UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

"DISEÑO DE UNA TARJETA INTELIGENTE PARA EL CONTROL DE EXISTENCIAS DURANTE EL EQUIPAMIENTO DE EVENTOS SOCIALES"

TESIS

PARA OBTENER EL TÍTULO DE

INGENIERA EN COMPUTACIÓN

PRESENTA:

JESSICA IVETTE MONSALVO OBREGÓN

Y

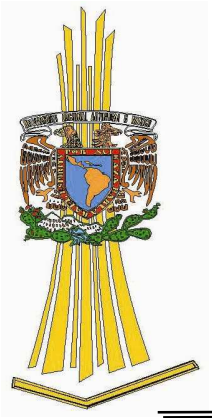
PARA OBTENER EL TÍTULO DE

INGENIERO MECÁNICO ELECTRICISTA

ÁREA: INGENIERÍA ELÉCTRICA ELECTRÓNICA

PRESENTA

CRISTIAN ALEJANDRO RIVERA SUÁREZ



ING. FRANCISCO RAUL ORTIZ GONZALEZ

SAN JUAN DE ARAGÓN, EDO. DE MÉXICO, 2010.

Primeramente quiero dar gracias a mi Dios, él cual ha estado a cada instante de mi vida ayudándome por su infinito amor, protección y gracia, que siempre ha estado caminando junto a mí.

A mi mami preciosa, te amo con todo mi ser, eres la máximo en mi vida, yo no sería y no hubiera llegado hasta donde estoy sino fuera por ti, me has dado más de lo que pudiese necesitar, muchas gracias por todo.

A mi hermana Erika, a quien agradezco el haber estado conmigo apoyándome y escuchado cuando más lo he necesitado, gracias hermanita te quiero mucho.

A Pablo que ha sido de gran apoyo y que he podido contar con él en cada momento de mi vida, gracias por todo.

A Cristian porque hemos logrado esta meta, has tenido paciencia, dedicación y esfuerzo en todo el tiempo que hemos estado juntos, muchas gracias por todo, eres una persona muy especial para mí, te quiero mucho.

A mi familia: mis tíos, primos y sobrinos que en todo momento he tenido su incondicional apoyo, muchas gracias a cada uno de ustedes, que ha colocado un granito de arena.

A mi asesor, el cual nos oriento en cada momento y por el interés mostrado en nuestro trabajo, gracias.

Jessica Ivette Monsalvo Obregón

A Dios por su respaldo permanente a lo largo de toda mi existencia, por su amor y su misericordia que me han rodeado, sin ti simplemente nada sería posible gracias Jesucristo.

A mis padres: A mi Papá por su apoyo incondicional a lo largo de todos mis estudios eres parte fundamental en mi vida, a mi Mamá gracias por brindar tu vida entera para que pudiera salir adelante y cumplir mis metas gracias por guiar mis pasos con tus consejos y oraciones, sin ustedes no lo hubiera logrado.

A mis Hermanos: Gracias por estar conmigo y brindarme su apoyo cuando ha sido necesario, son mis mejores amigos, los quiero mucho.

A Jessica: Por todo este tiempo juntos, por tu esfuerzo, dedicación, apoyo y comprensión, eres una persona muy especial para mí, sin ti este trabajo no sería posible gracias te quiero mucho.

A mi Familia: Gracias a todos mis abuelitos, tíos y primos no hace falta decir nombres cada uno de ustedes saben lo fundamental que han sido sus vidas para mí.

A mi asesor: Por creer en mí y dedicar su tiempo para dirigir este trabajo de tesis.

Cristian Alejandro Rivera Suárez

CONTENIDO

Pág.

INTRODUCCIÓN

La ingeniería ha estado ligada a la humanidad desde el principio de la misma, desde que el ser humano comenzó a pensar en cómo utilizar objetos para facilitar sus labores y suplir sus diversas necesidades, en la actualidad la finalidad ingenieril sigue siendo la misma

El diseño de software en conjunto con las nuevas tecnologías, han facilitado la tarea del sector empresarial para satisfacer sus necesidades y el ámbito de los eventos sociales no ha escapado a los beneficios de esta alianza tecnológica.

Un banquete es una fiesta pública, que se completa con platos principales y postres. Por regla general sirve a un propósito festivo o de celebración, tal y como puede ser: la celebración de un evento familiar, una boda, una ceremonia, etc. Uno de los principales objetivos del banquete es la reunión en torno a una mesa y el deleite común de los sentidos, no es de extrañar que la importancia de un banquete se llegue a medir por el número de invitados. La persona que convoca, organiza y corre con los gastos del banquete se denomina anfitrión, esta es a la que se suele dedicar y agasajar en algunas de las celebraciones culinarias realizadas sobre la mesa.

Existen muchos tipos de banquetes de forma muy extendida por todas las culturas de la tierra, entre ellos el banquete de bodas o banquete nupcial en el que se congregan familiares y amigos de ambos cónyuges celebrando el matrimonio. En algunas otras ocasiones el anfitrión quiere mostrar el paso de una etapa, como es el banquete de la fiesta de quince años en México, donde una chica que pasa esa edad ofrece a familiares una comida para celebrarlo.

En algunas culturas el banquete se celebra en pequeñas comunidades de vecinos en torno a una barbacoa, como puede ser el khorovats tan celebrado por los armenios o las barbacoas turcas. Algunos banquetes son puramente familiares, en las que participan tan sólo los integrantes de la familia como puede ser: la cena de Acción de Gracias y la cena de Navidad. Otros son puramente religiosos y su simbología trae a la memoria de los asistentes ciertos recuerdos del pasado, como puede ser el plato del seder y la conmemoración de la Pascua Judía, otros banquetes también religiosos pueden ser el banquete de bautizo, primera comunión, presentación entre otros.

En la antigua Grecia se celebraban diversos banquetes con objeto puramente festivo como pueden ser las solterías y los simposios que han quedado reflejados en la literatura ("El banquete" de los diálogos de Platón) y los grabados de la época (por ejemplo en las representaciones de la necrópolis de Tarquinia).

Los banquetes de la Edad Media y el Renacimiento europeo se celebraban en las clases más pudientes y en la mayoría de los casos servían a fines prácticos al poder, en ellos se cerraban pactos, alianzas y convenios, reconciliaciones entre enemigos, etc.

En la actualidad la razón de los banquetes sigue siendo en algunos aspectos la misma, incluso existen lugares específicos para realizar dichas festividades como pueden ser salones de fiestas o jardines de eventos sociales, así mismo también existen empresas que operan dichos lugares u ofrecen el servicio de banquetes a domicilio.

La empresa dedicada a los banquetes debe contar con una gran variedad de servicios que a su vez forman el producto final como lo es una fiesta, ya sea desde un desayuno familiar, una cena empresarial, una boda o unos XV años, la empresa debe ofrecer los siguientes servicios:

- Un lugar para realizar el banquete.- esto en caso de que el servicio no sea a domicilio, además de un horario de inicio y un horario de término del evento.

- Equipo y mobiliario.- es la renta de las mesas en las que se lleva a cabo el banquete, ya que estas pueden ser redondas, rectangulares, cuadradas, o tipo lounge; además, las mesas van acompañadas de sillas de plástico, acojinadas, tipo Tiffany, tipo Versace o taburetes estos en el caso de las mesas tipo lounge; finalmente otro tipo de equipo a rentar en el caso de un lugar al aire libre como puede ser un jardín, son las carpas que cumplen el servicio de resguardar a los invitados de los aspectos climáticos, es decir, realizan la función de un salón.

- Mantelería.- en este aspecto del servicio se ofrece la renta de manteles y estos pueden ser lisos, de encaje y brocados entre otros, además de la variedad en colores; también se ofrece la renta del cubremantel que al igual que los manteles son lisos, de encaje y brocados, así como su variedad de colores; otro ámbito de la mantelería es la funda para las sillas estas regularmente son de color blanco y solo se usan en las sillas de plástico y acojinadas; como toque final de la mantelería se tiene la renta de la banda con la que se realizan la gran variedad de moños para adornar las sillas.

- Vajilla, cristalería y cubertería.- dichos aspectos son vitales en la realización del banquete, ya que son los elementos en los cuales se sirven y con los que se degustan los platillos ofrecidos en el evento, cabe mencionar que la vajilla y la cubertería puede variar según los tiempos manejados en el banquete, estos momentos pueden variar desde dos hasta cinco tiempos en algunos casos modificando de esta manera la presentación de la vajilla y cubertería, finalmente la cristalería que va de la mano con la variedad de bebidas a ofrecer en el evento.

- Menú.- sin duda un aspecto primordial en la realización del banquete, ya que es el momento clímax de la fiesta, este es el momento en el que tanto los invitados como el anfitrión se reúnen alrededor de la mesa para degustar los platillos a ofrecer en honor de la celebración, dichos platillos conforman el menú, el cual puede variar desde una taquiza con platillos regionales, hasta la comida más sofisticada del arte culinario, cabe mencionar que los platillos se acompañan de diferentes bebidas según sea la ocasión y las reglas del buen comer.

- Personal de servicio.- son las personas encargadas de atender a los comensales en todas sus necesidades en cuanto al banquete se refiere, este servicio es lo que comúnmente conocemos como servicio de meseros.

- Variedad.- este punto también ha llegado a ser primordial, ya que desde la antigüedad en los banquetes existieron músicos amenizando los eventos, en la actualidad no ha cambiado mucho este aspecto, solo que la música varía dependiendo la región y el gusto musical de los anfitriones, la variedad puede ir desde un grupo de música versátil hasta un cuarteto de cuerdas y piano o a últimos a tomado mucho auge en este ámbito el ocupar Disc Jockey para amenizar el evento.

PREFACIO

Con el paso del tiempo la tecnología ha avanzado con una rapidez muy significativa, condescendiendo la computadora como una herramienta de apoyo para las áreas eléctrica, electrónica, computacional, etc. Utilizando esta herramienta los ingenieros han realizado diseños o desarrollado soluciones tecnológicas a necesidades sociales, industriales o económicas.

La ingeniería aplicada en el área computacional y electrónica en el diseño de software, así como también la comunicación con dispositivos de entrada y salida, como lo es el uso de tarjetas inteligentes, ha sido de gran utilidad en la satisfacción de las diferentes necesidades del ámbito empresarial.

El presente trabajo esta orientado a los salones de eventos sociales y banquetes a través de un software de aplicación, facilitando el control de existencias durante el equipamiento del personal para la realización de los banquetes mediante el uso de la tarjeta inteligente.

A continuación se describirán los aspectos principales en los que se encuentran comprendidos los capítulos de esta tesis.

Capítulo I.- Se documenta la historia de los Sistemas Operativos con los avances más notables de los últimos años, así como la clasificación de los lenguajes de programación que se emplean en una computadora, además de mencionar los sistemas que ayudan a realizar tareas específicas como son los sistemas de aplicación.

Capítulo II.- Por la magnitud de datos que se manipulan en las diferentes necesidades de las empresas, el uso de las bases de datos ha tomado cada vez mayor importancia ya que han facilitado la satisfacción de dichas necesidades, es por ello que en este capítulo se menciona la evolución de las bases de datos, así como sus principales características y la relación directa en el diseño de programas.

Capitulo III.- Se presentan los principales elementos de programación y estructuras, que se deben de tomar en cuenta para el desarrollo y diseño del software, en este caso se utilizó el lenguaje de programación Java y se hace mención de sus diferentes aplicaciones.

Capítulo IV.- Debido a que la computadora ha sido una principal herramienta para el desarrollo de esta tesis, este capítulo esta enfocado en los diferentes componentes del sistema que hacen funcionar una computadora como son el software y hardware, presentando los dispositivos de entrada y salida de una manera estructurada mostrando sus características.

Capítulo V.- Este capítulo documenta una breve historia de las tarjetas inteligentes, además de presentar los diferentes tipos de tarjetas, así como su uso, estándares, proceso de fabricación, características físicas, funcionamiento y programación, haciendo mención de sus múltiples aplicaciones y beneficios.

Capítulo VI.- En base al software diseñado para esta tesis, se ha elaborado un manual de usuario, mostrando gráficamente el empleo del sistema de aplicación.

CAPÍTULO I.

SOFTWARE DEL SISTEMA OPERATIVO

I. 1. GENERALIDADES.

Cualquier computadora es parte de un sistema computacional, dicho sistema consta de cuatro partes fundamentales que son: Hardware, Software, Datos y Usuarios.

FIGURA I. 1.
Computadora actual.

Donde:

El término hardware se refiere a cualquier parte de la computadora que se pueda tocar. El hardware consiste de dispositivos electromecánicos interconectados para controlar la operación, por medio de la entrada y salida de la computadora.

El término software se refiere al conjunto de instrucciones por medio de pulsos eléctricos que le dicen al hardware que debe hacer. Estos conjuntos de instrucciones también se conocen como programas y cada uno tiene un propósito específico. Aunque la serie de programas disponibles es vasta y variada, la mayor parte del software cae en dos categorías principales: software de sistemas y software de aplicación. El software de sistema es llamado software de sistema operativo, este le dice a la computadora como usar sus propios componentes. El software de aplicación, es un sistema el cual le dice a la computadora como realizar tareas especificas para el usuario, como procesamiento de palabras o dibujo y en este caso para el control de existencias durante el equipamiento de eventos sociales.

Los datos se refieren a los elementos crudos que la computadora puede manipular. Los datos pueden consistir en letras, números, sonidos o imágenes. No importa que clase de datos sean introducidos en una computadora, ésta los convierte en números. En consecuencia, los datos computarizados son digitales. Dentro de la computadora, los datos son organizados en archivos, estos archivos son un conjunto de datos o instrucciones de programa a los que se les da un nombre específico con la respectiva extensión, esto con la finalidad de diferenciarlos. Un archivo contiene datos que el usuario puede actualizar, modificar y dar de alta o baja, según la información que el usuario requiera.

Por último el usuario es la persona que interactúa con la computadora.

I. 2. LOS SISTEMAS OPERATIVOS.

Un sistema operativo (SO) es un programa de control maestro de la computadora. El SO proporciona las herramientas (comandos) que le permiten interactuar con la PC. Cuando se emite un comando, el SO lo traduce en un código que la computadora interpreta. El SO también asegura que el resultado de sus acciones sean desplegadas en pantalla o impresos.

Cuando se activa una computadora, ésta pasa por varios pasos antes de poder usarla. El primer paso es la autocomprobación. La computadora identifica los dispositivos conectados a ella, identifica la cantidad de memoria disponible y también comprueba rápidamente si la memoria esta funcionando en forma apropiada. Esta rutina es iniciada por una parte del software de sistema localizado en la ROM (Read Only Memory: Memoria de Solo Lectura), un chip que contiene breves instrucciones (en lenguaje ensamblador) permanentes para lograr que la computadora comience a operar.

A continuación la computadora busca un sistema operativo, ya sea en la unidad de CD (Compact Disk: Disco Compacto) y luego en la unidad de disco duro, esto dependerá de la configuración. El sistema operativo le dice a la computadora cómo interactuar con el usuario y cómo usar dispositivos como las unidades de disco, teclado y monitor. Cuando encuentra el sistema operativo, la computadora lo carga en la memoria. Debido a que el sistema operativo es necesario para controlar las funciones básicas de la computadora, éste continua mientras la computadora esté encendida.

Después de que la computadora encuentra y ejecuta el sistema operativo, está lista para aceptar comandos de un dispositivo de entrada, por lo general el teclado, un ratón o actualmente touch screen (Monitores de pantalla plana sensible al tacto utilizando rayos infrarojos) . En este punto, el usuario puede emitir comandos a la computadora.

I. 2. A. EVOLUCIÓN DE LOS SISTEMAS OPERATIVOS.

La lógica o software de las computadors se ha complicado conforme han evolucionado éstos. Las primeras computadoras, a partir de 1944, en que Aiken construyó el Mark-I, sólo podían programarse en lenguaje máquina, y puede decirse que el sistema operativo aún no existía.

FIGURA I. 2.
Computadora Mark-I.

Jhon von Neumann, fue quién dio el primer paso en la informática y en las relaciones hombre-máquina con el concepto de programa almacenado, que consiste en archivar en la computadora un conjunto de instrucciones máquina para posteriormente ejecutarlas. La aparición del Assembler, lenguaje mnemotécnico-simbólico, constituyó un gran avance sobre la primitiva programación en

código máquina. La forma de explotar los sistemas ha ido evolucionando con el tiempo; los más sencillos funcionaban con monoprogramación. Monoprogramación o monoejecución es el sistema de explotación en el que se ejecuta solamente un programa cada vez y no comienza la ejecución de otro hasta terminar con el anterior.

En 1948, con Noam Chomsky, surge la teoría de las gramáticas generativas transformacionales, que es la base de los traductores de lenguajes. En 1955, comenzó el desarrollo de los lenguajes de alto nivel, y al mismo tiempo se empezó a dividir el trabajo entre personas: operadores y programadores. Las funciones del operador tenían más directamente que ver con la administración y control de los recursos del sistema operativo y la carga de trabajos, y las de los programadores con la codificación de los programas; actualmente, al estar superado el concepto de monoprogramación, es frecuente mas común la familiarización con los conceptos de multiprogramación, tiempo compartido, multiproceso y tiempo real.

Las tendencias de futuro prevén el desarrollo de sistemas operativos con potencialidad de:

- Explotación del proceso en paralelo y de forma concurrente.
- Integración mediante informática corporativa del tratamiento en distintos sistemas operativos de microcomputadoras, minicomputadoras y mainframes, a través de interfaces gráficas de usuario y la arquitectura cliente-servidor.

Hay tres grandes familias de computadoras con sus respectivos sistemas operativos: las grandes computadoras o mainframes, que llevan incorporados varios microprocesadores funcionando a la vez (en paralelo). Las computadoras de tipo medio (minis), que tienen de 1 a 4 procesadores muy versátiles funcionando en paralelo. Y por último las computadoras personales, que tienen un procesador que efectúa todas las operaciones.

I. 2. B. ORIGEN DEL SISTEMA OPERATIVO.

Las computadoras de 1960, no poseían programas que ayudaran a gestionar su funcionamiento. A esta organización se le llamó Sistema Monolítico, facilitando el trabajo a los usuarios. Tampoco existían lenguajes de alto nivel que permitieran al programador sortear las limitaciones de la computadora a la que programaba. Si el usuario quería leer un archivo, debía escribir el mismo las rutinas para poner en marcha el motor de la unidad de cintas, buscar la información y comprobar que no existían errores. Además, todo ello debía realizarse en el código binario del procesador con el que estuviera trabajando.

FIGURA I. 3.
Computadora de 1960.

Conforme se mejoraban los circuitos electrónicos, se fueron añadiendo facilidades vía grupos de programas que permitían organizar la ejecución de los procesos, así como el almacenamiento de los datos.

A finales de los años 60 del siglo XX, dentro del mundo universitario en los Estados Unidos de América (EUA), se diseñó el primer sistema operativo moderno: Multics. Este sistema permitía un uso racional de los recursos de la computadora, automatizando el sistema de archivos, la gestión de procesos y permitiendo el trabajo de "múltiples" usuarios en una misma computadora.

Posteriormente se crearon otros sistemas operativos, pero el más importante fue UNIX. Este, era descendiente directo de Multics, y fue implementado por primera vez para una computadora PDP-7 en 1969. Una característica que los distinguió desde el principio es que no dependía de la computadora en la que funciona. Sólo una pequeñísima parte de su código estaba en ensamblador, y el resto en lenguaje C, por lo que se extendió muy rápidamente a distintos equipos de cómputo.

Estos sistemas operativos, que funcionaban en las antiguas y costosísimas computadoras, debían permitir el uso de varias personas simultáneamente para aprovechar al máximo el rendimiento de la computadora. Se establecía una estructura en la que la computadora era el centro y, alrededor suyo, se establecían múltiples terminales sin capacidad de proceso. A estos sistemas operativos se les llamó de tiempo compartido.

I. 2. C. ORGANIZACIÓN DEL SISTEMA OPERATIVO.

La organización del sistema operativo por lo general consta de 4 niveles. En el primer nivel y más bajo, se encuentra el núcleo [kernel], que es el que tiene contacto directo con los circuitos electrónicos. En el segundo se encuentran las rutinas que implementan los servicios que ofrece el sistema operativo, como el manejo de lo discos, el monitor, teclado y la gestión de los procesos. En el tercero se encuentran el gestor de la memoria y de archivos. Por último, en el cuarto, están los procesos que permiten la comunicación del usuario con el sistema operativo: el caparazón [shell] y las órdenes propias del sistema operativo. La comunicación sólo es posible entre los niveles inmediatamente superior e inferior.

Nivel 4	Caparazón, Interfase Gráfica, Procesos del Sistema.
Nivel 3	Gestores de Memoria y Archivos.
Nivel 2	Controladores de Dispositivos y Gestor de Tareas.
Nivel 1	Núcleo.
Nivel 0	Circuitos Electrónicos.

TABLA I. 1.
Estructura Básica de un Sistema Operativo.

Los sistemas operativos son los programas más caros y difíciles de desarrollar. Esto es así porque se les exige un funcionamiento sin errores, ya que depende de ellos la ejecución de todas las aplicaciones.

I. 2. D. MS DOS.

Este sistema operativo fue patentado por las empresas Microsoft Corporation e IBM, utilizándose dos versiones similares (una de cada empresa) llamadas MS DOS (Microsoft Disk Operating System: Sistema Operativo de Disco de Microsoft) y PC-DOS Personal Computer Disk Operating System: Computadora Personal con Operativo de Disco de Microsoft).

El MS DOS al cumplir las condiciones de monousuario y monotarea, hacen que el procesador este en cada momento dedicado en exclusividad a la ejecución de un proceso, por lo que la planificación del procesador es simple y se dedica al único proceso activo que pueda existir en un momento dado.

Al nombre de MS DOS le acompañan unos números que indican la versión. Si la diferencia entre dos versiones es la última cifra representa pequeñas variaciones. Sin embargo, si es en la primera cifra representa cambios fundamentales. Las versiones comenzaron a numerar por 1.0 en agosto de 1981. En mayo de 1982 se lanzó la versión 1.1 con soporte de disquetes de dos caras. La versión 2.0 se creó en marzo de 1983 para gestionar el PC XT (Personal Computer Extended Technology: Computadora Personal con Tecnología Extendida), que incorporaba disco duro de 10 Mb, siendo su principal novedad el soporte de estructura de DOS que ya no estén disponibles.

En agosto de 1984, con la aparición de las computadoras del tipo AT, que empleaban un procesador 80286, que funcionaba a 8 Mhz de velocidad y tenían soporte de disquetes de 5 1 ¼" de alta densidad (HD 1.2 Mb), MS DOS evolucionó hacia la versión 3.0; esta versión podía ser instalada en computadoras más antiguas, pero no se podía realizar la operación a la inversa.

La versión 3.2 se lanzó en diciembre de 1985, para admitir unidades de disquete de 3 ½¨ (DD 720 Kb y HD 1.44 Mb). La versión 3.3 se lanzó en abril de 1987, con posibilidades de crear múltiples particiones en discos duro.

La versión 4.0 apareció en noviembre de 1988, gestionando discos duros de particiones de más de 32 Mb hasta 512 Mb. Además disponía de una nueva interfaz gráfica y soporte de memoria expandida, esta versión permitía además el empleo de la memoria expandida de la computadora. MS DOS es un sistema patentado por Microsoft Corporation para computadoras personales PC's.

La versión 5.0 se lanzó en junio de 1991, y proporciona drivers para gestionar ampliaciones de memoria y se incorpora un editor de pantalla y un shell (interfaz usada para interactuar con el núcleo de un sistema operativo) bastante potente, además de poder instalarse independientemente de la versión anterior de sistema operativo.

La versión 6.0 se lanzó en abril de 1993 y como contenía abundantes errores fue sustituida el mismo año por la versión 6.2. Las mejoras de la versión 6.0 incluían: herramientas de compresión de discos, antivirus, programas de copias de seguridad por menú, desfragmentador de disco y otras utilidades.

I. 2. E. SISTEMAS OPERATIVOS DE WINDOWS.

El MS DOS soporta una nueva capa de software de sistema que permite a los usuarios interactuar con varios programas a la vez y permite el empleo de una forma gráfica, este software es el Windows.

I. 2. E. a. Windows 1.0.

Windows 1.0 es la primera versión independiente de Microsoft; fue lanzada el 20 de noviembre de 1985, y empleó 55 programadores. Windows 1.0 tenía muy pocas funcionalidades y consiguió poca popularidad, probablemente porque las aplicaciones que traía no tenían la potencia necesaria para usuarios de negocios. De todas maneras, permitía el uso de mouse, menús desplegables y gráficos de pantalla e impresora independiente del dispositivo.

Originalmente se llamaría Interfaz Manager, pero Rowland Hanson, uno de los jefes de marketing de Microsoft, convenció a la compañía de que el nombre "Windows" sería más atrayente para el mercado. Windows 1.0 no fue un sistema operativo en sí mismo, sino que fue una extensión del MS DOS, sistema operativo que ya estaba siendo desarrollado por Microsoft.

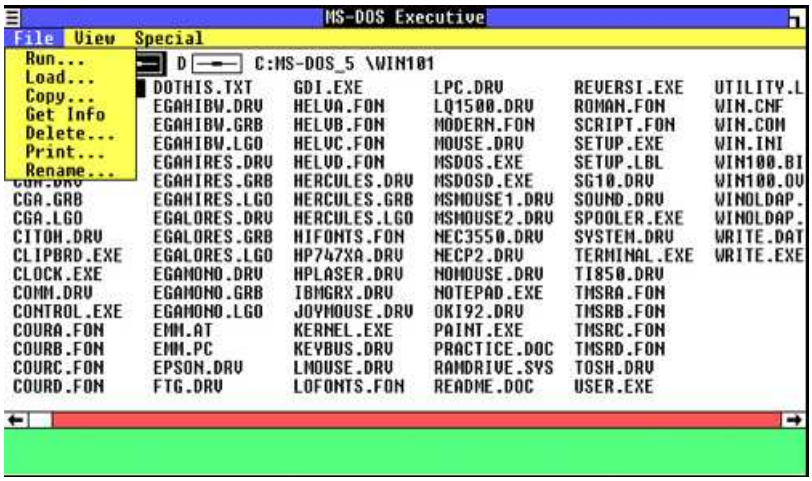


FIGURA I. 4.
Pantalla de Microsoft Windows 1.0.

I. 2. E. b. Windows 2.0.

Windows 2.0, fue lanzado el 9 de diciembre de 1987, y alcanzó un poco más de popularidad que su predecesor (Windows 1.0). Su éxito se debió a la incorporación de nuevas aplicaciones como Excel y Word. También incorporó íconos, ventanas traslapadas y archivos PIF (Program Information File: Archivo de Información del Programa) para DOS. Las versiones 2.0x de Windows utilizaban memoria en modo real, lo que limitaba a 1 Mb la memoria RAM.

Luego, fueron lanzadas dos versiones: Windows/286 2.1 y Windows/386 2.1. Como en las versiones anteriores, Windows/286 2.1 utilizaba la memoria en modo real, pero fue la primera versión en soportar HMA (High Memory Area: Area de Memoria Alta).

Todas las aplicaciones Windows y DOS en ese momento eran en modo real, ejecutándose sobre el kernel protegido empleando el modo 8086 virtual, que era nuevo con el procesador 80386.

FIGURA I. 5.
Computadora 8086.

FIGURA I. 6.
Pantalla de Microsoft Windows 2.03.

I. 2. E. c. Windows 3.0.

Windows/386 permitía múltiples máquinas virtuales DOS con multitarea o, en otras palabras, ejecutar más de una aplicación DOS al mismo tiempo.

La versión del sistema operativo Microsoft Windows 3.0 lanzado en 1990, pertenece a los llamados Windows 3.x. Gracias a la introducción de la memoria virtual, Windows 3.0 permitía un mejor uso de la multitarea.

También los gráficos mejorados disponibles en PC's gracias a las tarjetas de video VGA (Video Graphics Array: Matriz gráfica de video. Tarjeta gráfica de computadoras PC y compatibles, evolución de la MCGA, que permitía trabajar también a 16 colores con 640x480 puntos) y los modos protegidos, realzados que permitían a las aplicaciones de Windows emplear más memoria de una mejor manera que en DOS.

Windows 3.0 podía ejecutarse en modos real, estándar o 386 mejorado (o extendido), y era compatible con los procesadores de Intel desde el 8086/8088 al 80286 y 80386.

Esta versión intentaba autodetectar en qué modo ejecutarse, de todas maneras podía ser forzado a ejecutarse en un modo específico empleando determinados parámetros. Esta fue la primera versión de Windows en ejecutar programas en modo protegido.

Gracias a la compatibilidad con las versiones anteriores, las aplicaciones también podían ser compiladas en entornos de 16 bits, sin siquiera usar todas las capacidades 32 bits de las CPU (Unit Procces Central: Unidad de Proceso Central) 386.

Otras características incorporadas fueron: el soporte para red, el Administrador de Programas y el Administrador de Archivos, y el soporte para más de 16 colores.

Varios meses después, fue lanzada una versión multimedia limitada: Windows 3.0 con Multimedia Extensión 1.0. Esta fue adjuntada con el primer kit multimedia con tarjeta de sonido y CD ROM.

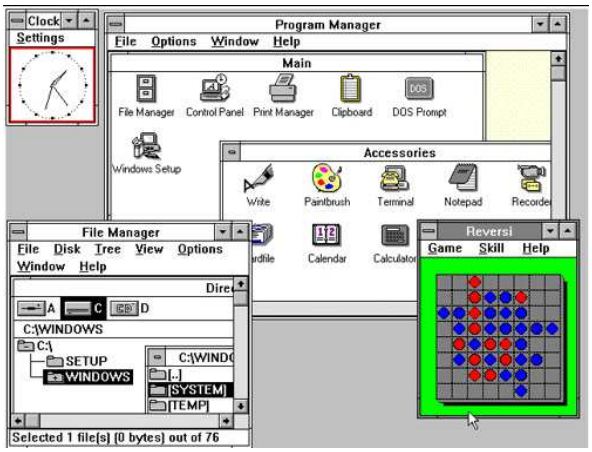


FIGURA I. 7.
Pantalla de Microsoft Windows 3.0.

I. 2. E. d. Windows 3.1.

Windows 3.0 vendió alrededor de 10 millones de copias dos años antes de lanzar la versión 3.1, convirtiéndose así en la mayor fuente de ingresos de Microsoft.

En respuesta al inminente lanzamiento del OS/2 v2.0, Microsoft desarrolló Windows 3.1 en 1992, que incluyó múltiples mejoras al Windows 3.0, además de corrección de errores, soporte multimedia y dejó de soportar el modo real. Sólo se ejecutaba en un procesador 80286 o superior.
Otras características de Windows 3.1, es el soporte para fuentes True Type, OLE, soporte de API de multimedia y red.

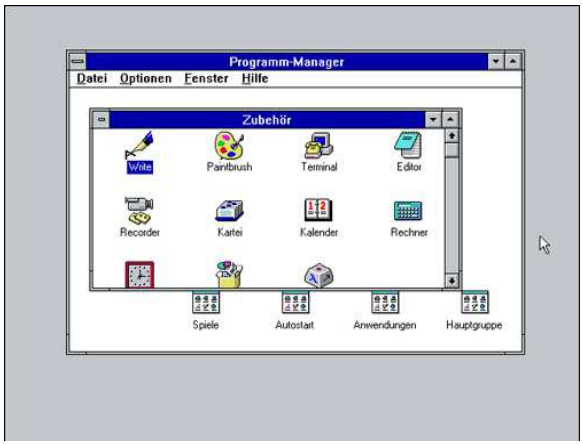


FIGURA I. 8.
Pantalla de Microsoft Windows 3.1.

I. 2. E. e. Windows 3.11.

Luego fue lanzado Windows 3.11, que incluía todos los parches y actualizaciones para Windows 3.1. Última versión de Windows como aplicación sujeta a DOS pues no era considerado como un verdadero sistema operativo. La versión que le sucedió fue Windows 95.

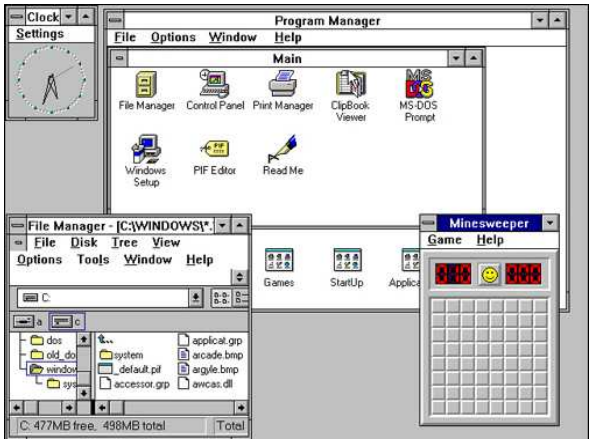


FIGURA I. 9.
Pantalla de Microsoft Windows 3.11.

I. 2. E. f. Windows 95.

Primera versión de Windows realmente como sistema operativo. Fue lanzado el 24 de agosto de 1995, con una gran aceptación por parte del público.

Desarrollador:	Microsoft.
Familia:	Windows.
Tipo:	Software no libre.
Núcleo:	Win16.
Nombre clave:	Chicago.
Lanzamiento:	24 de agosto de 1995.
Ultima versión:	OEM SR 2.5 1997.
Estado actual:	Sin soporte.

TABLA I. 2.
Sistema Operativo Windows 95.

Reemplazó al MS DOS como sistema operativo y a Windows 3.x como entorno gráfico empleando 224 caracteres alfanuméricos como máximo. Una de las novedades más visibles fue la posibilidad de escribir nombres largos para archivos; recordemos que antes los nombres de archivos sólo podían ocupar 8 caracteres para el nombre y 3 para la extensión.

Otras grandes novedades, que se mantienen hasta las versiones modernas, fueron la incorporación de la barra de tareas y el botón de inicio. También fue el primer sistema operativo en utilizar tecnología Plug and Play.

Su versión OSR2 (Original Equipment Manufacturer Service Release: Servicio Público del Fabricante de Equipos Originales) salida en 1996, incorporó la versión 2.0 de Internet Explorer.

Desde la versión OSR2 incorporó el sistema de archivos FAT32, anteriormente sólo soportaba FAT16. También incorporó la versión 3.0 de Internet Explorer. Su sucesor fue Windows 98.

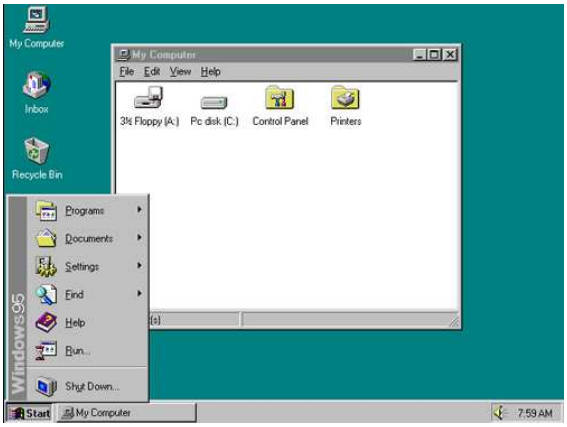


FIGURA I. 10.
Pantalla de Microsoft Windows 95.

I. 2. E. g. Windows 98.

Se destacó por tener muchas mejoras con respecto a su predecesor como mejoramiento general del rendimiento y más estabilidad. Fue el primero en traer la herramienta Windows Update.

Windows 98 constituyó también el primer paso hacia un sistema operativo autooptimizable.

Desarrollador:	Microsoft.
Familia:	Windows.
Tipo:	Software no libre.
Núcleo:	Win32.
Nombre clave:	Memphis.
Ultima versión:	25 de junio de 1998.
Estado actual:	Sin soporte.

TABLA I. 3.
Sistema Operativo Windows 98.

Su primera versión fue lanzada al mercado el 25 de julio de 1998, con un sistema de archivos FAT (File Allocation Table: Tabla de Asignación de Ficheros) 32 que permitía mayor capacidad de almacenamiento en el disco duro que los 2 GB (Múltiplo del byte: un gigabyte son 1.024 MegaBytes, cerca de 1.000 millones de bytes) máximos permitidos en Windows 95.

En 1999, fue lanzado Windows 98 Segunda Edición, no como actualización sino como un sistema operativo completamente nuevo.

FIGURA I. 11.
Pantalla de Microsoft Windows 98.

I. 2. E. h. Windows 2000.

Windows 2000 (también conocido como Win2K) es un interrumpible, gráfico y de negocios orientados hacia el sistema operativo que fue diseñado para trabajar con cualquiera de monoprocesador simétrico o multi-procesador de 32-bit Intel x86. Es parte de la línea de Microsoft Windows NT de sistemas operativos y fue puesto en libertad el 17 de febrero de 2000. Agregó confiabilidad, facilidad de uso, compatibilidad con internet y soporte con la computación móvil. Fue sucedido por el de Windows XP en octubre de 2001. Windows 2000 está clasificado como un híbrido del núcleo del sistema operativo.

Desarrollador:	Microsoft.
Modelo de desarrollo:	Software no libre.
Núcleo:	NT.
Tipo de núcleo:	Híbrido.
Licencia:	Microsoft CLUF (EULA).
Última versión estable:	Service Pack 4 / Junio 2003.
Estado actual:	Período de soporte extendido hasta el 13 de julio de 2010.
Sitio Web:	Windows 2000.

TABLA I. 4.
Sistema Operativo Windows 2000.

Existen cuatro variantes de Windows 2000 que son: Professional, Server, Advanced Server y Datacenter Server. Estas dos últimas variantes no son más que ampliaciones del propio Windows 2000 Server. Windows 2000 Server es el sistema operativo de servidor principal para empresas de todos los tamaños y es ideal para ejecutar sus servidores de red o los servidores de archivo, impresión, intranet o de aplicaciones.

Windows 2000 Advanced Server, el sucesor de Windows NT Server 4.0 Enterprise Edition, es un sistema operativo de servidor más eficaz, ideal para ejecutar aplicaciones de línea de negocios, soluciones de comercio electrónico y punto.com. Ofrece una estructura completa de clústeres para alta disponibilidad y escalabilidad y admite el multiprocesamiento simétrico de ocho vías SMP (Symmetric multiprocessing: Multiprocesamiento simétrico.) además de memoria hasta de 8 GB con la Extensión de dirección física de Intel PAE (Physical Address Extension: Extensión de Dirección Física)

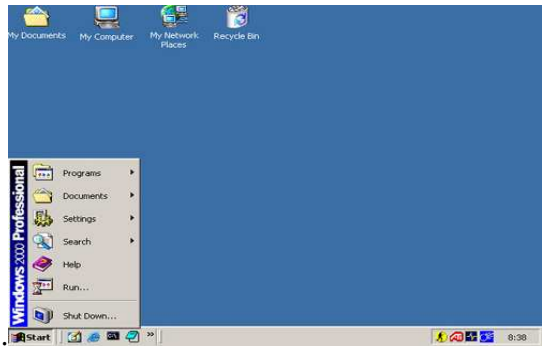


FIGURA I. 12.
Pantalla de Microsoft Windows 2000.

I. 2. E. i. Windows ME (2000).

Versión del sistema operativo Windows de Microsoft lanzado el 14 de septiembre de 2000. Diseñado par el hogar. Windows ME ofrecía mejoras en la música, videos y redes caseras.

El sistema operativo está basado en la familia Windows 9x. Con respecto a su antecesor (Windows 98) sólo incluyó algunas actualizaciones menores como Internet Explorer 5.5, Windows Media Player 7, Movie Maker, etc.

A partir de este sistema operativo MS DOS perdió la importancia que tenía para el sistema especialmente cuando de resolver problemas se trataba.

Desarrollador:	Microsoft.
Familia:	Windows.
Tipo:	Software no libre.
Núcleo:	Win32.
Nombre clave:	Millennium.
Lanzamiento:	14 de septiembre de 2000.
Ultima versión:	4.10.2222 A.
Estado actual:	Sin soporte.

TABLA I. 5.
Sistema Operativo Windows ME (Millenium).

Windows ME (Millenium) fue altamente cuestionado por su inestabilidad y problemas de seguridad. Microsoft no sacó una segunda versión del ME y ya no utiliza la familia de Windows 9x para desarrollar sistemas. El próximo que lanzó fue Windows XP (eXPerience: experiencia) pero basado en la familia NT (New Technology: Nueva Tecnología).

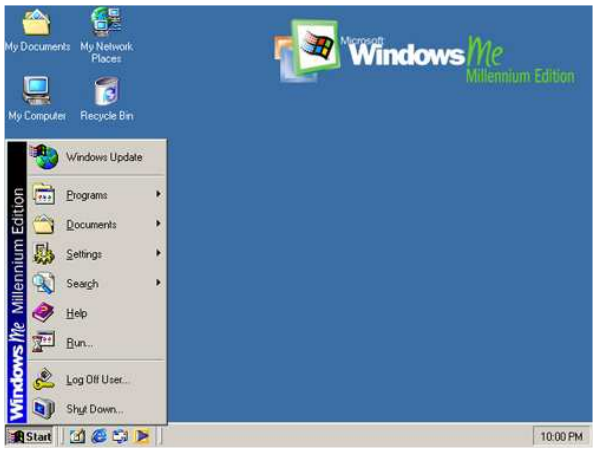


FIGURA I. 13.
Pantalla de Microsoft Windows ME.

I. 2. E. j. Windows XP.

Versión del sistema operativo Windows desarrollado por Microsoft lanzado el 25 de octubre de 2001. XP proviene de "experiencia".

Desarrollador:	Microsoft.
Familia:	Windows NT.
Tipo:	Software no libre.
Núcleo:	NT.
Nombre clave:	Whistler.
Lanzamiento:	25 de octubre de 2001.
Ultima versión:	5.1 Agosto 2004.
Estado actual:	Con soporte.

TABLA I. 7.
Sistema Operativo Windows XP.

Windows XP posee gran cantidad de ediciones como Home, Professional, IA-64, Media Center (2002, 2003, 2004 & 2005), Tablet PC, Starter, Embedded, etc. que tienen diferentes mercados.

El sistema operativo marcó el fin de la familia de los Windows 9x, pues está basado en la familia de Windows NT.

Windows XP dio un salto importante en la estabilidad y eficiencia, no así en su seguridad, pues tuvieron que salir grandes parches desde el principio. Los parches más grandes fueron el Service Pack 1, el Service Pack 2 y el Service Pack 3 que traerá características tomadas de Windows Vista.

FIGURA I. 14.
Pantalla de Microsoft Windows XP.

I. 2. E. k. Windows 2003.

Windows Server 2003 es un sistema operativo desarrollado por Microsoft, lanzado el 24 de abril de 2003 como sucesor de Windows 2000 Server. Es más escalable y posee un mejor rendimiento que su predecesor. Sus ediciones son: Standard, Enterprise, Datacenter, Web y XP Pro x64.

En términos generales, Windows Server 2003 se podría considerar como un Windows XP modificado, no con menos funciones, sino que estas están deshabilitadas por defecto para obtener un mejor rendimiento y para centrar el uso de procesador en las características de servidor, por ejemplo, la interfaz gráfica denominada Luna de Windows XP viene desactivada y viene con la interfaz clásica de Windows. Sin embargo, es posible volver a activar las características mediante comandos services.msc.

Su sistema de archivos es NTFS, encriptación, compresión de archivos, carpetas y unidades completas. Permite montar dispositivos de almacenamiento sobre sistemas de archivos de otros dispositivos al estilo Linux, Active Directory, etc.

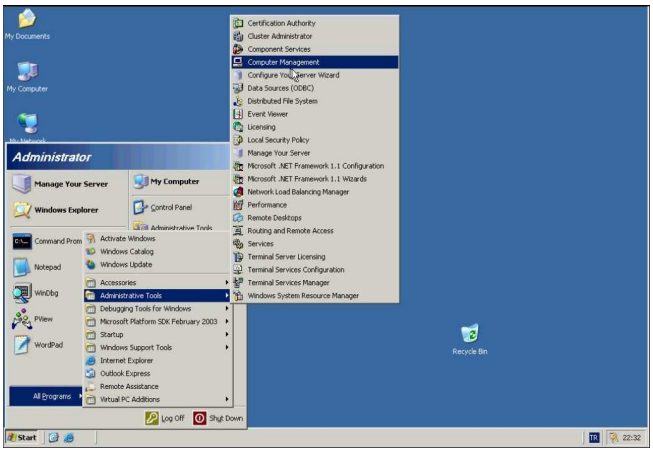


FIGURA I. 15.
Pantalla de Microsoft Windows Server 2003.

I. 2. E. l. Windows Vista.

Versión del sistema operativo Windows desarrollado por Microsoft sucesor de Windows XP. Está basado en la familia de los sistemas Windows NT.

Fue lanzado el 30 de noviembre de 2006, sólo en su versión empresarial, para el usuario final la versión se lanzó el 30 de enero de 2007 conjuntamente con Microsoft Office 2007 y Exchange Server 2007.

Desarrollador:	Microsoft.
Familia:	Windows NT.
Tipo:	Software no libre.
Núcleo:	NT.
Nombre clave:	Longhorn.
Lanzamiento:	Enero de 2007.
Ultima versión:	Versión empresarial.
Estado actual:	Con soporte.

TABLA I. 8
Sistema Operativo Windows Vista

Viene en las siguientes ediciones: Windows Vista Home Basic, Vista Home Premium, Vista Business, Vista Enterprise y Vista Ultimate.

Windows Vista es una línea de sistemas operativos desarrollada por Microsoft para ser usada en computadores de escritorio, portátiles, Tablet PC y centros multimedia. Antes de ser anunciado oficialmente el 22 de julio de 2003 su nombre en código fue "Longhorn" en español, "Cuerno Largo"

El proceso de desarrollo terminó el 8 de noviembre de 2006 y en los siguientes tres meses fue entregado a los fabricantes de hardware y software, clientes de negocios y canales de distribución. El 30 de enero de 2007 fue lanzado mundialmente y fue puesto a disposición para ser comprado y descargado desde el sitio web de Microsoft.

La aparición de Windows Vista viene más de 5 años después de la introducción de su predecesor, Windows XP, es decir el tiempo más largo entre dos versiones consecutivas de Microsoft Windows. La campaña de lanzamiento fue incluso más costosa que la de Windows 95, ocurrido el 25 de agosto de 1995, debido a que incluye además a otros productos como Microsoft Office 2007, y Exchange Server 2007.

FIGURA I. 16.
Pantalla de Microsoft Windows Vista.

I. 2. E. m. Windows 7.

Windows 7 (seven) es la versión más reciente de Microsoft Windows, línea de sistemas operativos producida por Microsoft Corporation. Esta versión está diseñada para uso en PC, incluyendoequipos de escritorio en hogares y oficinas, equipos portátiles, "tablet PC", "netbooks" y equipos "media center". El desarrollo de Windows 7 se completó el 22 de julio de 2009, siendo entonces confirmada su fecha de venta oficial para el 22 de octubre de 2009 junto a su equivalente para servidores Windows Server 2008 R2.

A diferencia del gran salto arquitectónico y de características que sufrió su antecesor Windows Vista con respecto a Windows XP, Windows 7 fue concebido como una actualización incremental y focalizada de Vista y su núcleo NT 6.0, lo que permitió el mantener cierto grado de compatibilidad con aplicaciones y hardware en los que éste ya era compatible. Sin embargo, entre las metas de desarrollo para Windows 7 se dio importancia en mejorar su interfaz para volverla más accesible al usuario e incluir nuevas características que permitieran hacer tareas de una manera más fácil y rápida, al mismo tiempo en que se realizarían esfuerzos para lograr un sistema más ligero, estable y rápido.

Diversas presentaciones dadas por la compañía en el 2008 se enfocaron en demostrar capacidades multitáctiles, una interfaz rediseñada junto con una nueva barra de tareas y un sistema de redes domésticas fácil de usar denominado Grupo en el Hogar, además de grandes mejoras en el rendimiento general del equipo.

FIGURA I. 17.
Pantalla de Microsoft Windows 7 (seven).

I. 2. E. n. LINEA DE SISTEMAS OPERATIVOS Y FAMILIA Windows NT.

Línea de sistemas operativos tipo Windows desarrollado por Microsoft dedicados originalmente a estaciones de trabajo y servidores de red y luego orientados al hogar y profesionales desde su versión de Windows XP en adelante.

Las características comunes a todas sus versiones son:

- Sistema de archivos NTFS (New Technology File System: Sistema de Archivos con Nueva Tecnología).
- Multitarea, multiusuario y multiprocesador.
- Arquitectura de 32 bits y actualmente 64 bits.
- Compatibilidad con otros sistemas de red.

Versión.	Nombre.	Ediciones.	Lanzamiento.
NT 3.1.	Windows NT 3.1.	Workstation, Advanced Server.	Jul 1993.
NT 3.5.	Windows NT 3.5.	Workstation, Server.	Sep 1994.
NT 3.51.	Windows NT 3.51.	Workstation, Server.	May 1995.
NT 4.0.	Windows NT 4.0.	Workstation, Server, Server Enterprise Edition, Terminal Server, Embedded.	Jul 1996.
NT 5.0.	Windows 2000.	Professional, Server, Advanced Server, Datacenter Server.	Feb 2000.
NT 5.1.	Windows XP.	Home, Professional, Media Center 04 y 05, Tablet PC, Starter, Embedded.	Oct 2001.
NT 5.2.	Windows Server 2003.	Standard, Enterprise, Datacenter, Web, XP Pro x64.	Abr 2003.
NT 6.0.	Windows Vista.	Starter, Home Basic, Home Premium, Business, Enterprise, Ultimate.	Ene 2007.
NT 6.1.	Windows 7.	Starter Edition, Home Basic, Home Premium, Professional, Enterprise, Ultimate.	Oct 2009.

TABLA I. 6.
Versiones de la familia Windows NT.

I. 2. F. OTROS SISTEMAS OPERATIVOS PARA COMPUTADORAS PERSONALES.

Aunque Microsoft se ha convertido en el líder incuestionable del mercado en sistemas operativos, esto no siempre ha sido así. Otros sistemas operativos también han sido populares, y algunos continúan manteniendo la lealtad inquebrantable de sus seguidores.

I. 2. F. a. EL SISTEMA OPERATIVO MACINTOSH.

El equipo Macintosh es una computadora puramente gráfica. En sus inicios, a mediados de la década de 1980, la integración estrecha de su hardware, su sistema operativo y su GIU (Interfaz gráfica de usuario: Graphical User Interface) la hicieron la favorita de los usuarios que no deseaban lidiar con la interfaz de línea de comandos de DOS. Otra gran ventaja de Macintosh era que todas sus aplicaciones funcionaban en forma parecida, aplicando los conceptos de acceso de usuario común y haciendo que fueran más fáciles de aprender que las aplicaciones de DOS. Ahora que las GUI se han vuelto la norma, es difícil apreciar el gran avance que esto fue en realidad.

EL sistema operativo Macintosh también estaba delante de Windows con respecto a muchas otras características, que incluían compatibilidad de hardware de conectar y usar, y conexiones en red incorporada. No obstante el sistema operativo Mac sólo funciona en hardware Macintosh y compatible, mientras que DOS y todas las variedades de Windows sólo trabajan en computadoras compatibles con IBM (las cuales abarcan el 85% del mercado de las PC's). Aun así, Mac sigue siendo la primera opción de muchos editores, desarrolladores de multimedia, artistas gráficos y escuelas.

Para mediados de la década de 1990, Apple controlaba alrededor de 10% del mercado internacional de las computadoras personales, superando a su anterior rival, IBM. Sin embargo, en la actualidad, Apple controla una parte mucho menor del mercado, de 5% a 7%, conforme los usuarios emigran en forma creciente a los sistemas basados en Windows. Esta emigración es el resultado de la incapacidad de Apple para actualizar el sistema operativo Macintosh para mantener el paso de los competidores, problema que ha desilusionado tanto a los usuarios como a los desarrolladores del software Macintosh.

FIGURA I. 18.
Pantalla Mac-osx.

I. 2. F. b. OS/2.

Aunque ahora son rivales feroces, IBM y Microsoft fueron aliados alguna vez. Después de la introducción del procesador Intel 80286 en 1982, ambas compañías reconocieron la necesidad de sacar ventaja de las nuevas capacidades de esta CPU. Trabajaron en equipo para sacar OS/2, un moderno sistema operativo multitareas para microprocesadores Intel. La sociedad no duró mucho. Diferencias de opinión técnica, y la percepción de IBM de que Microsoft Windows sería amenaza para el OS/2 (que resultó ser cierto) causaron la ruptura de esta relación, que al final condujo a la disolución de la sociedad.

IBM continuó desarrollando y promoviendo OS/2 como piedra angular estratégica de SAA (System Application Arquitecture: Arquitectura de Sistema de Aplicación), un plan general para computación comercial global.

Como Windows NT, OS/2 es un sistema operativo multitareas de un solo usuario con una interfaz de señalar y hacer clic. También es un sistema multitareas verdadero.

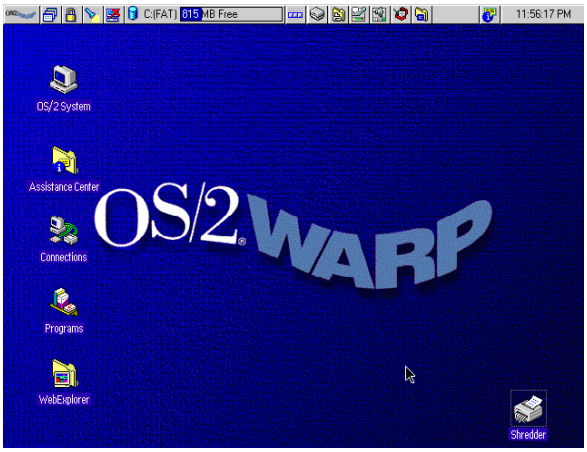


FIGURA I. 19.
Pantalla OS/2 WARP.

