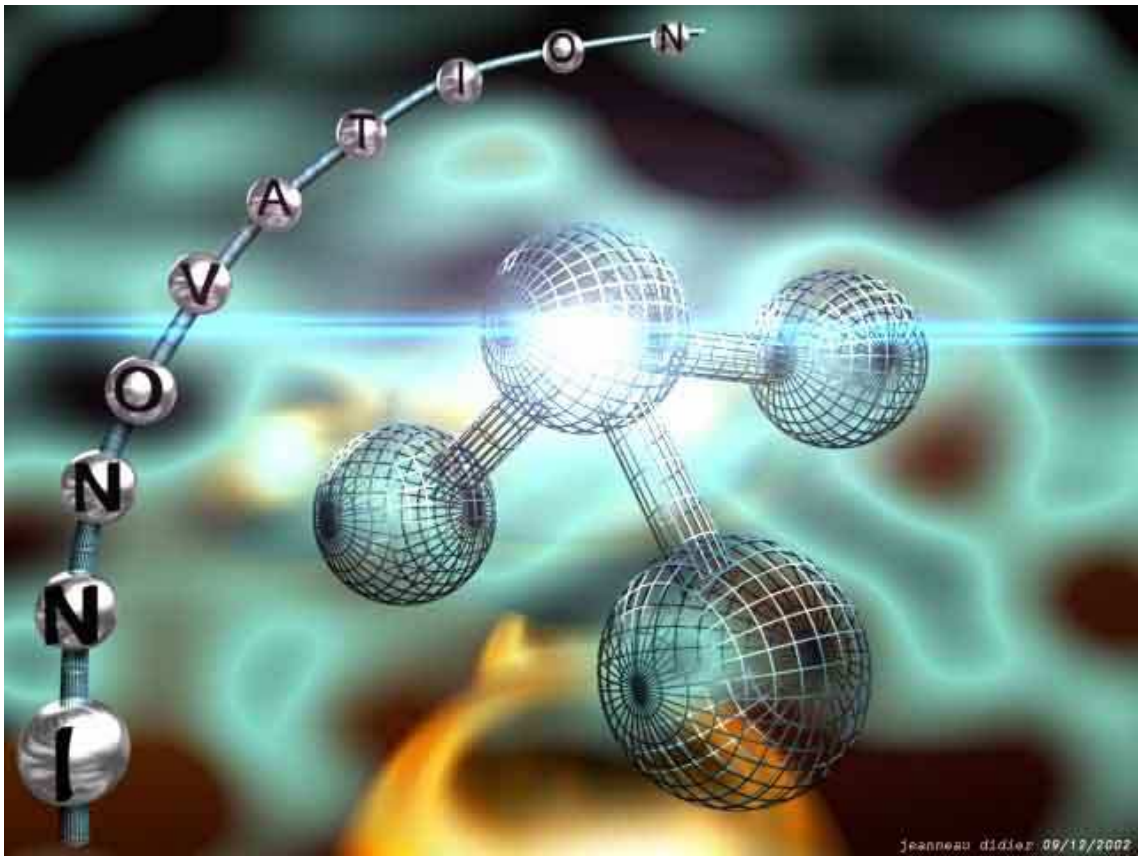
La innovación tecnológica y su gestión.



La Innovación= Nuevos conceptos, productos, servicios +productividad

La Innovación:

Es la aplicación de **nuevas ideas, conceptos, productos, servicios y prácticas**, con la intención de ser útiles para el incremento de la productividad. Un elemento esencial de la innovación es su aplicación exitosa de forma comercial. No sólo hay que inventar algo, sino, por ejemplo, introducirlo (Difusión (negocios)) en el mercado para que la gente pueda disfrutar de ello. En ese sentido, Innovar proviene del latín *innovare*, que significa acto o efecto de innovar, tornarse **nuevo o renovar**, introducir una novedad.

Definición de Tecnología

La tecnología es un concepto amplio que abarca un **conjunto de técnicas, conocimientos y procesos**, que sirven para el diseño y construcción de objetos para satisfacer necesidades humanas.

En la sociedad, la tecnología es consecuencia de la ciencia y la ingeniería, aunque muchos avances tecnológicos sean posteriores a estos dos conceptos.

La palabra tecnología proviene del griego tekne (técnica, oficio) y logos (ciencia, conocimiento).

¿A qué hace referencia la palabra "tecnología"?

La tecnología puede referirse a objetos que usa la humanidad (como máquinas, utensilios, hardware), pero también abarca sistemas, métodos de organización y técnicas.

El término también puede ser aplicado a áreas específicas como "tecnología de la construcción", "tecnología médica", "tecnología de la información", "tecnología de asistencia", etc.

Diferencia entre técnica y tecnología

A veces no se distingue entre técnica y tecnología, pero sí pueden diferenciarse:
* La tecnología se basa en aportes científicos, en cambio la técnica por experiencia social;
* La actividad tecnológica suele ser hecha por máquinas (aunque no necesariamente) y la técnica es preferentemente manual;
* La tecnología se suele poder explicar a través de textos o gráficos científicos, en cambio la técnica es más empírica.

