas con la gran mayoría de las actividades humanas, dentro de las cuales tenemos.

- Seguridad.
- Localización.
- Evacuación.
- Tiempo y asistencia.
- Accesos críticos en áreas peligrosas.
- Control de pacientes.
- Bibliotecas.
- Comedores
- Centros deportivos, clubes.
- Control de acceso en la operación de:
- Maquinaria industrial.
- Equipo de procesamiento de datos.
- Equipo de comunicaciones.
- Equipo médico.
- Equipo de transporte.
- Equipo delicado.
- Control de procesos.
- Acceso a información electrónica.
- Acceso a archivos e imágenes.

Dentro de las múltiples aplicaciones que se pueden dar con la implantación de controles de acceso, ésta investigación pretende estudiar las áreas de tiempo, asistencia y control de procesos para monitorear y asignar el uso de computadoras en el centro de cómputo de la PUCESA. por medio de carnets que incluyen códigos de barra de los estudiantes y personal docente y administrativo de la escuela de sistemas de la institución.

6.2. TEMA

"Diseñar y construir un sistema que permita monitorear y asignar el uso de computadoras en el centro de cómputo de la PUCESA utilizando dispositivos de lectura de códigos de barras."

6.3. TITULO

"SISTEMA DE MONITOREO Y CONTROL DE ACCESO AL CENTRO DE COMPUTO DE LA PUCESA".

6.4. FUENTE.

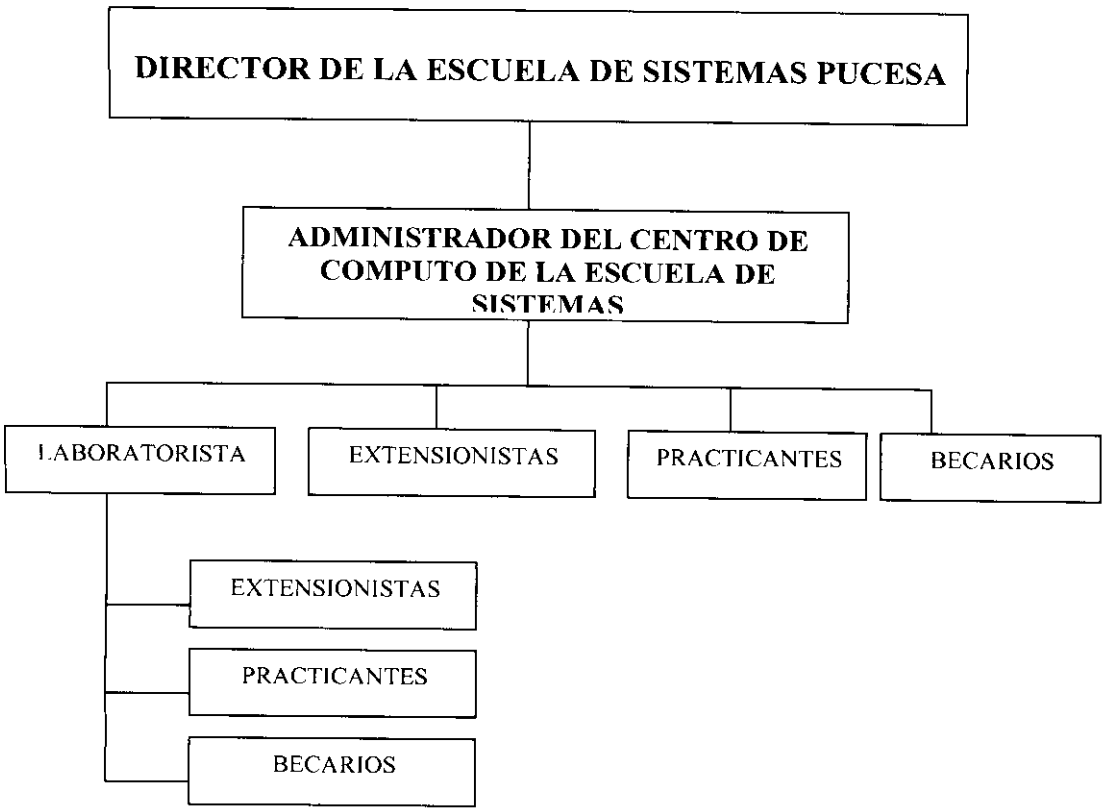
- Centro de Cómputo de la Pontificia Universidad Católica del Ecuador Sede Ambato.
- Administrador del Centro de Cómputo.
- Laboratorista del Centro de Cómputo.
- Extensionistas del Centro de Computo