I. 2. F. c. UNIX.

UNIX (UNICS Uniplexed Information And Computing: Sistema Operativo Portable, Multitarea y Multiusuario) es más antiguo que todos los demás sistemas operativos para PC's, y de muchos modos sirvió como modelo para ello. Desarrollado al principio por Bell Labs y dirigido hacia las telecomunicaciones. UNIX fue vendido a Novell a principios de los 90 del siglo XX. Debido a su uso inicial en ámbitos universitarios, se desarrollaron varias versiones. La más conocida es Berkeley UNIX. Otras versiones incluyen A/UX para Mac y AIX para estaciones de trabajo IBM de alto rendimiento.

UNIX no era tan solo un sistema multitarea como los otros sistemas operativos sino también era un sistema operativo multiusuario y multiprocesamiento; es decir, permitía que multiusuarios trabajan desde mas de un teclado y un monitor conectado a una sola CPU cómo lo hace una mainframe con

terminales no inteligentes. UNIX permitió que una PC trabajara con más de una CPU a la vez, hazaña admirable para las normas iniciales de la PC.

UNIX corre con muchos tipos diferentes de computadoras: en supercomputadoras Cray, en PC y de todo lo que esta en el medio computacional, incluyendo mainframes y minicomputadoras. Debido a su capacidad de trabajar con tantas clases de hardware, se convirtió en la columna vertebral de internet. Gracias a su potencia y otros usuarios de software CAD (Computer Aided Design: Diseño Asistido por Computador), CAM (Computer Aided Manufacturing*:* Manufactura Asistida por Computador) y CIM (Computer Intergrated Manufacturing, Manufactura Integrada por Computador) UNIX tiene un uso popular en estaciones de trabajo RISC como las de Sun Microsystem, Hewelett-Packard, IBM y Silicon Graphics.

Aunque es un sistema operativo robusto y capaz, la líneas de comandos UNIX no son débiles de corazón porque requieren de muchos comandos incluso para hacer cosas simples. Un sistema UNIX de ventanas llamado X-Windows incorpora una GUI a las computadoras UNIX, pero continúa siendo un sistema operativo para ingenieros y usuarios técnicos. Aunque algunas de sus capacidades han entrado en la corriente principal, UNIX ha perdido terreno en forma gradual ante sistemas operativos como DOS, Windows y Mac, los cuales por lo general se han percibido como más fáciles de aprender y usar.

I. 3. LENGUAJES DE PROGRAMACIÓN.

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una computadora, particularmente una computadora. Consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

Un lenguaje de programación permite a uno o más programadores especificar de manera precisa, sobre qué datos una computadora debe operar, cómo deben ser estos almacenados, transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural, tal como sucede con el lenguaje Léxico. Una característica relevante de los lenguajes de programación es precisamente que más de un programador pueda tener un conjunto común de instrucciones que puedan ser comprendidas entre ellos para realizar la construcción del programa de forma conjunta.

Los procesadores usados en las computadoras son capaces de entender y actuar según lo indican programas escritos en un lenguaje fijo llamado lenguaje de máquina. Todo programa escrito en otro lenguaje puede ser ejecutado de dos maneras:

- Mediante un programa que va adaptando las instrucciones conforme son encontradas. A este proceso se lo llama interpretar y a los programas que lo hacen se los conoce como intérpretes.

- Traduciendo este programa al programa equivalente escrito en lenguaje de máquina. A ese proceso se le llama compilar y al traductor se lo conoce como un malhecho compilador.

I. 3. A. EL LENGUAJE MÁQUINA.

El lenguaje máquina es el lenguaje mas básico, consisten en cadenas de números binarios (0's y 1's) y son definidos por el diseño del hardware. En otras palabras, el lenguaje máquina para una Macintosh no es el mismo para una PC. Una computadora comprende solo su lenguaje máquina original, los comandos de su equipo de instrucción. Estos comandos le dan instrucciones a la computadora para realizar operaciones elementales: cargar, almacenar, añadir y sustraer. Esencialmente el código máquina consiste por completo de los 0's y 1's del sistema numérico binario.

I. 3. B. EL LENGUAJE DE BAJO NIVEL.

Los lenguajes de banjo nivel o ensambladores fueron desarrollados usando nemotécnicos similares a las palabras del idioma inglés. Los programadores trabajaban en editores de texto, que son simples procesadores de palabras, para crear archivos fuente. Los archivos fuente contienen instrucciones para que la computadora las ejecute, pero tales archivos deben primero traducirse al lenguaje máquina. Los investigadores crearon programas traductores llamados ensambladores para realizar la conversión. Los lenguajes ensambladores aun son altamente detallados y secretos, pero leer un código ensamblador es mucho más rápido que batallar con el lenguaje máquina. Los programadores rara vez escriben programas de tamaño significativo en un lenguaje ensamblador. (Una excepción se encuentra en los juegos de acción en donde la velocidad del programa es decisiva). En su lugar se usan lenguajes ensambladores para afinar partes importantes de programas escritos en un lenguaje de nivel superior.

I. 3. C. EL LENGUAJE DE ALTO NIVEL.

Los lenguajes de alto nivel fueron desarrollados para hacer más fácil la programación. Estos lenguajes son llamados de alto nivel por que su sintaxis es mas cercana al lenguaje humano que el código del lenguaje máquina o ensamblador. Usan palabras familiares en lugar de comunicar en el detallado embrollo de los dígitos que comprenden las instrucciones de la máquina. Para expresar las operaciones de la computadora estos lenguajes usan operadores, como los símbolos de más o menos, que son los componentes familiares de las matemáticas. Como resultado, leer, escribir y comprender programas de computo es mas fácil con un programa de alto nivel, a pesar de que las instrucciones todavía deban ser traducidas al lenguaje máquina antes de que la computadora pueda comprenderlas y llevarlas acabo.

I. 3. C. a. INTÉRPRETES.

Estos lenguajes también denominados lenguajes de tercera generación, tienen la capacidad de soportar programación estructurada lo cual significa que proporciona estructuras explicitas para diagramas de árbol y ciclos. Además, debido a que son los primeros lenguajes que usan fraseo similar al inglés, compartir el desarrollo entre los programadores también es más fácil. Los miembros del equipo pueden leer el código de cada uno de los demás y comprender la lógica y el flujo de control del programa.

Estos programas también son portátiles. En oposición a los lenguajes ensambladores, los programas en estos lenguajes pueden ser compilados para ejecutarse en numerosas CPU's. Además no importa la versión aun así se ejecutan.

I. 3. C. a. i. FORTRAN.

El lenguaje Fortran, es un lenguaje de programación desarrollado en los años 50 y activamente utilizado desde entonces. Acrónimo de "Formula Translator". Fortran se utiliza principalmente en aplicaciones científicas y análisis numérico. Desde 1958 ha pasado por varias versiones, entre las que destacan FORTRAN II, FORTRAN IV, FORTRAN 77, Fortran 90, Fortran 95, Fortran 2003 y Fortran 2008. Si bien el lenguaje era inicialmente un lenguaje imperativo, las últimas versiones incluyen elementos de la programación orientada a objetos. El lenguaje fue diseñado tomando en cuenta que los programas serían escritos en tarjetas perforadas de 80 columnas. Así por ejemplo, las líneas debían ser numeradas y la única alteración posible en el orden de ejecución era producida con la instrucción go to. Estas características han evolucionado de versión en versión. Las versiones actuales contienen subprogramas, recursión y una variada gama de estructuras de control.

I. 3. C. a. ii. COBOL.

El lenguaje COBOL (acrónimo de COmmon Business Oriented Language: Lenguaje Común Orientado a Negocios) fue creado en el año 1960, con el objetivo de crear un lenguaje de programación universal que pudiera ser usado en cualquier computadora, ya que en esa década, existían numerosos modelos de computadoras incompatibles entre sí, y que estuviera orientado principalmente a los negocios, es decir, a la llamada informática de gestión.

COBOL fue dotado por diseño de unas excelentes capacidades de autodocumentación, una buena gestión de archivos y una excelente gestión de los tipos de datos para la época, a través de la conocida sentencia PICTURE para la definición de campos estructurados. Para evitar errores de redondeo en los cálculos que se producen al convertir los números digitales a números binarios y que son inaceptables en temas comerciales, COBOL puede emplear y emplea por defecto números en base diez. Para facilitar la creación de programas en COBOL, la sintaxis del mismo fue creada de forma que fuese parecida al idioma inglés, evitando el uso de símbolos que se impusieron en lenguajes de programación posteriores.

Pese a esto, a comienzos de los ochenta del siglo XX se fue quedando anticuado respecto a los nuevos paradigmas de programación y a los lenguajes que los implementaban. En la revisión de 1985, se solucionó, incorporando a COBOL variables locales, recursividad, reserva de memoria dinámica y programación estructurada. En la revisión de 2002, se le añadió orientación a objetos, aunque desde la revisión de 1974, se podía crear un entorno de trabajo similar a la orientación a objetos, y un método de generación de pantallas gráficas estandarizado.

Antes de la inclusión de las nuevas características en el estándar oficial, muchos fabricantes de compiladores las añadían de forma no estándar. Este proceso se vio con la integración de COBOL con internet. Existen varios compiladores que permiten emplear COBOL como lenguaje de scripting y de servicio Web. También existen compiladores que permiten generar código COBOL para la plataforma .NET y Web.

I. 3. C. a. iii. ALGOL.

Se denomina Algol a un lenguaje de programación. Algol es un acrónimo de las palabras inglesas Algorithmic Language (lenguaje algorítmico).

Fue muy popular en las universidades durante los años 60 del siglo pasado, pero no llegó a solidificarse como lenguaje de utilización comercial. Sin embargo, Algol influyó profundamente en varios lenguajes posteriores que sí alcanzaron gran difusión, como Pascal, C y Ada.

Hacia 1965, dos corrientes se distinguieron sobre el tema de un sucesor para Algol. Como resultado se definieron los lenguajes Algol W que es un lenguaje minimalista, rápidamente implementado y distribuido y, por otra parte, Algol 68 que para la época está en la frontera entre un lenguaje para programar en él y un lenguaje para investigar sobre él.

EL Lenguaje Algol W elaborado diseñado por Niklaus Wirth y Tony Hoare a partir de los trabajos del grupo Algol de la IFIP. Se trata de un lenguaje conciso, simple de implementar, que evita todos los defectos conocidos del lenguaje Algol e incluye sus propias características adicionales. Sin embargo, el grupo Algol no lo adoptó como sucesor de Algol prefiriendo en su lugar al que terminó siendo Algol 68. Algol W fue utilizado por gran cantidad de usuarios y sembró el camino para el nacimiento del lenguaje Pascal.

Entre las características del lenguaje se destacan: aritmética de doble precisión, números complejos, strings y estructuras de datos dinámicas, evaluación por valor, pasaje de parámetros por valor, valor resultado o resultado.

La definición del lenguaje fue presentada en la reunión del comité Algol de la IFIP (International Federation for Information Processing: Federación Internacional de Sociedades para el Proceso de la Información) en 1965. Luego de varios años de revisión del diseño se llegó a una versión definitiva en 1968. Al principal autor es Adriaan van Wijngaarden.

Los objetivos principales de ALGOL 68 son el permitir comunicar algoritmos, el permitir una eficiente ejecución de los mismos en diferentes arquitecturas y el de servir como herramienta para la enseñanza. Una característica interesante de ALGOL 68 es que su semántica fue definida formalmente antes de ser implementado en base al formalismo llamado gramáticas de dos niveles.

I. 3. C. b. COMPILADORES.

Un compilador es un programa que permite traducir el código fuente de un programa en lenguaje de alto nivel, a otro lenguaje de nivel inferior (típicamente lenguaje máquina). De esta manera un programador puede diseñar un programa en un lenguaje mucho más cercano a como piensa un ser humano, para luego compilarlo a un programa más manejable por una computadora.

Normalmente los compiladores están divididos en dos partes:

- Front End: es la parte que analiza el código fuente, comprueba su validez, genera el árbol de derivación y rellena los valores de la tabla de símbolos. Esta parte suele ser independiente de la plataforma o sistema para el cual se vaya a compilar.

- Back End: es la parte que genera el código máquina, específico de una plataforma, a partir de los resultados de la fase de análisis, realizada por el Front End.

Esta división permite que el mismo Back End se utilice para generar el código máquina de varios lenguajes de programación distintos y que el mismo Front End que sirve para analizar el código fuente de un lenguaje de programación concreto sirva para generar código máquina en varias plataformas distintas.

El código que genera el Back End normalmente no se puede ejecutar directamente, sino que necesita ser enlazado por un programa enlazador (linker).

I. 3. C. b. i. Fox Pro.

FoxPro que originalmente era FoxBASE Profesional, es un lenguaje de programación orientado a procedimientos (procedures), a la vez que es un Sistema Gestor de Bases de datos o Database Management System (DBMS), publicado originalmente por Fox Software y posteriormente por Microsoft, para los sistemas operativos MS DOS, MS Windows, Mac OS y UNIX.

Aunque FoxPro es un DBMS y como tal soporta relaciones entre las tablas de datos, no se le considera como un sistema administrador de bases de datos relacionales (RDBMS), por no soportar las transacciones.

FoxPro surgió como mejoras del dBase de Ashton-Tate, con el que comparten la base sintáctica y la gestión del formato DBF de fichero de base de datos, pero que difieren en la gestión de los campos MEMO y los archivos de índices. Así los ficheros de campo de FoxBASE tienen extensión .ftp y presentan una mejor gestión y una mayor resistencia a la corrupción en caso de cuelgue de la computadora. Como la mayoría de dialectos xBASE, FoxBASE es además un compilador que genera ficheros .EXE independientes.

Precisamente una de las novedades de Fox Pro fueron los archivos de índice múltiple con extensión .cdx. En lugar de tener un archivo por cada índice creado a la tabla de datos DBF, Fox presentaba un fichero único (con la ventaja del ahorro de espacio, algo muy importante en computadoras basadas en disquete donde el disco duro solía estar reservado a empresas), pero además el índice tenía eficacia respecto del resto de competidores, por lo que, gracias a librerías de terceros, derivo en estándar de factor de índices para los sistemas xBase.

En aquel entonces la mayoría de equipos se basaban en una interfaz de línea de comandos en modo texto (aunque dispusieran de capacidades gráficas). Un tercer avance de FoxPro es la integración de un sistema de ventanas en su escritorio, que le da un aspecto muy parecido al DESQview (aunque desde luego sin sus capacidades multitarea). Este sistema tenía soporte de mouse, con botones para cerrar las ventanas. Además integra SQL en el lenguaje.

En 1993, Microsoft lanza FoxPro 2.5 para Windows, la primera versión de Fox con soporte de interfaz gráfica, que en menos de un año es sustituido por FoxPro 2.6, la cual es considerada la última versión de FoxPro propiamente dicha. Otra de las novedades de la 2.6 es la aparición de una versión para Apple Macintosh, con la integración en el lenguaje de sentencias y opciones exclusivas del Mac. FoxPro 2.6 para UNIX, FPU26 (Floating Point Unit: Unidad de Punto Flotante), ha sido instalado en Linux y FreeBSD utilizando la librería de soporte ibcs2. Varios proyectos Open Source

derivados de xBASE incluyen en sus desarrollos el soporte de algunas de las particularidades de FoxPro, como los archivos de índice CDX (Central Data Exchange: Intercambio central de datos).

Visual FoxPro (VFP) proviene de FoxPro, creado por Fox Technologies en 1984; fue adquirido por Microsoften 1992.

Visual FoxPro 3.0, fue la primera versión "Visual", redujo su compatibilidad a solo Mac y Windows (La última versión de FoxPro 2.6) corría en MS DOS, MS Windows, Mac OS y UNIX), versiones posteriores fueron solo para Windows. La versión actual se basa en archivos COM y Microsoft ha declarado que no piensan crear una versión .NET.

En la versión 5.0 se integra en Microsoft Visual Studio añadiéndosele el soporte de Microsoft Source Safe. Hasta entonces es visto típicamente por el público como meramente un Sistema de gestión de base de datos (SGBD), ignorando el hecho de que no solo incluye el entorno SGBD, sino un completo lenguaje de programación.

Visual FoxPro 6.0, publicado en 1999, no supone un cambio radical respecto de la anterior versión sino únicamente una mejora en sus diversas funcionalidades y una adaptación al mundo internet y al mundo de los objetos. Esta versión hace más atractivo a los desarrolladores el tratamiento de los datos en los entornos COM. Es un paso más en la evolución de este producto desde un entorno de aplicaciones monousuario o de redes pequeñas centradas en los datos hacia una herramienta orientada a objeto diseñada para la construcción de la lógica del negocio en los entornos multi-tier con una fuerte orientación hacia los tratamientos intensivos de datos en Internet. Pese a su relativa antigüedad, es hoy todavía ámpliamente utilizado en grandes empresas (por ej., la compañía de seguros Mapfre) por su estabilidad.

Visual FoxPro 7.0, publicado en 2001, supuso su salida de Visual Studio, pues aunque en un principio se pensaba incluir a Fox en .NET, no era posible sin romper con la herencia de anteriores versiones. Esta versión incorporó por primera vez el IntelliSense, y se mejoró el manejo de arrays, acercándolo al de cursores.

A finales del 2002, algunos miembros de comunidades demostraron que Visual FoxPro puede correr en Linux usando un reimplememtador del API de Windows (Win16 y Win32) llamado Wine. En el 2003, esto llevo a quejas de Microsoft: se dijo que el desarrollo de código de FoxPro para rutinas en máquinas no-Windows viola el Acuerdo de Licencia de Usuario Final. FoxPro, ha tenido el tiempo de vida de soporte más largo para un producto de Microsoft (hasta el 2014).

Visual FoxPro 9 (VFP 9) fue lanzado el 17 de diciembre del 2004 y el equipo de Fox luego trabajó en un proyecto cuyo nombre clave fue Sedna. Este fue construido sobre el código base de VFP 9 y consistió principalmente en componentes Xbase que soportando un número de escenarios interoperables con varias tecnologías de Microsoft incluyendo SQL Server 2005, .NET, WinFX, Windows Vista y Office 12. Lamentablemente el proyecto no prosperó y fue cancelado por Microsoft.

Visual FoxPro no va a desaparecer ya que una empresa llamada etecnologia "www.etecnologia.net" ha desarrollado el .NET EXTENDER que permite utilizar el .net framework en visual foxpro, y han anunciado que a finales del 2009 van a sacar su "VFP Developer Studio" herramienta la cual convierte a VFP en un lenguaje .Net.

I. 3. C. b. ii. QuickBASIC.

Microsoft QuickBASIC es un descendiente del lenguaje de programación BASIC que Microsoft Corporation desarrolló para su uso con el sistema operativo MS DOS. Estaba ligeramente basado en GW-BASIC pero añadía tipos definidos por el usuario, estructuras de programación mejoradas, mejores gráficos y soporte de disco, y un compilador además del intérprete. Microsoft sacó a la venta QuickBasic como un paquete de desarrollo comercial.

Microsoft publicó la primera versión de QuickBASIC el 18 de agosto de 1985, en un único disquete de 5.25". QuickBASIC usaba un entorno de desarrollo integrado IDE (Integrated Development Environment: Entorno Integrado de Desarrollo) radicalmente diferente al que acompañaba a las versiones anteriores de BASIC. Los números de línea ya no eran necesarios puesto que los usuarios podían insertar y quitar líneas directamente mediante un editor de textos en pantalla.

QuickBASIC incluía el "Compilador PC BASIC", que servía para compilar los programas en ejecutables DOS. El editor poseía un intérprete con el cual el usuario podía ejecutar un programa sin tener que cerrar el editor en absoluto, y podía usarse para depurar un programa antes de crear el fichero ejecutable. Desgraciadamente, había ciertas pequeñas diferencias entre el intérprete y el compilador, por lo que programas que ejecutaban perfectamente en el intérprete no lo hacían en su versión compilada o incluso no llegaban a compilar.

La última versión de QuickBASIC fue la 4.5 aunque se siguió desarrollando como el PDS (Professional Development System*:* Sistema de Desarrollo Profesional), cuya última versión fue la 7.1. A la versión PDS también se le llamó a veces QuickBASIC Extendido.

El sucesor de QuickBASIC y PDS fue Visual Basic 1.0, que salió en dos versiones incompatibles entre sí, una para DOS y otra para Windows. Las versiones posteriores de Visual Basic no incluían versiones para DOS, ya que Microsoft quería que los desarrolladores se concentraran en las aplicaciones para Windows.

I. 3. C. c. .NET.

.NET es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de plataforma y que permita un rápido desarrollo de aplicaciones. Basado en esta plataforma, Microsoft intenta desarrollar una estrategia horizontal que integre todos sus productos, desde el sistema operativo hasta las herramientas de mercado.

.NET podría considerarse una respuesta de Microsoft al creciente mercado de los negocios en entornos Web, como competencia a la plataforma Java de Sun Microsystems.

Debido a las ventajas que la disponibilidad de una plataforma de este tipo puede darle a las empresas de tecnología y al público en general, muchas otras empresas e instituciones se han unido a Microsoft en el desarrollo y fortalecimiento de la plataforma .NET, ya sea por medio de la implementación, el desarrollo de lenguajes de programación adicionales o la creación de bloques adicionales para la plataforma dentro y fuera del sistemas operativos Windows (como controles, componentes y bibliotecas de clases adicionales) siendo algunas de ellas software libre.

Con esta plataforma Microsoft incursiona de lleno en el campo de los servicios Web y establece el XML (eXtensible Markup Language: Lenguaje de Descripción de Página) como norma en el transporte de información en sus productos y lo promociona como tal en los sistemas desarrollados utilizando sus herramientas.

.NET intenta ofrecer una manera rápida y económica pero a la vez segura y robusta de desarrollar aplicaciones (o soluciones) permitiendo a su vez una integración más rápida y ágil entre empresas y un acceso más simple y universal a todo tipo de información desde cualquier tipo de dispositivo.

Este es el lenguaje insignia del marco de trabajo .NET, y pretende reunir las ventajas de lenguajes como C/C++ y Visual Basic en un solo lenguaje.

La norma BCL (Base Class Library: Biblioteca de Clases Base), tal vez el más importante de los componentes de la plataforma, define un conjunto funcional mínimo que debe implementar la biblioteca de clases base, para que el marco de trabajo sea soportado por un sistema operativo. Aunque Microsoft implementó esta norma para su sistema operativo Windows.

Lla publicación de la norma abre la posibilidad de que sea implementada para cualquier otro sistema operativo existente o futuro, permitiendo que las aplicaciones corran sobre la plataforma independientemente del sistema operativo para el cual haya sido implementada.

El Proyecto Mono emprendido por Ximian pretende realizar la implementación de la norma para varios sistemas operativos adicionales bajo el marco del software libre o código abierto.

Los principales componentes del marco de trabajo son: el conjunto de lenguajes de programación, la BCL y el CLR (Common Language Runtime: Entorno Común de Ejecución para Lenguajes).

Debido a la publicación de la norma para la CLI (Common Language Infrastructure: Infraestructura Común de Lenguajes), el desarrollo de lenguajes se facilita, por lo que el marco de trabajo .NET soporta ya más de 20 lenguajes de programación y es posible desarrollar cualquiera de los tipos de aplicaciones soportados en la plataforma con cualquiera de ellos, lo que elimina las diferencias que existían entre lo que era posible hacer con uno u otro lenguaje.

I. 3. C. c. i. Visual Studio.

Microsoft Visual Studio es un Entorno Integrado de Desarrollo para sistemas Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET, aunque actualmente se han desarrollado las extensiones necesarias para muchos otros.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones Web, así como servicios Web en cualquier entorno que soporte la plataforma .NET (a partir de la versión 6.0). Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas Web y dispositivos móviles.

I. 3. C. c. ii. Visual Studio 6.0.

La siguiente versión, la 6.0, se lanzó en 1998, y fue la última versión en ejecutarse en la plataforma Win9x. Los números de versión de todas las partes constituyentes pasaron a 6.0, incluyendo Visual J++ y Visual InterDev que se encontraban en las versiones 1.1 y 1.0 respectivamente. Esta versión fue la base para el sistema de desarrollo de Microsoft para los siguientes 4 años, en los que Microsoft migró su estrategia de desarrollo al Framework .NET.

Visual Studio 6.0 fue la última versión en que Visual Basic se incluía de la forma en que se conocía hasta entonces; versiones posteriores incorporarían una versión muy diferente del lenguaje con muchas mejoras, fruto de la plataforma .NET. También supuso la última versión en incluir Visual J++, que proporcionaba extensiones de la plataforma Java, lo que lo hacía incompatible con la versión de Sun Microsystems. Esto acarreó problemas legales a Microsoft, y se llegó a un acuerdo en el que Microsoft dejaba de comercializar herramientas de programación que utilizaran la máquina virtual de Java.

Aunque el objetivo a largo plazo de Microsoft era unificar todas las herramientas en un único entorno, esta versión en realidad añadía un entorno más a Visual Studio 97: Visual J++ y Visual Interdev se separaban del entorno de Visual C++, al tiempo que Visual FoxPro y Visual Basic seguían manteniendo su entorno específico.

I. 3. C. c. iii. Visual Studio .NET 2002.

En esta versión se produjo un cambio sustancial, puesto que supuso la introducción de la plataforma .NET de Microsoft, .NET es una plataforma de ejecución intermedia multilenguaje, de forma que los programas desarrollados en .NET no se compilan en lenguaje máquina, sino en un lenguaje intermedio CIL (Common Intermediate Language: Lenguaje Intermedio Común) denominado MSIL (Microsoft Intermediate Language: Lenguaje Intermedio de Microsoft). En una aplicación MSIL, el código no se convierte a lenguaje máquina hasta que ésta se ejecuta, de manera que el código puede ser independiente de plataforma (al menos de las soportadas actualmente por .NET). Las plataformas han de tener una implementación de Lenguaje Intermedio Común para poder ejecutar programas MSIL.

Visual Studio .NET 2002 supuso también la introducción del lenguaje C#, un lenguaje nuevo diseñado específicamente para la plataforma .NET, basado en C++ y Java. Se presentó también el lenguaje J# (sucesor de J++) el cual, en lugar de ejecutarse en una máquina virtual de Java, se ejecuta únicamente en el framework .NET. El lenguaje Visual Basic fue remodelado completamente y evolucionó para adaptarse a las nuevas características de la plataforma .NET, haciéndolo mucho más versátil y dotándolo con muchas características de las que carecía. Algo similar se llevó a cabo con C++, añadiendo extensiones al lenguaje llamadas Managed Extensions for C++ con el fin de que los programadores pudieran crear programas en .NET. Por otra parte, Visual FoxPro pasa a comercializarse por separado. Todos los lenguajes se unifican en un único entorno. La interfaz se mejora notablemente en esta versión, siendo más limpia y personalizable.

Visual Studio .NET puede usarse para crear programas basados en Windows (usando Windows Forms en vez de COM), aplicaciones y sitios Web (ASP.NET y servicios Web), y dispositivos móviles (usando el .NET Compact Framework). Esta versión requiere un sistema operativo basado en NT. La versión interna de Visual Studio .NET es la 7.0.

I. 3. C. c. iv. Visual Studio .NET 2003.

Visual Studio .NET 2003 supone una actualización menor de Visual Studio .NET. Se actualiza el .NET Framework a la versión 1.1. También se añade soporte con el fin de escribir aplicaciones para determinados dispositivos móviles, ya sea con ASP.NET o con el .NET Compact Framework. Además el compilador de Visual C++ se mejora para cumplir con más estándares, el Visual C++ Toolkit 2003.

Visual Studio 2003 se lanza en 4 ediciones: Academic, Professional, Enterprise Developer, y Enterprise Architect. La edición Enterprise Architect incluía una implementación de la tecnología de modelado Microsoft Visio, que se centraba en la creación de representaciones visuales de la arquitectura de la aplicación basadas en UML (Unified Modeling Language: Lenguaje Unificado de Modelado). También se introdujo "Enterprise Templates", para ayudar a grandes equipos de trabajo a estandarizar estilos de programación e impulsar políticas de uso de componentes y asignación de propiedades. Microsoft lanzó el Service Pack 1 para Visual Studio 2003 el 13 de Septiembre de 2006. La versión interna de Visual Studio .NET 2003 es la 7.1 aunque el formato del archivo es 8.0.

I. 3. C. c. v. Visual Studio .NET 2005.

Whidbey (nombre en código que puso Microsoft a esta versión en referencia a la isla Whidbey), se empezó a comercializar a través de internet en Octubre de 2005, y llegó a los comercios unas semanas más tarde. Microsoft eliminó la coletilla .NET, pero eso no indica que se alejara de la plataforma .NET, de la cual se incluyó la versión 2.0 de la máquina virtual.

La actualización más importante que recibieron los lenguajes de programación fue la inclusión de tipos genéricos, similares en muchos aspectos a las plantillas de C++. Con esto se consigue encontrar muchos más errores en la compilación en vez de en tiempo de ejecución, incitando a usar comprobaciones estrictas en áreas donde antes no era posible. C++ tiene una actualización similar con la adición de C++/CLI como sustituto de C++ manejado.

Se incluye un diseñador de implantación, que permite que el diseño de la aplicación sea validado antes de su implantación. También se incluye un entorno para publicación Web y pruebas de carga para comprobar el rendimiento de los programas bajo varias condiciones de carga.

Visual Studio 2005 también añade soporte de 64-bit. Aunque el entorno de desarrollo sigue siendo una aplicación de 32 bits. Visual C++ 2005 soporta compilación para x86-64 (AMD64 e Intel 64) e IA-64 (Itanium). El SDK (Software Development Kit: Kit de Desarrollo de Software) incluye compiladores de 64 bits así como versiones de 64 bits de las librerías. Visual Studio 2005 tiene varias ediciones radicalmente distintas entre sí: Express, Standard, Professional, Tools for Office, y 5 ediciones Visual Studio Team System. Éstas últimas se proporcionaban conjuntamente con suscripciones a MSDN (Microsoft Developer Network: Plataforma para Desarrolladores de Microsoft) cubriendo los 4 principales roles de la programación: Architects, Software Developers, Testers, y Database Professionals. La funcionalidad combinada de las 4 ediciones Team System se ofrecía como la edición Team Suite.

Tools for the Microsoft Office System está diseñada para extender la funcionalidad a Microsoft Office. Las ediciones Express se han diseñado para principiantes, aficionados y pequeños negocios, todas disponibles gratuitamente a través de la página de Microsoft se incluye una edición

independiente para cada lenguaje: Visual Basic, Visual C++, Visual C#, Visual J# para programación .NET en Windows, y Visual Web Developer para la creación de sitios Web ASP.NET. Las ediciones express carecen de algunas herramientas avanzadas de programación así cómo de opciones de extensibilidad.

Se lanzó el service Pack 1 para Visual Studio 2005 el 14 de Diciembre de 2006. La versión interna de Visual Studio 2005 es la 8.0, mientras que el formato del archivo es la 9.0.

I. 3. C. c. vi. Visual Studio .NET 2008.

Visual Studio 2008, Code-Named Orcas, es el sucesor de Visual Studio 2005. El nombre en código Orcas es, al igual que Whidbey, una referencia a una isla en Puget de sonido, Orcas isla. El sucesor de Visual Studio 2008 es cuyo nombre en código es Rosario. La primera versión beta disponible públicamente era en septiembre 2006 CTP, publicadas de 28 de septiembre de 2006.

Visual Studio 2008 está orientado al desarrollo de Windows Vista, Office 2007 y aplicaciones Web. Entre otras cosas, aporta una nueva característica de lenguaje, LINQ, nuevas versiones de los lenguajes C# y Visual Basic, un diseñador visual para Windows Presentation Foundation y las mejoras a .NET Framework.. Visual Studio 2008 requiere .NET Framework 3.5 y configura ensamblados compilados para que se ejecute en .NET Framework 3.5 de forma predeterminada pero también admite elegir a los programadores qué versión de Common Language Runtime de .NET (fuera de 2.0, 3.0, Silverlight CoreCLR o .NET Compact Framework motores en tiempo de ejecución) se ejecutará en el ensamblado. Visual Studio 2008 también tiene nuevas herramientas de análisis de código, incluida la nueva herramienta código medidas.

El depurador de Visual Studio contiene las características de destino para que sea más fácil la depuración de las aplicaciones de múltiples subprocesos. En la depuración modo, en la ventana Subprocesos, que enumera todos los subprocesos, colocar el puntero sobre un subproceso mostrará el seguimiento de la pila de ese subproceso en información sobre herramientas. Los subprocesos pueden ser directamente denominados y marcados para una identificación más fácil de esa ventana.

I. 3. C. c. vii. Visual Studio .NET 2010.

Visual Studio 2010. Es la versión más reciente de esta herramienta, acompañada por .NET Framework 4.0. La fecha prevista para el lanzamiento de la versión final ha sido el 12 de Abril de 2010. Hasta ahora, uno de los mayores logros de la versión 2010 de Visual Studio ha sido el de incluir las herramientas para desarrollo de aplicaciones para Windows 7, tales como herramientas para el desarrollo de la Taskbar (System.Windows.Shell) y la Ribbon Preview para WPF.

I. 3. C. d. WEB.

La programación Web, parte de las siglas www, que significan World Wide Web o telaraña mundial. Para realizar una pagina con la programación Web, se deben tener claros, tres conceptos fundamentales los cuales son, el URL (Uniform Resource Locators: Localizador de Recursos Universal), es un sistema con el cual se localiza un recurso dentro de la red, este recurso puede ser una pagina Web, un servicio o cualquier otra cosa. En resumen el URL no es más que un nombre,

que identifica una computadora, dentro de esa computadora un archivo que indica el camino al recurso que se solicita.

El siguiente concepto dentro de la programación Web, es el protocolo encargado de llevar la información que contiene una página Web por toda la red de internet, como es el http (Hypertext Transfer Protocol: Protocolo de Transferencia de Páginas). Y por ultimo el lenguaje necesario cuya funcionalidad es la de representar cualquier clase de información que se encuentre almacenada en una pagina Web, este lenguaje es el HTML (Hypertext Markup Language: Lenguaje de descripción de páginas).

En la programación Web, el HTML es el lenguaje que permite codificar o preparar documentos de hipertexto, que viene a ser el lenguaje común para la construcción de una pagina Web.

Con el comienzo de internet y la programación Web, se desfasaron los diseños gráficos tradicionales, con lo que se empezaron a diseñar interfaces concretas para este medio, buscando ficheros pequeños para facilitar la carga de los mismos. La programación Web se orientaba a un diseño muy cargado e interactuando con el usuario, mientras que al empezar a competir con millones de Web's se ha optado más por el diseño sencillo y de fácil comprensión.

En programación Web se creó la necesidad de conocer a fondo diferentes lenguajes de programación como HTML, JavaScript y HTTP.

Con esto se creó un nuevo profesional de la informática, el diseñador Web, experto en estos menesteres, que viene siendo algo así como un experto en programación Web, interactuado entre el diseñador gráfico tradicional y el programador de aplicaciones llevadas a internet.

I. 3. C. d. i. HTML.

El HTML no es más que una aplicación del SGML (Standard Generalized Markup Language: Lenguaje Estándar de Marcación General), un sistema para definir tipos de documentos estructurados y lenguajes de marcas para representar esos mismos documentos. El término HTML se suele referir a ambas cosas, tanto al tipo de documento como al lenguaje de marcas.

A medida que nos afianzamos en el manejo de internet cada uno de nosotros pasa por tres etapas diferentes: Al principio solamente conocemos unas pocas páginas, luego nos damos cuenta que existen buscadores lo cual lo hace más interesante y por último nos damos cuenta que en internet no solamente se puede ver la información sino que también se puede publicar. internet tiene acceso a todos los rincones del mundo.

Para que varias personas se comuniquen es necesario que hablen un mismo idioma. El lenguaje que utilizan las computadoras que están conectadas a internet es HTML.

El HTML, (Hyper Text Markup Language: Lenguaje de marcación de Hipertexto) es el lenguaje de marcas de texto utilizado normalmente en la World Wide Web. Fue creado en 1986, por el físico nuclear Tim Berners-Lee; el cual tomó dos herramientas preexistentes: El concepto de Hipertexto (conocido también como link o ancla) el cual permite conectar dos elementos entre si y el SGML el cual sirve para colocar etiquetas o marcas en un texto que indique como debe verse. HTML no es propiamente un lenguaje de programación como C++, Visual Basic, etc., sino un sistema de

etiquetas. HTML no presenta ningún compilador, por lo tanto algún error de sintaxis que se presente éste no lo detectará y se visualizara en la forma como éste lo entienda.

El entorno para trabajar HTML es simplemente un procesador de texto, como el que ofrecen los sistemas operativos Windows (Bloc de notas), UNIX (el editor vi o ed) o el que ofrece MS Office (Word). El conjunto de etiquetas que se creen, se deben guardar con la extensión .htm o .html. Estos documentos pueden ser mostrados por los visores o "browsers" de paginas Web en internet, como Netscape Navigator, Mosaic, Opera y Microsoft Internet Explorer.

Para crear una página Web se pueden utilizar varios programas especializados en esto, como por ejemplo, el Microsoft Front Page o el Macromedia Dreamweaver 3. Otra forma de diseñar un archivo .html, es copiar todo en el Bloc de Notas del Windows, ya que este sencillo programa cumple con un requisito mínimo que es la posibilidad de trabajar con las etiquetas que emplea este lenguaje.

I. 3. C. d. ii. HTTP.

El protocolo de transferencia de hipertexto (HTTP, HyperText Transfer Protocol) es el protocolo usado en cada transacción de la Web (www). HTTP fue desarrollado por el consorcio W3C y la IETF, colaboración que culminó en 1999, con la publicación de una serie de RFC`s (Request For Comments: Petición de Comentarios), siendo el más importante de ellos el RFC 2616, que especifica la versión 1.1.

HTTP define la sintaxis y la semántica que utilizan los elementos software de la arquitectura Web (clientes, servidores, proxies) para comunicarse. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Al cliente que efectua la petición (un navegador o un spider) se lo conoce como "user agent" (agente del usuario). A la información transmitida se le llama recurso y se identifica mediante un URL. Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento, etc.

HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. El desarrollo de aplicaciones Web necesita frecuentemente mantener estado. Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente. Esto le permite a las aplicaciones Web instituir la noción de "sesión", y también permite rastrear usuarios ya que las cookies pueden guardarse en el cliente por tiempo indeterminado. Una transacción HTTP consiste de un encabezado seguido, opcionalmente, por una línea en blanco y algún dato. El encabezado especificará cosas como la acción requerida del servidor, o el tipo de dato retornado, o el código de estado. El uso de campos de encabezados enviados en las transacciones HTTP le da gran flexibilidad al protocolo. Estos campos permiten que se envíe información descriptiva en la transacción, permitiendo así la autenticación, cifrado e identificación de usuario.
Un encabezado es un bloque de datos que precede a la información propiamente dicha, por lo que muchas veces se hace referencia a él como metadato, porque tiene datos sobre los datos. Si se reciben líneas de encabezado del cliente, el servidor las coloca en las variables de ambiente de CGI con el prefijo HTTP_ seguido del nombre del encabezado. Cualquier caracter guión (-) del nombre del encabezado se convierte a caracteres "_". El servidor puede excluir cualquier encabezado que ya esté procesado, como Authorization, Content-type y Content-length. El servidor puede elegir

excluir alguno o todos los encabezados si incluirlos exceden algún límite del ambiente de sistema. Ejemplos de esto son las variables HTTP_ACCEPT y HTTP_USER_AGENT.

HTTP_ACCEPT. Los tipos MIME (estándar que permite enviar ficheros de cualquier tipo adjuntos a un mensaje de texto a través de internet), que el cliente aceptará, dado los encabezados HTTP. Otros protocolos quizás necesiten obtener esta información de otro lugar. Los items en esta lista deben estar separados por una coma, como lo dice la especificación HTTP: tipo, tipo HTTP_USER_AGENT El navegador que utiliza el cliente para enviar el pedido. El formato general para esta variable es: software/versión librería/versión El servidor envía al cliente: Un código de estado que indica si el pedido fue exitoso o no. Los códigos de error típicos indican que el archivo solicitado no se encontró, que el pedido no está en forma o que se requiere autenticación para acceder al archivo.

La información propiamente dicha. Como HTTP permite enviar documentos de todo tipo y formato, es ideal para transmitir multimedia, como gráficos, audio y video. Esta libertad es una de las mayores ventajas de HTTP. También envía información sobre el objeto que se retorna. Confirmando que la lista no es una lista completa de los campos de encabezado y que algunos de ellos sólo tienen sentido en una dirección.

I. 4. LOS SISTEMAS DE APLICACIÓN.

El sistema operativo existe de manera predominante para beneficio de la computadora. Se requieren otros programas para hacer que la computadora sea útil para las personas. Los programas que ayudan a la gente a realizar tareas específicas se denominan sistemas de aplicación. Posee ciertas características que le diferencia de un sistema operativo (que hace funcionar la computadora), de una utilidad (que realiza tareas de mantenimiento o de uso general) y de un lenguaje (con el cual se crean los programas informáticos). Suele resultar una solución informática para la automatización de ciertas tareas complicadas como puede ser la contabilidad o la gestión de un almacén. Ciertas aplicaciones desarrolladas, a medida, suelen ofrecer una gran potencia ya que están exclusivamente diseñadas para resolver un problema específico. Otros, llamados paquetes integrados de software, ofrecen menos potencia pero a cambio incluyen varias aplicaciones, como un programa procesador de textos, de hoja de cálculo y de base de datos.

En la actualidad las herramientas informáticas se han vuelto un factor determinante para las organizaciones que deseen tener un control total de sus procesos o etapas por las cuales atraviesa cotidianamente y que sin un sistema administrativo seria muy complicado o casi imposible de controlar.

I. 4. A. SAE.

Aspel-SAE es el Sistema Administrativo Empresarial que controla el ciclo de todas las operaciones de compra-venta de la empresa en forma segura, confiable y de acuerdo con la legislación vigente; proporciona herramientas de vanguardia tecnológica que permiten una administración y comercialización eficientes. La integración de sus módulos (clientes, facturación, vendedores, cuentas por cobrar, compras, proveedores, cuentas por pagar y estadísticas) asegura que la información se encuentre actualizada en todo momento. Genera reportes, estadísticas y gráficas de

alto nivel e interactúa con los demás sistemas de la línea Aspel para lograr una completa integración de procesos.

Esta versión de Aspel-SAE presenta opciones novedosas que permiten incorporar en la administración de las empresas tanto funciones que fortalecen los procesos de atención y seguimiento comercial de los clientes CRM (Customer relationship management: Gestión de las Relaciones con los Cliente) como elementos tecnológicos de actualidad (factura electrónica). Asimismo, se robustecen múltiples aspectos de control y operación cotidiana en todos los módulos del sistema.

Aspel-SAE te facilita el cumplimiento de la Declaración Informativa de Operaciones con Terceros IVA (Impuesto al Valor Agregado) generando la bitácora con la información de los pagos a proveedores e impuestos relacionados, también proporciona el archivo de texto con la estructura requerida por el SAT lista para realizar la carga batch.

Aspel-SAE tiene interfase con: COI.

I. 4. B. COI.

Procesa, integra y mantiene actualizada la información contable y fiscal de la empresa, en forma segura y confiable. Proporciona diversos reportes y gráficas que permiten evaluar el estado financiero de la organización así como generar oportunamente las diferentes declaraciones fiscales e informativas. Calcula la depreciación de los activos fijos. Mantiene interfases con los demás sistemas Aspel e interactúa con hojas de cálculo, lo que contribuye a lograr una eficiente administración de la empresa.

Aspel-COI tiene interfase con: NOI.

I. 4. C. NOI.

Aspel-NOI 4.5 Automatiza el control de todos los aspectos de la nómina, considerando la legislación fiscal y laboral vigente incluyendo los cálculos de impuesto local y retención de ISR (Impuesto Sobre la Renta), de acuerdo a la nueva Reforma fiscal 2008.

Es la mejor solución para las micro, pequeñas y medianas empresas, ya que automatiza el control de todos los aspectos involucrados en la nómina de la empresa, lo hace considerando la legislación mexicana, de tal manera que el Usuario puede tener plena confianza en que los cálculos que el sistema realiza los hace apegados a las diferentes leyes, como la Ley del Impuesto Sobre la Renta, la Ley del IMSS (Instituto Mexicano del Seguro Social), la Ley Federal del Trabajo, entre otras.

Desde Aspel-NOI 4.5 es posible enviar los movimientos afiliatorios al lMSS vía internet (IDSE) firmados con el certificado digital sin necesidad de otras aplicaciones, además cuenta con una Bitácora para el registro automatizado de todos los movimientos con acuses de recibo y acuses de respuesta teniendo en todo momento el control en un solo sistema.

CAPÍTULO II.

BASES DE DATOS Y LA PROGRAMACIÓN

II. 1. ANTECEDENTES.

Las bases de datos tienen su origen en el desarrollo de la eficiencia de los sistemas ficheros, dichos sistemas surgieron al tratar de informatizar el manejo de los archivadores manuales con objeto de proporcionar un acceso más eficiente a los datos.

Se dice que los sistemas de bases de datos tienen sus raíces en el proyecto estadounidense Apolo, el cual mandó al hombre a la luna, en los años sesenta del siglo XX. En aquella época, no había ningún sistema que permitiera gestionar la inmensa cantidad de información que requería el proyecto. La primera empresa encargada del proyecto, NAA (North American Aviation: Aviación Norte Americana), desarrolló un software denominado GUAM (General Update Access Method: Método de Acceso Actualizado) que estaba basado en el concepto de que varias piezas pequeñas se unen para formar una pieza más grande, y así sucesivamente hasta que el producto final estuviere ensamblado. Esta estructura, que tiene la forma de un árbol, es lo que se denomina una estructura jerárquica.

FIGURA II. 1.
Hombre en la luna.

A mediados de los sesenta del siglo pasado, IBM se unió a NAA para desarrollar GUAM en lo que ahora se conoce como IMS (Information Management System: Sistema de Gestión de Información). El motivo por el cual IBM restringió al IMS en el manejo de jerarquías de registros, fue el permitir el uso de dispositivos de almacenamiento en serie, más exactamente las cintas magnéticas, ya que era un requisito del mercado por aquella época.

A mitad de los sesentas del siglo anterior, se desarrolló IDS (Integrated Data Store: Almacén de Datos Integrados), de General Electric. Este trabajo fue dirigido por uno de los pioneros en los sistemas de bases de datos, Charles Bachmann. IDS era un nuevo tipo de sistema de bases de datos conocido como sistema de red, que produjo un gran efecto sobre los sistemas de información de aquella generación. El sistema de red se desarrolló, por una parte, para satisfacer la necesidad de representar relaciones entre datos más complejas que las que se podían modelar con los sistemas jerárquicos, y en otra parte, para imponer un estándar de bases de datos.

Para ayudar a establecer dicho estándar, CODASYL (Conference on Data Systems Languages: Consulta en los Lenguajes de Sistemas de Datos), formado por representantes del gobierno de EUA y representantes del mundo empresarial, crearon un grupo denominado DBTG (Data Base Task Group: Grupo de Tareas de Bases de Datos), cuyo objetivo era definir especificaciones estándar que permitieran la creación de bases de datos y el manejo de los datos. El DBTG presentó su informe final en 1971, y aunque éste no fue formalmente aceptado por ANSI (American National Standards Institute: Instituto de Estándares Nacional Americano), muchos sistemas se desarrollaron siguiendo la propuesta del DBTG. Estos sistemas son los que se conocen como sistemas de red, o sistemas CODASYL o DBTG.

Los sistemas jerárquico y de red constituyen la primera generación de los SGBD (Sistema de gestión de base de datos) o en inglés Database management system (DBMS). Pero estos sistemas presentan algunos inconvenientes, como:

- Es necesario escribir complejos programas de aplicación para responder a cualquier tipo de consulta de datos, por simple que ésta sea.

- La independencia de datos es mínima.

- No tienen un fundamento teórico.

En 1970, Codd, de los laboratorios de investigación de IBM, escribió un artículo presentando el modelo relacional. En este artículo, presentaba también los inconvenientes de los sistemas previos, el jerárquico y el de red. Entonces, se comenzaron a desarrollar muchos sistemas relacionales, apareciendo los primeros a finales de los setenta y principios de los ochenta del siglo pasado. Uno de los primeros es System R, de IBM, que se desarrolló para probar la funcionalidad del modelo relacional, proporcionando una implementación de sus estructuras de datos y sus operaciones. Esto condujo a dos grandes desarrollos:

- El primero fue, el desarrollo de un lenguaje de consultas estructurado denominado SQL, Structured Query Language o Lenguaje Estructurado de Consultas, que se ha convertido en el lenguaje estándar de los sistemas relacionales.

- Y el segundo, la producción de varios SGBD relacionales durante la siguiente década (80's), como DB2 y SLQ/DS de IBM, y ORACLE de ORACLE Corporation.

Hoy en día, existen cientos de SGBD relacionales, tanto para microcomputadoras como para sistemas multiusuario, aunque muchos no son completamente fieles al modelo relacional.

Otros sistemas relacionales multiusuario son INGRES de Computer Associates, Informix de Informix Software Inc. y Sybase de Sybase Inc. Ejemplos de sistemas relacionales de microcomputadoras son Paradox y dBase IV de Borland, Access de Microsoft, FoxPro y R:base de Microrim.

Los SGBD relacionales constituyen la segunda generación de los SGBD. Sin embargo, el modelo relacional también tiene sus fallas, siendo uno de ellos su limitada capacidad al modelar los datos. Se ha hecho mucha investigación desde entonces tratando de resolver este problema.

En 1976, Chen, presentó el modelo entidad-relación, que es la técnica más utilizada en el diseño de bases de datos. En 1979, Codd, intentó subsanar algunas de las deficiencias de su modelo relacional con una versión extendida denominada RM/T (1979) y más recientemente RM/V2 (1990). Los intentos de proporcionar un modelo de datos que represente al mundo real de un modo más fiel han dado lugar a los modelos de datos semánticos.

Como respuesta a la creciente complejidad de las aplicaciones que requieren las bases de datos, han surgido dos nuevos modelos: el modelo de datos orientado a objetos y el modelo relacional extendido. Sin embargo, a diferencia de los modelos que los preceden, la composición de estos modelos no está clara. Esta evolución representa la tercera generación de los SGBD.

II. 2. BASE DE DATOS.

Una base de datos es un almacén para colecciones de datos o hechos relacionados, ordenados en una estructura específica. En una base de datos computarizada, por lo general se introducen datos, en una tala bidimensional que consiste de columnas y filas, similares a la estructura de una hoja de cálculo. Las bases de datos se usan a través de los SGBD.

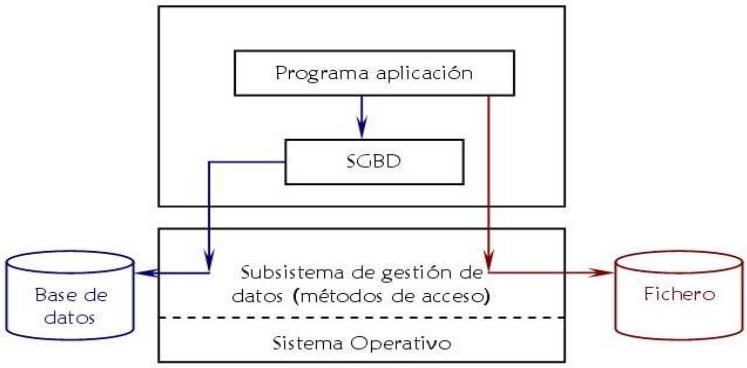


FIGURA II. 2.
SGBD (Sistema de gestión de base de datos).

Un archivo es un elemento de información conformado por un conjunto de registros. Estos registros a su vez están compuestos por una serie de caracteres o bytes. Algunas organizaciones están utilizan bases de datos para generar resultados o para compartir dicha información con otros sistemas. Sin embargo, los principales componentes de las bases de datos son los archivos, los cuales están formados por:

- **Registro:** Es una colección de campos (atributos).Un registro, es el conjunto de información referida a una misma persona u objeto. Un registro vendría a ser algo así como una ficha.

- **Campo:** Unidad básica de una base de datos. Un campo puede ser, por ejemplo, el nombre de una persona. Los nombres de los campos, no pueden empezar con espacios en blanco y caracteres especiales. No pueden llevar puntos, ni signos de exclamación o corchetes. Si pueden tener espacios en blanco en medio. La descripción de un campo, permite aclarar información referida a los nombres del campo. El tipo de campo, permite especificar el tipo de información que cargáremos en dicho campo, esta puede ser:

  - **Texto:** para introducir cadenas de caracteres hasta un máximo de 255.
  - **Memo:** para introducir un texto extenso. Hasta 65.535 caracteres.
  - **Numérico:** para introducir números.
  - **Fecha / Hora**: para introducir datos en formato fecha u hora.
  - **Moneda:** para introducir datos en formato número y con el signo monetario.
  - **Autonúmerico:** en este tipo de campo, se numera automáticamente el contenido.
  - **Si / No:** campo lógico. Este tipo de campo es sólo si queremos un contenido del tipo Si / No, verdadero / Falso, etc.
  - **Objeto OLE:** para introducir una foto, gráfico, hoja de cálculo, sonido, etc.
  - **Hipervínculo:** podemos definir un enlace a una página Web.
  - **Asistente para búsquedas:** crea un campo que permite elegir un valor de otra tabla o de una lista de valores mediante un cuadro de lista o un cuadro combinado.

Las formas en las cuales pueden organizarse los archivos, son archivos secuenciales o archivos directos. En los archivos secuenciales los registros están almacenados en una secuencia que dependen de algún criterio definido.