Breve historia de la tecnología

La humanidad comienza a formar tecnología convirtiendo los recursos naturales en herramientas simples. El descubrimiento prehistórico de controlar el fuego incrementa la disponibilidad de fuentes de comida, y la invención de la rueda ayuda a los humanos a viajar y controlar su entorno.

La tecnología formal tiene su origen cuando la técnica (primordialmente empírica) comienza a vincularse con la ciencia, sistematizándose así los métodos de producción. Ese vínculo con la ciencia, hace que la tecnología no sólo abarque "el hacer", sino también su reflexión teórica. Tecnología también hace referencia a los productos resultados de esos procesos.

Muchas tecnologías actuales fueron originalmente técnicas. Por ejemplo, la ganadería y la agricultura surgieron del ensayo (de la prueba y error). Luego se fueron tecnificando a través de la ciencia, para llegar a ser tecnologías.

Actualmente, el mercado y la competencia en general, hacen que deban producirse nuevas tecnologías continuamente (tecnología de punta), ayudado muchas veces por la **gran transferencia de tecnología** mundial. También existe una tendencia a la miniaturización de los dispositivos tecnológicos.

Distinción entre tecnología, ciencia y arte

Una buena forma de distinguir tecnología, de ciencia y de arte es a través de su finalidad:
- Las ciencias buscan la verdad a través de los métodos científicos.
- Las artes buscan llegar a los sentimientos humanos, el placer intelectual, la belleza de todas las cosas.
- Las tecnologías buscan satisfacer necesidades y deseos humanos, buscan resolver problemas prácticos usando en parte la ciencia.

Clasificación de tecnologías

Existen múltiples formas de clasificación de las tecnologías, la más general suele ser la que separa entre:
*** Tecnologías blandas: básicamente aquellas que son intangibles.**
*** Tecnologías duras: básicamente aquellas que son tangibles.**

Definición de Tecnología blanda:

Tipo o clasificación de tecnologías que hacen referencia a los conocimientos tecnológicos de tipo organizacional, administrativo y de comercialización, **excluyendo los aspectos técnicos**. En otras palabras, hace referencia al know-how, las habilidades y las técnicas. Es "blanda" pues se trata de información no necesariamente tangible.

Por ejemplo, las técnicas de conservación de una comunidad de agricultores o las técnicas de entrenamiento en el manejo de vida silvestre, podrían considerarse tecnologías blandas.

Definición de Tecnología dura:

Tipo o clasificación de tecnologías que hace referencia a aquellas que son tangibles, contrastando así con las tecnologías blandas.

Una computadora o cualquier dispositivo electrónico son ejemplos de tecnologías duras.

Innovación tecnológica

Hasta los años 60, no se asume la importancia de los problemas de la innovación tecnológica. En ese momento se inicia una corriente de conocimiento que señala a la innovación como un elemento fundamental en la prosperidad de las naciones avanzadas y a la tecnología como principal factor de innovación.
Se convierte así la innovación tecnológica en un ingrediente vital para el mantenimiento de la prosperidad de una nación y de la empresa. (I+D)

Las empresas se han visto influidas por los cambios tecnológicos en esta última década y, como consecuencia de ello se ha ido imponiendo la necesidad:

- El enfoque estratégico de la dirección de una empresa
- Una mejor comprensión del proceso de innovación tecnológica
- La consideración de la tecnología como variable estratégica
- La dirección estratégica de la innovación.

Es necesario que los administradores de las organizaciones sepan gestionar el cambio, el objetivo es q las organizaciones sean mas efectivas para satisfacer nuestras necesidades.

Aunque no hay duda que un banco necesita una administración distinta a un hospital, una universidad o una empresa química, en todo los casos es necesaria una

administración que asuma la responsabilidad de alcanzar los objetivos, logre un trabajo un trabajo productivo y de calidad, encauce la innovación, afronte el crecimiento, la diversidad y la complejidad, y sepa dirigir al mismo tiempo la organización existente y la nueva organización, se necesita una administración para el cambio.

La Informática

La Informática= Automatización + información.

Dispositivos Electrónicos + Sistemas Computacionales.

Tareas básicas:

- Entrada: Captación de la información digital.
- Proceso: Tratamiento de la información.
- Salida: Transmisión de resultados binarios.

Aplicaciones de la Informática y sus beneficios:

Actualmente, la informática tiene múltiples aplicaciones en el mundo, esta presente en todas las áreas o campos; a continuación se mencionan algunos campos:

- **En el área Administrativa:** Se hace imprescindible, siendo de gran utilidad para ejecutivos, administradores, contadores, donde se efectúan tareas rutinarias tales como: Control reinventario, nómina de empleados y obreros, cuentas por pagar, cuentas por cobrar, facturación, control de bancos, análisis

de costos y ventas. Igualmente, existen aplicaciones con salidas graficas y estadísticas, que pueden predecir las tendencias futuras.

