1. ANTECEDENTES GENERALES

1.1 INTRODUCCION

En la actualidad resulta difícil pensar que no exista un paquete de software preprogramado para responder a funciones administrativas específicas. Cada vez nos cubren más con nuevos lenguajes y entornos de desarrollo que tienen esto y lo otro, nos dicen e informan que las nuevas tendencias van hacia aquí o allá. Por esta razón, los proyectos de sistemas inician con la selección de software o herramientas de desarrollo, en donde debemos evaluar aspectos más importantes que se deben considerar para identificar las necesidades de compra, los factores que pueden ser críticos en el proceso de adquisición, con la finalidad de elegir la herramienta más adecuada para cada caso.

La oferta actual de herramientas de desarrollo, es bastante amplia, aunque no es menos cierto que son cuatro o cinco entornos los más usados, teniendo los demás un uso bastante selectivo. Qué opción elegir es una tarea a veces compleja, sobre todo si no se conocen las características de cada una de las ofertas existentes. Lógicamente el factor de mayor peso deben ser las necesidades de desarrollo que se tengan, buscando una herramienta que pueda satisfacerlas, pero es habitual tomar en consideración otros aspectos no menos importantes, como la facilidad de aprendizaje de la nueva herramienta, la potencia y facilidad del lenguaje que usa, los componentes disponibles si se trata de una entorno RAD, etc.

Al desarrollar una metodología de evaluación nos permitirá encontrar una solución más acorde a nuestras necesidades y posibilidades, además de conocer otros productos de software, ampliando de esta manera nuestros conocimientos y criterios sobre las diferentes herramientas de desarrollo.

1.2 ESTUDIO DE METODOLOGIAS

1.2.1 MUR 97 (METODOLOGÍA DE EVALUACIÓN DE HERRAMIENTAS CASE)

Esta metodología intenta dar solución a los problemas inherentes en los proyectos de generación de aplicaciones informáticas. Está basado en el ciclo de vida de los sistemas, recoge información acerca de las tareas que se van ha realizar, hace referencia a las técnicas de empezar bien desde el principio y mejorar la calidad, para producir empresas competitivas. Utiliza las técnicas numéricas aplicadas con mayor frecuencia como el método de análisis de decisión multidiscreta, que se basa en la comparación entre las características del hardware o software ofertados y las especificaciones técnicas y requisitos funcionales.

A continuación se describe la técnica de valorización de uso bastante extendido, basada en los métodos de análisis de decisión multidiscreta.

Terminología

Se utilizará la siguiente terminología:

Alternativas a valorar: A_1, A_2, ..., A_i, ..., A_m (Cada una corresponderá a una oferta diferente).

Criterios de valoración : C_1, C_2, ..., C_j, ..., C_n (Cada uno corresponderá a un factor o característica a valorar)

Valoraciones parciales relativas : $X_{11,}$, X_{12}, ..., X_{ij}, ..., X_{mn} (Representan la valoración relativa otorgada a la alternativa A_i en relación con el criterio C_j)

v Pesos relativos de los criterios de valoración : w_1, ..., w_j, ..., w_n (Cada uno refleja la importancia relativa de cada factor C_j en el conjunto de las características valoradas)

Aplicación

La aplicación de este método se basa en obtener para cada alternativa $A_{i,}$ las valorizaciones parciales relativas correspondientes X_{ij} y reducir finalmente la valoración de cada cosa ofertada mediante la aplicación de la siguiente expresión:

Valoración de $A_i = {}_j (X_{ij} * w_j)$

Para realizar este proceso de forma ordenada se deberán seguir los siguientes pasos:

1. Enumeración e identificación de las posibles alternativas A_i
2. Identificación de los factores susceptibles de valoración C_j
3. Obtención de los pesos relativos de los criterios w_j Para ello se recomienda asignar los pesos como porcentajes, reflejando de este modo su importancia relativa de cada factor en el proceso de decisión)
4. Obtención de las valoraciones parciales relativas X_{ij} y formación de la matriz (X_{ij}).

Una forma bastante directa de realizar las valoraciones parciales, consiste en comparar las características de cada equipo ofertado con las exigidas en las bases técnicas, asignando un valor unidad, si la característica en cuestión es idéntica a lo establecido en las bases y valores proporcionalmente más elevados, en la medida que sea más favorable que el valor mínimo exigido. Si alguna alternativa incumple de forma manifiesta los mínimos exigidos en las bases técnicas, deberá ser desechada.

5. Formación de la matriz ($X_{ij} * w_j$)
6. Obtención de la valoración de cada alternativa [WWW 01 020]

1.2.2 TQM (TOTAL QUALITY MANAGEMENT)

La metodología de gestión de calidad total, es un sistema que busca el éxito de la compañía, basado en la participación de todos tanto departamental como interdepartamental; la mejora continua, satisfacción de los clientes, la competitividad de la empresa y en la orientación a los recursos.

Esta metodología es la suma de calidad de lo que elaboramos lo que tiene relación con el proceso de fabricación de los productos, y la calidad de lo que hacemos, es decir la forma de gestionar la Organización; todo ello orientado hacia la mejora de los productos en el mercado. [WWW 01 021]

1.2.3 PRISMA (PLANIFICACIÓN DE RECURSOS INFORMÁTICOS Y SISTEMAS)

La metodología de Planificación de Recursos Informáticos y Sistemas, considera los aspectos estratégicos necesarios para la consecución de una planificación Top-Down (de arriba hacia abajo) de los recursos de información, lo cual se complementa con una implementación Bottom-Up (de abajo hacia arriba). Es conveniente utilizar estos dos modelos para planificar las aplicaciones informáticas, porque se compromete a las personas comenzando con la administración superior hacia abajo, y se estudia la organización desde lo general hasta el nivel de detalle, y así emprender la implementación de las aplicaciones hacia niveles superiores.

Los objetivos y propósitos principales de PRISMA son los siguientes:

- Detectar las áreas donde el empleo de la Tecnología informática permite a la organización incrementar si eficiencia.
- Proporcionar una planificación a mediano plazo de los proyectos informáticos, que soporte las necesidades de información, e incorporarlo a los planes estratégicos de toda la organización.
- Programar las inversiones efectuadas en sistemas, asegurando un equilibrio en costos, y que los beneficios se midan en relación con los objetivos planteados.
- Conformar soluciones informáticas estableciendo los recursos informáticos, de hardware, software, personal, capacitación y seguridad.

- Optimizar el uso de los recursos informáticos.

La figura 1.01 representa el esquema procedural de la metodología de planificación estratégica PRISMA y muestra la organización de sus fases. [LIB 01 001]

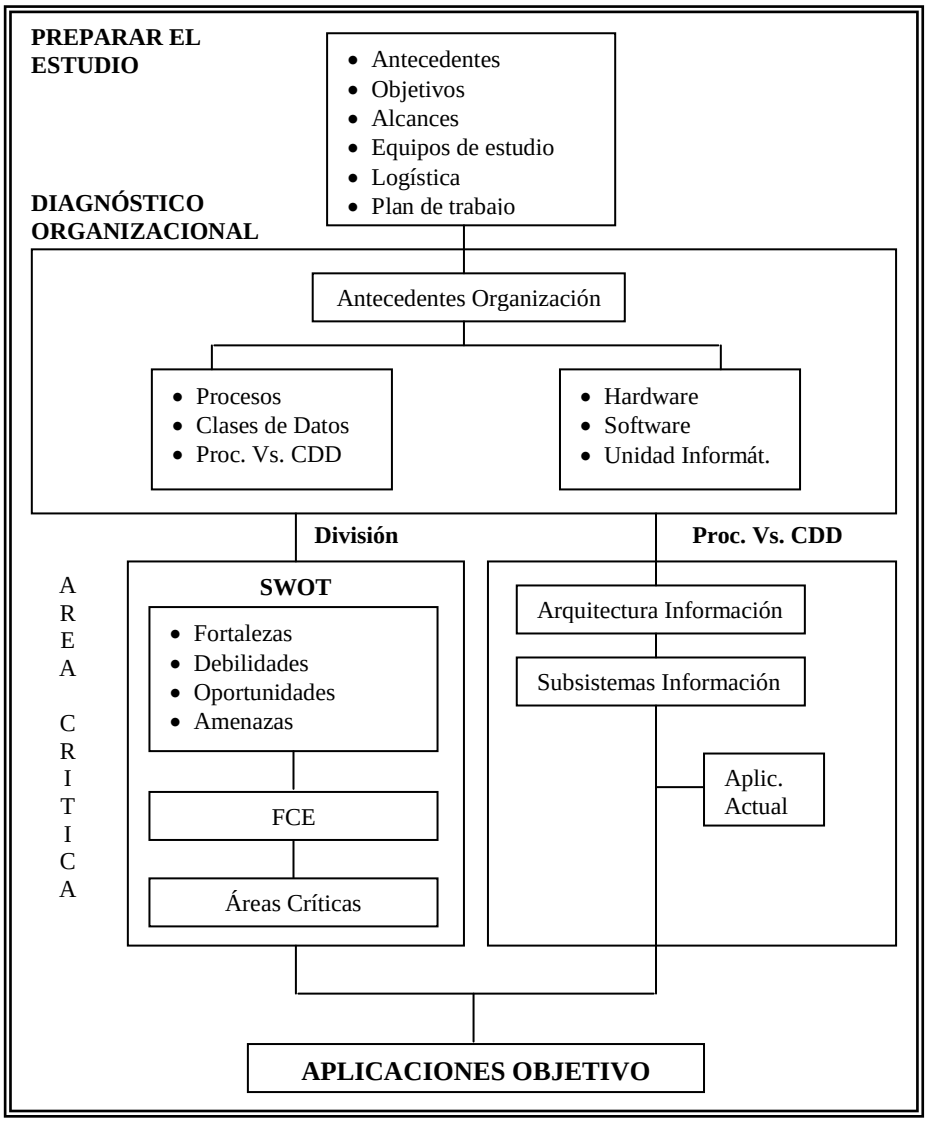


Fig. 1.01 Esquema Procedural de la Metodología Prisma

1.3 EVOLUCIÓN DE LOS LENGUAJES DE DESARROLLO Y BASES DE DATOS.

Los primeros orígenes de lo que hoy es una computadora, tuvieron lugar como respuesta a una de las más viejas aspiraciones del hombre: simplificar sus tareas. En realidad el hombre siempre buscó tener dispositivos que le ayudaran a efectuar cálculos precisos y rápidos; una breve reseña histórica nos permitirá, comprender cómo llegamos a las computadoras actuales.

Los chinos hace más de 3000 años A.C. desarrollaron el ABACO, con éste realizaban cálculos rápidos y complejos. PASCAL en 1642 crea una máquina mecánica de sumar, parecida al cuenta kilómetros que utilizan en la actualidad los automóviles. Pero ésta tenía algunos problemas con las sumas largas; en 1671 LEIBNITZ le agregó la posibilidad de restar, sumar, multiplicar y dividir.

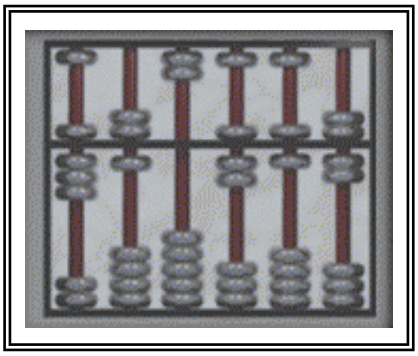


Fig. 1.02 Ábaco

Otra evolución en esta historia fue la que realizó BABBAGE, éste diseñó y desarrolló la primera computadora de uso general. Llamó a su descubrimiento

"Máquina de las diferencias". En 1833 con la ayuda de Lady Ada Augusta Lavolace (considerada la primera mujer programadora en tarjetas perforadas), concibió una segunda máquina que le llevó 20 años en construirla, la que era capaz de realizar una suma en segundos y necesitaba un mínimo tiempo de atención del operador. A ésta segunda máquina la llamó "Analítica".

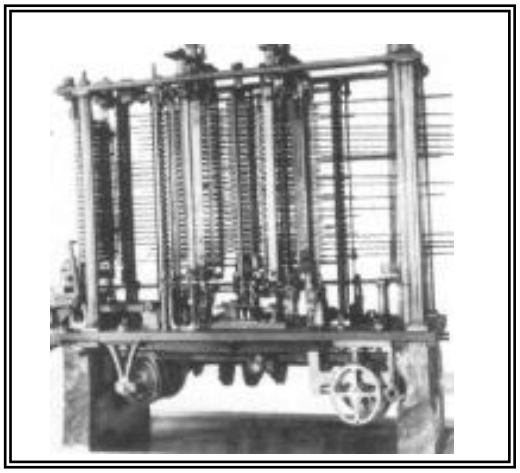


Fig. 1.03 Máquina diferencial

La primera operación de procesamiento de datos fue lograda en 1890 por HERNAN HOLLERITH. Éste desarrolló un sistema mecánico para calcular y agrupar datos de censos. El nuevo sistema se basaba en tarjetas perforadas. En 1896 HOLLERITH fundó la Tabuilating Machine Company que se fusionó en 1911 con otras para crear Computing-Tabulating-Recording Company. En 1924 el director general Thomas J. Watson, cambió su nombre a International Busines Machines Corporation IBM. [WWW 01 022]

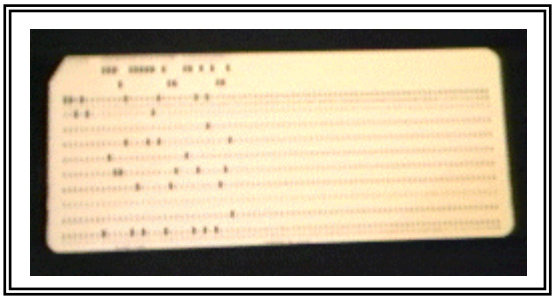


Fig. 1.04 Tarjetas Perforadas

En 1930, el norteamericano Vannevar Bush diseñó en el MIT (Massachusetts Institute of Technology) el analizador diferencial, marcando el inicio de nuestra era de computadoras; el "analizador" era una máquina electrónica que media grados de cambio en un modelo.

La primera computadora totalmente electrónica fue la ENIAC (Electric Numeric Integrator And Calculator), fue construida en 1943 y 1945 por JOHN MANCHI y J. PROPER ECKUT. Podía multiplicar 10.000 veces más rápido que cualquier máquina anterior, pero tenía sus problemas. Como estaba construida con casi 18,000 válvulas era enorme la energía que consumía y el calor que producía. Esto hacia que las válvulas se quemaran rápidamente y que las casas de alrededor tuvieran cortes de luz.

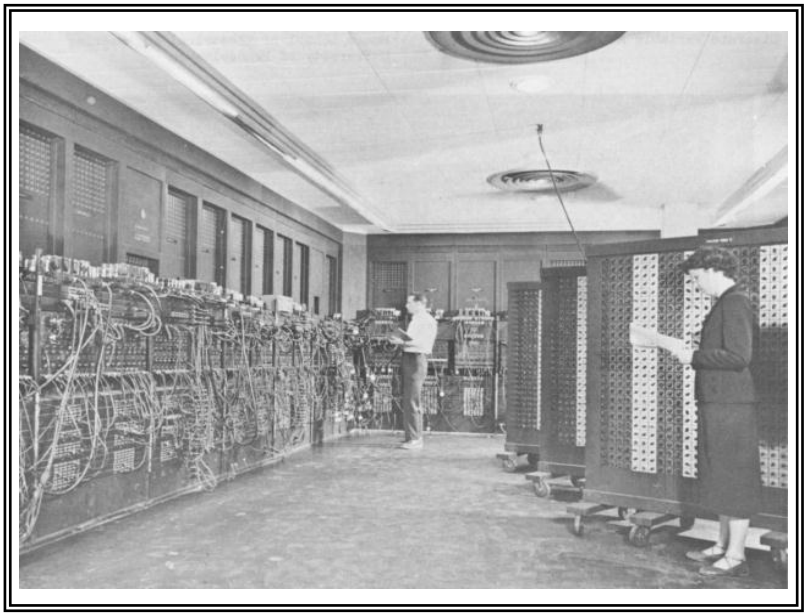


Fig. 1.05 ENIAC

El primer intento de sobreponerse a las limitaciones de velocidad y errores de cálculo fue de HOWARD AIKEN. Trabajó con ingenieros de I.B.M, crearon una calculadora automática Llamada MARK I (en 1944). Luego sé construyó MARK II. (Estas máquinas no pudieron satisfacer las necesidades de ese momento ya que eran millones los datos para guardar y resolver, aunque sirvieron de base para que cuando se crearan las válvulas al vacío comenzara la computación electrónica.

Fig. 1.06 MARK 1

En 1945 JHON VON NEWMAN estableció la base del programa almacenado, donde es fundamental para el futuro de las computadoras. El avance primario fue el proveer a la máquina de transferencia de control condicional y por almacenar todas las instrucciones del programa junta con los datos en la misma unidad de memoria.

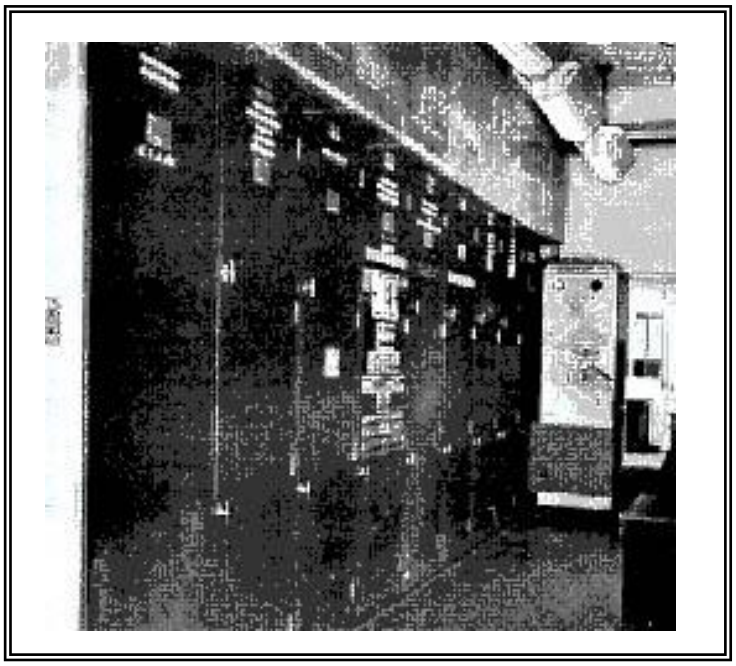


Fig. 1.07 EDVAC

A continuación se desarrolló el circuito integrado o "IC" que pronto recibiría el sobrenombre de "chip". Se atribuye el mérito de este invento a Robert Noyce. Pronto fue seguido por la capacidad de integrar hasta 10 transistores miniaturizados y eventualmente 1.000 piezas varias en el mismo espacio. Alrededor de 1971, el microprocesador había sido desarrollado por la nueva compañía de Noyce, Intel. Esta novedad colocó en un finito microchip los circuitos para todas las funciones usuales de un computador. Esto hizo que la computación fuera más rápida y más flexible, al tiempo que los circuitos

mejorados permitieron al computador realizar varias tareas al mismo tiempo y reservar memoria con mayor eficacia.

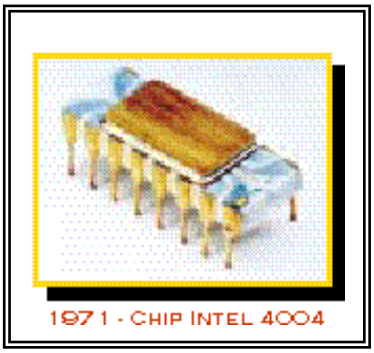


Fig. 1.08 Chip Intel 4004

Desde los años 80's surge la era más importantes para el desarrollo de las computadoras, la que se toma en cuenta según el microproceador, además se estandarizó su diseño y se abarataron costos. El equipo XT, año 1981, máquina pionera de los PC su gabinete era horizontal, el monitor era monocromático con fondo negro y letras de color verde.

Luego el desarrollo de las computadoras se vuelve imparable acreditable únicamente a grandes casas ensambladoras como IBM y COMPAQ, este desarrollo va de acuerdo a los microprocesadores. Así tenemos el equipo AT 80286 gabinete horizontal, los monitores eran a color con tecnología EGA (baja resolución comparados con los actuales VGA). Los AT 80386 con monitores súper VGA, módem interno de 14.400 baudios por segundo.

Para los equipos AT 80486, ya no se limita la creación a las grandes marcas sino que aparecen los denominados "clones" o genéricos que hasta hoy reciben la

preferencia de los usuarios por su bajo costo. Estos tienen monitores con resolución .28. Pasamos luego a los actuales Equipos Pentium (r) y Pentium Celeron que se diferencian por su procesador.

Las generaciones de Pentium II, Pentium III y Pentium IV básicamente utilizan los mismos elementos de base Otros procesadores: los K6-2 y Athlon de AMD.

Esta ha sido la evolución de la parte dominante de los computadores las denominadas PC, las que hasta hoy conforman la mayor parte del mercado informático. Pero existe otro tipo de computadores que también, aunque en menor proporción abarcan la atención de los usuarios, las llamadas MAC de APPLE.

Steven Wozniak y Steven Jobs luego de abandonar sus estudios universitarios se emplearon en Hewlett-Packard y en Atari). Wozniak, que había estado trabajando en el diseño de una computadora por un tiempo, diseñó en 1976 lo que sería la Apple I. Jobs, que tenía visión de futuro, insistió en que él y Wozniak comercializaran la máquina, y el 1 de abril de 1976 nació Apple Computer.

Los interesados en las computadoras no tomaron muy en serio a la Apple I y Apple no comenzó a crecer hasta 1977, cuando la Apple II hizo su aparición en una exposición local de computadoras. La Apple II fue una máquina que llamaba la atención, porque era la primera computadora personal que venía en

una carcasa de plástico e incluía gráficos en color. a principios del '78, las ventas crecieron aún más con la introducción de la Apple Disk II, la disquetera más económica y fácil de usar (para la época). En 1979, Jobs comenzó a trabajar en el Macintosh, cuyo objetivo era una computadora personal de US$ 500. Jobs se encargó de que fuera mucho más.

En 1985, Apple puso todo su empeñó en una batalla judicial contra Bill Gates, de Microsoft, por la introducción del Windows 1.0, que era muy similar a la GUI (siglas de Graphical User Interface – Interface Gráfica de Usuario) del Mac. Gates, finalmente, acordó firmar una declaración por la que Microsoft se comprometía a no usar tecnología del Mac en Windows 1.0, pero nada se decía allí sobre las futuras versiones de Windows. Apple había perdido, efectivamente, los derechos exclusivos sobre su diseño de interface.

En 1987, Apple introdujo al mercado la Mac II. Concebida para ser expandible, la Mac II convirtió a la línea Macintosh en una familia de computadoras viable y poderosa. No fue así. En 1990 el mercado se saturó con clones de PC con todas las configuraciones imaginables, y Apple era la única compañía que vendía Mac's. A fines de mayo, Microsoft presentó el Windows 3.0, que podía ejecutarse prácticamente en todos los clones de PC del mundo.

En 1997 Apple sostuvo una alianza con Microsoft. A cambio de US$ 150 millones en acciones de Apple, Microsoft y Apple tendrían una licencia cruzada de cinco años sobre patentes, pero quedaba todavía un obstáculo más grande por

superar: los clones. Los clones no habían podido expandir el mercado del Mac OS y, en cambio, le habían quitado clientes a Apple.

El 10 de noviembre de 1997 Apple anunció dos nuevos equipos de Apple: la PowerMac G3 y el PowerBook G3. En julio de 1999, cuando lanzó la iBook, se llenó el cuarto y último casillero de la "Matriz de Productos de Apple", en julio de 2000, Apple anunció un gran número de máquinas nuevas, entre ellas la PowerMac G4 Cube, que agregó una quinta categoría a la anterior estrategia de Apple de cuatro líneas de producto. El Cube fue la respuesta de Apple a quienes querían una iMac sin monitor, y al mismo tiempo un desafío a la industria de la informática para continuar reduciendo el tamaño de las computadoras mientras aumenta su atractivo visual.

Todo lo que hemos visto constituye la evolución de la estructura electromecánica de la computadora o lo que llamamos Hardware, pero el hardware por si solo no puede hacer nada, pues es necesario que exista la estructura lógica llamada Software, definido como el conjunto de instrucciones que las computadoras emplean para manipular datos. Sin el la computadora sería un conjunto de medios sin utilizar. [WWW 01 023]

Tabla 1.01 Cuadro Resumen de la evolución de la estructura electromecánica.

Nombre	Año	Autor	Característica
Abaco	5000 años atrás		Se utiliza en la educación, principios de conteo aritmético.
Pascaline	1642	Blas Pascal	Solo sumar y restar, ocupa una caja de zapatos. Su diseño se utilizó en las calculadoras mecánicas de los años 60´s. (Leonardo de Vince tuvo una visión 150 años antes).
M.Diferencial	1822	Charles Babbage	Calcula tablas matemáticas impulsada con vapor, no fue terminada y se corto el presupuesto en 1842; tenía 2 m de alto, 3 m de longitud y 4000 partes, pesando 3 ton.
M.Analítica	1833-50	Charles Babbage	Incluía una unidad de almacenamiento +, - ,* , / en 60 op/min. Era impulsada por una locomotora y ocupaba un campo de Fútbol.
M. Tabularora	1887-90	Hernan Hollerith	Máquina tabuladora con tarjetas perforadas, acumulaba y clasificaba la información. Se utilizó para el censo de 1890 y le redituó 40,000 dólares y el Gobierno de los Estados Unidos se ahorro 5 millones de dólares.
MARK I	1944	Howard Aiken	Primera computadora electromecánica, 17 m largo y 2.5 m de alto. Un adelanto significativo, pero IBM no creía que sustituiría a la de tarjetas perforadas.
ENIAC	1946	J.Presper Eckert John	Electronic Numerical Integrator and Computer (Integrador Numérico Electrónico y

		W. Mauchly	Computadora). Se utilizó en la 2a Guerra Mundial en cálculos balísticos. Su tamaño fue de 1400 m2, 30 ton y de 1800 tubos al vacío; cuando funcionaba dejaba sin electricidad a Filadelfia.
EDVAC	1945	John von Newman	Trabajo con Eckert y Mauchly para su construcción. Este grupo incluyó en sus equipos memoria RAM.
ERA DEL COMPUTA-DOR			
XT	1981	IBM	Con procesador 8086 u 8088 de Intel, usaba 128K, 256 o 512 Kilo bites de memoria Ram, disco duro de 5 o 10 megabytes de capacidad, Su velocidad de procesamiento estaba entre los 4 y 8 MHz.
AT 80286		COMPAQ IBM	16 MB DE RAM, adicionándole una placa de expansión especial, el disco duro normal para él era de 30 o 40MB, las unidades de disquetes de 1.2 y 1.44 MB de capacidad. Su velocidad promediaba los 25MHz.
AT 80386			Usaban en promedio 8 y 16 MB de ram, utilizando módulos removibles de memoria, tipo SIMM de 32 pines, el disco disco duro promediaba los 512 MB, Velocidad promedio: 40MHz. Aparece la unidad Cd rom de simple velocidad.
AT 80486	1990		Ram promedio de 8 y 16 MB, discos duros de 1 gigabyte promedio, módem de 28.800 bps.

			Unidad de CD ROM de 2, 4 y 8 velocidades.
Equipos Pentium			Equipos Pentium de 75MZ, ram promedio de 16 MB expandible a 128 MB, discos duros de 3 gigas o mayor, módem de 33.600 bps, coprocesador matemático y memoria caché interna. Unidad de CDROM de 16 velocidades.
Pentium Celeron			Con velocidades desde 300 MHz a 1.3 GHz, discos duros mayores de 6 gigas como promedio, ram promedio de 32 MB expandible, motherboard multifuncional, tanto en equipos genéricos como de marca, incluyen normalmente sonido, video y módem fax incorporado en la placa madre, la velocidad de los módems promedio es de 56.600 bps, las unidades de CDROM alcanzan velocidades de 40X.
Pentium II, III y IV			Motherboard multifuncional, ram promedio de 128 MB, discos de 15, 30, 40 o más gigas, multimedias de 52x, módem de 56.600, y sus velocidades varían desde los 350MHz a 550MHz los pentium II, de 500, 1 GHz los pentium III y 1.4 a 2GHz los pentium IV.
APPLE	1976	Steven Wozniak y Steven Jobs	Los interesados en las computadoras no tomaron muy en serio a la Apple I y Apple no comenzó a crecer hasta 1977, cuando la Apple II hizo su aparición en una exposición local de computadoras. La Apple II fue una máquina que llamaba la atención, porque era la primera computadora personal que venía en una carcasa

			de plástico e incluía gráficos en color. a principios del '78, las ventas crecieron aún más con la introducción de la Apple Disk II, la disquetera más económica y fácil de usar (para la época). En 1979, Jobs comenzó a trabajar en el Macintosh, cuyo objetivo era una computadora personal de US$ 500. Jobs se encargó de que fuera mucho más.

[WWW 01 024]

1.3.1 EVOLUCIÓN DE LOS LENGUAJES DE PROGRAMACIÓN.

El Software es un conjunto de programas, documentos, procedimientos, y rutinas asociados con la operación de un sistema de cómputo. Distinguiéndose de los componentes físicos llamados Hardware.

Es en general el conjunto de instrucciones individuales que se le proporciona al microprocesador para que pueda procesar los datos y generar los resultados esperados.

Los lenguajes de programación cierran el abismo entre las computadoras, que sólo trabajan con números binarios, y los humanos, que preferimos utilizar palabras y otros sistemas de numeración. Son los programas mediante los que se indica a la computadora que tarea debe realizar y cómo

efectuarla, pero para ello es preciso introducir estas órdenes en un lenguaje que el ordenador pueda entender. En principio, el ordenador sólo entiende las instrucciones en código máquina, es decir, el específico de la computadora. Sin embargo, a partir de éstos se elaboran los llamados lenguajes de alto y bajo nivel.

El software como componente principal del ordenador, surge a mediados de la década de los 60, como integrante del propio hardware, es decir, los fabricantes que diseñaban el hardware elaboraban su propio software, dependiendo de la misma empresa productora. Fue hasta el año de 1968 que se convocó a una reunión en Garmisch, Alemania Oriental, estimulándose el interés hacia los aspectos técnicos y administrativos utilizados en el desarrollo y mantenimiento de software.

John Von Neumann desarrolló el modelo que lleva su nombre, para describir este concepto de "programa almacenado". En este modelo, se tiene una abstracción de la memoria como un conjunto de celdas, que almacenan simplemente números. Estos números pueden representar dos cosas: los datos, sobre los que va a trabajar el programa; o bien, el programa en sí.

¿Cómo es que describimos un programa como números? Se tenía el problema de representar las acciones que iba a realizar la computadora, y que la memoria, al estar compuesta por switches correspondientes al

concepto de bit, solamente nos permitía almacenar números binarios. La solución que se tomó fue la siguiente: a cada acción que sea capaz de realizar nuestra computadora, asociarle un número, que será su código de operación (opcode). Por ejemplo, una calculadora programable simple podría asignar los opcodes :

1 = SUMA 2 = RESTA 3 = MULTIPLICA 4 = DIVIDE.

Cadro 1.1 Opcode utilizado por calculadora.

Supongamos que queremos realizar la operación 5 * 3 + 2, en la calculadora descrita arriba. En memoria, podríamos "escribir" el programa de la siguiente forma:

Localidad Opcode Significado Comentario 0 5 5 En esta localidad, tenemos el primer número de la fórmula 1 3 * En esta localidad, tenemos el opcode que representa la multiplicación. 2 3 3 En esta localidad, tenemos el segundo número de la fórmula 3 1 + En esta localidad, tenemos el opcode que representa la suma. 4 2 2 En esta localidad, tenemos el último número de la fórmula

Podemos ver que con esta representación, es simple expresar las operaciones de las que es capaz el hardware (en este caso, nuestra calculadora imaginaria), en la memoria.

La descripción y uso de los opcodes es lo que llamamos lenguaje de máquina. Es decir, la lista de códigos que la máquina va a interpretar como instrucciones, describe las capacidades de programación que tenemos de ella; es el lenguaje más primitivo, depende directamente del hardware, y requiere del programador que conozca el funcionamiento de la máquina al más bajo nivel.

El primer gran avance que se dio, fue la abstracción dada por el Lenguaje Ensamblador, y con él, el nacimiento de las primeras herramientas automáticas para generar el código máquina. Esto redujo los errores triviales, como podía ser el número que correspondía a una operación, que son sumamente engorrosos y difíciles de detectar, pero fáciles de cometer. [WWW 01 025]

En los años 50s y 60s, con el desarrollo de algoritmos de más elevado nivel, y el aumento de poder del hardware, empezaron a entrar al uso de computadoras científicos de otras ramas; ellos conocían mucho de Física, Química y otras similares, pero no de Computación, y por supuesto, les era sumamente complicado trabajar con lenguaje Ensamblador en vez de fórmulas. Así, nació el concepto de Lenguaje de Alto Nivel, con el primer compilador de FORTRAN (FORmula TRANslation), que, como su

nombre indica, inició como un esfuerzo de traducir un lenguaje de fórmulas, al lenguaje ensamblador y por consiguiente al lenguaje de máquina. A partir de FORTRAN, se han desarrollado innumerables lenguajes, que siguen el mismo concepto: buscar la mayor abstracción posible, y facilitar la vida al programador, aumentando la productividad, encargándose los compiladores o intérpretes de traducir el lenguaje de alto nivel, al lenguaje de computadora.

Los lenguajes de bajo nivel utilizan códigos muy cercanos a los de la máquina, lo que hace posible la elaboración de programas muy potentes y rápidos, pero son de difícil aprendizaje. Por el contrario, los lenguajes de alto nivel son de uso mucho más fácil, ya que en ellos un solo comando o instrucción puede equivaler a millares en código máquina. El programador escribe su programa en alguno de estos lenguajes mediante secuencias de instrucciones. Antes de ejecutar el programa la computadora lo traduce a código máquina de una sola vez (lenguajes compiladores) o interpretándolo instrucción por instrucción (lenguajes intérpretes).

Dentro de los lenguajes de alto nivel se ha presentado una evolución, la que ha partido de los lenguajes de instrucciones sintácticas secuenciales como lo eran los lenguajes Basic, ForTran. Los lenguajes de " Alto Nivel" son los más utilizados como lenguaje de programación. Aunque no son fundamentalmente declarativos, estos lenguajes permiten que los algoritmos se expresen en un nivel y estilo de escritura fácilmente legible y

comprensible por otros programadores. Además, los lenguajes de alto nivel tienen normalmente las características de "Transportabilidad". Es decir, están implementadas sobre varias máquinas de forma que un programa puede ser fácilmente "Transportado" (Transferido) de una maquina a otra sin una revisión sustancial. En ese sentido se llama "Independientes de la maquina". Luego aparece el concepto de la programación estructurada en Pascal y C.