Cada fila representa un registro, el cual es un conjunto de datos para cada entrada en la base de datos. Cada columna de la tabla representa un campo, el cual agrupa cada pieza o elemento de los datos entre los registros en categorías o tipos específicos de datos.

Puede la tabla organizar los datos de cada registro por el mismo conjunto de campos, pero almacenar cualquier cantidad de registros. El único límite es la capacidad de almacenamiento del disco duro. Cualquier registro en la tabla no necesariamente tiene datos en todos los campos. Sin embargo, para que exista un registro, debe tener datos al menos en un campo.

El orden de los campos en una tabla define en forma escrita la ubicación de los datos en cada registro. El conjunto de campos en cualquier tabla proporciona una definición sensible de la base de datos para aquellos que pueden tener acceso a sus datos.

II. 2. A. TIPOS DE BASES DE DATOS.

Las aplicaciones más usuales son para la gestión de empresas e instituciones públicas. También son ampliamente utilizadas en entornos científicos con el objeto de almacenar la información experimental.

Las bases de datos pueden clasificarse en dos criterios, por la variabilidad de los datos almacenados y por su contenido.

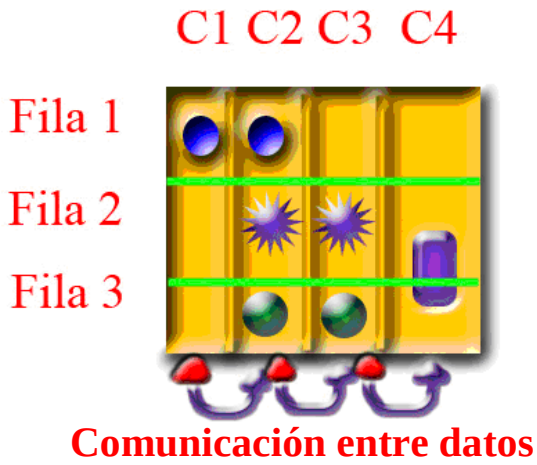


FIGURA II. 3.
Base de Datos.

II. 2. A. a. VARIABILIDAD DE CONTENIDO.

Las variables de contenido se dividen en dos grupos: estáticas y dinámicas.

- **Bases de datos estáticas.**

  Son de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

- **Bases de datos dinámicas.**

  En donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede ser la base de datos utilizada en un sistema de información de una tienda de abarrotes, una farmacia, un salón de eventos sociales, etc.

II. 2. A. b. CONTENIDO.

Existen diferentes clasificaciones según su utilidad y datos que contengan.

- **Bases de datos bibliográficas.**

  Solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque sino estaríamos en presencia de una base de datos a texto completo o de fuentes primarias. Como su nombre lo indica, el contenido son cifras o números.

- **Bases de datos de texto completo.**

  Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

II. 2. B. MODELOS DE BASES DE DATOS.

Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos.

Un modelo de datos es básicamente una "descripción" de algo conocido como contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores. Los modelos de datos no son cosas físicas: son abstracciones que permiten la implementación de un sistema eficiente de base de datos; por lo general se refieren a algoritmos, y conceptos matemáticos.

II. 2. B. a. MODELO JERÁRQUICO.

En una base de datos jerárquica, los registros se organizan en una estructura tipo árbol. Se dice que la relación entre tipos de registros es una relación de padre a hijo, en la que cualquier tipo de hijo se relaciona sólo con un tipo de padre único.

Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento.

Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

II. 2. B. b. MODELO DE RED.

La base de datos en red es similar a la estructura jerárquica excepción que cualquier tipo de registro puede relacionarse con cualquier número de otros tipos de registros. Como la estructura jerárquica, la estructura de base de datos en red se usa en sistema antiguo, sobre todo mainframes.

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; pero, aun así, la dificultad que significa administrar la información en una base de datos de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales.

II. 2. B. c. MODELO ORIENTADO A OBJETOS.

Una base de datos orientada a objetos es una estructura que en sus inicios generó un gran interés. Esta estructura agrupa elementos de datos y sus características, atributos y procedimientos asociados, en elementos complejos llamados objetos.

Desde el punto de vista físico, un objeto puede ser cualquier cosa: un producto, un evento, una casa, un aparato, un tejido, una obra de arte, un juguete, un cliente o incluso compra. Un objeto se define por sus características, atributos y procedimientos. Las características de un objeto pueden ser texto, sonido, gráficos y video. Los atributos podrían ser el color, el tamaño, el estilo, la cantidad y el precio. Un procedimiento se refiere al procesamiento o manejo que puede estar asociado con un objeto:

- **Encapsulación:** Propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.

- **Herencia:** Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.

- **Polimorfismo:** Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos. Una operación (llamada función) se especifica en dos partes. La interfaz (o signatura) de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz.

II. 2. B. d. MODELO RELACIONAL.

Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente.

Su idea fundamental es el uso de relaciones. Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados tuplas. Pese a que esta es la teoría de las bases de datos relacionales creadas por Edgar Frank Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar. Esto es, pensando en cada relación como si fuese una tabla que esta compuesta por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla).

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario casual de la base de datos. La información puede ser recuperada o almacenada por medio de consultas que ofrecen una amplia flexibilidad y poder para administrar la información. El lenguaje más común para construir las consultas a bases de datos relacionales es SQL, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

Este modelo considera la base de datos como una colección de relaciones. De manera simple, una relación representa una tabla, en que cada fila representa una colección de valores que describen una entidad del mundo real. Cada fila se denomina tupla o registro y cada columna campo.

Una relación consiste en:

- **Esquemas.**

  - Nombre de la relación.
  - Nombre de los atributos y sus dominios:
    - El dominio se establece por nombres como character, integer, date, etc.
    - Un dominio tiene asociado un conjunto de valores homogéneos.
    - Los atributos deben tomar valores dentro del domino asignado.

- **Instancias.**

  - Caracteres y números.
  - Registro o columna.
  - Conjunto de tuplas.
  - Tabla con filas y columnas.
    - Cada fila es una tupla. El número de filas es llamado cardinalidad.
    - El numero de columnas es llamado aridad o grado.

II. 2. B. e. OTROS.

- **Bases de datos documentales.**

  Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes. Tesaurus es un sistema de índices optimizado para este tipo de bases de datos.

- **Base de datos deductivas.**

  Un sistema de base de datos deductivas, es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. También las bases de datos deductivas son llamadas base de datos lógica, a raíz de que se basan en lógica matemática.

- **Gestión de bases de datos distribuida.**

  La base de datos está almacenada en varias computadoras conectadas en red. Surgen debido a la existencia física de organismos descentralizados. Esto les da la capacidad de unir las bases de datos de cada localidad y acceder así a distintas universidades, sucursales de tiendas, etcétera.

II. 3. MySQL.

MySQL surge de la necesidad de mayor rapidez y flexibilidad que SQL podía ofrecer, por lo que se creo una nueva interfaz a SQL, además de ser uno de los sistemas más estables.

MySQL es el sistema de gestión más popular de bases de datos, se dice que es un sistema SQL open source, ya que es posible para cualquiera usar y modificar el software, además este sistema permite trabajar las bases de datos en modo relacional.

II. 3. A. CARACTERÍSTICAS.

MySQL tiene varias características que son interioridades y portabilidad, seguridad, escalabilidad, conectividad, localización, modos, organización de los datos y funciones. Las cuales se describirán brevemente cada una de ellas.

- **Interioridades y portabilidad.**

  Esta escrito en los Lenguajes C y C++. Funciona en diferentes plataformas. Puede utilizarse en multiples CPU's si estas están disponibles. Proporciona sistemas de almacenamientos transaccionales y no transaccionales. Las funciones SQL están implementadas usando una librería altamente optimizada y deben ser tan rápidas como sea posible. Normalmente no hay reserva de memoria tras toda la inicialización para consultas. El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor. También está disponible como biblioteca y puede ser incrustado (linkeado) en aplicaciones autónomas. Dichas aplicaciones pueden usarse por sí mismas o en entornos donde no hay red disponible.

- **Seguridad.**

  Tiene un sistema de privilegios y contraseñas que es muy flexible y seguro, permite verificación basada en el host. Las contraseñas son seguras porque todo el tráfico de contraseñas está encriptado cuando se conecta con un servidor.

- **Escalabilidad.**

  Soporte a grandes bases de datos, se permiten hasta 64 índices por tabla, cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas, el máximo ancho de límite son 1000 bytes. El tamaño efectivo máximo para las bases de datos en MySQL usualmente los determinan los límites de tamaño de ficheros del sistema operativo, y no por límites internos de MySQL.

- **Conectividad.**

  Los clientes pueden conectar con el servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows de la familia NT (NT, 2000, XP, 2003, Vista y Seven), los clientes pueden usar named pipes para la conexión. En sistemas Unix, los clientes pueden conectar usando ficheros socket Unix. La interfaz para el conector ODBC (MyODBC) proporciona a MySQL soporte para programas clientes que usen conexiones ODBC (Open Database Connectivity: Conexión Abierta a Bases de Datos).

- **Localización.**

  El servidor puede proporcionar mensajes de error a los clientes en muchos idiomas. Todos los datos se guardan en el conjunto de caracteres elegido. Todas las comparaciones para columnas normales de cadenas de caracteres son case-insensitive. Los desarrolladores pueden añadir MySQL Server en varias aplicaciones y dispositivos electrónicos, donde el usuario final no tiene conocimiento que hay una base de datos subyacente. MySQL Server incrustado es ideal para uso tras aplicaciones en Internet, kioskos públicos, combinación de hardware/software en llaveros, servidores de alto rendimiento de Internet, bases de datos autocontenidas distribuidas en CD-ROM.

- **Modos.**

  MySQL Server puede operar en distintos modos SQL y puede aplicar dichos modos de forma diferente para distintos clientes. Esto permite a una aplicación adaptar el funcionamiento del servidor a sus propios requerimientos. Los modos definen la sintaxis que MySQL debe soportar y qué clase de validaciones debe efectuar a los datos. Esto hace más fácil usar MySQL en un conjunto de entornos diferentes y usar MySQL junto con otros servidores de bases de datos.

- **Organización de los datos y funciones.**

  MySQL Server mapea cada base de datos a un directorio bajo el directorio de datos de MySQL, y las tablas dentro de cada directorio como ficheros. Para facilitar a los usuarios que vienen de otros entornos SQL, MySQL Server soporta alias para varias funciones. Todas las comparaciones de cadenas de caracteres son case-insensitive por defecto, con la ordenación determinada por el conjunto de caracteres actual.

II. 3. B. MOTORES DE ALMACENAMIENTO DE MySQL.

MySQL soporta varios motores de almacenamiento que tratan con distintos tipos de tabla. Los motores de almacenamiento de MySQL incluyen algunos que tratan con tablas transaccionales y otros que no lo hacen:

- MyISAM trata tablas no transaccionales. Proporciona almacenamiento y recuperación de datos rápida, así como posibilidad de búsquedas fulltext. MyISAM se soporta en todas las configuraciones MySQL, y es el motor de almacenamiento por defecto a no ser que tenga una configuración distinta a la que viene por defecto con MySQL.

- Los motores de almacenamiento InnoDB y BDB proporcionan tablas transaccionales. BDB se incluye en la distribución binaria MySQL-Max en aquellos sistemas operativos que la soportan. InnoDB también se incluye por defecto en todas las distribuciones binarias de MySQL 5.0. En distribuciones fuente, puede activar o desactivar estos motores de almacenamiento configurando MySQL a su gusto.

Cuando se instala MySQL en Windows usando el MySQL Configuration Wizard, InnoDB es el motor de almacenamiento por defecto en lugar de MyISAM.

Si trata de usar un motor de almacenamiento que no está compilado o que está desactivado, MySQL crea una tabla de tipo MyISAM. Este comportamiento es conveniente cuando quiere copiar tablas entre servidores MySQL que soportan distintos motores. (Por ejemplo, en una inicialización de replicación, tal vez su maestro suporte un motor de almacenamiento transaccional para más seguridad, pero los esclavos usan un motor de almacenamiento no transaccional para mayor velocidad).

La sustitución automática del tipo MyISAM cuando se especifica un tipo no especificado puede ser confuso para nuevos usuarios. En MySQL, se genera una advertencia cuando se cambia un tipo de tabla automáticamente.

Los motores de almacenamiento individuales crean los ficheros adicionales necesarios para las tablas que administran.

II. 3. C. REGLAS DE CODD.

En 1985 el Dr. Edgar Frank Codd publicó 12 reglas para evaluar si un DBMS (DataBase Management System: Sistema de Gestión de Base de Datos) puede considerarse un RDBMS (Relational DataBase Management System: Sistema de Gestión de Base de Datos Relacional), o dicho más concisamente, si un sistema de bases de datos puede considerarse o no relacional.

- **Regla 0.**

  Para que un sistema se denomine sistema de gestión de bases de datos relacionales, este sistema debe usar (exclusivamente) sus capacidades relacionales para gestionar la base de datos.

- **Regla 1: regla de la información.**

  Toda la información en una base de datos relacional se representa explícitamente en el nivel lógico exactamente de una manera: con valores en tablas.

  - Por tanto los metadatos (diccionario, catálogo) se representan exactamente igual que los datos de usuario.
  - Y puede usarse el mismo lenguaje para acceder a los datos y a los metadatos (regla 4)
  - Un valor posible es el valor nulo, con sus dos interpretaciones:
    - Valor desconocido.
    - Valor no aplicable.

- **Regla 2: regla del acceso garantizado.**

  Para todos y cada uno de los datos (valores atómicos) de una Base de Datos Relacional (BDR) se garantiza que son accesibles a nivel lógico utilizando una combinación de nombre de tabla, valor de clave primaria y nombre de columna.

  - Cualquier dato almacenado en una BDR tiene que ser direccionado unívocamente. Para ello hay que indicar en qué tabla está, cuál es la columna y cuál es la fila (mediante la clave primaria).
  - Por tanto se necesita el concepto de clave primaria, que no es soportado en muchas implementaciones. En estos casos, para lograr un efecto similar se puede hacer lo siguiente:
    - Hacer que los atributos clave primaria no puedan ser nulos (NOT NULL).
    - Crear un índice único sobre la clave primaria.
    - No eliminar nunca el índice.

- **Regla 3: tratamiento sistemático de valores nulos.**

  Los valores nulos (que son distintos de la cadena vacía, blancos, 0,...) se soportan en los SGBD totalmente relacionales para representar información desconocida o no aplicable de manera sistemática, independientemente del tipo de datos.

  - Se reconoce la necesidad de la existencia de valores nulos, para un tratamiento sistemático de los mismos.
  - Hay problemas para soportar los valores nulos en las operaciones relacionales, especialmente en las operaciones lógicas.
  - Lógica trivaluada. En una posible solución. Existen tres valores de verdad: Verdadero, Falso y Desconocido (null). Se crean tablas de verdad para las operaciones lógicas:
    - null Y null = null.
    - Verdadero Y null = null.
    - Falso Y null = Falso.
    - Verdadero O null = Verdadero.
    - etc.

  Un inconveniente es que de cara al usuario el manejo de los lenguajes relacionales se complica pues es más difícil de entender.

- **Regla 4: diccionario dinámico en línea basado en el modelo relacional.**

  La descripción de la base de datos se representa a nivel lógico de la misma manera que los datos normales, de modo que los usuarios autorizados pueden aplicar el mismo lenguaje relacional a su consulta, igual que lo aplican a los datos normales.

  - Es una consecuencia de la regla 1 que se destaca por su importancia. Los metadatos se almacenan usando el modelo relacional, con todas las consecuencias. Nombre de los atributos y sus dominios.

- **Regla 5: regla del sublenguaje de datos completo.**

  Un sistema relacional debe soportar varios lenguajes y varios modos de uso de terminal (ej: rellenar formularios, etc.). Sin embargo, debe existir al menos un lenguaje cuyas sentencias sean expresables, mediante una sintaxis bien definida, como cadenas de caracteres y que sea completo, soportando:

  - Definición de datos.
  - Definición de vistas.
  - Manipulación de datos (interactiva y por programa).
  - Limitantes de integridad.
  - Limitantes de transacción (iniciar, realizar, deshacer) (Begin, commit, rollback).
  - Además de poder tener interfaces más amigables para hacer consultas, etc. siempre debe de haber una manera de hacerlo todo de manera textual, que es tanto como decir que pueda ser incorporada en un programa tradicional.
  - Un lenguaje que cumple esto en gran medida es SQL.

- **Regla 6: regla de actualización de vistas.**

  Todas las vistas que son teóricamente actualizables se pueden actualizar por el sistema.

  - El problema es determinar cuáles son las vistas teóricamente actualizables, ya que no está muy claro.
  - Cada sistema puede hacer unas suposiciones particulares sobre las vistas que son actualizables.

- **Regla 7: inserción, actualización y borrado de alto nivel.**

  La capacidad de manejar una relación base o derivada como un solo operando se aplica no sólo a la recuperación de los datos (consultas), si no también a la inserción, actualización y borrado de datos.

  - Esto es, el lenguaje de manejo de datos también debe ser de alto nivel (de conjuntos). Algunas bases de datos inicialmente sólo podían modificar las tuplas de la base de datos de una en una (un registro de cada vez).

- **Regla 8: independencia física de datos.**

  Los programas de aplicación y actividades del terminal permanecen inalterados a nivel físico cuando quiera que se realicen cambios en las representaciones de almacenamiento o métodos de acceso.

  - El modelo relacional es un modelo lógico de datos, y oculta las características de su representación física.
  - Es la capacidad de modificar el esquema interno sin tener que alterar el esquema conceptual (o los externos).

- **Regla 9: independencia lógica de datos.**

  Los programas de aplicación y actividades del terminal permanecen inalterados a nivel lógico cuando quiera que se realicen cambios a las tablas base que preserven la información.

  - Cuando se modifica el esquema lógico preservando información (no valdría p.ej. eliminar un atributo) no es necesario modificar nada en niveles superiores.
  - Ejemplos de cambios que preservan la información:
    - Añadir un atributo a una tabla base.
    - Sustituir dos tablas base por la unión de las mismas. Usando vistas de la unión puedo recrear las tablas anteriores.

- **Regla 10: independencia de integridad.**

  Los limitantes de integridad específicos para una determinada base de datos relacional deben poder ser definidos en el sublenguaje de datos relacional, y almacenables en el catálogo, no en los programas de aplicación.

  - El objetivo de las bases de datos no es sólo almacenar los datos, si no también sus relaciones y evitar que estas (limitantes) se codifiquen en los programas. Por tanto en una BDR se deben poder definir limitantes de integridad.
  - Cada vez se van ampliando más los tipos de limitantes de integridad que se pueden utilizar en los RDBMS (Relational Database Management System: Sistema de Gestión de Base de Datos Relacional o SGBDR).
  - Como parte de los limitantes inherentes al modelo relacional (forman parte de su definición) están:
    - Una BDR tiene integridad de entidad. Es decir, toda tabla debe tener una clave primaria.
    - Una BDR tiene integridad referencial. Es decir, toda clave externa no nula debe existir en la relación donde es primaria.

- **Regla 11: independencia de distribución.**

  Una BDR tiene independencia de distribución.

  - Las mismas órdenes y programas se ejecutan igual en una BD (Base de Datos) centralizada que en una distribuida.

- Las BDR son fácilmente distribuibles:
  - Las tablas se dividen en fragmentos que se distribuyen.
  - Cuando se necesitan las tablas completas se recombinan usando operaciones relacionales con los fragmentos.
- Sin embargo se complica más la gestión interna de la integridad, etc.
- Esta regla es responsable de tres tipos de transparencia de distribución:
  - Transparencia de localización. El usuario tiene la impresión de que trabaja con una BD local. (aspecto de la regla de independencia física).
  - Transparencia de fragmentación. El usuario no se da cuenta de que la relación con que trabaja está fragmentada. (aspecto de la regla de independencia lógica de datos).
  - Transparencia de replicación. El usuario no se da cuenta de que pueden existir copias (réplicas) de una misma relación en diferentes lugares.

➢ **Regla 12: regla de la no subversión.**

Si un sistema relacional tiene un lenguaje de bajo nivel (un registro de cada vez), ese bajo nivel no puede ser usado para saltarse (subvertir) las reglas de integridad y los limitantes expresados en los lenguajes relacionales de más alto nivel (una relación (conjunto de registros) de cada vez).

- Algunos problemas no se pueden solucionar directamente con el lenguaje de alto nivel.
- Normalmente se usa SQL inmerso en un lenguaje anfitrión para solucionar estos problemas. Se utiliza el concepto de cursor para tratar individualmente las tuplas de una relación. En cualquier caso no debe ser posible saltarse los limitantes de integridad impuestos al tratar las tuplas a ese nivel.

II. 3. D. TRANSACCIONES Y OPERACIONES ATÓMICAS.

MySQL Server soportan transacciones con los motores trasaccionales InnoDB y BDB. Los otros motores no transaccionales en MySQL Server (como MyISAM) siguen un paradigma diferente para integridad de datos llamado "operaciones atómicas".

MySQL Server soporta ambos paradigmas, puede decidir si su aplicación necesita la velocidad de operaciones atómicas o el uso de características transaccionales. Esta elección puede hacerse para cada tabla.

Como se ha dicho, el compromiso entre tipos de tablas transaccionales y no transaccionales reside principalmente en el rendimiento. Tablas transaccionales tienen requerimientos significativamente mayores para memoria y espacio de disco, y mayor carga de CPU. Por otra parte, tipos de tablas transaccionales como InnoDB también ofrece muchas características significativas. El diseño modular de MySQL Server permite el uso concurrente de distintos motores de almacenamiento para cumplir distintos requerimientos y mostrarse óptimo en todas las situaciones.

En las situaciones en las que la integridad es de máxima importancia, MySQL Server ofrece integridad a nivel de transacción incluso para tablas no transaccionales.

Atómico, en el sentido en que nos referimos, no es nada mágico. Se trata que puede asegurar que mientras cada actualización específica está ejecutándose, ningún otro usuario puede interferir con

ellas, y que nunca puede haber un rollback automático (lo que puede ocurrir con tablas transaccionales si no se es muy cuidadoso). MySQL Server garantiza que no hay dirty reads (lecturas sucias).

Una base de datos puede contener tablas de distintos tipos.

Las tablas transaccionales (TSTs) tienen varias ventajas sobre las no transaccionales (NTSTs):

- Más seguras. Incluso si MySQL cae o tiene problemas de hardware, puede recuperar los datos, mediante recuperación automática o desde una copia de seguridad más el log de transacciones.
- Puede combinar varios comandos y aceptarlos todos al mismo tiempo con el comando COMMIT (si autocommit está desactivado).
- Puede ejecutar ROLLBACK para ignorar los cambios (si autocommit está desactivado).
- Si falla una actualización, todos los cambios se deshacen. (Con tablas no transaccionales, todos los cambios son permanentes).
- Motores de almacenamiento transaccionales pueden proporcionar mejor concurrencia para tablas que tienen varias actualizaciones concurrentes con lecturas.

Tablas no transaccionales tienen varias ventajas al no tener una sobrecarga transaccional:

- Más rápidas.
- Menor requerimiento de espacio.
- Menos memoria para actualizaciones.

Puede combinar tablas transaccionales y no transaccionales en el mismo comando para obtener lo mejor de ambos mundos. Sin embargo, en una transacción con autocommit desactivado, los cambios de tablas no transaccionales son permanentes inmediatamente y no pueden deshacerse.

II. 4. LA PROGRAMACIÓN.

Las guerras mundiales que azotaron a la humanidad produjeron todo tipo de avances tecnológicos, los cuales la industria aprovecho. Las necesidades que la guerra planteaba pasaron de curiosidades científicas a necesidades de Estado, una de ellas fue el cálculo de tablas balísticas, así como los cálculos indispensables para diseñar nuevo armamento.

El gobierno de los Estados Unidos contrató a cientos de mujeres que realizaran cálculos a mano, razón por la cual se les llamaba calculadoras a estas personas, mas no se daban abasto para realizar todas las operaciones que se requerían en el frente de batalla y menos en los talleres de diseño.

Bajo estas circunstancias, un proyecto que abordara la computación de alta velocidad era de interés. En el verano de 1941 Atanasoff recibió la visita de Mauchly, mostrando sus avances en la máquina especializada; sin embargo Mauchly estaba interesado en lago aún más grande, la computación de

propósito general, la cual sería capaz de realizar cálculos a voluntad del ser humano, que permitieran aplicarlos a distintos tipos de problemas. Los mecanismos de cálculo ya estaban presentes en la computadora de Atanosoff-Berry, lo cual ayudo ampliamente estas ideas a Mauchly.

Mauchly compartió sus ideas con J. Presper Eckert Jr. de veinte años, el cual creo un dispositivo para medir el campo magnético y grabar en cinta los resultados. Eckert era un excelente ingeniero, en tanto que Mauchly tenía todo el modelo para construir una calculadora electrónica. En 1942 mauchly escribió un documento de 5 páginas titulado "The Use of Vacuum Tube Devices in Calculating" (el uso de bulbos en cálculo). Este documento se convirtió en la base del reporte enviado a la Moore School de los Laboratorios de Investigación en Balística de la Armada (Army's Ballistic Research Laboratory).

En tanto, la guerra exigía cálculos. Definir previamente las trayectoria de un proyectil, era una labor de cerca de 40 horas; cada tabla involucraba cientos de trayectorias que tardaban cerca de un mes en terminarse, Mauchly afirmó que todo podía realizarse en minutos con la fabricación de la ENIAC.

II. 4. A. ENIAC.

ENIAC es un acrónimo inglés de Electronic Numerical Integrator And Computer (Computador e Integrador Numérico Electrónico), utilizada por el Laboratorio de Investigación Balística del Ejército de los Estados Unidos.

El 5 de junio de 1942, y con el contrato No. W-670-ORD-4926, se firmó el acuerdo entre los directivos de la Universidad de Pennsylvania y el departamento de Defensa de los Estados Unidos, con John Grist Brainerd como supervisor del proyecto, John Presper Eckert como ingeniero en jefe y John William Mauchly como consultor principal.

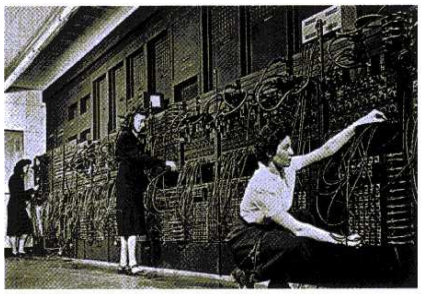


FIGURA II. 4.
ENIAC.

Se dedicó al desarrollo de un contador de década (un dispositivo que sea capaz de contar del cero al nueve). Sin embargo un reto aún mayor fue encontrar un bulbo realmente confiable. Ya que físicamente, la ENIAC tenía 17,468 bulbos o tubos de vacío en operación y funcionando a 100,000 ciclos por segundo, tiene la oportunidad de cometer 1,748,000,000 errores cada segundo. Si un bulbo fallará alteraría la trayectoria de un misil, lo cual hacía resaltar la importancia de la confiabilidad de la computadora.

Además está relacionada con el Colossus, que fue usado para descifrar código alemán durante la Segunda Guerra Mundial y destruido tras su uso para evitar dejar pruebas, siendo recientemente restaurada para un museo británico. Era totalmente digital, es decir, que ejecutaba sus procesos y operaciones mediante instrucciones en lenguaje máquina, a diferencia de otras computadoras contemporáneas de procesos analógicos.

Otras características que tenía la ENIAC era que tenía 7,200 diodos de cristal, 1.500 relés, 70.000 resistencias, 10.000 condensadores y 5 millones de soldaduras. Ocupaba una superficie de 167 m² y operaba con un total de 17.468 válvulas electrónicas o tubos de vacío. Pesaba 27 tn, medía 2,4 m x

0,9 m x 30 m; utilizaba 1.500 conmutadores electromagnéticos y relés; requería la operación manual de unos 6.000 interruptores, y su programa o software, cuando requería modificaciones, tardaba semanas de instalación manual.

La ENIAC elevaba la temperatura del local a 50ºC y fue terminada en 1944. Para efectuar las diferentes operaciones era preciso cambiar, conectar y reconectar los cables como se hacía, en esa época, en las centrales telefónicas, de allí el concepto. Este trabajo podía demorar varios días dependiendo del cálculo a realizar.

El primer trabajo que se le encomendó fue realizar millones de cálculos asociados a la construcción de la bomba de hidrógeno.

Uno de los mitos que rodea a este aparato es que la ciudad de Filadelfia, donde se encontraba instalada, sufría de apagones cuando la ENIAC entraba en funcionamiento, pues su consumo era de 160 kW. Esto no es cierto, ya que ésta tenía un sistema aparte de la red eléctrica.

A las 23.45 hr del 2 de octubre de 1955, la ENIAC fue desactivada para siempre. La computadora más grande del mundo para la 2 guerra mundial que obtuvo fuerte desequilibrio para las demás naciones.

II. 4. B. HISTORIA DE LA PROGRAMACIÓN.

Se hace mención de la ENIAC por que es un dispositivo que no es de propósito específico, por lo cual rompió con las ataduras del dispositivo a la aplicación y se volvió más general. Podía aplicarse a distintos problemas que involucraran cálculos, pero tenía un inconveniente, el cual era que las instrucciones se proporcionaban alambrándolas. Los operadores colocaban alambres físicos que indicaban a la computadora qué es lo que debía de hacer.

Von Newman, pensó que por medio de alambres no era la manera más fácil de proporcionar instrucciones a la computadora, sino que se podía almacenar en la memoria de la computadora internamente. Al modelo propuesto se le conoce como Arquitectura Von Newman y a la fecha este modelo es utilizado para construir computadoras.

Un equipo actual basado en la arquitectura de Von Newman, implica que la computadora debe tomar una a una las instrucciones. Éstas se proporcionan desde un dispositivo llamado entrada. Las instrucciones se transfieren a la memoria de la computadora, después a la unidad de control, la cual descifra cada una de las instrucciones y la ejecuta la ALU (unidad lógica aritmética), cuando así lo indique el programa, el resultado se envía a la unidad de salida.

La unidad de control es la encargada de indicar que tipo de instrucción ejecuta. Los números que significan instrucciones se encuentran almacenados en la memoria, al igual que los datos, sin embargo previamente se determina cuáles son las instrucciones que deben seguir el dispositivo y en que orden deben ser ejecutadas.

La UNIVAC I (UNIVersal Automatic Computer I, computadora automática universal I) fue la primera computadora que empleó este modelo de programas almacenados en memoria. La ENIAC leía estas instrucciones desde un tablero de control en donde estaban alambradas.

La primera computadora UNIVAC fue desarrollada para la Oficina del Censo en 1951 por los ingenieros John Mauchly y John Presper Eckert, que empezaron a diseñarla y construirla en 1946. A diferencia de la ENIAC, UNIVAC procesaba cada dígito serialmente, pero su diseño superior permitía realizar la suma de dos números de 10 dígitos a una velocidad de 100,000 sumas por segundo. Internamente funcionaba con un reloj de 2.25 Mhz de frecuencia y empleaba memoria de retraso de mercurio, estas líneas no permitían el acceso inmediato a los datos almacenados en su memoria, pero solucionaba problemas asociados con las tecnologías de tubos de rayos catódicos, que son los que usaban normalmente.

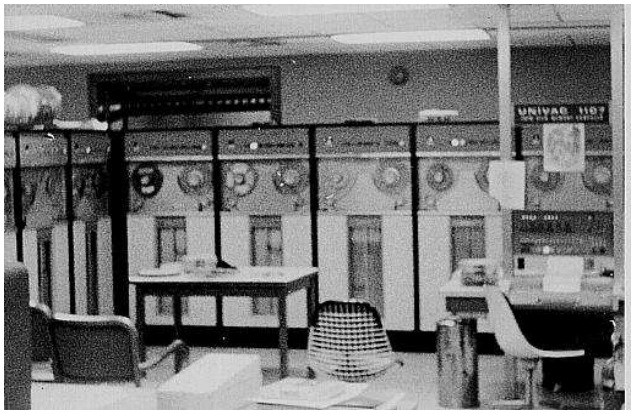


FIGURA II. 5.
UNIVAC.

Los equipos que le siguieron fueron adquiriendo mayor poderío de procesamiento, más almacenaje de memoria y velocidad de transferencia, pero el lenguaje de programación estaba atado a la forma como se construía una computadora.

Muy pronto se vio la necesidad de crear lenguajes que fueran más parecidos a un lenguaje humano, y que no cambia de una computadora a otra. Este paso fue tan importante que representó un cambio total en la forma de pensar de los programadores. Después no fue necesario aprender el lenguaje de cada modelo UNIVAC que iba saliendo, sino que bastaba con aprender FORTRAN.

La idea fue tan revolucionaria que von Newman decía que rea absurdo escribir un lenguaje sobre otro lenguaje, después se vio la necesidad de traducir de FORTRAN al lenguaje máquina para cada equipo existente. El programa encargado de realizar esta traducción se llamó compilador, el cuales diseñado para un lenguaje específico y para generar instrucciones sobre un cierto microprocesador, es un sistema operativo determinado.

El lenguaje FORTRAN era independiente del equipo y era mucho más parecido al inglés que los lenguajes que se manejaban antes. Muchos lenguajes han surgido desde entonces y cada uno propone nuevas ideas o nuevas formas de ver el problema fundamental de la programación.

II. 4. C. ¿QUÉ ES PROGRAMACIÓN?

Se tiene comunicamos con las computadoras por medio de un lenguaje. Este lenguaje suele tener características muy particulares que dependen del fabricante del equipo, del entorno de funcionamiento de la computadora o del tipo de problema que se desea resolver. Con estos lenguajes los seres humanos constituyen paquetes de instrucciones que le dicen a la computadora cómo resolver problemas. Los paquetes de instrucciones se conocen como programas.

Un programa es ejecutado instrucción por instrucción. El microprocesador de una computadora sólo puede atender una instrucción a la vez, y aunque ya existe la capacidad de atender múltiples programas de manera casi simultanea. Los programas generalmente son escritos pensando en que se ejecutarán una sola instrucción a la vez.

La programación consiste en desarrollar programas para procesar información. Se conoce como programación de computadoras a la implementación de un algoritmo en un determinado lenguaje de programación, conformando un programa. Una computadora es totalmente inútil si no dispone de un programa capaz de procesar información. Para que se realice dicho procesamiento de información habrá sido necesario construir una computadora (hardware), pensar y crear un programa (software) y ejecutar dicho programa o aplicación en el computador.

La última de estas fases es la que realiza el usuario, las anteriores son realizadas por técnicos que construyen el hardware y por programadores que desarrollan el software.La programación tiene como objetivo el tratamiento de la información correctamente, con lo que se espera que un programa de el resultado correcto y no uno erróneo. Así que cada aplicación debe funcionar según lo esperado en términos de programación.

Otro objetivo fundamental de la programación es que sean de códigos claros y legibles, con lo que si un programador inicia un programa y no lo termina, otro programador sea capaz de entender la codificación y poder terminarlo.

Por ultimo la programación pretende que sus programas sean útiles y eficientes. De multitud de maneras la programación nos dará el mismo resultado de un programa, un buen programador llegara al mismo resultado con un mínimo de código posible. De los anteriormente nombrados objetivos de la programación el más importante es el de la corrección, ya que un código claro y legible facilita el mantenimiento de la aplicación o sistema.

II. 4. D. COMPILACIÓN O INTERPRETACIÓN DE LENGUAJES DE PROGRAMACIÓN.

Si un programa está escrito en un lenguaje de programación comprensible para un humano, se le llama código fuente. El código fuente se puede convertir en un archivo ejecutable con la ayuda de un compilador o también puede ser ejecutado de inmediato por medio de un intérprete.

Los programas que son compilados comúnmente son llamados ejecutables, imágenes binarias, o simplemente como binarios, ya que la forma en que se almacena el código de los ejecutables es en binario. Los compiladores se utilizan para traducir el código fuente de un lenguaje de programación, ya sea a código objeto o a código de máquina. El código objeto necesita una transformación más para convertirse en código de máquina, y el código de máquina es el código nativo del procesador,

listo para su ejecución. Un lenguaje de programación utilizado comúnmente para compilar es el lenguaje C.

Los programas interpretados podrían primeramente ser decodificados e inmediatamente después ejecutarse, o también puede darse el caso que se transforme a una eficiente representación intermedia para su futura ejecución. BASIC, Perl, y Python son ejemplos de lenguajes en los cuales los programas se ejecutan inmediatamente. De forma alternativa, los programas escritos en Java primeramente son compilados y almacenados en un código independiente de la computadora al cual se le llama bytecode. Un intérprete llamado máquina virtual ejecuta dicho bytecode cuando se le solicita.

Un lenguaje de programación no es estricta y exclusivamente compilado o interpretado. La clasificación usualmente refleja el método más popular de la ejecución del lenguaje. Por ejemplo, BASIC se trata como un lenguaje interpretado y C como un lenguaje compilado, a pesar de la existencia de compiladores para BASIC e intérpretes para C.

II. 4. E. PROGRAMAS QUE SE AUTO-MODIFICAN.

Un programa en ejecución se trata de forma diferente que los datos en los cuales opera. De cualquier forma, en algunos casos ésta distinción es ambigua, especialmente cuando un programa se modifica a sí mismo. El programa modificado es secuencialmente ejecutado como parte del mismo programa. Se pueden escribir programas auto-modificables en lenguajes como Lisp, COBOL y Prolog.

II. 4. F. EJECUCIÓN Y ALMACENAMIENTO DE LOS PROGRAMAS.

Típicamente, los programas se almacenan en la memoria no volátil, para que luego un usuario de la computadora, directa o indirectamente, solicite su ejecución. Al momento de dicha solicitud, el programa se carga en la memoria de acceso aleatorio, por medio del software llamado sistema operativo, el cual puede acceder directamente al procesador. El procesador ejecuta (corre) el programa, instrucción por instrucción hasta que termina. A un programa en ejecución se le llama proceso. Un programa puede terminar de forma normal o a causa de un error, dicho error puede ser de software o de hardware.

II. 4. F. a. PROGRAMAS EMPOTRADOS EN HARDWARE.

Algunos programas están empotrados en el hardware. Una computadora con arquitectura de programas almacenados requiere un programa inicial almacenado en su ROM para arrancar. El proceso de arranque es para identificar e inicializar todos los aspectos del sistema, desde los registros del procesador, controladores de dispositivos hasta el contenido de la memoria RAM. Seguido del proceso de inicialización, este programa inicial carga al sistema operativo e inicializa al contador de programa para empezar las operaciones normales.

FIGURA II. 6.
El microcontrolador a la derecha de
la Memoria USB está controlada por
un firmware empotrado.

Independiente de la computadora, un dispositivo de hardware podría tener firmware empotrado para el control de sus operaciones. El firmware se utiliza cuando se espera que el programa cambie en raras ocasiones o nunca, o cuando el programa no debe perderse cuando haya ausencia de energía.

II. 4. F. b. PROGRAMAS CARGADOS MANUALMENTE.

Los programas históricamente se cargaron manualmente al procesador central mediante interruptores. Una instrucción era representada por una configuración de estado abierto o cerrado de los interruptores. Después de establecer la configuración, se ejecutaba un botón de ejecución. Este proceso era repetitivo. También, históricamente los programas se cargaban manualmente mediante una cinta de papel o tarjetas perforadas. Después de que el programa se cargaba, la dirección de inicio se establecía mediante interruptores y el botón de ejecución se presionaba.

FIGURA II. 7.
Interruptores para la carga manual en
una Data General Nova 3.

II. 4. F. c. PROGRAMAS GENERADOS AUTOMÁTICAMENTE.

La programación automática es un estilo de programación que crea código fuente mediante clases genéricas, prototipos, plantillas, aspectos, y generadores de código para aumentar la productividad del programador. El código fuente se genera con herramientas de programación tal como un procesador de plantilla o un IDE. La forma más simple de un generador de código fuente es un procesador macro, tal como el preprocesador de C, que reemplaza patrones de código fuente de acuerdo a reglas relativamente simples.

Un motor de software da de salida código fuente o lenguaje de marcado que simultáneamente se vuelve la entrada de otro proceso informático. Podemos pensar como analogía un proceso manejando a otro siendo el código máquina quemado como combustible. Los servidores de aplicaciones son motores de software que entregan aplicaciones a computadoras cliente. Por ejemplo, un software para wikis es un servidor de aplicaciones que permite a los usuarios desarrollar contenido dinámico ensamblado a partir de artículos. Las Wikis generan HTML, CSS, Java, y Javascript los cuales son interpretados por un navegador web.

II. 4. F. d. EJECUCIÓN SIMULTÁNEA.

Muchos programas pueden correr simultáneamente en la misma computadora, a lo cual se le conoce como multitarea y puede lograrse a través de mecanismos de software o de hardware. Los sistemas operativos modernos pueden correr varios programas a través del planificador de procesos, un mecanismo de software para conmutar con frecuencia la cantidad de procesos del procesador de modo que los usuarios puedan interactuar con cada programa mientras estos están corriendo.

También se puede lograr la multitarea por medio del hardware; las computadoras modernas que usan varios procesadores o procesadores con varios núcleos pueden correr muchos programas a la vez.

II. 4. G. PROGRAMACIÓN ESTRUCTURADA.

La programación estructurada evolucionó en los años sesentas y setentas del siglo pasado. El nombre se refiere a la práctica de construir programas usando un conjunto de estructuras bien definidas. Una meta de la programación estructurada es la eliminación de la instrucción go to de la programación tipo spaghetti.

Los desarrolladores de software descubrieron se obtiene una eficiencia mejorada, pero continúan batallando con el proceso de construir software rápido y correctamente.

El reusó es reconocido como la clave de la solución. Reusar código permite que los programas se construyan rápida y correctamente. Las funciones, que son lo bloques constructores de la programación estructurada, son un paso a lo largo de este camino.

Los investigadores demostraron que los programas podían escribirse con tres estructuras de control:

- **Estructura de la secuencia.**

La estructura de la secuencia define el control automático en un programa. Normalmente esta estructura se construye en lenguajes de programación. Como resultado a menos que se dirija de otra manera una computadora ejecuta líneas de código en el orden en el cual están escritas. Los comandos están escritos en pseudocódigo, que es un lenguaje informal que los programadores usan mientras están trabajando con la lógica de un programa. Después de que la secuencia de comandos es desarrollada, los programadores traducen el pseudocódigo a un lenguaje específico de cómputo.

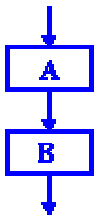


FIGURA II. 8.
Estructura de Secuencia.

- **Estructura de Selección.**

Las estructuras de selección se construyen con base en una declaración condicional. Si está es verdadera ciertas líneas de código son ejecutadas si es falsa, esas líneas de código no son ejecutadas. Las dos estructuras de selección más comunes son: If-Then e If Else (algunas veces llamada If-Then-Else).

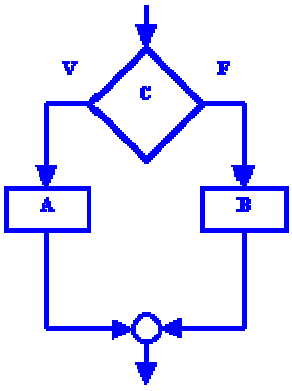


FIGURA II. 9.
Estructura de Selección.

- **Estructura de repetición.**

  Las estructuras de repetición (o de ciclo) también se construyen con base en instrucciones condicionales. Si la condición es verdadera entonces un bloque de uno o mas comandos se repite hasta que la condición es falsa. La computadora primero prueba la condición y, si es verdadera ejecuta el bloque de comando una vez. Entonces prueba la condición otra vez. Si aún es verdadero, el bloque de comando se repite. Debido a esté funcionamiento cíclico, las estructuras de repetición son llamadas también ciclos. Tres estructuras cíclicas son: For-Next, While y Do-While.

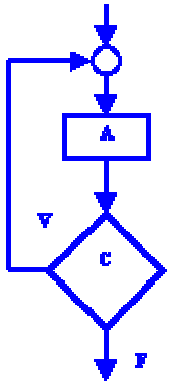


FIGURA II.10.
Estructura de Repetición.

II. 4. G. a. PASCAL.

El lenguaje de programación Pascal fue desarrollado originalmente por Niklaus Wirth, un miembro de la International Federation of Information Processing (IFIP). El Profesor Niklaus Wirth desarrolló Pascal para proporcionar rasgos que estaban faltando en otros idiomas en ese entonces.

Sus principales objetivos en el lenguaje eran ser eficientes para llevarse a cabo y ejecutar los programas, permitiendo el desarrollo de estructuras y también organizar programas, para servir como un vehículo para la enseñanza de los conceptos importantes de programación de la computadora.

Pascal que se nombró así gracias al matemático Blaise Pascal, es un descendiente directo de ALGOL 60, qué ayudó a su desarrollo. Pascal también tomó componentes de ALGOL 68 y ALGOL-W.

El original idioma de Pascal aparecido en 1971, con última revisión publicada en 1973. Fue diseñado para enseñar las técnicas de programación y otros temas a los estudiantes de la universidad y era el idioma de opción en las décadas comprendidas entre 1960 y 1980.

II. 4. G. b. Lenguaje C.

C es un lenguaje de programación creado en 1969 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL. Al igual que B, es un lenguaje orientado a la implementación de sistemas operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

Se trata de un lenguaje débilmente tipificado de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos.

El lenguaje C reúne características de programación intermedia entre los lenguajes ensambladores y los lenguajes de alto nivel; con gran poderío basado en sus operaciones a nivel de bits (propias de ensambladores) y la mayoría de los elementos de la programación estructurada de los lenguajes de alto nivel, por lo que resulta ser el lenguaje preferido para el desarrollo de software de sistemas y aplicaciones profesionales de la programación de computadoras.

En 1970, Ken Thompson de los laboratorios Bell se había propuesto desarrollar un compilador para el lenguaje Fortran que corría en la primera versión del sistema operativo UNIX tomando como referencia el lenguaje BCPL; el resultado fue el lenguaje B (orientado a palabras) que resultó adecuado para la programación de software de sistemas. Este lenguaje tuvo la desventaja de producir programas relativamente lentos.

En 1971, Dennis Ritchie, con base en el lenguaje B desarrolló NB que luego cambió su nombre por C; en un principio sirvió para mejorar el sistema UNIX por lo que se le considera su lenguaje nativo. Su diseño incluyó una sintaxis simplificada, la aritmética de direcciones de memoria (permite al programador manipular bits, bytes y direcciones de memoria) y el concepto de apuntador; además, al ser diseñado para mejorar el software de sistemas, se buscó que generase códigos eficientes y una portabilidad total, es decir el que pudiese correr en cualquier computadora. Logrados los objetivos anteriores, C se convirtió en el lenguaje preferido de los programadores profesionales.

En 1980, Bjarne Stroustrup de los laboratorios Bell de Murray Hill, New Jersey, inspirado en el lenguaje Simula67 adicionó las características de la programación orientada a objetos (incluyendo la ventaja de una biblioteca de funciones orientada a objetos) y lo denominó C con clases. Para 1983, dicha denominación cambio a la de C++. Con este nuevo enfoque surge la nueva metodología que aumenta las posibilidades de la programación bajo nuevos conceptos.

II. 4. H. PROGRAMACIÓN MODULAR.

Valiéndose de la técnica del diseño estructurado para el diseño de algoritmos consigue desarrollar programas a partir de un conjunto de módulos, cada uno de los cuales desempeña una tarea necesaria para el correcto funcionamiento del programa global. Los módulos son interdependientes, y son codificados y compilados por separado.

La modularidad es la capacidad que tiene un sistema de ser estudiado, visto o entendido como la unión de varias partes que interactúan entre sí y que trabajan para alcanzar un objetivo común, realizando cada una de ellas una tarea necesaria para la consecución de dicho objetivo. Cada una de esas partes en que se encuentre dividido el sistema recibe el nombre de módulo.

Idealmente un módulo debe poder cumplir las condiciones de caja negra, es decir, ser independiente del resto de los módulos y comunicarse con ellos (con todos o sólo con una parte) a través de unas entradas y salidas bien definidas.

En el caso de que el conjunto de módulos implicado esté sometido a una jerarquía, estaríamos hablando de un sistema de programación por capas, en la que cada "capa" representaría un nivel en la jerarquía de los módulos.

II. 4. H. a. MÓDULO.

En programación un módulo es una parte de un programa de computadora. De las varias tareas que debe realizar un programa para cumplir con su función u objetivos, un módulo realizará una de dichas tareas (o quizá varias en algún caso).

En un caso general (no necesariamente relacionado con la programación), un módulo recibirá como entrada la salida que haya proporcionado un módulo anterior o los datos de entrada al sistema (programa) si se trata del módulo inicial de éste; y proporcionará una salida que será utilizada como entrada de un módulo posterior o que será la salida final del sistema (programa) si se tratase del módulo final.

Particularmente, en el caso de la programación, los módulos suelen estar organizados jerárquicamente en niveles, de forma que hay un módulo superior que realiza las llamadas oportunas a los módulos del nivel inferior.

Cuando un módulo es llamado, recibe como entrada los datos proporcionados por el módulo de nivel superior que ha hecho la llamada, realiza su tarea, a su vez este módulo puede llamar a otro u otros módulos de nivel inferior si fuera necesario; cuando finaliza su tarea, devuelve la salida pertinente al módulo superior que lo llamo inicialmente, y es este módulo superior el que continúa con la ejecución del programa.

Características de un módulo:

Cada uno de los módulos de un programa idealmente debería cumplir las siguientes características:

- **Tamaño pequeño:** Facilita aislar el impacto que pueda tener la realización de un cambio en el programa, bien para corregir un error, bien por rediseño del algoritmo correspondiente.

- **Independencia modular:** Cuanto más independientes son los módulos entre sí más fácilmente se trabajará con ellos, esto implica que para desarrollar un módulo no es necesario conocer detalles internos de otros módulos. Como consecuencia de la independencia modular un módulo cumplirá:

    - Características de caja negra, es decir abstracción (ver abstracción en programación orientada a objetos).
    - Aislamiento de los detalles mediante encapsulamiento (ver encapsulamiento en programación orientada a objetos).

II. 4. I. PROGRAMACIÓN POR CAPAS.

La programación por capas es un estilo de programación en la que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño.

La ventaja principal de este estilo, es que el desarrollo se puede llevar a cabo en varios niveles y en caso de algún cambio solo se ataca al nivel requerido sin tener que revisar entre código mezclado. Un buen ejemplo de este método de programación sería: Modelo de interconexión de sistemas abiertos.

Además permite distribuir el trabajo de creación de una aplicación por niveles, de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles; simplemente es necesario conocer la API que existe entre niveles.

En el diseño de sistemas informáticos actual se suele usar las arquitecturas multinivel o Programación por capas. En dichas arquitecturas a cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

El diseño más en boga actualmente es el diseño en tres niveles (o en tres capas).

II. 4. I. a. CAPAS O NIVELES.

- **Capa de presentación**: es la que ve el usuario (hay quien la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario dando un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio.

- **Capa de negocio:** es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él.

- **Capa de datos:** es donde residen los datos y es la encargada de acceder a los datos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

Todas estas capas pueden residir en una única computadora (no es lo típico). Si bien lo más usual es que haya una multitud de computadoras en donde reside la capa de presentación (son los clientes de la arquitectura cliente/servidor).

Las capas de negocio y de datos pueden residir en la misma computadora, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más computadoras. Así, si el tamaño o complejidad de la base de datos aumenta, se puede separar en varias computadoras los cuales recibirán las peticiones de la computadora en que resida la capa de negocio.

Si por el contrario fuese la complejidad en la capa de negocio lo que obligase a la separación, esta capa de negocio podría residir en una o más computadoras que realizarían solicitudes a una única base de datos. En sistemas muy complejos se llega a tener una serie de computadoras sobre las cuales corre la capa de datos, y otra serie de computadoras sobre las cuales corre la base de datos.

En una arquitectura de tres niveles, los términos "capas" y "niveles" no significan lo mismo ni son similares.

El término "capa" hace referencia a la forma como una solución es segmentada desde el punto de vista lógico. En cambio, el término "nivel", corresponde a la forma en que las capas lógicas se encuentran distribuidas de forma física.

II. 4. J. PROGRAMACIÓN ORIENTADA A OBJETOS.

La Programación Orientada a Objetos (POO u OOP según sus siglas en inglés) es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora. Está basada en varias técnicas, incluyendo herencia, modularidad, polimorfismo, y encapsulamiento. Su uso se popularizó recién a principios de la década de 1990. Actualmente varios lenguajes de programación soportan la orientación a objetos.

Los objetos son entidades que combinan estado, comportamiento e identidad. El estado está compuesto de datos, y el comportamiento por procedimientos o métodos. La identidad es una propiedad de un objeto que lo diferencia del resto. La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar.

De esta forma, un objeto contiene toda la información que permite definirlo e identificarlo frente a otros objetos pertenecientes a otras clases e incluso frente a objetos de una misma clase, al poder tener valores bien diferenciados en sus atributos. A su vez, los objetos disponen de mecanismos de interacción llamados métodos que favorecen la comunicación entre ellos.

Esta comunicación favorece a su vez el cambio de estado en los propios objetos. Esta característica lleva a tratarlos como unidades indivisibles, en las que no se separan ni deben separarse el estado y el comportamiento.

Los métodos y atributos están estrechamente relacionados por la propiedad de conjunto. Esta propiedad destaca que una clase requiere de métodos para poder tratar los atributos con los que cuenta.

El programador debe pensar indistintamente en ambos conceptos, sin separar ni darle mayor importancia a ninguno de ellos. Hacerlo podría resultar en seguir el hábito erróneo de crear clases contenedoras de información por un lado y clases con métodos que la manejen por el otro. De esta manera se estaría llegando a una programación estructurada disfrazada en un lenguaje de programación orientado a objetos.