6.5. OBJETIVO GENERAL DEL PROYECTO.

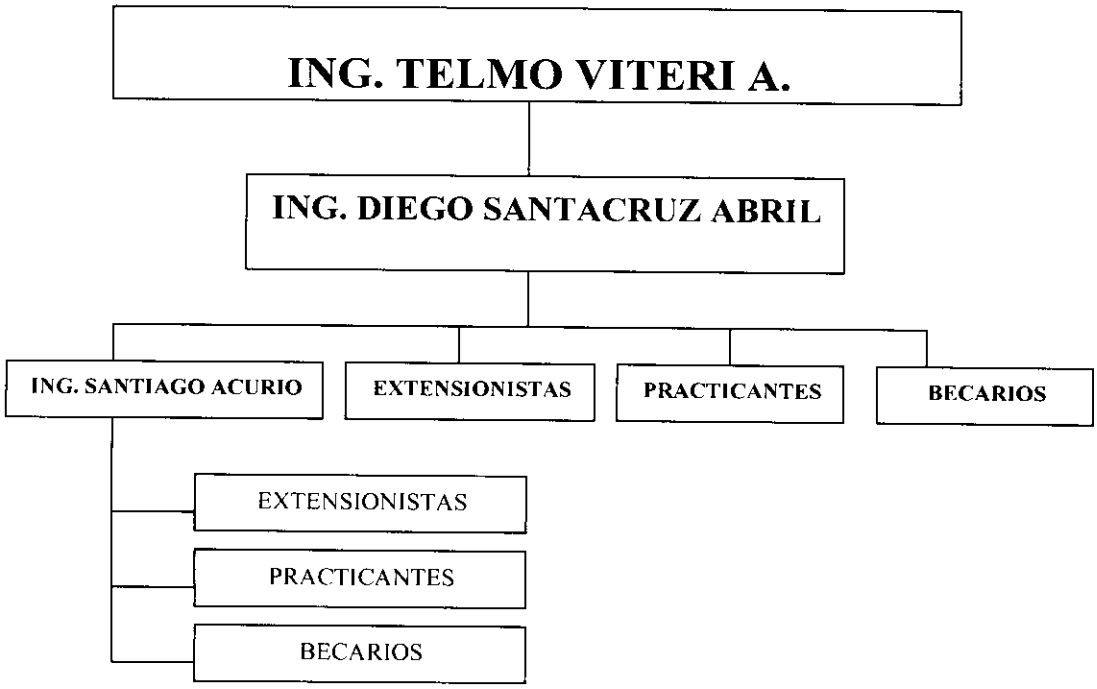
Diseñar y construir un sistema que permita monitorear y asignar el uso de computadores en el centro de cómputo de la PUCESA a través de dispositivos de lectura de códigos de barras.

6.6. ORGANIZACION DEL CENTRO DE COMPUTO DE LA PUCESA.

6.6.1. ORGANIGRAMA ESTRUCTURAL DEL CENTRO DE COMPUTO



6.6.2. ORGANIGRAMA POSICIONAL DEL CENTRO DE COMPUTO



La disposición del Centro de Cómputo de la PUCESA esta conformado de la siguiente manera:

- Administrador del Centro de cómputo.
- Laboratoristas.
- Ayudantes.

6.6.2.1. Administrador del Centro de cómputo

6.6.2.1.1. MISIÓN DEL PUESTO

Ser un ente de apoyo en todo lo concerniente a la Administración efectiva de los bienes y servicios que posee el centro de cómputo, y servir de apoyo técnico en asuntos informáticos a las demás instancias Universitarias.

6.6.2.1.2. ACTIVIDADES

Se presenta a continuación una tabla con las principales actividades que debe realizar el Administrador del Centro de cómputo en forma regular.

DIARIAS

- Revisar el buen funcionamiento del servidor Web y de Correo electrónico.
- Revisar el funcionamiento del servicio de Internet y reportar inconvenientes inmediatamente a proveedores.
- Revisar cumplimiento de actividades del personal bajo su cargo.
- Autorizar el uso de la sala de conferencias PC/TV.
- Coordinar y controlar el desarrollo de Proyectos en ejecución.
- Revisar estado de equipos en laboratorios y oficinas en base a reportes de Laboratoristas y ayudantes.
- Revisar estado del área física de laboratorios y oficinas y coordinar su buen estado con personal de aseo.
- Monitorear actividades en equipos de laboratorios.
- Prestar el apoyo necesario a la planta de Docentes en caso de ser necesario para efectos de didáctica.
- Autorizar el ingreso a los Laboratorios fuera del horario de clases.

SEMANAL

- Notificar al Director de la Escuela cualquier cambio en la ubicación de equipos y posibles fallas detectadas para tomar medidas correctivas.
- Notificar cualquier incumplimiento de actividades de subordinados.
- Revisión continua de cómo se desarrollan los procesos internos para planificar cambios y nuevas estrategias.

- Asistir técnicamente al resto de Departamentos Administrativos y otras Escuelas de la PUCESA.

- Verificar que las hojas de control se encuentren completas, así también como los equipos y el software del centro.

MENSUAL

- Medir el rendimiento del canal de Internet para establecer estadísticas de uso.

- Obtener respaldos de la información más importante generada en el servidor y otros datos ubicados en puestos estratégicos.

- Obtener copias del Software más utilizado por la Planta de Docentes para facilitar su reinstalación en los Laboratorios.

- Crear nuevas cuentas de clientes RAS.

- Notificar cuentas caducadas a clientes del RAS.