- **En la Educación:** El papel del computador y la informática en el proceso de aprendizaje es muy importante, ya que reúne una gran cantidad de medios relevantes para una situación de enseñanza efectiva a través de medios de múltiples medios (Multimedia). El Internet se ha convertido en una herramienta imprescindible para la elaboración de trabajo e investigaciones, es una ventana al mundo. Por otra parte, se ha desarrollado la educación virtual (Elearning) a nivel mundial a través de software especializados como Moodle. En fin, la informática ha evolucionado la educación de manera muy significativa.
- **En la Navegación:** En el área marítima se controla la fijación de posiciones o situaciones geográficas mediante satélites. En los puertos, una gran parte de las operaciones de carga y descarga se realizan de acuerdo a un programa establecido por el computador.
- **En la Aeronáutica:** Control de trafico aéreo, se han fabricado aviones que no requieren de un piloto, y pueden volar a una gran velocidad, siguiendo una ruta trazada por un software. Asimismo, pueden dirigir el lanzamiento de naves espaciales.
- **En la Ciencia:** Proporciona gran ayuda al hombre en las diferentes ramas científicas y de investigación, entre otras aplicaciones tenemos: analizar datos, almacenar y recuperar información, simplificar expresiones llevar información estadística de procesos, controlar experimentos, identificas moléculas, entre otros.
- **Transporte Urbano:** Controlar el servicio de autobuses. En una pequeña ciudad de Alemania se ha implantado un sistema que permite controlar el servicio de autobuses, según la demanda de servicio. Por ejemplo, cunado no hay pasajeros en una determinada ruta el computador no permite el envió de autobuses a esa ruta.
- **En la Industria:** Se puede controlar y operar los procesos industriales, siendo muy utilizados en las siguientes áreas: La industria del cemento, el caucho, los metales, el papel, el petróleo, el vidrio, los productos químicos, los textiles, la energía, la manufactura, entre otros.
- **En la Seguridad y Vigilancia:** Son usados en muchas tareas, desde los pagos del personal, hasta información instantánea acerca de carros robados, falsificación de documentos, valores y el análisis de prueba. Igualmente, en el área de criminología se hace uso exhaustivo de la informática.
- **En la Medicina:** Es posible hacer diagnostico médicos, pudiendo detectar, por ejemplo, cuando el paciente ha sufrido un ataque cardíaco; también, el estudio y análisis de electroencefalogramas, electrocardiogramas y ecografías.
- **Otras Aplicaciones:** Diseño de modas, elaboración de diarios y periódicos, producción de videos, producción de música, en la arquitectura, elaboración de libros y revistas, en la recreación a través de juegos interactivos, en la hotelería, y muchas más áreas.

Beneficios: Disminución de tiempo en los procesos, menor costo y empleo en horas hombres, procesos mas rápidos, óptimos y efectivos para una mayor productividad.

Hardware y Software

Hardware: Parte Física de un Computador.

Definición: Término inglés que hace referencia a cualquier componente físico tecnológico, que trabaja o interactúa de algún modo con la computadora. No sólo incluye elementos internos como el disco duro, CD-ROM, unidad de disco, sino que también hace referencia al cableado, circuitos, entre otros. Igualmente, hace referencia a elementos externos como la impresora, el mouse, el teclado, el monitor y demás periféricos.

El hardware no es frecuentemente cambiado, en tanto el software puede ser creado, borrado y modificado sencillamente.

Hardware típico de una computadora

El típico hardware que compone una computadora personal es el siguiente:

• Su chasis o gabinete
• La placa madre, que contiene: CPU, cooler, RAM, BIOS, buses (PCI, USB, HyperTransport, CSI, AGP, etc)
• Fuente de alimentación
• Controladores de almacenamiento: IDE, SATA, SCSI
• Controlador de video
• Controladores del bus de la computadora (paralelo, serial, USB, FireWire), para conectarla a periféricos
• Almacenamiento: disco duro, CD-ROM, disquetera, ZIP driver y otros
• Tarjeta de sonido
• Redes: módem y tarjeta de red

El hardware también puede incluir componentes externos como:

• Teclado
• Mouse, trackballs
• Joystick, gamepad, volante
• Escáner, webcam
• Micrófono, parlante
• Monitor (LCD, o CRT)
• Impresora

Distintas clasificaciones del hardware

Clasificación por la funcionalidad del hardware

* **Hardware básico:** dispositivos necesarios para iniciar la computadora. Los más básicos son la placa madre, la fuente de poder, el microprocesador y la memoria. Se podrían incluir componentes como monitor y teclado, aunque no son estrictamente básicos.

*** Hardware complementario:** aquellos dispositivos que complementan a la computadora, pero que no son fundamentales para su funcionamiento, como ser, impresora, unidades de almacenamiento, etc.

Clasificación por la ubicación del hardware

*** Periféricos (componentes externos):** dispositivos externos a la computadora
* **Componentes internos**: dispositivos que son internos de la computadora.
*** Puertos:** conectan los periféricos con los componentes internos.