Esto difiere de la programación estructurada tradicional, en la que los datos y los procedimientos están separados y sin relación, ya que lo único que se busca es el procesamiento de unos datos de entrada para obtener otros de salida. La programación estructurada anima al programador a pensar sobre todo en términos de procedimientos o funciones, y en segundo lugar en las estructuras de datos que esos procedimientos manejan.

En la programación estructurada sólo se escriben funciones que procesan datos. Los programadores que emplean éste nuevo paradigma, en cambio, primero definen objetos para luego enviarles mensajes solicitándoles que realicen sus métodos por sí mismos.

II. 4. J. a. C++.

El C++ es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. Se puede decir que C++ es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos.

Las principales características del C++ son las facilidades que proporciona para la programación orientada a objetos y para el uso de plantillas o programación genérica (templates). Además posee una serie de propiedades difíciles de encontrar en otros lenguajes de alto nivel: posibilidad de redefinir los operadores (sobrecarga de operadores), identificación de tipos en tiempo de ejecución (RTTI).

C++ está considerado por muchos como el lenguaje más potente, debido a que permite trabajar tanto a alto como a bajo nivel, sin embargo es a su vez uno de los que menos automatismos trae (obliga a hacerlo casi todo manualmente al igual que C) lo que "dificulta" mucho su aprendizaje.

El nombre C++ fue propuesto por Rick Masciatti en el año 1983, cuando el lenguaje fue utilizado por primera vez fuera de un laboratorio científico. Antes se había usado el nombre "C con clases"."C++" significa "incremento de C" y se refiere a que es una extensión de C.

Paradigma:	multiparadigma: orientado a objetos, imperativo, programación genérica.
Apareció en:	1985.
Diseñado por:	Bjarne Stroustrup.
Tipo de dato:	fuerte, estático.
Implementaciones:	GNU Compiler Collection, Microsoft Visual C++, Borland C++ Builder, Dev-C++, C-Free.
Dialectos:	ISO C++, ANSI C++ 1998, ANSI C++ 2003.
Influido por:	C, Simula.
Ha influido:	Ada, C#, Java, PHP, D.

TABLA II. 1.
Lenguaje de Programación C++.

II. 4. K. PROGRAMACIÓN ORIENTADO A EVENTOS.

La programación orientada a eventos es un paradigma de programación en el que tanto la estructura como la ejecución de los programas van determinados por los sucesos que ocurran en el sistema o que ellos mismos provoquen. Para entender la programación orientada a eventos, podemos oponerla a lo que no es: mientras en la programación secuencial (o estructurada) es el programador el que define cuál va a ser el flujo del programa, en la programación orientada a eventos será el propio usuario o lo que sea que esté accionando el programa, el que dirija el flujo del programa.

Aunque en la programación secuencial puede haber intervención de un agente externo al programa, estas intervenciones ocurrirán cuando el programador lo haya determinado, y no en cualquier momento como puede ser en el caso de la programación orientada a eventos.

El creador de un programa orientado a eventos debe definir los eventos que manejará su programa y las acciones que se realizarán al producirse cada uno de ellos, lo que se conoce como el manejador de evento. Los eventos soportados estarán determinados por el lenguaje de programación utilizado, por el sistema operativo e incluso por eventos creados por el mismo programador.

En la programación orientada a eventos, al comenzar la ejecución del programa se llevarán a cabo las inicializaciones y demás código inicial y a continuación el programa quedará bloqueado hasta que se produzca algún evento. Cuando alguno de los eventos esperados por el programa tenga lugar, el programa pasará a ejecutar el código del correspondiente manejador de evento. La programación orientada a eventos es la base de lo que llamamos interfaz de usuario.

II. 4. K. a. Microsoft Visual Basic.

Visual Basic es un lenguaje de programación desarrollado por Alan Cooper para Microsoft. El lenguaje de programación es un dialecto de BASIC, con importantes añadidos. Su primera versión fue presentada en 1991, con la intención de simplificar la programación utilizando un ambiente de desarrollo completamente gráfico que facilitaría la creación de interfaces gráficas y en cierta medida también la programación misma.

Visual Basic constituye un IDE (entorno de desarrollo integrado o en inglés Integrated Development Enviroment) que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código (programa donde se escribe el código fuente), un depurador (programa que corrige errores en el código fuente para que pueda ser bien compilado), un compilador (programa que traduce el código fuente a lenguaje de máquina), y un constructor de interfaz gráfica o GUI (es una forma de programar en la que no es necesario escribir el código para la parte gráfica del programa, sino que se puede hacerlo de forma visual).

Es un lenguaje de fácil aprendizaje pensado tanto para programadores principiantes como expertos, guiado por eventos, y centrado en un motor de formularios que facilita el rápido desarrollo de aplicaciones gráficas. Su sintaxis, derivada del antiguo BASIC, ha sido ampliada con el tiempo al agregarse las características típicas de los lenguajes estructurados modernos. No requiere de manejo de punteros y posee un manejo muy sencillo de cadenas de caracteres. Posee varias bibliotecas para manejo de bases de datos, pudiendo conectar con cualquier base de datos a través de ODBC (Informix, DBase, Access, MySQL, SQL Server, Postgre). Es utilizado principalmente para aplicaciones de gestión de empresas, debido a la rapidez con la que puede hacerse un programa que utilice una base de datos sencilla, además de la abundancia de programadores en este lenguaje.

CAPÍTULO III.

DISEÑO DE SOFTWARE

III. 1. ORIGENES DE JAVA.

Originalmente, el lenguaje de programación Java no fue creado para la red Internet. La primera versión empezó en 1991 y fue escrita en 18 meses en Sun Microsystems. De hecho, en ese momento, ni siquiera se llamó Java; se llamó Oak y se utilizó en Sun para uso interno. La idea original para Oak era crear un lenguaje orientado a objetos independiente de la plataforma. Por entonces, muchos programadores se limitaban a la programación del IBM PC, pero el entorno corporativo podía incluir toda clase de plataformas de programación, desde el PC hasta los grandes sistemas. Lo que había detrás de Oak era crear algo que se pudiera usar en todos los ordenadores y, ahora que Java se ha hecho popular gracias a la red Internet, cada vez más corporaciones están adoptándolo para uso interno en lugar de **C++,** precisamente por esa razón. El lanzamiento original de Oak no fue especialmente fascinante; Sun quería crear un lenguaje que se pudiera usar en electrónica. Oak pasó a llamarse Java en 1995, cuando se lanzó para el uso público y supuso un éxito casi inmediato. En ese momento, Java había adoptado un modelo que lo hizo perfecto para la red de Internet, el modelo bytecode(códigos exclusivos de los programas realizados en Java).

III. 2. BYTECODES.

Cualquier programa en Visual C++ de Microsoft es grande, normalmente un programa MFC puro no baja de 5MB sin incluir las librerías de enlace dinámico (DLLs) que la plataforma Windows necesita para ejecutar. En otras palabras, los programas C++ son completamente ejecutables en el ordenador en el que residen, lo que justifica que tengan un gran tamaño. Imaginemos que intentamos descargar todo eso como parte de una página Web para permitir que esa página haga algo interactivo en su ordenador.

Los programas Java, por el contrario, se construyen de forma diferente. El propio lenguaje Java está implementado como la máquina virtual de Java (Java Virtual Machine), que es la aplicación que actualmente ejecuta un programa Java. Cuando JVM se instala en un ordenador, éste puede ejecutar programas Java. Los programas Java, por lo tanto, no necesitan ser autosuficientes, y no tienen por qué incluir todo el código máquina que se ejecuta en el ordenador. En cambio, los programas Java son compilados creando bytecodes compactos y, son estos bytecodes de lo que JVM lee e interpreta para ejecutar el programa. Cuando descarga una applet Java de la red Internet, lo que realmente está descargando es un archivo de bytecodes.

De esta forma, su programa Java puede ser muy pequeño, ya que todo el código máquina necesario para ejecutarlo está ya en el ordenador de destino y no tiene que descargarse. Para distribuir Java entre una gran variedad de ordenadores, Sun sólo tuvo que rescribir JVM para que funcionara en esos ordenadores. Dado que su programa está almacenado en un archivo de bytecode, se ejecutará en cualquier ordenador en el que JVM esté instalado.

Aunque en un principio se suponía que los programas Java eran interpretados por JVM, es decir, ejecutados bytecode por bytecode, la interpretación podía ser un proceso lento. Por esta razón, Java incorpora la compilación Justo en el tiempo JIT (Just In Time) en JVM. El compilador JIT realmente lee los bytecodes por secciones y los compila de forma interactiva en lenguaje máquina, por lo que el programa puede ejecutarse más rápido (todo el programa Java no se compila de una vez, ya que Java va realizando comprobaciones en tiempo de ejecución en varias secciones del código). Desde su punto de vista, esto quiere decir que su programa Java se ejecutará más rápido con el nuevo compilador JIT.

Utilizar bytecodes significa, que los programas Java son muy compactos, lo que les hace ideales para descargarlos en la red Internet. Y otra ventaja a la hora de ejecutar tales programas con JVM, mayor que la descarga de programas, es la seguridad.

III. 3. LA SEGURIDAD EN JAVA.

Cuando se ejecuta un programa, JVM puede monitorizar estrictamente lo que va ocurriendo, lo cual es importante para las aplicaciones de la red de Internet. Fuera de toda duda, la seguridad ha llegado a ser un asunto extremadamente importante en la red de Internet, y Java se ha lanzado a esa tarea. JVM puede ver todo lo que un programa hace, y si hay algo dudoso, como puede ser tratar de escribir en un archivo, puede prevenir esa operación. Eso sólo hace que, para la red de Internet, Java sea más atractivo que C++, porque no tiene restricciones.

Los programas Java son pequeños, seguros, independientes de la plataforma, orientados a objetos, y potentes. Además presentan otras características que gustan a los programadores. Los programas Java son robustos (lo que significa que son fiables y pueden gestionar bien los errores), con frecuencia son sencillos de escribir comparados con C++, son multihilo (pueden ejecutar un número de tareas al mismo tiempo, lo cual es útil cuando se quiere continuar haciendo otras cosas mientras se espera a que un archivo de datos se descargue) y ofrecen un alto rendimiento. El resultado final es que se puede escribir un programa Java una vez y puede descargarse fácilmente y ejecutarse en todo tipo de computadoras, por lo que es la receta perfecta para la red de Internet.

Esta es la razón por la que Java ha llegado tan alto. Java no tiene como único objetivo a la red de Internet; de hecho, hay dos tipos de programas Java, uno para el uso de la red de Internet y otro para el uso en máquina local.

III. 4. PROGRAMACIÓN JAVA.

Los programas Java son de dos tipos principales: aplicaciones y applets. Los applets son programas Java que pueden descargarse y ejecutarse como parte de una página Web, y son las que han hecho que Java sea tan popular.

Además de applets descargables, Java soporta aplicaciones que están diseñadas para ejecutarse localmente. Las aplicaciones Java funcionan como otras aplicaciones de ordenador, puede instalarlas y ejecutarlas en el suyo. Al estar instaladas localmente en vez de ser descargadas con una página Web, las aplicaciones tienen más privilegios que las applets, como es la capacidad para leer y escribir archivos. La base para poder operar cualquier producto que utiliza java es el JDK de la plataforma correspondiente, por lo cual antes de instalar el servidor, se debe instalar el JDK.

Existen dos versiones del JDK, para este propósito sirven cualquiera de las dos, a saber:

- JRE (Java Runtime Environment).
- SDK (Software Development Kit).

III. 4. A. CREACIÓN DE ARCHIVOS DE CÓDIGO.

Los programas Java son archivos de texto planos formados por instrucciones y declaraciones Java. Guardar texto en formato de texto plano es una ejecución sencilla que la mayor parte de los procesadores de texto soportan. Podría tener problemas con procesadores de texto como Microsoft Word, por ejemplo, aunque puede guardar archivos de texto con, Word ejecutando el comando Guardar como del menú Archivo.

La regla general es que si al editar un archivo desde la línea de comando no se ve ningún carácter que no sea alfanumérico suelto, se trata de un archivo de texto plano. La prueba real, por supuesto, es que el compilador de Java, que traduce su programa en un archivo de bytecode, pueda leer e interpretar dicho programa. Además, los programas deben estar almacenados en archivos que tengan extensión "java". Por ejemplo, si está escribiendo una aplicación llamada app, debería guardar el programa Java en un archivo llamado app.java. Pase este archivo al compilador de Java para crear el archivo de bytecode.

Cuando se escriba código Java, deberá saber que Java reserva ciertas palabras clave como parte del lenguaje. No hay muchas, de cualquier modo.

A continuación se muestran:

- **Abstract:** Especifica la clase o método que se va a implementar más tarde en una subclase.
- **Boolean:** Tipo de dato que sólo puede tomar los valores verdadero o falso.
- **Break:** Sentencia de control para salirse de los bucles.
- **Byte:** Tipo de dato que soporta valores en 8 bits.
- **Byvalue:** Reservada para uso futuro.
- **Base:** Se utiliza en las sentencias switch para indicar bloques de texto.
- **Cast:** Reservada para uso futuro.
- **Catch:** Captura las excepciones generadas por las sentencias try.
- **Char:** Tipo de dato que puede soportar caracteres Unicode sin signo en 16 bits.
- **Class:** Declara una clase nueva.
- **Const:** Reservada para uso futuro.
- **Continue:** Devuelve el control a la salida de un bucle.
- **Default:** Indica el bloque de código por defecto en una sentencia switch.

- **Do:** Inicia un bucle do-while.
- **Double:** Tipo de dato que soporta números en forma flotante, 64 bits.
- **Else:** Indica la opción alternativa en una sentencia if.
- **Extends:** Indica que una clase es derivada de otra o de una interfaz.
- **Final:** Indica que una variable soporta un valor constante o que un método no se sobrescribirá.
- **Finally:** Indica un bloque de código en una estructura try - catch que siempre se ejecutará.
- **Flota:** Tipo de dato que soporta un número en coma flotante en 32 bits.
- **For:** Utilizado para iniciar un bucle for.
- **Future:** Reservada para uso futuro.
- **Generic:** Reservada para uso futuro.
- **Goto:** Reservada para uso futuro.
- **If:** Evalúa si una expresión es verdadera o falsa y la dirige adecuadamente.
- **Implements:** Especifica que una clase implementa una interfaz.
- **Import:** Referencia a otras clases.
- **Inner:** Reservada para uso futuro.
- **Instanceof:** Indica si un objeto es una instancia de una clase específica o implementa una interfaz específica.
- **Int:** Tipo de dato que puede soportar un entero con signo de 32 bits.
- **Interface:** Declara una interfaz.
- **Long:** Tipo de dato que soporta un entero de 64 bits.
- **Native:** Especifica que un método está implementado con código nativo (específico de la plataforma).
- **New:** Crea objetos nuevos.
- **Null:** Indica que una referencia no se refiere a nada.
- **Operator:** Reservado para uso futuro.

- **Outer:** Reservado para uso futuro.
- **Package:** Declara un paquete Java.
- **Private:** Especificador de acceso que indica que un método o variable sólo puede ser accesible desde la clase en la que está declarado.
- **Protected:** Especificador de acceso que indica que un método o variable sólo puede ser accesible desde la clase en la que está declarado (o una subclase de la clase en la que está declarada u otras clases del mismo paquete).
- **Public:** Especificador de acceso utilizado para clases, interfaces, métodos y variables que indican que un tema es accesible desde la aplicación (o desde donde la clase defina que es accesible).
- **Rest:** Reservada para uso futuro.
- **Return:** Envía control y posiblemente devuelve un valor desde el método que fue invocado.
- **Short:** Tipo de dato que puede soportar un entero de 16 bits.
- **Static:** Indica que una variable es un método de una clase (más que estar limitado a un objeto particular).
- **Super:** Se refiere a una clase base de la clase (utilizado en un método o constructor de clase).
- **Switch:** Sentencia que ejecuta código basándose en un valor.
- **Synchronized:** Especifica secciones o métodos críticos de código multihilo.
- **This:** Se refiere al objeto actual en un método o constructor.
- **Throw:** Crea una excepción.
- **Throws:** Indica qué excepciones puede proporcionar un método.
- **Transient:** Especifica que una variable no es parte del estado persistente de un objeto.
- **Try:** Inicia un bloque de código es comprobado para las excepciones.
- **Var:** Reservado para uso futuro.
- **Void:** Especifica que un método no devuelve ningún valor.
- **Volatile:** Indica que una variable puede cambiar de forma asíncrona.
- **While:** Inicia un bucle while.

III. 4. B. DISEÑO DE PROGRAMAS JAVA.

El diseño de los programas en Java no es tarea fácil. Un buen diseño de programación involucra muchos aspectos, de hecho, uno de los aspectos más importantes de la creación de una nueva aplicación es diseñarla. Si no se selecciona bien puede que sea necesario hacer muchas revisiones del producto.

Este proceso de diseño se divide en cuatro áreas, las cuales explicaremos a continuación.

III. 4. B. a. RENDIMIENTO.

El rendimiento es un tema de diseño que es duro de argumentar. Si los usuarios no consiguen lo que quieren con su aplicación, esto se convierte claramente en un problema. En general, el rendimiento depende de las necesidades de los usuarios. Para algunas personas, la velocidad es esencial; para otros, la robustez o el uso eficiente de los recursos es lo que están buscando. Globalmente, el rendimiento de una aplicación es una indicación de lo bien que responde a las necesidades de los usuarios. Algunos aspectos generales de rendimiento que se deben considerar cuando se codifican programas en Java son:

- Eficiencia del algoritmo.
- Velocidad de CPU.
- Diseño y normalización eficiente de la base de datos.
- Limitación de accesos externos.
- Velocidad de la red.
- Temas de seguridad.
- USO de recursos.
- Velocidad de acceso a la Web.

III. 4. B. b. MANTENIMIENTO.

El mantenimiento es la medida de lo que fácilmente puede adaptarse su aplicación a necesidades futuras. Este asunto se deriva de buenas prácticas de programación. Buena parte de esto es de sentido común, simplemente tener en mente las necesidades de codificación futura al escribir el código. Algunos puntos de la "programación óptima" son los siguientes:

- Evitar el uso de bucles y condicionales anidados.
- Evitar el paso de variables globales a procedimientos.

- Ser modular cuando se escribe el código.
- Dividir el código en paquetes.
- Documentar los cambios de programa.
- Dar a cada procedimiento un único propósito.
- Asegurarse de que la aplicación puede extenderse sin problemas a más tareas y más usuarios.
- Planificar la reutilización de código.
- Programar de forma defensiva.
- Uso de procedimientos para el acceso a datos sensibles.
- Uso de comentarios.
- Uso de nombres de variables consistentes.
- Uso de constantes en lugar de números "mágicos".

III. 4. B. c. EXTENSIBILIDAD.

La extensibilidad es la capacidad de la aplicación para extenderse de una forma bien definida y relativamente fácil. Generalmente, supone una preocupación en las aplicaciones grandes y, con frecuencia, involucra a toda una interfaz especialmente diseñada para la extensión de módulos. De hecho, Java, en sí mismo, está diseñado para ser extendido, usando el framework de extensiones Java.

III. 4. B. d. DISPONIBILIDAD.

La disponibilidad es medir el tiempo en que la aplicación puede utilizarse, en comparación con el tiempo que los usuarios quieren utilizarla. Esto lo incluye todo, desde que no se quede congelada cuando se ejecuta una tarea larga (al menos, dar al usuario el estado de la operación), hasta trabajar con técnicas y métodos que no se cuelguen, hacer backups de datos críticos y planificar el uso alternativo de recursos, si es posible, cuando el acceso al recurso deseado esté bloqueado.

Globalmente el proceso de diseño es de los que más tiempo requiere. De hecho, todo el ciclo de desarrollo es tema de muchos estudios, puede resultar sorprendente saber que algunos de ellos consideran que el diseño es, al menos, el quince por ciento del proyecto total cuando se prueba un campo y se añaden planificación, diseño y pruebas de interfaz de usuario.

III. 4. C. VARIABLES, ARRAYS Y CADENAS.

El trabajo con datos es parte fundamental de cualquier programa, por tanto el almacenamiento y recuperación de los mismos en las variables, arrays y cadenas.

III. 4. C. a. VARIABLES.

Las variables pueden ser de diferentes tipos y actúan como gestores de memoria de los datos. Los diferentes tipos tienen que ver con el formato de los datos que se almacenan en ellas, así como con la memoria que es necesaria para gestionar ese dato. Por ejemplo, la variable de tipo entero, int, es de 4 bytes (o 32 bits) y se utiliza para almacenar valores enteros. Esto hace que un dato de tipo int pueda tomar un rango de valores que va desde -2,147,483,648 hasta 2,147,483,647.

Existen más tipos de las clases de variables como son:

- **Enteros:** Estos tipos son byte, short, int y long, que guardan el signo y el valor.
- **Números en coma flotante:** Estos tipos son float y double, que almacenan números en coma flotante con signo.
- **Caracteres:** Este es el tipo char, que guarda las representaciones de los caracteres, tanto letras como números.
- **Booleano:** Este tipo está diseñado para guardar sólo dos valores: verdadero (true) o falso (false).

III. 4. C. b. TIPOS DE DATOS.

Todas las variables sencillas deben tener un tipo y, de hecho, toda expresión, toda combinación de términos que Java puede evaluar para obtener un resultado, también tiene un tipo. Además, Java es muy particular para mantener la integridad de esos tipos, especialmente si se intenta asignar un valor de un tipo a una variable de otro tipo. Java es más insistente en cuanto al tipo de datos que un lenguaje como C++; en C++, por ejemplo, se puede asignar un número en coma flotante a un entero y C++ gestionará la conversión de tipos, pero eso no se puede hacer en Java.

III. 4. C. c. ARRAYS.

Los tipos sencillos son buenos para almacenar datos simples, pero con frecuencia, los datos son más complejos. Utilizando un array, podrá agrupar tipos de datos sencillos en estructuras más complejas y hacer referencia a esa nueva estructura por su nombre. Lo que es más importante: mediante un índice numérico, podrá hacer referencia a los datos individuales almacenados en el array. Eso es importante, porque los ordenadores se destacan por ejecutar millones de operaciones muy rápidamente, por lo tanto si se puede hacer referencia a los datos con un índice numérico, se puede trabajar muy rápido con un conjunto de datos, simplemente incrementando el índice del array para así, acceder a todos sus elementos.

III. 4. C. d CADENAS.

Una de las cosas de C++ que hacía felices a los programadores era que la mayoría de las implementaciones incluían una clase String, que se podía usar igual que cualquier otro tipo de datos. Java continúa con este uso, implementando las cadenas como una clase, no como un tipo de dato intrínseco, pero la gestión de cadenas es tan fundamental para la programación, que tiene sentido revisar las cadenas.

En Java, las cadenas se gestionan como si fueran objetos. Una de las ventajas de esto es que un objeto de tipo string tiene gran variedad de métodos que se pueden usar en muchos lenguajes, las cadenas de texto son tipos de datos fundamentales inherentes al lenguaje, pero en Java, las cadenas son gestionadas con las clases String y StringBufSer. Veamos primero la clase String. Los objetos de tipo string gestionan las cadenas de texto que no se pueden cambiar; si se quiere cambiar el texto actual de la cadena, se debería usar la clase StringBuffer.

La clase String es muy poderosa, pues con los métodos que proporciona permite convertir la cadena en un array de caracteres, convertir números en cadenas, buscar cadenas, crear substrings, cambiar la cadena de mayúsculas a minúsculas o viceversa, obtener la longitud de la cadena, comparar cadenas y mucho más.

III. 4. D. OPERADORES, CONDICIONALES Y BUCLES.

Almacenar muchos datos en el programa es bueno, siempre y cuando se haga algo con ellos. Utilizando operadores, se pueden manipular los datos, sumar, restar, multiplicar, dividir y mucho más. Con los condicionales, se puede alterar el flujo de un programa evaluando los valores de los datos. Con los bucles, se puede iterar sobre todos los datos de un grupo, como un array, trabajando con ellos sucesivamente de forma fácil.

III. 4. D. a OPERADORES.

La forma más básica de trabajar con los datos en un programa es hacerlo con los operadores de Java. A continuación mostramos una lista de los operadores de Java:

- **--** (decremento).
- **-** (resta).
- **!** (No lógico).
- **!=** (distinto).
- **%** (módulo).
- **%=** (asignación del módulo).
- **&** (AND a nivel de bit)

- **&&** (AND en cortocircuito).
- **&=** (asignación de AND).
- (multiplicación).
- ***=** (asignación del producto).
- ***1*** (división).
- **/=** (asignación de la división).
- ?: (if-then-else).
- **^** (Xor a nivel de bit).
- **^=** (asignación de Xor).
- **1** (OR a nivel de bit).
- **ll** (OR en cortocircuito).
- **l=** (asignación de OR).
- **–** (NOT unario a nivel de bit).
- **+** (suma).
- **++** (incremento).
- **+=** (asignación de la suma).
- **<**(menor que).
- **c<** (desplazamiento de bits hacia la izquierda).
- **<c=** (asignación del desplazamiento de bits a la izquierda).
- **c=** (menor o igual que).
- **=** (asignación).
- **-=** (asignación de la resta).
- **==** (igual a).
- (mayor que).

- **>=** (mayor o igual que).
- **>>** (desplazamiento a la derecha).
- **>>=** (asignación del desplazamiento a la derecha).
- **>>>** (desplazamiento a la derecha con relleno de ceros).
- **>>>=** (asignación del desplazamiento a la derecha con relleno de ceros).

Aquellos que tienen sólo un operando se llaman operadores unarios. Los que tienen dos operandos, por ejemplo, la suma (a + b), se llaman binarios. Además, hay un operador, ?:, que tiene tres operandos: el operador ternario.

III. 4. D. b. CONDICIONALES.

Después de los operadores, el siguiente paso es usar las sentencias condicionales, también llamadas instrucciones de control de flujo. Se utilizan para tomar decisiones basadas en el valor de los datos y dirigir el flujo del programa de acuerdo a esos valores. Cuando se quiere controlar el flujo del código, es el momento de usar las sentencias condicionales de Java, como la sentencia if.

Las sentencias if pueden ser más complejas si se añaden las cláusulas else. Deben seguir a la sentencia if y se ejecutan cuando su condición es falsa. También se pueden tener sentencias anidadas una dentro de otra (esto es, definirlas dentro de otras sentencias). También es posible crear una secuencia entera de sentencias if-else, que se conoce como escala if-else. La sentencia switch tiene la misma funcionalidad que una cadena de sentencias if-else.

III. 4. D. c. BUCLES.

Los bucles son fundamentales en la programación y permiten gestionar distintas tareas repitiendo la ejecución de cierto código específico. Por ejemplo, puede que quiera gestionar los elementos de un grupo de datos trabajando con cada uno de ellos en serie, o quizás quiera ejecutar una tarea hasta que cierta condición sea verdadera.

III. 4. D. c. i. BUCLE FOR.

El bucle básico involucra a la sentencia for, que permite ejecutar un bloque de código usando un índice. Cada vez que se pasa por el bucle, el índice tendrá un valor diferente y se puede usar para hacer referencia a cada elemento del conjunto de datos. El índice del bucle se usa como si fuera un índice de un array. El bucle for de Java es una buena elección cuando se quiere usar un índice numérico que se incremente o decremente automáticamente cada vez que se pase por el bucle, como ocurre cuando se está trabajando con un array.

III. 4. D. c. ii. BUCLE WHILE Y DO WHILE.

El cuerpo de un bucle while, puede ser una sentencia compuesta por un grupo de sentencias simples encerradas entre llaves; se ejecuta mientras una condición lógica sea verdadera.

El bucle do-while es como un bucle while, salvo en que la condición es evaluada al final del bucle, no al comienzo. Este es el formato del bucle dowhile. La principal razón para usar do-while en vez de while es que se necesite que el cuerpo del bucle se ejecute al menos una vez.

III. 4. D. c. iii. SENTENCIA BREAK.

Algunos lenguajes incluyen la sentencia goto que se puede utilizar para saltarse alguna sentencia del código; pero la mayor parte de los lenguajes consideran que goto no es estructurada (Java es de ésos y no incluye la sentencia goto). Como Java no tiene una sentencia goto, soporta la sentencia break con este propósito.

III. 4. E. AWT: APPLETS, APLICACIONES Y GESTIÓN DE EVENTOS.

Java AWT (Abstract Windowing Toolkit: Paquete de Herramientas de Ventanas Abstractas) la forma original de Java para trabajar con gráficos. AWT fue desarrollado muy rápidamente para la primera release de Java, de hecho, en sólo seis semanas.

Los desarrolladores del AWT original usaron una ventana para cada uno de sus componentes, es decir, para cada botón, cuadros de texto, casilla de activación, etc., y por lo tanto, tenían su propia ventana cuando al sistema operativo le interesaba. Esto producía un consumo considerable de los recursos del sistema cuando los programas llegaban a ser grandes.

Sun introdujo el paquete Swing, en el que los componentes se visualizan usando métodos gráficos de la applet o aplicación que los contiene, ellos no tienen sus propias ventanas del sistema operativo.

Los componentes AWT se llaman componentes pesos pesados debido al uso significativo que hacen de los recursos del sistema y los componentes Swing se llaman componentes peso ligero, ya que no necesitan sus propias ventanas. ¿Qué significa esto? Está claro que para Sun, Swing es el futuro. Hay más componentes Swing que AWT, y de hecho, hay un componente Swing para cada componente AWT.

En el futuro, probablemente, Sun no ampliará el conjunto de componentes AWT mucho más, mientras que sí se espera que Swing crezca. Por otro lado, Swing, en sí mismo, está basado en AWT; las ventanas que Swing utiliza para visualizar sus componentes, es decir, ventanas, marcos de ventanas, applets y diálogos, están todos ellos basados en contenedores AWT. AWT no va a desaparecer y para trabajar con Swing, se necesita AWT.

III. 4. E. a. ABATRACT WINDOWING TOOLKIT.

No es una exageración decir que la aparición de Abstract Windowing Toolkit fue obligada tras la popularidad de Java. Se puede crear y visualizar botones, etiquetas, mentís, cuadros de lista desplegables, cuadros de texto y otros controles de la interfaz de usuario que se esperan utilizar en la programación basada en ventanas utilizando AWT. Esta es una lista de las clases de AWT más populares:

- **Applet**: Crea una applet.
- **Button**: Crea un botón.
- **Canvas**: Crea un área de trabajo en el que se puede dibujar.
- **Checkbox**: Crea una casilla de activación.
- **Choice**: Crea un control de opción.
- **Label**: Crea una etiqueta.
- **Menu**: Crea un menú.
- **ComboBox**: Crea un cuadro de lista desplegable.
- **List**: Crea un cuadro de lista.
- **Frame**: Crea un marco para las ventanas de aplicación.
- **Dialog**: Crea un cuadro de diálogo.
- **Panel**: Crea un área de trabajo que puede tener otros controles.
- **PopupMenu*:*** Crea un menú emergente.
- **RadioButton*:*** Crea un botón de opción.
- **ScrollBar*:*** Crea una barra de desplazamiento.
- ***ScrollPane:*** Crea un cuadro de desplazamiento.
- **TextArea*:*** Crea un área de texto de dos dimensiones.
- **TextField*:*** Crea un cuadro de texto de una dimensión (en otros lenguajes se llama TextBox***).***
- ***TextPane:*** Crea un área de texto.
- **Window*:*** Crea una ventana.

La clase **Applet** de AWT es aquella en la que están basadas las **applets** AWT.

III. 4. E. b. APPLETS.

Una applet es un fichero de clase que se escribe específicamente para visualizar gráficos en la red Internet. Las applets se incluyen en las páginas Web utilizando la etiqueta HTML <APPLET>. Cuando se ejecutan en una página Web, las applets de Java se descargan automáticamente y el browser las ejecuta, visualizándose en el espacio de la página que se ha reservado para ellas. Pueden hacer de todo, desde trabajar con gráficos hasta visualizar animaciones, gestionar controles, cuadros de texto y botones. El uso de las applets hace que las páginas Web sean activas, no pasivas, que es su principal atracción. Cuando se trabaja con AWT, el proceso es como sigue: se crea una applet nueva, basándola en la clase java.applet.App1et que, a su vez, está basada en la clase Component de AWT.

La applet se compila en un fichero de bytecode con extensión ".class". Una vez que se tiene este fichero, se sube a un proveedor de servicios de la red Internet (ISP). A una applet se le puede dar la misma protección que se daría a una página Web, asegurándose de que cualquiera pueda leer el fichero de la applet con extensión ".class".

III. 4. E. c. APLICACIONES.

Las ventanas de aplicación AWT están basadas en la clase Frame de AWT, que crea una ventana con un marco que visualiza botones y un título. Uno de los aspectos más importantes de la creación de applets y aplicaciones es permitir al usuario interactuar con el programa por medio de los eventos. Cuando el usuario ejecuta alguna acción: hacer clic sobre un botón; cerrar una ventana; seleccionar un elemento de una lista o usar el ratón, por ejemplo Java lo considera un evento. El usuario puede utilizar el ratón para hacer clic sobre botones, lo que inicia alguna acción en el programa, como es escribir texto en un cuadro de texto. De hecho, el clic sobre los botones, es quizás el evento más básico que Java soporta.

III. 4. E. d. GESTIÓN DE EVENTOS.

La gestión de eventos, proceso de respuesta que se genera al hacer clic sobre el botón, los movimientos del ratón, etc; ha llegado a ser un tema complejo en Java. El modelo actual se llama gestión de eventos delegado. En este modelo, se debe registrar específicamente en Java si se quiere gestionar un evento, como puede ser hacer: clic sobre un botón. La idea es que se mejora la ejecución si sólo se informa de los eventos al código que necesita gestionarlos y no al resto.

Los eventos se registran implementando una interfaz de listener de eventos. Estos son los eventos de listeners disponibles y los tipos de eventos que gestionan:

- **ActionListener**: Gestiona los eventos de acción, como hacer clic sobre los botones.
- **AdjustementListener**: Gestiona los casos en los que un componente es escondido, movido, redimensionado o mostrado.
- **ContainerListener**: Gestiona el caso en el que un componente coge o pierde el foco.
- **ItemListener**: Gestiona el caso en el que cambia el estado de un elemento.
- **KeyListener**: Recibe los eventos de teclado.

- **MouseListener**: Recibe en los casos en que es pulsado el ratón, mete un componente, sale un componente o es presionado.
- **MouseMotionListener**: Recibe en el caso en que se arrastra o mueve el ratón.
- **TextListener**: Recibe los cambios de valor de texto.
- **WindowListener**: Gestiona los casos en que una ventana está activada, desactivada, con o sin forma de icono, abierta, cerrada o se sale de ella.

Cada listener es una interfaz, y se deben implementar los métodos de la interfaz. A cada uno de estos métodos se le pasa un tipo de objeto que corresponde al tipo de evento:

- **ActionEvent**: Gestiona botones, el hacer doble clic en la lista o hacer clic en un elemento del menú.
- **AdjustementEvent**: Gestiona los movimientos de la barra de desplazamiento.
- ***ComponentEvent:*** Gestiona el caso en el que un componente es escondido, movido, redimensionado o llega a ser visible.
- ***FocusEvent:*** Gestiona el caso en el que un componente coge o pierde el foco.
- ***InputEvent:*** Gestiona la marca de activación en una casilla de activación y el hacer clic en un elemento de la lista, hacer selecciones en los controles de opción y las selecciones de los elementos de un menú.
- ***KeyEvent:*** Gestiona la entrada desde el teclado.
- ***MouseEvent:*** Gestiona los casos en que se arrastra el ratón, se mueve, se pulsa, se presiona, se suelta o entra o sale un componente.
- ***TextEvent:*** Gestiona el valor de un cuadro de texto o si ha cambiado.
- ***WindowEvent:*** Gestiona el caso en el que una ventana está activada, desactivada, en forma de icono, sin forma de icono, abierta, cerrada o abandonada.

III. 4. F. PROGRAMACIÓN MULTIHILO.

Un hilo es un flujo de ejecución de código y, mediante el uso de hilos, se puede hacer que nuestros programas aparentemente realicen varias tareas al mismo tiempo. Por ejemplo, su código podría interaccionar con el usuario mientras realiza tareas en segundo plano de gran consumo de tiempo. Los hilos separados realmente no se ejecutan al mismo tiempo, por supuesto (a menos que tenga un equipo de cómputo multiprocesador; en realidad, cada hilo obtiene secuencias de tiempo del mismo procesador.

El resultado aparente, no obstante, es que los diversos hilos se ejecutan al mismo tiempo y puede resultar impresionante. De hecho, Java es uno de los pocos lenguajes de programación que soporta multihilos explícitamente.

Cuando se inicia un programa Java, éste tiene un hilo: el hilo principal. Se puede interaccionar con el hilo principal de diversas formas, por ejemplo obteniendo o asignando su nombre, deteniéndolo y mucho más. Sin embargo, podemos también iniciar otros hilos. Se puede iniciar el código en esos hilos llamando al método start del objeto y situando el código que queremos que utilice el hilo en el método run.

Existen dos formas para crear nuevos hilos. Una es declarar una clase que extienda Thread. Esta subclase sobrescribiría el método run de la clase Thread y a continuación podemos alojar e iniciar una instancia de esa clase.

La otra forma de crear un hilo es declarar una clase que implemente la interfaz Runnable. Esa clase entonces implementará el método run. Una instancia de la clase puede alojarse (es decir, pasarse como argumento cuando crea un objeto Thread).

También es interesante observar que cada hilo tiene un nombre para propósitos de identificación (de hecho, más de un hilo puede tener el mismo nombre). Cualquier hilo también tiene una prioridad y los hilos con prioridades mayores se ejecutan de forma preferente con respecto a los hilos de prioridad menor.

III. 4. G. CREACIÓN DE PAQUETES, INTERFACES, ARCHIVOS JAR.

Cuando tenemos varios archivos de clase, es buena idea distribuirlos en el disco duro utilizando una jerarquía de directorios. De hecho, los paquetes Java fueron diseñados originalmente para reflejar esa organización de archivos. Puede distribuir archivos de clase en una jerarquía de directorios y permitir que Java sepa lo que sucede con los paquetes. Por ejemplo, si se tiene un paquete llamado packagel, que contiene una clase app.class, el archivo de clase iría en un directorio llamado packagel.

Como el directorio packagel se encuentra en una localización donde Java buscará Java buscará los archivos de clase que utilice como parte del paquete en ese directorio. Tiene que indicarle a qué paquete pertenece un archivo de clase, utilizando la sentencia package en su código. Una vez compilada app.class y almacenada en el directorio packagel, podemos entonces importar la clase app en el código.

III. 4. G. a. CREACIÓN DE PAQUETES.

Cuando tenemos gran cantidad de archivos de clase, podemos organizarlos en una estructura de paquetes bastante compleja y, lo hacemos creando la estructura de directorios correspondiente en el disco, incluyendo las estructuras de subdirectorios dadas. Para crear esta estructura de paquetes en el código, basta utilizar el separador de paquetes punto.

III. 4. G. b. INTERFACES.

Cuando creamos una interfaz, especificamos los métodos de la interfaz y cuando la implementamos, proporcionamos el código de estos métodos. Para crear una interfaz, utilizamos simplemente la sentencia interface y listamos los prototipos (declaraciones sin cuerpo) de los métodos que queremos en la interfaz.

III. 4. G. c. ARCHIVOS JAR.

Un archivo Java (JAR) puede contener varios archivos. De hecho, los archivos JAR comprimen sus contenidos utilizando el formato ZIP, por lo que si su applet necesita una gran cantidad de archivos grandes, es conveniente crear un archivo JAR. Los navegadores de la red de Internet únicamente necesitan una conexión en vez de nuevas conexiones para cada nuevo archivo, lo que puede mejorar los tiempos de descarga. También puede firmar digitalmente los archivos de un archivo JAR para probar su origen.

Las opciones posibles para utilizar la herramienta jar son las siguientes:

Funciones.	**Descripción.**
c.	Crea un archivo nuevo o vacío en la salida estándar.
t.	Muestra la tabla de contenidos en la salida estándar.
x file.	Extrae todos los archivos o sólo los nombres de archivos. Si file se omite, se extraen todos los archivos; en cualquier otro caso, únicamente se extraen los archivos especificados.
f.	El segundo argumento especifica un archivo JAR para trabajar.
v.	Genera una salida "duplicada" en stderr.
m.	Incluye información manifiesta de un archivo manifest especificado.
O.	Indica "sólo almacenar", sin utilizar compresión ZIP.
M.	Especifica que un archivo manifest no debería crearse por las entradas.
u.	Actualiza un archivo JAR existente añadiendo o cambiando los archivos del manifest.
C.	Cambia los directorios durante la ejecución del comando jar. Por ejemplo, jar añadiría todos los archivos dentro del directorio clases, pero no el propio directorio clases, al archivo jarfile.jar.

III. 4. H. JDBC.

JDBC es un API incluido dentro del lenguaje Java para el acceso a bases de datos. Consiste en un conjunto de clases e interfaces escritos en Java que ofrecen un completo API para la programación de bases de datos, por lo tanto es la una solución 100% Java que permite el acceso a bases de datos.

Debido a que JDBC está escrito completamente en Java también posee la ventaja de ser independiente de la plataforma. No será necesario escribir un programa para cada tipo de base de datos, una misma aplicación escrita utilizando JDBC podrá manejar bases de datos Oracle, Sybase, o SQL Server. Además podrá ejecutarse en cualquier sistema que posea una Máquina Virtual de Java, es decir, serán aplicaciones completamente independientes de la plataforma.

Básicamente el API JDBC hace posible la realización de las siguientes tareas:

- Establecer una conexión con una base de datos.
- Enviar sentencias SQL.
- Manipular los datos.
- Procesar los resultados de la ejecución de las sentencias.

JDBC es una API de bajo nivel ya que hace llamadas SQL directas, Sun desea que JDBC pueda ser llamado desde otra API de más alto nivel que pueda simplificar la labor del programador, aunque la utilización de JDBC es sencilla y potente.

III. 4. H. a. ESTRUCTURA DE JDBC.

La columna vertebral de JDBC es el Driver Manager (gestor de drivers) que se encuentra representado por la clase java.sql.DriverManager. El gestor de drivers es pequeño y simple y su función primordial es la de seleccionar el driver adecuado para conectar la aplicación o applet con una base de datos determinada, y acto seguido desaparece. Se puede considerar que JDBC ofrece dos conjuntos de clases e interfaces bien diferenciados, aquellas de más alto nivel que serán utilizados por los programadores de aplicaciones para el acceso a bases de datos, y otras de más bajo nivel enfocadas hacia los programadores de drivers que permiten la conexión a una base de datos.

III. 4. H. b. INTERFACE DE CONEXIÓN.

Un objeto Connection representa una conexión con una base de datos. Una sesión de conexión con una base de datos incluye las sentencias SQL que se ejecuten y los resultados que devuelvan.

Una sola aplicación puede tener una o más conexiones con una misma base de datos, o puede tener varias conexiones con diferentes bases de datos. Connection es un interfaz, ya que como ya se había dicho anteriormente, JDBC ofrece una plantilla o especificación que deben implementar los fabricantes de drivers de JDBC.

Los métodos presentes en el interfaz Connection son:

- **void clearWarnings**(**):** Elimina todas las advertencias ofrecidas por un objeto Connection.
- **void close**(): Cierra una conexión, liberando todos los recursos asociados a la misma.
- **void commit**(): Lleva a cabo todos los cambios realizados dentro de una transacción y libera todos los bloqueos correspondientes.
- **Statement createStatement**(): Crea una sentencia SQL, representada mediante un objeto Statement, para poder enviar sentencias al origen de datos.
- **Statement createStatement(int tipoResultSet, int tipoConcurrencia)**: Cera también una sentencia SQL pero que generará en su ejecución un objeto ResultSet con unas características determinadas.
- **boolean getAutoCommit**(): Comprueba si la conexión se encuentra en estado de auto-commit, este concepto se comentará detenidamente más adelante.
- **String getCalalog**(): Devuelve una cadena con el nombre del catálogo utilizado por la conexión.
- **DatabaseMetaData getMetaData**(): Devuelve un objeto DatabaseMetaData que contiene información detallada sobre la base de datos a la que nos encontramos conectados.
- **int getTransactionIsolation**(): Devuelve el grado de aislamiento que se ha establecido para la conexión y que se utilizará a la hora de realizar transacciones.
- **Map getTypeMap**(): Devuelve el mapa de tipos utilizados para una conexión.
- **SQLWarning getWarnings**(): Devuelve todos los avisos generados por la base de datos a la que estamos conectados.
- **boolean isClosed**(): Devuelve verdadero si la conexión está cerrada.
- **boolean isReadOnly**(): Devuelve true si la conexión es de sólo lectura.
- **String nativeSQL(String sql)**: Convierte la sentencia SQL que se pasa como parámetro, en la gramática SQL nativa del sistema gestor de bases de datos a los que nos encontramos conectados.
- **CallableStatement prepareCall(String sql)**: Crea un objeto CallableStatement que va a permitir llamar y ejecutar procedimientos almacenados de la base de datos a la que estamos conectados.
- **CallableStatement prepareCall(String sql, int tipoResultSet, int tipoConcurrencia)**: Tiene la misma finalidad que el método anterior, pero permite especificar características relativas al objeto RecordSet que se generará a la hora de ejecutar el objeto CallableStatement.

- **PreparedStatement prepareStatement(String sql)**: Crea un objeto PreparedStatement que va a permitir enviar y ejecutar sentencias SQL parametrizadas a la base de datos correspondiente.

- **PreparedStatement prepareStatement(String sql, int tipoResultSet, int tipoConcurrencia)**: Igual que el método anterior, pero permite especificar la características del ResultSet que se obtendrá a la hora de ejecutar el objeto PreparedStatement correspondiente.

- **void rollback**(): Deshace las modificaciones realizadas en una transacción y libera todos los bloqueos asociados. Es otro método relacionado con la ejecución de transacciones en JDBC.

- **void setAutoCommit(boolean autoCommit)**: Establece el modo auto-commit de la conexión para tratar las transacciones.

- **void setCatalog(String catálogo)**: Establece el catálogo de la conexión para poder acceder a él.

- **void setReadOnly(boolean soloLectura)**: Establece si una conexión posee el modo de sólo lectura o no, dependiendo del argumento de tipo booleano.

- **void setTransactionIsolation(int nivel)**: Establece el nivel de aislamiento de una conexión a la hora de ejecutar y tratar transacciones.

- **void setTypeMap(Map mapa)**: Establece el mapa de tipos utilizado en la conexión.

III. 4. H. c. REALIZANDO LA CONEXIÓN.

La forma común de establecer una conexión con una base de datos es llamar al método getConnection() de la clase DriverManager, a este método se le debe codificar como parámetro la URL de JDBC que identifica a la base de datos con la que queremos realizar la conexión. La ejecución de este método devolverá un objeto Connection que representará la conexión con la base de datos.

Este método esta sobrecargado, le podemos pasar solamente la URL, la URL con el identificador de usuario y la contraseña o bien una URL y una lista de propiedades. Debido a que la conexión la obtenemos a través de un método de la clase DriverManager.

Una restricción de seguridad que se debe tener en cuenta es la localización de la base de datos, en el caso de que estemos utilizando JDBC desde un applet y nos queramos conectar a una base de datos. Esta base de datos debe residir en el mismo servidor desde el que se cargo el applet, es decir, el applet y la base de datos deben estar en el mismo servidor.

Una vez que hemos terminado de utilizar la base de datos, invocaremos el método close() sobre el objeto Connection para cerrar la base de datos y de esta forma liberar recursos.

III. 5. LIBRERÍAS Y PROPIEDADES UTILIZADAS EN EL DISEÑO DE SOFTWARE.

A continuación presentamos las principales librerías y funciones del lenguaje Java utilizadas en el diseño del software, así como la explicación de la activación del puerto comm, para la comunicación del lector de tarjetas inteligentes.

III. 5. A. LIBRERIAS.

Estas son algunas de las librerías:

- **import java.awt.* :** El paquete Abstract Windowing Toolkit (awt) contiene clases para generar widgets y componentes GUI (Interfaz Gráfico de Usuario). Incluye las clases Button, Checkbox, Choice, Component, Graphics, Menu, Panel, TextArea y TextField.
- **import java.awt.event.*:** Librería para todos los eventos en JAVA.
- **import javax.swing.*:** Libreria para el manejo de graficos en JAVA.
- **import java.sql.*:** Libreria que nos permite conectarnos con la Base de Datos, a través del conector (mysql-connector-java-5.0.5-bin), mediante esta librería también podemos manejar las consultas SQL en nuestro sistema.
- **import java.util.*:** Contiene colecciones de datos y clases, el modelo de eventos, facilidades horarias, generación aleatoria de números, y otras clases de utilidad.
- **import javax.comm.*:** Libreria que nos permite utilizar el Puerto serial.

III. 5. B. PROPIEDADES.

A continuación se describen algunas propiedades:

- **SetIcon():** Nos permite colocar una imagen en un jButton.
- **setText():** Nos permite colocar texto en un jButton, en un jText y jLabel.
- **getText:** Extrae el texto contenido en un jText.
- **equals():** Compara 2 cadenas o números, si son iguales devuelve True, si son diferentes devuelve False.
- **getString():** Devuelve el valor contenido en la columna de nuestra Base de Datos encerrada entre paréntesis y entre comillas o por el numero de la columna, posterior a esta instrucción se antepone la variable que contiene el resultado de la sentencia SQL.
- **JOptionPane.showMessageDialog():** Imprime un mensaje o datos, se debe de anteponer la palabra reservada null.

- **grabFocus():** Coloca el cursor en el jText que le indiquemos.
- **addItem():** Permite a agregar datos a un jComboBox ya sea en tiempo de ejecución o no.
- **removeAllItems():** Elimina todos los datos contenidos en un jComboBox.
- **getSelectedItem():** Extrae el contenido seleccionado de un jComboBox.
- **setEnabled():** Permite que un objeto sea usado o no en tiempo de ejecución. Los valores que devuelve son True (se puede trabajar con el objeto), False (no se puede trabajar con el objeto).
- **setEditable();** Permite que un objeto sea usado o no en tiempo de ejecución, parecido al **setEnabled(),** pero **setEditable()** es mas enfocado a los jText. Devuelve True (se puede escribir sobre el jText), False (no se puede escribir sobre el jText).

III. 5. C. CONEXIÓN CON LA BASE DE DATOS.

JDBC es un API (aplication programming interface: interfaz de programación de aplicaciones) incluido dentro del lenguaje Java para el acceso a bases de datos. Consiste en un conjunto de clases e interfaces escritos en Java que ofrecen un completo API para la programación de bases de datos, por lo tanto es la una solución 100% Java que permite el acceso a bases de datos.

Para utilizar esta herramienta lo primero que tenemos que hacer es registrar nuestro Driver de la siguiente manera:

DriverManager.registerDriver(neworg.gjt.mm.mysql.Driver());

Posterior a esta acción creamos una variable de tipo conexión que es la que nos hará la conexión con nuestra base de datos.

Proporcionamos el primer parámetro del método getConnection(), que es es un String donde se localiza la URL de la base de datos:

- **JDB**: MySQL porque estamos utilizando un driver JDBC para MySQL.
- **Localhost**: Es el servidor de base de datos. Aquí puede ponerse una IP o un nombre de máquina que esté en la red.
- **Restaurante**: Es el nombre de la base de datos que hemos creado dentro de mysql. Se debe poner la base de datos dentro del servidor de MySQL a la que se quiere uno conectar. Los otros dos parámetros son dos String. Corresponden al nombre de usuario y password para acceder a la base de datos. Al instalar MySQL se crea el usuario root y se pide la password para él. Como no hemos creado otros usuarios, usaremos este mismo que no tiene password.

La conexión con la base de datos se realiza de la forma siguiente:

```
conexion=DriverManager.getConnection("jdbc:mysql://localhost/restaurante","root","");
```

Una vez conectados la forma de consultar nuestra base de datos es a través de los comandos siguientes:

- **rs.First ():** Nos posiciona al princio de nuestra consulta SQL.
- **rs.next ():** Avanzamos sobre nuestra consulta (1 a 1).
- **rs.last ():** Nos posicionamos al final de nuestra consulta SQL.
- **rs.beforeFirst ():** Volvemos a posicionarnos al principio de nuestra consulta, estoy es muy útil cuando: hemos avanzado en los registros de nuestra consulta o hemos llegado al final de la misma y queremos volver a posicionarnos al principio de nuestra consulta.
- **rs.getRow ():** Obtenemos el número de registros contenidos en nuestra consulta SQL
- **rs.close ():** Cerramos nuestra conexión con la Base de Datos

CAPÍTULO IV

COMPONENTES DEL SISTEMA COMPUTACIONAL

IV. 1. COMPONENTES DE UNA COMPUTADORA.

Antes de enumerar los distintos componentes de un sistema computacional, deberíamos definir qué entendemos por "computadora". Una computadora es un dispositivo electrónico compuesto básicamente de procesador, memoria y dispositivos de entrada/salida. Los componentes de una computadora pueden clasificarse en dos: Hardware, Software.

IV. 1. A. HARDWARE.

Corresponde a todas las partes físicas y tangibles de una computadora: sus componentes eléctricos, electrónicos, electromecánicos y mecánicos; sus cables, gabinetes o cajas, periféricos de todo tipo y cualquier otro elemento físico involucrado; contrariamente al soporte lógico e intangible que es llamado software. Se define como el "Conjunto de los componentes que integran la parte material de una computadora".

Una de las formas de clasificar el Hardware es en dos categorías: por un lado, el "básico", que abarca el conjunto de componentes indispensables necesarios para otorgar la funcionalidad mínima a una computadora, y por otro lado, el "Hardware complementario", que, como su nombre indica, es el utilizado para realizar funciones específicas (más allá de las básicas), no estrictamente necesarias para el funcionamiento de la computadora.