Actualmente aparece un nuevo concepto, la Programación Orientada a Objetos, sus inicios y técnicas de programación se iniciaron a principios de los 70s por David Parnas. Se puede definir programación orientada a objetos (POO) como una técnica de programación que utiliza objetos como bloque esencial de construcción. La POO, es un tipo de programación más cercana al razonamiento humano. La POO surge como una solución a la programación de grandes programas, y para solventar el mantenimiento de dichas aplicaciones, ya que en la programación estructura el más mínimo cambio supone la modificación de muchas funciones relacionadas, en cambio con la POO solo es cuestión de añadir o modificar métodos de una clase o mejor, crear una nueva clase a partir de otra (Herencia).

TABLA 1.02 Cuadro Resumen de lenguajes de Programación

ORIGEN DEL NOMBRE	LENGUAJE	COMENTARIO	AÑO
StriNg Oriented symBOlic Languaje	SNOBOL	Creado para Inteligencia Artificial por Mc. Cartly.	1950
FORmula TRANslation	FORTRAN	Lenguaje de programación compilado de alto nivel, inicialmente diseñado para uso científico y de ingeniería: precursor de muchos conceptos como : variable, expresiones condicionales, compilación separada. Creada por Backups.	1954
Common Business Oriented Language	COBOL	Lenguaje de programación muy parecido al inglés, con mucho énfasis en las estructuras de datos; muy usado especialmente en los negocios. Creado por el Dpto. De defensa de los EE.UU.	1959
List Processing	LISP	Lenguaje de programación orientado a las listas, principalmente usado en la manipulación de listas e investigación; Considerado en lenguaje de programación standard para los proyectos de Inteligencia Artificial	1960
ALGOrithmic Language	ALGOL	Primer lenguaje de programación procedural estructurado; usado principalmente para resolver problemas matemáticos	1960
A Programming Language	APL	Lenguaje interpretado que usa un gran número de símbolos especiales; usado especialmente por matemáticos	1961

Blaise Programming Language One	PL/I	Diseñado para combinar las características de FORTRAN, COBOL Y ALGOL, es un lenguaje de programación complejo, compilado y estructurado; tiene algunos errores de manejo y de multitareas; usado en algunas academias y en ambientes de investigación	1964
Beginners All - Purpose Simbolic Instruction Code	BASIC	Lenguaje de programación de alto nivel, muy popular; usado principalmente por programadores novatos	1965
Derivado del griego logos que significa "palabra"	LOGO	Lenguaje de programación muy usado por los niños; posee un ambiente simple de dibujo y varias características de alto nivel de Lisp; es un lenguaje primordialmente educativo	1968
Blaise Programmed Inquiry Language Or Teaching	PILOT	Lenguaje de programación usado principalmente para crear aplicaciones para instrucciones de ayuda por computadora; contiene una sintaxis muy pequeña	1969
FOURTH Generation Language	FORTH	Lenguaje de programación interpretado y estructurado; fácilmente extensible; provee gran funcionalidad en espacio limitado	1970
Blaise PASCAL, inventor y matemático	PASCAL	Lenguaje de programación compilado y estructurado basado en Algol; añade tipos de datos y estructuras que simplifican las sintaxis; al igual que el lenguaje C, es un lenguaje de desarrollo estándar de microcomputadoras	1971
Su predecesor	C	Lenguaje de programación compilado y	1972

fue el lenguaje de programación B, de los Laboratorios BELL		estructurado; usado en muchos lugares de trabajo por la fácil transportabilidad de sus programas	
Llamado así por lo "Básico" de sus sentencias	BASIC	Creado por Bob Albrech y Dennis Allison. Empiezan los lenguajes con instrucciones tipo sentencias, pero todavía no es un lenguaje estructurado.	1975
Augusta ADA Byron	ADA	Derivado del Pascal, usado principalmente el aspecto militar	1979
MODULAR language; diseñado como fase segunda de Pascal (Niclaus Wirth diseñó ambos)	MODULA - 2	Lenguaje de programación de alto nivel basado en Pascal, que enfatiza la programación modular, posee un conjunto de funciones y procedimientos estándar	1980
Sucesor de Basic, en una versión estructurada y gráfica	VISUAL BASIC	Bill Gates y Paúl Allen de Microsoft, escriben una versión de BASIC, con un entorno gráfico y dispositivos de multimedia y enlaces a bases de datos.	1992
Derivado del Lenguaje C	C++	Lenguaje de programación compilado y estructurado; con librerías y comienza la terminología de orientado a Objetos. Creado por Borland International Inc.	1995
Practical	PERL	Es un lenguaje creado por Larry Wall con el	1997

Extraction and Report Language		objetivo principal de simplificar las tareas de administración de un sistema UNIX.	
Coffee	JAVA	Es un lenguaje con tintes revolucionarios, creado por James Gosling e introducido por Sun Microsystems con la idea de que sea pequeño, simple y portable. Principal aplicación programación para web	1995

[REV 01 010]

[WWW 01 026]

[WWW 01 027]

1.3.2 EVOLUCIÓN DE LAS BASES DE DATOS

Las Bases de datos se han convertido en un producto estratégico de primer orden, al constituir el fundamento de los sistemas de información. Desde que se empezaron a introducir los ordenadores para automatizar la gestión de las empresas en la década de los sesenta, la evolución de los sistemas de información ha tenido una considerable repercusión en la gestión de los datos, desplazándose el centro de gravedad de la informática, que estaba situado en el proceso, hacia la estructuración de los datos. Es así que en la década de los 60 hasta la actualidad, han sucedido tres generaciones distintas de BDDs, constituidas en tres modelos de desarrollo diferentes como son el jerárquico, en red y relacional. [WWW 01 028]

Modelo Jerárquico.- Dominó el mercado hasta mediados de los 80. Durante este mismo período, surgió el modelo en red con el que se pretendía sustituir a las BDDs jerárquicas, lo que no se consiguió. Los sistemas de BDDs jerárquicos fueron los primeros en aparecer. Una base de datos jerárquica se puede visualizar como una estructura en árbol, bastante rígidas. Una vez diseñada la base de datos, es complejo cambiarla y, además, es necesario un conocimiento amplio de la forma en la que se han almacenado los datos para poder recuperarlos de forma efectiva (fig. 1.09). Por ello, a pesar de haber dominado el mercado en sus comienzos, estas BDDs han ido decayendo y actualmente no se encuentran en el mercado.

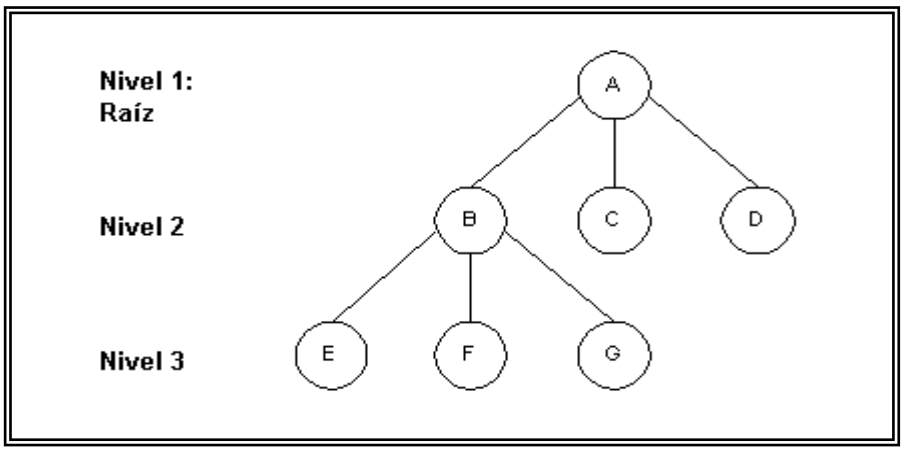


Fig. 1.09 Estructura de Datos Jerárquico

Modelo en Red.- Surge así, a finales de los sesenta y principios de los setenta, la primera generación de productos de bases de datos en red. Las mismas que fueron una evolución del modelo jerárquico. En una base de datos en red, cada uno de los registros están enlazados entre sí, pero no

necesariamente siguiendo una estructura en árbol. El modelo en red elimina parte de la rigidez del modelo jerárquico, pero aumenta la complejidad para modificar la estructura de la base de datos (fig. 1.10). Por ello, a pesar de su buen rendimiento, el número de instalaciones con BDDs en red siempre ha sido pequeño y, hoy en día, tampoco se encuentran en el mercado. Sin embargo, aún quedan instalaciones basadas en estos dos modelos de datos, que responden con gran eficiencia y plena satisfacción de sus usuarios.

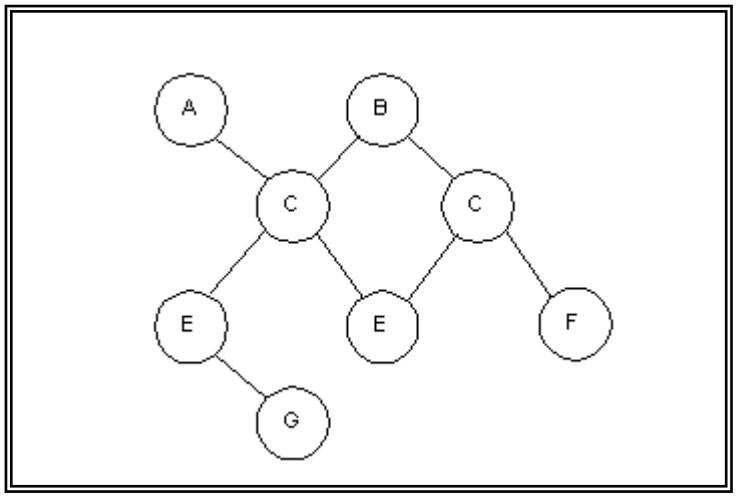


Fig. 1.10 Estructura de Datos en Red

Modelo Relacional.- En 1970, el Dr. Codd propuso el modelo relacional, no podía pensar que lo que se consideraba más bien una elegante teoría matemática sin posibilidad de implementación eficiente en productos comerciales iba a convertirse, en los años ochenta, en la segunda generación de productos de bases de datos, que actualmente domina el mercado.

En una base de datos relacional, se representan los datos como un conjunto de tablas bidimensionales compuestas de filas y columnas. Cada fila representa una relación entre un conjunto de valores y está identificada por una clave única. Las BDDs relacionales son muy flexibles y de fácil manejo. Un factor decisivo en la implantación de los BDDs relacionales, ha sido el lenguaje SQL (Structured Query Language) para la interrogación y el manejo de datos del modelo relacional. [WWW 01 029]

Se caracteriza por proporcionar capacidades de gestión de datos, objetos y gestión de conocimiento y pretende responder a las necesidades de aplicaciones tales como: CASE (Ingeniería del software asistida por ordenador), CAD/CAM/CIM, SIG (sistemas de información geográfica), información textual, aplicaciones científicas, sistemas médicos, publicación digital, educación y formación, sistemas estadísticos, comercio electrónico, etc.

Los términos formales del modelo relacional a menudo son sustituidos por otros de uso más común, debido a que estos términos son demasiado abstractos para ser usados en la práctica.

Tabla 1.03 Términos formales del modelo relacional

Término relacional formal	Equivalente informal
Relación	Tabla
Tupla	Fila o registro

Cardinalidad	Número de filas o registros
Atributo	Columna o campo
Grado	Número de columnas o campos
Clave primaria	Identificador único
Dominio	Fondos de valores legales

Como ejemplos de estos tres modelos se puede citar a las siguientes Bases de Datos:

Tabla 1.04 Modelos de Bases de Datos

Modelo Jerárquico	- IMS (Information Management System) de IBM Corporation - SYSTEM 2000 desarrollado por MRI Corporation y más tarde adquirido por Intel Corporation
Modelo en Red	- DMS 1100 de UNIVAC - Total de Cincom - IDMS de Cullinane - EDMS, de Xerox - PHOLAS de Philips - DBOMP de IBM - IDS de Honeywell
Modelo Relacional	- DB2 - SQL/DS de IBM - Progress - Informix - Ingress - Sybases - Oracle

A la hora de clasificar estos avances en el campo de las bases de datos, podemos identificar tres dimensiones: rendimiento, funcionalidad/ inteligencia y distribución/ integración.

Tabla 1.05 Dimensiones de calificación de BDD.

Dimensiones de clasificación:	**Comentario:**	**Destacan los siguientes tipos de tecnologías:**
Rendimiento	Los datos almacenados crecen de forma exponencial, se empieza a hablar de bases de datos de *petabytes* (10 15).	- bases de datos paralelas - bases de datos en tiempo real - bases de datos en memoria principal.
Inteligencia	La funcionalidad de las bases de datos ha ido aumentando de forma considerable, pues gran parte de la semántica de los datos que se encontraba dispersa en los programas ha ido migrando hacia el servidor de datos. También el tiempo y la incertidumbre se suman a ello.	- bases de datos activas - deductivas - orientadas a objetos - multimedia - temporales - seguras - difusas - Los almacenes de datos (datawarehousing) - minería de datos (datamining).
Distribución	El avance espectacular de las comunicaciones así como la difusión cada día mayor del fenómeno Internet/Web, ha revolucionado el mundo de las bases de datos. También la aparición de la informática	- bases de datos distribuidas - federadas - multibases de datos - bases de datos móviles - bases de datos y web.

	móvil o computación nómada obliga a replantearse algunos conceptos fundamentales de las bases de datos.	

Es así que dando un paso en la evolución de las Bases de Datos, aparecen las ***Bases de Datos Orientadas a Objetos*** (BDOO) que han sido diseñadas para soportar el análisis, diseño y programación Orientada a Objetos. Estás permiten el desarrollo y mantenimiento de aplicaciones complejas pues se puede utilizar un mismo modelo conceptual y así aplicarlo al análisis, diseño y programación, esto reduce el problema entre los diferentes modelos a través de todo el ciclo de vida, con un costo significativamente menor. Además las BDOO ofrecen un mejor rendimiento de la máquina que las bases de datos relacionales, para aplicaciones ó clases con estructuras complejas de datos. Sin embargo, las BDOO coexistirán con las bases de datos por relación como una forma de estructura de datos dentro de una BDOO.

Se puede indicar algunas desventajas al considerar la adopción de la tecnología orientada a objetos, la inmadurez del mercado de BDOO constituye una posible fuente de problemas por lo que debe analizarse con detalle la presencia en el mercado del proveedor para adoptar su producto en una línea de producción sustantiva. Por eso, algunas casas distribuidoras y diseñadores de bases de datos proponen que se explore esta tecnología en un proyecto piloto.

El segundo problema es la falta de estándares en la industria orientada a objetos. Sin embargo, el "Grupo Manejador de Objetos" (OMG), es una organización Internacional de proveedores de sistemas de información y usuarios dedicada a promover estándares para el desarrollo de aplicaciones y sistemas orientados a objetos en ambientes de cómputo en red. La implantación de una nueva tecnología requiere que los usuarios iniciales acepten cierto riesgo. Aquellos que esperan resultados a corto plazo y con un costo reducido quedarán desilusionados. [WWW 01 030]

1.3.2.1 Definición de Base de Datos y Conceptos Básicos

Base de Datos, es un conjunto de datos no redundantes, almacenados en un soporte informático, organizado de forma independiente de su utilización y accesible simultáneamente por distintos usuarios y aplicaciones.

Es decir, la diferencia de una BD respecto a otro sistema de almacenamiento de datos, es que éstos se almacenan de forma que cumplan tres requisitos básicos:

Cuadro 1.2. Requisitos Básicos para una BBD.

No redundancia	Los datos se almacenan una sola vez. Si varias aplicaciones necesitan los mismos datos, no crearán cada una su propia copia sino que todas accederán a

	la misma
Independencia	Los datos se almacenan teniendo en cuenta la estructura inherente a los propios datos y no la de la aplicación que los crea. Esta forma de trabajar es la que permite que varias aplicaciones puedan utilizar los mismos datos
Concurrencia	Varios usuarios, ejecutando la misma o diferente aplicación, podrán acceder simultáneamente a los datos

Los principales conceptos que se manejan en bases de datos son:

Cuadro 1.3. Principales Conceptos de BBD.

Diccionario de datos	Reúne la información sobre los datos almacenados en la BD (descripciones, significado, estructuras, consideraciones de seguridad, edición y uso de las aplicaciones, etc.)
Repositorio	Es una BD utilizada como soporte en el Ciclo de Vida del Desarrollo de Sistemas de Información con Base de Datos. Permiten describir y registrar las características o atributos de cada componente u objeto de un Sistema o Base de Datos
Modelo de datos	Es un conjunto de conceptos, reglas y convenciones que permiten describir y manipular los datos
Modelo relacional	En este modelo, los datos se estructuran en tablas manteniendo la independencia de esta estructura lógica, respecto al modo de almacenamiento u otras características físicas. Las tablas se manejan mediante

	operaciones de la teoría de conjuntos y el álgebra relacional
DDL (Data Definition Language)	Lenguaje de definición de datos, se utiliza para crear y mantener la base de datos y los elementos que contiene a nivel externo, lógico e interno. Permite definir entidades, identificadores (claves), atributos, interrelaciones, autorizaciones de acceso, restricciones de integridad, etc. A nivel interno facilita la definición del espacio físico, longitud de los campos, representación de los datos (binario, alfanumérico...), caminos de acceso (punteros, índices...), etc.
DML (Data Manipulation Language).	Lenguaje de manipulación de datos, se utiliza para la actualización y consulta de los datos almacenados en la base de datos. Permite añadir, seleccionar, suprimir o modificar los datos de la BD, respetando las reglas establecidas por el DDL
SQL (Structured Query Language)	El SQL es un lenguaje de alto nivel, no procedual, normalizado, que permite la consulta y actualización de los datos de BD relacionales. Se ha convertido en el estándar de acceso a BD relacionales. La primera versión se aprobó como norma ISO en 1987 y la segunda, conocida como SQL2 y vigente actualmente, en 1992. Actualmente se trabaja en la norma SQL3 que soportará bases de datos orientadas a objetos y bases de datos activas
El SQL	facilita un lenguaje de definición de datos y un lenguaje de manipulación de datos. Además, incluye un interfase que permite el acceso y la manipulación de la BD a usuarios finales. El SQL estándar no es un

	lenguaje de programación, aunque sus sentencias se pueden utilizar en lenguajes de tercera generación como COBOL o FORTRAN
Transacción	Conjunto de modificaciones sobre una BD que son una unidad inseparable. Es decir, si se realiza alguna de las modificaciones deben realizarse todas, en caso contrario no debe realizarse ninguna
Commit	Los SGBDs ofrecen sentencias especializadas para la gestión de transacciones. La nomenclatura habitual en bases de datos relacionales es: COMMIT WORK, finalizar una transacción y ROLLBACK, deshacerla
Two-Phase Commit	Proceso necesario para realizar Commit en BD distribuidas. Para garantizar que todas las BD involucradas quedarán correctamente modificadas, el Commit se divide en dos fases. Primero, se comprueba que todos los nodos involucrados están listos para realizar la actualización. Segundo, se modifican las bases de datos si, y sólo, si todos los nodos están preparados
Bloqueo	Cuando una transacción necesita asegurarse de que el contenido de un recurso de la BD (un archivo, un registro u otro) no cambiará hasta que la transacción finalice, se bloquea. El bloqueo impide que otras transacciones lo modifiquen. Existen dos tipos principales de bloqueos: bloqueos exclusivos y bloqueos compartidos. Si una transacción realiza un bloqueo exclusivo sobre un recurso, ninguna otra podrá ejecutar ningún tipo de bloqueo contra el recurso. Se utilizan cuando la transacción va a

	actualizar el recurso. Si una transacción realiza un bloqueo compartido, otras transacciones podrán realizar bloqueos compartidos (pero no exclusivos) sobre ese mismo recurso. Esta última técnica se utiliza cuando la transacción no va a actualizar los datos, pero desea evitar que otras transacciones puedan modificarlo
Interbloqueos	Los interbloqueos se producen cuando dos transacciones que acceden a una base de datos, se bloquean mutuamente al intentar realizar un bloqueo exclusivo sobre los mismos recursos. Todo SGBD debe implementar técnicas automáticas para evitar los interbloqueos, ya que si se producen, ninguna de las transacciones puede continuar y permanecerán en ese estado, hasta que el SGBD lo resuelva
Inconsistencia	Una base de datos está inconsistente si dos datos que deberían ser iguales no lo son. Por ejemplo, un empleado aparece en una tabla como activo y en otra como jubilado
Integridad.	Se refiere a la exactitud y precisión de la información. El dato debe permanecer como fue colocado inicialmente, sin ser modificado si no cuenta con la respectiva autorización
Redundancia controlada	En ocasiones, es necesario introducir voluntariamente redundancia en la BD por consideraciones de rendimiento. En estos casos los administradores del sistema repiten conscientemente algunos datos y, a la vez, preparan al sistema para mantener automáticamente las distintas copias y que no se pierda

	la integridad
Confidencialidad.	Consiste en proteger la BD contra accesos no autorizados. Debe asegurarse no sólo que los usuarios no autorizados no consigan acceso a la BD sino, también, que los usuarios legítimos acceden sólo a los datos autorizados
Recuperación	Su objetivo es proteger a la BD contra fallos (lógicos o físicos) que destruyan su contenido parcial o totalmente. Los SGBDs suelen incluir los llamados "ficheros de log", en los que se almacenan todos los cambios antes de almacenarlos en la BD, así como, marcas de comienzo y final de transacción. A partir de ellos, el SGBD puede decidir después de un fallo, si una transacción estaba terminada o no y, por tanto si hay que mantenerla o deshacerla
Normalización	Según el modelo relacional, las tablas deben definirse siguiendo una serie de reglas precisas, para asegurarse de que no se producirán anomalías en la actualización de la base de datos. Para ello, es habitual que se necesite descomponer las tablas iniciales en otras más simplificadas que no presenten dichos problemas. Este proceso es lo que se conoce como normalización y es un método formalizado con diferentes niveles, a cada uno de los cuales se le llama forma normal
Middleware	es un módulo intermedio que actúa como conductor entre dos módulos de software. Para compartir datos, los dos módulos de software no necesitan saber cómo comunicarse entre ellos, sino cómo comunicarse con el módulo de middleware

	El middleware debe ser capaz de traducir la información de una aplicación y pasársela a la otra. El concepto es muy parecido al de ORB (Object Request Broker), que permite la comunicación entre objetos y servicios de gestión básicos para aplicaciones de objetos distribuidos [A 1.07]
Data Mart	Es un conjunto de Bases de datos y herramientas destinadas a resolver un problema comercial específico. Aunque el tamaño no define a los Datamart tienden a ser más pequeños que los almacenes de datos.
Data WhereHouse	Es una Base de Datos diseñada para soportar la toma de decisiones en una empresa. Está orientada al proceso por lotes "Batch" y estructurada para realizar consultas rápidas on line y generar informes par los directores. Pueden contener cantidades enormes de datos.
Data Mining	Exploración de transacciones comerciales detalladas implica la búsqueda en enormes cantidades de datos para descubrir tendencias y relaciones dentro de la actividad y el historial de la empresa.

[WWW 01 031]

2. AMBIENTES INTEGRADOS DE ESTUDIO Y ANÁLISIS COMPARATIVO CON LOS TRADICIONALES

2.1 CARACTERÍSTICAS DE AMBIENTES INTEGRADOS DE ESTUDIO

Los Ambientes Integrados son tomados de diferentes perspectivas. Es importante para un programador decidir cuales conceptos emitir o cuales incluir en la programación. Con frecuencia el programador utiliza combinaciones de conceptos que hacen al lenguaje más fácil de usar, de entender e implementar. Cada programador tiene en mente un estilo particular de programación, la decisión de incluir u omitir ciertos tipos de datos que pueden tener una significativa influencia en la forma en que el Lenguaje es usado, depende del programador. A continuación se detallan las características más relevantes de los Ambientes Integrados considerados para el estudio.

2.1.1 BORLAND C++ BUILDER

REQUERIMIENTOS:

Para una correcta ejecución de sus aplicaciones, se recomienda un procesador Pentium II en adelante, por lo menos 64 MB en RAM aunque Borland recomienda 126 MB en RAM. Ya que el desarrollo en C++ Builder corre en sistemas operativos de 32 bits, bajo Windows 9x, ME, 2000 o Windows NT; la capacidad del disco duro depende de la versión de C++ Builder que esté usando.

CARACTERÍSTICAS DEL LENGUAJE:

Borland C++ Builder no es una herramienta visual de programación normal y corriente, es cien por ciento orientado a objetos, es la primera herramienta que ofrece la velocidad del desarrollo visual, la productividad de los componentes reutilizables y la potencia de C++. Este entorno hará las delicias de expertos programadores, pues facilita la gestión de las clases e incluye infinidad de librerías para cualquier tipo de desarrollo, permitiendo ahorrar tiempo en la fase de desarrollo de aplicaciones e-bussiness con Web Services. Además soporta las tecnologías emergentes tales como Web Services y Cross Plataform con ANSI/ISO y el poder y desempeño que los desarrolladores necesitan. La tecnología CLX cross-platform permite a empresas y desarrolladores construir aplicaciones en Windows y pasarlas a la plataforma Linux listas para ser ejecutadas, usando extensiones nativas o scripts CGI.

Borland C++ Builder mantiene mejor compatibilidad con ANSI/ISO C++ y entrega el código más rápido y estructurado. Simplifica radicalmente el desarrollo C++ de objetos distribuido con un total soporte integrado para los estándares COM y CORBA, con los cuales puede entregar poderosas aplicaciones de misión crítica.

C++Builder le entrega la productividad de las componentes reutilizables, más de 100 que encapsulan totalmente los más comunes controles de Windows , con una completa extensión que incluye un completo soporte

para las componentes ActiveX. Borland C++Builder otorga además el ilimitado poder del lenguaje C++, rapidez, confiabilidad, seguridad, optimización de la compilación, enlace incremental de objetos, monitoreo de la CPU y herramientas para la línea de comandos.

Los grandes proyectos, proyectos multinivel, servidores COM y clientes COM pueden ahora ser gestionados, editados y depurados dentro del IDE de C++Builder. La gestión de proyectos permite a los desarrolladores compilar varios objetivos como archivos EXE, LIB, RES, RC y DLL, así como crear sus propios objetivos.

C++Builder incluye implementaciones de componentes inmediatas de más de 17 protocolos estándares de Internet: Sockets, UDP, TCP, SMTP, POP3, NNTP, HTTP, FTP, HTML y Time. Además, la tecnología WebBroker de C++Builder ayuda a los desarrolladores a crear extensiones ISAPI y NSAPI para la difusión de datos a alta velocidad en la Web. Como WebBroker de C++Builder es neutral para las plataformas, puede funcionar y soportar totalmente estrategias de Internet de Microsoft y Netscape.

Ensamblador integrado para un mayor rendimiento, el nuevo enlazador incremental de alta velocidad de C++ Builder reduce el tiempo necesario a medida que las aplicaciones crecen, permitiendo al desarrollador escribir y depurar menos código entre compilaciones.

C++Builder asegura la interoperabilidad del lenguaje con Delphi, C++, Java, Visual Basic, PowerBuilder y Java Script implementando estándares de sistema Microsoft para COM. C++Builder utiliza también ActiveX en un entorno multinivel para proporcionar soluciones comerciales integradas y reutilizables.

[WWW 02 020]

[WWW 02 021]

Existen tres versiones del programa:

C++ Builder Standard. Es la versión para usuarios que necesiten crear aplicaciones para Windows sin necesidad de realizar complejos accesos a Bases de datos ni opciones muy avanzadas. Este paquete trae todo lo necesario para realizar cualquier tipo de aplicación.

C++ Builder Cliente/Servidor. Herramienta ideal para realizar aplicaciones de interconexión entre distintas máquinas. Este paquete es casi igual al profesional, menos en la parte de base de datos, en esa parte está mejor equipado.

C++ Builder Profesional. Esta es la más avanzada de todas, trae los mismos componentes que las anteriores y toda la parte de cliente SQL, ODBC.

Otra característica es la terminación final del producto, en la que juega un papel importante la completa documentación que aporta, y C++ Builder no brinda esta facilidad. [REV 02 010]

Ambiente De Desarrollo Integrado.- Cuando usted inicia C++ Builder, le presenta un grupo de ventanas dispersas al rededor de su ventana principal.

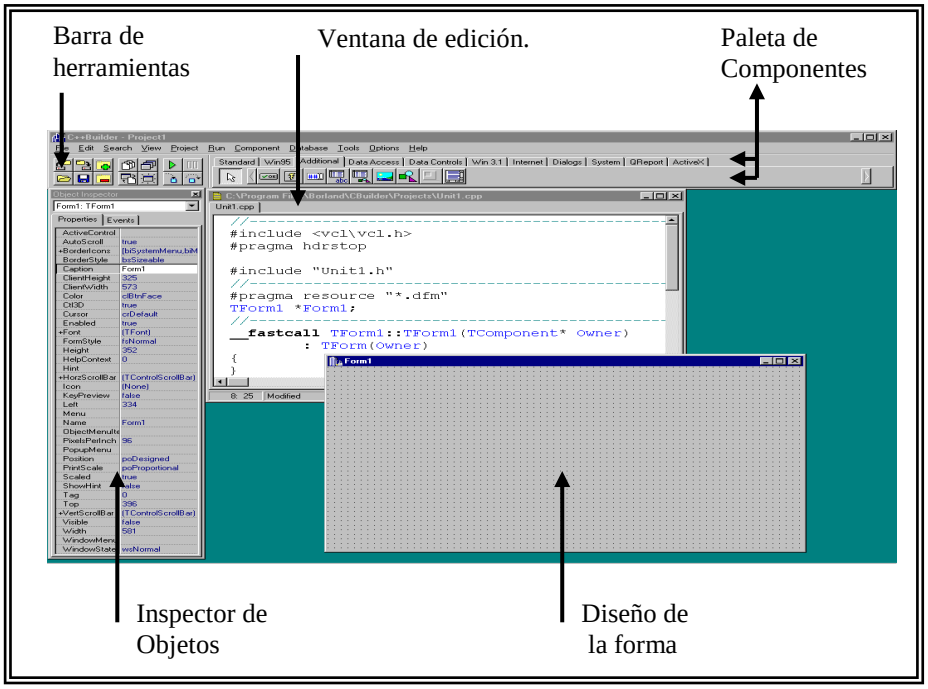


Fig. 2.01 Ambiente de desarrollo integrado C++ Builder.

En la figura se presenta los principales elementos de C++ Builder (Ambiente de Desarrollo Integrado, por sus siglas en Ingles IDE). Cada parte en el ambiente de desarrollo trabaja conjuntamente, diseños visuales y editor de código donde la edición es similar a otros editores; solo que con el ambiente de dasarrollo integrado, usted puede observar realmente lo que esta construyendo al momento de crearlo.

***Descripción De La Paleta De Componentes*.-** La paleta de componentes es algo como un catálogo de objetos que puedes usar de acuerdo a las necesidades de construcción de tus aplicaciones. Está dividida en páginas o grupos de acuerdo a sus funciones. Para implantar uno de estos componentes en tu aplicación, solo tienes que seleccionarlo con el mouse haciendo un clic en el objeto deseado y hacer clic en la forma principal (Forma de edición, ventana punteada) para que ya puedas utilizar ese objeto. C++ Builder soporta docenas de componentes.

Standard: Esta tabla contiene los objetos para hacer eficaces y elegantes tus aplicaciones Windows, incluye componentes para desplegar y editar texto, botones, barras de estado y menús.

Additional: La tabla de adicionales contiene algunos de los mejores y variados de la paleta de componentes, como mapas de bits, botones aceleradores y componentes de apariencia.

Data Access y Data Controls: Se pueden accesar bases de datos y hacer consultas dentro de las aplicaciones que construyas con las facilidades que permite estos 2 grupos de objetos.

Win32: Controles en Win9X y equivalentes en WinME, 2000 NT, Internet: Esta tabla dada por C++ Builder, comprende lo referente al grupo de herramientas de Internet.

ActiveX: Esta tabla de componentes, contiene un checador de ortografía así como objetos gráficos impresionantes.

En C++ BUILDER se trabaja con editores para los diferentes ambientes como tenemos el EDITOR DE FORMA, cada forma representa una ventana individual en tu aplicación; en la forma podemos diseñar, añadir, eliminar, reconfigurar los componentes según las necesidades de la aplicación.

El Editor De Objetos.- permite ver las propiedades o características de los objetos que comprendan tu proyecto, por medio de él se pueden cambiar estas propiedades también muestra los eventos asociados a los objetos de la aplicación.

Cuando se selecciona un objeto, el editor de objetos automáticamente cambia al contenido y propiedades de este objeto.

Propiedades: Cuando se comienza un proyecto el editor de objetos despliega las propiedades de la forma principal como son: nombre, color, altura, ancho, posición etc.

Recordemos que al seleccionar otro objeto, automáticamente mostrará las propiedades de ese objeto.

Eventos: La tabla de eventos despliega para cada objeto los eventos como son:

Al activar el objeto, al oprimir una tecla, al oprimir el mouse, al soltar el mouse, etc. esos eventos son disparados con acciones del usuario, o del sistema operativo mismo.

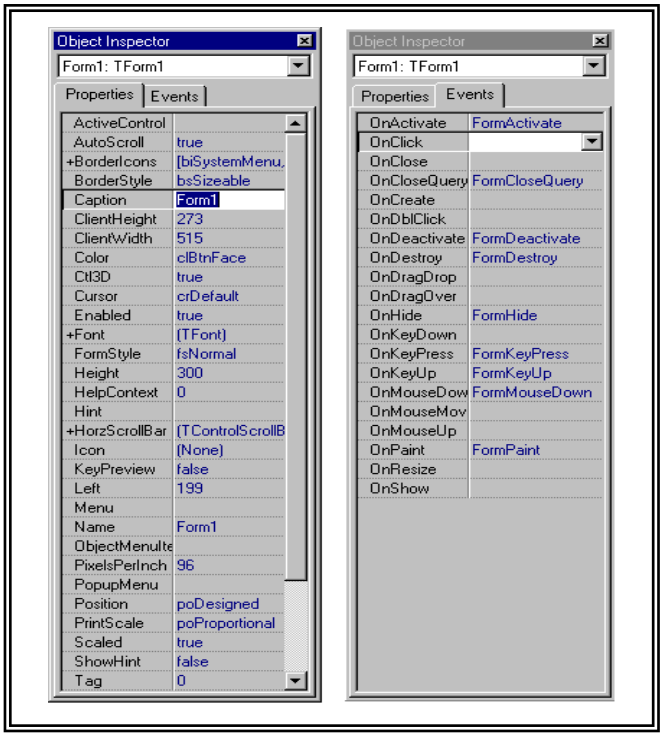


Fig. 2.02 Inspector de objetos.