- Suspender cuentas caducadas y no facturadas.

- Reactivar cuentas suspendidas y facturadas.

SEMESTRAL

- Planificar y coordinar proyectos de actualización de servicios principalmente de información en Página Web de la Pucesa con la participación de practicantes, Extensionistas o becarios.

- Asistir a reuniones o sesiones de trabajo según lo indique el Director de la Escuela.

- Entregar tarjetas de Internet para su utilización fuera del horario de clases en laboratorios.

- Crear cuentas de usuarios de correo electrónico.

- Buscar nuevas versiones de sistemas operativos que puedan prestar mejor servicio en el servidor WEB y de correo electrónico.

- Coordinar la ejecución del mantenimiento preventivo y correctivo.

6.6.2.1.3. RESPONSABILIDADES

Equipos y maquinaria:	Propios del Centro de Cómputo
Herramientas:	Para el mantenimiento y revisión de equipos
Material:	De oficina
Producto:	ninguno
Dinero:	ninguno
Información o doc. confidenciales:	Manuales técnicos y documentos administrativos
Toma de decisiones:	Asignación de laboratorios, sala de conferencias, préstamo
De equipos y software,	Asignación de tareas y proyectos internos, Instalación de Software.
No. de personas que supervisa:	1 Laboratorista y 2 Practicantes (variable).

6.6.2.2. LABORATORISTAS

6.6.2.2.1. MISIÓN DEL PUESTO

Asegurar el acceso y uso adecuado de los laboratorios del centro de cómputo y su mejor conservación, así como brindar asistencia técnica a los usuarios de los laboratorios.

6.6.2.2.2. ACTIVIDADES

DIARIA

- Permitir el acceso puntualmente a los laboratorios según lo indique el horario de clases establecido.
- Instalar Software para uso de Docentes previa autorización del Administrador.
- Cuidar el buen estado de computadoras y equipos de comunicación.

- Entregar y archivar las hojas de control interno, ingreso a laboratorios, préstamos, y otros controles implementados.
- Permitir el acceso a Internet.
- Restringir el acceso a laboratorios y oficina de personal no autorizado por el Jefe inmediato.

SEMANAL

- Comunicar cambios en la ubicación de equipos por alguna causa especial o mal funcionamiento de los mismos al Administrador.
- Coordinar actividades con practicantes y ayudantes a su cargo en los proyectos encargados.
- Revisión de equipos de cada laboratorio, chequear el buen funcionamiento de teclados, mouse, monitores, discos duros (scandisk), unidades de disquetes.

MENSUAL

- Presentar informe sobre el estado de equipos de laboratorios.
- Presentar informe sobre el estado actual de proyectos encargados y su progreso.

SEMESTRAL

- Participar en tareas de mantenimiento preventivo y correctivo de equipos de laboratorios y de otros departamentos de la Pucesa.

6.6.2.2.3. RESPONSABILIDADES

Equipos y maquinaria:	Propios del Centro de Cómputo
Herramientas:	Para la revisión y mantenimiento de equipos
Material:	De oficina
Producto:	ninguno

Dinero:	ninguno
Información o doc. confidenciales:	Manuales técnicos y Administrativos del Centro
Toma de decisiones:	
No de personas que supervisa:	Un Becario, Practicante, o Extensionista asignado

6.6.2.3. AYUDANTES

6.6.2.3.1. MISIÓN DEL PUESTO

Apoyar en la realización eficaz y eficiente de cada proyecto planificado con participación de estudiantes de la Escuela de Sistemas en calidad de Extensionistas, practicantes o becarios.

6.6.2.3.2. ACTIVIDADES

Las actividades serán definidas por la persona que sea designada como su tutor o guía para efectos de realización de cualquier proyecto o funciones encargadas, se definirán así mismo su frecuencia y responsabilidades adicionales.

6.6.2.3.3. RESPONSABILIDADES

Equipos y maquinaria:	Que sean asignados para el proyecto de trabajo
Herramientas:	Las asignadas por el Jefe inmediato superior
Material:	ninguno
Producto:	ninguno
Dinero:	ninguno
Información o doc. confidenciales:	Manuales y documentos internos del Centro de Cómputo.
Toma de decisiones:	n/a
No de personas que supervisa:	n/a

6.6.3. FUNCIONES

6.6.3.1. FUNCIONES DEL ADMINISTRADOR

- Planificar, organizar y controlar las actividades inherentes a la Administración de los bienes y servicios propios del centro de cómputo de la Escuela de Sistemas
- Planificar y coordinar proyectos informáticos en los que puedan participar el personal a su cargo.
- Instalar y configurar Servidores Windows y LINUX SPARC.
- Administrar los servicios de valor agregado a través del Server Sun Microsystems e250.
- Instalar y Configurar redes para laboratorios y habilitar el servicio de Internet
- Definir y controlar las tareas que cumplirá el personal bajo su cargo.
- Desarrollar el Plan de mantenimiento preventivo para equipos de laboratorios y oficina del centro.
- Revisión y mejoramiento continuo de los procesos internos del centro de cómputo, así como la sugerencia de nuevos servicios que fomenten la participación de docentes y estudiantes.
- Otras que señale el jefe inmediato superior según lo establecido en el reglamento interno de la Pucesa.