Los componentes del soporte físico o hardware más importantes son los siguientes:

- Procesador.
- Memoria RAM (Random Access Memory: Memoria de Acceso Aleatorio).
- Disco duro.
- Unidad de CD ROM (Compact Disc Read Only Memory: Disco Compacto de Memoria Solo de Lectura).
- Unidad de CD RW (Compact Disc ReWritable: Disco Compacto Regrablable).
- Módem.
- Caché secundario.
- Tarjeta madre.
- Puertos USB (Universal Serial Bus: Conductor Universal en Serie).
- Unidad de DVD ROM (Digital Versatil Disc Read Only Memory: Disco Digital Versátil de Memoria de Solo Lectura).
- Teclado.

- Impresora.
- Escáner.
- Monitor (ya que ellos nos ayudan a cumplir nuestro propósito).

IV. 1. B. SOFTWARE.

El soporte lógico o software, es el conjunto de instrucciones que una computadora emplea para manipular datos: por ejemplo, un procesador de textos o un videojuego. Estos programas suelen almacenarse y transferirse a la CPU (Central Processing Unit: Unidad de Proceso Central) a través del hardware de la computadora.

El software también rige la forma en que se utiliza el hardware, como por ejemplo la forma de recuperar información de un dispositivo de almacenamiento. La interacción entre el hardware de entrada y de salida es controlada por un software llamado BIOS (siglas en inglés de 'sistema básico de entrada / salida').

Aunque, técnicamente, los microprocesadores todavía se consideran hardware, partes de su función también están asociadas con el software. Como los microprocesadores tienen tanto aspectos de hardware como de software, a veces se les aplica el término intermedio de microprogramación, o firmware.
Software o programas de computadoras. Son las instrucciones responsables de que el hardware (el equipo computacional) realice su tarea. Como concepto general, el software puede dividirse en varias categorías basadas en el tipo de trabajo ha realizar. Las dos categorías primarias de software son los sistemas operativos (software del sistema), que controlan los trabajos de la computadora , y el software de aplicación, que dirige las distintas tareas para las que se utilizan .

El software del sistema procesa tareas tan esenciales, aunque a menudo invisibles, como el mantenimiento de los archivos del disco y la administración de la pantalla, mientras que el software de aplicación lleva a cabo tareas de tratamiento de textos, gestión de bases de datos y similares. Constituyen dos categorías separadas el software de red, que permite comunicarse a grupos de usuarios, y el software de lenguaje utilizado para escribir programas

Además de estas categorías basadas en tareas, varios tipos de software se describen basándose en su método de distribución. Entre estos se encuentran los así llamados programas enlatados, el software desarrollado por compañías y vendido principalmente por distribuidores, el freeware y software de dominio público, que se ofrece sin costo alguno, el shareware, que es similar al freeware, pero suele conllevar una pequeña tasa a pagar por los usuarios que lo utilicen profesionalmente y, por último, el infame vapourware, que es software que no llega a presentarse o que aparece mucho después de lo prometido.

IV. 2. FUNCIONAMIENTO INTERNO DE LA COMPUTADORA.

Al iniciar el arranque, en la mayoría de las computadoras, cualquiera que sea su tamaño o potencia, el control pasa mediante circuito cableado a unas memorias de tipo ROM, grabadas con información permanente (datos de configuración, fecha y hora, dispositivos, etc.)

Después de la lectura de esta información, el circuito de control mandará a cargar en la memoria principal desde algún soporte externo (disco duro o disquete) los programas del sistema operativo que controlarán las operaciones a seguir, y en pocos segundos aparecerá en pantalla el identificador o interfaz, dando muestra al usuario que ya se está en condiciones de utilización.

Si el usuario carga un programa con sus instrucciones y datos desde cualquier soporte de información, bastará una pequeña orden para que dicho programa comience a procesarse, una instrucción tras otra, a gran velocidad, transfiriendo la información desde y hacia donde esté previsto en el programa con pausas, en las que se pide al usuario entradas de información.

Finalizada esta operación de entrada, la computadora continuará su proceso secuencial hasta culminar la ejecución del programa, presentando sus resultados en pantalla, impresora o cualquier periférico.

Cada una de las instrucciones tiene un código diferente expresado en formato binario. El proceso de una instrucción se descompone en operaciones muy simples de transferencia de información u operaciones aritméticas y lógicas elementales, que realizadas a gran velocidad le proporcionan una gran potencia que es utilizada en múltiples aplicaciones. Realmente, esa información digitalizada en binario, a la que se refiere con unos y ceros, la computadora diferencia porque se trata de niveles diferentes de voltaje.

Cuando se emplean circuitos integrados, los niveles lógicos bajo y alto, que se representan por ceros y unos, corresponden a valores muy próximos a cero y cinco voltios en la mayoría de los casos. Cuando las entradas de las puertas lógicas de los circuitos digitales se les aplica el nivel alto o bajo de voltaje, el comportamiento es muy diferente.

Si se le aplica nivel alto conducen o cierran el circuito; en cambio si se aplica nivel bajo no conducen o dejan abierto el circuito. Para que esto ocurra, los transistores que constituyen los circuitos integrados trabajan en conmutación, pasando del corte a la saturación.

IV. 3. PRINCIPALES COMPONENTES DEL SISTEMA COMPUTACIONAL.

El conjunto de dispositivos electrónicos y electromecánicos interconectados que almacenan y transforman símbolos en base a las instrucciones especificadas por los componentes del software se les denomina componentes del sistema computacional.

IV. 3. A. LA PLACA MADRE O MAINBOARD.

La placa madre es la tarjeta principal y es considerada como el centro nervioso de la computadora. Todos los demás componentes de los que consta un sistema se conectan a ella y quedan bajo su administración. Físicamente, se trata de una tarjeta electrónica de material sintético sobre la cual existe un circuito electrónico que conecta diversos elementos que se encuentran comunicados a ella. Los principales son:

- El microprocesador.
- La memoria.
- Los slots o ranuras de expansión.
- Diversos chips de control, entre ellos la BIOS.
- Conectores internos, externos y eléctricos.
- Elementos integrados.

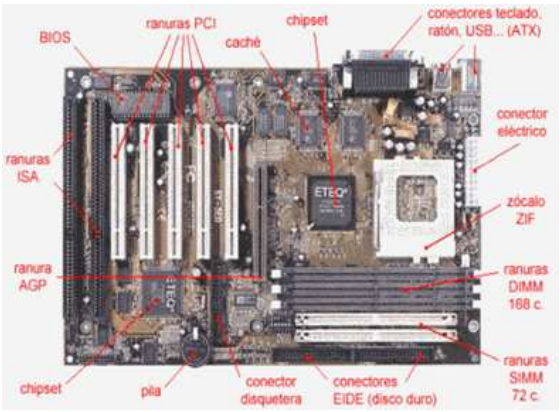


FIGURA IV. 1.
Ejemplo de placa madre en la que se señalan sus elementos más importantes.

IV. 3. B. BUS.

El bus es el "camino" que recorre un impulso eléctrico transportando datos de un componente hacia otro. Estos caminos son los medios de comunicación que permiten el intercambio de datos. En una PC se pueden encontrar varios tipos de bus:

- **Bus del procesador.**

  El bus del procesador es el canal de comunicación entre el procesador y los componentes de soporte, como por ejemplo la memoria caché externa. Además este bus transmite los datos hacia el bus principal del sistema. Este bus opera regularmente mucho más rápido que el resto de buses. Se compone de tres tipos de circuitos: de direcciones, de control y de datos.

- **Bus de memoria.**

  Este bus sirve de canal de comunicación entre el procesador y la memoria. De acuerdo al fabricante de la mainboard, este bus puede ser independiente del bus del procesador o estar integrado a él.

- **Bus de direcciones.**

  El bus de direcciones es un subconjunto de los buses del procesador y de la memoria. Este tipo de bus es utilizado para indicar una dirección de memoria o de sistema. Este bus también determina el tamaño de la memoria de acuerdo a la cantidad de direcciones que pueda direccionar.

- **Bus de entrada/salida o ranuras de expansión.**

  Este tipo de bus es el más conocido y usualmente se le llama simplemente "bus". Tiene una gran importancia, pues permite la comunicación con los dispositivos de video, disco o impresora. También es llamado bus principal del sistema. Su identificación física es simple, pues esta representada por las llamadas ranuras de expansión. Su evolución ha ido siempre a razón de su estandarización y su velocidad.

IV. 3. C. UNIDAD CENTRAL DE PROCESAMIENTO (CPU).

La CPU o Unidad Central de Proceso es la unidad donde se ejecutan las instrucciones de los programas y se controla el funcionamiento de los distintos componentes del computador, es un microchip con una alta escala de integración, es decir, que aloja millones de transistores en su interior. Se dice que si la mainboard es el sistema nervioso de la computadora, el procesador o correctamente llamado CPU, es el cerebro.

Suele estar integrada en un chip denominado microprocesador. Sin microprocesador la computadora no podría funcionar.

El CPU gestiona cada paso en el proceso de los datos. Actúa como el conductor de supervisión de los componentes de hardware del sistema. El CPU está compuesto por: registros, la unidad de control, la unidad aritmético-lógica, y dependiendo del procesador, una unidad de coma flotante.

Cada fabricante de microprocesadores tiene sus propias familias de productos y cada familia su propio conjunto de instrucciones.

El microprocesador realiza en varias fases de ejecución la realización de cada instrucción:

- Lee la instrucción desde la memoria principal.
- Decodifica la instrucción, es decir, determinar qué instrucción es y por tanto qué se debe hacer.

- Realiza la operación correspondiente.
- Ejecuta la operación.
- Escribe los resultados en la memoria principal o en los registros.

Cada una de estas fases se realiza en uno o varios ciclos de CPU dependiendo de la estructura del procesador. La duración de estos ciclos viene determinada por la frecuencia de reloj, y nunca podrá ser inferior al tiempo requerido para realizar la tarea individual (realizada en un solo ciclo) de mayor coste temporal.

El microprocesador dispone de un oscilador o cristal de cuarzo capaz de generar pulsos a un ritmo constante de modo que genera varios ciclos (o pulsos) en un segundo.

IV. 3. C. a. VELOCIDADES DE LA CPU.

Trabaja en frecuencias de Megahercios (MHz) o Gigahercios (GHz), lo que quiere decir millones o miles de millones, respectivamente, de ciclos por segundo. El indicador de la frecuencia de un microprocesador es un buen referente de la velocidad de proceso del mismo, pero no el único.

La cantidad de instrucciones necesarias para llevar a cabo una tarea concreta, así como la cantidad de instrucciones ejecutadas por ciclo son los otros dos factores que determinan la velocidad de la CPU. La cantidad de instrucciones necesarias para realizar una tarea depende directamente del juego de instrucciones disponible.

IV. 3. D. MEMORIA.

Este es uno de los componentes principales y que es más sensible a los cambios en su estándar comercial. A lo largo de la historia, han sido muchos los esfuerzos para mejorar su performance y su capacidad de almacenamiento. Los avances en el software y en las demandas de video han exigido que las memorias siempre estén un paso adelante de estas necesidades.

La memoria o memoria RAM (Random Acces Memory: Memoria de Acceso Aleatorio) como es comúnmente conocida se clasifica en:

- SIMM (Single In-line Memory Module: Modulo de Memoria lineal).
- DIMM (Dual In-line Memory Module: Módulo de Memoria en línea doble).
- DDR / DDR2 (Doble Data Rate: Doble Tasa de Transferencia).
- SO-DIMM (Small Outline-Dual In-line Memory Module: Línea Compacta de Salida del Módulo de Memoria en línea doble).
- RIMM (Rambus Inline Memory Module: Asignación en Línea de Módulos de Memoria).

Las diferencias entre ellas radican en el número de contactos, en su capacidad de almacenamiento y en su velocidad de acceso.

Podemos decir que en la actualida las memorias DDR2 son las más rápidas alcanzando 400 Mhz. A esta importante característica se añade su bajo consumo de energía y por consiguiente su baja producción de calor.

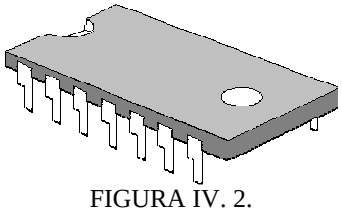


FIGURA IV. 2.
Memoria ROM.

La memoria ROM (Read Only Memory: Memoria de Solo Lectura) sirve para almacenar el programa básico de iniciación, instalado desde fábrica. Este programa entra en función en cuanto es encendida la computadora y su primer función es la de reconocer los dispositivos, incluyendo memoria de trabajo.

IV. 3. E. FUENTE DE PODER.

Este es tal vez uno de los componentes menos considerados por los usuarios de la computadora. Sin embargo, cumple una función muy importante al suministrar la energía eléctrica necesaria para el funcionamiento del sistema. La fuente de poder recibe la corriente eléctrica alterna desde la línea pública y la transforma en continua. Debido a que muchos componentes de la computadora funcionan a 3 / 5 v. y otros a 12 v. la fuente de poder se debe encargar del suministros de ambos voltajes.

IV. 3. F. UNIDADES DE ALMACENAMIENTO.

- **Lectores de CD-ROM**

  Los CD ROM son leídos por un lector de CD ROM y escritos por grabadores de CD (a menudo llamadas "quemadores"). Los lectores CD ROM pueden ser conectados a la computadora por la interfaz IDE (Integrated Device Electronics: Dispositivo Electrónico Integrado) ATA (Advance Tecnology Atachment: Accesoria de Tecnología Avanzada), por una interfaz SCSI (Small Computer Sistem Interface: Sistema de Interfaz para Microcomputadoras) o a través del puerto USB. La mayoría de los lectores de CD ROM leen CD de audio y CD de vídeo (VCD) con el software apropiado.

- **Lectores de DVD**

  Los lectores de DVD ROM también leen los DVD Video y los formatos en CD, como CD ROM, CD R, CD RW y CD's de Video. Para que pueda leer los DVD y CD los lectores contienen en su parte interna un ojo óptico en la cual lee el formato de unos y ceros con la que se encuentran grabados estos tipos de CD's.

- **USB drive o flash drive**

  Este dispositivo sin duda es ya el más utilizado por el público, ya que esta tecnología aprovecha la gran estandarización con que cuentan los puertos USB generalmente incluido en todas las mainboard. Además, casi no existe sistema operativo que no le brinde soporte. Sus dimensiones muy pequeñas hacen que no solo sea utilizado para el almacenamiento de datos sino también como reproductor de música y video.

IV. 3. G. EL DISCO DURO.

Se llama disco duro (en inglés hard disk, abreviado con frecuencia HD o HDD) al dispositivo encargado de almacenar información de forma persistente en un computador. Los discos duros generalmente utilizan un sistema de grabación magnética analógica. El disco aloja dentro de una carcasa una serie de platos metálicos apilados girando a gran velocidad. Sobre estos platos se sitúan los cabezales encargados de leer o escribir los impulsos magnéticos. Hay distintos estándares a la hora de comunicar un disco duro con el sistema. Los más utilizados son IDE/ATA, SCSI, Ultra ATA y Serial ATA.

IV. 3. G. a. ESTRUCTURA FÍSICA DEL DISCO DURO.

Dentro de un disco duro hay varios platos, que son discos (de aluminio o cristal) concéntricos y que giran todos a la vez.

El cabezal de lectura y escritura es un conjunto de brazos alineados verticalmente que se mueven hacia dentro o fuera según convenga, todos a la vez. En la punta de dichos brazos están las cabezas de lectura/escritura, que gracias al movimiento del cabezal pueden leer tanto zonas interiores como exteriores del disco.

Cada plato tiene dos caras, y es necesaria una cabeza de lectura/escritura para cada cara (no es una cabeza por plato, sino una por cara).En realidad, cada uno de los brazos es doble, y contiene 2 cabezas: una para leer la cara superior del plato, y otra para leer la cara inferior. Por tanto, hay 8 cabezas para leer 4 platos.

Las cabezas de lectura/escritura nunca tocan el disco, sino que pasan muy cerca (hasta a 3 nanómetros). Si alguna llega a tocarlo, causaría muchos daños en el disco, debido a lo rápido que giran los platos (uno de 7,200 revoluciones por minuto se mueve a 120 Km/h en el borde).

IV. 3. G. b. ESTRUCTURA LÓGICA DEL DISCO DURO.

Dentro del disco se encuentran:

El Master Boot Record (en el sector de arranque), que contiene la tabla de particiones.

Las particiones, necesarias para poder colocar los sistemas de ficheros

IV. 3. G. c. CARACTERÍSTICAS DE UN DISCO DURO.

Las características que se deben tener en cuenta en un disco duro son:

- **Tiempo medio de acceso.**

  Tiempo medio que tarda en situarse la aguja en el cilindro deseado; suele ser aproximadamente un 1/3 del tiempo que tarda en ir desde el centro al exterior o viceversa.

- **Tiempo de Giro.**

  Es el tiempo que tarda el disco en girar media vuelta, que equivale al promedio del tiempo de acceso (tiempo medio de acceso). Una vez que la aguja del disco duro se sitúa en el cilindro el disco debe girar hasta que el dato se sitúe bajo la cabeza; el tiempo en que esto ocurre es, en promedio, el tiempo que tarda el disco en dar medio giro; por este motivo la latencia es diferente a la velocidad de giro, pero es aproximadamente proporcional a ésta.

- **Tasa de transferencia**

  Velocidad a la que se transfiere la información al computador.

- **Caché de pista**

  Es una memoria de estado sólido, tipo RAM, dentro del disco duro de estado sólido. Los discos duros de estado sólido utilizan cierto tipo de memorias construidas con semiconductores para almacenar la información. El uso de esta clase de discos generalmente se limita a las supercomputadoras, por su elevado precio.

IV. 3. H. Puertos.

Un puerto es una forma genérica de denominar a una interfaz por la cual diferentes tipos de datos pueden ser enviados y recibidos. Dicha interfaz puede ser de tipo físico, o puede ser a nivel de software (por ejemplo, los puertos que permiten la transmisión de datos entre diferentes computadoras), en cuyo caso se usa frecuentemente el término puerto lógico.

IV. 3. H. a. PUERTO LÓGICO.

Se denomina así a una zona, o localización, de la memoria de una computadora que se asocia con un puerto físico o con un canal de comunicación, y que proporciona un espacio para el almacenamiento temporal de la información que se va a transferir entre la localización de memoria y el canal de comunicación.

En el ámbito de Internet, un puerto es el valor que se usa, en el modelo de la capa de transporte, para distinguir entre las múltiples aplicaciones que se pueden conectar al mismo host, o puesto.

Aunque muchos de los puertos se asignan de manera arbitraria, ciertos puertos se asignan, por convenio, a ciertas aplicaciones particulares o servicios de carácter universal. De hecho, la IANA (Internet Assigned Numbers Authority) determina, las asignaciones de todos los puertos comprendidos entre los valores [0, 1023], (la IANA sólo controlaba los valores desde el 0 al 255).

Los servicios y las aplicaciones que se encuentran en el listado denominado Selected Port Assignments. De manera análoga, los puertos numerados en el intervalo [1024, 65535] se pueden registrar con el consenso de la IANA, vendedores de software y organizaciones.

IV. 3. H. b. PUERTO FÍSICO.

Un puerto físico, es aquella interfaz, o conexión entre dispositivos, que permite conectar físicamente distintos tipos de dispositivos como: monitores, impresoras, escáneres, discos duros externos, cámaras digitales, memorias pendrive (USB), etc. Estas conexiones tienen denominaciones particulares como, por ejemplo, los puertos "serie" y "paralelo" de una computadora.

IV. 3. H. b. i. PUERTOS EN SERIE.

Un puerto serie recibe y envía información fuera de la computadora mediante un determinado software de comunicación o un drive del puerto serie.

El Software envía la información al puerto, carácter a carácter, convirtiendo en una señal que puede ser enviada por cable serie o un módem. Cuando se ha recibido un carácter, el puerto serie envía una señal por medio de una interrupción indicando que el carácter está listo. Cuando la computadora ve la señal, los servicios del puerto serie leen el carácter.

- **Ubicación en el sistema informático.**

  Se ubican en la parte trasera del case, podremos identificar estos puertos por los nombres COM 1, COM 2, COM 3. La cantidad de puertos serie dependen de la tarjeta, ya que hay algunas tarjetas que son capaces de tener 4 u 8 puertos.

IV. 3. H. b. ii. PUERTO PARALELO.

Este puerto de entrada/salida (E/S) envía datos en formato paralelo (donde 8 bits de datos, forman un byte, y se envían simultáneamente sobre ocho líneas individuales en un solo cable.) El puerto paralelo usa un conector tipo D-25 (es de 25 pin) El puerto paralelo se utiliza principalmente para impresoras.

La mayoría de los software usan el término LPT (impresor en línea) más un número para designar un puerto paralelo (por ejemplo, LPT1). Un ejemplo donde se utiliza la designación del puerto es el procedimiento de instalación de software donde se incluye un paso en que se identifica el puerto al cual se conecta a una impresora.

- **Ubicación en el sistema informático**

  Se encuentra en la parte trasera del case, se pueden identificar fácilmente ya que la mayoría de los software utilizan el término LPT (que significa impresión en línea por sus siglas en inglés). También en algunos modelos se pueden localizar en la parte inferior al puerto del Mouse.

- **Tipos de puerto paralelo:**

Los tipos de puertos paralelos se clasifican de la siguiente manera:

- Puerto paralelo estándar (Standard Parallel Port SPP).
- Puerto paralelo PS/2 (bidireccional).
- Enhanced Parallel Port (EPP).
- Extended Capability Port (ECP).

IV. 3. H. b. iii. PUERTO USB (UNIVERSAL SERIAL BUS).

El puerto USB fue creado a principio de 1996. La sigla USB significa Bus Serie Universal (Universal Serial Bus). Se llama universal, porque todos los dispositivos se conecten al puerto. Conexión que es posible, porque es capaz de hacer conectar hasta un total de 127 dispositivos.

Unas de las razones más importantes que dieron origen a este puerto fueron:

- Conexión del PC con el teléfono.
- Fácil uso.
- Expansión del puerto.

Las principales características de este puerto es que permite la conexión entre la PC y el teléfono, además, nos elimina la incomodidad al momento de ampliar la PC.

- **Ubicación en el sistema Informático.**

  El puerto USB está ubicado en la mayoría de los case en la parte frontal o lateral y en la parte trasera del mismo.

- **Tipos de transferencia:** El puerto USB permite cuatro tipos de transferencia, que son:

  - **Transferencias de control:** Es una transferencia no esperada, no se realiza periódicamente, sino que la realiza el software para iniciar una petición/respuesta de comunicación. Normalmente se utiliza para realizar operaciones de control o estado.

  - **Transferencias Isocrónicas:** Es periódica, una comunicación continúa entre el controlador y el dispositivo, se usa normalmente para información. Este tipo de transferencia envía la señal de reloj encapsulando en los datos, mediante comunicaciones NZRI.

  - **Transferencias Continúa:** Son datos pequeños no muy frecuentes, que provocan la espera de otras transferencias hasta que son realizadas.

- **Transferencias de Volumen:** No son transferencias periódicas. Se trata de paquetes de gran tamaño, usados en aplicaciones donde se utiliza todo el ancho de banda disponible en la comunicación. Estas transferencias pueden quedar a la espera de que el ancho de banda quede disponible.

IV. 3. H. b. iv CONECTORES RCA.

El conector RCA es un tipo de conector eléctrico común en el mercado audiovisual. El nombre RCA deriva de La Radio Corporation Of America, que introdujo el diseño en 1940.

- **Ubicación en el sistema informático**

  Éste está ubicado en la parte trasera del case, exactamente en la ranura donde fue colocada la tarjeta gráfica o de sonido. El conector RCA de video mayormente está presente en la tarjeta de video y el conector RCA de audio siempre está presente en la tarjeta de sonido.

IV. 3. H. b. v. CONECTOR DE VIDEO VGA.

El equipo utiliza un conector D subminiatura de alta densidad de 15 pines en el panel posterior para conectar al equipo un monitor compatible con el estándar VGA (Video Graphics Array (Arreglo de gráficos de videos). Los circuitos de video en la placa base sincronizan las señales que controlan los cañones de electrones rojo, verde y azul en el monitor.

- **Ubicación en el sistema informático**

  Se encuentran en la parte de atrás del case, no tienen un lugar en especifico pero en algunos modelos se pueden ubicar arriba de los conectores RCA y por un símbolo de red; en la mayoría de los casos solo se encuentra un solo puerto en el case.

IV. 3. H. b. vi. CONECTOR PS-2.

Es un conector de clavijas de conexión múltiples, DIN, (acrónimo de Deutsche Industriy Norm) miniatura, su nombre viene del uso que se le daba en las antiguas computadoras de IBM PS/2 (Personal System/2). Actualmente los teclados y ratones utilizan este tipo de conector y se supone que en unos años casi todo se conectará al USB, en una cadena de periféricos conectados al mismo cable.

IV. 3. H. b. vii. CONECTOR RJ-45.

El RJ45 es una interfaz física comúnmente usada para conectar redes de cableado estructurado, (categoría 4, 5, 5e y 6). RJ es un acrónico inglés de Registered que a su vez es parte del código federal de regulaciones de Estados Unidos. Posee ocho pines o conexiones eléctricas.

IV. 3. H. b. viii. CONECTOR RJ-11.

Es el conector modular común del teléfono. Es universal en los teléfonos, los módems, los faxes, y artículos similares y utilizado en receptores de la TV vía satélite.

- **Ubicación en el sistema informático.**

  El conector del módem RJ-11 se encuentra en la parte posterior de la computadora. La ficha RJ-11 es un enchufe modular con 4 pines.

IV. 3. H. b. ix. PUERTOS INALÁMBRICOS.

Las conexiones en este tipo de puertos se hacen, sin necesidad de cables, a través de la conexión entre un emisor y un receptor utilizando ondas electromagnéticas. Si la frecuencia de la onda, usada en la conexión, se encuentra en el espectro de infrarrojos se denomina puerto infrarrojo. Si la frecuencia usada en la conexión es la usual en las radio frecuencias entonces sería un puerto Bluetooth.

La ventaja de esta última conexión es que el emisor y el receptor no tienen porque estar orientados el uno con respecto al otro para que se establezca la conexión. Esto no ocurre con el puerto de infrarrojos. En este caso los dispositivos tienen que verse mutuamente, y no se debe interponer ningún objeto entre ambos ya que se interrumpiría la conexión.

IV. 4. LOS DISPOSITIVOS DE ENTRADA/SALIDA.

Son aquellos que permiten la comunicación entre la computadora y el usuario. Las computadoras electrónicas modernas son una herramienta esencial en muchas áreas: industria, gobierno, ciencia, educación, en realidad en casi todos los campos de nuestras vidas.

El papel que juegan los dispositivos periféricos de la computadora es esencial; sin tales dispositivos ésta no sería totalmente útil. A través de los dispositivos periféricos podemos introducir a la

computadora datos que nos sean útiles para la resolución de algún problema y por consiguiente obtener el resultado de dichas operaciones, es decir; poder comunicarnos con la computadora.

La computadora necesita de entradas para poder generar salidas y éstas se dan a través de dos tipos de dispositivos periféricos existentes: Dispositivos periféricos de entrada y Dispositivos periféricos de salida.

IV. 4. A. DISPOSITIVOS DE ENTRADA.

Son aquellos que sirven para introducir datos a la computadora para su proceso. Los datos se leen de los dispositivos de entrada y se almacenan en la memoria central o interna. Los dispositivos de entrada convierten la información en señales eléctricas que se almacenan en la memoria central.

Los dispositivos de entrada típicos son los teclados, otros son: lápices ópticos, palancas de mando (joystick), CD-ROM, discos compactos (CD), etc. Hoy en día es muy frecuente la utilización del llamado ratón que mueve un puntero electrónico sobre una pantalla que facilita la interacción usuario-máquina.

IV. 4. A. a. MOUSE.

La función principal del ratón es transmitir los movimientos de nuestra mano sobre una superficie plana hacia la computadora. Allí, el software denominado driver se encarga realmente de transformarlo a un movimiento del puntero por la pantalla dependiendo de varios parámetros.

En el momento de activar el ratón, se asocia su posición con la del cursor en la pantalla. Si desplazamos sobre una superficie el ratón, el cursor seguirá dichos movimientos. Es casi imprescindible en aplicaciones dirigidas por menús o entornos gráficos, como por ejemplo Windows. Hay cuatro formas de realizar la transformación y por tanto cuatro tipos de ratones:

- **Mecánicos.**

  Se basan en una bola de silicona que gira en la parte inferior del ratón a medida que desplazábamos éste. Dicha bola hace contacto con dos rodillos, uno perpendicular al ratón y otro transversal, de forma que uno recoge los movimientos de la bola en sentido horizontal y el otro en sentido vertical.

En cada extremo de los ejes donde están situados los rodillos, existe una pequeña rueda conocida como "codificador", que gira en torno a cada rodillo. Estas ruedas poseen en su superficie, y a modo de radios, una serie de contactos de metal, que a medida que gira la rueda toca con dos pequeñas barras fijas conectadas al circuito integrado en el ratón.

Cada vez que se produce contacto entre el material conductor de la rueda y las barras, se origina una señal eléctrica. Así, el número de se señales indicará la cantidad de puntos que han pasado éstas, lo que implica que, a mayor número de señales, mayor distancia habrá recorrido el ratón. Tras convertir el movimiento en señales eléctricas, se enviaban al software de la computadora por medio del cable.

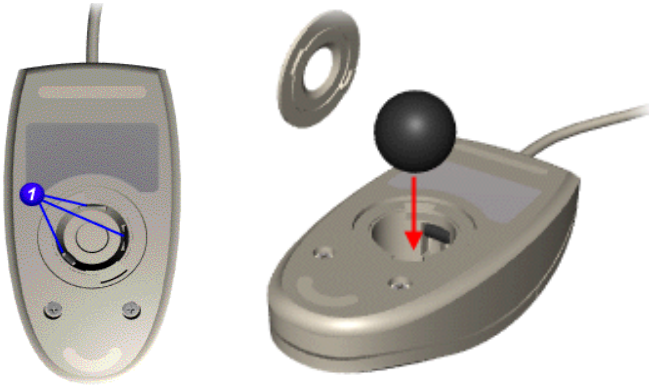


FIGURA IV. 3.
Bola y zonas de contacto con los rodillos.

Los botones son simples interruptores. Debajo de cada uno de ellos se encuentra un microinterruptor que en estado de "reposo" interrumpe un pequeño circuito. En cuanto se ejerce una ligera presión sobre estos, se activa el circuito, dejando pasar una señal eléctrica que será única en caso de que sólo se haga "clic" con el botón, o continua en caso de dejarlo pulsado.

Por último las señales se dan cita en el pequeño chip que gobierna el ratón, y son enviadas a la computadora a través del cable con los une. Allí el controlador del ratón decidirá, en función del desplazamiento vertical y horizontal detectado, el movimiento final que llevará el cursor. También será capaz de aumentar o disminuir ese movimiento, dependiendo de factores como la resolución que se le haya especificado al ratón.

FIGURA IV. 4.
Esquema general de un ratón mecánico.

- **Los ratones opto-mecánicos**.

  Trabajan según el mismo principio que los mecánicos, pero aquí los cilindros están conectados a codificadores ópticos que emplean pulsos luminosos a la computadora, en lugar de señales eléctricas. El modo de capturar el movimiento es distinto. Los tradicionales rodillos que giran una rueda radiada ahora pueden girar una rueda ranurada, de forma que un haz de luz las atraviesa. De esta forma, el corte intermitente del haz de luz por la rueda es recogido en el otro lado por una célula fotoeléctrica que decide hacia donde gira el ratón y a que velocidad

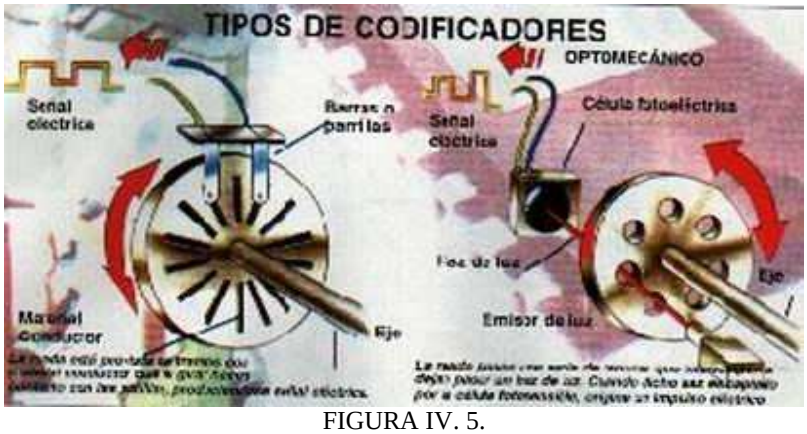


FIGURA IV. 5.
Codificadores del ratón.

- **Los ratones de ruedas.**

  Sustituyen la bola giratoria por unas ruedas de material plástico, perpendiculares entre sí, dirigiendo así a los codificadores directamente.

- **Los ratones ópticos**.

  Carecen de bola y rodillos, y poseen unos foto-sensores o sensores ópticos que detectan los cambios en los patrones de la superficie por la que se mueve el ratón. Antiguamente, estos ratones necesitaban una alfombrilla especial, pero actualmente no. Microsoft ha denominado a este sistema IntelliEye en su ratón IntelliMouse y es capaz de explorar el escritorio 1500 veces por segundo, sobre multitud de superficies distintas como madera plástico o tela. La ventaja de estos ratones estriba en su precisión y en la carencia de partes móviles, aunque son lógicamente algo más caros que el resto.

Una característica a tener en cuenta será la resolución, o sensibilidad mínima del sistema de seguimiento: en el momento en que el ratón detecte una variación en su posición, enviará las señales correspondientes a la computadora. La resolución se expresa en puntos por pulgada (ppp). Un ratón de 200 ppp podrá detectar cambios en la posición tan pequeños como 1/200 de pulgada, y así, por cada pulgada que se mueva el ratón, el cursor se desplazará 200 píxeles en la pantalla.

Una de las cosas que está cambiando es el medio de transmisión de los datos desde el ratón a la computadora. Se intenta evitar el empleo del cable que siempre conduce la información debido a las dificultades que añadía al movimiento. En la actualidad estos están siendo sustituidos por sistemas de infrarrojos o por ondas de radio.

Esta última técnica es mejor, pues los objetos de la mesa no interfieren la comunicación. Los dos botones o interruptores tradicionales han dejado evolucionado a multitud de botones, ruedas, y palancas que están dedicados a facilitar las tareas de trabajo con la computadora, sobre todo cuando se trabaja con Internet.

Hay modelos que no sólo tienen mandos que incorporan las funciones más comunes de los buscadores o navegadores, sino que tienen botones para memorizar las direcciones más visitadas por el usuario. Naturalmente, los fabricantes han aprovechado para poner botones fijos no configurables con direcciones a sus páginas.

La tecnología force-feedback consiste en la transmisión por parte de la computadora de pulsaciones a través del periférico. Podremos sentir diferentes sensaciones dependiendo de nuestras acciones. Por ejemplo, si nos salimos de la ventana activa, podremos notar que el ratón se opone a nuestros movimientos. Por supuesto, un campo también interesante para esto son los juegos.

En los juegos de golf, se podría llegar a tener sensaciones distintas al golpear la bola dependiendo de si esta se encuentra en arena, hierba, etc. Lamentablemente, este tipo de ratones si se encuentra estrechamente unido a alfombrillas especiales.

Existen dos tipos de conexiones para el ratón: Serie y PS/2. En la práctica no hay ventaja de un tipo de puerto sobre otro.

IV. 4. A. b. TECLADO.

El teclado es un dispositivo periférico de hardware utilizado para la introducción de datos y órdenes en un ordenador. Su diseño y estructura procede de las antiguas máquinas de escribir. Aunque no es muy valorado por los usuarios, el teclado debe ser de buena calidad para facilitar el uso del ordenador, sobre todo en el caso de que usemos habitualmente procesadores de texto o lenguajes de programación.

IV. 4. A. b. i. FUNCIONES DEL TECLADO.

El teclado funciona gracias a una estructura matricial, cada tecla está asociada a un código numérico, y es el software informático el que le aplica a ese código numérico un significado. Gracias a este sistema se puede utilizar un mismo teclado para diferentes idiomas, independientemente de los caracteres serigrafiados en él. El teclado esta dividido en 4 partes fundamentales, el teclado alfanumérico, el teclado de función, el teclado numérico y el teclado especial.

- **Teclado alfanumérico:** es un conjunto de 62 teclas entre las que se encuentran las letras, números, símbolos ortográficos, Enter, alt.etc.

- **Teclado de Función:** es un conjunto de 13 teclas entre las que se encuentran el ESC, tan utilizado en sistemas informáticos, más 12 teclas de función. Estas teclas suelen ser configurables pero por ejemplo existe un convenio para asignar la ayuda a F1.

- **Teclado Numérico:** se suele encontrar a la derecha del teclado alfanumérico y consta de los números así como de un Enter y los operadores numéricos de suma, resta, etc.

- **Teclado Especial:** son las flechas de dirección y un conjunto de 9 teclas agrupadas en 2 grupos; uno de 6 (Inicio y fin entre otras) y otro de 3 con la tecla de impresión de pantalla entre ellas.

IV. 4. A. b. ii. TIPOS DE TECLADO.

Algunos de los tipos de teclado son los siguientes:

- **De Membrana:** Fueron los primeros que salieron y como su propio nombre indica presentan una membrana entre la tecla y el circuito que hace que la pulsación sea un poco más dura.

- **Mecánico:** Estos nuevos teclados presentan otro sistema que hace que la pulsación sea menos traumática y más suave para el usuario.

- **Teclado para internet:** El nuevo Internet Keyboard incorpora 10 nuevos botones de acceso directo, integrados en un teclado estándar de ergonómico diseño que incluye un apoya manos. Los nuevos botones permiten desde abrir nuestro explorador Internet hasta hojear el correo electrónico. El software incluido, IntelliType Pro, posibilita la personalización de los botones para que sea el teclado el que trabaje como nosotros queramos que lo haga.

- **Teclados inalámbricos:** Pueden fallar si están mal orientados, pero no existe diferencia con un teclado normal. En vez de enviar la señal mediante cable, lo hacen mediante infrarrojos, y la controladora no reside en el propio teclado, sino en el receptor que se conecta al conector de teclado en el PC. Si queremos conectar a nuestro equipo un teclado USB, primero debemos tener una BIOS que lo soporte y en segundo lugar debemos tener instalado el sistema operativo con el "Suplemento USB". Un buen teclado USB debe tener en su parte posterior al menos un conector USB adicional para poderlo aprovechar como HUB y poder conectar a él otros dispositivos USB como ratones, altavoces, etc.

IV. 4. A. c. ESCÁNER.

Aparato digitalizador de imágenes. Por digitalizar se entiende la operación de transformar algo analógico (algo físico, real, de precisión infinita) en algo digital (un conjunto finito y de precisión determinada de unidades lógicas denominadas bits). El caso que nos ocupa se trata de coger una imagen (fotografía, dibujo o texto) y convertirla a un formato que podamos almacenar y modificar con la computadora. Realmente un escáner no es ni más ni menos que los ojos de la computadora.

IV. 4. A. c. i. FUNCIONAMIENTO.

El proceso de captación de una imagen resulta casi idéntico para cualquier escáner: se ilumina la imagen con un foco de luz, se conduce mediante espejos la luz reflejada hacia un dispositivo denominado CCD que transforma la luz en señales eléctricas, se transforma dichas señales eléctricas a formato digital en un DAC (conversor analógico-digital) y se transmite el caudal de bits resultante a la computadora.

El CCD (Charge Coupled Device, dispositivo acoplado por carga -eléctrica-) es el elemento fundamental de todo escáner, independientemente de su forma, tamaño o mecánica. Consiste en un elemento electrónico que reacciona ante la luz, transmitiendo más o menos electricidad según sea la intensidad y el color de la luz que recibe; es un auténtico ojo electrónico. Hoy en día es bastante

común, puede que usted posea uno sin saberlo: en su cámara de vídeo, en su fax, en su cámara de fotos digital.

La calidad final del escaneado dependerá fundamentalmente de la calidad del CCD; los demás elementos podrán hacer un trabajo mejor o peor, pero si la imagen no es captada con fidelidad cualquier operación posterior no podrá arreglar el problema.

Teniendo en cuenta lo anterior, también debemos tener en cuenta la calidad del DAC (Digital to Analog Conversion, Convertidor Digital Analógico), puesto que de nada sirve captar la luz con enorme precisión si perdemos mucha de esa información al transformar el caudal eléctrico a bits.

IV. 4. A. c. ii. TIPOS DE ESCÁNER.

- **Flatbed:** significa que el dispositivo de barrido se desplaza a lo largo de un documento fijo. En este tipo de escáneres, como las fotocopiadoras de oficina, los objetos se colocan boca abajo sobre una superficie lisa de cristal y son barridos por un mecanismo que pasa por debajo de ellos. Otro tipo de escáner flatbed utiliza un elemento de barrido instalado en una carcasa fija encima del documento.

- **Escáner de mano:** también llamado hand-held, porque el usuario sujeta el escáner con la mano y lo desplaza sobre el documento. Estos escáneres tienen la ventaja de ser relativamente baratos, pero resultan algo limitados porque no pueden leer documentos con una anchura mayor a 12 ó 15 centímetros.

- **Lector de código de barras:** dispositivo que mediante un haz de láser lee dibujos formados por barras y espacios paralelos, que codifica información mediante anchuras relativas de estos elementos. Los códigos de barras representan datos en una forma legible por la computadora, y son uno de los medios más eficientes para la captación automática de datos.

FIGURA IV.6
Escáner

IV. 4. A. d. WEBCAM.

Una cámara web en la simple definición, es una cámara que esta simplemente conectada a la red o internet.

En la Webcam radica un concepto sencillo; tenga en funcionamiento continuo una cámara de video, obtenga un programa para captar un imagen en un archivo cada determinados segundos o minutos, y cargue el archivo de la imagen en un servidor Web para desplegarla en una página Web.

Unos de los tipos más comunes de cámaras personales que están conectadas a las computadoras del hogar, funcionando con la ayuda de algunos programas usuarios comparten una imagen en movimiento con otros. Dependiendo del usuario y de los programas, estas imagines pueden ser publicadas disponibles en el internet por vía de directorios especificados, o algunos disponibles a los amigos de usuarios que ahora poseen la propia dirección para conectarse. Estas cámaras son típicamente solo cuando los usuarios de las computadoras están encendidos y conectados a Internet.

IV. 4. A. d. i. TIPOS DE CÁMARAS.

- **Cámara de fotos digital:** Toma fotos con calidad digital, casi todas incorporan una pantalla LCD (Liquid Cristal Display: Pantalla de Cristal Liquido) donde se puede visualizar la imagen obtenida. Tiene una pequeña memoria donde almacena fotos para después transmitirlas a una computadora.

- **Cámara de video:** Graba videos como si de una cámara normal se tratara, pero las ventajas que ofrece en estar en formato digital, es que es mucho mejor la imagen, tiene una pantalla LCD por la que ves simultáneamente la imagen mientras grabas. Se conecta al PC y este recoge el video que has grabado, para poder retocarlo posteriormente con el software adecuado.

IV. 4. A. e. LÁPIZ ÓPTICO.

Dispositivo señalador que permite sostener sobre la pantalla (fotosensible) un lápiz que está conectado a la computadora con un mecanismo de resorte en la punta o en un botón lateral, mediante el cual se puede seleccionar información visualizada en la pantalla. Cuando se dispone de información desplegada, con el lápiz óptico se puede escoger una opción entre las diferentes alternativas, presionándolo sobre la ventana respectiva o presionando el botón lateral, permitiendo de ese modo que se proyecte un rayo láser desde el lápiz hacia la pantalla fotosensible.

El lápiz contiene sensores luminosos y envía una señal a la computadora cada vez que registra una luz, por ejemplo al tocar la pantalla cuando los píxeles no negros que se encuentran bajo la punta del lápiz son refrescados por el haz de electrones de la pantalla.

La pantalla de la computadora no se ilumina en su totalidad al mismo tiempo, sino que el haz de electrones que ilumina los píxeles los recorre línea por línea, todas en un espacio de 1/50 de segundo. Detectando el momento en que el haz de electrones pasa bajo la punta del lápiz óptico, la computadora puede determinar la posición del lápiz en la pantalla. El lápiz óptico no requiere una

pantalla ni un recubrimiento especiales como puede ser el caso de una pantalla táctil, pero tiene la desventaja de que sostener el lápiz contra la pantalla durante periodos largos de tiempo llega a cansar al usuario.

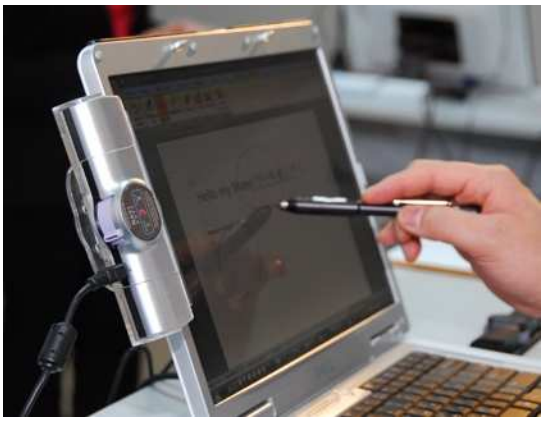


FIGURA IV. 7.
Lápiz Óptico para monitor.

IV. 4. A. f. JOYSTICK.

Palanca que se mueve apoyada en una base. Se trata, como el ratón, de un manejador de cursor. Consta de una palanca con una rótula en un extremo, que permite efectuar rotaciones según dos ejes perpendiculares. La orientación de la palanca es detectada por dos medidores angulares perpendiculares, siendo enviada esta información a la computadora. Un programa adecuado convertirá los ángulos de orientación de la palanca en desplazamiento del cursor sobre la misma.

Principalmente existen dos diferentes tipos de joystick: los analógicos y los digitales. Para la construcción de uno analógico se necesitan dos potenciómetros, uno para la dirección X y otro para la dirección Y, que dependiendo de la posición de la palanca de control producen un cambio en la tensión a controlar. Contienen además un convertidor tensión / frecuencia que proporciona los pulsos que se mandan por el puerto según la señal analógica de los potenciómetros. Los digitales no contienen elementos analógicos para obtener las señales de control, sino que los movimientos son definidos por el software de control que incluirá el dispositivo en cuestión.

IV. 4. A. f. i. SISTEMA DE CONEXIÓN.

Van conectados al puerto juegos de la placa, al de la tarjeta de sonido, al puerto o puertos de una tarjeta de juegos, o eventualmente, al puerto serie o paralelo. Aunque la opción del puerto de la tarjeta de sonido es con mucho la más utilizada por ahorro de recursos.

IV. 4. A. f. ii. TECNOLOGÍA.

Aquí dependiendo del tipo de joystick que estemos hablando (palanca, joypad, volante, etc) la tecnología utilizada es variopinta. A pesar de ello es útil optar por mandos robustos y que ofrezcan buen soporte de software. Los basados en tecnología digital son ideales para los que requieran precisión.

Muchos joystick permiten de forma sencilla y simplemente mediante el uso de un cable especial (en forma de Y), la utilización de dos dispositivos simultáneos.

Posibles problemas: Lo más frecuente son los provenientes de la mala configuración del software. Estos dispositivos necesitan ser instalados y calibrados mediante los programas incluidos antes de poder ser utilizados.

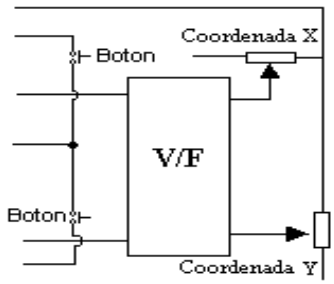


FIGURA IV. 8.
Diagrama de un joystick analógico.

IV. 4. A. g. LECTORES DE TARTETAS INTELIGENTES.

Como el propio nombre lo indica, un lector de tarjetas es una interfaz que permite la comunicación entre una tarjeta y otro dispositivo. Los terminales se diferencian unos de otros en la conexión con la computadora, la comunicación con la tarjeta y el software que poseen.

FIGURA IV. 9.
Lectores de Tarjeta Inteligentes

IV. 4. A. g. i. TIPO DE LECTORES PARA CADA APLICACIÓN.

Existe lectores de tarjetas inteligentes para cada aplicación y los podemos dividir en:

- **Lectores conectados a un PC:** Como su nombre lo indica son lectores fabricados para ser usados conectándolo a un computador, esta conexión puede ser a través de un puerto serie, usb, pcmcia, etc.

- **Lectores conectados a un equipo específico:** Son lectores que se pueden instalar (previo fabricación y diseño) en un aparato determinado para cumplir con una función. Estos lectores se pueden instalar en:

  - Cajeros automáticos, máquinas expendedoras, parquímetros, puertas (control de acceso), motores, etc.

- **Lectores Portátiles:** Son equipos que no necesitan de otro aparato para cumplir su función. Generalmente poseen todos los recursos integrados como baterías, memoria, pin pad, etc.

Los lectores vienen en muchas formas, factores y capacidades. La forma más fácil de describir un lector de tarjetas inteligentes es por el método de su interfaz a la PC. Los lectores de tarjetas se utilizan para leer y escribir datos en las tarjetas inteligentes, también pueden ser fácilmente integrados a una PC utilizando las diferentes plataformas de sistemas operativos.

IV. 4. A. g. ii. TIPOS DE LECTORES SEGÚN EL MÉTODO DE INTERFAZ.

- **Lector de contacto:** Se nombra así a los lectores donde las tarjetas deben ser insertadas en la ranura del lector para poder operar con ellas. A través de estos contactos el lector alimenta eléctricamente a la tarjeta y transmite los datos oportunos para operar con ella conforme al estándar.

FIGURA IV. 10.
Contactos del chip de una tarjeta con contactos.

La serie de estándares ISO/IEC 7816 e ISO/IEC 7810 definen:

- La forma física (parte 1).
- La posición de las formas de los conectores eléctricos (parte 2).
- Las características eléctricas (parte 3).
- Los protocolos de comunicación (parte 3).
- El formato de los comandos (ADPU's) enviados a la tarjeta y las respuestas retornadas por la misma.
- La dureza de la tarjeta.
- La funcionalidad.

➢ **Lector sin contacto:** Son lectores en donde las tarjetas con etiquetas RFID (identificación por radio frecuencia - radio frequency identification) se comunican de al lector a través de inducción a una tasa de transferencia de 106 a 848 Kb/s).

El estándar de comunicación de tarjetas inteligentes sin contacto es el ISO/IEC 14443 del 2001. Define dos tipos de tarjetas sin contacto (A y B), permitidos para distancias de comunicación de hasta 10 cm. Ha habido propuestas para la ISO 14443 tipos C, D, E y F que todavía tienen que completar el proceso de estandarización. Un estándar alternativo de tarjetas inteligentes sin contacto es el ISO 15693, el cual permite la comunicación a distancias de hasta 50 cm.

IV. 4. B. DISPOSITIVOS DE SALIDA.

Son los que permiten representar los resultados (salida) del proceso de datos. El dispositivo de salida típico es la pantalla o monitor. Otros dispositivos de salida son: impresoras (imprimen resultados en papel), trazadores gráficos (plotters), bocinas, entre otros.

IV. 4. B. a. MONITOR O PANTALLA.

Es el dispositivo en el que se muestran las imágenes generadas por el adaptador de vídeo del ordenador o computadora. El término monitor se refiere normalmente a la pantalla de vídeo y su carcasa. El monitor se conecta al adaptador de vídeo mediante un cable. Evidentemente, es la pantalla en la que se ve la información suministrada por la computadora. En el caso más habitual se trata de un aparato basado en un tubo de rayos catódicos (CRT) aunque últimamente han tomado gran auge las pantallas planas de cristal líquido (LCD).

FIGURA IV. 11.
Pantalla plana de cristal liquido.

IV. 4. B. b. IMPRESORAS.

Como indica su nombre, la impresora es el periférico que la computadora utiliza para presentar información impresa en papel.

Si queremos clasificar los diversos tipos de impresoras que existen, el método más lógico es hacerlo atendiendo a su tecnología de impresión, es decir, al método que emplean para imprimir en el papel, e incluir en dicha clasificación como casos particulares otras consideraciones como el uso de color, su velocidad, etc. Eso nos lleva a los tres tipos clásicos: matriciales, de tinta y láser.

IV. 4. B. b. i. IMPRESORAS MATRICIALES.

Fueron las primeras que surgieron en el mercado. Se les denomina "de impacto" porque imprimen mediante el impacto de unas pequeñas piezas (la matriz de impresión) sobre una cinta impregnada en tinta, la cual suele ser fuente de muchos problemas si su calidad no es la deseable.

Según cómo sea el cabezal de impresión, se dividen en dos grupos principales: de margarita y de agujas. Las de margarita incorporan una bola metálica en la que están en relieve las diversas letras y símbolos a imprimir; la bola pivota sobre un soporte móvil y golpea a la cinta de tinta, con lo que se imprime la letra correspondiente. El método es absolutamente el mismo que se usa en muchas máquinas de escribir eléctricas, lo único que las diferencia es la carencia de teclado.

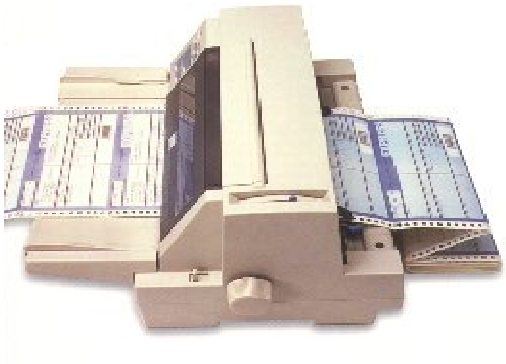


FIGURA IV. 12.
Impresoras de impacto (matriciales).