***El Editor De Código*.-** es la ventana de edición de código muestra el código actual de tu aplicación C++ Builder. Al añadir objetos y hacer doble clic sobre ellos, automáticamente se editará en la ventana de edición la llamada a la función que asociará al evento de ese objeto, dejando el espacio en blanco para que se codifique la acción que se desee para ese evento.

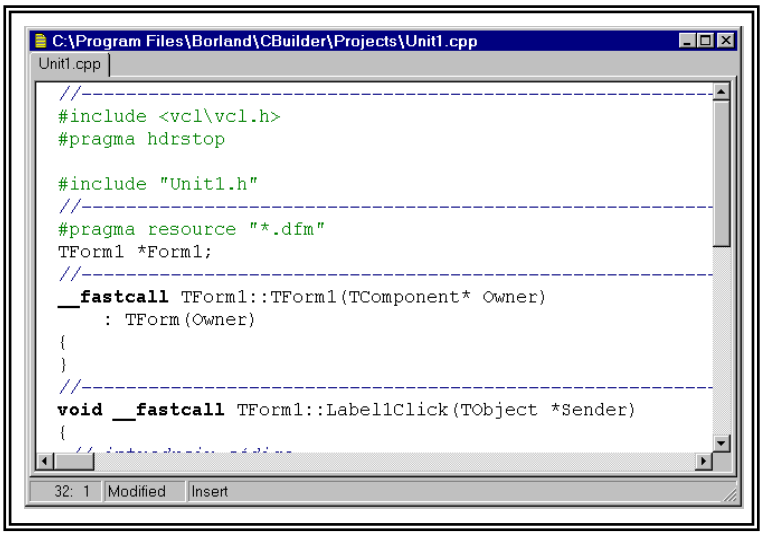


Fig. 2.03 Ventana de edición de código.

Un sencillo proyecto de C++ Builder está conformado por solo una forma y su código, pero en aplicaciones muy grandes, puede conformarse un proyecto por varias formas, código y varios archivos de cabecera distintos a las librerías que por omisión ya necesita la aplicación, para saber cuales son los archivos que comprende un proyecto, usaremos el MANEJADOR DE PROYECTOS, que muestra el árbol de archivos involucrados en el orden en que fueron añadidos. [WWW 02 022]

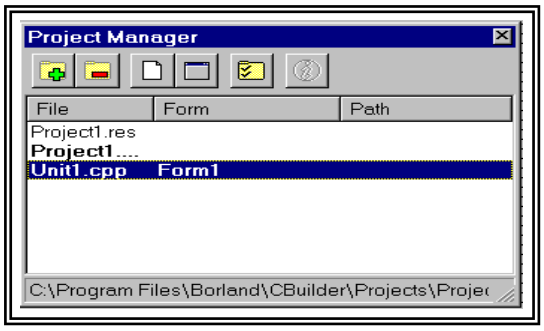


Fig. 2.04 Manejador de Proyectos

Archivos Fuente Generados Por C++ Builder.- Los archivos que genera C++ Builder son los siguientes:.

Project1.mak: Este es el archivo principal de opciones del proyecto. Un archivo .mak se requiere en cada aplicación; es un archivo de texto que contiene instrucciones de cómo C++ Builder construirá el archivo (.exe) ejecutable para el proyecto.

Project1.Cpp: Este archivo contiene el código principal de la aplicación, comparte el mismo nombre del proyecto, lo crea automáticamente C++ Builder al darle nombre al proyecto.

Unit1.cpp: Este es el código que usted le da a los eventos de los objetos que tendrá en su forma final, este código es el que introduce en la ventana de edición, o en su editor de código.

Unit1.h: Para cada archivo .cpp, C++ Builder crea automáticamente un .h correspondiente. El archivo de cabecera contiene la declaración de la forma y menciona a C++ Builder la lista de componentes y los eventos que tendrá la aplicación.

Unit1.dfm: El archivo .dfm contiene la información, definición y declaración de la forma y otros detalles importantes como: tamaño, color, títulos, fondos etc. así como detalles del resto de los componentes utilizados en la forma.

La extensión .dfm indica que este archivo oculta los datos de la forma en formato binario. Este archivo no se puede leer, pero puede convertirlo para observar su contenido.

Unit1.obj Cuando se compila el proyecto se crea el archivo binario con extensión .obj. Cada aplicación contendrá un archivo .obj, este archivo cambiará cada vez que se reconstruya el proyecto.

Project1.exe o Project1.dll: Este es el archivo final según se halla elegido en su proyecto, el .exe podrá ejecutarse, y los dll son librerías dinámicas que pueden utilizarse desde otros programas.

Project1.dsk: Contiene la información de la configuración que tenía su hardware al momento de crear o finalizar su proyecto, para al momento de ejecutarse lo haga sobre la misma configuración.

Project1.il?: Al ver la extensión .il? indica que el archivo es usado al enlazar, C++ Builder usa una tecnología de compilación muy rápida, lo que hace que la compilación o recompilación de sus proyectos sea en cuestión de segundos.

VARIABLES Y TIPOS DE DATOS

Tipo	Tamaño/Rango	Descripción
byte	8-bit 0-255	Entero de un byte
int	4-bit –2147483648 – 2147483648	Entero
long	64-bit -2^{63} a $2^{63}-1$	Entero largo
float	32-bit Reales de precisión simple	Valor real
Double	64-bit Reales de precisión doble	Valor real
Boolean	8-bit true o false	Valor Booleano
Char	8-bit 0-255	Caracteres

Tabla 2.01 Variables y Tipos de datos en C++ Builder.

C++ Builder tiene su base de programación en el conocido lenguaje C++, por lo que utiliza las mismas variables y tipos de datos. Lo que en si se resalta de C++builder es el desarrollo de proyectos dentro de su ambiente integrado. [LIB 02 001]

ESTRUCTURAS DE CONTROL

Soporta las estructuras condicionales básicas if, if-else, y switch y los bucles for-while y do-while y los bucles for, while y do-while. Sin embargo existen otras instrucciones de control exclusivas de C, como el operador condicional ? y las instrucciones break y continue. [WWW 02 022]

Ejemplo:

```
Object Form ;  TForm1
    Left = 200
    Top =111
    Width =775
    Height = 606
    Caption =”MI PRIMER PROGRAMA”
    Color = clTeal
    Font.Name = Monotype Coraiva
    Font.Style = (FsBold, FsItalic)
    FixelsPerInch = 120
   TextHeight = 33
```

Programa que imprime un texto en la pantalla.

2.1.2 VISUAL J++

Microsoft Visual J++ creado para aprovechar la productividad del lenguaje java y la eficacia de Windows para generar aplicaciones, nos permitirá escribir, compilar, depurar y ejecutar aplicaciones y *applets Java*, todo ello desde un único entorno de desarrollo. El entorno consiste en una serie de herramientas incluyendo herramientas de compilación y depuración, asistentes para la creación automática de aplicaciones, un editor de texto que resalta la sintaxis de *Java* y editores de recursos.

REQUERIMIENTOS:

- Sistema operativo Microsoft Window95 o posterior, con Service Pack 3.0 o posterior.
- Memoria 48 MB en RAM como mínimo.
- 350 MB de espacio en disco [WWW 02 023]

CARACTERISTICAS DEL LENGUAJE:

Las principales características de Microsoft Visual J++ son:

Soporte para trabajo con Base de datos*:* Visual J++ añade la posibilidad de generar parte del código visualmente, cuenta con el Database Wizard, este asistente facilita la creación de un applet que acceda a una base de datos, bien sea a través de DAO (Access) o RDO (Remote Data Objects, ODBC). El asistente nos conduce por una serie de pasos para especificar la fuente de datos a la que deseamos el applet acceda, las tablas que deseamos

manejar y los campos que queremos que se presenten. El resultado del proceso es un applet con botones de navegación y campos de edición de texto que acceden a la base de datos y nos presentan los resultados en el documento Web en el que ubiquemos el applet.

Mejor depuración: una característica muy relevante es que resulta posible depurar applets mientras se están ejecutando bajo Internet Explorer. También es posible utilizar Netscape para esto, dado que Microsoft proporciona un *plug-in* para soporte de depuración bajo dicho navegador. La posiblidad de depurar applets mientras se ejecutan en un navegador resulta muy útil.

ActiveX Wizard*:* Microsoft ofrece un completo soporte de ActiveX dentro de su entorno de desarrollo. Existe la posibilidad de utilizar y/o crear objetos ActiveX con Visual J++ , mediante dos nuevos wizards que automatizan la tarea. es decir, permite crear un control *ActiveX* con la misma funcionalidad que una determinada clase *Java.*

Editores de Recursos: Estos editores permiten crear y modificar menús, diálogos, *string tables, accelerator tables, version resources*, y objetos gráficos, como iconos, cursores, y bitmaps. Cuando se crea o se abre un recurso, el editor apropiado se abre automáticamente. Estos recursos, podrán ser después traducidos a código *Java* mediante *Java Resource Wizard.*

Paneles de vistas jerárquicos*: Microsoft Developer Studio* organiza la información relativa a una determinada aplicación en árboles jerárquicos que permiten apreciar las relaciones entre los diversos componentes incluidos en un proyecto. Estos paneles jerárquicos permiten ver las clases y los ficheros que incluye el proyecto. El primero de ellos es *ClassView* y nos permite explorar la estructura jerarquizada de nuestro proyecto desde el punto de vista de las clases.

FileView es el nombre del segundo panel, que presenta las relaciones entre los ficheros incluidos en el proyecto, permitiendo acceder directamente al código fuente

Un tercer panel, *Data View*, nos permite examinar nuestro proyecto jerarquizado desde el punto de vista de las bases de datos asociadas al mismo, siempre que estén asociadas a fuentes de datos *ODBC*.

Ambiente Visual: Los entornos de desarrollo RAD ofrecen la posibilidad de generar gran parte del código visualmente, sobre todo en lo que respecta al interfase de usuario. Lo más corriente es trabajar con un modelo en que los componentes visuales tienen asociadas propiedades (por ejemplo, el color del texto) y eventos, a los cuáles se puede asociar código específico de modo que este se ejecuta cuando ocurra un evento. Un ejemplo podría ser un evento *OnClick* que se produce cuando el usuario pulsa sobre un

botón con el ratón: podríamos asignar código para guardar cierta información, cerrar una ventana, etc. [WWW 02 024]

Programacion Orientada A Objetos: Visual J++ es un lenguaje orientado a objetos, esta programación es una filosofía que se ha ido imponiendo en los últimos años frente a las técnicas tradicionales de programación estructurada, sobretodo en ámbitos que responden a modelos de programación controlada por sucesos. En POO el programa está formado por una serie de componentes independientes y cerrados o mejor, autocontenidos, que cooperan para realizar las acciones de la aplicación completa. La POO se fundamenta en los conceptos de objeto y clase. Una clase es una *plantilla* que abstrae las características de una cierta entidad. Un objeto es cada representación concreta de la abstracción de la clase.

VARIABLES Y TIPOS DE DATOS:

En Visual Java existen tres tipos de variables: de objeto, de clase y locales. Las primeras definen propiedades de objeto y las segundas comunes a todos los de una clase. Lo más reseñable es que en *Visual J++* no existen variables globales.

Las declaraciones de variables en Visual J++, como las de C++, consisten simplemente en un tipo y el nombre de una variable:

```
int Numero;
String Nombre;
```

Cada declaración de variable debe poseer un tipo de datos que defina cuales son los valores que se encuentran en el dominio de la misma. Permite 8 tipos básicos predefinidos, clases, y *arrays.*

Tipos básicos predefinidos:

Tipo	Tamaño/Rango	Descripción
Byte	8-bit 0-255	Entero de un byte
Short	16-bit -32768 – 32767	Entero corto
Int	4-bit -2147483648 – 2147483648	Entero
Long	64-bit -2^{63} a $2^{63}-1$	Entero largo
Flota	32-bit Reales de precisión simple	Valor real
Double	64-bit Reales de precisión doble	Valor real
Bolean	8-bit true o false	Valor Booleano
Char	16-bit Caracteres ANSI o Unicote	Caracteres

Tabla 2.02 Tipos de datos Visual J++.

Comentarios: Los comentarios en Visual J++ se los presenta de la siguiente manera:

```
//Comentarios de una línea
/*Comentarios de una o mas líneas*/
```

OPERADORES:

Los operadores que se utilizan en Visual J++ son:

Categoría	Operador	Operación
Asignación	=	Asignación
Aritméticos	+	Suma
	-	Resta
	*	Multiplicación
	/	División
	%	Módulo
Booleanos	*!*a	Negación (NOT)
	a & b ó a && b	Producto lógico (AND)
	a | b ó a | | b	Suma lógica (OR)
	A ^ b	Suma exclusiva (XOR)
Bit	*&*	Producto a nivel de bit
	|	Suma a nivel de bit
	^	*or* exclusiva a nivel de bit
	~	Complemento
	<<	Desplazamiento izquierda
	>>	Desplazamiento derecha
	>>>	Desplazamiento a la derecha con relleno con ceros.
	&=	x = x & y
	| =	x = x | y
	^=	x = x ^ y
Comparación	==	Igualdad
	!=	Desigualdad
	<	Menor que
	>	Mayor que
	<=	Menor o igual que

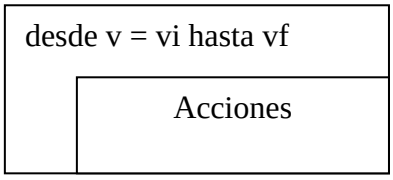
	>=	Mayor o igual que

Tabla 2.03 Operadores Visual J++.

ESTRUCTURAS DE CONTROL:

If...Else*:* Es la más fundamental de las sentencias de ejecución condicional. Permite condicionar la ejecución de una porción de código al cumplimiento de una determinada condición. La porción de código denominada *código1* se ejecutará si la *condición* se cumple, en caso contrario y si la cláusula *else* está presente, se ejecutará *código2:*

```
If condición
    código1;
[else // opcional
código2; ]
```



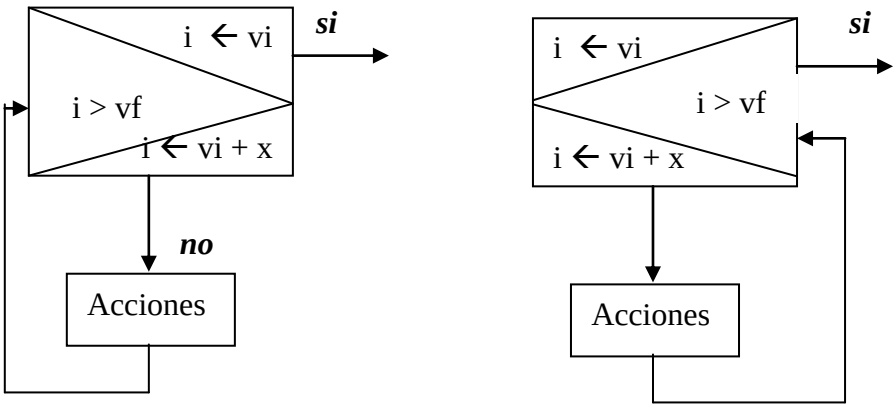
Switch: Permite al programador comparar una variable con un conjunto predefinido de valores y ejecutar una porción u otra de código en función del contenido de dicha variable. Esta misma función puede realizarse mediante una serie de *if/else* anidados:

```
switch (variable test) {
    case valor1:
Código1;
break;
```

```
…
case valorn:
Códigon;
break;
        default:
Código por defecto;
```

for: Esta sentencia permite repetir la ejecución de un bloque de código un número definido de veces. Al comenzar su ejecución se inicializa *variable* para que adopte el *valor inicial*. Al final de la ejecución de *código* se incrementa el valor de la variable y se comprueba si se cumple una determinada *condición;* en caso de que no se cumpla la ejecución de la sentencia finaliza.

for ***(****variable = valor inicial; condición; incremento){ código }*



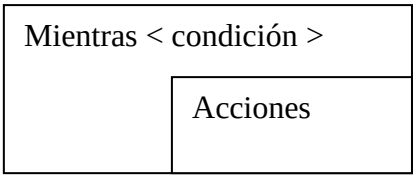
do…while: Permite repetir la ejecución de una porción de código mientras se satisfaga una cierta condición. En esta sentencia *código* se ejecuta como mínimo una vez.

```
do {
```

```
    código;
} while (condición)
```

while: Es muy similar a la anterior con la diferencia de que la verificación del cumplimiento de la condición se lleva a cabo antes de ejecutarse.

```
while (condición) {
        código;
}
```



[WWW 02 025]

2.1.3 PERL PARA LINUX

REQUERIMIENTOS:

Existen versiones de Perl para casi todos los Sistemas Operativos, la última versión la 5.0 puede ejecutarse desde cualquier plataforma. No limita el espacio de disco o memoria pero si se tiene la suficiente memoria, Perl puede manejar el archivo entero como un solo string y las tablas de hash usadas por los arreglos asociativos crecen lo suficiente para prevenir que se degrade el desempeño.

CARACTERÍSTICAS DEL LENGUAJE:

Perl (Practical Extraction y Report Language) es un lenguaje de programación desarrollado por Larry Wall a inicios de los noventa, creado originalmente para facilitar la elaboración de tareas comunes en sistemas tipo UNIX, donde tradicionalmente las tareas de administración y proceso de datos se realiza con herramientas muy rudimentarias y por demás hostiles al usuario o administrador.

Perl surgió como una opción para una gran cantidad de herramientas de UNIX, como son: ed, grep, awk, c-shell, sed, sh en las cuales basa su propia sintaxis, buscando el mínimo sacrificio de su desempeño por una máxima facilidad de programación e integración, sigue la filosofía de mantener un ambiente que sea capaz de detectar y corregir pequeñas omisiones del programador, y de proporcionarle una forma abreviada de realizar múltiples tareas. En una palabra, es una utilería que pretende

facilitar el proceso de grandes volúmenes de información sin sacrificar el rendimiento.

A diferencia de la mayoría de las utilidades de UNIX, Perl no limita arbitrariamente el tamaño de los datos. Perl utiliza técnicas sofisticadas para encontrar patrones lo que le permite escanear grandes cantidades de datos rápidamente. Aunque el Perl está optimizado para escanear texto, puede también trabajar con datos binarios, y puede hacer que archivos dbm se vean como arreglos asociativos.

Con el crecimiento del WWW se vio que era necesario realizar programas CGI y Perl se convirtió en la elección natural para los que ya estaban familiarizados con este lenguaje. El aumento de sitios Web ha transformado el papel de Perl de un lenguaje de Script oscuro y desconocido a la herramienta principal de programación CGI.

Perl no establece ninguna filosofía de programación (de hecho, no se puede decir que sea orientado a objetos, modular o estructurado aun cuando soporta directamente todos estos paradigmas), los objetivos que se tuvieron en cuenta al diseñar la sintaxis de Perl fueron la facilidad de aprendizaje y de uso y la claridad de código.

Perl no es ni un compilador ni un intérprete, esta en un punto intermedio, cuando mandamos a ejecutar un programa en Perl, se compila el código

fuente a un código intermedio en memoria, se le optimiza (como si fuésemos a elaborar un programa ejecutable) pero es ejecutado por un motor, como si se tratase de un interprete. El resultado final, es que utilizamos algo que se comporta como un intérprete pero que tiene un rendimiento comparativo al de programas compilados. [WWW 02 026]

Perl no nos forza a nada en cuanto a programación , pero como es lógico hay ciertas reglas para facilitar el trabajo en este lenguaje:

- **Claridad.** En la mecánica de programación actual, los programas deben de ser entendibles por la persona que nos suceda en tareas de mantenimiento.
- **Indentación.** Una costumbre ya clásica de la programación, es la de "indentar" el código con dos espacios hacia adelante al abrir cada bloque, y se termina la indentación al terminar el bloque, de modo que las llaves de apertura y cierre quedan a la vista y en la misma columna.
- **Nombres de variables y demás**. Procurar dar la máxima claridad a los nombres de las variables sin hacerlos demasiado grandes, y los nombres de archivos con mayúsculas (ARCHENT, ARCHSAL, etc) y las clases su primera letra con mayúscula.
- **Comentarios**. Para facilitar la comprensión de un este debe ser detallado, y los comentarios son el medio ideal para hacerlo.
- **Sencillez**. Es cómodo en ocasiones el comprimir una serie de instrucciones en una sola línea, queda al criterio decidir cuando se

gana en claridad con un código mas o menos extenso, pero no debe titubearse en comentar el código.

Las plataformas donde Perl se ha desarrollado mas son los servidores UNIX, por sus necesidades de administración y lo robusto de su manejo de memoria y de procesos (requisitos de PERL hacia el S.O.) además de la facilidad de Perl para realizar los así llamados CGIs, interfaces para comunicar recursos del servidor con un servicio de Internet particular (como podría ser WWW o gopher), En otras plataformas, PC en particular, se han desarrollado versiones que mantienen un razonable grado de funcionalidad, pero en realidad, el sistema DOS no tiene un manejo lo bastante bueno de los procesos o de la memoria para permitir a PERL dar un buen desempeño, además de que no es común ver en PC necesidades de administración de la magnitud de un servidor institucional.

En resumen, Perl es un lenguaje muy utilizado en los dos campos siguientes:

1. ***La administración de sistemas operativos*.** Debido a sus características Perl es muy potente en la creación de pequeños programas que pueden ser usados como filtros para obtener información de ficheros, realizar búsquedas, etc.
2. **La creación de formularios en la Web.** Es decir, se utilizan para la creación de scripts CGI (Common Gateway Interface). Estos scripts realizan el intercambio de información entre aplicaciones

externas y servicios de información, es decir, se encargan de tratar y hacer llegar la información que el cliente WWW manda al servidor a través de un formulario.

Instalación.- Dependiendo del sistema operativo que se utilice, habrá que utilizar una distribución de Perl u otra. La principal referencia figura en:

http://www.perl.com/.

No obstante, en http://www.cpan.org/ podemos encontrar más distribuciones, disponiendo de al menos una para cada plataforma.

Es conveniente utilizar como directorio base de la instalación:

C:\Perl, y añadir al PATH la ruta C:\PERL\BIN.

VARIABLES Y TIPOS DE DATOS:

En cuanto a las listas de caracteres que representan a nuestras variables decir que los caracteres permitidos son las letras, dígitos y el carácter underscore (_). Las letras mayúsculas y minúsculas son diferenciadas en los nombres de variables. Los nombre de las variables siempre deben comenzar por una letra. Se recomienda que los nombre de las variables estén en consonancia con lo que ellas representan, sin embargo estos tampoco deben ser demasiado largos.

Perl permite representar los tipos de datos básicos como son los reales, los enteros, las cadenas de caracteres y el tipo booleano.

Los tipos numéricos (reales y enteros).

Los valores numéricos expresados literalmente se presentan en forma de valores reales codificados en doble precisión. Este formato interno se utiliza para todas las operaciones aritméticas. Por ejemplo:

```
$x = 0.897; # un real
$y = 6.23e-24; # un real
$n = 567; # un entero
$i = -234; # un entero
```

Nota: El que todas las variables contengan un $ significan que representan un escalar.

Los valores enteros no pueden empezar por cero porque esto permite especificar un entero mediante su codificación octal o hexadecimal. El código octal se antecede cero 0; el código hexadecimal se antecede con un 0x o 0X. Por ejemplo:

```
$x = 0377; # equivale a 255
$y = 0xff; # equivale a 255
```

Las cadenas de caracteres.- se especifican literalmente por medio de un sucesión de caracteres delimitada por comillas ("..") o apóstrofes ('..'). Estas dos representaciones se distinguen por la interpretación hecha por Perl de las cadenas de caracteres. Cuando van delimitadas por comillas ("..."), toda variable referenciada en el interior de la cadena se evalúa y se reemplaza por su valor. Por ejemplo:

```
$wld = "mundo";
$str = "¡Hola $wld!";
```

El tipo booleano.- El tipo booleano existe, al igual que en C, de modo implícito, es decir, un número es falso si es igual a cero y verdadero en cualquier otro caso. Como el cero está asociado a la ristra vacía (""), ésta también equivale al valor falso. [WWW 02 027]

OPERADORES:

En Perl distinguiremos tres tipos de operadores dependiendo de la representación de datos sobre la que queremos actuar. De este manera habrá tres tipos de operadores: los operadores asociados a los escalares, los asociados a los arrays y por último, los vinculados a las listas asociativas.

Los operadores asociados a escalares.- Los operadores definidos en Perl tienen todos los operadores estándar (de tipo C) a los que añaden operadores específicos para la manipulación de cadenas de caracteres.

Operadores Aritméticos.- Perl suministra los siguientes operadores aritméticos.

OPERADOR	OPERACION
+	Suma
-	Resta
*	Multiplicación
**	Exponenciación
/	División
%	Resto

.	Concatenacion de dos cadenas
x	Repetición de dos caracteres

Tabla 2.04 Operadores Aritméticos Perl.

Entre los operadores artiméticos también distinguimos los de autoincremento (++) y autodecremento (--). Estos operadores son unarios y se realiza el incremento o decremento de la variable que se le aplica. Además de la acción de modificar la variable devuelven el valor de la variable.

```
$n = $k++;
# el valor de k se asigna a n y después se incrementa k
```

Operadores relacionales.- Perl distingue dos tipos de operadores relacionales: los operadores específicos a valores numéricos y los propios de las cadenas de caracteres. Estos operadores se resumen en la siguiente tabla:

Operador Relacional	***Numérico***	***Cadena De Caracteres***
Igualdad	=	*eq*
Diferencia	!=	*ne*
Inferior	<	*lt*
Superior	>	*gt*
Inferior o igual	<=	*le*
Superior o igual	=>	*ge*

Tabla 2.05 Operadores Relacionales Perl.

A parte de los operadores que hay en la tabla cabe distinguir otros operadores únicamente característicos del lenguaje Perl.

- cmp. Este operador es utilizado para comparar caracteres, de manera que, retorna 0 si los caracteres comparados son iguales, 1 si la cadena de la derecha se encuentra al comienzo de la de la izquierda, y -1 en el caso contrario. Para aclarar el funcionamiento de este operador he aquí un ejemplo:

```
'one' cmp 'one' # devuelve
0
'one dog ' cmp 'one' #
devuelve 1
'dog one' cmp 'one' #
devuelve -1
```

- <=>. Este operador se utiliza para comparar valores numéricos, retornando 0 cuando son iguales, 1 cuando el termino de la derecha es menor que el de la izquierda y -1 en el caso contrario.
- =~. Este operador es usado en las expresiones regulares para indicar la presencia de un patrón de comparación dentro de una variable que contiene una cadena de caracteres. Por ejemplo:

```
if ($ristra =~ /str/) {
print $ristra;
}
else {
print "No se encuentra el
patrón";
}
# verifica si 'str' se está en
$ristra
```

- Meditante este operador se verifica la no existencia del patrón de búsqueda en una cadena. He aquí un ejemplo:

```
if ($ristra !~ /str/) {
print "No se encuentra el patrón";
}
else {
print $ristra;
}
```

Operadores lógicos.- Los operadores lógicos están relacionados con los relacionales ya que normalmente los operadores que se usan son resultado de expresiones relacionales.

Operadores lógicos	
%%	and
||	or
!	not

Tabla 2.06 Operadores Lógicos Perl.

Operador de selección.- Es un operador triario que requiere una condición y dos expresiones. Se utiliza para ejecutar una expresión u otra dependiendo de la condición. Su formato es el siguiente:

Condición? Exp1: Exp2

Si se cumple la condición se evalúa y devuelve la expresión Exp1 si no la Exp2.

Operadores de asignación.- Una asignación también es un operador que devuelve la variable modificada. El la siguiente tabla veremos los operadores de asignación contenidos en Perl que como se podrá observar son muy parecidos a los del lenguaje C.

A parte de estos operadores Perl posee el operador =~ que también es un operador de asignación, ya que este operador se utiliza dentro de una expresión regular de sustitución para sustituir un patrón de comparación por otra cadena. *Ejemplo:*

Operadores a nivel de bits	
&	And bit a bit entre dos operandos.
|	Or bit a bit entre dos operandos.
^	Or exclusivo bit a bit entre los dos operandos.
~	Complemento a uno de bits. Invierte todos los bits del operando.
<<	Desplazamiento a la izquierda del primer operando tantas veces como indique el segundo.
>>	Dezplazamiento a la derecha del primer operando tantas veces como indique el segundo.

Tabla 2.07 Operadores de Asignación .

Operadores a nivel de bits.- Al igual que C, Perl toma como uno de sus objetivos no alejarse de la máquina, para ello posee esta serie de operadores a nivel de bits.

El operador ",".- El operador "," evalúa varias expresiones donde la sintaxis solamente permite una, siendo el valor resultante la última expresión evaluada. La evaluación se realiza de izquierda a derecha. En

Perl 5 también se usa el operador => para lo mismo que la coma ",". Ejemplo:

```
i = (f(3), j+5, k*8, 4/2); # i = 4/2
```

Símbolos de puntuación.- Además en Perl tenemos los símbolos de puntuación que a diferencia de otros lenguajes, son considerados operadores en Perl. Básicamente tenemos cuatro símbolos que nos permiten agrupar otros símbolos para darles una interpretación especial.

- ' '. Especifica valores literales. No hace sustituciones. Ejemplo:

```
$fecha = "14 de Julio";
print 'hoy es $fecha'; # imprime "hoy es $fecha"
```

- " ". Especifica valores literales. Realiza sustituciones. Ejemplo:

```
$fecha = "14 de Julio";
print "hoy es $fecha";
# imprime "hoy es 14 de Julio"
```

- ` `. Ejecuta el comando contenido haciendo las sustituciones indicadas. Igual que muchos Shells de Unix. Por Ejemplo:

```
print `date`; # ejecuta comando date
```

- / /. Delimita expresiones regulares. Ejemplo:

```
($var =~/exe/) {
  print $var;
  }
  # Se imprime $var si contiene el patrón exe
```

Precedencia de operadores.- Normalmente las expresiones en Perl se evalúan de izquierda a derecha, aunque hay algunas excepciones que es necesario tener en cuenta. No se evalúan de izquierda a derecha los operadores de asignación y el operador de selección. Cuando una expresión tiene diversas posibilidades se aplica el orden de precedencia para establecer el cálculo a realizar. La precedencia de los operadoes se mostrará en la siguiente tabla, aunque se recomienda emplear paréntesis para que nunca haya duda en el orden de evaluación.

Negación/in(de)cremento	!	++	--		
Mult/Exp/Div/Resto	*	**	/	%	
Suma/Resta	+	-			
Propagación	<<	>>			
Relacionales	>	<	<=	>=	
Igualdad	==	!=			
And bit a bit	&				
Xor bit a bit	^				
Or bit a bit	|				
And lógico	&&				
Or lógico	||				
Operador de selección	?:				
Asignación	+=	*=	%=	<<=	...

Tabla 2.08 Precedencia de Operadores .

[WWW 02 028]

Ejemplo: El típico programa “Hola, mundo” en Perl se realiza poniendo en un fichero (supongamos “hola.pl”) las siguientes instrucciones:

```
c:/perl/perl
print “Hola Mundo\n
```

No hay necesidad de abrir el programa , ni de cerrarlo, ni de incluir librerías estándar o no. Solamente para poder programar en PERL, hay que tener en cuenta:

- Perl es un lenguaje case-sensitive.
- Para editar el código fuente necesitamos simplemente un editor de texto. El Notepad puede valer.
- Se ejecuta desde la línea de comandos de una ventana del sistema operativo.
- Los comentarios comienzan con el carácter **#.**
- Las instrucciones terminan en punto y coma.
- La función **print** sirve para mostrar información por pantalla, y admite formatos muy diversos aunque sencillos de comprender.
- En Perl hay mucha flexibilidad para escribir los argumentos:

  print("Un texto", "Otro texto"); # con paréntesis

  print "Un texto", "Otro texto"; # sin paréntesis

- Perl ofrece una ayuda en línea desde la consola de comandos.
- Por ejemplo, para obtener ayuda sobre la función print, escribiremos en una ventana MSDOS:

  perldoc –f print

Para ejecutar basta con escribir, desde una ventana de MS-DOS:

perl hola.pl

[WWW 02 029]

2.1.3 JAVA PARA LINUX

Java es un lenguaje de programación desarrollado originalmente por Sun Mycrosystems, nació como un lenguaje para la red y para darle la seguridad que el HTML no tiene; sin embargo, Java es un lenguaje de propósito general y, en contra de una creencia bastante extendida, su uso no se limita al *WWW*. De hecho, existen aplicaciones completas en *Java*, y proyectos para desarrollar versiones de suites de aplicaciones en este lenguaje, sirve para crear todo tipo de aplicaciones empresariales, Intranets e Internet. [WWW 02 030]

REQUERIMIENTOS:

- Cualquier sistema operativo pero de preferencia un sistema operativo Multithreading.
- Memoria mínimo 32 MB en RAM
- 20 MB en Disco Duro para la instalación (JDK ver. 1.1.1)

CARACTERÍSTICAS DEL LENGUAJE:

Las características principales que nos ofrece JAVA respecto a cualquier otro lenguaje de programación, son:

Simple.- Java ofrece toda la funcionalidad de un lenguaje potente, pero sin las características menos usadas y más confusas de éstos. C++ es un lenguaje que adolece de falta de seguridad, pero C y C++ son lenguajes

más difundidos, por ello Java se diseñó para ser parecido a C++ y así facilitar un rápido y fácil aprendizaje.

Gestiona la Memoria Automáticamente.- Java elimina muchas de las características de otros lenguajes como C++, para mantener reducidas las especificaciones del lenguaje y añadir características muy útiles como el garbage collector (reciclador de memoria dinámica). No es necesario preocuparse de liberar memoria, el reciclador se encarga de ello y como es un thread de baja prioridad, cuando entra en acción, permite liberar bloques de memoria muy grandes, lo que reduce la fragmentación de la memoria. Java reduce en un 50% los errores más comunes de programación.

Orientado a objetos.- Java trabaja con sus datos como objetos y con interfaces a esos objetos. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo. Las plantillas de objetos son llamadas, como en C++, clases y sus copias, instancias. Estas instancias, como en C++, necesitan ser construidas y destruidas en espacios de memoria.