Clasificación por el flujo de información del hardware

* **Periféricos de salida:** monitor, impresora, etc.
* **Periféricos de entrada:** teclado, mouse, etc.
* **Periféricos/dispositivos de almacenamiento:** disco duro, memorias, etc.
* **Periféricos de comunicación:** módem, puertos, etc.
* **Dispositivos de procesamiento:** CPU, microprocesador, placa madre, etc.

Costos:

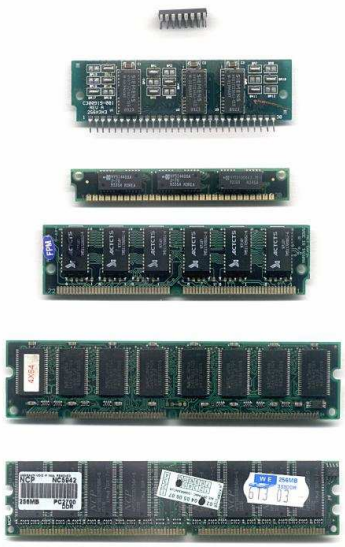
1. Características del equipo.
2. Marca.
3. Garantía.
4. Garantía en Sitio

5. Condiciones de la Oferta
   5.1. Tiempo de Entrega
   5.2. Condición de Pago
   5.3. Validez de la Oferta
6. Precio Ofertado.

Elementos a Considerar para la Adquisición de un Hardware.

Procesador (Micropocesador): Microchip más importante en una computadora, es considerado el cerebro de una computadora. Está constituido por millones de transistores integrados. Este dispositivo se ubica en un zócalo especial en la placa madre y dispone de un sistema de enfriamiento (generalmente un ventilador).

Su "velocidad" es medida por la cantidad de operaciones por segundo que puede realizar: la frecuencia de reloj. La frecuencia de reloj se mide en MHz (megahertz) o gigahertz (GHz).

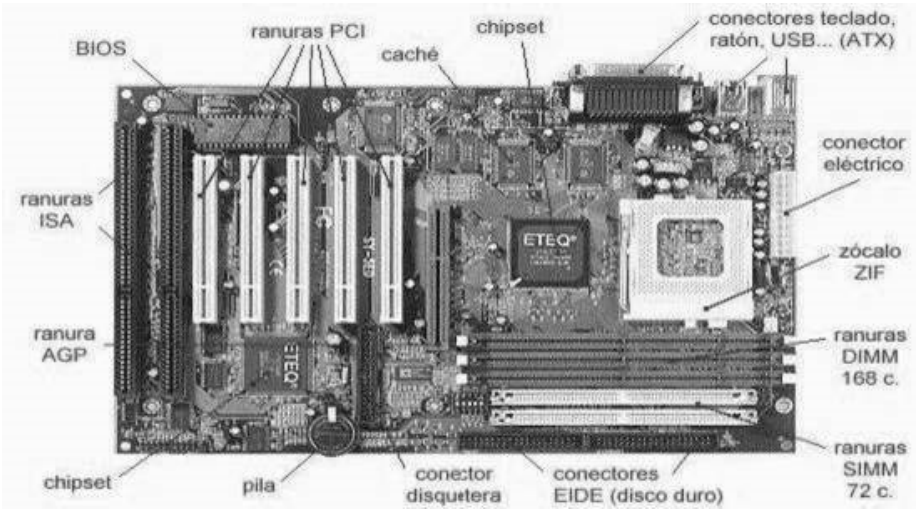


Memoria Ram (Tamaño y velocidad): Random Access Memory - Memoria de acceso aleatorio). Tipo de memoria donde la computadora guarda información para que pueda ser procesada más rápidamente. En la memoria RAM se almacena toda información que está siendo usada en el momento.

Su capacidad de almacenamiento se mide en megabytes y más recientemente en gigabytes.

La información que contienen es renovada continuamente y cuando la computadora se reinicia o se apaga, toda la información contenida se pierde, por eso es llamada memoria volátil.

- **Tipos de memoria RAM: SRAM, DRAM.**
- **Tipos de módulo de memorias RAM: SIMM, DIMM, RIMM.**



Memoria Cache: Una memoria caché es una memoria en la que se almacenas una serie de datos para su rápido acceso. Existen muchas memorias caché (de disco, de sistema, incluso de datos, como es el caso de la caché de Google).
Básicamente, la memoria caché de un procesador es un tipo de memoria volátil (del tipo RAM), pero de una gran velocidad.

En la actualidad esta memoria está integrada en el procesador, y su cometido es almacenar una serie de instrucciones y datos a los que el procesador accede continuamente, con la finalidad de que estos accesos sean instantáneos. Estas instrucciones y datos son aquellas a las que el procesador necesita estar accediendo de forma continua, por lo que para el rendimiento del procesador es imprescindible que este acceso sea lo más rápido y fluido posible.