- **Las impresoras de margarita:** otros métodos que usan tipos fijos de letra están en completo desuso debido a que sólo son capaces de escribir texto; además, para cambiar de tipo o tamaño de letra deberíamos cambiar la matriz de impresión (la bola) cada vez. Por otra parte, la calidad del texto y la velocidad son muy altas, además de que permiten obtener copias múltiples en papel de autocopia o papel carbón.

- **Las impresoras de agujas:** muchas veces denominadas simplemente matriciales, tienen una matriz de pequeñas agujas que impactan en el papel formando la imagen deseada; cuantas más agujas posea el cabezal de impresión mayor será la resolución, que suele estar entre 150 y 300 ppp, siendo casi imposible superar esta última cifra.

Aunque la resolución no sea muy alta es posible obtener gráficos de cierta calidad, si bien en blanco y negro, no en color. El uso de color implica la utilización de varias cintas más anchas, además de ser casi imposible conseguir una gama realista de colores, más allá de los más básicos.

En general, las impresoras matriciales de agujas se posicionan como impresoras de precio reducido, calidad media-baja, escaso mantenimiento y alta capacidad de impresión.

IV. 4. B. b. ii. IMPRESORAS DE TINTA.

Por supuesto, las impresoras matriciales utilizan tinta, pero cuando nos referimos a impresora de tinta nos solemos referir a aquéllas en las que la tinta se encuentra en forma más o menos líquida, no impregnando una cinta como en las matriciales.

FIGURA IV. 13.
Cartuchos para impresoras de tinta.

La tinta suele ser impulsada hacia el papel por unos mecanismos que se denominan inyectores, mediante la aplicación de una carga eléctrica que hace saltar una minúscula gota de tinta por cada inyector, sin necesidad de impacto. Estas impresoras se destacan por la sencilla utilización del color. La resolución de estas impresoras es en teoría bastante elevada, hasta de 1.440 ppp, pero en realidad la colocación de los puntos de tinta sobre el papel resulta bastante deficiente, por lo que no es raro encontrar que el resultado de una impresora láser de 300 ppp sea mucho mejor que el de una de tinta del doble de resolución. Por otra parte, suelen existir papeles especiales, mucho más caros que los clásicos folios de papelería, para alcanzar resultados óptimos a la máxima resolución o una gama de colores más viva y realista.

FIGURA IV. 14.
Impresoras de tinta.

IV. 4. B. b. iii. IMPRESORAS LÁSER.

Son las de mayor calidad del mercado, si entendemos por calidad la resolución sobre papel normal que se puede obtener, unos 600 ppp reales. En ellas la impresión se consigue mediante un láser que va dibujando la imagen electrostáticamente en un elemento llamado tambor que va girando hasta impregnarse de un polvo muy fino llamado tóner (como el de fotocopiadoras) que se le adhiere debido a la carga eléctrica. Por último, el tambor sigue girando y se encuentra con la hoja, en la cual imprime el tóner que formará la imagen definitiva.

FIGURA IV. 15.
Impresora Laser.

El único problema de importancia de las impresoras láser es que sólo imprimen en blanco y negro. En realidad, sí existen impresoras láser de color, que dan unos resultados bastante buenos, pero su precio es absolutamente desorbitado.

Las impresoras láser son muy resistentes, mucho más rápidas y mucho más silenciosas que las impresoras matriciales o de tinta, y aunque la inversión inicial en una impresora láser es mayor que en una de las otras, el tóner sale más barato a la larga que los cartuchos de tinta, por lo que a la larga se recupera la inversión. Por todo ello, las impresoras láser son idóneas para entornos de oficina con una intensa actividad de impresión, donde son más importantes la velocidad, la calidad y el escaso coste de mantenimiento que el color o la inversión inicial.

CAPÍTULO V.

TARJETA

INTELIGENTE

V. 1. HISTORIA.

Debido a la avanzada tecnología que se presenta en el mundo se hace necesario que constantemente se éste en evolución y aprovechando las ventajas que está nos ofrece, en cualquiera de los servicios donde se aplique. Hasta ahora, la banda magnética de las tarjetas de crédito y de débito, ha sido la tecnología dominante en el mercado; sin embargo, en ellas sólo se puede almacenar una pequeña cantidad de información, de modo que la gran mayoría de los datos personales y de las operaciones de la tarjeta magnética, residen en servidores centrales de la compañía que las emite.

Con una tarjeta inteligente, toda la información necesaria para las transacciones está alojada en el microprocesador insertado en la misma, lo que significa que el tráfico de información es mucho menor con respecto al de las tarjetas de banda magnética, incrementándose así el nivel de seguridad de las operaciones.

Con las tarjetas inteligentes se puede operar desde un simple control de acceso del personal a una empresa o escuela, hasta complejas combinaciones que pueden incluir la información personal del usuario, su historial clínico y algún sistema de cliente frecuente, incluyendo servicios financieros como monedero electrónico o tarjetas de débito y de crédito.

Las tarjetas inteligentes cada vez son más utilizadas. Los niveles de seguridad y la capacidad de almacenamiento que manejan, han llevado a los bancos y a otras instituciones financieras a reemplazar poco a poco sus tarjetas convencionales de banda magnética por tarjetas de chip. La posibilidad de almacenar y procesar información en este sofisticado y diminuto mecanismo, facilita la realización de procesos y permite administrar la información de mejor manera.

V. 2. ¿QUÉ SON LAS TARJETAS INTELIGENTES?

Es bastante frecuente denominar a todas las tarjetas que poseen contactos dorados o plateados sobre su superficie, como tarjetas inteligentes. Sin embargo, este término es bastante ambiguo y conviene hacer una clasificación más correcta. ISO (International Standard Organization) prefiere usar el término “tarjeta de circuito integrado” (Integrated Circuit Card o ICC), para referirse a todas aquellas tarjetas que posean algún dispositivo electrónico. Este circuito contiene elementos para realizar transmisión, almacenamiento y procesamiento de datos. La transferencia de datos puede llevarse a cabo a través de los contactos, que se encuentran en la superficie de la tarjeta, o sin contactos por medio de campos electromagnéticos.

A continuación vamos a ver un poco de los orígenes de la tarjeta inteligente.

V. 3. APARICIÓN DE LAS TARJETAS INTELIGENTES.

La tarjeta inteligente surge ante nuevas necesidades del mercado, las cuales no pueden ser satisfechas por la tarjeta de banda magnética. Esta tecnología tiene su origen en la década de los 70´s del siglo pasado cuando inventores de Alemania, Japón y Francia inscribieron las patentes originales. Debido a varios factores que se presentaron, y de los cuales la inmadura tecnología de semiconductores tuvo un mayor peso, muchos trabajos sobre tarjetas inteligentes (smart cards) estuvieron en investigación y desarrollo hasta la primera mitad de los años ochenta.

A continuación mostraremos la aparición cronológica de esta tecnología:

- Tarjetas en década de 1950. Primero sólo plástico, luego banda magnética.
- J. Dethloff y H. Grotrupp en 1968: Circuito integrado incorporado a tarjeta.
- K. Arimura en 1970: Integración de lógica aritmética y almacén en chip.
- R. Moreno en 1974: Patente actual, vendida a Bull.
- Primer prototipo en 1979.
- Primeras tarjetas telefónicas en 1983.
- Primeras tarjetas de débito en 1984.
- Estándares ISO (ref 7816-X) en 1987.
- Primera versión de especificación EMV para aplicaciones financieras en 1994.
- Primer monedero electrónico en 1998.
- Primeras "tarjetas Java" en 1999.
- 2003 Primer tarjeta de ICC Windows 2003 powered.
- 2007 Primer tarjeta de ICC para SunRay Workstation.
- Especificación para aplicaciones financieras (última actualización en 2008).

V. 4. TARJETAS DE BANDA MAGNÉTICA.

La tarjeta magnética convencional se desarrolló a finales de los 60's del siglo XX para satisfacer varias necesidades. Una de ellas es permitir a los clientes de los bancos y entidades de ahorro activar y operar de forma rápida y efectiva con los cajeros automáticos. También, para proporcionar un medio con el que operar en puntos de venta específicos.

El objetivo de esta tarjeta es identificar a un cliente para acceder a una base de datos remota con la que se establece una conexión. La información que posee la base de datos permite aceptar o rechazar esa transacción.

FIGURA V. 1.
Tarjeta de Banda Magnética.

En la actualidad, la utilización de la tarjeta magnética se ha generalizado de tal forma que, al año, se producen y utilizan una media de 1,400 millones de tarjetas magnéticas en el mundo.

Las tarjetas magnéticas han producido importantes resultados en el mercado financiero pero no ofrecen soluciones para los nuevos mercados y servicios que aparecen: televisión interactiva, telefonía digital, etc.

El problema se debe a que las tarjetas magnéticas actuales se han utilizado para dar solución a problemas que aparecieron hace 25 años y están ligados a esas tecnologías: dependencias de ordenadores centrales y grandes redes dedicadas, a diferencia de los sistemas distribuidos actuales y de las nuevas soluciones. Además, la tarjeta magnética ofrece muy baja densidad de datos, baja fiabilidad y poca o ninguna seguridad en la información que lleva.

V. 5. TIPOS DE TARJETAS INTELIGENTES.

Dado que el acceso a la información se realiza a través de un puerto serie y supervisado por el propio sistema operativo de la tarjeta, es posible escribir datos confidenciales que no puedan ser leídos por personas no autorizadas. En principio, las funciones de escritura, lectura y borrado de la memoria pueden ser controladas tanto por el hardware como por el software, o por ambos a la vez. Esto permite una gran variedad de mecanismos de seguridad. Siendo el chip integrado el componente más importante, las tarjetas están clasificadas según el tipo de circuito.

V. 5. A. TARJETA INTELIGENTE DE CONTACTO.

La mayoría de las tarjetas poseen en su superficie 8 contactos, los cuales representan el único interfaz electrónico existente entre la tarjeta y el terminal lector. Todas las señales eléctricas circulan a través de estos contactos, sin embargo se reservan dos contactos para un uso futuro.

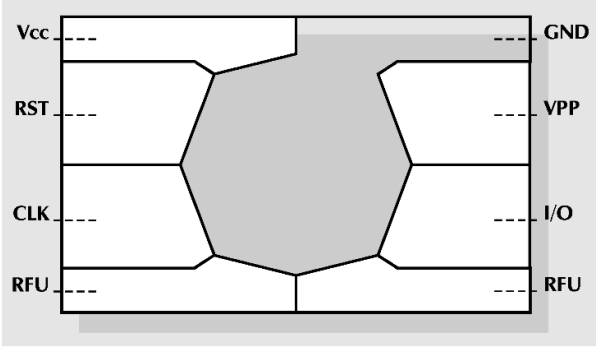


FIGURA V. 4.
Contactos del circuito integrado de la tarjeta inteligente.

Donde:

- **Vcc** - el chip es similar a un PC. Precisa, por tanto, de energía para funcionar. Vcc es el "toma de corriente" del chip. La energía la proporciona el dispositivo hardware con el que la tarjeta interactúa en cada operación.
- **RST** - es el mecanismo que pone en funcionamiento la interrelación entre una tarjeta inteligente y cualquier elemento Externo adecuado con el que se ponga en contacto.
- **CLK** - el "reloj" determina la velocidad de funcionamiento de la tarjeta.
- **RFU** - no tiene asignadas funciones por el momento.
- **GND** - la "tierra" de la "toma de corriente".
- **VPP** – Voltaje externo para programar la memoria de la tarjeta.
- **I/O** - punto de entrada y salida de la información.
- **RFU** - no tiene asignadas funciones por el momento.

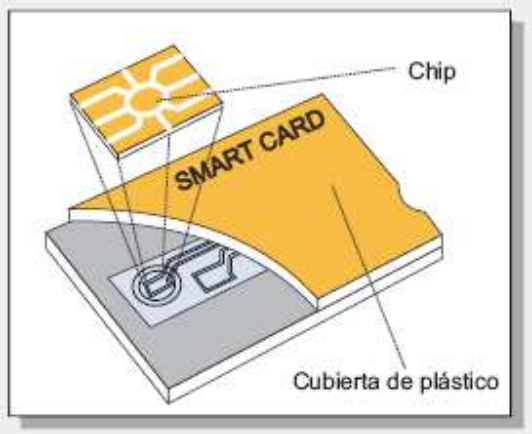


FIGURA V. 2.
Tarjeta Inteligente.

Estas tarjetas presentan bastantes ventajas en comparación con las de bandas magnéticas:

- Son capaces de almacenar más información.
- Pueden proteger la información que almacenan en sus memorias de posibles accesos no autorizados.

Estas tarjetas son las que necesitan ser insertadas en una terminal con lector inteligente para que por medio de contactos pueda ser leída. Existen dos tipos de tarjeta inteligente de contacto: Las sincrónicas y las asincrónicas.

V. 5. A. a. TARJETAS INTELIGENTES SINCRÓNICAS O TARJETAS DE MEMORIA.

Los datos que se requieren para las aplicaciones con tarjetas de memoria son almacenados en una EEPROM.

Estas tarjetas son desechables cargadas previamente con un monto o valor que va decreciendo a medida que se utiliza y una vez que se acaba el monto se vuelve desechable.

- **Memoria Libre:** Carece de mecanismos de protección para acceder a la información. Las funciones que desempeñan están optimizadas para aplicaciones particulares en las que no se requieren complejos mecanismos de seguridad. Se utilizan para el pago de peajes, teléfonos públicos, maquinas dispensadoras y espectáculos.

- **Memoria Protegida:** Poseen un circuito de seguridad que proporciona un sistema para controlar los accesos a la memoria frente a usuarios no autorizados. Este sistema funciona mediante el empleo de un código de acceso que puede ser de 64 bits o más.

V. 5. A. b. TARJETAS INTELIGENTES ASINCRÓNICAS.

Estas tarjetas poseen en su chip un microprocesador, que además cuenta con algunos elementos adicionales como son:

- ROM enmascarada.
- EEPROM.
- RAM.
- Un puerto de Entrada/Salida.

La memoria ROM enmascarada contiene el sistema operativo de la tarjeta, y se graba durante el proceso de fabricación.

La EEPROM es la memoria no volátil del microprocesador, y en ella se encuentran datos del usuario o de la aplicación, así como el código de las instrucciones que están bajo el control del sistema operativo. También puede contener información como el nombre del usuario, número de identificación personal o PIN (Personal Identification Number: Número de Identificación Personal).

La RAM es la memoria de trabajo del microprocesador. Al ser volátil se perderá toda la información contenida en ella al desconectar la alimentación.

El puerto de entrada y salida normalmente consiste en un simple registro, a través del cual la información es transferida bit a bit. Las tarjetas con microprocesador son bastantes flexibles, puesto que pueden realizar bastantes funciones. En el caso más simple, sólo contienen datos referentes a una aplicación específica, esto hace que dicha tarjeta solo se pueda emplear para esa aplicación, sin embargo, los sistemas operativos de las tarjetas más modernas hacen posible que se puedan integrar programas para distintas aplicaciones en una sola tarjeta. En este caso la ROM contiene sólo el

sistema operativo con las instrucciones básicas, mientras que el programa específico de cada aplicación se graba en la EEPROM después de la fabricación de la tarjeta.

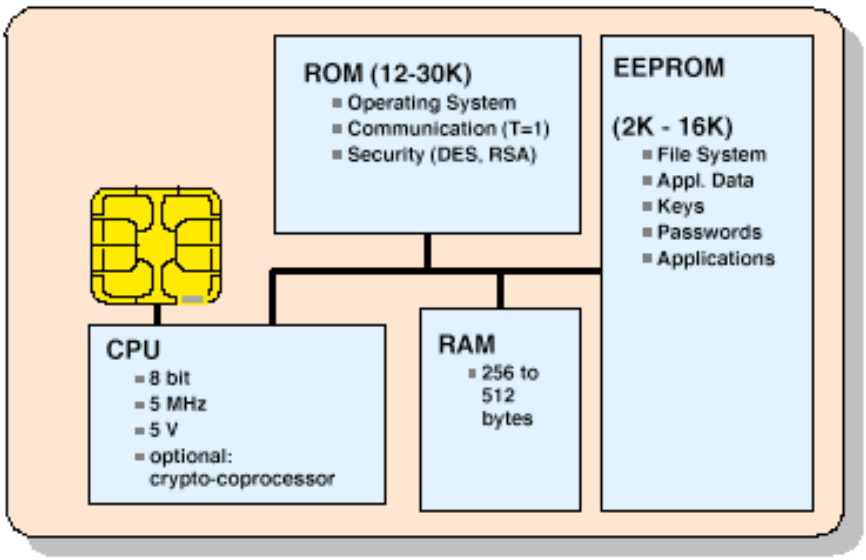


FIGURA V. 3.
Elementos Adicionales del Microprocesador.

V. 5. B. TARJETAS INTELIGENTES SIN CONTACTO.

Son similares a las de contacto con respecto a lo que pueden hacer y a sus funciones pero utilizan diferentes protocolos de transmisión en capa lógica y física, no utilizan contacto galvanico sino de interfase inductiva. Poseen además del chip, una antena de la cual se valen para realizar transacciones. Son ideales para las transacciones que tienen que ser realizadas muy rápidamente.

Esta tecnología ofrece ventajas con respecto a la de las tarjetas de contacto. Cuando en una tarjeta de contactos se producen fallos de funcionamiento, casi siempre se deben al deterioro en la superficie de contacto o a la suciedad adherida a los mismos. Una de las ventajas de las tarjetas sin contactos es que los problemas técnicos antes mencionados no ocurren, debido claro está, a que carecen de contactos. Otra de las ventajas es la de no tener que introducir la tarjeta en un lector. Esto es una gran ventaja en sistemas de control de accesos donde se necesita abrir una puerta u otro mecanismo, puesto que la autorización de acceso puede ser revisada sin que se tenga que sacar la tarjeta del bolsillo e introducirla en un terminal.

Este tipo de tarjetas se comunican por medio de radiofrecuencias. Según la proximidad necesaria entre tarjeta y lector, existen dos tipos:

- Tarjeta cercana: debe estar a unos pocos milímetros del lector para que sea posible la comunicación.

- Tarjeta lejana: la distancia varía entre centímetros y unos pocos metros.

Desde el punto de vista de cómo se alimentan, existen dos tipos:

- Uno en el cual la tarjeta incorpora junto al chip una batería que alimenta a los circuitos.
- Otro tipo que incorpora un hilo metálico incrustado. Este hilo se somete a un campo electromagnético variable que a su vez induce una corriente eléctrica capaz de alimentar los circuitos de la tarjeta.

V. 5. C. TARJETAS INTELIGENTES HÍBRIDAS Y DUALES.

Una tarjeta híbrida es una tarjeta sin contacto a la cual se le agrega un segundo chip de contacto. Ambos chips pueden ser o chips microprocesadores o simples chips de memoria. El chip sin contacto es generalmente usado en aplicaciones que requieren transacciones rápidas. Por ejemplo el transporte, mientras que el chip de contacto es generalmente utilizado en aplicaciones que requieren de alta seguridad como las bancarias.

Una tarjeta de interfaz dual es similar a la tarjeta híbrida en que la tarjeta presenta ambas interfaces con y sin contacto. La diferencia más importante es el hecho de que la tarjeta de interfaz dual tiene un solo circuito integrado.

V. 5. D. TARJETAS SÚPER INTELIGENTES.

Este tipo de tarjetas cumplen las mismas funciones que las tarjetas inteligentes con microprocesador pero también están equipadas con un teclado, una pantalla LCD y una pila. Esta tarjeta no tiene necesidad de usar una terminal para la comunicación con la computadora.

V. 5. D. a. VENTAJAS.

A continuación se mencionan algunas de las ventajas de las tarjetas súper inteligentes:

- Gran capacidad de memoria.
- Altos niveles de seguridad.
- Reducción del fraude.
- Información organizada.
- Confiabilidad.
- Alto manejo de información.
- Seguridad en la información.
- Facilidad de usos sin necesidad de conexiones en línea o vía telefónica.

- Comodidad para el usuario.
- A través de Internet los usuarios de tarjetas inteligentes podrán comprar por computador y pagar por red.
- Garantizar operaciones económicas, 100% efectivas y a prueba de robos.
- Caída de los costos para empresarios y usuarios.
- Estándares específicos ISO 7810, 7811, 9992, 10536.
- Tarjetas inteligentes multiservicio.
- Privacidad.
- Administración y control de pagos más efectivo.

V. 6. FASES DE VIDA DE LA TARJETA INTELIGENTE.

Existen diferentes fases en la vida de las tarjetas inteligentes, tales como la fabricación, distribución, empleo y caducidad, a su vez estas fases encierran procesos como los que a continuación se mencionan:

Fabricación:

- Desarrollo del SO y su implementación como mascara ROM.
- Producción industrial del chip.

Distribución:

- Inicialización y pre-personalización de la tarjeta según uso futuro.
- Envío al expendedor de la tarjeta.

Empleo:

- Personalización.

Caducidad:

- Uso.
- Fin de la vida activa.

V. 6. A. PROCESO DE FABRICACIÓN.

La fabricación de tarjetas inteligentes abarca normalmente los siguientes pasos:

- **Fabricación del chip**, o muchos chips en una oblea. Varios miles de chips de circuito integrado se fabrican a la vez en la forma de obleas de silicio con aproximadamente 3.000 a 4.000 unidades.

- **Empaquetado de los chips.** individuales para su inserción en una tarjeta. Una vez que se termina una oblea, cada chip se prueba individualmente, se divide la oblea y se realizan las conexiones eléctricas del chip.

- **Fabricación de la tarjeta.** La tarjeta está compuesta de cloruro de polivinilo o de un material similar. Las características químicas y las dimensiones de la tarjeta y sus tolerancias son reguladas por estándares internacionales. El material de la tarjeta se produce en una hoja grande, plana del grosor prescrito. Para muchos tipos de tarjetas producidas en serie, estas hojas entonces se imprimen con los elementos gráficos comunes a todas las tarjetas. Las tarjetas individuales entonces se cortan de esta hoja plana y los bordes de cada tarjeta se lijan.

- **Inserción del chip en la tarjeta.** Una vez que el chip y la tarjeta estén preparados, los dos se unen: se hace un agujero en la tarjeta, y el chip se pega en él con pegamento.

- **Pre-personalización.** Una vez la tarjeta está completa, la mayoría de los usos inteligentes de la misma requieren que ciertos ficheros de los programas o de datos estén instalados en cada chip (tarjeta) antes de que la tarjeta se pueda personalizar para un titular específico. Esta preparación general del software o de los archivos en la tarjeta se hace con una operación llamada la pre-personalización, que se hace a través de los contactos del chip y por lo tanto puede proceder solamente a la velocidad proporcionada por esa interfaz.

- **Personalización.** El procedimiento de la personalización implica el poner de la información tal como nombres, perfiles o números de cuenta en la tarjeta; a partir de la realización de este proceso la tarjeta está asignada a una persona en particular. Normalmente esta personalización será gráfica (estampando o troquelando datos personales del titular sobre la superficie plástica de la tarjeta) y/o eléctrica (grabando información personal del titular en algún fichero de la tarjeta).

V. 7. ESTÁNDARES.

Todas las características que definen a una tarjeta inteligente desde su aspecto físico hasta la manera de transferir la información, están definidas por diferentes normas que a continuación presentamos.

ISO: normas que afectan a las características más básicas, incluso físicas, de los componentes de las tarjetas Inteligentes.

EMV: los tres operadores internacionales más importantes de medios de pago están ultimando el desarrollo de estándares que se refieren al proceso transaccional.

CEN: normativa del Comité Europeo de Normalización que tiene por objeto el funcionamiento de las aplicaciones. La homogeneidad de este aspecto posibilita la convivencia de diferentes utilidades en una misma tarjeta.

V. 7. A. LOS ESTÁNDARES ISO.

La tarjeta inteligente más básica cumple los estándares de la serie ISO 7816, partes 1 a 10. Este estándar detalla la parte física, eléctrica, mecánica y la interfaz de programación para comunicarse con el microchip.

La descripción de cada una de las partes de la ISO 7816 es:

- **7816-1**: Características Físicas.
- **7816-2**: Dimensiones y ubicaciones de los contactos.
- **7816-3**: Señales Electrónicas y Protocolo de Transmisión.
- **7816-4**: Comandos de intercambio inter-industriales.
- **7816-5**: Sistema de Numeración y procedimiento de registración.
- **7816-6**: Elementos de datos inter-industriales.
- **7816-7**: Comandos inter-industriales y Consultas Estructuradas para una Tarjeta.
- **7816-8**: Comandos inter-industriales Relacionados con Seguridad.
- **7816-9**: Comandos adicionales inter-industriales y atributos de seguridad.
- **7816-10**: Señales electrónicas y Respuesta al Reset para una Smart Card Síncrona.

V. 7. A. a. CARACTERÍSTICAS FÍSICAS.

El rasgo más distintivo de una tarjeta es sin duda su aspecto físico. Otra característica notable a simple vista es la presencia o no del área de contactos, que tiene la forma de un cuadrado dorado o plateado, y que se encuentra en la superficie de la tarjeta. En algunos casos esta área no existe (tarjetas sin contacto).

Existe una íntima relación entre el cuerpo de la tarjeta y el chip que lleva implantado dentro, de nada sirve que el cuerpo de la tarjeta sea capaz de soportar temperaturas extremas, si el microprocesador no comparte esa característica. Ambos componentes deben de satisfacer todos los requisitos tanto por separado como conjuntamente.

Por ejemplo la tarjeta conforme con ISO 7810, 7813, debe :

- Resistir ataques con rayos X y luz Ultravioleta.
- Tener superficie plana.
- Permitir cierto grado de torsión.
- Resistir altos voltajes, campos electromagnéticos, electricidad estática.
- No disipar más de 2,5 W.

Existen tres tamaños en la clasificación de de las tarjetas inteligentes:

- ID-1 El tamaño más habitual, tamaño tarjeta de crédito.
- ID-00 Un tamaño intermedio poco utilizado comercialmente.
- ID-000 Usadas para teléfonos móviles. También acostumbran a tener este formato las tarjetas SAM utilizadas para la autenticación criptográfica mutua de tarjeta y terminal.

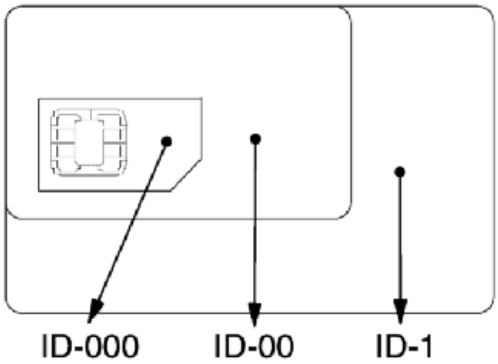


FIGURA V. 5.
Tamaños de Tarjeta.

V. 7. A. b. DIMENSIÓN Y LOCALIZACIÓN DE LOS CONTACTOS.

Dado que el microprocesador requiere de una vía por donde tomar la alimentación para sus circuitos o para llevar a cabo la trasmisión de datos, es necesaria una superficie física de contacto que haga de enlace entre el lector y la tarjeta.

Esta superficie consiste en 8 contactos que se encuentran en una de las caras de la tarjeta. El tamaño de los contactos no deber ser nunca inferior a1.7 mm de alto y 2 mm para el ancho, el valor máximo de estas medidas no está especificado.

V. 7. .A. c. SEÑALES ELECTRÓNICAS Y PROTOCOLOS DE TRANSMISIÓN.

Toda comunicación que se realice con una tarjeta es iniciada siempre por el dispositivo externo, esto quiere decir que la tarjeta nunca transmite información sin que se haya producido antes una petición externa. Esto equivale a una relación maestro-esclavo, siendo la terminal el maestro y la tarjeta el esclavo.

Cada vez que se inserta una tarjeta en la terminal lector, sus contactos se conectan a dicha terminal y ésta procede a activarlos eléctricamente, a continuación, la tarjeta inicia un reset de encendido y envía una respuesta llamada ATR (Answer To Reset: Respuesta para el Reinicio) hacia el terminal. Esta respuesta contiene información referente a cómo ha de ser la comunicación tarjeta-lector, estructura de los datos intercambiados, protocolo de transmisión, etc.

Una vez que el lector interpreta el ATR procede a enviar la primera instrucción. La tarjeta procesa la orden y genera una respuesta que es enviada hacia la terminal. El intercambio de instrucciones y respuestas acaba una vez que la tarjeta es desactivada.

Entre la respuesta ATR y la primera orden enviada, la terminal puede transmitir una instrucción de selección del tipo de protocolo o PTS (Protocol Type Selection). Esto sucede cuando la tarjeta especifica más de un protocolo en la respuesta al reset y el terminal no sabe cuál ha de usar.

Todos los datos enviados por la línea de comunicación son digitales y usan los valores 0’s y 1’s. Los valores de tensión usados son 0 y 5 voltios. Parte de la información contenida en la ATR tiene la misión de informar sobre qué valor de tensión se va a aplicar a cada digito binario.

Existen dos convenios: el de la lógica directa que asigna 5 Voltios al “1” lógico, y 0 Voltios al “0” lógico, y el de lógica inversa que asigna 5 Voltios al “0”, y 0 Voltios al “1” lógico.

Los protocolos de transmisión especifican con precisión cómo han de ser las instrucciones, las respuestas a las mismas y el procedimiento a seguir en caso de que se produzcan errores durante la transmisión.

V. 8. SISTEMA OPERATIVO Y ESTRUCTURA DE FICHEROS.

En contraste con los sistemas operativos conocidos, los sistemas basados en tarjetas inteligentes no permiten al usuario el almacenamiento externo de información, siendo las prioridades más importantes la ejecución segura de los programas y el control de acceso a los datos.

V. 8. A. SISTEMA OPERATIVO.

El sistema operativo basado en tarjetas inteligentes no permite el uso de “puertas traseras”, que son bastante frecuentes en sistemas grandes. Esto quiere decir que es imposible hacer una lectura desautorizada de los datos contenidos usando el código propio de la tarjeta.

Debido a la restricción de memoria, la cantidad de información que se puede grabar es bastante pequeña. Los módulos de programa se graban en la ROM, lo cual posee la desventaja de no permitir al usuario programar el funcionamiento de la tarjeta según sus propios criterios, ya que una vez grabado el sistema operativo es imposible realizar ningún cambio. Por esto el programa grabado en la memoria ROM debe ser bastante fiable y robusto.

Existen otras funciones que desempeña el código almacenado en la ROM:

- Transmisión de datos desde y hacia la tarjeta.
- Control de la ejecución de los programas.
- Administración de los datos.
- Manejo y administración de algoritmos criptográficos.

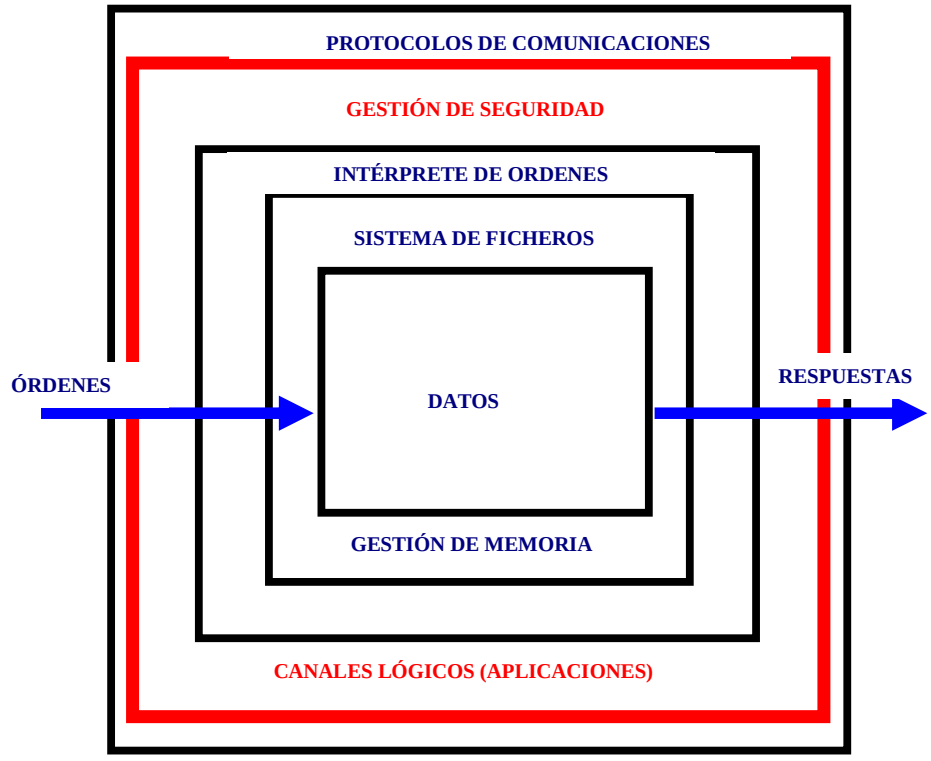


FIGURA V. 6.
Estructura del Sistema Operativo de las Tarjetas Inteligentes.

V. 8. B. ESTRUCTURA DE FICHEROS.

La estructura de los ficheros contenidos en una tarjeta inteligente es similar a los sistemas DOS y UNÍX. Existen varios directorios que hacen las funciones de carpetas que contienen ficheros. Los tipos de ficheros existentes son:

- Fichero Maestro (MF):
  - Es el directorio raíz y es seleccionado de manera automática después de iniciar la tarjeta. En él están contenidos todos los directorios y ficheros. El fichero maestro representa a toda la memoria disponible en la tarjeta para almacenar datos.
- Fichero Dedicado (DF):
  - Situado debajo del MF.
  - Es un directorio que puede contener ficheros o incluir a otros DF. El nivel de anidamiento es infinito pero ha de restringirse debido a la escasez de memoria.
- Fichero Elemental (EF):
  - Situado debajo del MF o de un DF.
  - Los datos de usuario necesarios para la aplicación están almacenados en estos ficheros.

V. 9. TARJETAS INTELIGENTES Y SEGURIDAD.

Una de las razones principales por las cuales las tarjetas inteligentes existen es la seguridad. La tarjeta por sí misma provee una plataforma de cómputo en la cual se puede almacenar información y su procesamiento se puede realizar de forma segura. Además la tarjeta inteligente es altamente portátil y fácil de llevar, por lo tanto, es ideal para funcionar como un medio para mejorar la seguridad de otros sistemas.

Cada fichero de la tarjeta lleva asociadas unas condiciones de acceso y deben ser satisfechas antes de ejecutar un comando sobre ese fichero. En el momento de personalización de la tarjeta (durante su fabricación) se pueden indicar qué mecanismos de seguridad se aplican a los ficheros. Normalmente se definirán de la forma siguiente:

- Ficheros de acceso libre.
- Ficheros protegidos por claves: Pueden definirse varias claves con distintos propósitos. Normalmente se definen claves para proteger la escritura de algunos ficheros y claves específicas para los comandos de consumo y carga de las aplicaciones de monedero electrónico. De ese modo la aplicación que intente ejecutar comandos sobre ficheros protegidos tendrá que negociar previamente con la tarjeta la clave oportuna.
- Ficheros protegidos por PIN: El PIN es un número secreto que va almacenado en un fichero protegido y que es solicitado al usuario para acceder a este tipo de ficheros protegidos. Cuando el usuario lo introduce y el programa procesa la operación que abre el fichero en cuestión el sistema valida que el PIN sea correcto para dar acceso al fichero.

Finalmente, indicar que la negociación de claves se realiza habitualmente apoyándose en un Módulo SAM (Serial Access memory: memoria de acceso serial), que no deja de ser otra cosa más que una tarjeta inteligente en formato ID-000 alojada en un lector interno propio dentro de la carcasa del lector principal o del TPV(terminal de punto de venta) y que contiene aplicaciones criptográficas que permiten negociar las claves oportunas con la tarjeta inteligente del usuario. Operando de este modo se está autenticando el lector, la tarjeta y el módulo SAM involucrados en cada operación.

V. 10. TARJETAS INTELIGENTES Y SU PROGRAMACIÓN.

Al aproximarse a la programación de tarjetas inteligentes hay que distinguir dos ámbitos claramente diferenciados, la programación de aplicaciones para el chip de la tarjeta, es decir, las aplicaciones que se almacenan y ejecutan dentro del chip de la tarjeta cuando ésta recibe alimentación eléctrica de un lector. Otro es la programación de aplicaciones para los sistemas en los que se utiliza la tarjeta, esto es, aplicaciones que se ejecutan en ordenadores (en un sentido genérico, ya que pueden ser TPV's empotrados, cajeros automáticos, PC's de escritorio, etc.) a los que se conecta un lector de tarjetas en el que se inserta (o aproxima si es un lector sin contactos) una tarjeta inteligente. Estas aplicaciones se comunican con el lector, el cual se comunica con la tarjeta y sus aplicaciones.

V. 10. A. PROGRAMACIÓN DE APLICACIONES PARA EL CHIP DE LA TARJETA.

Este tipo de programación es de muy bajo nivel y depende normalmente del tipo y proceso de fabricación de las propias tarjetas. En la mayoría de las tarjetas inteligentes el sistema operativo de la tarjeta y las aplicaciones que van dentro del chip se cargan en el propio proceso de fabricación y no pueden ser luego modificadas una vez que la tarjeta ha sido fabricada.

Una excepción clara a este caso pueden ser las Java Cards (tarjetas JAVA), que son tarjetas que en el proceso de fabricación incorporan un sistema operativo y una máquina virtual Java específica para este entorno. Una vez fabricada la tarjeta, los desarrolladores pueden implementar mini-aplicaciones (applets) Java para ser cargadas en la tarjeta (mediante un procedimiento que garantice la seguridad del sistema).

V. 10. B. PROGRAMACIÓN DE LAS APLICACIONES PARA LOS SISTEMAS.

Existen varias API´s de programación estandarizadas para comunicarse con los lectores de tarjetas inteligentes desde un ordenador. Las principales son:

- **PC/SC** (Personal Computer/Smart Card:Computadora Personal/Tarjeta inteligente), especificado por el PC/SC Workgroup. Existe una implementación para Microsoft Windows y también el proyecto MUSCLE proporciona una implementación casi completa de esta especificación para los sistemas operativos GNU Linux-UNIX.

- OCF (OpenCard Framework: Entorno de trabajo OpenCard), especificado por el grupo de empresas OpenCard. Este entorno intenta proporcionar un diseño orientado a objetos fácilmente extensible y modular. El consorcio OpenCard publica el API (Interface de programación de aplicaciones) y proporciona una implementación de referencia en Java. Existe un adaptador para que OCF trabaje sobre PC/SC.

En ambos casos, el modelo de programación que utilizan las tarjetas inteligentes está basado en protocolos de petición-respuesta. La tarjeta (su software) expone una serie de comandos que pueden ser invocados. Estos comandos interactúan con los ficheros que subyacen a cada aplicación de la tarjeta y proporcionan un resultado. Desde la terminal se invocan estos comandos a través de cualquiera de las API´s antes descritas componiendo APDUs (Application Protocol Data: Protocolo de unidades de aplicación de datos) que son enviados a la tarjeta para que ésta responda.

V. 11. APLICACIONES DE LAS TARJETAS INTELIGENTES.

Las aplicaciones fundamentales de las tarjetas inteligentes son:

- **Identificación** del titular de la misma.
- **Pago** electrónico de bienes o servicios mediante dinero virtual.
- **Almacenamiento seguro** de información asociada al titular.

Las tarjetas inteligentes también son muy utilizadas como un monedero electrónico. Estas aplicaciones disponen normalmente de un fichero protegido que almacena un contador de saldo y comandos para decrementar e incrementar el saldo (esto último sólo con unas claves de seguridad especiales, obviamente). Con esta aplicación, el chip de la tarjeta inteligente puede ser 'cargado' con dinero los que pueden ser utilizados en parquímetros, máquinas expendedoras u otros mercados. Protocolos criptográficos protegen el intercambio de dinero entre la tarjeta inteligente y la máquina receptora.

Cuando las tarjetas son criptográficas las posibilidades de identificación y autenticación se multiplican ya que se pueden almacenar de forma segura certificados digitales o características biométricas en ficheros protegidos dentro de la tarjeta de modo que estos elementos privados nunca salgan de la tarjeta y las operaciones de autenticación se realicen a través del propio chip criptográfico de la tarjeta.

V. 11. A. PROGRAMAS DE FIDELIZACIÓN.

Cada vez más programas de fidelización en sectores como líneas aéreas, hoteles, restaurantes de comida rápida y grandes almacenes, utilizan las tarjetas inteligentes, que registran los puntos y premios, y que ofrecen datos detallados sobre los hábitos y experiencias de los clientes a los operadores de dichos programas, a fin de elaborar sus campañas de marketing con mayor precisión.

V. 11. B. MONEDERO ELECTRÓNICO.

Las empresas que utilizan la tecnología de las tarjetas inteligentes como una alternativa al manejo de dinero en efectivo son cada día más. La seguridad de éste sistema de pago se basa en dos elementos básicos: el hardware del chip de la tarjeta y el software que controla los movimientos del valor de las tarjetas, es decir, el dinero que tienen guardado.

Este sistema es parecido a las tarjetas de débito, pero el monedero electrónico permite a los negocios aceptar inmediata y directamente los pagos sin requerir autorización, como si se tratara de dinero en efectivo.

Los negocios que aceptan este sistema de pago, tienen instalada una terminal lectora de tarjetas, donde se inserta el monedero electrónico. Una pantalla en la terminal despliega el importe total de la compra y muestra que la operación de pago con la tarjeta se está realizando.

Tan pronto como se comprueba que el usuario o comprador tuvo suficiente efectivo en su tarjeta, la pantalla de la terminal mostrará, en un parpadeo, que la transacción está completada. El dinero exacto del monto de la compra pasó de la tarjeta del usuario, al lector de la tarjeta instalado en la terminal del negocio.

V. 11. C. GRUPOS CERRADOS.

Las tarjetas inteligentes están muy extendidas entre los grupos cerrados de usuarios (residentes de una ciudad, personal de una universidad, personal de una compañía, aficionados de un equipo deportivo, clientes de un parque de atracciones, etc.), los cuales pueden utilizar tarjetas con múltiples aplicaciones para pagar sus cuotas o acceder a servicios (por ejemplo, descuento en compras, entrada para partidos, máquinas expendedoras, credenciales de biblioteca, parques de atracciones, estacionamientos, etc.), de forma ilimitada de acuerdo con la autorización y circunstancias personales .

V. 11. D. TARJETAS DE ASISTENCIA MÉDICA (CLÍNICAS Y SEGUROS).

Las tarjetas inteligentes o smarts cards, pueden almacenar expedientes médicos, información para casos de emergencia y situación en materia de seguros de enfermedad.

V. 11. E. DOCUMENTOS Y CREDENCIALES (SEGURIDAD Y CONTROL DE ACCESO).

Los gobiernos que deseen reducir costos y papeleo pueden utilizar las tarjetas inteligentes para emitir licencias de conducir, pasaportes y visas. Las smart cards pueden programarse para permitir el acceso a edificios o datos, dependiendo del cargo y del nivel de autorización.

V. 11. F. DÉBITO-CRÉDITO.

En muchos países, las versiones para cinta magnética de éstas aplicaciones residen conjuntamente con las tarjetas inteligentes, pero las nuevas normas EMV (Euro Pay, MasterCard y Visa) implican que los adquirientes deben actualizar sus redes, tarjetas y terminales. EMV establece el uso de las tarjetas inteligentes en sustitución de las tarjetas de banda magnética.

V. 12. CONFIGURACIÓN PARA EL USO DE LA TARJETA INTELIGENTE.

Para realizar la comunicación con tarjetas inteligentes debemos tomar en cuenta una serie de configuraciones, así como la instalación de algunas aplicaciones que permitirán trabajar libremente este tipo de dispositivos.

V. 12. A. API DE COMUNICACIONES.

El API (aplication Programming Interface: interfaz de programación de aplicaciones) de comunicaciones es una extensión standard que nos permite realizar interacciones con los puertos serie RS-232 y el paralelo IEEE-1284, esto nos ayudará a realizar aplicaciones de comunicación que utilizan los puertos (lectores de tarjetas inteligentes, fax, lectores de barras) independientes de la plataforma.

El API de comunicaciones no se encuentra incluido en el JDK, mejor dicho es una extensión de este, así que antes realizar cualquier comunicación con los puertos debemos instalar este API en las computadoras que vayan a ejecutar programas realizados con esta aplicación.

Una vez instalado el API obtendremos un fichero con el paquete de comunicación llamado javax.comm que soportará los driver de puertos, todo esto a través de una serié de clases que nos permitirán manejar diferentes niveles de programación como los siguientes:

- **Nivel alto**: En este nivel tenemos las clases CommPortIdentifier y CommPort que nos permitirán el acceso a los puertos de comunicación.

- **Nivel medio**: Las clases SerialPort y ParallelPort se cubrirán las interfaces físicas de RS-232 para el puerto serie y IEEE-1284 para el puerto paralelo.

- **Nivel bajo**: En este nivel se interactúa directamente con el sistema operativo y en él se encuentra el desarrollo de los drivers.

Además de los niveles de programación que esta aplicación nos permite manejar podemos realizar las siguientes acciones:

- Obtener los puestos disponibles así como sus características.

- Abrir y mantener una comunicación en los puertos.

- Resolver colisiones entre aplicaciones. Gracias a este servicio podremos tener varias aplicaciones Java funcionando y utilizando los mismos puertos

De esta manera y a través las clases contenidas en javax.comm disponemos de métodos para el control de los puertos de entrada/salida a bajo nivel, de esta forma no solo nos limitamos a enviar y recibir datos sino que podemos saber en que estado se encuentra el puerto. Así mismo en un puerto serie podremos no solo cambiar los estados sino que podemos programar un evento que nos notifique el cambio de cualquier estado.

V. 12. B. COMUNICACIÓN CON LA TARJETA INTELIGENTE.

Primeramente antes de intentar cualquier comunicación con la tarjeta a través del puerto se debe revisar si dicho puerto tiene o no un propietario, para obtener esta información se debe obtener el objeto de la función CommPortIdentifier correspondiente al puerto a través de alguno de los siguientes métodos:

- **getPortIdentifiers():** es el método utilizado en nuestro programa y que nos entregará un enumerado con tantos objetos CommPortIdentifier como puertos dispongamos.

- **getPortIdentifier(String):** Nos dará el objeto correspondiente al puerto que se le otorgue como parámetro, este será el método que normalmente usaremos ya que lo normal es que siempre nos conectemos por el mismo puerto. Este método deberá saber manejar la excepción NoSuchPortException que será utilizada en el caso de que solicitemos un puerto inexistente.

Una vez que se obtuvo el objeto del puerto se tiene una serie de métodos que permitirán obtener información del puerto y abrirlo para poder iniciar una comunicación. Estos métodos son los siguientes:

- **getName():** Dará el nombre del puerto. En el caso de haber utilizado el método

- **getPortIdentifier(String):** Será el mismo valor que otorgamos como parámetro.

- **getPortType():** Devuelve un entero que nos informa del tipo de puerto (serie o paralelo), para no tener que estar recordando el valor de cada tipo de puerto disponemos de las constantes PORT_PARALLEL y PORT_SERIAL.

- **isCurrentlyOwned():** Este método nos informa si el puerto esta libre o no. En caso de que este ocupado podremos saber quien lo está utilizando mediante el método getCurrentOwner().

- **open(String, int):** Abre y por lo tanto reserva un puerto, en el caso de que intentemos abrir un puerto que ya se encuentre en uso se utilizará la excepción PortInUseException.

Los parámetros que se proporcionan para la realización de las acciones anteriores son:

- Un **String** con el nombre de la aplicación que reserva el puerto.
- Un **int** que indica el tiempo de espera para abrir el puerto.

Una vez que se tiene este objeto se deberán orientar sus salidas a OutputStream y las entradas a InputStream, ahora la escritura y lectura de los puertos será tan fácil como utilizar los métodos de estas clases que pertenecen al standard del JDK.

- **addPortOwnershipListener(CommPortOwnershipListener):** permite añadir una clase que escuche los cambios de propietarios en los puertos.
- **setSerialPortParams(int, int, int, int):** permite configurar los parámetros del puerto serie, este método deberá manejar la excepción UnsupportedCommOperationException que será utilizada en el caso de que introduzcamos valores no soportados.

Los parámetros de este método son:

- La velocidad en el caso de proporcionar un valor no valido.
- Bit de datos. Para indicar el valor utilizaremos las constantes de la clase que se tienen (DATABITS_5, DATABITS_6, DATABITS_7 o DATABITS_8).
- El bit o bits de stop que utilizaremos y que puede ser 1,2 o uno y medio. Las constantes que definen estas configuraciones son: STOPBITS_1, STOPBITS_2 y STOPBITS_1_5 respectivamente.
- Paridad que utilizaremos y que puede ser PARITY_NONE en el caso de no utilizar paridad, PARITY_ODD para la paridad impar, PARITY_EVEN paridad par, PARITY_MARK paridad por marca y PARITY_SPACE paridad por espacio.

CAPÍTULO VI.

MANUAL

Este manual describe el procedimiento a seguir para la administración del programa LA FONTAINE.

Primero se accede a la computadora y se localiza el icono en el escritorio con el nombre passwd1.java y se dará doble clic con el botón izquierdo del Mouse.

FIGURA VI. 1.
Icono de passwd1.java para acceso
al programa LA FONTAINE.

Posterior a esta acción se abrirá una ventana del programa JCreator, en la cual seleccionaremos de la barra de menú la opción Run y se dará clic en la operación Run Project.

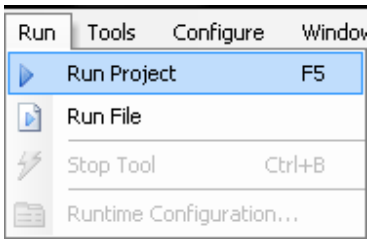


FIGURA VI. 2.
Ubicación de la operación Run Project
en el programa Jcreator.

A continuación tecleamos el nombre de usuario y password y se dará clic en el botón Aceptar.

FIGURA VI. 3.
Ventana de AUTENTIFICACION DE USUARIO.

El sistema mostrará un cuadro de diálogo indicando que el usuario ha sido autentificado y se dará clic en el botón Aceptar.

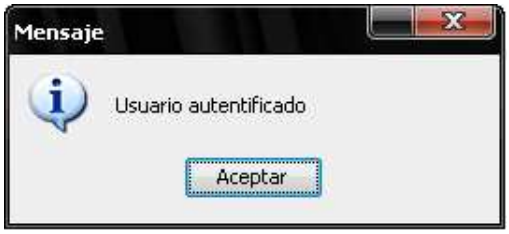


FIGURA VI. 4.
Cuadro de diálogo de Usuario autentificado.

Una vez autentificado el usuario y el password, el sistema muestra la siguiente ventana principal del programa:

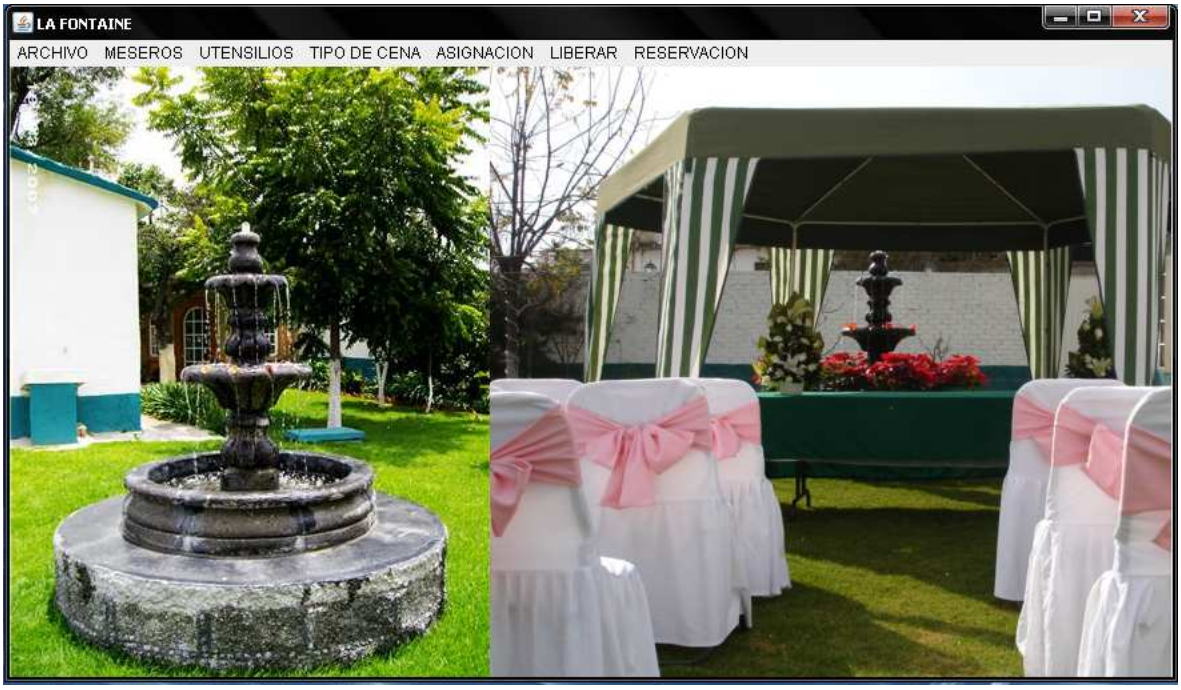


FIGURA VI. 5.
Ventana principal del programa LA FONTAINE.

La primera opción que se muestra en la barra de menú es ARCHIVO, aquí podremos realizar la opción de abandonar el programa en el momento que se desee con la operación SALIR.

FIGURA VI. 6.
Operación SALIR del programa.