6.6.3.2. FUNCIONES DE LABORATORISTAS

- Preparar computadoras para uso en docencia.
- Instalar y desinstalar software en equipos de laboratorios (controlando licencias).
- Controlar el uso adecuado de los equipos de computación durante actividades de docencia.
- Controlar el acceso a los laboratorios y llevar un registro permanente de su utilización.

- Controlar que los bienes ubicados en los laboratorios permanezcan en su sitio y no sean destruidos o mal utilizados por estudiantes.

- Colaborar en procesos de mantenimiento preventivo, y correctivo de los equipos de computación de los laboratorios y otros que indique el Administrador, tomando las medidas de seguridad correspondientes.

- Otras que señale el Administrador del centro de cómputo.

6.6.3.3. FUNCIONES DE AYUDANTES

- Colaborar en la planificación y ejecución de proyectos que realice el Centro de cómputo.

- Colaborar en el control del buen estado de todos los equipos de las oficinas y de laboratorios del Centro.

- Restringir el acceso a personal no autorizado a las oficinas del Centro de cómputo

- Las que se añadan en relación a las actividades encargadas

- Otras que señale el Administrador del Centro de cómputo.

6.7. FUNCIONAMIENTO DEL MÉTODO ACTUAL.

Actualmente todos los procedimientos se los realiza en forma manual.

El procedimiento para el registro de usuarios es el siguiente:

El usuario solicita a una de las personas que conforman el Centro de Cómputo la asignación de una computadora.

Si existe disponibilidad de computadoras, se le solicita el carnet del estudiante en caso de ser alumno o la cédula de identidad para otro tipo de usuarios.

El usuario ingresa al laboratorio asignado y ocupa una de las computadoras a disposición.

Cuando el usuario a finalizado sus tareas se acerca a retirar documento de ingreso.

6.8. DESCRIPCION DEL PROBLEMA

Actualmente los procedimientos de control de acceso al centro de cómputo de la Escuela de Ingeniería de Sistemas de la PUCESA son realizados íntegramente de una forma manual.

Debido a la carencia de no disponer de un sistema automatizado para registrar usuarios, asignar computadoras, monitorear disponibilidad de computadoras en la red, etc. Han surgido problemas tales como:

- No se dispone de información relevante y confiable que permita al Administrador del Centro de Cómputo planificar de manera óptima la utilización de los diferentes laboratorios.
- Impotencia para establecer responsabilidades en casos tales como : pérdida o destrucción de Periféricos.
- Mal aprovechamiento de los recursos, etc.

La automatización del control de acceso al centro de cómputo permitirá vigilar de forma permanente y eficaz el uso de cada uno de las computadoras; además se tendrá a disposición un registro de los usuarios. Con la implementación del sistema se pretende solucionar los inconvenientes antes mencionados.

6.9. OBJETIVOS ESPECIFICOS DEL PROYECTO.

- Implementar una nueva tecnología para el control de acceso al centro de cómputo de la PUCESA
- Verificar los datos de los usuarios a través de los lectores de barras.
- Controlar las rutinas de activación y desactivación de los computadores mediante software.
- Autorizar automáticamente el uso de una computadora que se encuentre disponible.
- Monitorear de forma periódica el estado en que se encuentran las estaciones de trabajo en la red.

6.10. ALCANCE DEL SISTEMA.

Cuatro son los temas generales que se desarrollaran en esta disertación:

1. Estudio de la estructura y funcionamiento de los lectores de códigos de barras.
2. El sistema operativo Windows para la implementación del sistema de control de acceso al centro de cómputo de la PUCESA.
3. MS SQL Server como motor de base de datos.
4. Analizar el uso de Delphi o Visual Basic como plataforma para el desarrollo del sistema de control de acceso al centro de cómputo de la PUCESA.

De manera específica la disertación abarcará las siguientes áreas:

- Características de los lectores de barras.
- Aplicaciones de los códigos de barras.
- Simbologías de los códigos de barras.
- Estudio de llamadas y ejecución de procesos remotos Cliente / Servidor.
- Implementación de funciones del sistema operativo que permitan determinar el estado de una estación de trabajo.
- Comunicación entre los equipos monitoreados.
- Desarrollo de un software para el control de acceso al centro de cómputo de la PUCESA. Que se extiende exclusivamente hasta:
- Utilización de un lector de código de barras para la verificación automática de los datos del usuario.
- Tareas de asignación de computadoras que se encuentren disponibles.
- Controlar las tareas de activación y desactivación de cada estación de trabajo a través de software.

- Monitoreo de forma periódica del estado en que se encuentran las computadoras en la red.

- Generación de la base de datos a partir del carnet estudiantil proporcionado por la universidad. (Actualmente el carnet estudiantil posee un código de barras de numero de cédula).

- Suspender el uso de la estación de trabajo a un usuario al cual no ha sido asignado.