Hay tres tipos diferentes de memoria caché para procesadores:

Caché de 1er nivel (L1):

Esta caché está integrada en el núcleo del procesador, trabajando a la misma velocidad que este. La cantidad de memoria caché L1 varía de un procesador a otro, estando normalmente entra los 64KB y los 256KB. Esta memoria suele a su vez estar dividida en dos partes dedicadas, una para instrucciones y otra para datos.

Caché de 2º nivel (L2):

Integrada también en el procesador, aunque no directamente en el núcleo de este, tiene las mismas ventajas que la caché L1, aunque es algo más lenta que esta. La caché L2 suele ser mayor que la caché L1, pudiendo llegar a superar los 2MB.
A diferencia de la caché L1, esta no está dividida, y su utilización está más encaminada a programas que al sistema.

Caché de 3er nivel (L3):

Es un tipo de memoria caché más lenta que la L2, muy poco utilizada en la actualidad.

En un principio esta caché estaba incorporada a la placa base, no al procesador, y su velocidad de acceso era bastante más lenta que una caché de nivel 2 o 1, ya que si bien sigue siendo una memoria de una gran rapidez (muy superior a la RAM, y mucho más en la época en la que se utilizaba), depende de la comunicación entre el procesador y la placa base.

Para hacernos una idea más precisa de esto, imaginemos en un extremo el procesador y en el otro la memoria RAM. Pues bien, entre ambos se encuentra la memoria caché, más rápida cuanto más cerca se encuentre del núcleo del procesador (L1).

Las memorias caché son extremadamente rápidas (su velocidad es unas 5 veces superior a la de una RAM de las más rápidas), con la ventaja añadida de no tener latencia, por lo que su acceso no tiene ninguna demora... pero es un tipo de memoria muy cara.
Esto, unido a su integración en el procesador (ya sea directamente en el núcleo o no) limita bastante el tamaño, por un lado por lo que encarece al procesador y por otro por el espacio disponible.

En cuanto a la utilización de la caché L2 en procesadores multinucleares, existen dos tipos diferentes de tecnologías a aplicar.

Por un lado está la habitualmente utilizada por Intel, que consiste en que el total de la caché L2 está accesible para ambos núcleos y por otro está la utilizada por AMD, en la que cada núcleo tiene su propia caché L2 dedicada solo para ese núcleo.

La caché L2 apareció por primera vez en los Intel Pentium Pro, siendo incorporada a continuación por los Intel Pentium II, aunque en ese caso no en el encapsulado del procesador, sino externamente (aunque dentro del procesador).

Disco Duro o Rígido: (Hard disk, HD, HDD, Disco duro). Dispositivo de almacenamiento permanente que pertenece a la categoría de discos magnéticos. Suelen ser rectangulares y protegidos por una caja metálica herméticamente cerrada.

La información se escribe/lee en discos que rotan (rpm) y que están recubiertos por una película magnética. Poseen diversas capacidades de almacenamiento que cada vez es más elevada y que actualmente llega a más de 500 GB.

Existen diversos estándares en la comunicación de la información entre el disco duro y la computadora, estos son: IDE/ATA, SCSI y SATA.

Un disco duro virgen para poder ser usado debe dársele un formato de bajo nivel, definirse una o más particiones y finalmente darle un formato compatible con nuestro sistema.

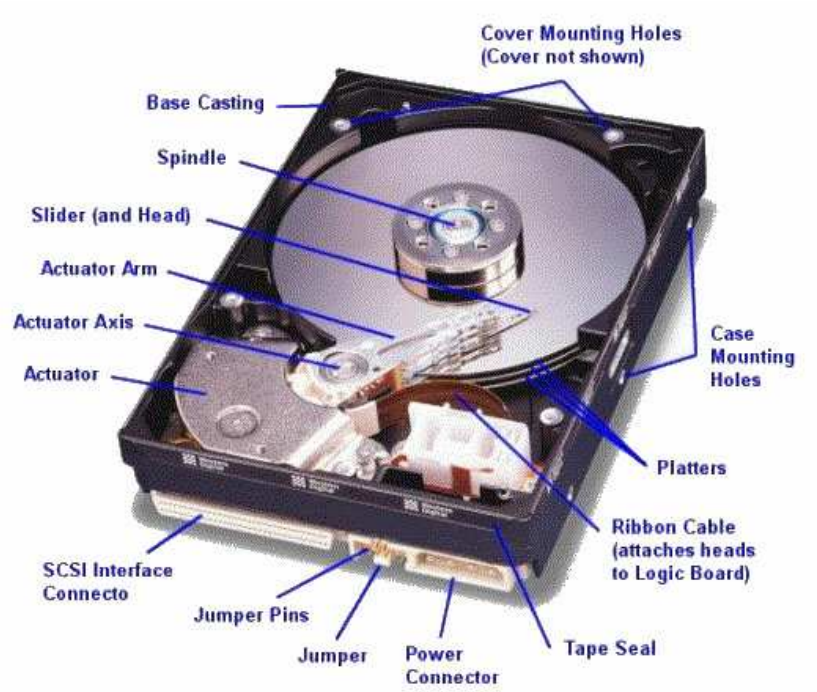
Actualmente la mayoría de los discos duros, poseen un sistema llamado SMART, que permite detectar posibles fallas mecánicas.