Para agregar un mesero a nuestra base de datos, seleccionamos la opción MESEROS, a continuación se dará un clic en la operación NUEVO MESERO.

FIGURA VI. 7.
Operación NUEVO MESERO.

El sistema mostrará la siguiente ventana:

FIGURA VI. 8.
Ventana de ALTA DE MESEROS.

En esta ventana se llenarán los campos con los datos correspondientes del mesero que se desea dar de alta, y se dará clic en el botón Aceptar, a continuación el sistema nos mostrará un cuadro de diálogo indicando que los Datos fueron insertados correctamente y se volverá a dar clic en el botón Aceptar, finalmente para salir de la ventana se dará clic en el botón cerrar.

FIGURA VI. 9.
Insertar Datos de un nuevo mesero.

Para modificar el perfil de un mesero, seleccionamos la opción MESEROS y se dará clic en la operación MODIFICAR MESERO.

FIGURA VI. 10.
Operación MODIFICAR MESERO.

El sistema mostrará la siguiente ventana:

FIGURA VI. 11.
Ventana de MODIFICACION DE MESEROS.

En la ventana seleccionamos la opción ver datos y el sistema buscará los meseros que ya están dados de alta y los desplegará.

FIGURA VI. 12.
Buscar datos de un mesero.

Seleccionamos el mesero a modificar y el sistema nos desplegará los datos correspondientes del mesero, a continuación el sistema nos mostrará un cuadro de diálogo indicando que el Mesero ha sido encontrado y se dará clic en el botón Aceptar.

FIGURA VI. 13.
Cuadro de diálogo de Mesero encontrado.

En seguida se llenan los campos con los datos bloqueados del mesero, para poder modificarlos, se dará clic en el botón Modificar, se hacen los cambios correspondientes y se dará clic en el botón Aceptar, el sistema desplegará un cuadro de diálogo notificando que los datos fueron actualizados correctamente, se saldrá de la ventana dando clic en el botón cerrar.

FIGURA VI. 14.
Desbloquear los datos para poder modificarlos.

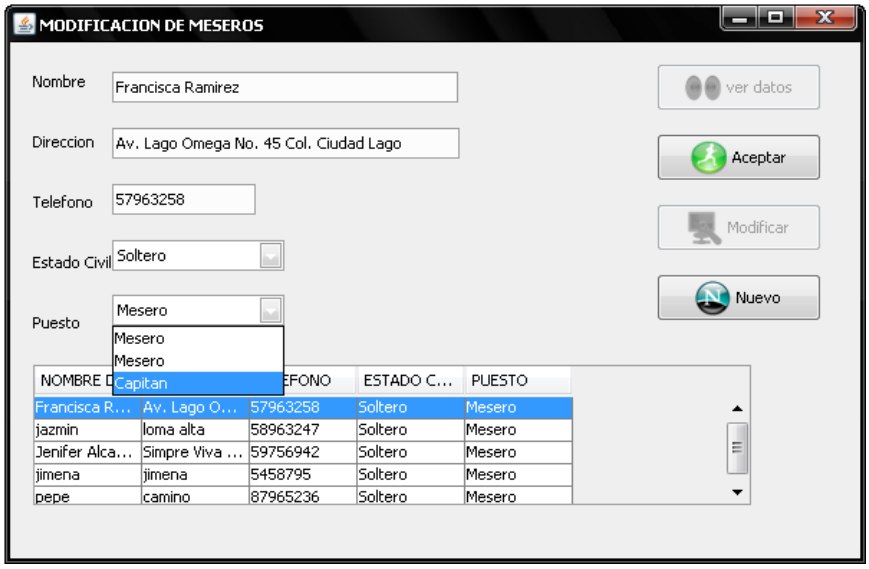


FIGURA VI. 15.
Modificando el Puesto del mesero.

FIGURA VI. 16.
Cuadro de diálogo de Datos actualizados correctamente.

Para eliminar a un mesero, seleccionamos la opción MESEROS, a continuación se dará un clic en la operación ELIMINAR MESERO.

FIGURA VI. 17.
Operación ELIMINAR MESERO.

El sistema mostrará la siguiente ventana:

FIGURA VI. 18.
Ventana de BAJA DE MESEROS.

Se dará clic en el botón Ver datos, el sistema buscará los meseros registrados en la base de datos y los desplegará.

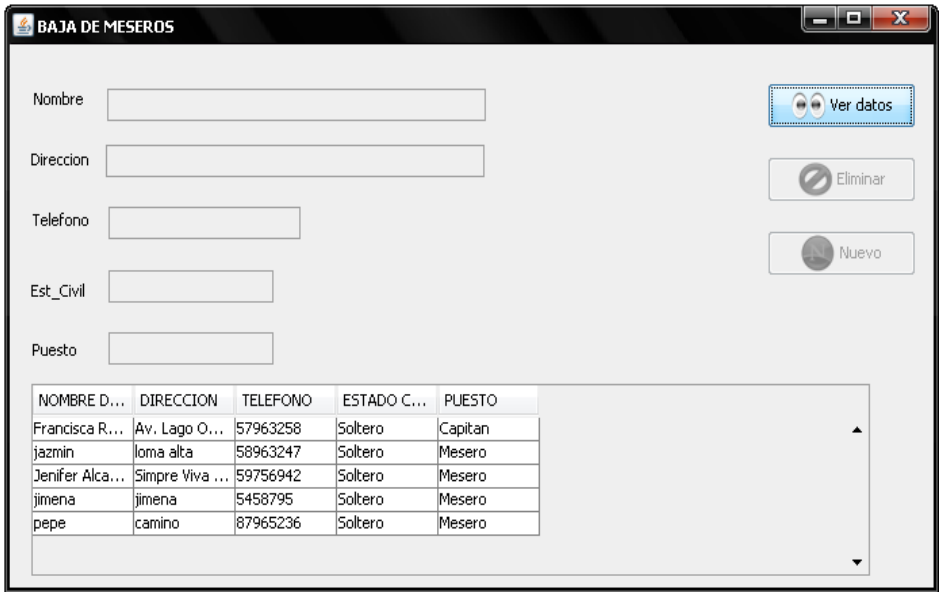


FIGURA VI. 19.
Buscar un mesero.

A continuación seleccionamos el mesero a modificar y el sistema nos desplegará los datos correspondientes del mesero, también nos mostrará un cuadro de diálogo notificando que el Mesero ha sido encontrado y se dará clic en el botón Aceptar.

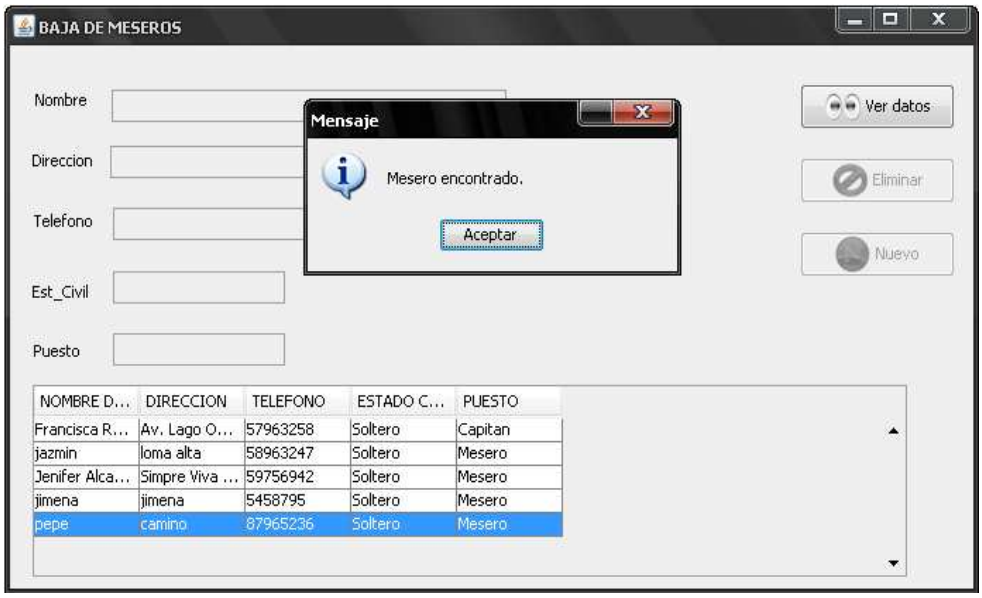


FIGURA VI. 20.
Encontrar un mesero.

Una vez encontrado el mesero se dará clic en el botón Eliminar, el sistema mostrara un cuadro de diálogo indicando la eliminación de los datos, el mesero será eliminado de la base de datos y finalmente se saldrá de la ventana dando clic en el botón cerrar.

FIGURA VI. 21.
Cuadro de diálogo de Datos eliminados.

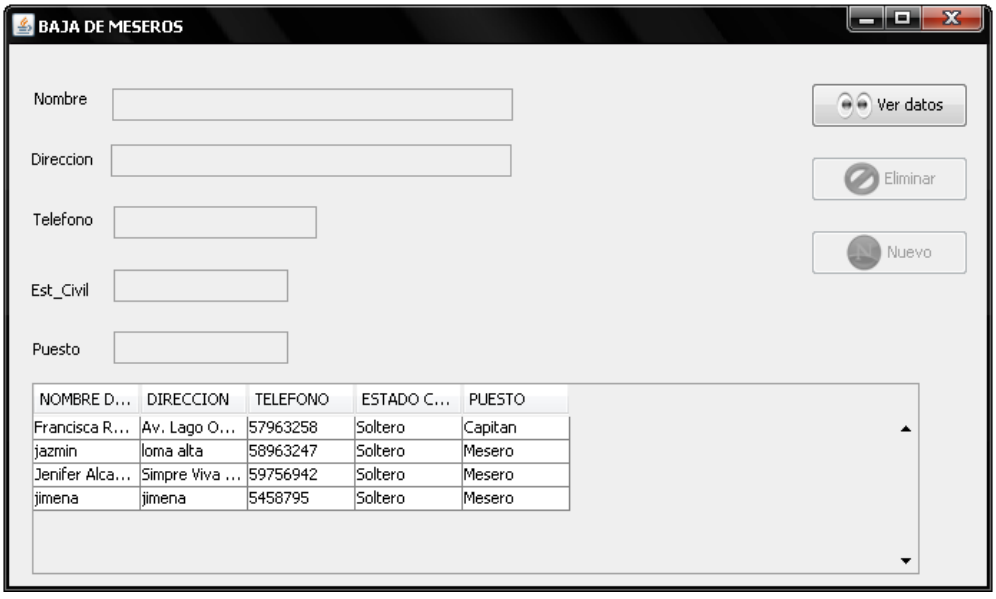


FIGURA VI. 22.
Mesero Eliminado de la base de datos.

Para agregar un utensilio, seleccionamos la opción UTENSILIOS, a continuación se dará un clic en la operación NUEVO UTENSILIO.

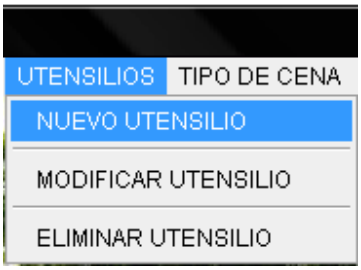


FIGURA VI. 23.
Operación NUEVO UTENSILIO.

El sistema mostrará la siguiente ventana:

FIGURA VI. 24.
Ventana de Alta de utensilios.

A continuación se llenan los campos con los datos correspondientes del utensilio que se desea dar de alta, y se dará clic en el botón Aceptar, en seguida el sistema nos mostrara un cuadro de diálogo indicando que los Datos fueron Insertados correctamente, se volverá a dar clic en el botón Aceptar y finalmente para salir se dará clic en el botón cerrar.

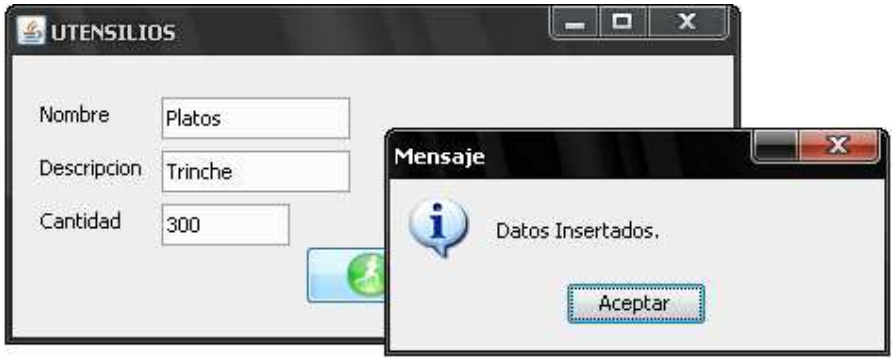


FIGURA VI. 25.
Insertar Datos de un utensilio.

Para modificar un utensilio se selecciona la opción UTENSILIOS, posteriormente se dará un clic en la operación MODIFICAR UTENSILIOS.

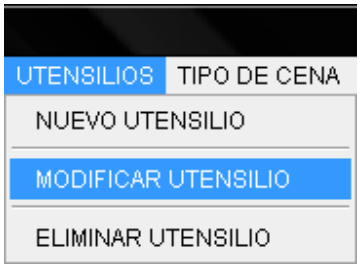


FIGURA VI. 26.
Operación MODIFICAR UTENSILIOS.

El sistema mostrará la siguiente ventana:

FIGURA VI. 27.
Ventana MODIFICACION DE UTENSILIOS.

Se dará clic en el botón Ver datos, posterior a esta selección, el sistema buscará los utensilios registrados en la base de datos y los desplegará.

FIGURA VI. 28.
Buscar Datos de un utensilio.

A continuación seleccionamos el utensilio a modificar, el sistema nos desplegará los datos bloqueados correspondientes al utensilio, para poder modificarlos, se dará clic en el botón Modificar, posterior a esta acción se harán los cambios correspondientes, se dará clic en el botón Aceptar, el sistema desplegara un cuadro de diálogo notificando que los datos han sido actualizados correctamente y para salir de la ventana se dará clic en el botón cerrar.

FIGURA VI. 29.
Desbloquear los datos para poder modificarlos.

FIGURA VI. 30.
Modificando Descripción del utensilio.

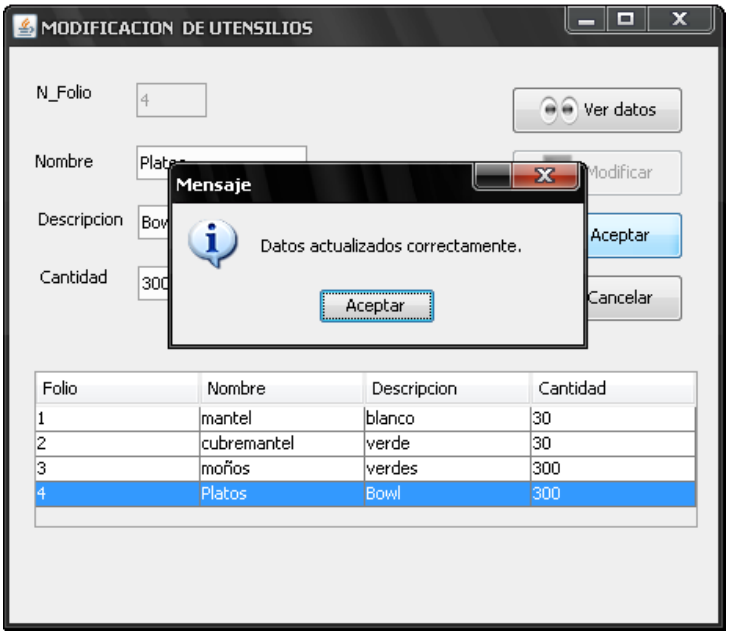


FIGURA VI. 31.
Cuadro de diálogo de Datos actualizados correctamente.

Para eliminar un utensilio el procedimiento es el siguiente, primeramente seleccionamos la opción UTENSILIOS y a continuación se dará un clic en la operación ELIMINAR UTENSILIO.

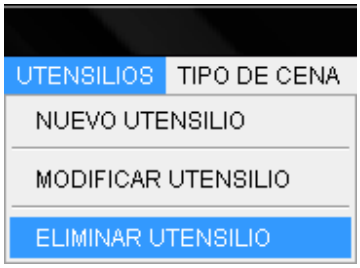


FIGURA VI. 32.
Operación ELIMINAR UTENSILIO.

El sistema mostrará la siguiente ventana:

FIGURA VI. 33.
Ventana BAJA DE UTENSILIOS.

Se dará clic en el botón ver datos y el sistema buscará los utensilios registrados en la base de datos y los desplegará.

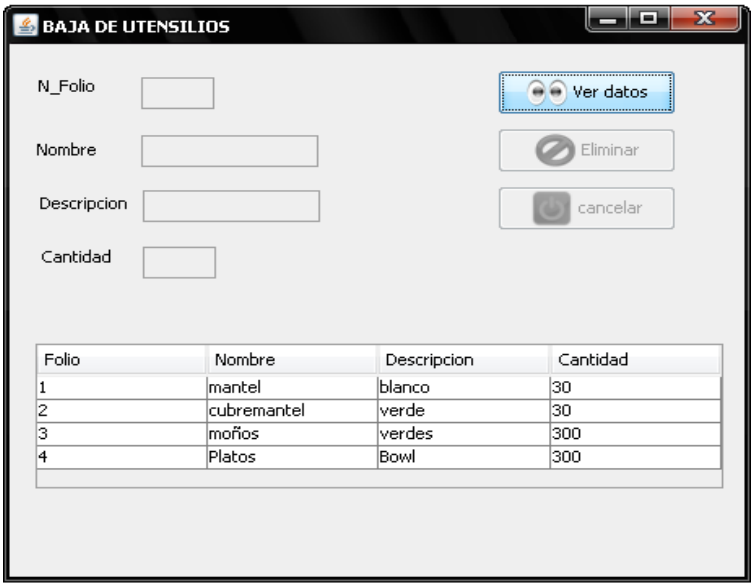


FIGURA VI. 34.
Buscar un utensilio.

A continuación seleccionamos el utensilio a modificar, el sistema nos desplegará los datos correspondientes al utensilio, se dará clic en el botón Eliminar, en seguida el sistema mostrara un cuadro de diálogo notificando que los datos fueron eliminados, el utensilio será eliminado de la base de datos y para finalizar esta ventana se dará clic en el botón cerrar.

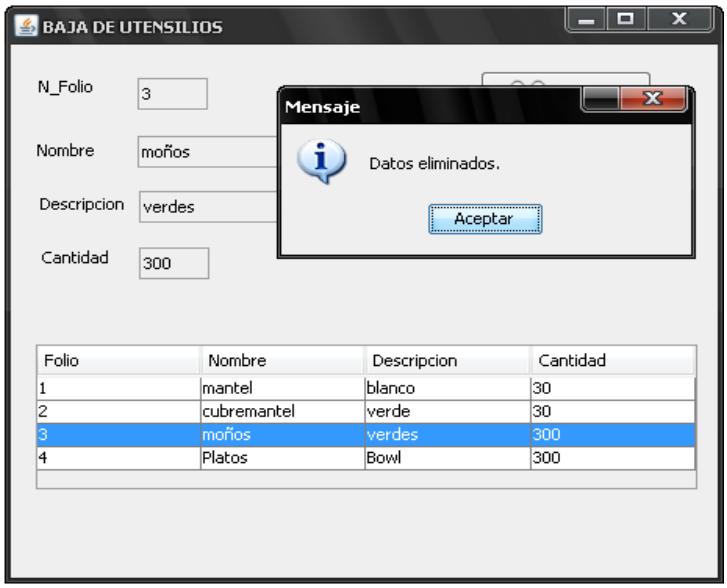


FIGURA VI. 35.
Cuadro de diálogo Datos eliminados.

FIGURA VI. 36.
Utensilio Eliminado de la base de datos.

Para Elegir el tipo de cena del evento a realizar, se elige la opción TIPO DE CENA, a continuación se dará clic en la operación TIPO.

FIGURA VI. 37.
Operación TIPO DE CENA.

El sistema mostrará la siguiente ventana:

FIGURA VI. 38.
Ventana TIPO DE CENA.

Dependiendo de la elección será la cantidad de utensilios entregados a cada mesero. En Tipo de Cena se elegirán los tiempos en que se servirá el servicio de la cena: 1 TIEMPO, 2 TIEMPOS, 3 TIEMPOS, 4 TIEMPOS.

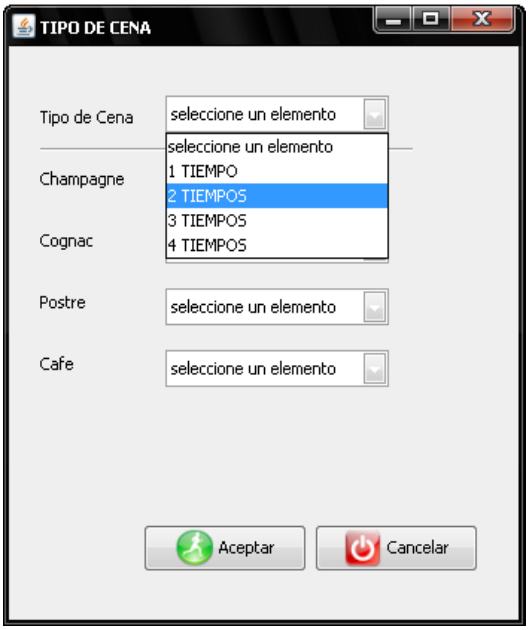


FIGURA VI. 39.
Seleccionar número de tiempos de cena en este caso 2 tiempos.

A continuación se elegirá si el evento requerirá: Copa Champagne; Copa Cognac; plato y cubierto para el Postre y Taza para Café. Al finalizar la selección correspondiente se dará clic en el botón Aceptar, el sistema nos mostrará un cuadro de diálogo que nos indicará la inserción de los datos y para salir de esta ventana se dará clic en el botón cerrar.

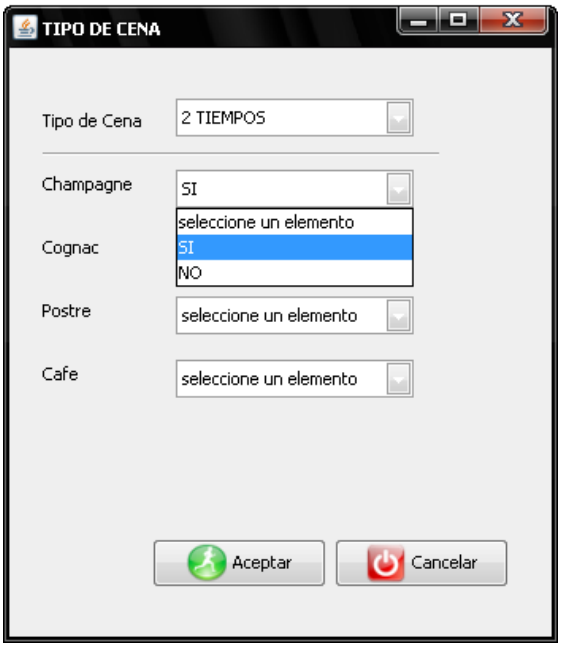


FIGURA VI. 40.
Seleccionando copa Champagne.

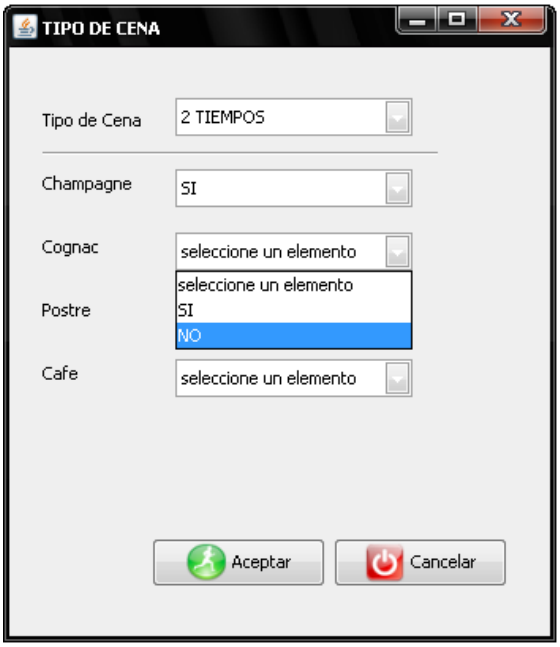


FIGURA VI. 41.
Seleccionando copa Cognac.

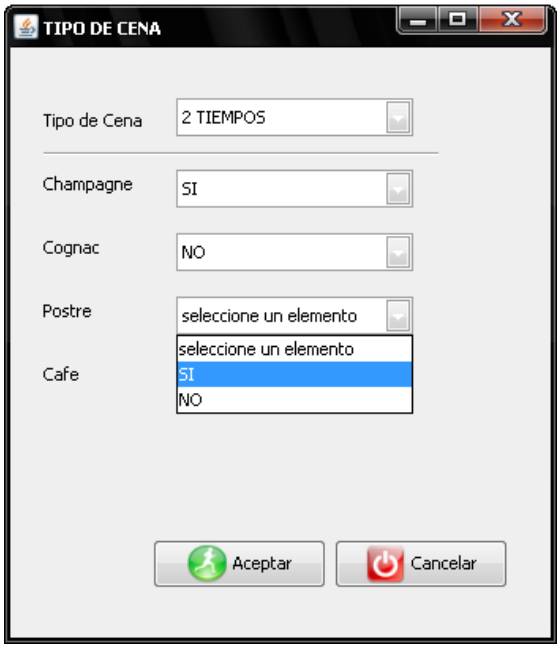


FIGURA VI. 42.
Seleccionando plato y cubierto para Postre.

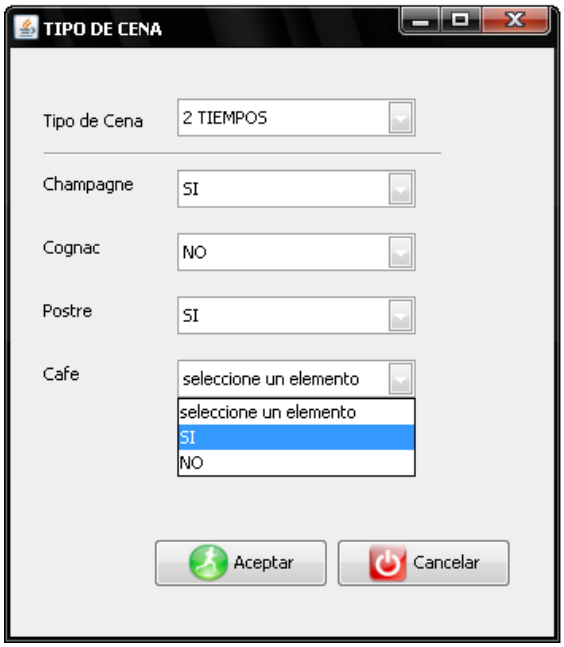


FIGURA VI. 43.
Seleccionando taza para Café.

FIGURA VI. 44.
Cuadro de diálogo de Datos Insertados.

Para realizar la asignación de mesas a cada mesero, se elegirá la opción de ASIGNACION, a continuación se dará clic en la operación MESAS.

FIGURA VI. 45.
Operación ASIGNACION.

El sistema mostrará la siguiente ventana con los datos seleccionados previamente en la opción de Tipo de Cena:

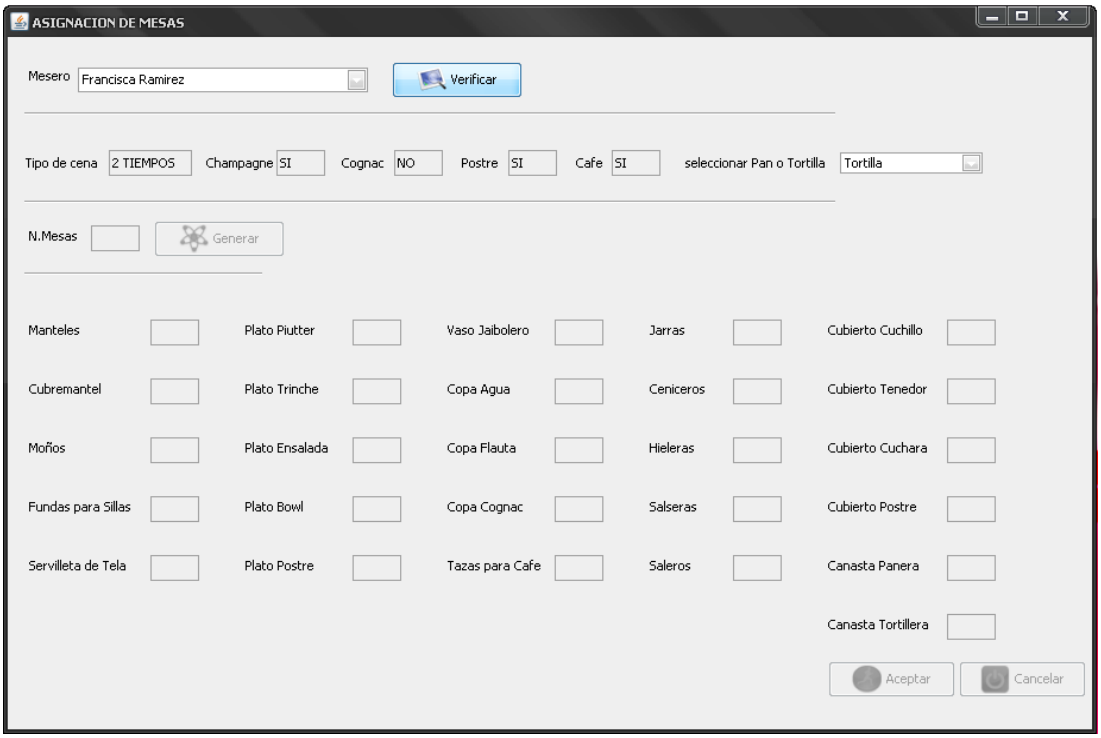


FIGURA VI. 46.
Ventana ASIGNACION DE MESAS.

Para la acción de asignación de mesas primeramente se seleccionará el mesero al cual se le desea asignar mesas, posterior a esta selección se dará clic en el botón Verificar y a continuación el sistema desplegara un mensaje que dirá de la siguiente forma, mesero no tiene asignadas mesas, en caso contrario el sistema mostrará un mensaje que indicará que el mesero ya tiene asignadas mesas y en este caso tendremos que escoger otro mesero.

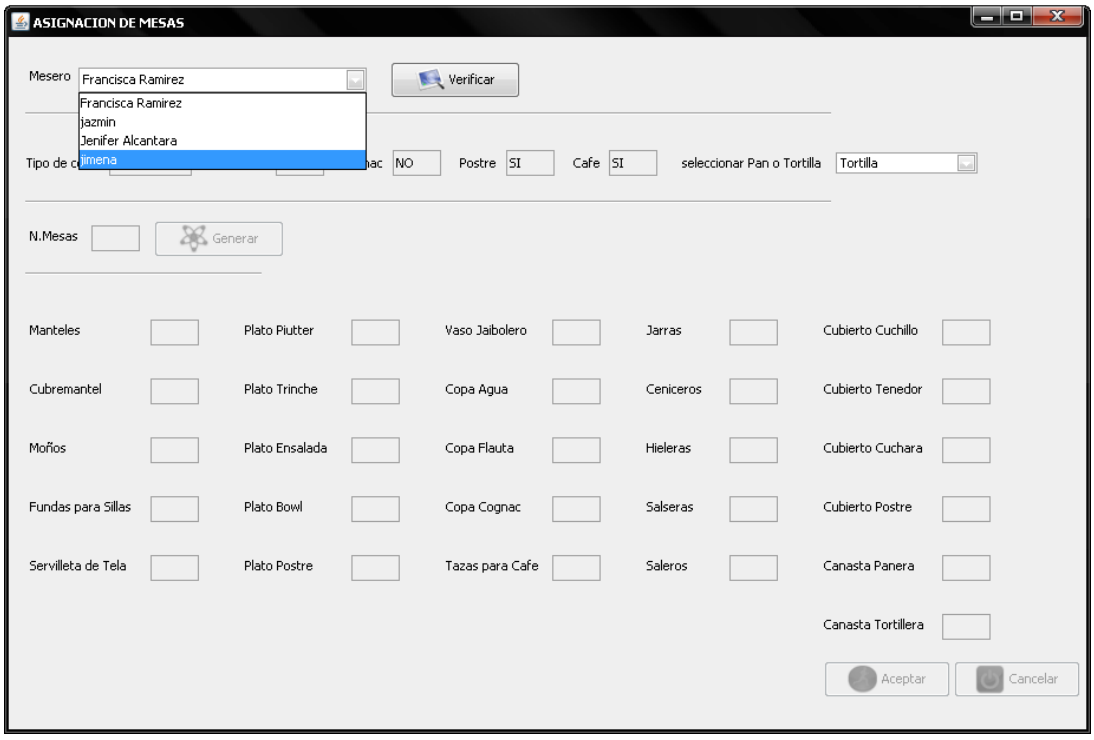


FIGURA VI. 47.
Elección de mesero.

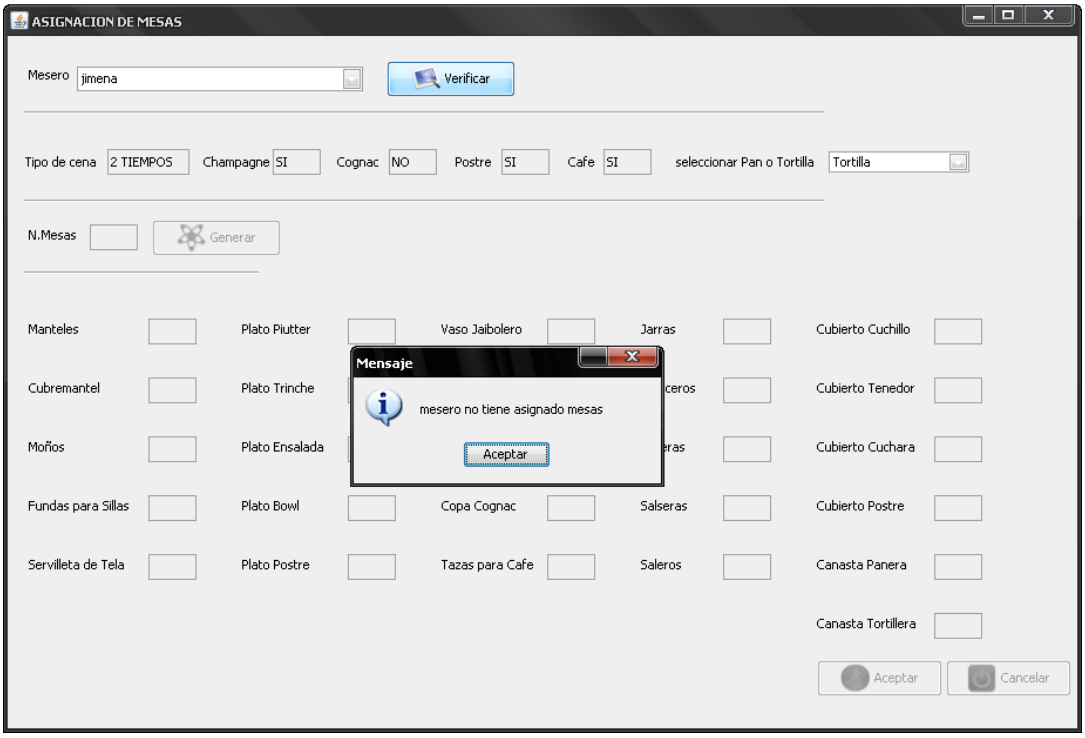


FIGURA VI. 48.
Cuadro de diálogo de mesero no tiene asignada mesas.

En esta ventana también se realizará la elección de canasta para pan, canasta para tortilla o ambas, además el sistema insertará el número de mesas que atenderá el mesero y para finalizar esta serie de elecciones se dará clic en la botón Generar, enseguida el sistema mostrará automáticamente la cantidad de utensilios que se deberán entregar al mesero.

Si los datos son correctos se dará clic en la botón Aceptar, posterior a esta acción el sistema notificará que los datos han sido insertados y se dará clic en el botón Aceptar.

En caso que los datos no sean correctos se dará clic en el botón Cancelar, y el sistema borrará la cantidad de cada utensilio y se volverán a repetir los pasos anteriores a la acción de Asignación de mesas.

Finalmente si todos los datos son correctos se saldrá de la ventana dando clic en el botón cerrar.

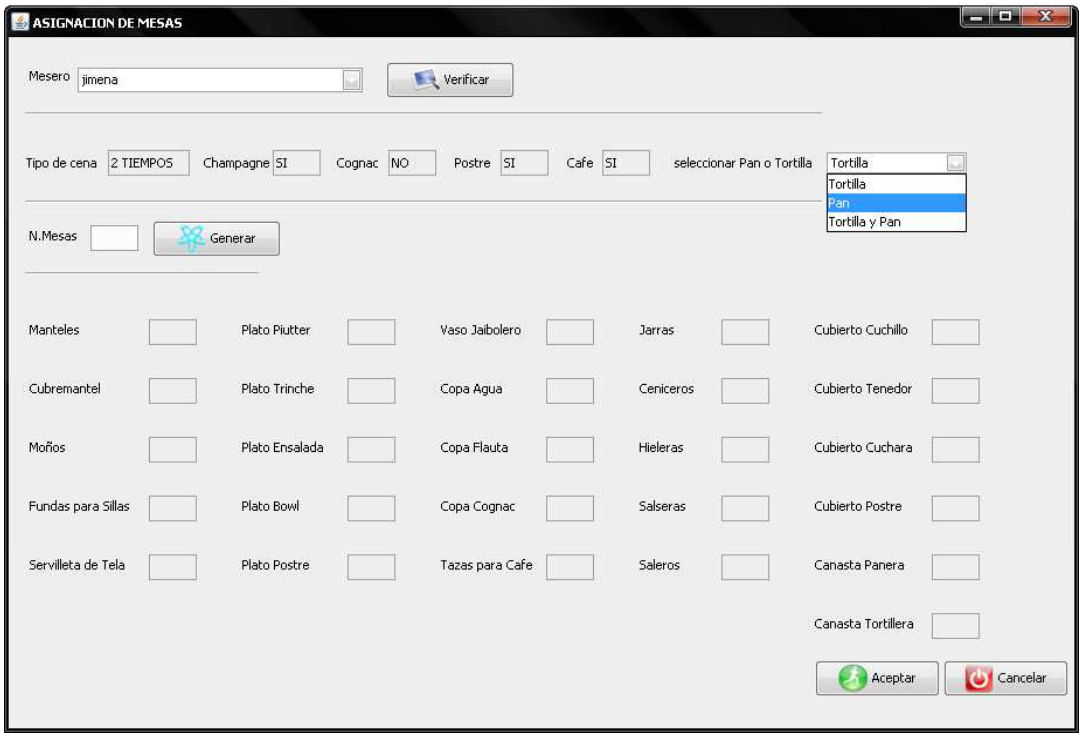


FIGURA VI. 49.
Selección canasta para Pan, canasta para Tortilla o ambas.

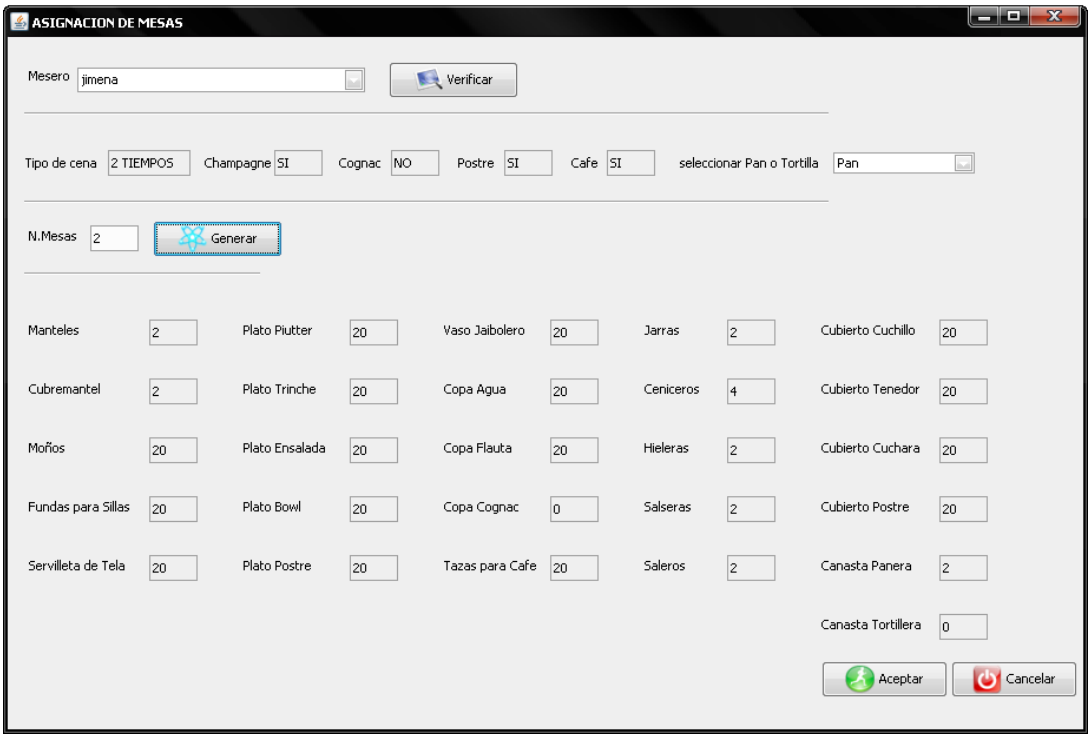


FIGURA VI. 50.
Asignación de mesas al mesero y generación de utensilios.

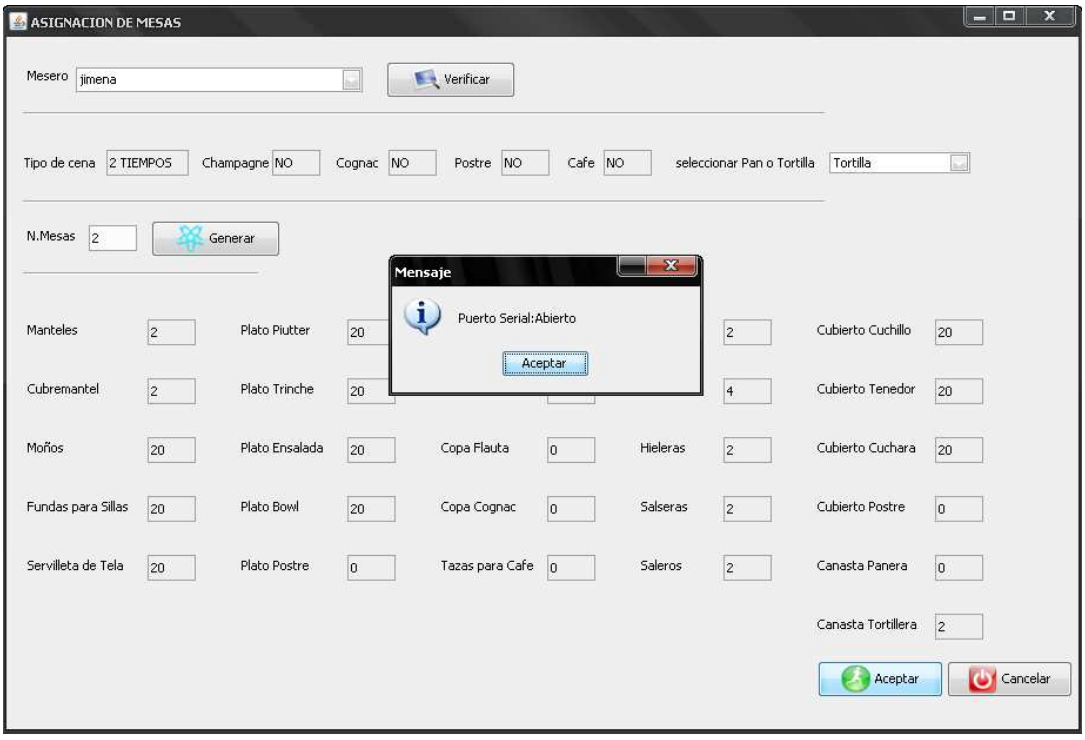


FIGURA VI. 51.
Cuadro de diálogo de Puerto Serial:Abierto.

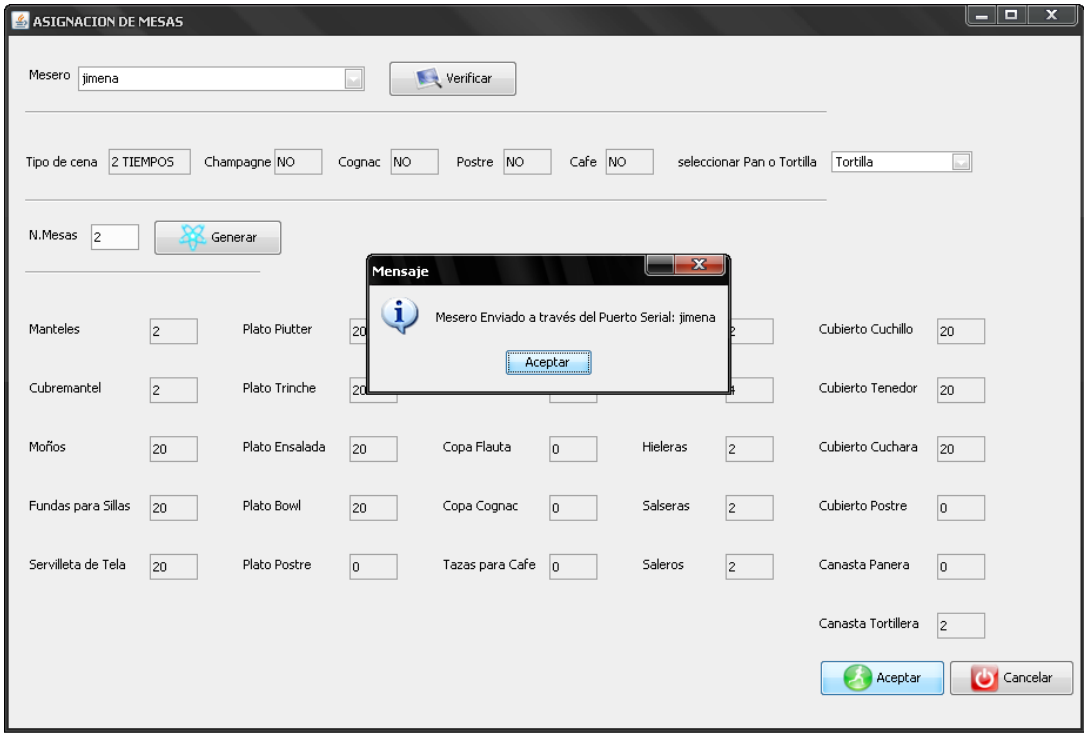


FIGURA VI. 52.
Cuadro de diálogo de Mesero Enviado a través del Puerto Serial: Jimena.

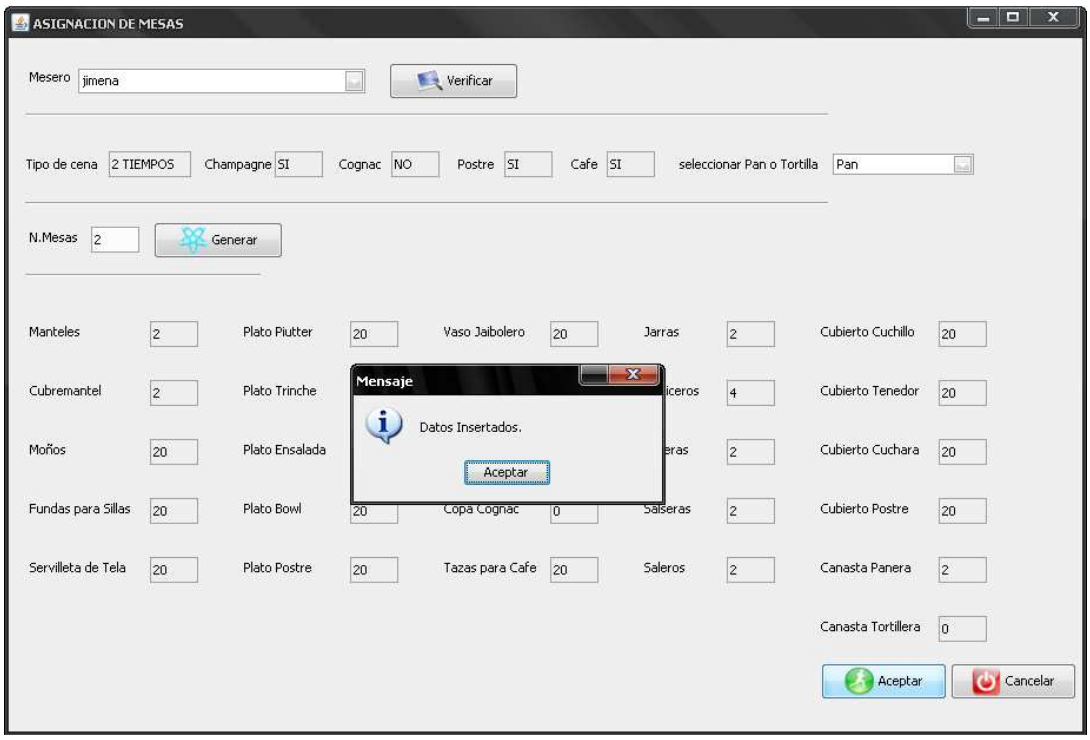


FIGURA VI. 53.
Los datos fueron eliminados.

Para liberar a un mesero de su asignación, se seleccionará la opción LIBERAR y se dará un clic en la operación ADEUDO.

FIGURA VI. 54.
Operación LIBERAR ADEUDO.

El sistema mostrará una ventana con la cantidad de utensilios que se le asigno a cada mesero:

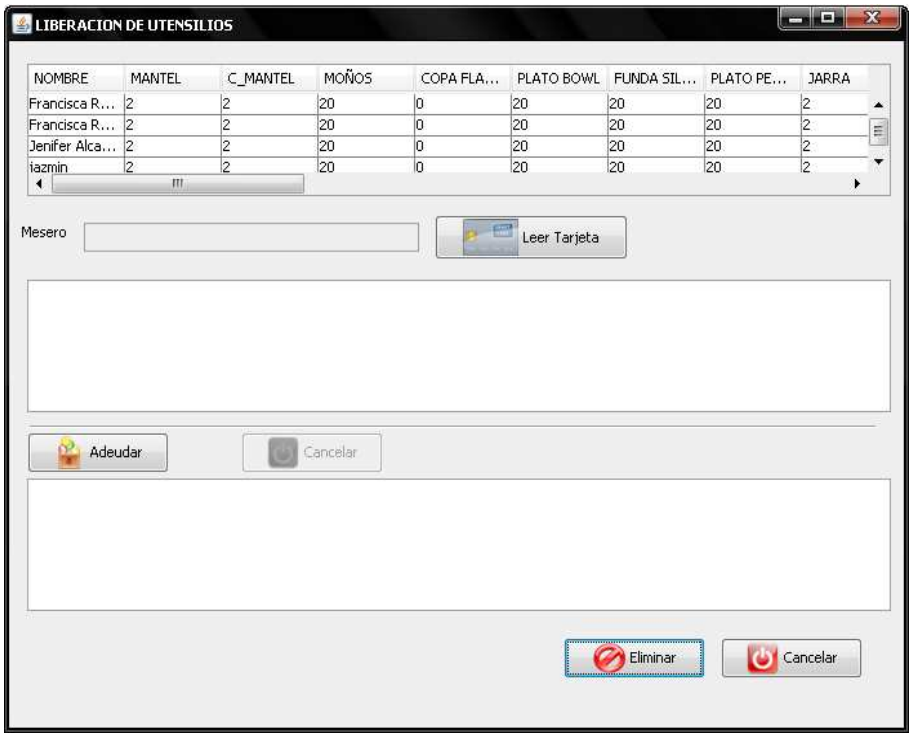


FIGURA VI. 55.
Ventana LIBERACIÓN DE UTENSILIOS asignados a cada mesero.

A continuación se seleccionará el mesero a liberar, en caso de que el mesero adeude algún utensilio se dará clic en el botón Adeudar, en ese momento se podrá insertar la descripción del adeudo en el recuadro inferior de la ventana se dará clic en la botón Aceptar y el sistema mostrará un cuadro de diálogo que nos indicará que los datos han sido eliminados.

En caso que el mesero no adeude ningún utensilio se dará clic en el botón Cancelar localizado en la parte inferior del recuadro de la descripción del adeudo y se procederá a dar clic en el botón Eliminar, posterior a esta acción el sistema notificará que los datos han sido eliminados, finalmente se saldrá de esta ventana dando clic en el botón cerrar.

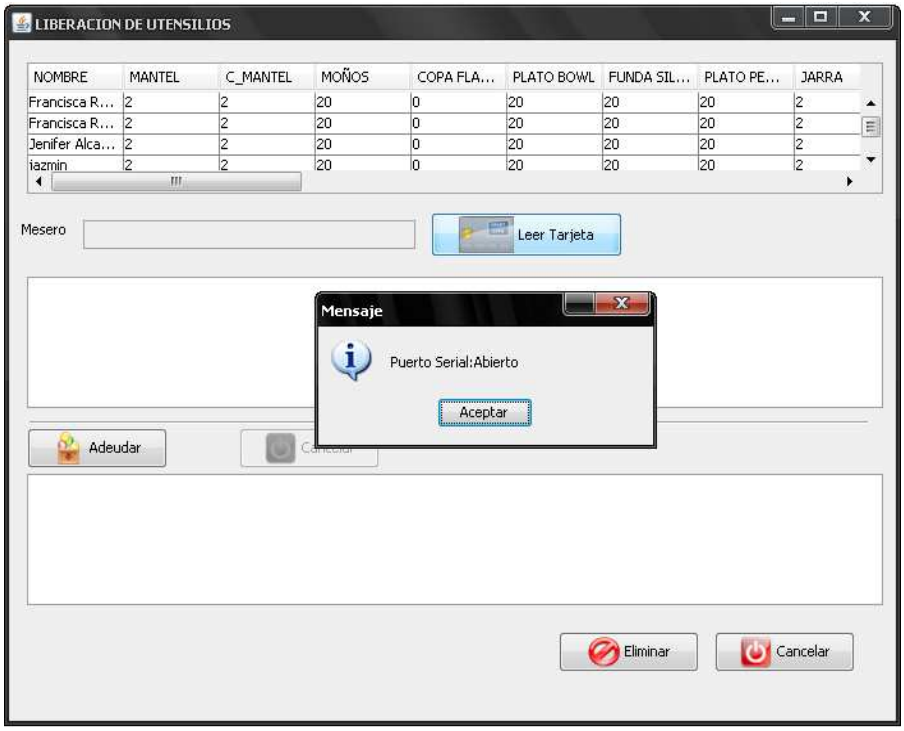


FIGURA VI. 56.
Lectura de la tarjeta Inteligente abriendo el Puerto Serial.