- Procesos manuales que no pueden ser sustituidos por las características de los equipos existentes tendrán que ser realizadas por el Administrador del Centro de Cómputo.

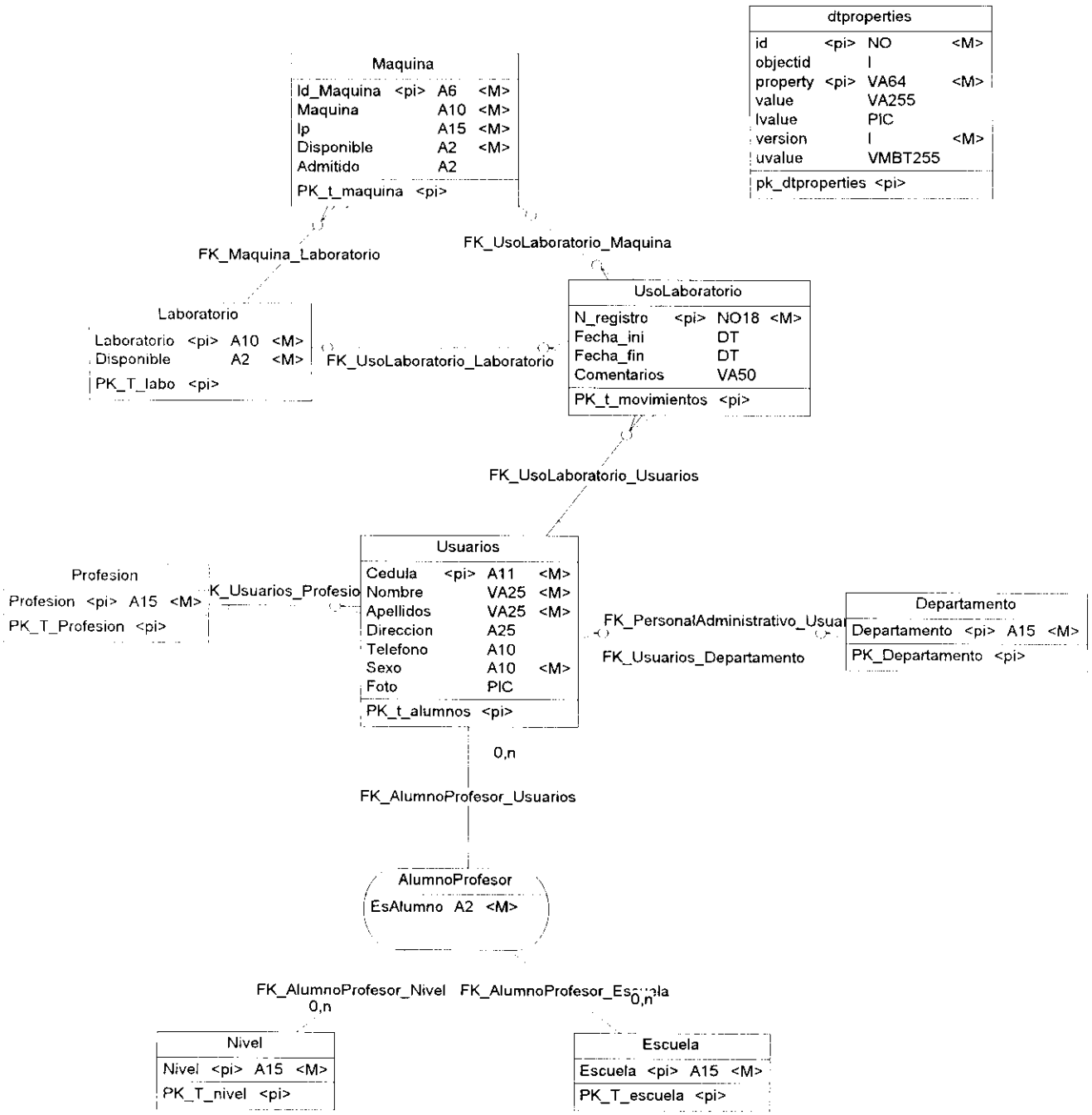
- Control del tiempo de caducidad del carnet estudiantil.

- Flexibilidad del sistema para acceder al Centro de Cómputo de la PUCESA mediante el número de cédula en forma temporal en caso de pérdida del carnet.