Componentes de un disco rígido

• **Cabeza de lectura/escritura:** son los elementos que se encargan de leer o escribir los datos de los discos magnéticos internos del disco duro. Son similares a los brazos de los tocadiscos.

• **Pistas del disco:** finas sendas concéntricas donde se almacenan los datos. Un giro completo del disco describe una pista. Las pistas se subdividen lógicamente en sectores o clústeres.

• **Cilindros y sectores:** los cilindros son el conjunto de pistas de los distintos discos internos que coinciden verticalmente.

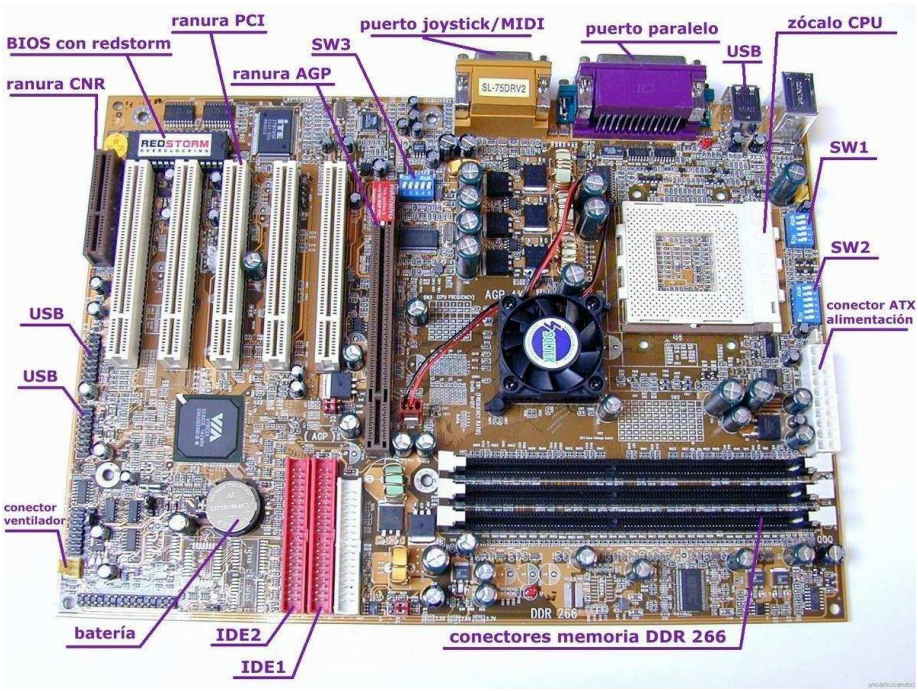


Tarjeta Madre: (placa madre, placa base, tarjeta madre, mainboard, system board, logic board). Tarjeta central de circuitos integrados para la interconexión entre el

microprocesador, las ranuras o slots para conectar tarjeta de expansión, memorias RAM, la ROM, dispositivos de almacenamiento y cableados.

Tipos más comunes de placas madre:
XT, AT, Baby-AT, ATX, EATX, Mini-ATX, microATX, LPX, Mini-LPX, NLX, FlexATX, Mini-ITX, Nano-ITX, BTX, MicroBTX, PicoBTX, WTX, ETX y PC/104.

Son fabricantes del motherboards: ABIT, AOpen, ASUS, ASRock, BFG, Technologies, Biostar, DFI, ECS, EPoX, Foxconn (fabricante de las placas de Intel), Gigabyte, Intel, Magic-Pro, MSI, Shuttle, Tyan, VIA.