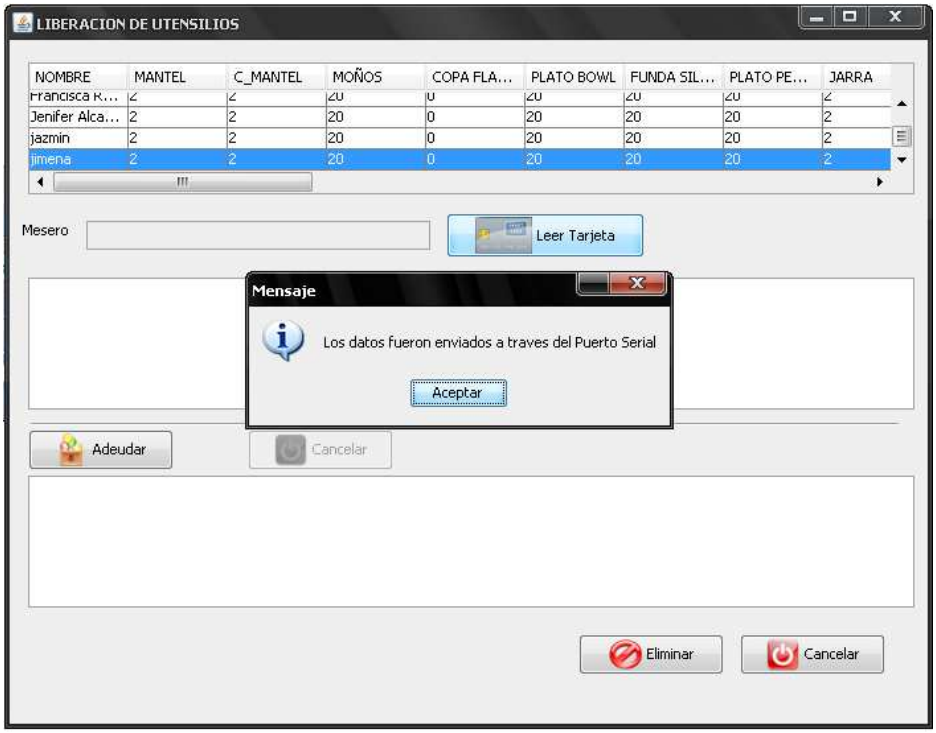


FIGURA VI. 57.
Cuadro de diálogo de confirmación de la lectura de tarjeta.

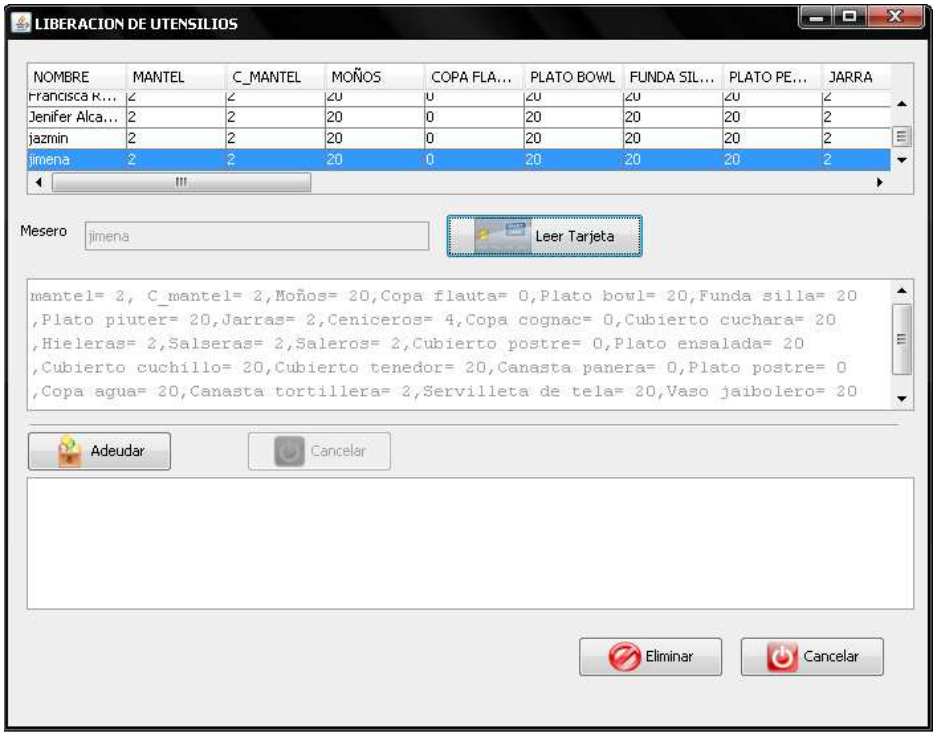


FIGURA VI. 58.
Se despliega el contenido de la tarjeta inteligente.

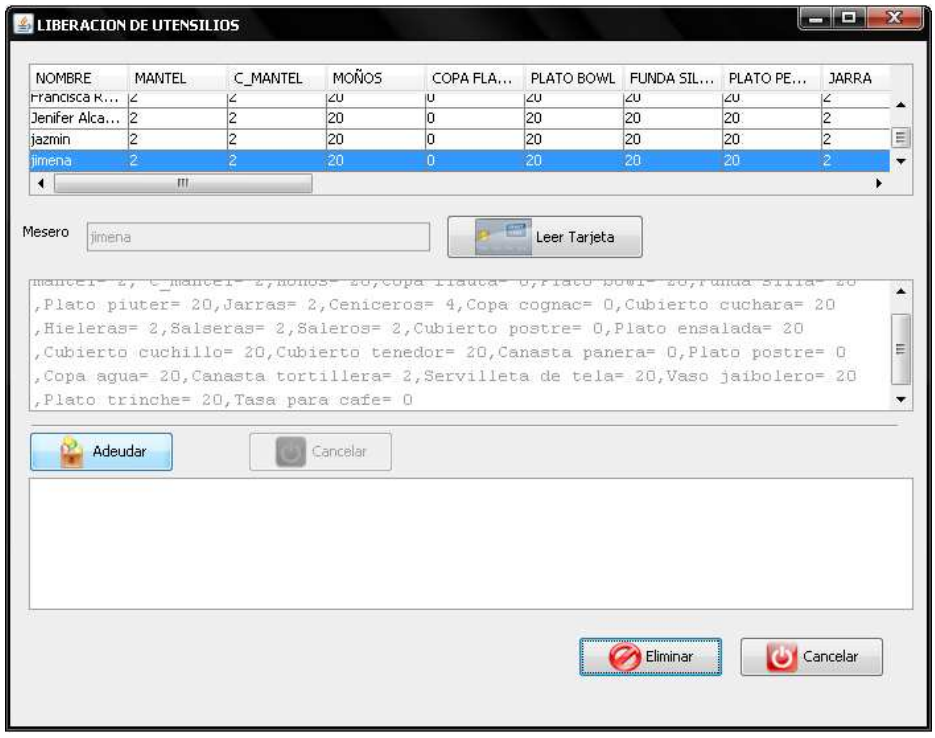


FIGURA VI. 59.
Se da clic en adeudar.

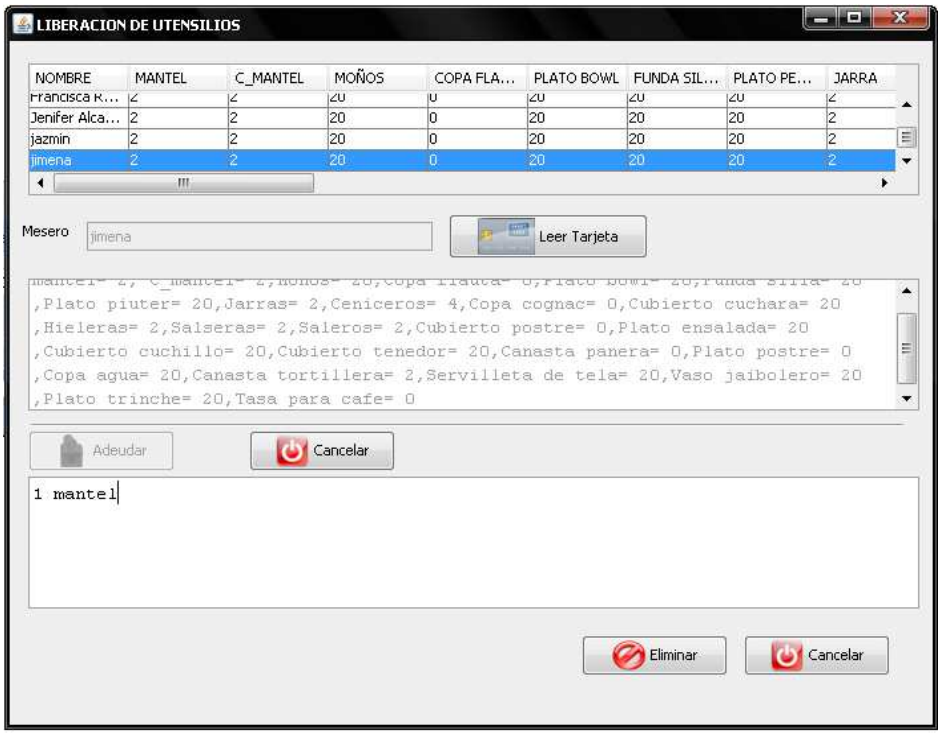


FIGURA VI. 60.
Agregar adeudo a un mesero.

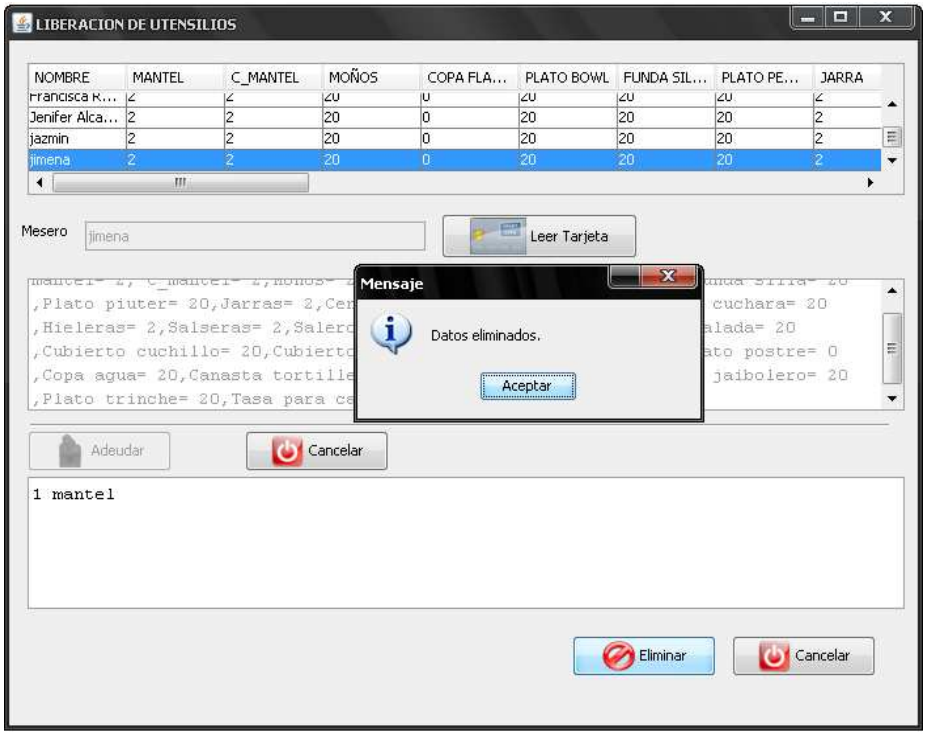


FIGURA VI. 61.
Eliminar asignación de un mesero con adeudo.

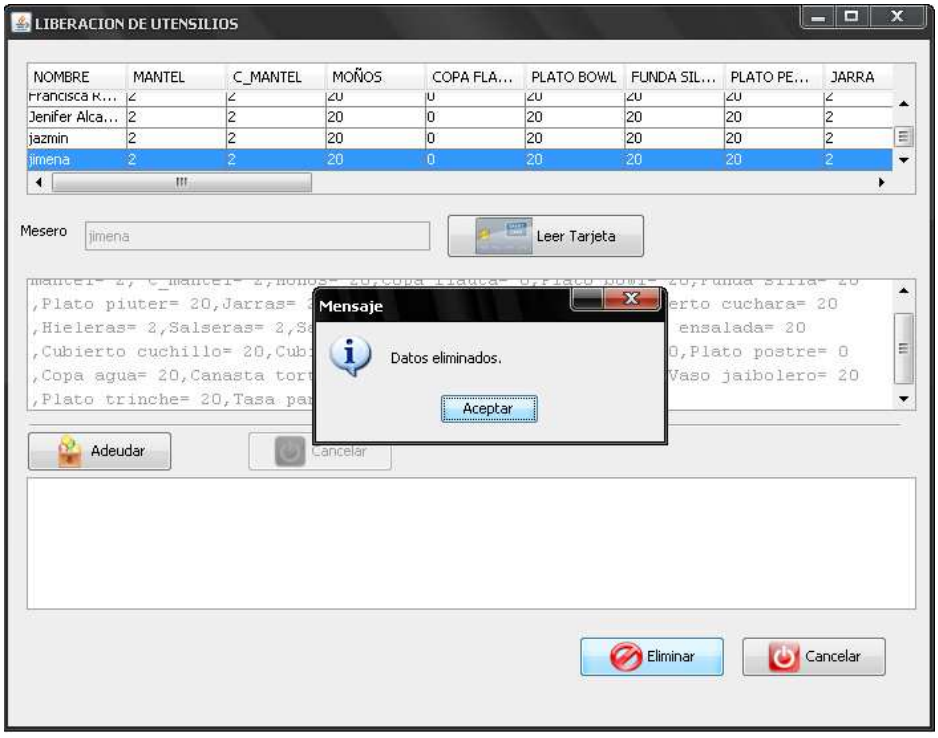


FIGURA VI. 62.
Eliminar asignación de un mesero sin adeudo.

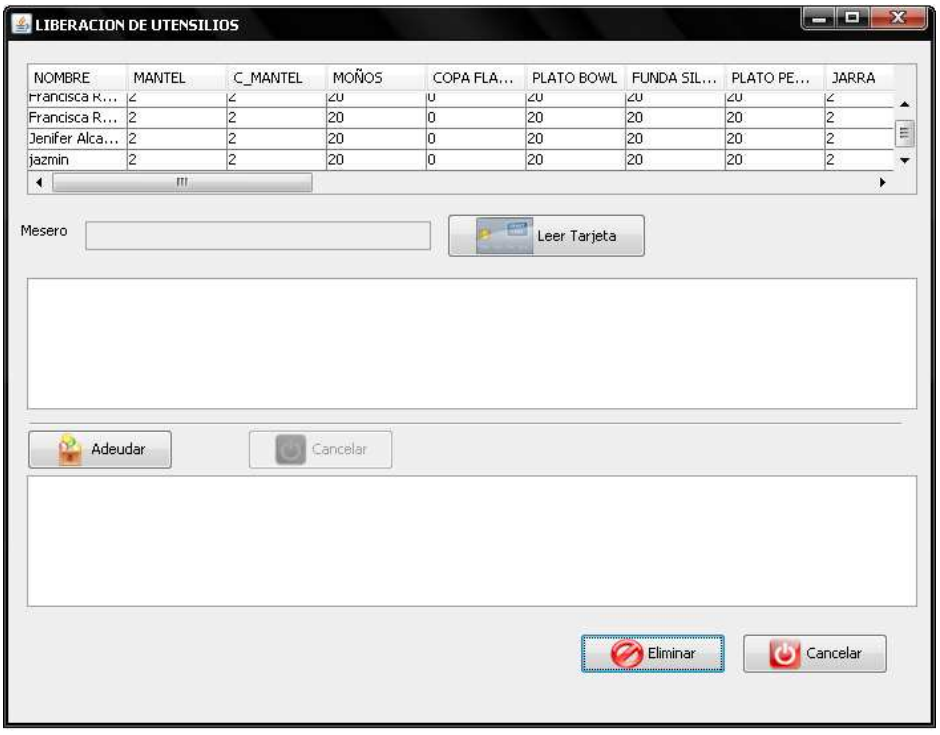


FIGURA VI. 63.
El mesero estará libre para una nueva asignación de mesas.

Para liberar a un mesero de un adeudo anterior, se seleccionará la opción LIBERAR, a continuación se dará un clic en la operación DEBE.

FIGURA VI. 64.
Operación LIBERAR DEBE.

El sistema mostrará la siguiente ventana con la cantidad de utensilios que adeuda cada mesero:

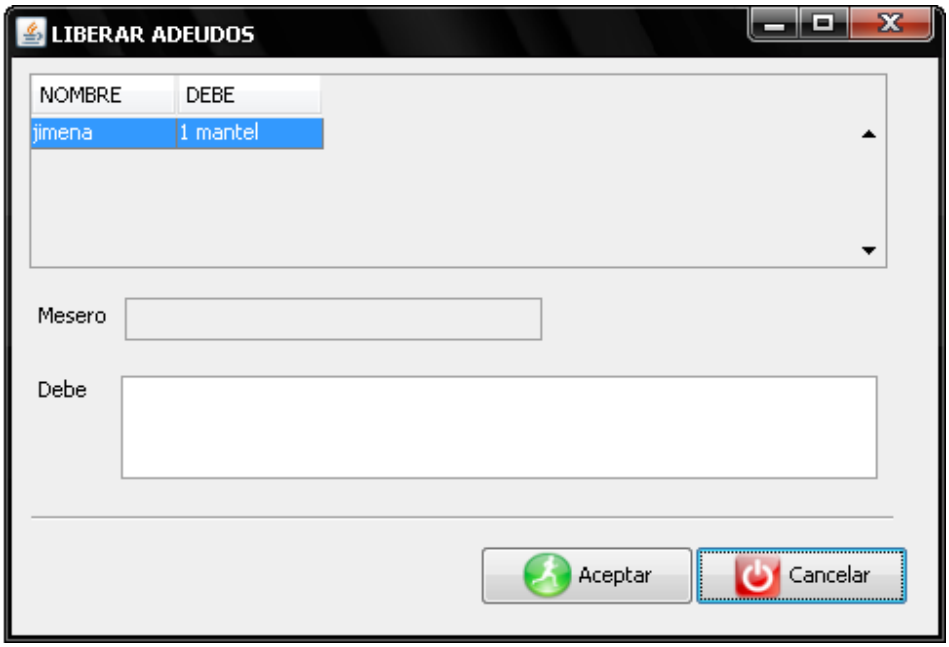


FIGURA VI. 65.
Ventana LIBERAR ADEUDOS.

A continuación se seleccionará el mesero a liberar del adeudo, una vez seleccionado el mesero el sistema mostrará los datos de los utensilios que adeuda, para eliminar dicho adeudo se dará clic en el botón Aceptar, el sistema notificará a través de un cuadro de diálogo que los datos han sido eliminados, finalmente se dará clic en el botón Aceptar y el adeudo será eliminado. Para salir se dará clic en el botón cerrar.

FIGURA VI. 66.
Selección de mesero con adeudo.

FIGURA VI. 67.
Cuadro de diálogo avisando que fue abierto el Puerto Serial.

FIGURA VI. 68.
Los datos fueron eliminados de la tarjeta inteligente.

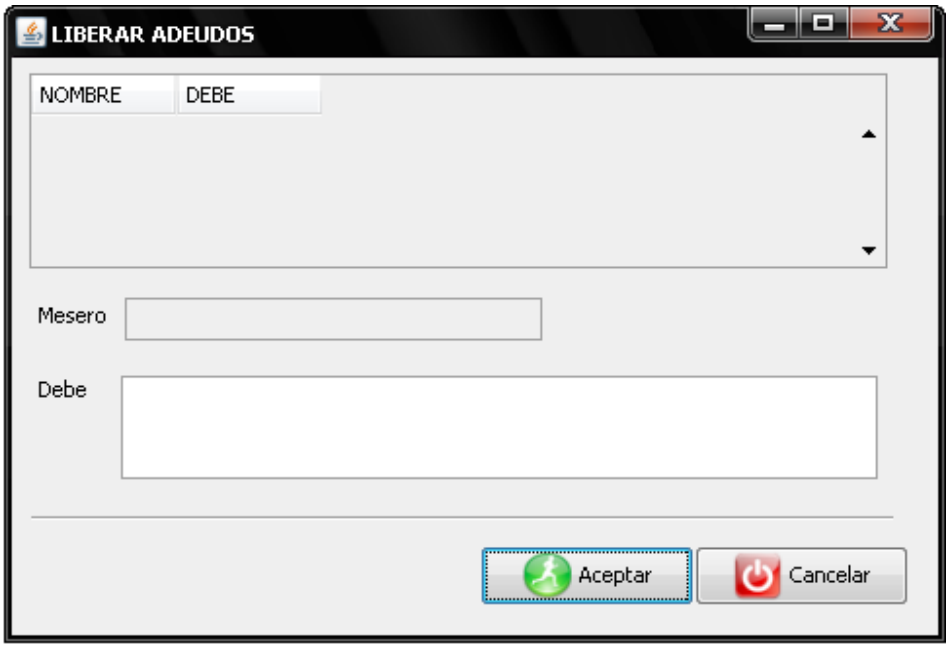


FIGURA VI. 69.
Adeudo de mesero eliminado.

Para agregar una reservación, se seleccionará la opción RESERVACION, a continuación se dará un clic en la operación NUEVA RESERVACION.

FIGURA VI. 70.
Operación NUEVA RESERVACION.

El sistema mostrará la siguiente ventana:

FIGURA VI. 71.
Ventana RESERVACION.

A continuación se llenarán los campos con los datos correspondientes de la reservación que se desea dar de alta, y se dará clic en el botón Aceptar, posterior a esta acción el sistema mostrará un cuadro de diálogo notificando que los Datos fueron Insertados correctamente y se volverá a dar clic en el botón Aceptar, para salir de la ventana se dará clic en el botón cerrar.

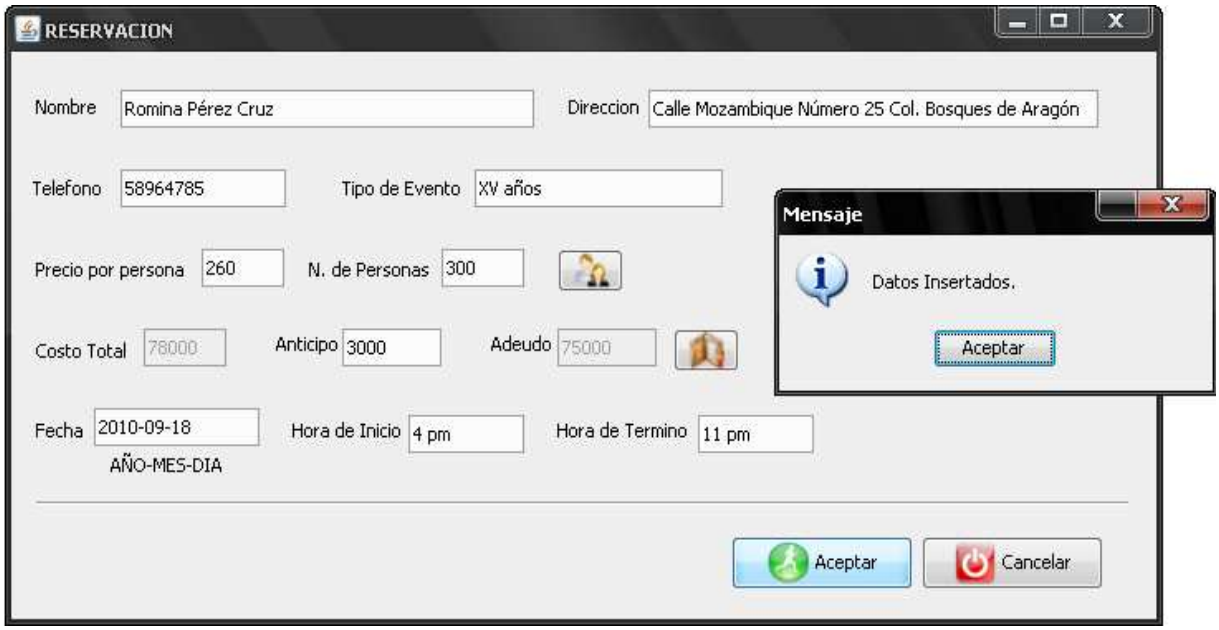


FIGURA VI. 72.
Insertar Datos de una reservación.

Para modificar una reservación, se seleccionará la opción UTENSILIOS, a continuación se dará clic en la operación MODIFICAR RESERVACION.

FIGURA VI. 73.
Operación MODIFICAR RESERVACION.

El sistema mostrará la siguiente ventana:

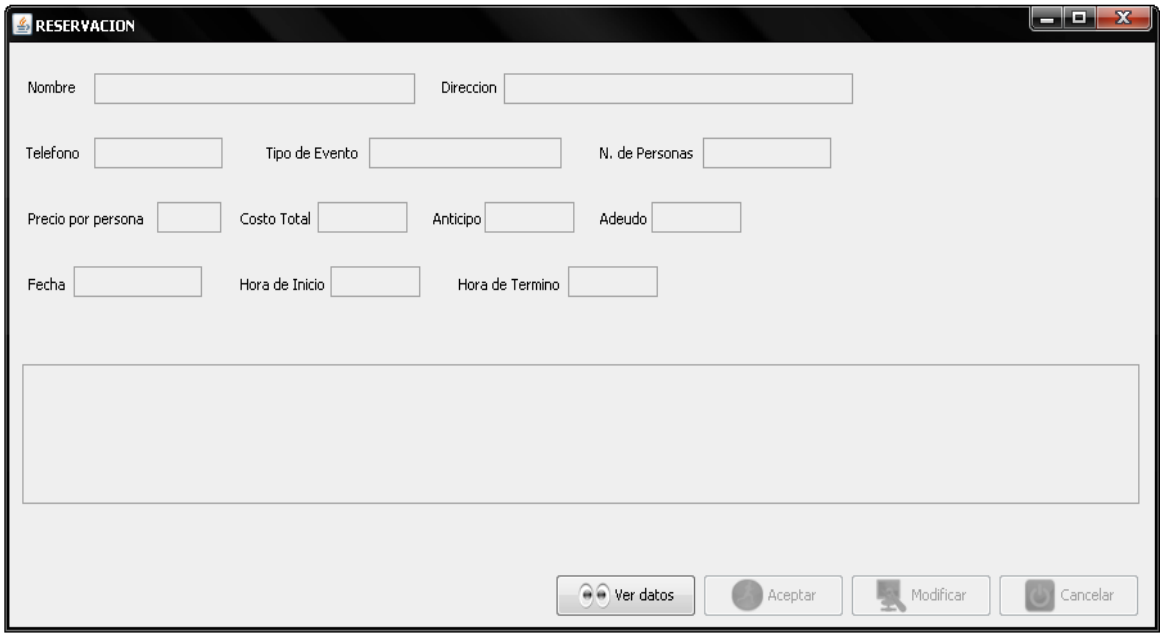


FIGURA VI. 74.
Ventana Modificación de una reservación.

A continuación se dará clic en el botón ver datos, posterior a esta acción el sistema buscará las reservaciones dadas de alta y las desplegará.

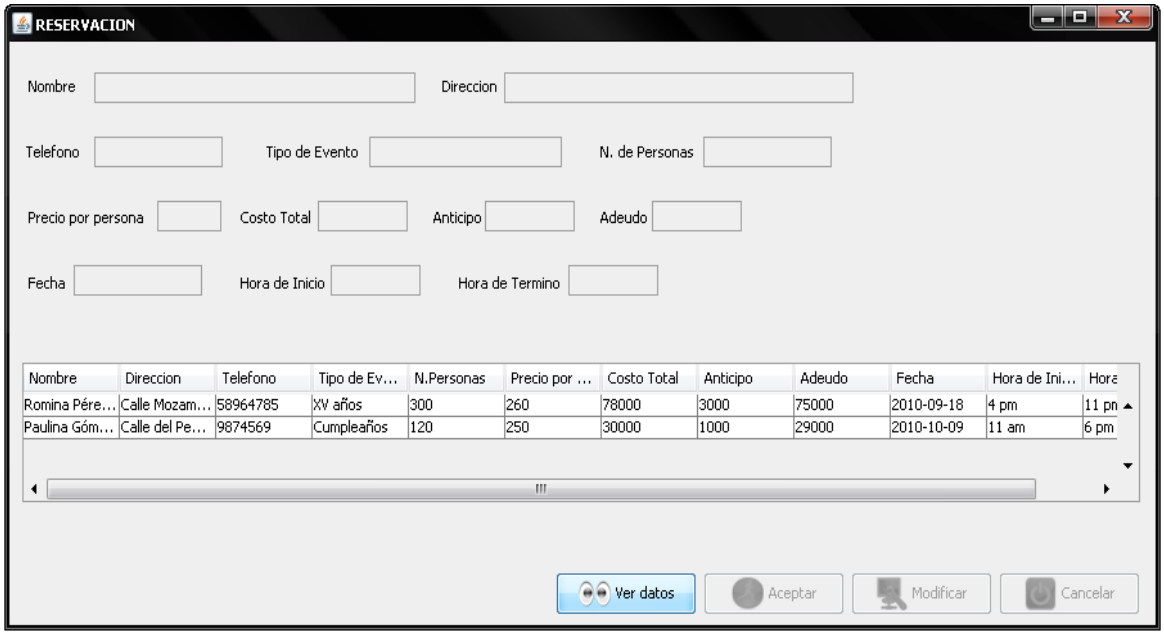


FIGURA VI. 75.
Buscar Datos de una reservación.

A continuación se seleccionará la reservación a modificar, aparecerá un cuadro de diálogo que nos indicará que el Encontrado y nos desplegará los datos atenuados correspondientes al utensilio, para poder modificarlos, se da clic en Modificar, se harán las correcciones correspondientes, se da clic en Aceptar, desplegara un mensaje Datos actualizados correctamente, salimos de la ventana dando clic en el botón X.

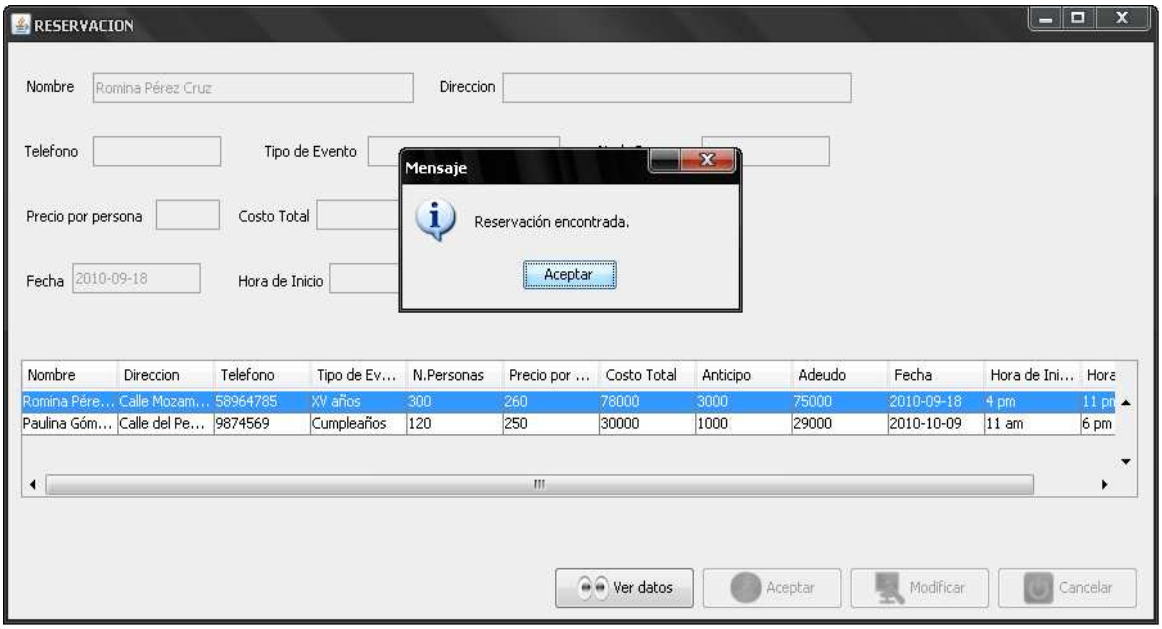


FIGURA VI. 76.
Encontrar una reservación ya existente.

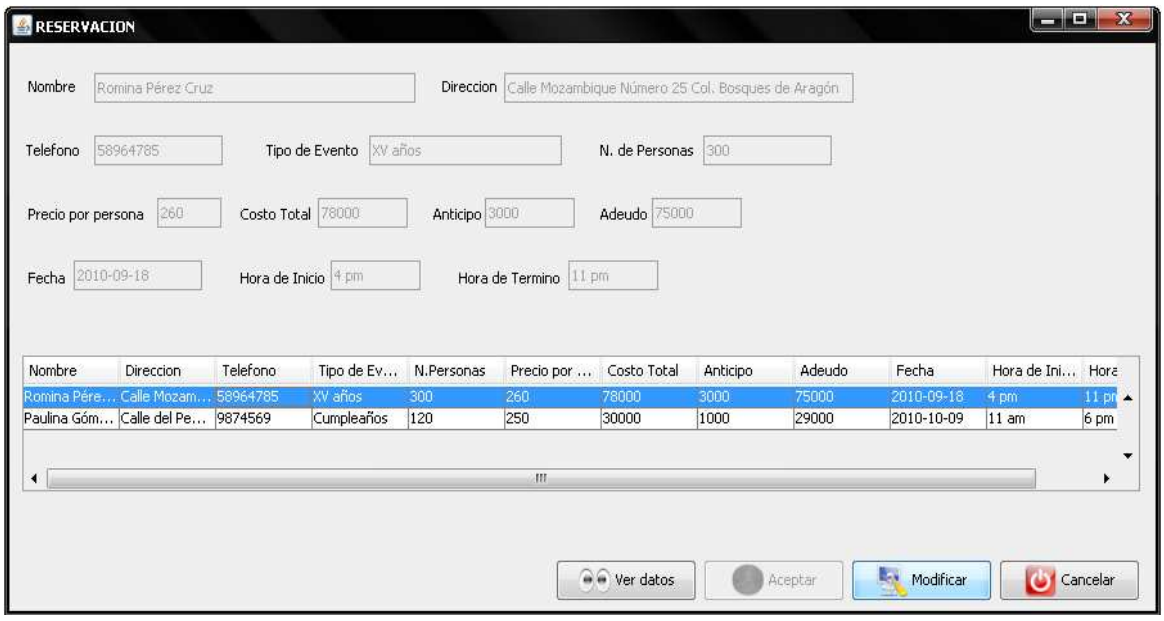


FIGURA VI. 77.
Des atenuar los datos para poder modificarlos.

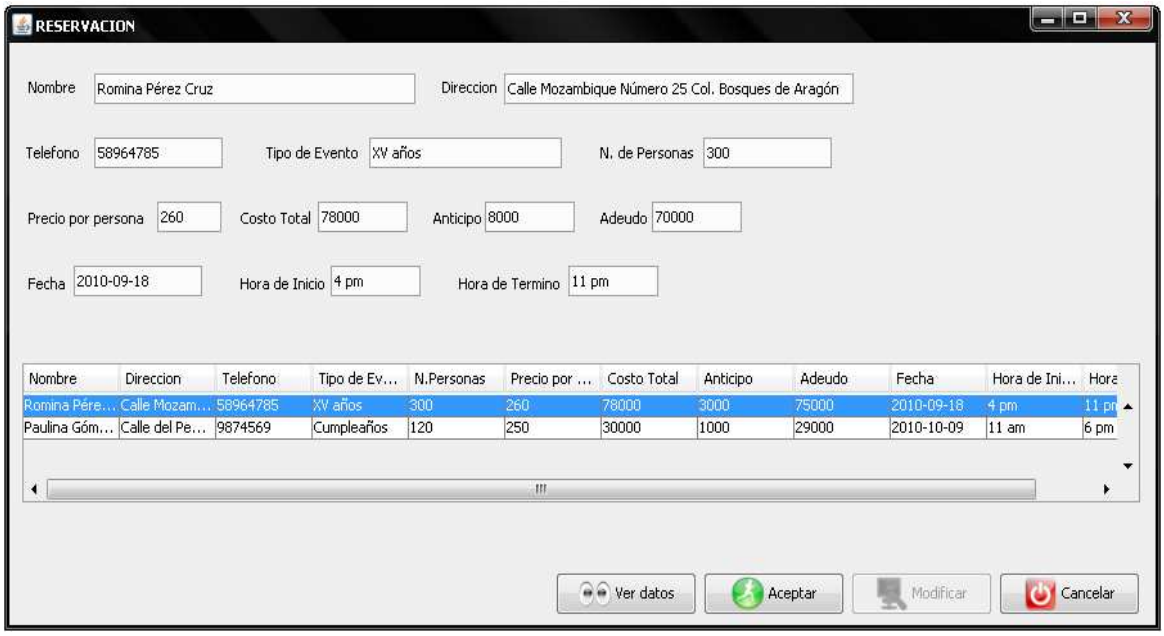


FIGURA VI. 78.
Modificando la cantidad del Anticipo y Adeudo.

FIGURA VI. 79.
Mensaje Datos actualizados correctamente.

Para eliminar una reservación, seleccionamos RESERVACIÓN, a continuación se da un clic en ELIMINAR RESERVACION.

FIGURA VI. 80.
Opción ELIMINAR RESERVACION.

El sistema muestra la siguiente pantalla:

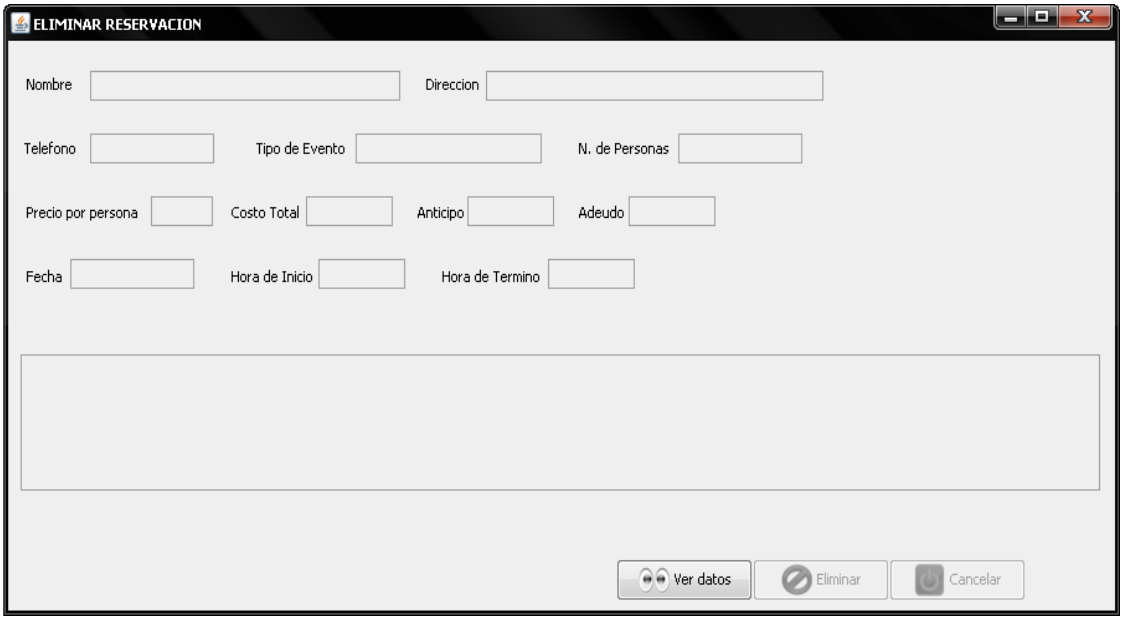


FIGURA VI. 81.
Baja de reservación.

Se selecciona ver datos, el sistema busca las reservaciones dadas de alta y las despliega.

FIGURA VI. 82.
Buscar una reservación.

Se selecciona la reservación a modificar, aparecerá un mensaje Folio encontrado y nos desplegará los datos correspondientes de la reservación, se da clic en Eliminar, mostrara un Mensaje Datos Eliminados, la reservación será eliminado de la base de datos, salimos de la ventana dando clic en el botón X.

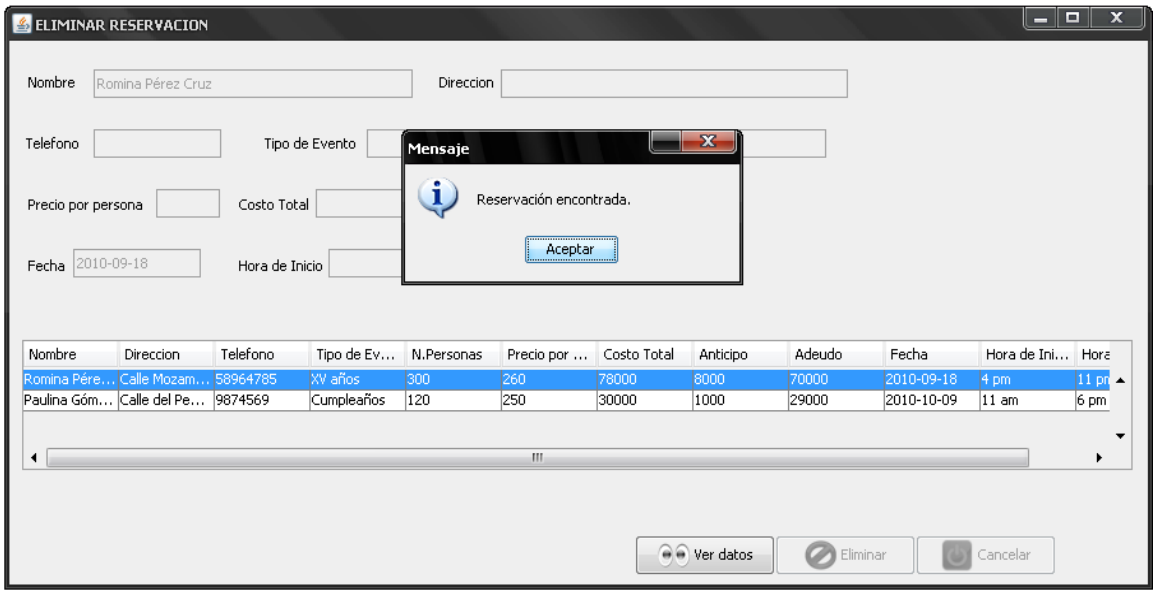


FIGURA VI. 83.
Mensaje Folio encontrado.

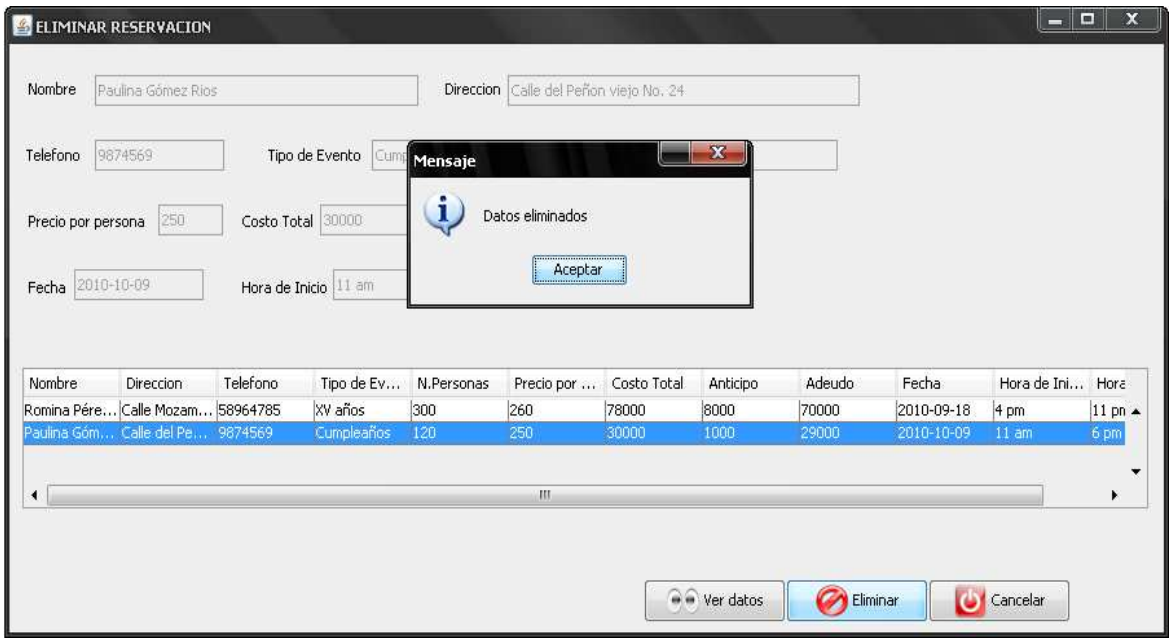


FIGURA VI. 84.
Mensaje Datos eliminados.

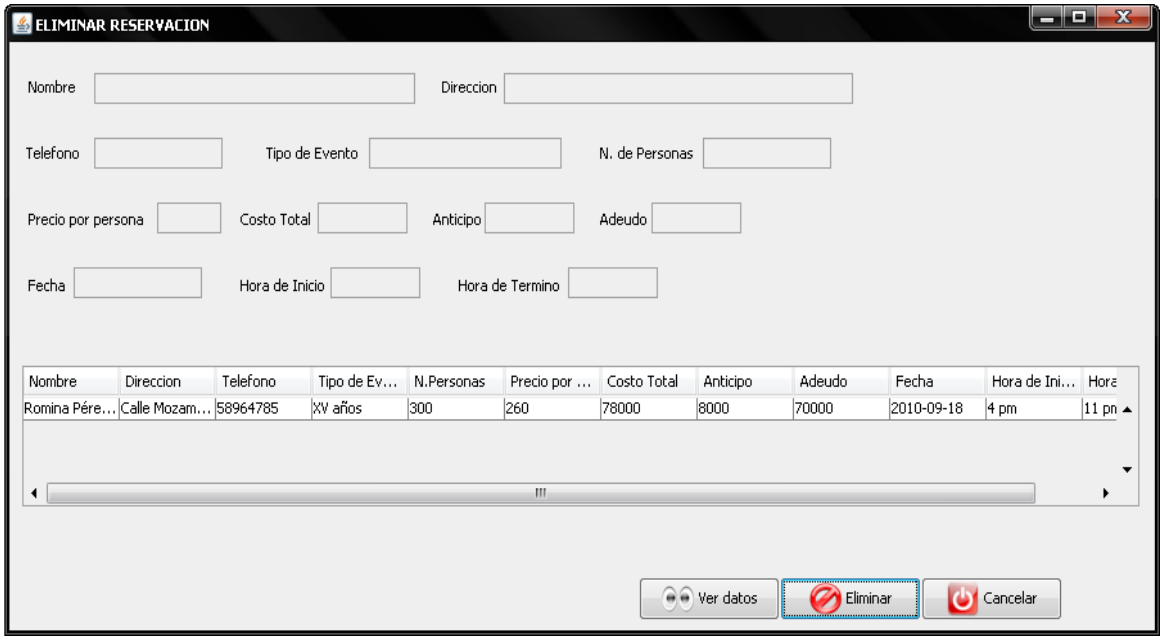


FIGURA VI. 85.
Datos eliminados de la base de datos.

CONCLUSIONES

Las grandes virtudes y aplicaciones que el lenguaje de programación Java ofrece así como su característica de universalidad, es decir la capacidad de que un programa desarrollado en Java puede ejecutarse en cualquier computadora independientemente de la plataforma, han hecho de este lenguaje uno de los más importantes en el desarrollo de software.

El desarrollo de aplicaciones utilizando la tecnología Java es realmente amplio ya que abarca desde aplicaciones para mainframe hasta aplicaciones para tarjetas inteligentes. Pero lo más importante es que permite desarrollar aplicaciones, no en el sentido particular, sino que estas aplicaciones pueden extenderse por toda la red de internet, ya que Java es el primer lenguaje de programación diseñado para el trabajo en red y para el desarrollo de aplicaciones web. Java es sin lugar a dudas la herramienta clave para vincular muchos tipos de dispositivos, ejecutando distintos sistemas operativos y varias clases de aplicaciones.

Otro aspecto importante del lenguaje Java es que permite establecer conexiones con diversas bases de datos de una manera directa, esto hace que el rendimiento al accesar a la base de datos sea óptimo, al ser un driver 100% Java. Además de permitir la manipulación de las mismas enviando sentencias SQL.

El lenguaje de programación Java permite ejecutar de forma segura aplicaciones en tarjetas inteligentes. Para el desarrollo de dichas aplicaciones Java cuenta con herramientas de programación para el acceso a la información de las tarjetas inteligentes, proporciona interfaces bien definidas que permiten la separación de las funciones del lector, la tarjeta y el sistema operativo, también contribuyen a facilitar el trabajo de los desarrolladores, promueve la interoperabilidad entre los distintos componentes de un sistema basado en tarjetas inteligentes.

En México el uso de las tarjetas inteligentes es cada vez mayor. Se siguen implementando diversos tipos de sistemas, tanto por parte de empresas. Por tal motivo, se debe destacar la importancia de desarrollar esta tecnología dentro del país.

BIBLIOGRAFÍA

- Carretero Pérez, Jesús. Sistemas Operativos una visión aplicada, Mc Graw Hill, 2001.
- Tanenbaum, Andrew S. Sistemas Operativos, Diseño E Implementación; Prentice Hall 2007.
- Stallings, William. Sistemas Operativos; Prentice Hall, 2da edición, 2006.
- Louden, Kenneth C. Lenguajes de programación, Principios y práctica; Thomson, 2da edición, 2004.
- Piattini, Mario. Diseño de Bases de Datos Relacionales; Alfaomega, 2007.
- White, Fisher, Cattell, Hamilton, Hapner. JDBC API Tutorial and Reference, Second Edition: Universal Data Access for the Java 2 Platform. Addison Wesley, 2006.
- Esteban, Ángel. Acceso a bases de datos con Java-JDBC 2.0, Grupo EIDOS, 2000.
- Holzner, Steven. La biblia de JAVA, Coriolis, 2004.
- Arnow, D., Weiss, G. Introducción a la programación con Java, Un enfoque orientado a objetos; Prentice Hall, 2006.
- Java Card™ Platform, Version 2.2.2; Sun Microsystems, Inc., U.S.A., 2006.
- Java 2 Platform, Standard Edition, v. 1.3. API Specification; Sun Microsystems, Inc., U.S.A., 2007.
- Anasagasti, Pedro de Miguel. Fundamentos de las computadoras, Editorial Paraninfo, 2000.
- Sandoval, Juan D. Mitos y realidades sobre las tarjetas inteligentes smart cards, Thomson Publishing Company, 2003.
- Urien, P., Badra, M., Dandjinou, M. EAP-TLS Smartcards, from Dream to Reality, U.S.A., 2006.
- Guthery, Scott. Smart Card Developer's Kit, Macmillan, Computer Publishing, 2002.

MESOGRAFÍA

- http://www.lenguajes-de-programacion.com/programacion.shtml
- http://www.infosistemas.com.mx/soto10.htm
- http://www.monografias.com/trabajos/tendprog/tendprog.html
- http://www.monografias.com/trabajos/lengprog/lengprog.html
- http://entren.dgsca.unam.mx/introduccion/lenguajes.html
- http://atenea.udistrital.edu.co/profesores/jdimate/basesdatos1/tema1_4.htm
- http://itlp.edu.mx/publica/tutoriales/basedat2/hseis6_1.htm
- http://alarcos.inf-cr.uclm.es/doc/bda/doc/trab/T0001_MAMoraga.pdf
- http://www3.uji.es/~mmarques/f47/apun/node38.html
- http://iio.ens.uabc.mx/~jmilanez/escolar/bases.de.datos/02020000.html
- http://inf.unitru.edu.pe/~edsh/documentos/bd_jerarquico.pdf
- http://dev.mysql.com/doc/refman/5.0/es/features.html
- http://dev.mysql.com/doc/refman/5.0/es/ansi-diff-views.html
- http://dev.mysql.com/doc/refman/5.0/es/storage-engines.html#
- http://dev.mysql.com/doc/refman/5.0/es/innodb.html
- http://dev.mysql.com/doc/refman/5.0/es/ndbcluster.html
- http://www.dma.eui.upm.es/historia_informatica/Doc/Maquinas/UNIVAC.htm
- http://sunsite.unam.mx/java.html
- http://www.euitt.upm.es/java/cursojava/1_Intro/1.3_OOP/oop.htm
- http://www.mailxmail.com/curso-arquitectura-computadoras
- http://www.tvirtual.com.mx/Tarjetas.html