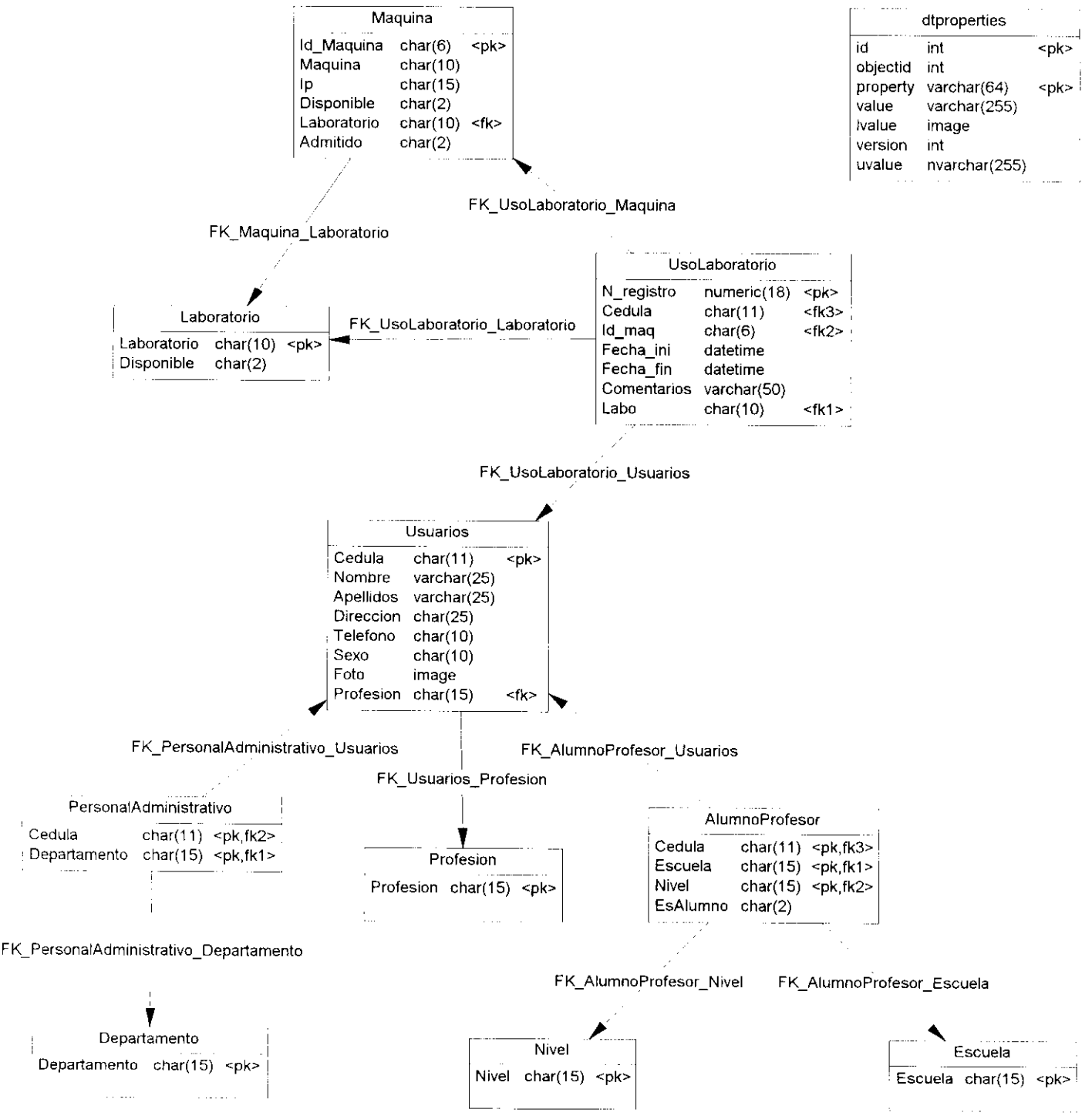
- Reportes básicos.