Cliente/servidor.- Java puede resultar algo misterioso al principio, sobre todo porque utiliza un vocabulario propio, pero, en esencia está basado en la aplicación al Internet que no es mas que un sistema Cliente-Servidor gigante. La idea primaria de un sistema cliente-servidor es que debe haber

un sitio donde se centraliza la información que se desea distribuir bajo demanda a un conjunto de personas o máquinas.

La clave de este concepto radica en que si se produce un cambio en la información del sistema central, inmediatamente es propagada a los receptores de la información, a la parte cliente; el problema se presenta cuando hay solamente un servidor que tiene colgados a muchos clientes, en donde el rendimiento general del sistema decrece de forma exponencial al aumento del número de clientes.

Distribuido.- Java se ha construido con extensas capacidades de interconexión TCP/IP. Existen librerías de rutinas para acceder e interactuar con protocolos como http y ftp. Esto permite a los programadores acceder a la información a través de la red con tanta facilidad como a los ficheros locales.

Robusto.- Java realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores, lo antes posible, en el ciclo de desarrollo. Java obliga a la declaración explícita de métodos, reduciendo así las posibilidades de error.

También implementa los arrays auténticos, en vez de listas enlazadas de punteros, con comprobación de límites, para evitar la posibilidad de

sobreescribir o corromper memoria resultado de punteros que señalan a zonas equivocadas. Estas características reducen drásticamente el tiempo de desarrollo de aplicaciones en Java.

Arquitectura neutral.- Para establecer Java como parte integral de la red, el compilador Java compila su código a un fichero objeto de formato independiente de la arquitectura de la máquina en que se ejecutará. Cualquier máquina que tenga el sistema de ejecución (run-time) puede ejecutar ese código objeto, sin importar en modo alguno la máquina en que ha sido generado. Actualmente existen sistemas run-time para Solaris 2.x, SunOs 4.1.x, Windows 95, Windows NT, Linux, Irix, Aix, Mac, Apple y probablemente haya grupos de desarrollo trabajando en el porting a otras plataformas.

Java para conseguir ser un lenguaje independiente del sistema operativo y del procesador que incorpore la máquina utilizada, es tanto interpretado como compilado. [WWW 02 031]

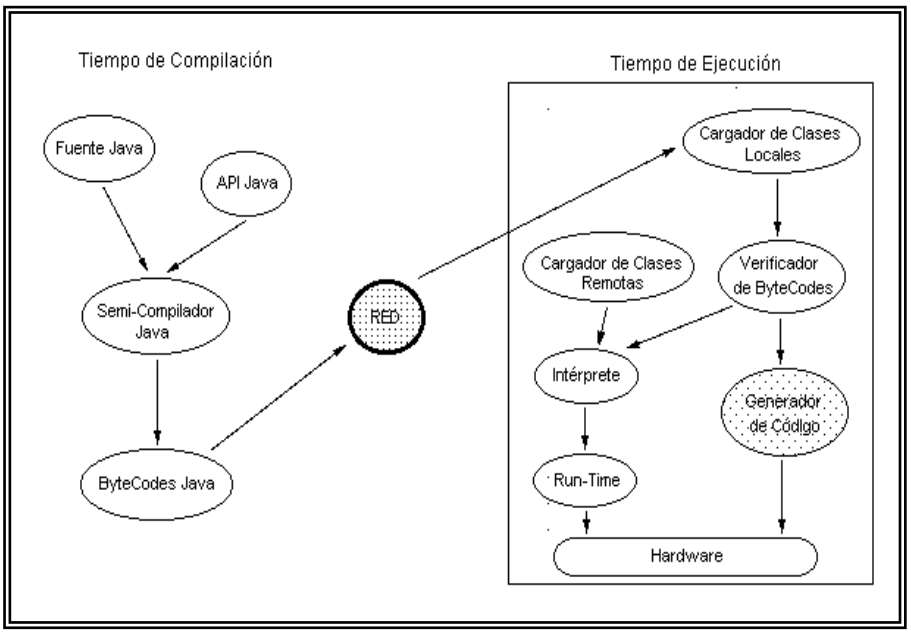


Fig. 2.05 Compilación Código Fuente Java

El código fuente Java se "compila" a un código de bytes de alto nivel independiente de la máquina. Este código (byte-codes) está diseñado para ejecutarse en una máquina hipotética que es implementada por un sistema run-time, que sí es dependiente de la máquina.

Seguro.- La seguridad en Java tiene dos facetas. En el lenguaje, características como los punteros o el casting implícito que hacen los compiladores de C y C++ se eliminan para prevenir el acceso ilegal a la memoria. Cuando se usa Java para crear un navegador, se combinan las características del lenguaje con protecciones de sentido común aplicadas al propio navegador.

Otra laguna de seguridad u otro tipo de ataque, es el Caballo de Troya. Se presenta un programa como una utilidad, resultando tener una

funcionalidad destructiva. Por ejemplo, en UNIX se visualiza el contenido de un directorio con el comando ls. Si un programador deja un comando destructivo bajo esta referencia, se puede correr el riesgo de ejecutar código malicioso, aunque el comando siga haciendo la funcionalidad que se le supone, después de lanzar su carga destructiva.

Las aplicaciones de Java resultan extremadamente seguras, ya que no acceden a zonas delicadas de memoria o de sistema, con lo cual evitan la interacción de ciertos virus. Java no posee una semántica específica para modificar la pila de programa, la memoria libre o utilizar objetos y métodos de un programa sin los privilegios del kernel del sistema operativo. Además, para evitar modificaciones por parte de los crackers de la red, implementa un método ultraseguro de autentificación por clave pública. El Cargador de Clases puede verificar una firma digital antes de realizar una instancia de un objeto. Por tanto, ningún objeto se crea y almacena en memoria, sin que se validen los privilegios de acceso. Es decir, la seguridad se integra en el momento de compilación, con el nivel de detalle y de privilegio que sea necesario.

Portable.- Más allá de la portabilidad básica por ser de arquitectura independiente, Java implementa otros estándares de portabilidad para facilitar el desarrollo. Los enteros son siempre enteros y además, enteros de 32 bits en complemento a 2. Java construye sus interfaces de usuario a

través de un sistema abstracto de ventanas de forma que las ventanas puedan ser implantadas en entornos Unix, Pc o Mac.

Interpretado.- El intérprete Java (sistema run-time) puede ejecutar directamente el código objeto. Enlazar un programa, normalmente, consume menos recursos que compilarlo, por lo que los desarrolladores con Java pasarán más tiempo desarrollando y menos esperando por el ordenador. Java es más lento que otros lenguajes de programación, como C++, ya que debe ser interpretado y no ejecutado como sucede en cualquier programa tradicional.

Multithreaded.- Al ser multithreaded (multitarea), Java permite muchas actividades simultáneas en un programa. Los threads (a veces llamados, procesos ligeros), son básicamente pequeños procesos o piezas independientes de un gran proceso. Al estar los threads contruidos en el lenguaje, son más fáciles de usar y más robustos que sus homólogos en C o C++.

El beneficio de ser miltithreaded consiste en un mejor rendimiento interactivo y mejor comportamiento en tiempo real. Aunque el comportamiento en tiempo real está limitado a las capacidades del sistema operativo subyacente (Unix, Windows, etc.), aún supera a los entornos de flujo único de programa (single-threaded) tanto en facilidad de desarrollo como en rendimiento.

Dinámico.- Java se beneficia todo lo posible de la tecnología orientada a objetos. Java no intenta conectar todos los módulos que comprenden una aplicación hasta el tiempo de ejecución. Las librería nuevas o actualizadas no paralizarán las aplicaciones actuales (siempre que mantengan el API anterior).

En resumen existen muchas razones por las que JAVA se ha convertido en un Lenguaje de programación muy codiciado, estos son:

- No debes volver a escribir el código si quieres ejecutar el programa en otra máquina. Un solo código funciona para todos los browsers compatibles con Java o donde se tenga una Máquina Virtual de Java (Mac's, PC's, Sun's, etc).
- Java es un lenguaje de programación orientado a objetos, y tiene todos los beneficios que ofrece esta metodología de programación.
- Un browser compatible con Java deberá ejecutar cualquier programa hecho en Java, esto ahorra a los usuarios tener que estar insertando "plug-ins" y demás programas que a veces nos quitan tiempo y espacio en disco.
- Java es un lenguaje y por lo tanto puede hacer todas las cosas que puede hacer un lenguaje de programación: Cálculos matemáticos, procesadores de palabras, bases de datos, aplicaciones gráficas, animaciones, sonido, hojas de cálculo, etc.

- Las páginas Web, ya no tienen que ser estáticas, se le pueden poner toda clase de elementos multimedia y permiten un alto nivel de interactividad, sin tener que gastar en paquetes costosos de multimedia.
- Porque es una buena tecnología para desarrollar aplicaciones corporativas en donde la red sea algo crítica, Java facilita tremendamente la vida de la programación corporativa.
- Usar Java en el desarrollo de la interface del cliente; Java es suficientemente estable para desarrollar una interface portable. Utilizar herramientas de programación más estables en los servidores, porque son la parte crítica.
- Portar o crear un servidor no-crítico en Java, de forma que tanto cliente como servidor estén escritos en Java.
- Utilizar Java en proyectos de envergadura tanto en el cliente como en el servidor, para valorar la efectividad de Java. [WWW 02 032]

TIPOS DE DATOS:

Los tipos de variables disponibles son básicamente tres:

- tipos básicos (no son objetos)
- arreglos (arrays)
- clases e interfases (al crear una clase o interfase se está definiendo un nuevo tipo)

Los **tipos básicos** son:

Tipo	Tamaño/Formato	Descripción
Byte	8-bit complemento a 2	Entero de un byte
short	16-bit complemento a 2	Entero corto
Int	32-bit complemento a 2	Entero
Long	64-bit complemento a 2	Entero largo
float	32-bit IEEE 754	Punto flotante, precisión simple
double	64-bit IEEE 754	Punto flotante, precisión doble
Char	16-bit caracter Unicode	Un caracter
boolean	true, false	Valor booleano (verdadero o falso)

Tabla 2.09 Tipos de datos en Java.

Los **arrays** son arreglos de cualquier tipo (básico o no).

Comentarios: En Java hay tres tipos de comentarios:

```
// Comentarios para una sola línea
/* Comentarios de una o más líneas*/
/** Comentario de documentación, de una o más líneas*/
```

OPERADORES:

En la siguiente tabla aparecen los operadores que se utilizan en Java, por orden de precedencia:

Posfijos [] . (params) expr++ expr—

Operadores unarios ++expr --expr +expr -expr ~ !

Creación y "cast" new (type)

Multiplicativos * / %

Aditivos + -Desplazamiento

<< >> >>>

Relacionales < > <= >= instanceof

Igualdad == !=

AND bit a bit &

OR exclusivo bit a bit ^

OR inclusivo bit a bit |

AND lógico &&

OR lógico ||

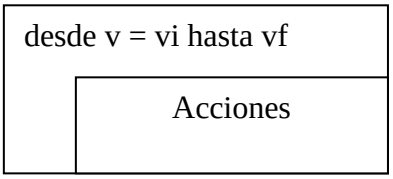
Condicional ? :

Asignación = += -= *= /= %= ^= &= |= <<= >>= >>>=

ESTRUCTURAS DE CONTROL:

if/else:

```
if( Boolean ) { sentencias; }
        else {
        sentencias;
}
```



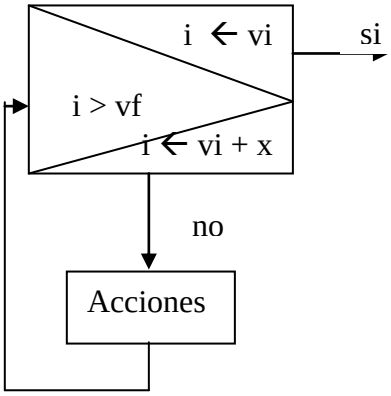
switch:

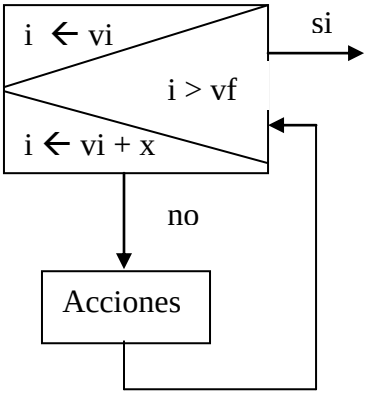
```
switch( expr1 ) {
          case expr2:sentencias;
        break;
          case expr3:sentencias;
        break;
    default:   sentencias;
```

```
        break;
}
```

Bucles for:

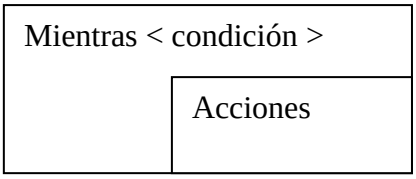
```
for( expr1 inicio; expr2 test; expr3 incremento ) {
        sentencias;
}
```





Bucles while:

```
while( Boolean ) {  sentencias;  }
```



Bucles do/while:

```
do {  sentencias; }while( Boolean );
```

Excepciones: **try-catch-throw**

```
try { sentencias; } catch( Exception ) {
sentencias; }
```

[WWW 02 033]

2.2 RELACIÓN ENTRE LOS AMBIENTES INTEGRADOS DE ESTUDIO Y LOS TRADICIONALES

2.2.1 ANÁLISIS COMPARATIVO ENTRE BORLAND C++ BUILDER VISUAL BASIC VER. 6.0 Y VISUAL J++

ANÁLISIS :

Las dos herramientas C++ Builder y Visual Basic ofrecen al programador la posibilidad de utilizar las últimas arquitecturas disponibles para el desarrollo de software. En cambio Visual J++ ha quedado descontinuado como muchos de los lenguajes que no sustentan su desarrollo para las diversas plataformas de trabajo.

C++ Builder soporta las librerías VCL (diseñadas por Borland) , siendo esta la principal ventaja que la herramienta nos ofrece, pues estas dan un acceso a alto nivel a cualquier objeto visual de Windows, dando al programador la posibilidad de interactuar con ellos de una forma totalmente transparente al API de Windows.

Visual Basic en su última versión, Visual Basic .NET logra adecuarse y presenta las facilidades necesarias al programador para desarrollar las muy requeridas aplicaciones para el internet e intranet. Pero el lenguaje con el que estableceremos la comparación es Visual Basic 6.0.

Visual Basic y Visual J++, son dos herramientas completamente visuales que tienen particularidades y diferencias muy significativas. Visual Basic es una herramienta fácil de aprender y utilizar por que el lenguaje de programación no es orientado a objetos, trabaja con objetos pero no sigue la estructura de la programación orientada a objetos. Visual J++ es una herramienta en la cual se debe tener conocimientos básicos de programación orientada a objetos para poder utilizar.

Estas herramientas básicamente no tienen muchos puntos de comparación por que están orientadas a aplicaciones diferentes, pero se realizará un análisis de acuerdo a parámetros que tengan similitud.

[WWW 02 034]

CONCLUSIONES :

Las tres herramientas son muy potentes, y proporcionan al programador un entorno consistente para la programación. Pero Builder C++ ofrece librerías más preparadas para el diseño de interfaces de usuario amigables y usables, sin dejar a un lado la consistencia de la aplicación final.

PARÁMETROS DE COMPARACION	C++ BUILDER	VISUAL J ++	VISUAL BASIC
Entorno totalmente visual	SI	SI	SI
Desarrollo de CORBA integrado	SI	NO	SI
Creación de controles ActiveX	SI, en un solo paso	SI	SI
Interfaces de Objetos Distribuidos	Tanto CORBA como COM	COM	Unicamente COM
Drivers RDM de alto rendimiento	Tanto Nativos como ODBC	Solo ODBC	Solo ODBC.
Crear ejecutable	SI	SI	SI
Desarrollo visual de Base de Datos	SI	SI	SI, pero es muy básico.
Trabajar con varios proyectos a la vez	SI	NO	SI
Programación Orientada a Objetos	SI	SI	NO
Librerías para componentes visuales	SI	SI	NO
Windows API	SI	SI	SI

SQL	SI	SI	SI
Depuración Remota	SI	NO	NO
Lenguaje Internacionalizado	NO	NO	SI
Ayuda On-Line e internacionalizada	NO	SI	SI
Manuales Internacionalizados	SI	SI	SI
Formación de clases Windows orientado a objetos y unificación de modelos de programación Win32 y HTML	SI	SI	NO
Asistentes para aplicaciones e instalación	SI	PARCIAL	SI
Depurador Just in Time, encuentra los errores aunque no este ejecutando el depurador	NO	SI	NO
Depuración de código HTML	NO	SI	SI

2.2.2 ANÁLISIS COMPARATIVO ENTRE JAVA, PERL Y C++ DE LINUX

Java es un lenguaje de programación que está arrasando en la Web, y gran cantidad de compañías dedicadas al desarrollo de herramientas de programación tienen o han anunciado la pronta aparición de herramientas de desarrollo basadas en Java, la mayor parte de ellas visuales, su campo de aplicación es el mismo del lenguaje C++, por otro lado Perl igualmente combina las mejores características de C++ convirtiéndose en un lenguaje especializado en buscar extraer y presentar información. Estos tres lenguajes no tienen un entorno de desarrollo visual y los tres se pueden compilar bajo cualquier plataforma.

Principales diferencias:

- En Java no existen funciones libres como en C++ o Perl; todo se hace a través de objetos, sin embargo, Java sí dispone de tipos primitivos como enteros, caracteres, etc.
- Java y Perl no dispone de templates, una característica de C++ extremadamente potente, ni de macros, ni va a dejar acceder directamente a la memoria: no hay punteros en Java ni Perl.
- En Perl y Java el código generado no es código máquina, sino un código intermedio, lo que le confiere su excepcional portabilidad al lenguaje.

- Java cuenta con tipos predefinidos similares a los de C++, pero aquí su tamaño está perfectamente delimitado: no vamos a encontrar un int de 32 bits en un compilador de Java y de 16 bits en otro. Java trabaja con cadenas Unicode, que soportan caracteres de otros alfabetos, como cirílico, kanji, etc., de ahí que un char tenga 2 bytes: con uno solo podría representar 255 caracteres distintos.
- En Java no hay uniones ni structs, algo innecesario si tenemos la posibilidad de definir clases, y los arrays son mucho más domésticos que en C++ y Perl; nos avisan cuando se produce un desbordamiento, elevando una excepción, algo que en C++ y Perl puede pasar inadvertido y es fuente de muchos problemas.
- En Java los arrays se crean con **new**, como cualquier objeto.
- Java incorpora clases para que poder trabajar con cadenas: no existe la cadena tipo **char***, dado que Java se niega a dejarnos utilizar punteros. En su lugar usamos objetos de una de dos clases: *String* si vamos a guardar una cadena que no va a ser modificada, o *StringBuffer* si vamos a guardar una cadena que deseamos sea modificable. El que existan dos clases distintas para gestionar cadenas se debe a la gran diferencia de eficiencia entre manejar cadenas de contenido y tamaño fijos, o permitir que puedan redimensionarse.
- A diferencia de C++, en Java la conversión de un objeto de un tipo a otro (*typecasting*) se chequea siempre, de modo que se comprueba

que es correcta la conversión: si no es así, se elevará una excepción del tipo *ClassCastException*.

- Por lo que respecta al uso de variables, Java incorpora un par de características que lo diferencian de C++ y Perl: en primer lugar, no existen variables globales en Java, todo debe estar dentro de una clase en última instancia, lo que no es problema porque se pueden tener variables de clase, como en C++ y Perl. En segundo lugar, Java no permite utilizar una variable a la que no se le haya asignado previamente un valor, lo que evita los problemas derivados de olvidar inicializar variables.
- Otra diferencia entre C++, Perl y Java en cuanto a los operadores es que en Java no es posible sobrecargar operadores: no será posible, por ejemplo, definir "+" como un operador que concatena Strings, algo muy común en C++ y Perl.
- En Java la declaración e implementación de una clase se escriben en un único archivo (xxx.java), mientras que en C++ lo normal es tener un archivo de cabecera para la declaración (xxx.h), y otro para la implementación (xxx.cpp).
- En Java no hay que preocuparse de liberar explícitamente la memoria asignada a los objetos, como en Perl y C++, puesto que Java ejecuta un proceso de recolección de memoria en segundo plano que se encarga de liberar la memoria de aquellos objetos que ya no son utilizables. Esta es una de las características más interesantes de Java.

- Una de las características más potentes de los lenguajes Orientados a Objeto es la capacidad que tienen de implementar *mecanismos*, en los que se define el modo en que interactúan varios objetos de distintas clases de forma genérica, y se deja habitualmente el manejo concreto de los detalles en manos de clases derivadas de las que participan en la interacción: es conveniente encapsular estos detalles en métodos, de modo que aquellas clases que quieran participar en el mecanismo redefinan dichos métodos de acuerdo a sus necesidades, respetando el esquema general de funcionamiento del mecanismo.
- Java, Perl y C++, proporciona un mecanismo de manejo de excepciones muy potente, y utiliza las mismas palabras reservadas, **throw**, **try** y **catch**.
- Java incorpora soporte para multithreading a través de las clases *Thread* y *Runnable* de su librería estándar, lo que significa que su soporte para programación concurrente es portable. Thread cuenta con métodos para hacer inactivo un thread indefinidamente o durante un tiempo determinado por el usuario (*sleep()* y *suspend()*, respectivamente), métodos para cambiar la prioridad, detener un thread, etc.

[WWW 02 035]

[WWW 02 036]

Cuadro comparativo de Lenguajes no visuales:

PARAMETROS DE COMPARACIÓN	JAVA	PERL	C++
Gestión de memoria	Si	No	No
Librería de clases Estándar	Si	Si	Si
Entorno de desarrollo totalmente visual	No	No	No
Permite crear aplicaciones ejecutables	No	Si	Si
Asistentes para desarrollo con bases de datos	No	No	No
Librerías para componentes visuales	Si	No	No
Conexión remota con bases de datos	Si	Si	Si
Herencia múltiple	Si	Si	Si
Soporte para Multithreading	Si	Si	Si
Independiente de la máquina	Si	Si	Si
El lenguaje es formalmente definido	No	No	Si
El lenguaje tiene un simple y fácil análisis gramatical y estructura léxica	Si	Si	Parcial

Se puede modificar la sintaxis del lenguaje	Si	No	No
Acceso a datos en tiempo real	Si	Si	No
Terminación asincrónica de los procesos	Si	Si	No
Herramientas del lenguaje y paquetes de aplicación	Si	Si	Si
Manejo de excepciones	Si	No	No

3. ELEMENTOS DE COMUNICACIÓN CON BASES DE DATOS POR MEDIO DE LA WEB

3.1 DEFINICIÓN DE JDBC DE JAVA

El JDBC (**Java Database Connectivity**) es la Interfaz de programas de aplicación (API) estándar de acceso a Base de Datos con JAVA. Permite ejecutar instrucciones SQL (Structured Query Language), lenguaje de alto nivel para crear, manipular, examinar y gestionar bases de datos relacionales. JDBC se utiliza comúnmente para conectar un programa del usuario con una base de datos por "detrás de la escena", sin importar qué software de administración o manejo de base de datos se utilice para controlarlo. JDBC suministra un API estándar para los desarrolladores y hace posible escribir aplicaciones de base de datos usando un API puro Java.

Usando JDBC es fácil enviar sentencias SQL virtualmente a cualquier sistema de base de datos. En otras palabras, con el API JDBC, no es necesario escribir un programa que acceda a una base de datos Sybase, otro para acceder a Oracle y otro para acceder a Informix. Un único programa escrito usando el API JDBC y el programa será capaz de enviar sentencias SQL a la base de datos apropiada. Y, con una aplicación escrita en el lenguaje de programación Java, tampoco es necesario escribir diferentes aplicaciones para ejecutar en diferentes plataformas. La combinación de Java y JDBC permite al programador escribir una sola vez y ejecutarlo en cualquier entorno.

JDBC es una interfase de bajo nivel, lo que quiere decir que se usa para invocar o llamar a comandos SQL directamente. En esta función trabaja muy bien y es más fácil de usar que otros API's de conexión a bases de datos, pero está diseñado de forma que también sea la base sobre la cual construir interfaces y herramientas de alto nivel. Una interfase de alto nivel es amigable, usa un API más entendible o más conveniente que luego se traduce en la interfase de bajo nivel tal como JDBC.

En resumen lo que hace posible el JDBC es lo siguiente:

- Establece una conexión con una Base de Datos que puede ser remota o no.
- Enviar sentencias SQL a la Base de Datos.
- Procesa los resultados obtenidos de la Base de Datos.

[WWW 03 020]

La API JDBC soporta dos modelos distintos de acceso a las Bases de Datos:

- ***Modelo de dos capas*** En este modelo la aplicación JAVA o el Applet, se conectan directamente con la BD. Tiene una capa cliente y una capa servidor, Así pues el driver JDBC específico para conectarse con la BD estará instalado en el sistema local (cliente). La BD puede estar en otra maquina y se accede a ella mediante red (servidor). Esta configuración también se llama Cliente/Servidor. El programa cliente envía

instrucciones SQL a la BD, y esta las procesa y envía los resultados de vuelta al usuario.

Una de las ventajas de este modelo es que puede reducir el tiempo de desarrollo debido al hecho de que todo el sistema es considerablemente más simple y más pequeño. Segundo, el acoplamiento fuerte puede mejorar potencialmente el rendimiento del sistema ya que el cliente puede fácilmente aprovecharse de las funcionalidades específicas del servidor que podrían no estar disponibles para sistemas con un acoplamiento más ligero.

Por otro lado, este acoplamiento puede provocar varios problemas. El más notable, el mantenimiento del sistema se puede volver más díficil porque los cambios en el servidor pueden romper al cliente y viceversa. Además, si la base de datos cambia, todo el código del cliente deberá ser modificado. Si el cliente está altamente distribuido, la propagación de los cambios en el sistema puede ser díficil, y en algunos escenarios imposible. Como resultado, las aplicaciones de dos capas pueden ser útiles en un entorno de LAN corporativa donde el completo control de todos los clientes se consigue, o al inicio, o en el estado prototipal de un proyecto donde diferentes opciones están siendo evaluadas.

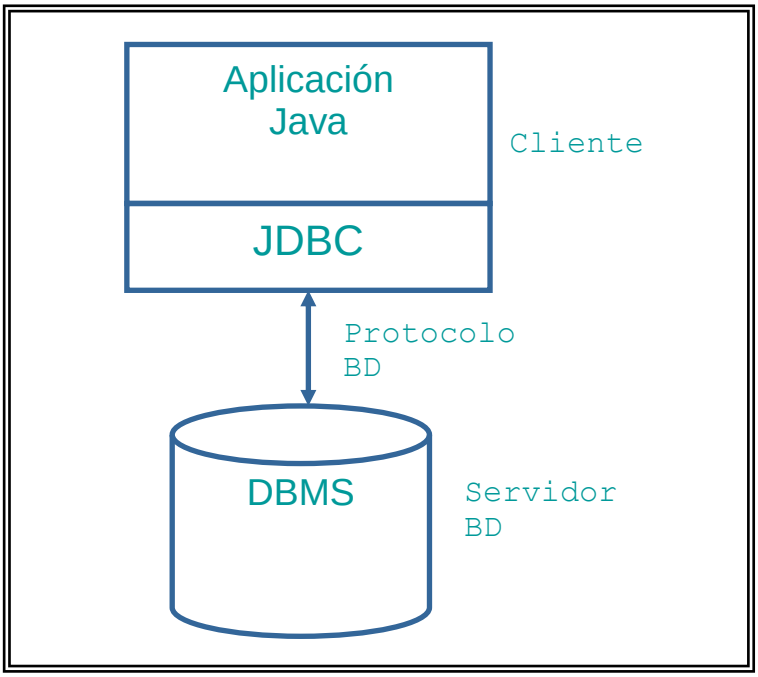


Fig. 3.1.1 Modelo de dos Capas

- ***Modelo de n capas:*** El modelo de n-capas tiene una capa cliente, una capa servidor, y al menos una capa intermedia. En este modelo, las instrucciones son enviadas a una capa intermedia que se encarga de enviar las sentencias SQL a la BD. El manejador de BD procesa las sentencias y retorna los resultados a la capa intermedia que se encarga de enviarlos al usuario. Debido a la capa extra, muchos de los problemas que afectan a los modelos de dos capas no afectarán. Por ejemplo, la capa media ahora mantiene información de la conexión a la base de datos. Esto significa que los clientes sólo tienen que conocer la capa media. Como la capa media generalmente está operando en la misma localización física que el servidor (por ejemplo, ambos componentes pueden estar detrás del mismo firewall), mantener la capa media es considerablemente más sencillo que mantener cientos de instalaciones clientes.

Otra ventaja de la aproximación de n-capas es que todo el sistema se puede escalar fácilmente para manejar más usuarios. Todo lo que necesitamos hacer es añadir más capas medias o más capas servidores, dependiendo de los resultados de las operaciones de perfilado. Las capas intermedias normalmente se implementan usando servidores Web usando tecnologías JavaServer Pages y Servlets. [WWW 03 021]

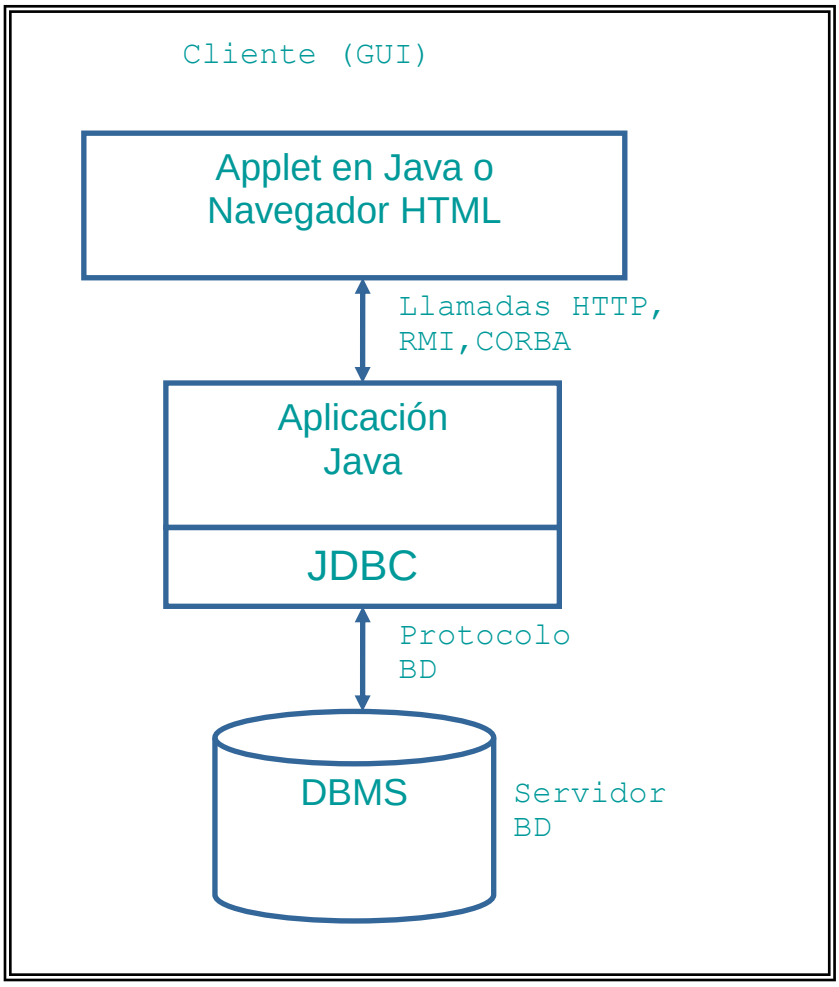


Fig. 3.1.2 Modelo de n-capas

3.2 DEFINICIÓN DE ODBC

ODBC (Open Database Connectivity) es la interfase de aplicaciones API para conectarse con Bases de Datos tanto relacionales como no relacionales, mas usada por los programadores de aplicaciones.

La norma ODBC permite a las aplicaciones utilizar el potente lenguaje de consulta ***SQL*** (Structured Query Language). Se utiliza generalmente para las conexiones a las bases implantadas en los servidores.

Los Drivers OBDC (archivos que contienen la entera descripción de la estructura de los archivos de otras aplicaciones y que permiten la coherencia de los intercambios entre su base y el archivo consultado) están disponibles para Oracle, Microsoft SQL Server, Sybase SQL Server y numerosos otros administradores de bases de datos en un gran número de plataformas.

El esquema que mostramos a continuación muestra el funcionamiento del ODBC. La aplicación envía una consulta SQL. El Driver Manager intercepta esta llamada y la dirige de nuevo hacia el driver ODBC adecuado. Este driver analiza la consulta SQL y luego recupera los datos por la fuente de datos. Se hacen volver después dichos datos a la aplicación. La fuente de datos es entonces el vínculo entre la aplicación y el archivo físico. [WWW 03 022]

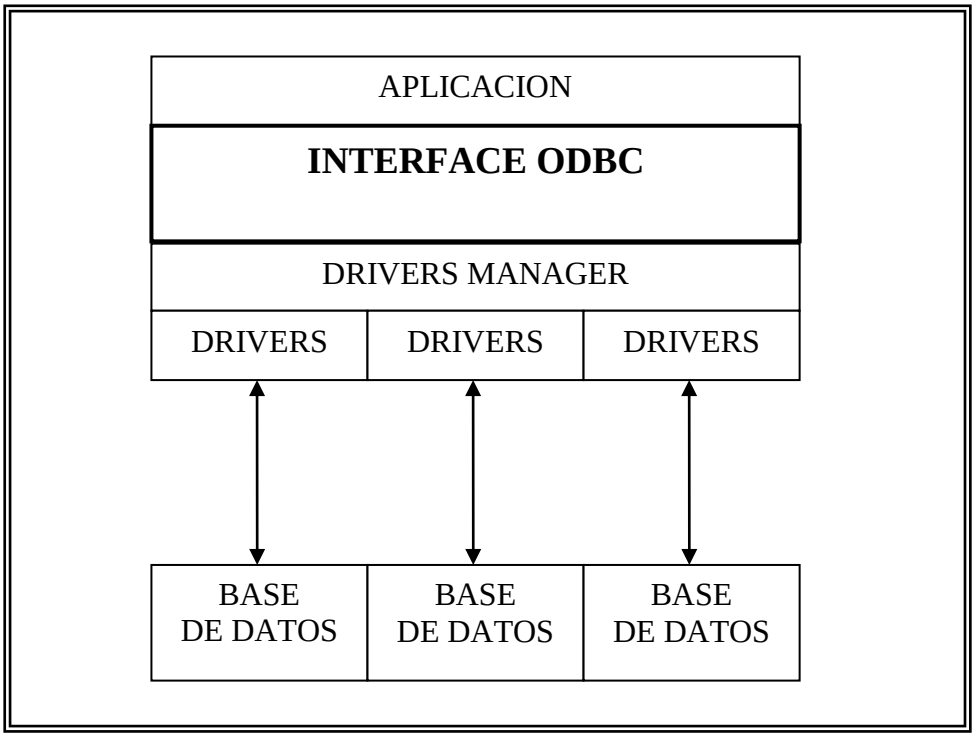


Fig. 3.2.1 Funcionamiento del ODBC

3.3 JDBC vs. ODBC

El ODBC de Microsoft (Open Database Connectivity), es probablemente el API más extendido para el acceso a bases de datos. Ofrece la posibilidad de conectar a la mayoría de las bases de datos en casi todas las plataformas, por lo que se puede usar ODBC desde Java, pero es preferible hacerlo con la ayuda de JDBC mediante el puente JDBC-ODBC. Nos preguntaremos que significado tiene entonces JDBC si ya existe una interfase popular que supuestamente hace lo mismo. La respuesta es que usaremos JDBC por diferentes razones:

- ODBC no es apropiado para su uso directo con Java porque usa una interfase C. Las llamadas desde Java a código nativo C tienen un número de inconvenientes en la seguridad, la implementación, la robustez y,

como C no es un lenguaje portable las aplicaciones JAVA perderían también automáticamente su portabilidad.