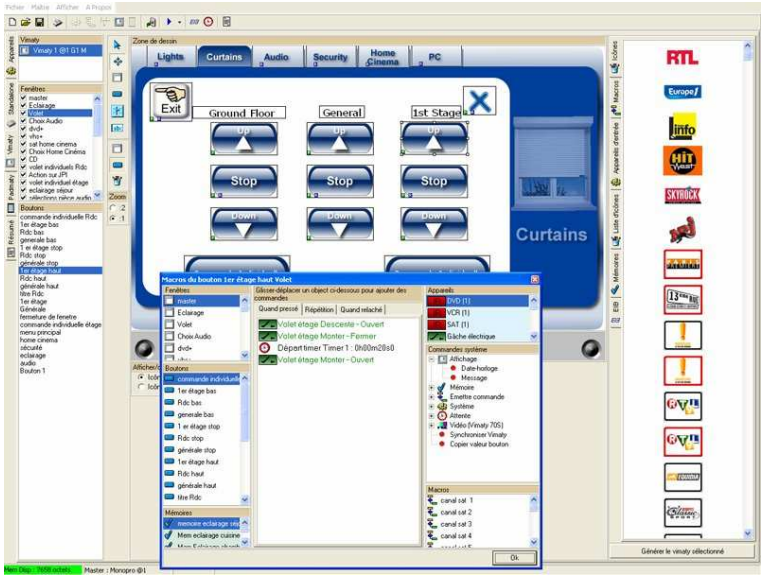


Baremo Técnico para Adquirir Hardware

Baremo Técnico Requisición Nro. 1		Rango Puntuación Máximo	 Mínimo
Computadores de Escritorios			
Ítem	**Descripción**		
1	Características del Equipo		
1.1	Marca	2	0
1.2	Modelo	0,5	0
1.3	Procesador –Tipo	2	0
1.4	Procesador – Cantidad/Velocidad/Actualización	4,5	0
1.5	RAM-Tipo	1,5	0
1.6	RAM-Tamaño (Gb) / Expansión	2	0
1.7	RAM-Velocidad (MHz)	1,5	0
1.8	Cache –Tamaño	0,5	0
1.9	Disco-Duro Tipo	0,5	0
1.10	Disco Duro- Cantidad		
1.11	Disco Duro-Tamaño Gb		
1.12	Disco Duro-Total Tamaño	2	0

1.13	Controlador Disco Duro	1	0
1.14	Tarjeta video/Cantidad Memoria	1	0
1.15	Tarjeta de Red/Velocidad	1	0
1.16	Chipset	1	0
1.17	Bus del Sistema	1	0
1.18	Mouse – Tipo	0,5	0
1.19	Teclado – Tipo	0,5	0
1.20	Teclado - Idioma	0,5	0
1.21	Teclado	0,5	0
1.22	Unidad de Disketes 1.44	0,5	0
1.23	Unidad de DVD-RW / Veloc.	2	0
1.24	Puerto Paralelo	0,5	0
1.25	Puerto Serial	0,5	0
1.26	Puertos USB 2.0	0,5	0
1.27	Puertos PS/2	0,5	0
1.28	Tipo Fuente	0,5	0
1.29	Ranuras Cantidad/Tipo	0,5	0
2	Compatibilidad (Software)	1,5	0
3	Garantía	4	0
3.1	Garantía en Sitio	1	0
4	Condiciones de la Oferta		
4.1	Tiempo de Entrega	5	1
4.2	Condición de Pago	2	1
4.3	Validez de la Oferta	1	0,5
5	Precio Ofertado	56	0

Total	100	2,5



Software: Parte no Física de un Computador.

Definición de Software: Es todo programa o aplicación programado para realizar tareas específicas. El término "software" fue usado por primera vez por John W. Tukey en 1957.

La palabra "software" es un contraste de "hardware"; el software se ejecuta dentro del hardware.

El software en sentido amplio

Una definición más amplia de software incluye mucho más que sólo los programas. Esta definición incluye:

- **La representación del software:** programas, detalles del diseño escritos en un lenguaje de descripción de programas, diseño de la arquitectura, especificaciones escritas en lenguaje formal, requerimientos del sistema, etc.

- **El conocimiento de la ingeniería del software:** Es toda la información relacionada al desarrollo de software (por ejemplo, cómo utilizar un método de diseño específico) o la información relacionada al desarrollo de un software específico (por ejemplo, el esquema de pruebas en un proyecto). Aquí se incluye información relacionada al proyecto, información sobre la tecnología de software, conocimiento acerca de sistemas similares y la información detallada relacionada a la identificación y solución de problemas técnicos.

El "software" como programa: El software, como programa, consiste en un código en un lenguaje máquina específico para un procesador individual. El código es una secuencia de instrucciones ordenadas que cambian el estado del hardware de una computadora.

El software se suele escribir en un lenguaje de programación de alto nivel, que es más sencillo de escribir (pues es más cercano al lenguaje natural humano), pero debe convertirse a lenguaje máquina para ser ejecutado.

El software puede distinguirse en tres categorías:

1. **software de sistema,**
2. **software de programación y**
3. **software de aplicación.**

- **Software de sistema:** ayuda a funcionar al hardware y a la computadora. Incluye el sistema operativo, controladores de dispositivos, herramientas de diagnóstico, servidores, sistema de ventanas, utilidades y más. Su propósito es evitar lo más posible los detalles complejos de la computación, especialmente la memoria y el hardware.

- **Software de programación:** provee herramientas de asistencia al programador. Incluye editores de texto, compiladores, intérprete de instrucciones, enlazadores, debuggers, asistentes y generadores, entre otros.

- **Software de aplicación:** permite a los usuarios finales hacer determinadas tareas. Algunos software de aplicación son los navegadores, editores de texto, editores gráficos, antivirus, mensajeros, etc.

El software puede clasificarse según su licencia y/o forma de distribución:

Distribución y Licenciamiento de Software

Distribución de software:

Una distribución de software, támbien conocido como software distro, es un compilado de software específico (o una colección de múltiple software, incluso un sistema

operativo), ya compilado y configurado. Generalmente pueden tomar formas de licencia, de entre la más usada es la licencia GPL u open source. También puede tomar la forma de una distribución binaria, un instalador (.exe) que puede ser descargado desde Internet. Distribución de software también se puede referir a los tipos de Otherware (como Careware y Donateware).