6.11. DIAGRAMAS DE FLUJO DE DATOS.

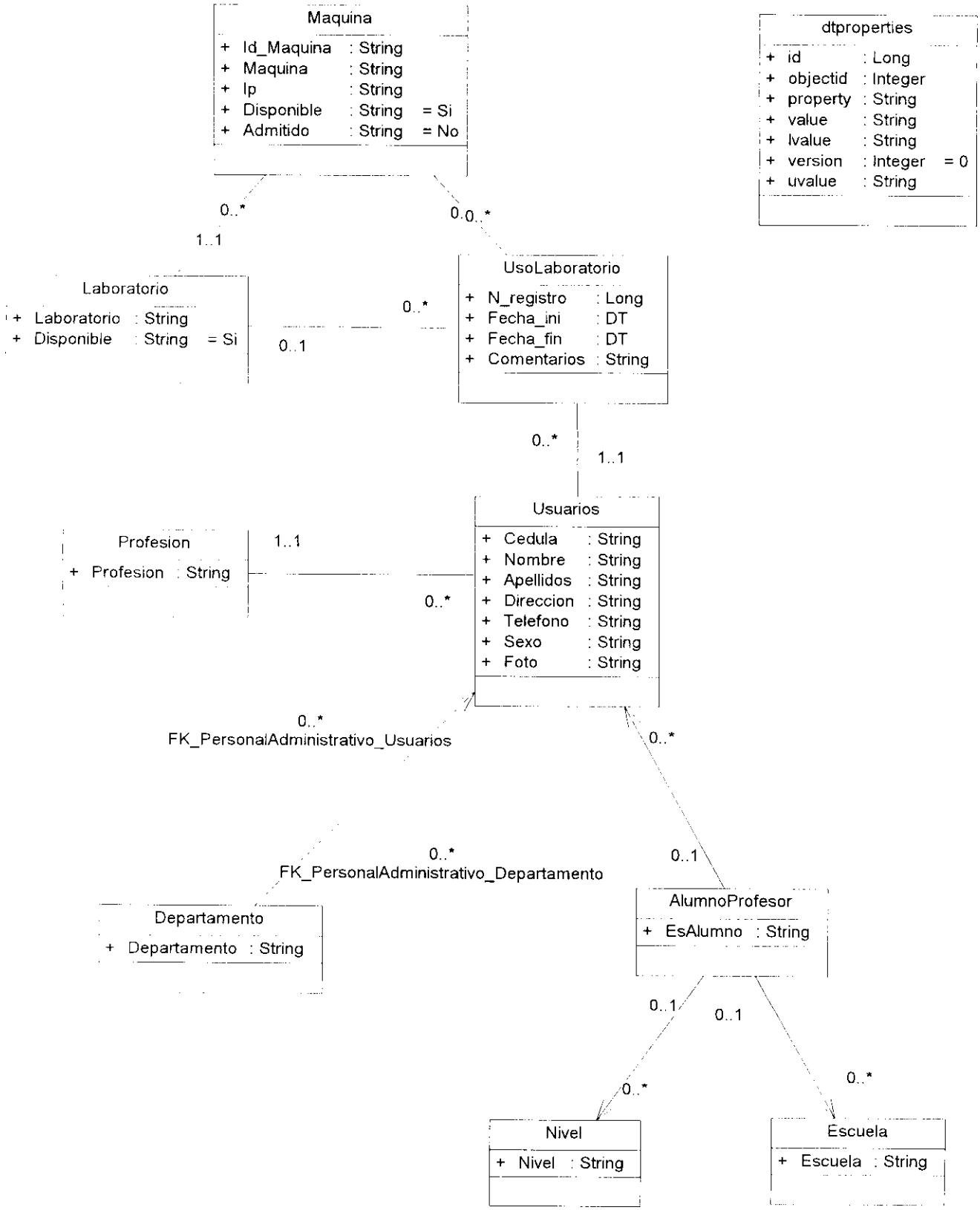
6.11.1. MODELO FÍSICO DE DATOS.



6.11.2. MODELO CONCEPTUAL DE DATOS.



6.11.3. MODELO ORIENTADO A OBJETOS



6.12. ADAPTACION DEL DISPOSITIVO DE ENTRADA DE DATOS.

6.13. CONCLUSIONES Y RECOMENDACIONES.

6.13.1 CONCLUSIONES.

Al finalizar el desarrollo de esta disertación de tesis, hemos podido extraer las siguientes conclusiones.

- La aplicación desarrollada cumple con todos los objetivos planteados en el proyecto de disertación propuesto.
- La utilización de dispositivos automáticos de identificación permitirá al administrador del centro de cómputo de la PUCESA mantener el control de los usuarios sin necesidad de digitar códigos.
- El software desarrollado mantendrá el control adecuado del centro de cómputo, permitiendo al administrador establecer responsabilidades en caso de mal uso de los PC's.
- El software podrá monitorear automáticamente la red para evitar el acceso de personas no autorizadas al centro de cómputo de la PUCESA, si este el caso, nuestra aplicación será capaz de apagar automáticamente el PC utilizado sin permiso.
- El administrador del centro de cómputo podrá bloquear o apagar el PC desde el servidor sin la necesidad de acercarse hacia la computadora.
- Un usuario puede utilizar uno o más PC's a la vez si es necesario podrá también utilizar todo el laboratorio, si está disponible.
- Nuestra aplicación es inteligente para asignar o liberar PC's, cuando un usuario pasa por el lector de códigos de barras el carnet de identificación.
- El software desarrollado presenta gráficamente los PC's disponibles y en uso diferenciados por la imagen que muestra para cada caso, clasificados de acuerdo al laboratorio al cual pertenecen.

- Durante la etapa de pruebas hemos visto la necesidad de utilizar hilos (Threads) para ciertos procesos que el software utiliza que fueron lentos debido al tráfico de red, ya que utiliza direcciones IP para monitorear los PC's.

- Debido a las limitaciones encontradas en el centro de cómputo, hemos visto la necesidad de utilizar Microsoft Access como una alternativa de Base de datos, ya que brinda flexibilidad y portabilidad entre equipos. Al mismo tiempo optimiza los recursos disponibles en el computador.

- La utilización de Delphi 6.0 como el lenguaje de programación nos ha facilitado la elaboración de nuestro proyecto debido a que la herramienta proporciona una serie de objetos que hemos empleado para cumplir con los objetivos planteados.

- La utilización del lenguaje de programación en su estructura orientada a objetos, ya facilita la creación de módulos independientes pero que se interrelacionan entre sí y evitar la redundancia de código.

- El software diseñado es capaz de correr bajo cualquier versión de Windows incluyendo Windows XP.

6.13.2. RECOMENDACIONES.

Las recomendaciones que se presentan a continuación se deben considerar como resultado de un análisis adecuado del funcionamiento del centro de cómputo de la PUCESA.