- Una traducción literal del API C de ODBC en el API Java podría no ser deseable. Por ejemplo, Java no tiene punteros, y ODBC hace un uso copioso de ellos, incluyendo el notoriamente propenso a errores "void*". Se puede pensar en JDBC como un ODBC traducido a una interfase orientada a objeto que es el natural para programadores Java.
- ODBC es difícil de aprender. Mezcla características simples y avanzadas juntas, y sus opciones son complejas para 'querys' simples. JDBC por otro lado, ha sido diseñado para mantener las cosas sencillas mientras que permite las características avanzadas cuando éstas son necesarias.
- Un API Java como JDBC es necesario en orden a permitir una solución Java "pura". Cuando se usa ODBC, el gestor de drivers de ODBC y los drivers deben instalarse manualmente en cada máquina cliente. Como el driver JDBC esta completamente escrito en Java, el código JDBC es automáticamente instalable, portable y seguro en todas las plataformas Java.

En resumen, el API JDBC es la interfase natural de Java para las abstracciones y conceptos básicos de SQL. JDBC retiene las características básicas de diseño de ODBC; de hecho, ambas interfaces están basados en el X/Open SQL CLI (Call Level Interface).

Recientemente Microsoft ha introducido nuevas API detrás de ODBC. RDO, ADO y OLE DB. Estos diseños se mueven en la misma dirección que JDBC en muchas maneras, puesto que se les da una orientación a objetos basándose en clases que se implementan sobre ODBC.

Hay que decir también, que existen drivers puente entre JDBC-ODBC. El puente JDBC-ODBC permite a los drivers ODBC usarse como drivers JDBC. Fue implementado como una forma de llegar rápidamente al fondo de JDBC y para proveer de acceso a los DBMS menos populares si no existen drivers JDBC para ellos. [WWW 03 023]

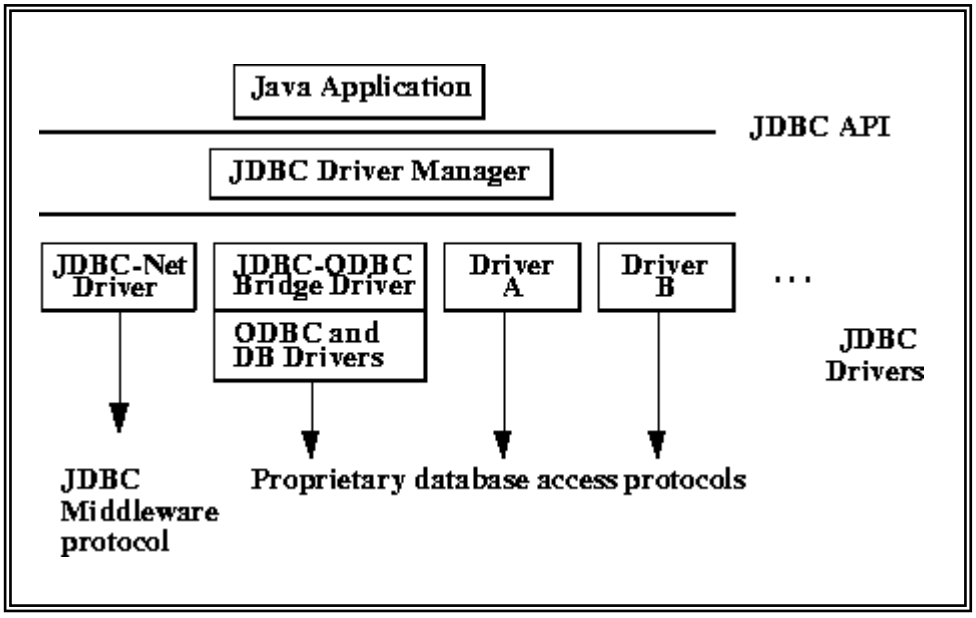


Fig. 3.3.1 Puente JDBC-ODBC

Entre estos tipos de drivers tenemos:

Puente JDBC-ODBC más driver ODBC: Los drivers de este tipo son diferentes al driver ODBC adjunto al puente **JDBC-ODBC,** este puente procesa las llamadas JDBC y las convierte en llamadas ODBC. La principal necesidad de

este tipo de driver es los DLLs ODBC en el cliente, lo que limita la portabilidad de nuestras aplicaciones a aquellas plataformas para las que exista driver ODBC en la máquina donde corra nuestra aplicación (cliente), y complica el mantenimiento.

Como ODBC se ha estado utilizando desde hace bastante tiempo (más que el lenguaje Java), los drivers ODBC son ubicuos. Esto hace de este tipo de drivers una buena elección para aprender cómo conectar programas Java a bases de datos. De hecho, incluso hay drivers ODBC que nos permiten asignar fuentes de datos ODBC a una aplicación Microsoft Excel o ficheros de texto plano. Sin embargo, el nivel extra de indirección, puede resultar una pérdida de rendimiento ya que el JDBC es transferido dentro de ODBC, que luego es transferido en el protocolo específico de la base de datos.

Otro problema potencial de este puente es su utilización en aplicaciones distribuidas. Como el propio puente no soporta comunicación distribuida, la única forma de que estos drivers puedan trabajar a través de la red es si el propio driver ODBC soporta interacción remota. Para drivers ODBC sencillos, esta no es una opción, y mientras que las grandes bases de datos tienen drivers ODBC que pueden trabajar de forma remota, no pueden competir con el mejor rendimiento de los drivers JDBC puro Java.

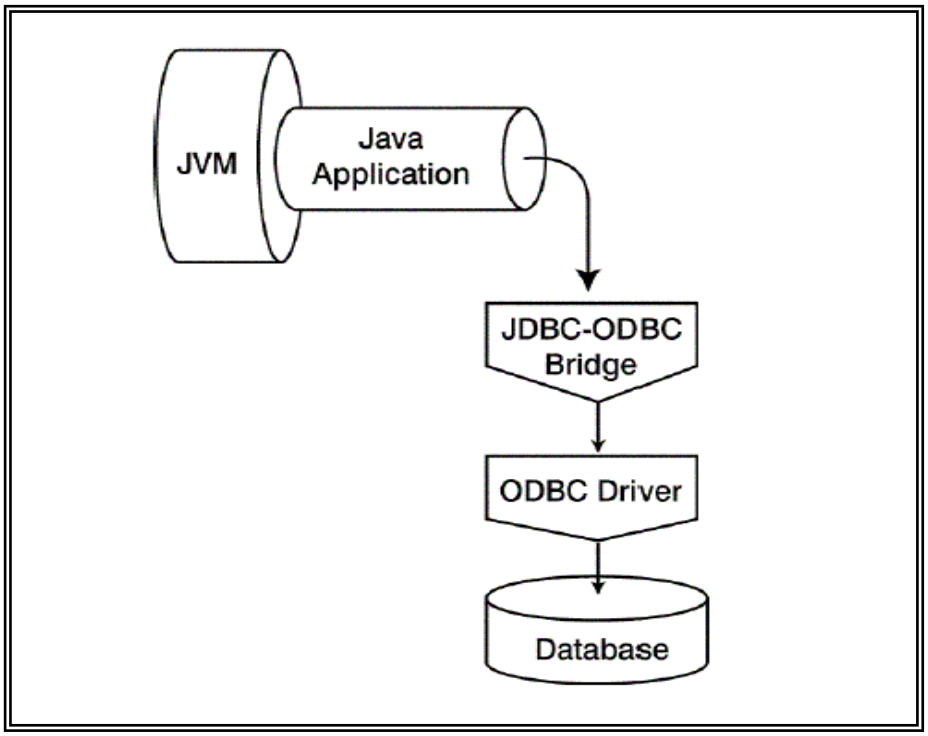


Fig. 3.3.2 Puente JDBC-ODBC más driver ODBC

Driver Java parcialmente Nativo: Este driver también es conocido como drivers Java parciales, porque traducen directamente el API JDBC en un API específico de la base de datos, al estar escritos solo parcialmente en Java comparten los problemas de portabilidad e instalación del puente JDBC/ODBC.

La aplicación cliente de base de datos debe tener las librerías cliente apropiadas para la base de datos, que podrían incluir código binario instalado y posiblemente ejecutándose. Para una aplicación distribuida, este requerimiento puede introducir problemas extra con las licencias, así como posibles pesadillas con los problemas de distribución de código. Por ejemplo, usar un modelo de este tipo restringe a los desarrolladores a utilizar plataformas y sistemas operativos soportados por la librería cliente de la base de datos.

Sin embargo, este modelo puede funcionar eficientemente, cuando la base cliente está fuertemente controlada. Esto ocurre típicamente en LANs corporativas.

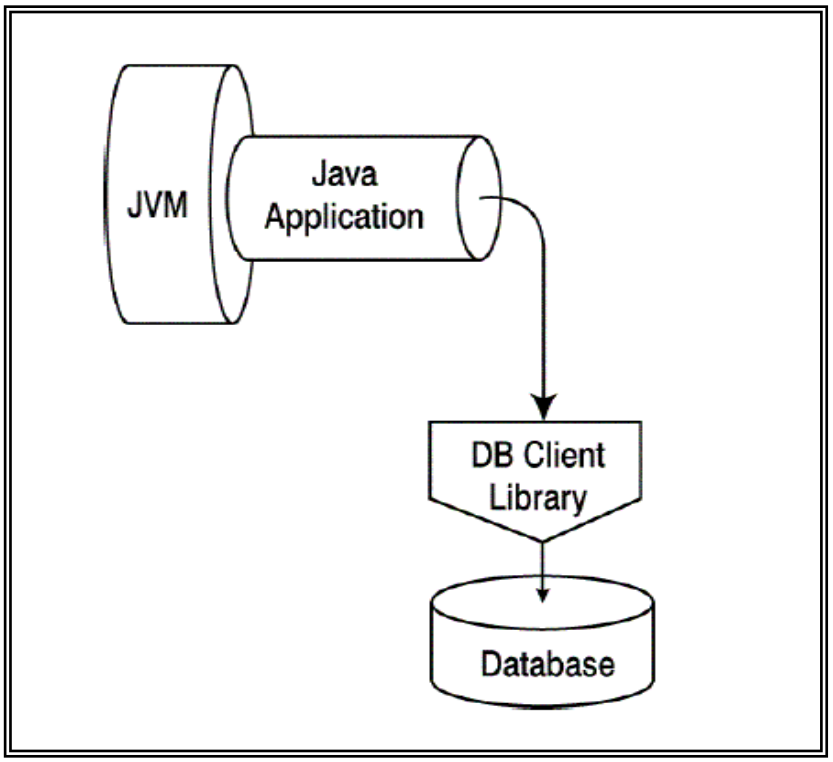


Fig. 3.3.3 Driver Java Parcialmente Nativo

Driver Java nativo JDBC-Net: Estos drivers lo constituyen aquellos que están escritos en Java completamente, y convierten las llamadas JDBC a un protocolo independiente de la base de datos: estas llamadas se transmiten por la red hasta una aplicación servidora que recibe las peticiones y las convierte en llamadas a funciones nativas de la base de datos utilizada. Este tipo de drivers es muy flexible, dado que las aplicaciones cliente son tan portables como lo sea Java, siendo la única limitación la necesidad de que el sistema operativo del servidor sea el que requiere la aplicación que hace las veces de servidor, lo que es mucho menos limitante. Un factor importante a tener en cuenta es que utilizando este tipo de drivers no será necesario configurar la máquina donde corra la aplicación,

solamente el servidor, lo que puede evitar bastantes quebraderos de cabeza a la hora de la instalación de la aplicación.

El driver JDBC no comunica directamente con la base de datos; comunica con un servidor de capa media, que a su vez comunica con la base de datos. Este nivel extra de indirección proporciona flexibilidad en que se puede acceder a diferentes bases de datos desde el mismo código porque el servidor de la capa media oculta las especificidades a la aplicación Java. Para cambiar a una base de datos diferente, sólo necesitamos cambiar los parámetros en el servidor de la capa media. (Un punto a observar: el formato de la base de datos a la que estamos accediendo debe ser soportado por el servidor de la capa media).

El lado negativo de estos drivers es que el nivel extra de indirección puede perjudicar el rendimiento general del sistema. Por otro lado, si una aplicación necesita interactuar con una variedad de formatos de bases de datos, un driver del tipo tres en una aproximación adecuada debido al hecho de que se usa el mismo driver JDBC sin importar la base de datos subyacente.

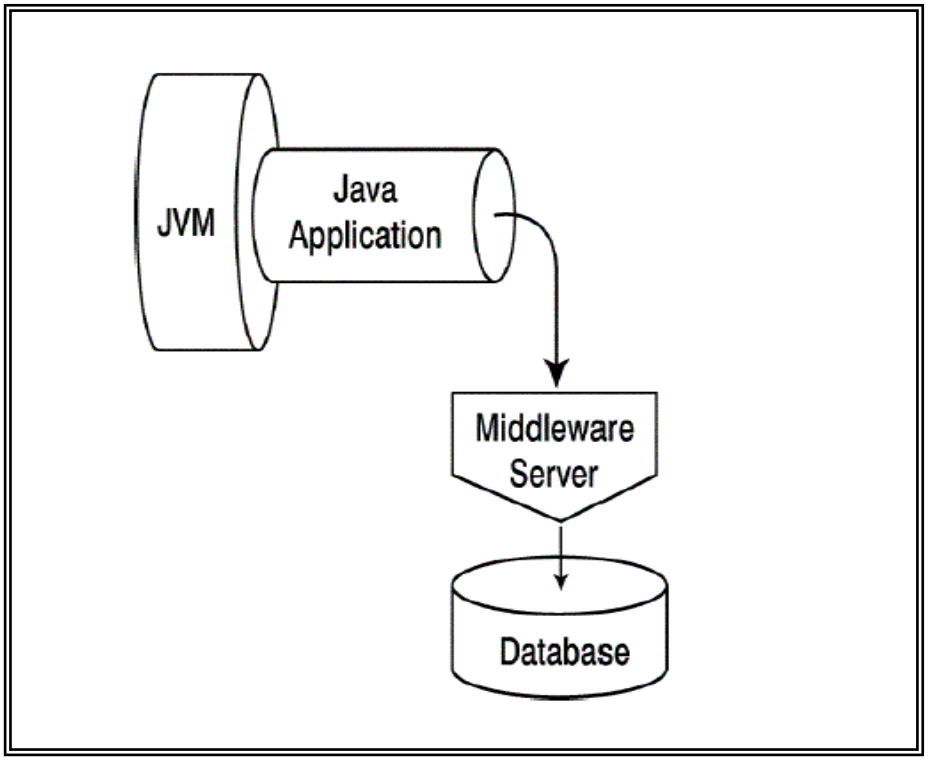


Fig. 3.3.4 Driver Java nativo JDBC-Net

Driver puro Java y nativo-protocolo: Este tipo de drivers lo constituyen aquellos que están escritos totalmente en Java, y convierten todas las llamadas JDBC a llamadas nativas del gestor de base de datos correspondiente. Muchos programadores consideran éste el mejor tipo de driver, ya que normalmente proporciona un rendimiento óptimo y permite al desarrollador utilizar las funcionalidades específicas de la base de datos. Por supuesto este acoplamiento puede reducir la flexibilidad, especialmente si necesitamos cambiar la base de datos subyacente en una aplicación. Este tipo de driver se usa frecuentemente en applets y otras aplicaciones altamente distribuidas. [WWW 03 024]

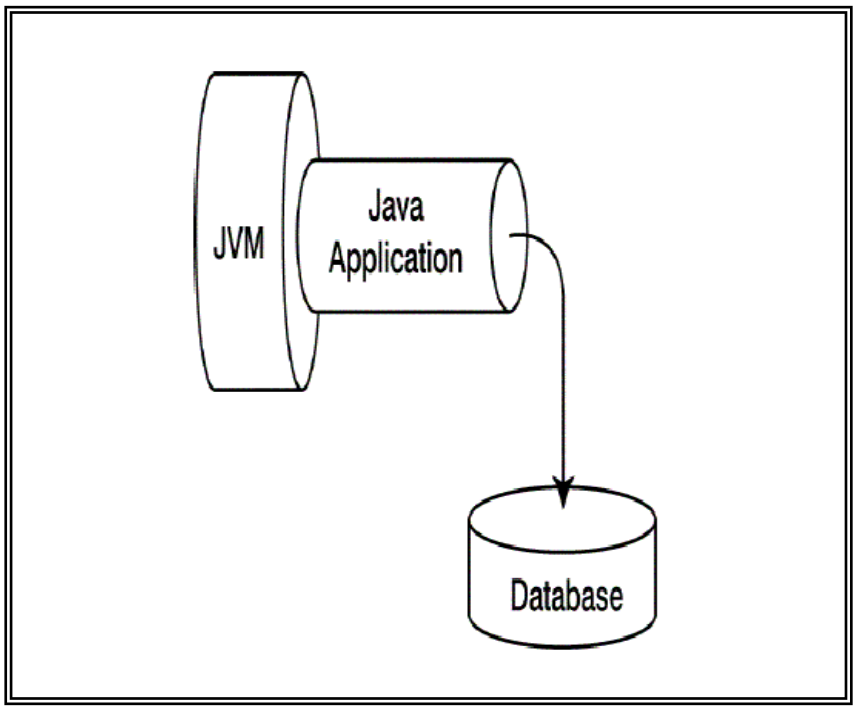


Fig. 3.3.4 Driver puro Java y nativo-protocolo

3.4 INSTALACIÓN Y CONFIGURACIÓN DE JDBC PARA JAVA

Lo primero que tenemos que hacer es asegurarnos de que disponemos de la configuración apropiada. Esto incluye los siguientes pasos.

- ***Instalar Java y el JDBC en nuestra máquina.***

  Para instalar tanto la plataforma JAVA como el API JDBC, simplemente tenemos que seguir las instrucciones de descarga de la última versión del JDK (Java Development Kit). Junto con el JDK también viene el JDBC. Podrás encontrar la última versión del JDK en la siguiente dirección:

  http://java.sun.com/products/JDK/CurrentRelease

- ***Instalar un driver en nuestra máquin***a.

  Nuestro Driver debe incluir instrucciones para su instalación. Para los drivers JDBC escritos para controladores de bases de datos específicos la instalación consiste sólo en copiar el driver en nuesta máquina; no se necesita ninguna configuración especial.

  El driver "puente JDBC-ODBC" no es tan sencillo de configurar. Si descargamos las versiones Solaris o Windows de JDK 1.1, automáticamente obtendremos una versión del driver Bridge JDBC-ODBC, que tampoco requiere una configuración especial. Si embargo, ODBC, si lo necesita. Si no tenemos ODBC en nuestra máquina, necesitaremos preguntarle al vendedor del driver ODBC sobre su instalación y configuración.

- ***Instalar nuestro Controlador de Base de Datos si es necesario.***

  Si no tenemos instalado un controlador de base de datos, necesitaremos seguir las instrucciones de instalación del vendedor. La mayoría de los usuarios tienen un controlador de base de datos instalado y trabajarán con un base de datos establecida.

- ***Establecer una Conexión***

  Lo primero que tenemos que hacer es establecer una conexión con el controlador de base de datos que queremos utilizar.

1.- Cargar el driver

2.- Hacer la conexión.

- ***Cargar los Drivers:***

Cargar el driver o drivers que queremos utilizar, solo implica una línea de código. Si, por ejemplo, queremos utilizar el puente JDBC-ODBC, se cargaría la siguiente línea de código.

Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

La documentación del driver nos dará el nombre de la clase a utilizar. Por ejemplo, si el nombre de la clase es **jdbc.DriverXYZ**, cargaríamos el driver con esta línea de código.

Class.forName("jdbc.DriverXYZ");

No necesitamos crear un ejemplar de un driver y registrarlo con el **DriverManager** porque la llamada a **Class.forName** lo hace automáticamente. Si hubiéramos creado nuestro propio ejemplar, crearíamos un duplicado innecesario, pero no pasaría nada.

Una vez cargado el driver, es posible hacer una conexión con un controlador de base de datos.

- ***Hacer la Conexión***

  El segundo paso para establecer una conexión es tener el driver apropiado conectado al controlador de base de datos. La siguiente línea de código ilustra la idea general.

  Connection con = DriverManager.getConnection(url, "myLogin", "myPassword");

  Hay que saber qué suministrar para **url**. Si estamos utilizando el puente JDBC-ODBC, el JDBC URL empezará con **jdbc:odbc:**. el resto de la URL normalmente es la fuente de nuestros datos o el sistema de base de datos. Por eso, si estamos utilizando ODBC para acceder a una fuente de datos ODBC llamada "**Hola**" por ejemplo, nuestro URL podría ser **jdbc:odbc:Hola**. En lugar de "**myLogin**" pondríamos el nombre utilizado para entrar en el controlador de la base de datos; en lugar de "**myPassword**" pondríamos nuestra password para el controlador de la base de datos. Por eso si entramos en el controlador con el nombre "Jane" y la password of "**J,**" estas dos líneas de código establecerán una conexión.

  String url = "jdbc:odbc:Hola";

  Connection con = DriverManager.getConnection(url, "Jane", "J");

  Si estamos utilizando un puente JDBC desarrollado por una tercera parte, la documentación nos dirá el subprotocolo a utilizar, es decir, qué poner después de **jdbc:** en la URL. Por ejemplo, si el desarrollador ha

registrado el nombre "acme" como el subprotocolo, la primera y segunda parte de la URL de JDBC serán **jdbc:acme:**. La documentación del driver también nos dará las guías para el resto de la URL del JDBC. Esta última parte de la URL suministra información para la identificación de los datos fuente.

Si uno de los drivers que hemos cargado reconoce la URL suministrada por el método **DriverManager.getConnection**, dicho driver establecerá una conexión con el controlador de base de datos especificado en la URL del JDBC. La clase **DriverManager**, como su nombre indica, maneja todos los detalles del establecimiento de la conexión detrás de la escena. A menos que estemos escribiendo un driver, posiblemente nunca utilizaremos ningún método del interface **Driver**, y el único método de **DriverManager** que realmente necesitaremos conocer es **DriverManager.getConnection**.

La conexión devuelta por el método **DriverManager.getConnection** es una conexión abierta que se puede utilizar para crear sentencias JDBC que pasen nuestras sentencias SQL al controlador de la base de datos.

[WWW 03 025]

Tabla 3.1 Instalación JDBC

1.- Instalar Java y el JDBC en nuestra máquina
2.-Instalar un driver en nuestra máquina.
3.- Instalar nuestro Controlador de Base de Datos.
4.- Establecer una Conexión
5.- Cargar los Drivers.
6.- Hacer la Conexión

3.5 INSTALACIÓN Y CONFIGURACION DE ODBC

En Panel de control en el icono Fuentes de Datos ODBC dar un click, abrirá el cuadro de diálogo del Administrador de Fuentes de Datos ODBC. ODBC requiere que cualquier base de datos que desee usar sea añadida bien como una DSN de usuario (si quiere que solo este disponible para usted), o como una DSN de sistema (si quiere que esté disponible para todos los usuarios en esa máquina). Dependiendo de su preferencia, deberá seleccionar la etiqueta **DSN de usuario** o **DSN de sistema**.

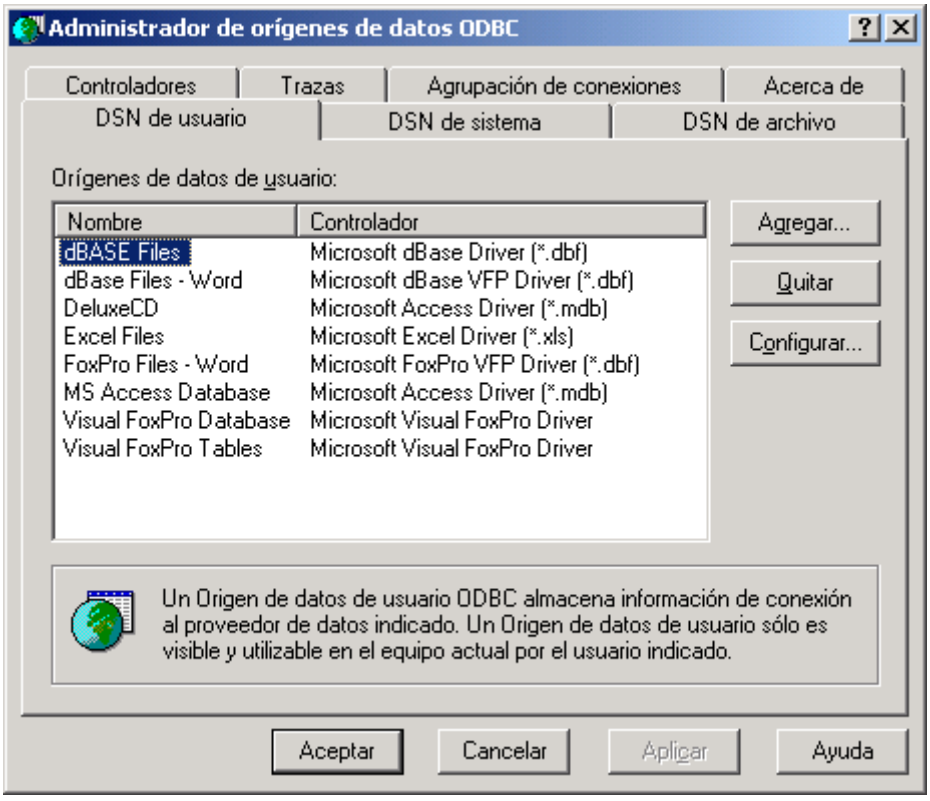


fig. 3.5.1 Administrador de fuentes de Datos ODBC

Haga clic en Agregar, aparecerá un cuadro de diálogo 'Crear nuevo origen de Datos' (una fuente de datos es, esencialmente, un *link* a una base de datos existente). Seleccione el *driver* de su base de datos, por ejemplo, **Driver do**

Microsoft Access (*.mdb), esto requiere tener instalado Microsoft Access. Haga clic en **Finalizar**

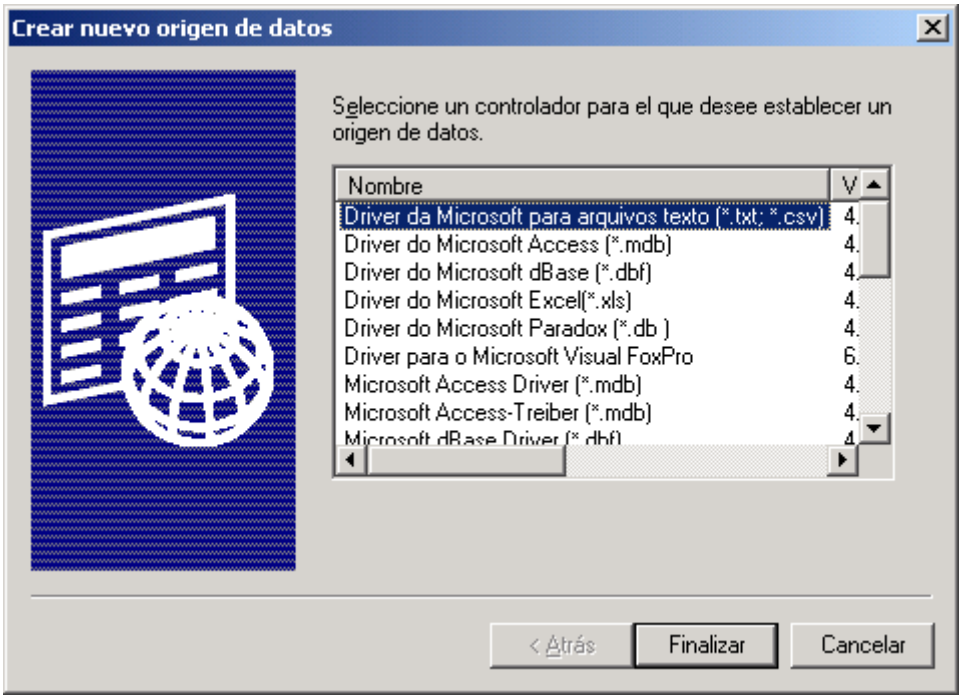


fig . 3.5.2 Crear una Nueva fuente de datos

En el cuadro de diálogo de configuración del ODBC de Microsoft Access, introduzca el nombre de la base de datos y la descripción, haga clic en **Seleccionar** y seleccione el archivo de base de datos. Haga clic en aceptar, y salga del Administrador de Fuentes de Datos ODBC. La fuente de datos ya se encuentra configurada y lista para ser utilizada.

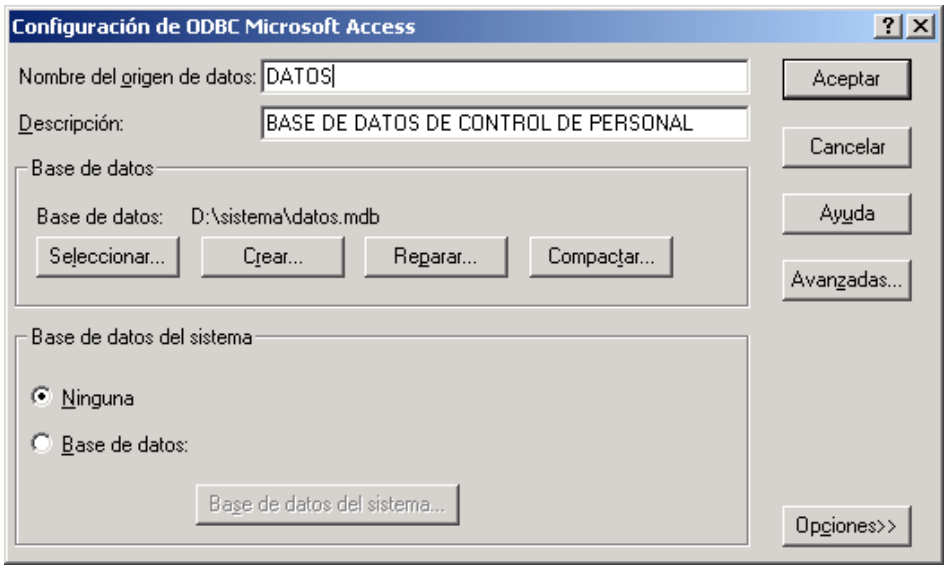


Fig. 3.5.3 Nombre de la Base de Datos, descripción y selección

[WWW 03 026]

Tabla 3.1 Instalación ODBC para C++ Builder

1.- Instalar el ODBC.
2.- Completar el Administrador de fuentes de Datos ODBC.
3.- Crear una Nueva fuente de datos.
4.- Nombre de la Base de Datos, descripción y selección

3.6 INSTALACIÓN Y CONFIGURACIÓN DE ODBC PARA PERL DE LINUX

DBI: bases de datos

El módulo DBI se usa para manipular una base de datos relacional, como Oracle, Access, SQL Server, MySQL, etc. DBI significa Data Base Interface, y supone una capa de alto nivel para acceder a una base de datos. Requiere tener instalado el módulo DBD (Data Base Driver) de la base de datos correspondiente, es independiente de la base de datos con la que se está trabajando, lo cual significa que podremos trabajar con bases de datos como Oracle, Sybase, Informix, MySql, mSql, bases de datos con soporte ODBC (MS-Access, SQL Server), etc. Por el momento DBI sólo trabaja con **bases de datos relacionales** y no con bases de datos orientadas a objetos

DBD (Driver de Base de Datos-*Data Base Driver*) es el driver de la base de datos, es decir, se encarga de llevar a cabo lo que se pide en que se haga en nuestro script Perl (usando DBI) en una base de datos específica. Existe un módulo DBD para cada tipo de base de datos, dicho módulo se encarga de pasar las peticiones que realizamos en DBI a peticiones a la base de datos sobre la que estamos trabajando. Para trabajar con una base de datos determinada nos hace falta tener controlado el módulo DBD correspondiente, por ejemplo, para trabajar con la base de datos MySql nos hace falta el módulo DBD-MySql.

La arquitectura de DBI se puede dividir en **dos** partes, como indica la siguiente figura, el propio interfaz **DBI** y los drivers, que se encargan de acceder a la base de datos correspondiente. Los drivers se implementan en el módulo **DBD**.

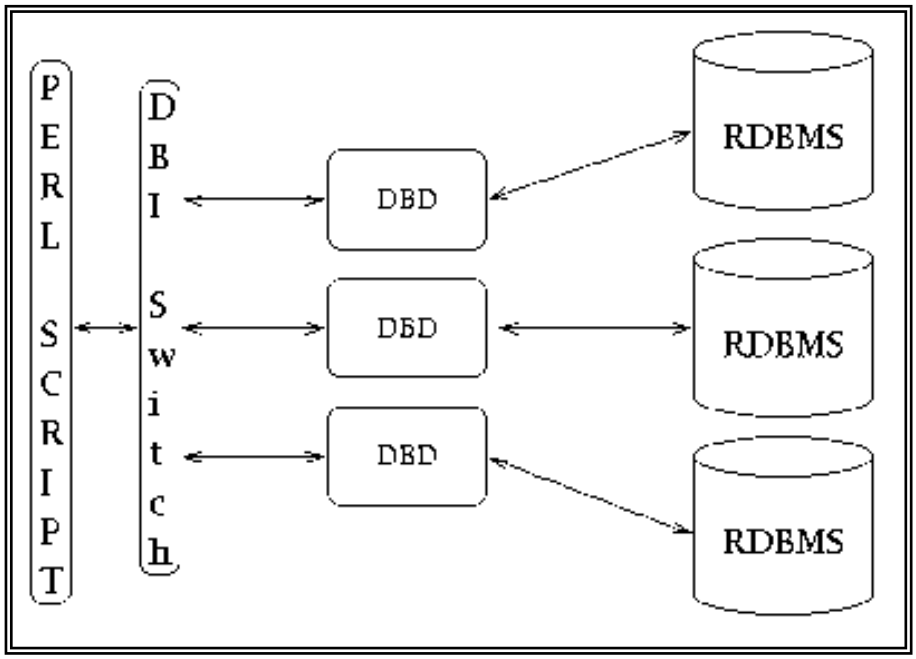


Fig. dfgds

Para trabajar con bases de datos en Perl sólo nos hace falta tener instalado Perl, la base de datos con la que vamos a trabajar, el módulo DBI y el módulo DBD para la base de datos instalada. Para programar con Perl y acceder a la base de datos nos hace falta saber programar en Perl, saber usar el módulo DBI y saber SQL, porque DBI se comunica con las base de datos a través de **SQL** (Lenguaje estrucuturado de consultas-Structured Query Language-).

DBI **no conoce nada** acerca de la base de datos sobre la que se está trabajando, eso depende del módulo DBD correspondiente. La separación de los drivers de DBI provoca que le módulo DBI se pueda extender a **casi cualquier base de**

datos que esté soportada, teniendo en cuenta, que también depende de la portabilidad de las sentencias SQL que utilicemos. Existen drivers para muchas de las bases de datos más populares.

- **Instalación DBI:**

1. Ejecutar el **ppm** (Perl Packge Manager) con: *c:\perl\bin\ppm.bat install DBD-ODBC*
2. Instalar el módulo de DBI, ejecutando dentro del ppm: *install DBI*
3. Instalar el módulo de DBD-BDD(seleccionada), ejecutando dentro del ppm: *install DBD-BDD*
4. Instalar el módulo de DBD-ODBC, ejecutando dentro del ppm: *install DBD-ODBC*

Si tenemos la distribución de Active State para plataformas Windows, debemos obtener el paquete DBI.zip para instalar a través de la herramienta PPM.

- ***Conexión a la base de datos:***

Supongamos que tenemos una base de datos MSAccess. Lo primero que tenemos que hacer es una DNS de sistema que enlace a dicha base de datos (se hace con el administrador de fuentes de datos ODBC, en el panel de control, como se indica en la sección anterior). Le ponemos el nombre AccessPerl:

Para trabajar con DBI lo primero (antes que nada) que hay que hacer es leer el módulo DBI:

```
use DBI;
```

Después y para cualquier tarea que se desee realizar hay que abrir una conexión con la base de datos, la información que se utiliza para establecer la conexión es:

```
use DBI;
use DBD::ODBC;
$db = DBI->connect('dbi:ODBC:AccessPerl', 'usuario', 'password');
# $db contiene la conexión a la base de datos
# Al tratarse de Access, los string usuario y password
# pueden estar vacíos.
if( ! defined $db ) {
      die “No se puede conectar a la base de datos\n”;
}
```

- **Nombre de la fuente de datos**: Esto es una cadena que contiene los siguientes elementos, separados por dos puntos:
  - dbi
  - *Nombre del driver*: Nombre del driver, por ejemplo, odbc, mysql, oracle, etc.
  - *Nombre de la base de datos*: Nombre de la base de datos con la que estamos trabajando.
- **Usuario:** Usuario que se va a conectar a la base de datos. Dentro de la base de datos hay distintos usuarios, cada uno, con sus propias tablas y con una serie de privilegios asociados.

- **Password**: Clave de seguridad del usuario.

- **Host**(OPCIONAL): Aqui indicamos la máquina donde está la base de datos.

- **Puerto**(OPCIONAL): Aquí indicamos el puerto que se va a usar para conectarse a la base de datos.

Una vez que se ha terminado de trabajar con la base de datos se debe **cerrar la conexión** con la base de datos, no es estrictamente necesario, pero si aconsejable, ya que algunas bases de datos pueden no terminar sus transacciones correctamente y sobre todo estamos consumiendo recursos inadecuadamente. En plataformas Windows puede originar, bajo algunas circunstancias errores de memoria graves. Para cerrar la conexión, haremos:

$db->disconnect();

Veamos un ejemplo:

```
#!/usr/bin/perl
#Usamos las librerias de acceso a BD
use DBI;

my $base_datos="ejemplo"; #Nombre de las base de datos
my $usuario="prog"; #Usuario de la BD
my $clave="utn"; #Password de la BD
my $driver="mysql"; #Utilizamos el driver de mysql

#Conectamos con la BD, si no podemos, ponemos un mensaje de
error
my $dbh = DBI->connect("dbi:$driver:$base_datos",$usuario,$clave)
|| die "\nError al abrir la base datos: $DBI::errstr\n";

#Decimos que hemos creado la tabla
print "\nSe ha conectado con la BD $base_datos del driver $driver\n";

#Nos desconectamos de la BD. Mostramos un mensaje si hay
#algun fallo
$dbh->disconnect || warn "\nFallo al desconectar.\nError:
$DBI::errstr\n";

#Terminamos
exit;
```

Ejemplo: script en Perl que conecta y desconecta a un sistema de bases de datos MySQL, cuya base de datos se llama ejemplo, el nombre del usuario es prog y el password utn.

Control de errores:

Cuando se genera un error en la comunicación con la base de datos (lo más común es por errores de programación SQL), Perl genera un error que muestra por la salida estándar de error (por defecto, la consola) y termina la ejecución inmediatamente. Podemos controlar el grado de errores que deseamos generar:

1. Sólo mensaje de error, sin finalizar la ejecución

   db = DBI->connect('dbi:ODBC:AccessPerl', '', '',

   { PrintError => 1, RaiseError => 0 });

2. Mensaje de error y se aborta la ejecución inmediatamente (lo que se hace por defecto)

   $db = DBI->connect('dbi:ODBC:AccessPerl', '', '',

   { PrintError => 1, RaiseError => 1 });

3. Ningún mensaje de error (habrá que consultar la variable ***$DBI:err***)

   $db = DBI->connect('dbi:ODBC:AccessPerl', '', '',

   { PrintError => 0, RaiseError => 0 });

[WWW 03 027]

[WWW 03 028]

4. DESARROLLO DE LA METODOLOGÍA DE EVALUACIÓN DE AMBIENTES INTEGRADOS DE DESARROLLO DE SOFTWARE.

Cuando se quiere seleccionar un producto de software es necesario tener como base un modelo para la evaluación del producto. El objetivo es formular un modelo de calidad de software considerando la eficiencia y la efectividad del software así como elementos del proceso desde un enfoque sistémico. Para lograr el objetivo se estudiaron diferentes modelos de calidad: ISO 9126, ISO 14598-5, ISO 15504 buscando identificar los aspectos presentes en estos modelos y que son considerados de importancia en el modelo de calidad sistémica. El alcance de este estándar internacional define seis características que describen la calidad del software: funcionalidad, fiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad.

Un Ambiente Integrado de Desarrollo es un software que provee todas las facilidades para que un programador, a través de él, pueda desarrollar otros programas. Pero no todos los ambientes están adecuados para el desarrollo de un tipo de software en particular, así surge la necesidad de evaluar la calidad de los mismos.

Lo que proponemos en el desarrollo de la tesis es plantear un Modelo de Calidad que contiene una serie de indicadores que permiten decidir si un determinado Ambiente Integrado de Desarrollo (AIDD) es de calidad, y si se ajusta a las necesidades del programador.

Una de las principales bondades de la metodología formulada, es que será aplicable a cualquier ambiente integrado de desarrollo, el cual no está limitado a los ambientes utilizados para elaborar esta tesis, sino que puede ser utilizado para medir la calidad de otros ambientes Integrados de Desarrollo. [WWW 04 020]

4.1 PARÁMETROS DE EVALUACIÓN DE LOS LENGUAJES.

Siguiendo el estándar Internacional ISO 14598, las fases del Proceso de Evaluación de Ambientes Integrados de Desarrollo que fueron utilizadas son:

Tabla 4.1 Proceso de Evaluación

Establecer los requerimientos de la evaluación	Funcionabilidad, Usabilidad, Eficiencia, Mantenimiento y Portabilidad.
Establecer los sub-criterios de la evaluación	Acceso a Base de Datos Componentes de Integración de Aplicaciones Soporte N-Capas Sistema Operativo Facilidades de Uso y Programación Campo de Aplicación Principal Orientación a Objetos Calidad de Documentación e Instalación Mantenimiento

	Costo y Soporte.
Establecer rangos para la evaluación.	Indicadores establecidos entre un rango de evaluación de 1 a 3.
Diseñar la evaluación	Unificación de los rangos, Presencia de las métricas, Indicadores de evaluación.
Evaluación de los ambientes Integrados de Desarrollo	Evaluación de los 4 Ambientes Integrados de Desarrollo.

[WWW 04 021]

4.1.1 ESTABLECER LOS REQUERIMIENTOS DE LA EVALUACIÓN:

El propósito de la evaluación es formular los criterios de selección para decidir si un Ambiente Integrado de Desarrollo es de calidad; es decir, si apoya la generación de las funcionalidades e implementa las tecnologías que deben contener los AIDD.

En el modelo de calidad se propone los elementos que lo conforman como se ve a continuación.

FUNCIONALIDAD. Un Ambiente Integrado de Desarrollo debe apoyar un conjunto de funcionalidades básicas: integración suficiente con los sistemas existentes que incluyen bases de datos, disposición de una

administración central; confiabilidad en las transacciones, posibilitar un acceso rápido y fiable a la información; ser eficientes; ser portables, deben funcionar sobre distintas plataformas, ser capaces de manejar grandes cantidades de tráfico, para lo cual es necesario soportar múltiples CPU.

USABILIDAD. Un ambiente de desarrollo debe reducir el esfuerzo necesario para su uso; es decir, que los desarrolladores de software disminuyan dramáticamente la curva de aprendizaje y de esta manera incrementen la productividad.

EFICIENCIA. Un ambiente de desarrollo debe ser eficiente en cuanto al uso de los recursos y, en cuanto al tiempo de respuesta de sus operaciones. La eficiencia en el uso de los recursos es vital para el desarrollo de aplicaciones, porque una de las funcionalidades que debe estar presente en un Ambiente de Desarrollo es la integración con sistemas ya existentes, base de datos, etc. Por esta razón la utilización óptima de los recursos será determinante para el manejo de un tiempo razonablemente corto.

MANTENIBILIDAD. El Ambiente Integrado de Desarrollo debe contar con el soporte del proveedor habilidad de las herramientas para hacer cambios , actualizaciones y expansibilidad.

PORTABILIDAD. El Ambiente Integrado de Desarrollo debe poseer portabilidad a diferentes plataformas de hardware, compatibilidad con

diferentes sistemas operativos, habilidad para mover datos entre versiones de la herramienta.

4.1.2 ESTABLECER LOS SUB-CRITERIOS DE LA EVALUACIÓN

Los sub-criterios de evaluación de los Ambientes Integrados de Desarrollo serán expresados en términos de las sub-características de calidad ISO 9126, entre los cuales se analizará los siguientes:

- ***Acceso a Bases de Datos.-*** Mediante una serie de módulos adicionales, un lenguaje puede tener acceso a las bases de datos, sean estas por los famosos ODBC o mediante sus propias herramientas primitivas de enlace. En la actualidad casi todos los lenguajes poseen estas herramientas que hacen más fácil el acceso a las bases de datos. Los lenguajes que vamos a estudiar deben tener también herramientas que accedan a bases vía WEB.

Tabla 4.2 Subcriterios de Acceso a BDD

CLASIFICACION	SUB-CRITERIOS
Acceso a Bases de Datos	Flexibilidad en el acceso a mayor número de Bases de Datos

	Acceso a Bases de Datos para Linux
	Acceso a Bases de Datos para Windows
	Permite la conexión con Bases de Datos por medio de ODBC
	Permite la conexión con Bases de Datos por medio de JDBC
	La configuración con Bases de Datos es de fácil manejo
	Conserva la integridad de grandes cantidades de información
	Posibilidad de Generar Database Wizard
	Presenta problemas de independencia en el manejo de los datos

- ***Componentes de Integración de aplicaciones.-*** La facilidad que pueden presentar los lenguajes de estudio para poder crear subprogramas o las llamadas librerías y poder utilizarlas sin complejidad. Las aplicaciones modernas están compuestas por multitud de pequeños programas que deben comunicarse los unos con los otros para que el usuario tenga la sensación de estar trabajando en un entorno coherente, un entorno en el que todo encaja y trabaja de forma coordinada. Para conseguir esta integración y la posibilidad de controlar las aplicaciones de esta forma es necesario un sistema que permita a los diferentes programas comunicarse entre ellos.

Tabla 4.3 Sub-criterios de Componentes de Integración de Aplicaciones.

CLASIFICACION	SUB- CRITERIOS.
Componentes de Integración de aplicaciones	Cuenta con componentes de integración de aplicaciones
	Grado de satisfacción de los componentes de integración
	Comparte datos con aplicaciones internas
	Posee componentes para llamadas a librerías externas

	Permite la integración con otras herramientas
	Permite el uso de Lenguaje Java/Java Script
	Permite el uso de Lenguaje C++
	Permite crear componentes COM, CORBA, DCOM, EJB
	Permite crear y utilizar controles Active X
	Utiliza componentes que pertenezcan a la lógica heredada

- ***Soporte N-Capas .-*** _La abstracción del lenguaje se refleja en las capas en las cuales se puede programar teniendo en cuenta desde el lenguaje ensamblador.

Tabla 4.4 Sub-criterios de Soporte N-Capas.

CLASIFICACION	SUB-CRITRIOS
Soporte N-Capas	Permite crear aplicaciones en dos capas
	Permite crear aplicaciones en 3 capas

- ***Sistemas Operativos.-*** Es sumamente importante lograr la ejecución de un lenguaje en diferentes plataformas o Sistemas Operativos,

pues en de esta manera se garantiza su usabilidad y portabilidad, sin incurrir en más gastos o intalaciones extras.

Tabla 4.5 Sub-criterios de Sistemas Operativos.

CLASIFICACION	SUB-CRITERIOS
Sistemas Operativos	Número de sistemas operativos que soporta
	Nivel de Integración con el Sistema Operativo.

- ***Facilidades de uso y programación.-*** Se analiza las estructuras del lenguaje, entorno de programación, editor, intérprete de comandos, que sean comprensibles. Facilidad de depurar, editar, compilar. Es importante también tomar en cuenta el tiempo de ejecución y compilación.

Se analizará también la ahora utilizada "ingeniería inversa " La ingeniería inversa del software es el proceso consistente en analizar un programa en un esfuerzo por crear una representación del programa con un nivel de abstracción más elevado que el código fuente.

Una de las necesidades importante de los sistemas de información corporativos de los últimos años es la de racionalizar el uso de sus

aplicaciones y de las plataformas físicas que la soportan, permitiendo la utilización de sistemas más pequeños, más baratos y que a pesar de todo, mucho más potentes que los anteriores grandes ordenadores. Entonces el término UPSIZING, analizado también en este parámetro, se utiliza para designar la integración de aplicaciones y ordenadores aislados en entornos de red, de forma que permita la compartición de datos. Un ejemplo sería la integración de bases de datos aisladas en un entorno cliente/servidor, con un potente servidor de bases da datos.

Tabla 4.6 Sub-criterios de Facilidades de uso y programación.

CLASIFICACION	SUB-CRITERIOS
Facilidades de uso y programación.	El tipo de editor es gráfico
	Posee editor propio de aplicaciones
	El visor del diseñador de módulo de datos es en árbol
	Permite el proceso de compilación en línea
	Permite el proceso de compilación Background Multi-hilo
	Posibilidad de Upsizing
	Posee facilidad de ingeniería inversa y decompilación
	Posibilidad de programación multihilo

	Presenta una sintaxis de programación generalizada
	Fácil de entender y aplicar
	El tiempo para aprender el lenguaje es corto
	Tiene un mejor desarrollo como aplicación API
	Tiene un mejor desarrollo como aplicación CGI
	Permite crear y ejecutar programas autónomos
	Posee librerías que simplifican la programación

- ***Campo de aplicación principal.-*** Existen lenguajes integrados específicos para determinadas aplicaciones lo cual debe ser analizado para no subestimar ni sobrestimar el lenguaje a elegir. Aunque en la actualidad los lenguajes son de propósito general.

  Determinaremos también un punto importante en la evaluación de estos lenguajes que es el Desarrollo de Aplicaciones Empresariales de Misión Crítica con lo que nos referimos al desarrollo y ejecución de aplicaciones en el Internet, Extranets e Intranet, las más

frecuentes son los Servidores de Internet de acceso a Internet (ISP), Servicios de Integración y Negocios On Line.

Tabla 4.7 Sub-criterios Campo de Aplicación Principal.

CLASIFICACION	SUB-CRITERIOS
Campo de aplicación principal	Se aplica en el desarrollo de aplicaciones empresariales de misión crítica
	Se puede utilizar en el desarrollo de aplicaciones de cualquier campo
	Se puede Trabaja con servidores de páginas Web
	Se puede desarrollar aplicaciones cliente/servidor
	Es compatible para desarrollar aplicaciones para Intranet e Internet
	Incluye herramientas para aplicaciones Web

- ***Orientación a objetos*.** Actualmente una de las áreas mayor actualidad en la informática y en el ámbito académico es la orientación a objetos. La orientación a objetos promete mejoras de amplio alcance en la forma de diseño, desarrollo y mantenimiento

del software ofreciendo una solución a largo plazo a los problemas y preocupaciones que han existido desde el comienzo en el desarrollo de software: la falta de portabilidad del código y reusabilidad, código que es difícil de modificar, ciclos de desarrollo largos y técnicas de codificación no intuitivas.

Un lenguaje orientado a objetos ataca estos problemas. Tiene tres características básicas: debe estar basado en objetos, basado en clases y capaz de tener herencia de clases. Muchos lenguajes cumplen uno o dos de estos puntos; muchos menos cumplen los tres. La barrera más difícil de sortear es usualmente la herencia. Los lenguajes de estudio serán analizados según estas "características básicas".

Tabla 4.8 Sub-criterios de Orientación a Objetos

CLASIFICACION	SUB-CRITERIOS
Orientación a objetos	Tiene como base de programación exclusiva la Orientación a objetos
	Los objetos poseen una serie de operaciones asociadas a ellos
	Permite construir una interfaz orientada a objetos sobre el sistema de BDD relacional
	Soporta ocultación de datos

- ***Calidad de documentación e Instalación .-*** Un lenguaje de programación indica su funcionabilidad y eficiencia en la calidad de documentación y la ayuda que presta al desarrollador. De igual forma la instalación debe ser estándar y fácil de aplicar y totalmente orientada.

Tabla 4.9 Sub-criterios de Calidad de documentación e instalación.

CLASIFICACION	SUB-CRITERIOS
Calidad de Documentación e Instalación	Posee versiones de ayuda instaladas
	Provee un sitio reconocido para ayuda en línea
	Existen manuales o tutoriales publicados en español
	Existe información suficiente en el Internet
	Existen medios de aprendizaje gratuitos
	Es fácil de instalar
	La instalación incluye todas las herramientas necesarias
	Existe información técnica del producto

- ***Mantenimiento, Costo y Soporte.-*** Se indicaría qué tipos de ayuda existe para el programador, licencias y subsidiarias para América y nuestro país.

  El factor económico es por lo general un factor que influye decisivamente en la elección de un determinado lenguaje de aplicación, especialmente en nuestro medio y por más que un lenguaje cumpla con los requerimientos técnicos y funcionales es relevante su costo por solución.

  Para la adquisición de un lenguaje es muy importante determinar que tiempo se ha encontrado ya en el mercado, especialmente en el campo de aplicación para la que le hemos elegido y en el tiempo entre versiones es conveniente tomar en cuenta pues puede que la versión que adquirimos ya este en poco tiempo discontinuada o haya que adquirir la actualización o peor tengamos que realizar varios parches para poder utilizarla. [WWW 04 022]

Tabla 4.10 Sub-criterios Mantenimiento, costo y soporte.

CLASIFICACION	SUB-CRITERIOS
Mantenimiento Costo y Soporte	Tiene respaldo de permanencia en el mercado

	Distribuye licencias gratuitas o demos
	El costo por licencia es alto, medio, bajo
	Existe un representante en nuestro país (Ecuador)
	Tiene un esquema licencias por solución informática
	Existe soporte por e-mail
	Existe representantes en Ecuador para soporte técnico personalizado
	El costo por mantenimiento alto, medio, bajo
	El costo por asistencia técnica alto, medio, bajo.
	El costo por actualizaciones es alto, medio, bajo

4.1.3 ESTABLECER RANGOS PARA LA EVALUACIÓN.

CLASIFICACION	METRICAS	RANGOS DE EVALUACION

Acceso a Bases de Datos	Flexibilidad en el acceso a mayor número de Bases de Datos	E=3, B=2, R=1
	Acceso a Bases de Datos para Linux	Si =3 puntos No =0 puntos
	Acceso a Bases de Datos para Windows	Si =3 puntos No =0 puntos
	Permite la conexión con Bases de Datos por medio de ODBC	Si =3 puntos No =0 puntos
	Permite la conexión con Bases de Datos por medio de JDBC	Si =3 puntos No =0 puntos
	La configuración con Bases de Datos es de fácil manejo	Si =3 puntos No =0 puntos

	Conserva la integridad de grandes cantidades de información	E = 3 B = 2 R = 1
	Posibilidad de Generar Database Wizard	Si =3 puntos No =0 puntos
	Presenta problemas de independencia en el manejo de los datos	Si =3 puntos No =0 puntos
Componentes de Integración de aplicaciones	Cuenta con componentes de integración de aplicaciones	Si =3 puntos No =0 puntos
	Grado de satisfacción de los componentes de integración	100% – 71% =3 pts. 70% - 41% =2 pts. 40% - 20% = 1 pts.
	Comparte datos con aplicaciones internas	Si =3 puntos No =0 puntos
	Posee componentes para llamadas a librerías externas	Si =3 puntos No =0 puntos

	Permite la integración con otras herramientas	Si =3 puntos No =0 puntos
	Permite el uso de Lenguaje Java/Java Script	Si =3 puntos No =0 puntos
	Permite el uso de Lenguaje C++	Si =3 puntos No =0 puntos
	Permite crear componentes COM, CORBA, DCOM, EJB	Si =3 puntos No =0 puntos
	Permite crear y utilizar controles Active X	Si =3 puntos No =0 puntos
	Utiliza componentes que pertenezcan a la lógica heredada	Si =3 puntos No =0 puntos
Soporte N-Capas	Permite crear aplicaciones en dos capas	Si =3 puntos No =0 puntos
	Permite crear aplicaciones en 3 capas	Si =3 puntos No =0 puntos
Sistemas Operativos	Numero de sistemas operativos que soporta	**>=5** =3 puntos **3=** 2 puntos <**2**=1 punto

	Grado de satisfacción con el soporte de sistema operativo	100% – 71% =3 pts. 70% - 41% =2 pts. 40% - 20% = 1 pts.
Facilidades de uso y programación	El tipo de editor es gráfico	Si =3 puntos No =0 puntos
	Posee editor propio de aplicaciones	Si =3 puntos No =0 puntos
	El visor del diseñador de módulo de datos es en árbol	Si =3 puntos No =0 puntos
	Permite el proceso de compilación en línea	Si =3 puntos No =0 puntos
	Permite el proceso de compilación Background Multi-hilo	Si =3 puntos No =0 puntos
	Posibilidad de Upsizing	Si =3 puntos No =0 puntos
	Posee facilidad de ingeniería inversa y decompilación	Si =3 puntos No =0 puntos
	Posibilidad de programación multihilo	Si =3 puntos No =0 puntos

	Presenta una sintaxis de programación generalizada	Si =3 puntos No =0 puntos
	Fácil de entender y aplicar	Si =3 puntos No =0 puntos
	El tiempo para aprender el lenguaje es corto	Si =3 puntos No =0 puntos
	Tiene un mejor desarrollo como aplicación API	Si =3 puntos No =0 puntos
	Tiene un mejor desarrollo como aplicación CGI	Si =3 puntos No =0 puntos
	Permite crear y ejecutar programas autónomos	Si =3 puntos No =0 puntos
	Posee librerías que simplifican la programación	Si =3 puntos No =0 puntos
Campo de aplicación principal	Se aplica en el desarrollo de aplicaciones empresariales de misión crítica	Si =3 puntos No =0 puntos
	Se puede utilizar en el desarrollo de aplicaciones de cualquier campo	Si =3 puntos No =0 puntos
	Se puede Trabaja con servidores de páginas Web	Si =3 puntos No =0 puntos
	Se puede desarrollar aplicaciones cliente/servidor	Si =3 puntos No =0 puntos

	Es compatible para desarrollar aplicaciones para Intranet e Internet	Si =3 puntos No =0 puntos
	Incluye herramientas para aplicaciones Web	Si =3 puntos No =0 puntos
Orientación a objetos	Tiene como base de programación exclusiva la Orientación a objetos	Si =3 puntos No =0 puntos
	Los objetos poseen una serie de operaciones asociadas a ellos	Si =3 puntos No =0 puntos
	Permite construir una interfaz orientada a objetos sobre el sistema de BDD relacional	Si =3 puntos No =0 puntos
	Soporta ocultación de datos	Si =3 puntos No =0 puntos
Calidad de Documentación e Instalación	Posee versiones de ayuda instaladas	Si =3 puntos No =0 puntos
	Provee un sitio reconocido para ayuda en línea	Si =3 puntos No =0 puntos
	Existen manuales o tutoriales publicados en español	Si =3 puntos No =0 puntos
	Existe información suficiente en el Internet	Si =3 puntos No =0 puntos

	Existen medios de aprendizaje gratuitos	Si =3 puntos No =0 puntos
	Es fácil de instalar	Si =3 puntos No =0 puntos
	La instalación incluye todas las herramientas necesarias	Si =3 puntos No =0 puntos
	Existe información técnica del producto	Si =3 puntos No =0 puntos
Mantenimiento Costo y Soporte	Tiene respaldo de permanencia en el mercado	>=5 años 3 pts 4 –2 años 2 pts. 1 año 1 pt.
	Distribuye licencias gratuitas o demos	Si =3 puntos No =0 puntos
	El costo por licencia es alto, medio, bajo	Ba = 3 pts. M = 2pts. A = 1 pts.
	Existe un representante en nuestro país (Ecuador)	Si =3 puntos No =0 puntos
	Tiene un esquema licencias por solución informática	Si =3 puntos No =0 puntos
	Existe soporte por e-mail	Si =3 puntos No =0 puntos

	Existe representantes en Ecuador para soporte técnico personalizado	Si =3 puntos No =0 puntos
	El costo por mantenimiento alto, medio, bajo	Ba = 3 pts. M = 2pts. A = 1 pts.
	El costo por asistencia técnica alto, medio, bajo.	Ba = 3 pts. Medio = 2pts. Alto = 1 pts.
	El costo por actualizaciones es alto, medio, bajo	Ba = 3 pts. M = 2pts. A = 1 pts.

4.1.4 DISEÑAR LA EVALUACIÓN

Una vez que se han formulado los rangos de evaluación para las métricas, se diseñará un plan de evaluación que consistirá en definir la aplicabilidad de los rangos propuestos. Los pasos que se siguieron para la aplicación del método son los siguientes:

Unificación de los rangos: se unificaron los valores de los rangos mediante la conversión a una escala común para todos los rangos propuestos a una escala del 1 al 3.

Presencia de las métricas: Se observo el valor de cada métrica si es superior o igual a dos la métrica está presente si es igual a uno o menor la métrica no está presente.

Los indicadores para la evaluación de un ambiente Integrado de desarrollo será el siguiente:

Rango de evaluación	**Evaluación del ambiente**	**Nomenclatura**	**Evaluación de costo**	**Nomenclatura**
3 puntos	Excelente	E	Bajo	Ba
2 puntos	Bueno	B	Medio	M
1 punto	Regular	R	Alto	A
-	No recomendable	N		

[WWW 04 023]

4.1.5 EVALUACIÓN DE LOS AMBIENTES INTEGRADOS DE DESARROLLO

Una vez tomados los valores de las métricas el siguiente paso es llevar a cabo la evaluación de los ambientes según los parámetros descritos en el plan de evaluación y aplicando los rangos propuestos.

En las siguientes tablas se presenta la calificación de la evaluación de los ambientes.

CLASIFICACION: ACCESO A BASES DE DATOS				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Flexibilidad en el acceso a mayor número de Bases de Datos	B	R	E	E
Acceso a Bases de Datos para Linux	SI	-	SI	SI
Acceso a Bases de Datos para Windows	SI	SI	SI	SI
Permite la conexión con Bases de Datos por medio de ODBC	SI	SI	SI	SI

Permite la conexión con Bases de Datos por medio de JDBC	-	-	SI	-
La configuración con Bases de Datos es de fácil manejo	SI	SI	SI	NO
Conserva la integridad de grandes cantidades de información	E	E	E	E
Posibilidad de Generar Database Wizard	SI	-	-	-
Presenta problemas de independencia en el manejo de los datos	NO	NO	NO	NO
TOTAL CUANTIFICADO:	20	13	21	15

PORCENTAJE	74.04	48.15	77.78	55.56

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 27 puntos

27 puntos = 100%

CLASIFICACION: COMPONENTES DE INTEGRACIÓN DE APLICACIONES				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Cuenta con componentes de integración de aplicaciones	SI	NO	SI	SI
Grado de satisfacción de los componentes de integración	50%	50%	90%	80%
Comparte datos con aplicaciones internas	SI	SI	SI	SI
Posee componentes para llamadas a librerías externas	NO	NO	SI	SI
Permite la integración con otras herramientas	SI	-	SI	-
Permite el uso de Lenguaje Java/Java Script	NO	SI	SI	NO
Permite el uso de Lenguaje C++	SI	SI	SI	SI

Permite crear componentes COM, CORBA, DCOM, EJB	SI	NO	SI	SI
Permite crear y utilizar controles Active X	SI	-	SI	SI
Utiliza componentes que pertenezcan a la lógica heredada	SI	SI	SI	SI
TOTAL CUANTIFICADO:	23	14	30	24
PORCENTAJE	76.66	46.66	100	80

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 30 puntos

30 puntos = 100%

CLASIFICACION: SOPORTE N-CAPAS				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Permite crear aplicaciones en dos capas	SI	NO	SI	SI
Permite crear aplicaciones en 3 capas	SI	-	SI	SI
TOTAL CUANTIFICADO:	6	0	6	6
PORCENTAJE	100	0	100	100

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 6 puntos

6 puntos = 100%

CLASIFICACION: SISTEMAS OPERATIVOS				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Numero de sistemas operativos que soporta	3	1	4	4
Grado de satisfacción con el soporte de sistema operativo	75%	60%	90%	90%
TOTAL CUANTIFICADO:	5	3	6	6
PORCENTAJE	83.33	50	100	100

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 6 puntos
6 puntos = 100%

CLASIFICACION: FACILIDADES DE USO Y PROGRAMACION				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
El tipo de editor es gráfico	SI	SI	NO	NO
Posee editor propio de aplicaciones	SI	SI	NO	NO
El visor del diseñador de módulo de datos es en árbol	SI	NO	NO	NO
Permite el proceso de compilación en línea	SI	NO	SI	SI
Permite el proceso de compilación Background Multi-hilo	SI	-	NO	SI
Posibilidad de Upsizing	SI	NO	SI	SI
Posee facilidad de ingeniería inversa y decompilación	SI	NO	SI	SI
Posibilidad de programación multihilo	NO	-	SI	SI
Presenta una sintaxis de programación generalizada	SI	SI	SI	SI

Fácil de entender y aplicar	SI	SI	SI	NO
El tiempo para aprender el lenguaje es corto	SI	NO	SI	NO
Tiene un mejor desarrollo como aplicación API	SI	NO	SI	SI
Tiene un mejor desarrollo como aplicación CGI	SI	NO	SI	NO
Permite crear y ejecutar programas autónomos	SI	-	SI	SI
Posee librerías que simplifican la programación	SI	SI	SI	SI
TOTAL CUANTIFICADO:	39	15	33	27
PORCENTAJE	86.66	33.33	73.33	60

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 45 puntos

45 puntos = 100%

CLASIFICACION: CAMPO DE APLICACIÓN PRINCIPAL				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Se aplica en el desarrollo de aplicaciones empresariales de misión crítica	SI	NO	SI	SI
Se puede utilizar en el desarrollo de aplicaciones de cualquier campo	SI	NO	SI	SI
Se puede Trabaja con servidores de páginas Web	SI	SI	SI	SI
Se puede desarrollar aplicaciones cliente/servidor	SI	NO	SI	SI
Es compatible para desarrollar aplicaciones para Intranet e Internet	SI	-	SI	SI
Incluye herramientas para aplicaciones Web	SI	SI	SI	SI
TOTAL CUANTIFICADO:	18	6	18	18
PORCENTAJE	100	33.33	100	100

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 18 puntos

18 puntos = 100%

CLASIFICACION: ORIENTACIÓN A OBJETOS

PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Tiene como base de programación exclusiva la Orientación a objetos	SI	NO	SI	SI
Los objetos poseen una serie de operaciones asociadas a ellos	SI	NO	SI	SI
Permite construir una interfaz orientada a objetos sobre el sistema de BDD relacional	SI	NO	SI	SI
Soporta ocultación de datos	SI	NO	SI	SI
TOTAL CUANTIFICADO:	12	0	12	12
PORCENTAJE	100	0	100	100

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 12 puntos

12 puntos = 100%

CLASIFICACION: CALIDAD DE DOCUMENTACIÓN E INSTALACION				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Posee versiones de ayuda instaladas	SI	NO	SI	SI
Provee un sitio reconocido para ayuda en línea	NO	NO	SI	SI
Existen manuales o tutoriales publicados en español	SI	SI	SI	SI
Existe información suficiente en el Internet	NO	NO	SI	SI
Existen medios de aprendizaje gratuitos	-	-	SI	SI
Es fácil de instalar	SI	SI	SI	SI
La instalación incluye todas las herramientas necesarias	NO	NO	SI	SI
Existe información técnica del producto	SI	NO	SI	SI
TOTAL CUANTIFICADO:	12	6	24	24
PORCENTAJE	50	25	100	100