- Es necesario que el PC este conectado al la red interna del centro de cómputo.

- Evitar escribir descripciones de los PC en Windows XP.

- Impedir que los usuarios manipulen el registro de Windows para evitar que se borren las palabras claves utilizadas por el software.

- Implementar políticas para asignar adecuadas direcciones IP a los PC's como también sus nombres.

INDICE DE ABREVIATURAS

OCR	Optical Character Recognition o reconocimiento óptico de caracteres.
UPC	Universal Product Code.
ROADS	Running Out of Address Space.
CIDR	Enrutamiento Inter – Dominio Sin Clases (Classless Inter – Domain Routing)
ISBN	International Standard Book Numbering.
ITF	código entrelazado 2 de 5 (Interleaved Two of Five).
CAD/CAM	Diseño asistido por computadora.
LCD	Visualizador de cristal líquido .
UPC / EAN	Universal Product Code / European Article Numbering.
PDF417	Portable Data File (Archivo de Información Portátil, PDF)
ASCII	Código Estandar Estadounidense para el Intercambio de Información (American Standard Code for Information Interchange).
LED	(Diodo emisor de luz)
VEA	Vigilancia Electrónica de Artículos
OCIA .	conexión a teclado de PC/POS .
CPU	Central Process Unit
PCB	Process Control Block.
PEP	Palabra de estado del programa.
SPT	Shortest Processing Time
API	Application Program Interface.
TCP/IP	Transmission Control Protocol / Internet Protocol.
UDP	Protocolo de datagramas de usuario.

IP	Internet Protocol.
DLL	Dynamic Lynk Library.
DNS	Domain Name System.
ICMP	Protocolo de mensajes de control de Internet.
OSI	Modelo de Referencia para la Intercomunicación de Sistemas Abiertos.
ISO	Organización Internacional de Estandarización. International Organization for Standarization
DLC	Data Link Control.
HDLC	High Level Data Link Control.
DSE	Intercambio por conmutación de datos.
DTE	Equipo Terminal de Datos.
ARPANET	Red de la Agencia de Proyectos de Investigación Avanzada.
TCP	Protocolo de Control de Transmisión. Transmission Control Protocol.
IP	Internet Protocol.
PPP	Point to Point Protocol
CNLP	Protocolo de Redes No Orientado a Conexión .
ACK	Acepción positiva al módulo TCP remitente.
SQL	Lenguaje de Consulta Estructurado. Structured Query Language.
DBMS	Database Management System.
ANSI	American National Standards Institute.

OLAP	Online Analytical Processing.
SAI	Sistemas de alimentación ininterrumpida .
IPC	Interprocess Communications.

GLOSARIO DE TERMINOS

Cifrado	Es un método para mantener la confidencialidad de información reservada.
Thread	Es un camino de ejecución dentro de un proceso.
socket	Es una representación abstracta del extremo (endpoint) en un proceso de comunicación.
MTU	La capacidad máxima de transferencia de datos de una red física se le llama MTU.
RS-232.	Puerto Serial RS-232.
Datagrama	Paquetes independientes dentro de una organización sin conexión.
Telemática	Nombre colectivo de los servicios de información públicos para uso de casas y oficinas.
Buffers	Almacén temporal de información en tránsito.
Enrutar	Es el proceso de selección de un camino para el envío de paquetes.

BIBLIOGRAFIA

- PRESSMAN Roger S (1998). Ingeniería del Software. Un enfoque práctico. Cuarta edición. Madrid. Ed. McGraw-Hill.
- INMON W. H. (1994). Developing Client/Server Aplication. Ed. Wiley.
- BERSON A. (1996). Client/Server Architecture. Segunda edición. Ed. McGraw-Hill.
- RAMOS Patricio, SOLIS Roberto (2000). Estudio de Dispositivos de Lectura de Datos no tradicionales y su implementación en el desarrollo de Sistemas bajo Plataforma Windows y D.O.S.
- Indice: Seguridad "Libros en pantalla de SQL Server".
- CENTRO DE COMPUTO DE LA PUCESA. Organización y Funcionamiento del Centro de Cómputo.

Direcciones de Internet

http://bari.ufps.edu.co/

http://tcp.com/

http://ip.com/

http://osi.com/

http://bari.ufps.edu.co/personal/150802A/comunicacion.htm

http://www.idsys.es/index2.html

http://www.azerty.com.mx/barcode/lectores.shtml

http://elpuntodeventa.com/catalogo_scanners.html

http://www.codigodebarras.com/

http://www.agfa.es/sistgraf/productos/

http://www.pscnet.com/html/vs_spanish.htm

http://www.q3.nu/trucomania/ftesp.html

http://www.lobocom.es/~claudio/sql.html

http://www.sockets.com/

http://www.somser.com/2000vb/Winsock.htm

http://usuarios.tripod.es/vteforte/ip.html

http://usuarios.tripod.es/vteforte/sockets.htm

http://www.somser.com/2000vb/Winsock.htm

Índice de Ilustraciones

Índice de Tablas