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

Evaluación sobre 24 puntos

24 puntos = 100%

CLASIFICACION: MANTENIMIENTO COSTE Y SOPORTE				
PARAMETROS	**C++ BUILDER**	**VISUAL J++**	**JAVA**	**PERL**
Tiene respaldo de permanencia en el mercado	+5	3	+5	+5
Distribuye licencias gratuitas o demos	NO	NO	SI	SI
El costo por licencia es alto, medio, bajo	ALTO	MEDIO	BAJO	BAJO
Existe un representante en nuestro país (Ecuador)	NO	SI	NO	NO
Tiene un esquema licencias por solución informática	SI	-	SI	SI
Existe soporte por e-mail	SI	SI	SI	NO
Existe representantes en Ecuador para soporte técnico personalizado	NO	SI	NO	NO
El costo por mantenimiento alto, medio, bajo	ALTO	MEDIO	BAJO	BAJO
El costo por asistencia técnica alto, medio, bajo.	ALTO	MEDIO	BAJO	BAJO

El costo por actualizaciones es alto, medio, bajo	ALTO	-	BAJO	BAJO
TOTAL CUANTIFICADO:	13	17	24	21
PORCENTAJE	43.33	56.66	80	70

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

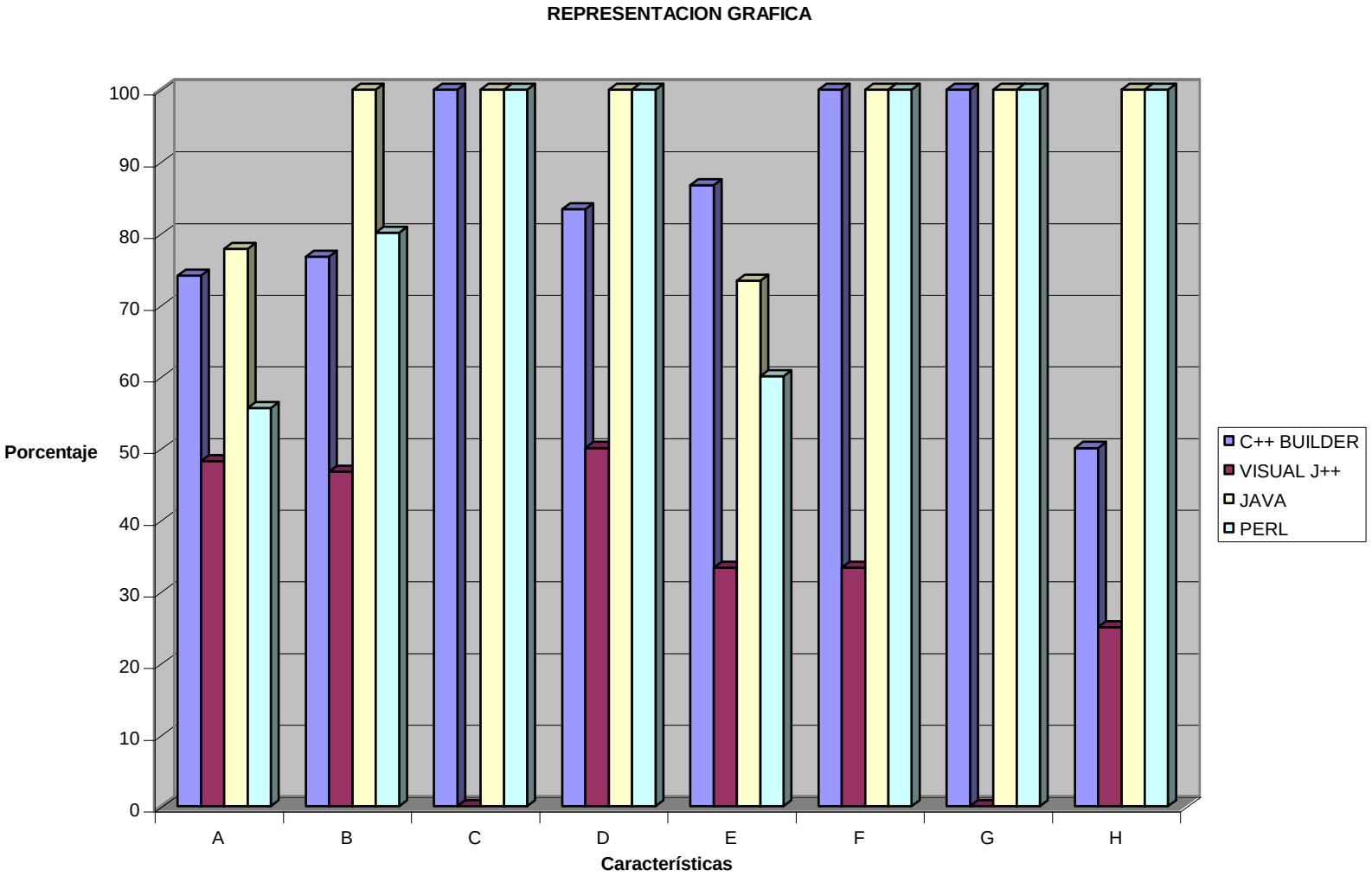
Evaluación sobre 30 puntos
30 puntos = 100%

4.2 RESULTADOS DE LA EVALUACIÓN

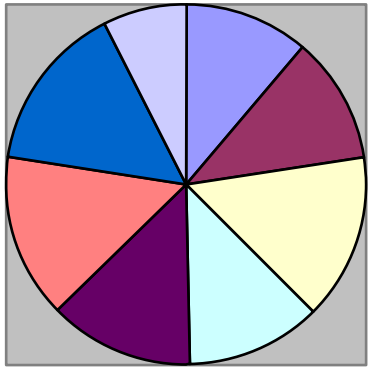
CARACTERÍSTICAS	PUNTAJE OBTENIDO			
	C++ BUILDER	VISUAL J++	JAVA	PERL
Acceso a Bases de Datos	20	13	21	15
Componentes de integración de Aplicaciones	23	14	30	24
Soporte n-Capas	6	0	6	6
Sistemas operativos	5	3	6	6
Facilidades de Uso y programación	39	15	33	27
Campo de aplicación principal	18	6	18	18
Orientación a objetos	12	0	12	12
Calidad de documentación e instalación	12	6	24	24
Mantenimiento coste y soporte	13	17	24	21
TOTAL	**148**	**74**	**174**	**153**
PORCENTAJE	**74.74**	**37.37**	**87.87**	**77.27**

E = Excelente **B** = Bueno **R** = Regular **N** = No recomendable

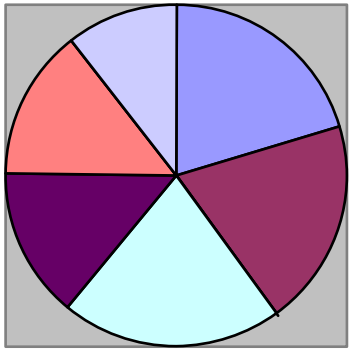
Representación Gráfica de los Resultados:



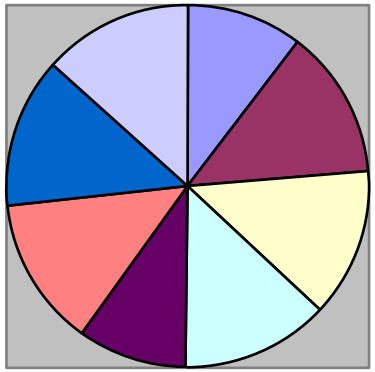
C++ BUILDER



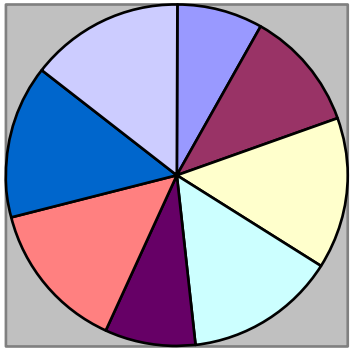
J++



JAVA



PERL



4.3 ANÁLISIS DE RESULTADOS

La evaluación de los lenguajes de estudio escogidos ha determinado, según nuestras necesidades al lenguaje Java como uno de los mejores, puede ser que en algunos parámetros no sea el más ideal pero se compensa en otros que en sí son los determinantes en la valoración que se le dio. Aunque uno de los motivos que nos hizo escoger este lenguaje es que en donde se va ha probar el aplicativo (G.P.I), prefirieron un lenguaje gratuito.

Seguidamente el C++ Builder es también la mejor opción, pero lamentablemente su costo de adquisición, capacitación y todo tipo de información es alto, lo que lo hace un lenguaje, en nuestro caso, inaccesible. El lenguaje C++ Builder es un lenguaje muy poderoso en lo que es el enlace de base de datos y por su edición totalmente gráfica se puede ahorrar mucho tiempo en diseño de pantallas y presentaciones.

Perl también es un lenguaje gratuito y ha cumplido muy satisfactoriamente con algunos parámetros pero se limita talvez en su entorno y librerías extra que puede tener. Es un lenguaje que está dando pasos gigantescos en lo que es facilidades de programación, por ejemplo existe las versiones para Windows con un editor gráfico para facilitar su utilización dentro de este ambiente.

Visual J++, además de no satisfacer muchos de los parámetros, prácticamente se encuentra desaparecido ya que Microsoft ha lanzado su máxima y actual

herramienta para desarrollo web el ya muy conocido y comercializado **.NET** que es un conjunto de tecnologías de software de Microsoft para la integración de software a través del uso de servicios Web XML. Además es un lenguaje muy limitado en su utilización pues solo se compila y ejecuta dentro de Windows.

4.4 ELECCIÓN DE LA HERRAMIENTA MÁS ÓPTIMA

Java ha sido el lenguaje que según la metodología de evaluación cumple con los parámetros que consideramos más relevantes. Esta medición la hemos realizado de acuerdo a los requerimientos y necesidades para el desarrollar el prototipo SISPER , quedando como el lenguaje escogido para la implementación de la aplicación.

Las características del lenguaje fueron presentadas anteriormente, se puede indicar además que con la base de datos escogida MYSQL y el servidor APACHE forma un conjunto de desarrollo FREEWARE que permitirá al programador acoplar sus requerimientos sin restricciones.

Además se puede indicar que los Servlets de Java son más eficientes, más poderosos, portables y fáciles de usar.

JAVA

HERRAMIENTA MAS OPTIMA

5. DESARROLLO DE LA APLICACIÓN PROTOTIPO SISPER

5.1 IDENTIFICACIÓN DEL PROBLEMA Y OBJETIVOS DEL PROTOTIPO

El Consejo Provincial de Imbabura es una entidad Gubernamental la cual está constituida por el Prefecto y 6 concejales; con ellos laboran 89 empleados y 122 trabajadores.

Cuentan con un sistema informático de registro de personal realizado en FoxPro ver 2.0, el cual se encuentra totalmente obsoleto y en desuso. Cuando las autoridades requieren un listado del personal existente el puesto que ocupa o cualquier información sobre el mismo, lo realizan manualmente y en un tiempo considerablemente lento.

Una Institución orientada al servicio de la comunidad requiere de eficiencia, modernización y buen trato al usuario: en consecuencia, la rapidez en la atención forma parte del buen servicio que estimula al usuario a sentirse satisfecho con la Institución, con lo que se beneficia enormemente tanto a la comunidad como a la entidad que presta su servicio.

OBJETIVOS:

- Realizar un sistema informático para el control de personal en el Consejo Provincial de Imbabura.

- Crear una interfaz amigable con el usuario de modo que permita el fácil manejo del sistema.

- Poder entregar información actualizada y oportuna mediante los diferentes reportes.

- Permitir ahorrar tiempo y espacio en las funciones que se realizan sobre control de personal.

- Entregar al usuario un sistema fácil de utilizar.

- Diseñar el sistema para presentar la información por medio de la web (intranet)

[LIB 05 001]

5.2 DESARROLLO DE TABLAS Y DIAGRAMAS E/R

Tablas que se utilizarán en el prototipo SISPER:

TABLAS	DESCRIPCIÓN
DIRECCIÓN	Se llevará la información de los Departamentos existentes en el Consejo Provincial de Imbabura, en total son 13 departamentos
SECCION	Los Departamentos se subdividen en secciones, en total son 20 secciones, aquí se guardan todas las secciones existentes.
UBICACIÓN	Se llevará la información de la ubicación de cada empleado.
GRADO	Los grados que se les asigna a cada empleado, depende de la dirección, sección, ubicación el grado que se le asigne
PROGRAMA	Se lleva la información del programa al que pertenece, existen 4 programas.
VACACION	Se lleva un control de las vacaciones asignadas a cada empleado, las mismas que se les asigna cuando hayan cumplido por lo menos 9 meses de trabajo.
LICENCIA	Se lleva un control de las licencias que haya solicitado el empleado.
PUESTO	El puesto que ocupe el empleado en el Consejo

	Provincial de Imbabura
CONTROL-ASIS	Se llevará un control de la hora de entrada-salida del empelado
EMPLEADOS	Se guardarán los datos personales de los empleados, el sueldo asignado, maquinaria o equipo que haya solicitado.
EQMAQ	Se lleva la información del empleado y que equipo o maquinaria ha solicitado
CARGAS_FAM	Se lleva la información de las cargas familiares dependientes del empleado
DESC_EQ_MAQ	Se lleva la información de la maquinaria y equipo existentes

DESCRIPCIÓN DE LOS CAMPOS DE LAS TABLAS:

Tabla: Dirección

Nombre	Descripcion	Tipo	P
Cod_dir	Codigo de la dirección	LongInteger	Yes
Nom_direc	Nombre de la direccion	Text(10)	No

Tabla: Sección

Nombre	Descripción	Tipo	P
Cod_sec	Codigo de la sección	LongInteger	Yes
Nom_sec	Nombre de la sección	Text(10)	No
Cod_dir	Código de la dirección	LongInteger	No
CI	Cédula de identidad del empleado	Text(10)	Yes

Tabla: Ubicación

Nombre	Descripción	Tipo	P
Cod_ubic	Codigo de la ubicación	LongInteger	Yes
Nom_cargo	Nombre del cargo	Text(10)	No
sueldot	Sueldo asignado al empleado	LongInteger	No

Tabla: Grado

Nombre	Descripción	Tipo	P
Cod_ubic	Código de la ubicación	LongInteger	Yes
Grado	Grado asignado al cargo	LongInteger	No
Categoria	Categoría asignada al cargo	LongInteger	No

Tabla: Programa

Nombre	Descripción	Tipo	P
Cod_prog	Código del programa	LongInteger	Yes
nom_prog	Nombre del programa	Text(10)	No

Tabla: Vacación

Nombre	Descripción	Tipo	P
CI	Cedula de identidad del empleado	Text(10)	Yes
Fecha_vac	Fecha en la que sale de vacaciones	DateTime	No
dias_vac	Dias que sale de vacaciones	LongInteger	No
dias_disp	Dias disponibles de vacaciones	LongInteger	No

Tabla: Licencia

Nombre	Descripción	Tipo	P
CI	Cédula de Identidad del empleado	Text(10)	Yes
Fecha_sal	Fecha de salida con licencia	DateTime	No
Fecha_ing	Fecha de ingreso de la licencia	DateTime	No
dias_licenci	Dias de licencia	LongInteger	No

Tabla: Puesto

Nombre	Descripción	Tipo	P
Cod_sec	Codigo Sección	LongInteger	Yes
CI	Cédula de identidad	Text(10)	Yes

Tabla: Control-Asis

Nombre	Descripción	Tipo	P
CI	Cédula de identidad del empleado	Text(10)	Yes
Hora_ent	Hora de entrada al trabajo	DateTime	No
Hora_sal	Hora de salida del trabajo	DateTime	No
Fecha	Fecha actual	DateTime	No
Horas_trab	Horas trabajadas	LongInteger	No
vacacion	Si se encuentra de vacaciones o con licencia	LongInteger	No

Tabla: Empleados

Nombre	Descripción	Tipo	P
CI	Cédula de identidad del empleado	Text(10)	Yes
Apellidos	Apellidos del empleado	Text(10)	No
Nombres	Nombres del empleado	Text(10)	No
Direccion	Domicilio del empleado	Text(10)	No
Fecha_nac	Fecha de nacimiento del empleado	DateTime	No
Telefono	Teléfono del empleado	LongInteger	No
Fecha_ing	Fecha de ingreso ala Institución	DateTime	No
Fecha_sal	Fecha de salida de la Institución	DateTime	No
Estado_civil	Estado Civil del empleado	Text(10)	No
Num_iess	Numero del IESS	LongInteger	No
Cargas_fam	Número de cargas familiares	LongInteger	No
Trab_otrasins	Si trabajó en otras instituciones	LongInteger	No
Asociacion	Si pertenece a la Asociación	Text(10)	No
Cargas_escol	Número de cargas escolares	LongInteger	No
tipo	Si es emp0leado o trabajador	Text(10)	No
Cod_prog	Código del programa al que pertenece	LongInteger	Yes
Cod_ubic	Código de la ubicación en la que se encuentra	LongInteger	Yes

Tabla: EqMaq

Nombre	Descripción	Tipo	P
CI	Cédula de identidad del empleado	Text(10)	Yes
cod_maq	Código de la maquinaria asignada	LongInteger	Yes

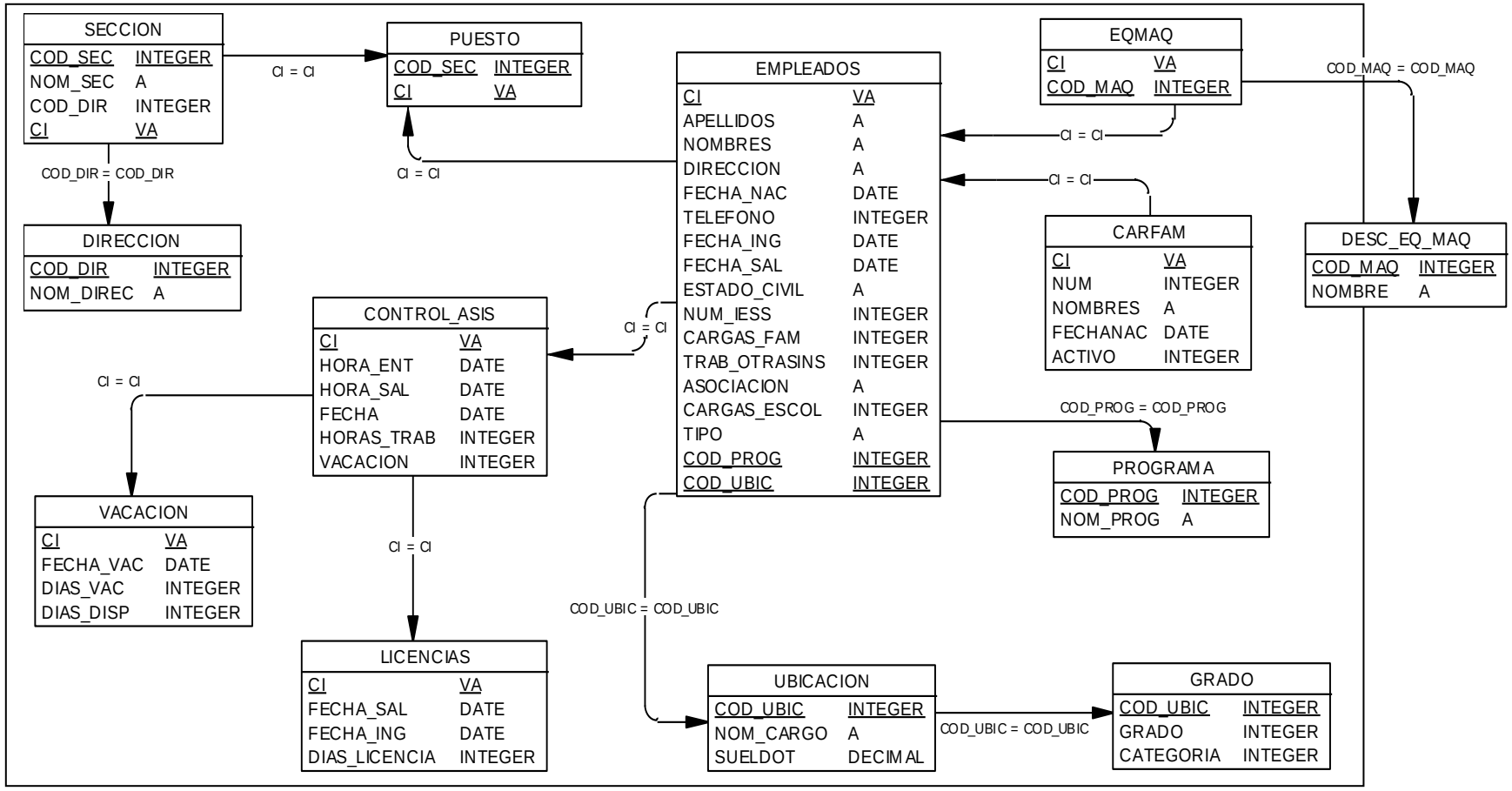
Tabla: Cargas_fam

Nombre	Descripción	Tipo	P
CI	Cédula de Identidad del empleado	Text(10)	Yes
Num	Número de hijos	LongInteger	No
Nombres	Nombres y apellidos de los hijos	Text(10)	No
Fechanac	Fecha de nacimiento de los hijos	DateTime	No
activo	Si el empleado se encuentra activo	LongInteger	No

Tabla: Desc_eq_maq

Nombre	Descripción	Tipo	P
cod_maq	Código de la maquinaria	LongInteger	Yes
nombre	Nombre de la maquinaria	Text(10)	No

Diagrama Entidad Relación (Modelo Gráfico)



5.3 DESARROLLO DEL PROTOTIPO SISPER PARA PLATAFORMA LINUX

Para el desarrollo del prototipo SYSPER sobre la plataforma Linux, se utilizó el siguiente software:

- Sistema Operativo RedHat ver. 7.0 o superior
- Jakarta-Tomcat 3.2.1
- Apache Server 1.3.2
- MySql 3.23.44
- JDK 1.2.3

5.4 DESARROLLO DEL PROTOTIPO SISPER PARA PLATAFORMA WINDOWS

Para el desarrollo del prototipo SYSPER sobre la plataforma Windows, se utilizó el siguiente software:

- Sistema Operativo windows 98 o Windows NT
- Jakarta-Tomcat 3.2.1
- Apache Server 1.3.2
- MySql 3.23.44
- MyODBC 5.50
- JDK 1.2.3

6. CONCLUSIONES Y RECOMENDACIONES

6.1 VERIFICACIÓN DE LA HIPÓTESIS

La aplicación de la metodología MAID2002 en los ambientes escogidos, nos lleva a verificar la hipótesis planteada:

"La utilización de una metodología de evaluación garantizará la elección de un ambiente integrado óptimo para el desarrollo de aplicaciones".

Al desarrollar el prototipo SISPER en el ambiente determinado como óptimo en el resultado de la metodología comprobamos la siguiente hipótesis:

"El uso de nuevos ambientes integrados de desarrollo de aplicaciones permitirá dar una alternativa más potente y amigable al programador, para solucionar los problemas inherentes a aplicaciones empresariales de misión crítica."

6.2. CONCLUSIONES.

- Al desarrollar la metodología de evaluación MAID2002 se vió la necesidad de plantearse, analizar y estudiar parámetros que consideramos necesarios para el desarrollo de una aplicación en determinado lenguaje.

- Para la aplicación de la metodología se ha escogido ambientes integrados no convencionales con la finalidad de comparar su funcionabilidad y efectividad con los lenguajes que usamos frecuentemente.

- Con la mayoría de Ambientes Integrados de Desarrollo que encontramos en la actualidad podemos desarrollar tecnología informática, basada en el poder de las redes y en la idea de que el mismo software debería ejecutarse en diferentes tipos de computadoras, dispositivos personales y otros equipos, por lo que ya no se debería hablar de ambientes integrados para plataforma Windows y ambientes integrados para sistema operativo Linux.

- Con respecto a la evaluación de los cuatro ambientes integrados propuestos hemos visto que no difiere mucho su funcionalidad ni recursos por lo que estuvieron los resultados estrechamente relacionados, con la excepción del lenguaje Visual J++ por su

virtual desaparición en el mercado de programación creando otros ambientes mucho más poderosos por Microsoft.

- Con la herramienta de comunicación la Internet y las redes se vuelven totalmente accesibles y los usuarios pueden hacer cosas antiguamente inimaginables, es toda una nueva forma de trabajar con computadoras. La comunicación es posible gracias a los ODBC, estudiados también para cada uno de los ambientes integrados de desarrollo escogidos.

- Con respecto al desarrollo del prototipo SISPER podemos indicar que es un inicio de lo que debería establecerse en una organización de la magnitud e importancia del Gobierno Provincial de Imbabura.

6.3. RECOMENDACIONES

- La información existente debe de encontrarse en un ciclo de depuración como mínimo cada año, ya que los datos personales del empleado o trabajador pueden cambiar, así como sus datos de ubicación y categorización dentro del CPI.

- Probar la metodología MAID2003 con otros Ambientes Integrados de Desarrollo u otro software para desarrollo, bases de datos, sistemas operativos y analizar resultados.

- Capacitación para todo el personal que se refiere a equipos de cómputo (hardware), utilitarios y sistemas internos del CPI. Igualmente la capacitación al personal que maneje es el sistema SISPER., también se dejará la información necesaria para que personal propio del CPI actualice y mantenga el Sistema.

- Generar una campaña para incentivar al personal para que pueda aceptar un cambio en su estructura, el cual debe darse en cada uno de los sistemas que disponen principalmente el sistema SISPER.

- Permitir se siga generando convenios de este tipo dando apertura a entidades universitarias y de formación para aportar con ideas y conocimientos al desarrollo integral de la institución.

- Aprobar una política de inversión en la adquisición de nueva tecnología, para poder dar una mejor imagen y servicio tanto a los mismos empleados como a la ciudadanía en general.

- Diseñar la página Web del Gobierno Provincial de Imbabura con la que se establecería el deseo de trabajar para la comunidad a la vez que da la posibilidad de interactuar directamente con esta.

REFERENCIAS

PAGINAS WEB

WWW 01 020 http://inei.gob.pe/cpi/bancopub/libfrce/lib615.htm

WWW 01 021 http://www.hsilver.com

WWW 01 022 http://www.servicioalpc.com/historia.html

WWW 01 023 http://fisica.uson.mx

WWW 01 024 http:// www.cs.buap.mx

WWW 01 025 http://w3.mor.itesm.mx/~lssaclced

WWW 01 026 http://www.miexamen.com.mx/Lenguajesprogramacion2.ht

WWW 01 027 m

WWW 01 028 http://www.arrakis.com

WWW 01 029 http://giportal.com

WWW 01 030 http://elies.rediris.es/elies9/index3.htm

http://www.monografias.com/trabajos5/tipbases/tipbases.sht

WWW 02 020 ml

WWW 02 021

WWW 02 022 http://www.jfactivesoft.com

WWW 02 023 http://www.arrakis.es/~rporcal

WWW 02 024 http://www.elguille.com/c++builder.html

WWW 02 025 http://www.microsoft.com/latam/ssafe/producto

WWW 02 026 http://www.ked.com.mx/capacitacion/temarios/curso1298.h

WWW 02 027 tml

WWW 02 028 http://www.edicurs.es/cursos/curso.asp

WWW 02 029	http://www.uco.es
WWW 02 030	http://www.perl.com
WWW 02 031	http://www.etsimo.uniovi.es/perl/tutor.html
WWW 02 032	http://www.cicei.cim/gsi/tutorial_perl/indice.html
WWW 02 033	http://www.iec.csic.es/criptonomicon/java/quesjava.html
WWW 02 034	http://www.fciencias.ens.uabc.mx/~mjava/java/carac.html
WWW 02 035	http://www.dis.eafit.edu.co/cursos/st829
WWW 02 036	http://www.lavariable.com/art/cgi/acgi003
	http://www.latiumsoftware.com/es/articles/00010.php3
WWW 03 020	http://www.elportaldelprogramador.com
WWW 03 021	http://www.cybercursos.com
WWW 03 022	
WWW 03 023	http://www.programacion.com/tutorial.jdbc.13.html
WWW 03 024	http://www.uv.es/~pedroj/presentjava.ppt
WWW 03 025	http://www.unisoft/site/cursos.htm
WWW 03 026	http://www.tu-chemnitz.de/urz/java/cjug/2/kap12.html
WWW 03 027	http://www..java.progrmacion.net
WWW 03 028	http://www.mmlabx.ua.es/instalacion/jdbc-instalacion.html
	http://www.sage.es/soporte/files/notas/odb/instalar.doc
WWW 04 020	http://www.geneura.urg.es
	http://www.uazuay.edu.ec/internet/manual.htm
WWW 04 021	
	http://www.issco.unige.ch/ewg95/node13.html
WWW 04 022	http://www.isaca.org.za/iso9126.htm
WWW 04 023	http://www.sztaki.hu/husei/events/SQAD/eload/

http://www.iso.ch/cate/d24902.html

http://www.dmi.vib.es/~bbuades/calidad/

http://www.get.cubaweb.cu/software_calidad.html

LIBROS

LIB 01 001 JOHANSON, MCHUGH, PENDLEBURY, WHEELER, Reingeniería de Procesos de Negocios

LIB 02 001 CHRIS HPAPAS, WILLIAM H. MURRAY, Manual de Borland C++

REVISTAS

REV 01 010 Revista Byte Pág. 121-122

REV 02 010 Revista SOLO PROGRAMADORES Pág. 32 Año III

ANEXO A

GLOSARIO

Active X

Lenguaje desarrollado por Microsoft con el fin de elaborar aplicaciones exportables a la red las cuales deben ser capaces de operar sobre cualquier plataforma a través de navegadores WWW de forma que le da dinamismo a las páginas web.

Administrador de Web (Webmaster)

Persona responsable de la gestión y mantenimiento de un servidor web, principalmente desde el punto de vista técnico; por lo que no debe ser confundido con un editor de web.

Apache

Servidor HTTP de dominio público el cual está basado en el sistema operativo Linux. Fue desarrollado en 1995 y actualmente es uno de los servidores HTTP más utilizados en la red.

Aplicación

Programa que lleva a cabo una función específica para un usuario en Internet tales como WWW, FTP, correo electrónico y Telnet.

Applet

Pequeña aplicación escrita en Java la cual se difunde a través de la red en orden de ejecutarse en el navegador cliente.

Archivo

Unidad significativa de información la cual puede ser manipulada por el sistema operativo de un ordenador debido a que tiene una identificación única formada por un "nombre" y un "apellido". El nombre suele ser de libre elección del usuario y el apellido debe identificar el contenido o el tipo de archivo. A manera de información, los archivos word tienen el apellido .doc; los de excel tienen .xls; los de texto .txt y así sucesivamente.

Browser:

Navegador u hojeador. Un cliente de World Wide Web. Una herramienta para obtención de información. Por ejemplo Mosaic, Netscape, Internet Explorer.

Cliente:

Programa ejecutado en la máquina local, que permite obtener información de Internet y el World Wide Web.

Clic

Situación en la cual se pulsa un determinado comando de un ratón una vez colocado el vínculo del mismo sobre una determinada área de la pantalla con el fin de dar una orden al ordenador.

Hardware:

Se refiere a la parte tangible de los equipos de computación.

Host:

Computadora lejana que presta servicio a uno o más usuarios.

HTML:

(HyperText Markup Language). Las reglas que gobiernan la creación de documentos que se pueden ver con un browser. La mayoría de los documentos desplegados en Mosaic y Netscape son documentos HTML.

HTTP:

(HyperText Transport Protocol). El protocolo de comunicación empleado por los servidores de WWW.

internet:

Conjunto cualquiera (aislado) de computadoras conectadas. (Con minúscula).

Internet:

Un conglomerado internacional de redes computacionales que conecta instituciones comerciales, gubernamentales y académicas. (Con mayúsculas).

ISO:

(International Standards Organization).

JAVA

Java es un lenguaje orientado a objetos y desarrollado por Sun Microsystem. Comparte similitudes con *C, C++* y *Objetive C*. Basándose en otros lenguajes

orientados al objeto, Java recoge lo mejor de todos ellos y elimina sus puntos más conflictivos.

LINUX:

Linux es una implementación independiente con *"espíritu"* POSIX (especificación para sistemas operativos). Tiene extensiones System V y BSD, y ha sido escrito completamente a base de aportaciones. Linux no tiene codigo propietario. Linux está distribuido libremente bajo "GNU Public License". Actualmente solo trabaja en IBM PC (o compatibles) y con arquitecturas ISA e EISA, y requiere un procesador 386 o superior.

PERL:

Perl es un lenguaje para manipular textos, ficheros y procesos. Perl proporciona una forma fácil y legible para realizar trabajos que normalmente se realizarían en C o en alguna Shells. Perl rueda en varios sistemas operativos y permite portar los fuentes a diferentes plataformas. No obstante, donde nació y donde más se ha difundido es bajo el sistema operativo UNIX.

Servidor:

Computadora que presta servicios a muchos usuarios.Provee información y programas a Internet.

URL:

(Uniform Resource Locator). La dirección de una fuente de información. Está compuesto por cuatro partes distintas:

el tipo de protocolo (http, ftp, gopher), el nombre de la máquina, la ruta del directorio y el nombre del archivo. Por ejemplo:

http://www.uca.edu.sv/cidai/pubel.html

World Wide Web = WWW = W3 = The Web:

Conjunto de información distribuida basada en hipertexto, (interfaz gráfica, atractiva y amigable) para un acceso a recursos de Internet, especialmente en el ámbito comercial concebida en el CERN, Ginebra.

ANEXO B

SISTEMA CLIENTE/SERVIDOR DE CONTROL DE PERSONAL

MANUAL DE USUARIO

INTRODUCCIÓN

La documentación de un Sistema Informático facilita la labor que determinada persona desea realizar dentro del mismo, facilitando en sí el manejo y utilización del Sistema.

Por lo expuesto anteriormente es indispensable dentro de las obligaciones básicas del programador el dar los manuales que contengan información verídica y clara sobre el uso de determinado sistema.

Se considera al presente documento como una herramienta de ayuda y apoyo para aquellos usuarios que deseen manejar el sistema SISPER, sean novatos o expertos, sencillamente entenderá su funcionamiento sin ingresar a leer el código fuente.

Requerimientos *Operacionales*

Para el correcto funcionamiento del presente sistema se hace necesario contar con los siguientes requerimientos mínimos:

- **Hardware**
  - Un PC compatible con IBM, mínimo Pentium III (Servidor)
  - Mínimo 128 MB en memoria RAM
  - Monitor
    - Teclado
    - Impresora de 80 columnas
    - Mouse

  - Un PC compatible con IBM, mínimo Pentium II (Clientes)
  - Mínimo 32 MB en memoria RAM mínimo.
  - Monitor
    - Teclado
    - Impresora de 80 columnas
    - Mouse

- **Software**
  - Sistema Operativo Windows 98 o superior.
  - Internet Explorer 5.0 o superior.
  - Proxy en el servidor.
  - Mysql instalado en el servidor.
  - El sistema propuesto como solución .

- **De Usuario**
  - Conocimientos básicos de operación de computadoras
  - Conocimiento de terminología técnica a emplear en el sistema.

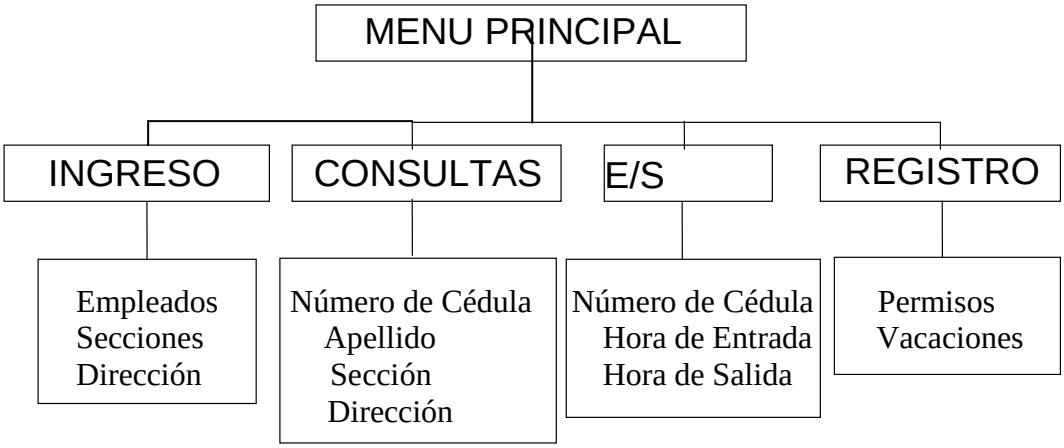
Pasos para Ejecutar el Sistema

Para poder ejecutar el presente sistema se sugiere seguir los siguientes pasos:

- Comprobar que todos los dispositivos estén correctamente conectados.
- Encender el CPU.
- Encender el monitor.
- Conectarse a la Intranet.
- Abrimos el Intermet Explorer y digitamos la dirección de localhost:8080

Estructura del Módulo del Sistema SISPER

Descripción de la Estructura del Sistema



Después de haber realizado el análisis del presente Sistema se llegó a determinar que el modelo antes mencionado en cuanto a presentación de opciones para el usuario es el adecuado. A continuación se especifica cada una de las opciones principales (opciones del menú principal) con respecto a su funcionalidad:

Pantalla de Acceso

PROPOSITO.- Esta pantalla es la primera que se visualiza al ingresar al sistema digitando la dirección de esta en el Internet Explorer:

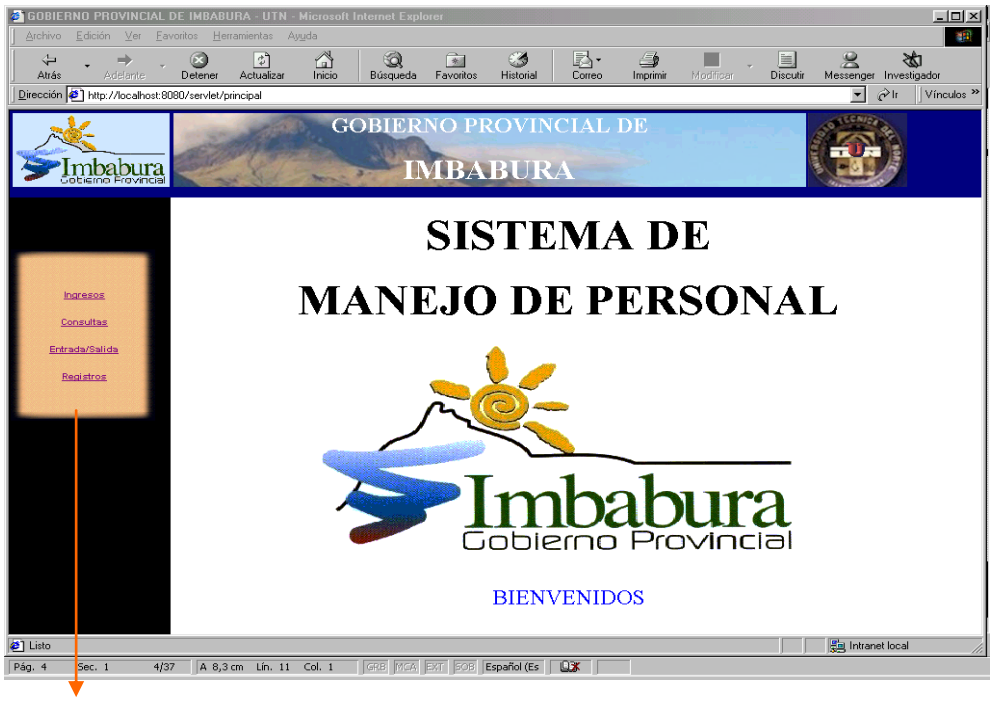
http://localhost:8080/index.htm

Debido a que el sistema esta diseñado tanto para el administrador como para los diferentes empleados, se ha limitado el acceso a las diferentes opciones por medio de la diferenciación de sesiones para cada USUARIO del sistema.

En esta pantalla se ingresa el nombre del usuario y la clave de acceso, es utilizada para restringir el acceso a los usuarios y poder mantener seguridades. estos datos serán entregados a cada Usuario.

Ingresados los datos anteriormente indicados hacemos clic en el botón Aceptar, si se ingresa la clave errónea se visualiza el mensaje: ERROR EN INGRESO DE CLAVE ..... HAGA CLIC EN ATRÁS. Cuando el usuario no ingresa clave se indica el mensaje INGRESE CONTRASEÑA..... HAGA CLIC EN ATRÁS.

Cuando el ingreso de la contraseña es correcto y se ha presionado el botón Aceptar se carga a la siguiente pantalla que es la **PANTALLA PRINCIPAL**. Esta pantalla se muestra diferente para cada usuarios, únicamente en el menú principal.

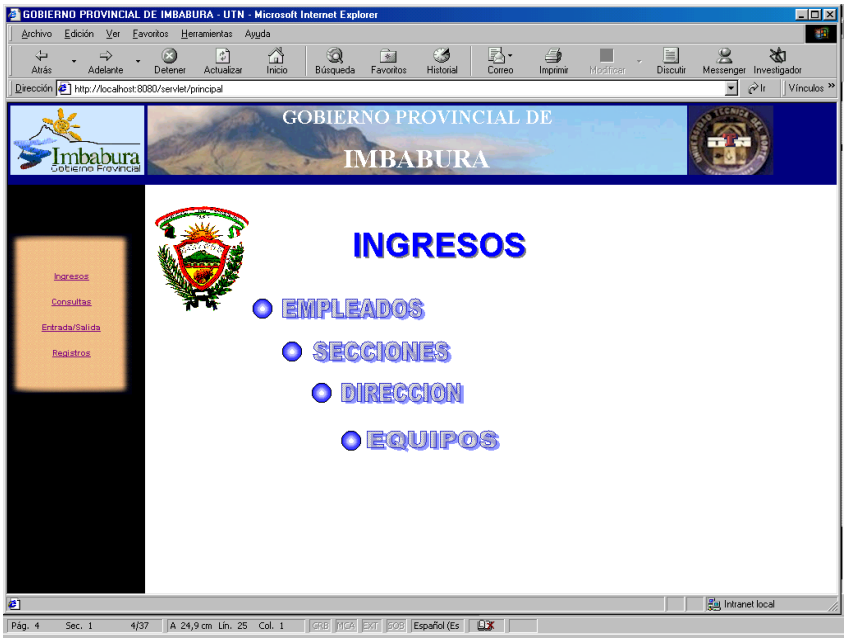


Menú Principal

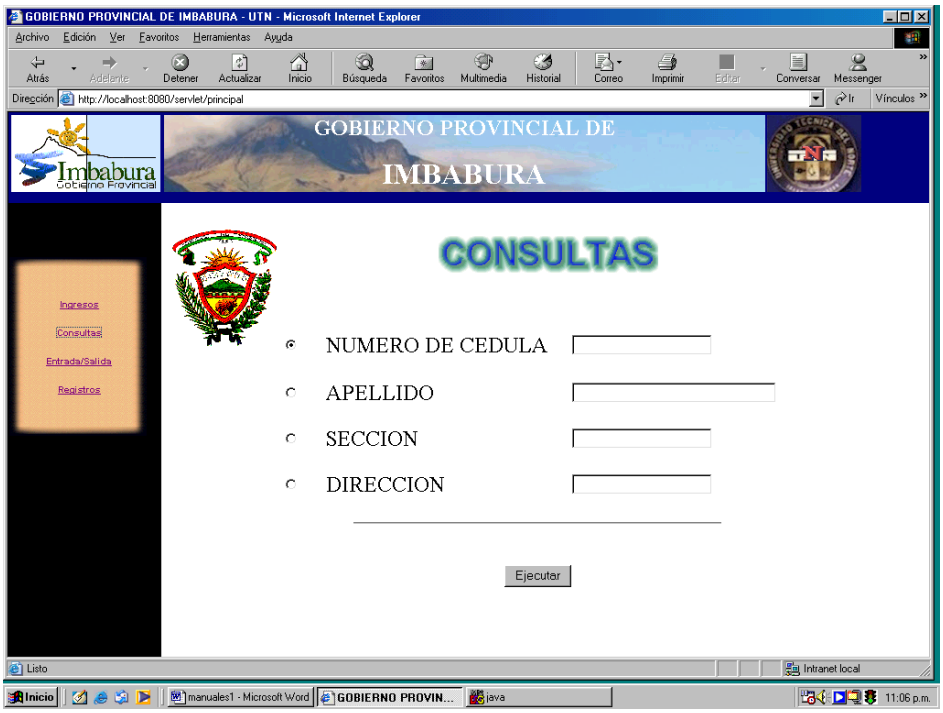
Menú Principal

La sección de menú principal permite acceder a los módulos del sistema para los cuales cada usuario tiene permiso de trabajar.

Ingresos.- Esta opción tiene la finalidad de permitir ingreso de los datos concernientes al personal laboral que forman parte del G.P.I. requeridos para su carpeta personal dentro de la nómina de la institución, agregar nuevas direcciones y secciones si estas fueran creadas, muestra la lista de equipos y agregar nuevos. Esta opción solo la tiene el administrador.



Consultas.- Permite realizar las consultas de los empleados sea esta por diferentes parámetros como número de cédula, apellidos, sección o dirección donde se encuentre ubicado. Esta opción la puede tener el director o jefe de personal. Sino ingresa ningún parámetro para la consulta saldrá el mensaje, DATO NULO....CLIC ATRÁS.

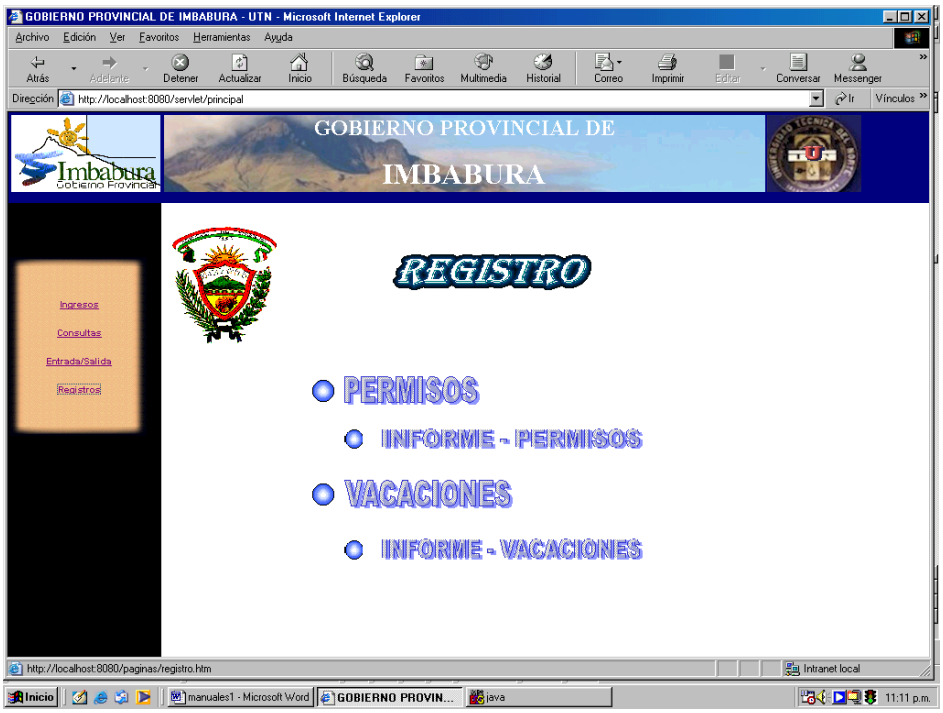


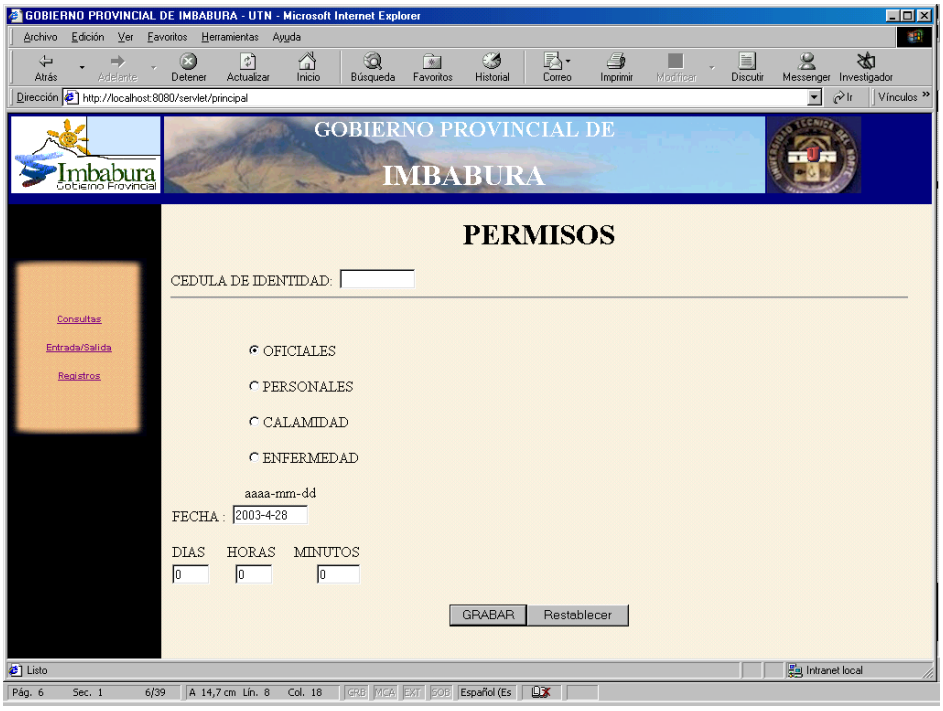
Entrada/Salida.

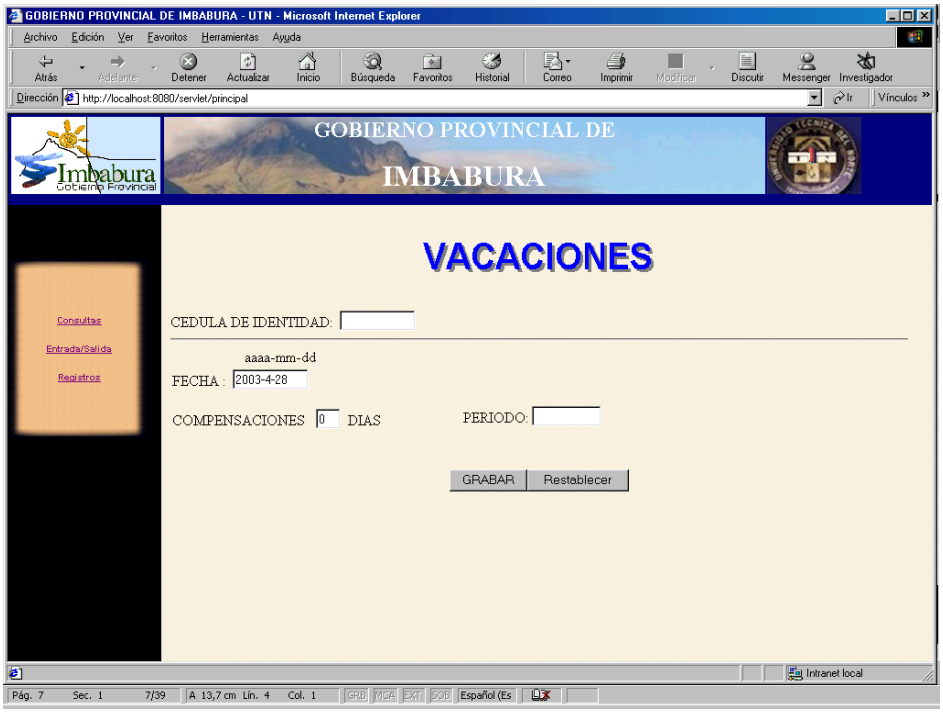
Esta es prácticamente la principal opción del presente sistema, pues aquí es donde se puede realizar el registro de la hora de entrada y salida del personal desde cualquier computador de la red, todos los usuarios tienen acceso a esta sesión del sistema. Se registra el dato de las entradas y salidas con el valor de la cédula de identidad.

Registros.

Nos permite llevar un control de los permisos y vacaciones de los empleados. Se lo realiza mediante el registro de estos eventos para poder posteriormente verificar la información. Tienen acceso a esta sesión el administrador del sistema y el jefe de personal.


GOBIERNO PROVINCIAL DE IMBABURA - UTN - Microsoft Internet Explorer
GOBIERNO PROVINCIAL DE
IMBABURA
Ingresos
Consultas
Entrada/Salida
Registros
REGISTRO
PERMISOS
INFORME - PERMISOS
VACACIONES
INFORME - VACACIONES


GOBIERNO PROVINCIAL DE IMBABURA - UTN - Microsoft Internet Explorer
GOBIERNO PROVINCIAL DE
IMBABURA
Consultas
Entrada/Salida
Registros
PERMISOS
CEDULA DE IDENTIDAD:
OFICIALES
PERSONALES
CALAMIDAD
ENFERMEDAD
aaaa-mm-dd
FECHA : 2003-4-28
DIAS HORAS MINUTOS
GRABAR
Restablecer



Para salir del sistema únicamente se cierra el Internet Explorer dentro de cualquier aplicación que se encuentre.

MANUAL TECNICO

Introducción

El problema general a nivel de los profesionales en informática es entregar sistemas sin la documentación técnica necesaria para poder realizar cambios o modificaciones sobre la estructura de un programa sin problema alguno. El no disponer de la documentación necesaria técnicamente especificada limita al programador a realizar un seguimiento sin sentido que demanda tiempo y recursos en caso de necesitar modificar ciertos procesos.

Basándonos en lo expuesto anteriormente consideramos indispensable el especificar que una de las obligaciones básicas del profesional es el dar los manuales que contengan información verídica y clara sobre la codificación y estructuración de los diferentes procesos que forman un sistema. El presente documento tiende a ser una herramienta de ayuda y apoyo para aquellos programadores que deseen modificar el presente sistema.

El servidor HTTP Apache

Apache es uno de los mejores servidores de Webs utilizados en la red internet desde hace mucho tiempo, únicamente le hace competencia un servidor de Microsoft, el IIS. Por lo que éste servidor es uno de los mayores triunfos del software libre.

Es un servidor de web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1)

- Implementa los últimos protocolos, aunque se base en el HTTP/1.1
- Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo y con la API de programación de módulos.
- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para solución de los mismos.

La versión actual del apache es la 1.3. En la nueva versión se incluyen características como el soporte para Windows NT y Windows 95, así como la inclusión de cuatro dígitos en las fechas para evitar los problemas del año 2000.

El protocolo HTTP

Es el que da vida a Internet, y gracias al cual, los clientes y servidores se permiten comunicar. Su funcionamiento básico consiste en que el cliente establece una conexión, utilizando el protocolo TCP, con el servidor ; Y luego genera una petición , el servidor le responde y se cierra la conexión. En la versión (http 1.0), el cliente sólo podía invocar tres operaciones en el servidor :

- GET => Para pedir una página.
- HEAD => Para pedir la cabecera de un página.
- POST => Para enviar datos a una URL.

Para comprobar que Apache funciona correctamente, lánzalo (en el menú inicio) y haz que el navegador cargue la página http://localhost/. Para desconectar Apache puedes usar la opción correspondiente del menú de inicio.

TOMCAT

Tomcat es un servidor HTTP especializado en servlets y jsp. Es un proyecto del equipo de desarrollo de Apache y también es *open source* y *freeware*. Está escrito principalmente en Java. La dirección en la que puedes encontrar más información sobre el proyecto es http://www.jakarta.org. Descomprime

el fichero jakarta-**tomcat** en algún directorio, como C:\Program Files\Apache Group\. Cuando configures el programa, ten en cuenta los siguientes aspectos:

Necesitas tener instalado un Java Development Kit (1.1.x, 1.2 o 1.3).

Se deben definir las variables del entorno

CLASSPATH

o

JAVA_HOME

La variable del entorno JAVA_HOME se debe definir con el path donde se encuentra instalada la versión de jdk que utilices. Alternativamente, también puedes fijar CLASSPATH con el path del fichero tools.jar, por ejemplo C:\jdk1.3.\lib\tools.jar

Concretando, puedes añadir al fichero autoexec.bat

la siguiente línea:

set JAVA_HOME=C:\JDK1.3

aunque también podrías usar:

set CLASSPATH=C:\JDK1.3\LIB\TOOLS.JAR

Otras variables del entorno que se deben actualizar son

set ANT_HOME=C:\Program Files\Apache Group\jakarta-**tomcat**

set TOMCAT_HOME=C:\Program Files\Apache Group\jakarta-**tomcat**

Ahora ya puedes ejecutar en la ventana DOS el fichero batch startup.bat que se encuentra en el directorio jakarta-tomcat\bin.

Puedes comprobar que el servidor está funcionando haciendo que el navegador cargue la dirección http://localhost:8080. Para detener el servidor puedes ejecutar el script shutdown.bat

.

Tomcat y Apache

Tomcat es bueno sirviendo servlets, pero es muy ineficiente en lo que se refiere a ficheros estáticos. Además, adolece de funcionalidades básicas de servidores Web, como son la gestión de permisos y usuarios o la posibilidad de ejecutar CGI's o llamadas PHP. Por último, Tomcat no es tan robusto como puede ser Apache, o cualquier otro servidor Web dedicado..

Por ello, lo normal es utilizar la versión *stand-alone* de Tomcat para desarrollo y depuración. En sitios web reales se deberá utilizar un servidor, como Apache, para las páginas estáticas y el acceso a las funcionalidades mencionadas, y se usará Tomcat como un añadido (*add-on*) a este servidor principal, especializado en el servicio de servlets y JSP. Veamos a continuación cómo instalar Tomcat como un módulo de Apache.

Primero debes obtener el fichero que implementa el módulo. En Windows, se llama ApacheModuleJServ.dll. Cópialo en el directorio modules de la instalación de Apache. Por ejemplo, en C:\Program Files\Apache Group\Apache\modules

Edita el fichero de configuración del servidor web, httpd.conf. Se puede encontrar, por ejemplo en el directorio C:\Program Files\Apache

Group\Apache\conf. Añade la siguiente línea al final del mismo: Include "C:/Program Files/Apache Group/jakarta- tomcat/conf/tomcat-apache.conf" asegúrate de que el camino de tomcat-apache.conf es correcto. Este fichero se genera automáticamente por tomcat a partir del fichero server.xml, por lo que debes ejecutar tomcat antes de hacerlo con Apache, para que se genere este fichero que después va a cargar Apache. Una de las funciones más importantes de este fichero de configuración es establecer el directorio en el que se encuentran las aplicaciones web, para que Apache pueda acceder a ellas.

De forma alternativa, una opción más sencilla es utilizar los valores por defecto que proporciona tomcat. Se encuentran en un fichero estático, tomcat.conf, que hay que cargar de la misma forma que el anterior. Este fichero configura la aplicación web por defecto examples

Después se debe comentar la línea:

LoadModule jserv_module libexec/mod_jserv.so

y descomentar la línea:

LoadModule jserv_module modules/ApacheModuleJServ.dll

Para comprobar que todo funciona:

Lanza **Tomcat**

Ejecuta Apache

Comprueba si está funcionando, cargando la página http://localhost/

Ve a la página http://localhost/examples/jsp

MySQL

MySQL es un Sistema de Gestión de Base de Datos.

Una Base de Datos es una colección estructurada de datos. Puede ser, desde una simple lista de artículos a las inmensas cantidades de información en una red corporativa.

Actualmente el gestor de base de datos juega un rol central en la informática, como única utilidad, o como parte de otra aplicación.

MySQL es un software de código abierto esto quiere decir que es accesible para cualquiera, para usarlo o modificarlo. Podemos descargar MySQL desde Internet y usarlo sin pagar nada, de esta manera cualquiera puede inclinarse a estudiar el código fuente y cambiarlo para adecuarlo a sus necesidades. MySQL usa el GPL (GNU Licencia Publica General) para definir que podemos y no podemos hacer con el software en diferentes situaciones. Entre otras cuestiones esta licencia aclara que no cuesta dinero a menos que lo incluyamos en un software comercial y tenemos el código fuente.

Porqué usar MySQL?

MySQL es muy rápido, confiable, robusto y fácil de usar tanto para volúmenes de datos grandes como pequeños (siempre, claro está, comparada con las de su categoría, como veremos mas adelante en este informe). Además tiene un conjunto muy practico de características desarrolladas en cooperación muy cercana con los usuarios. Sin embargo bajo constante desarrollo, MySQL hoy en día ofrece un rico y muy útil conjunto de

funciones. La conectividad, velocidad y seguridad hace de MySQL altamente conveniente para acceder a bases de datos en Internet.

IBM empezó a comercializar en 1.981 el SQL y desde entonces este producto ha tenido un papel importante en el desarrollo de la bases de datos relacionales. IBM propuso y fue aceptada , una versión de SQL al Instituto de Estándares Nacional Americano(ANSI) y desde entonces es utilizado de forma generalizada en las bases de datos relacionales. En 1.983 nació DB2 la más popular(por lo menos en los grandes ordenadores) de las bases de datos de este tipo hasta estos mismos momentos.

En el mundo GNU, una de las bases de datos que se reseña en cualquier referencia de aplicaciones de éste tipo bajo LINUX, es MySQL aunque no está incluida en ninguna distribución ya que no tiene licencia GNU como tal, para comercializarla a ella o a cualquier software que la utilice o se sirva de ésta habrá que adquirir una licencia.

Alrededor de la década del 90, Michael Windenis (monty@analytikerna.se) comenzó a usar mSQL para conectar tablas usando sus propias rutinas de bajo nivel (ISAM). Sin embargo, después de algunos testeos llego a la conclusión que mSQL no era lo suficientemente rápido ni flexible para sus necesidades. De todo esto surgió en una nueva interfaz SQL (claro que con código mas portable) con algunas apariencias similares en la API de C y en los nombres y funciones de muchos de sus programas. Esto había sido hecho para lograr con relativa facilidad portar aplicaciones y utilidades de MiniSQL a MySQL.

Servlets Java

Los Servlets son las respuesta de la tecnología Java a la programación CGI. Son programas que se ejecutan en un servidor Web y construyen páginas Web. Construir páginas Web al vuelo es útil.

Los Servlets Java son más eficientes, fáciles de usar, más poderosos, más portables, y más baratos que el CGI tradicional y otras muchas tecnologías del tipo CGI. (y lo que es más importante, los desarrolladores de servlets cobran más que los programadores de Perl :-).

Estructura Básica de un Servlet

Aquí tenemos un servlet básico que maneja peticiones GET. Las peticiones GET, para aquellos que no estemos familiarizados con HTTP, son peticiones hechas por el navegador cuando el usuario teclea una URL en la línea de direcciones, sigue un enlace desde una página Web, o rellena un formulario que no especifica un METHOD. Los Servlets también pueden manejar peticiones POST muy fácilmente, que son generadas cuando alguien crea un formulario HTML que especifica METHOD="POST".

Para ser un servlet, una clase debería extender HttpServlet y sobreescribir doGet o doPost (o ambos), dependiendo de si los datos están siendo enviados mediante GET o POST. Estos métodos toman dos argumentos: un HttpServletRequest y un HttpServletResponse.

Compilar e Instalar el Servlet

Debemos observar que los detalles específicos para instalar servlets varian de servidor en servidor.

SET JAVA_HOME=d:\tomcat

SET

CLASSPATH=c:\jdk1.3.1_02\lib;c:\jdk1.3.1_02;c:\tomcat\lib;c:\tomcat;c:\tomcat\lib\common;c:\tomcat\lib\common\servlets.jar;

Subir el Server

Dentro de Java Home o el directorio c:\tomcat\bin

Startup.bat para iniciar el servidor Tomcat

Shutdown.bat para detenerlo

URL http://localhost:8080/examples/servlet/

Carpeta donde copiar nuestro sistema c:\tomcat\webapps\root\WEB-INF\classes

HttpServletRequest

El HttpServletRequest tiene métodos que nos permiten encontrar información entrante como datos de un FORM, cabeceras de peticiòn HTTP, etc. El HttpServletResponse tiene métodos que nos permiten especificar líneas de respuesta HTTP (200, 404, etc.), cabeceras de respuesta (Content-Type, Set-Cookie, etc.), y, todavía más importante, nos permiten obtener un PrintWriter usado para envíar la salida de vuelta al cliente. Para servlets sencillos, la mayoría del esfuerzo se gasta en sentencias println que generan la página deseada.

Manejar Datos de Formularios

URLs de búsqueda como

http://localhost:8080/path?user=Marty+Hall&origin=bwi&d est =lax. La parte posterior a la interrogación (user=Marty+Hall&origin=bwi&dest=lax) es conocida como datos de formulario, pueden añadirse al final de la URL después de la interrogación (como arriba) para peticiones GET o enviada al servidor en una línea separada, para peticiones POST.

Una de las mejores características de los servlets Java es que todos estos análisis de formularios son manejados automáticamente. Simplemente llamamos al método getParameter de HttpServletRequest, y suministramos el nombre del parámetro como un argumento.

Connección a Bases de Datos JDBC

JDBC (Java DataBase Connectivity) es una parte del API de Java que proporciona clases para conectarse con bases de datos. Dichas clases forman parte del package java.sql, disponible en el jdk 1.1.7 y en jdk 1.2.

El nombre JDBC es fonéticamente similar a ODBC (Open DataBase Connectivity), que es el estándar más extendido para conectar PCs con bases de datos.

SQL (Structured Query Language) es un lenguaje estándar de alto nivel que permite leer, escribir y en general gestionar bases de datos.

CODIGO JAVA DE LA PANTALLA PRINCIPAL

```
import java.io.*;
import java.util.*;
import java.net.*;
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;


/*****************************************************
**
**  Principal.java
**  DISEÑADO POR:
**      SABINA LOPEZ
**      JANETH GUAIGUA
**
*****************************************************/

public class principal extends HttpServlet {
static Connection canal = null;
static ResultSet tabla= null;
static Statement instruccion=null;
String sUsuario = null;
String sPassword = "IMI";

public void doPost(HttpServletRequest request,HttpServletResponse
response)
throws ServletException, IOException {

    String Tipo = "0";
    response.setContentType("text/html");
    PrintWriter pagina = response.getWriter();
    pagina.println("<HTML><HEAD><TITLE>GOBIERNO PROVINCIAL
   DE     IMBABURA - UTN</TITLE></HEAD>");

        sPassword = request.getParameter("Password");
        if (sPassword.length() == 0)
           pagina.print("<I>Ingrese Password</I><br>");
        sUsuario = request.getParameter("Usuario");
        if (sUsuario.length() == 0)
           pagina.print("<I>Ingrese Usuario</I>");

   try { Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         // Realizar la conexión a la base de datos
         canal=DriverManager.getConnection("jdbc:odbc:MySql", "saby",
"saby");
instruccion = canal.createStatement(ResultSet.TYPE_SCROLL_
SENSITIVE,ResultSet.CONCUR_UPDATABLE);
```

```
        } catch(java.lang.ClassNotFoundException e){} catch(SQLException
e) {};

    String sSql = "SELECT usuario, password, tipo FROM Usuario " +
          "WHERE ((usuario = '"+ sUsuario +"') AND (password =
'"+sPassword +")) ";

if ((sUsuario.length() != 0) &&  (sPassword.length() != 0)) {  // or  ||
try {
  tabla = instruccion.executeQuery(sSql);
  tabla.next();
  Tipo = tabla.getString(3);

  HttpSession session = request.getSession(true);
  session.putValue("referringePage", request.getHeader("referer"));
  session.putValue("Usuario", sUsuario);
  session.putValue("Tipo",Tipo);

if (Tipo != "0") {
  pagina.println("<frameset border=false frameBorder=0 frameSpacing=0
rows=99,*> <frame name=titular marginHeight=1 marginWidth=1 noResize
scrolling=no");
  pagina.println("target=contenido
src=http://localhost:8080/paginas/titulo.htm>");
  pagina.println("<frameset cols=167,*>");
  pagina.println(" <frame marginHeight=1 marginWidth=1 noResize
scrolling=no name=contenido");
  pagina.println(" target=principal src=http://localhost:8080/servlet/indice>");
  pagina.println(" <frame name=principal
src=http://localhost:8080/paginas/inicial.htm>");
  pagina.println("</frameset>");
  pagina.println("<noframes>");
  pagina.println("<body>");
  pagina.println("<p>Esta página usa marcos, pero su explorador no los
admite.</p>");
  pagina.println(" </body>");
  pagina.println("</noframes>");
  pagina.println("</frameset>");
  pagina.println("</html>");

} else {
  pagina.println("<p>No existe este usuario o esta mal digitado el password
</p>");
}

  pagina.close();
  tabla.close(); } //fin try no usar ; al final de dos o mas catchs

catch(SQLException e) {};
```

```
destroy();
try {canal.close();} catch(SQLException e) {};
}; //fin del if
};//fin dopost
public void destroy(){super.destroy();};
} //fin clase
```