Pueden ser distribuciones oficiales de los autores originales del software, o distribuciones 3rd party.

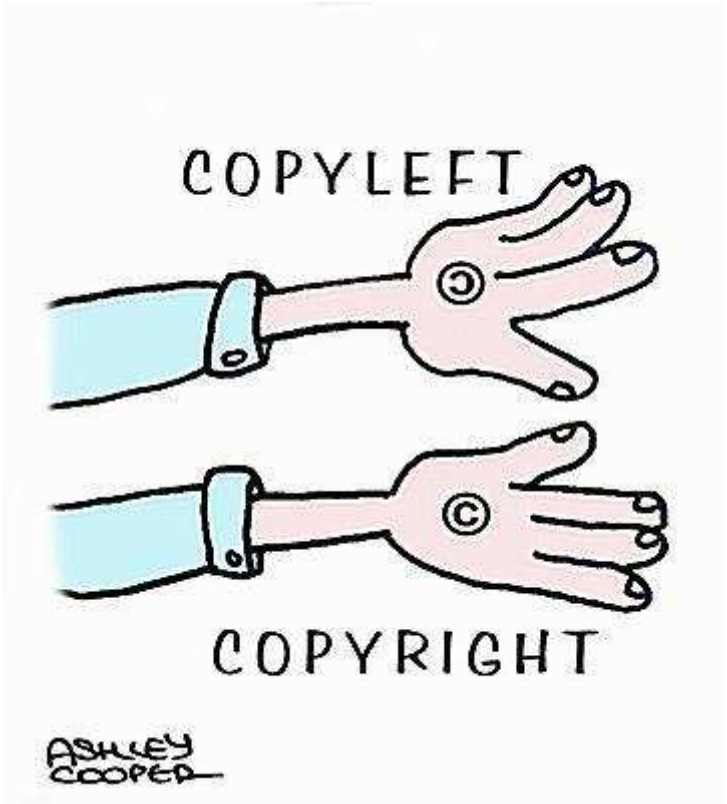
Licencia de software:

Una licencia de software es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribuidor) y el licenciatario del programa informático (usuario consumidor /usuario profesional o empresa), para utilizar el software cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas.

Las licencias de software pueden establecer entre otras cosas: la cesión de determinados derechos del propietario al usuario final sobre una o varias copias del programa informático, los límites en la responsabilidad por fallos, el plazo de cesión de los derechos, el ámbito geográfico de validez del contrato e incluso pueden establecer determinados compromisos del usuario final hacia el propietario, tales como la no cesión del programa a terceros o la no reinstalación del programa en equipos distintos al que se instaló originalmente.

Definición de Copyleft

Izquierdo de copia (traducción propuesta en los documentos GNU). Grupo de derechos inicialmente aplicables a cualquier trabajo informático, y actualmente a cualquier producción creativa. Nace como contraposición al copyright o derechos de autor.
Toda obra realizada bajo licencia copyleft permite usarla, modificarla o redistribuirla siempre bajo la misma licencia.
Fue Richard Stallman, creador del software libre en su proyecto GNU, quien acuñó este concepto. En general toda obra copyleft entra en licencia GNU también.



Definición de Copyright

Licencia a aquellas obras de cualquier tipo que protegen los derechos de autoría.

Definición de Copyleft

Izquierdo de copia (traducción propuesta en los documentos GNU). Grupo de derechos inicialmente aplicables a cualquier trabajo informático, y actualmente a cualquier producción creativa. Nace como contraposición al copyright o derechos de autor.
Toda obra realizada bajo licencia copyleft permite usarla, modificarla o redistribuirla siempre bajo la misma licencia.
Fue Richard Stallman, creador del software libre en su proyecto GNU, quien acuñó este concepto. En general toda obra copyleft entra en licencia GNU también.

Formas o licencias de distribución de software:
Adware • Beerware • Careware • Crippleware • Código abierto • Donationware • Donateware • Freeware • Greenware • Nagware • Postcardware • Ransomware • Registerware • Shareware • Software de distribución libre • Software libre • Software propietario • Trialware (Demoware)

Definición de Adware: (Contracción de ADvertisement + softWARE). Tipo de aplicaciones que incluyen alguna forma de publicidad mostrada cuando son ejecutados.

Los desarrolladores usan el adware como recurso para lograr ingresos económicos de sus programas, que usualmente son gratuitos. A veces los usuarios pueden pagar para que desaparezca la publicidad de las aplicaciones adware.

No deben asociarse este tipo de programas con aquellos que incluyen spywares, aunque es cierto que muchas veces van de la mano. Muchos programas adware monitorizan la actividad de sus usuarios sin su consentimiento. Un ejemplo de esto último es el Kazaa.

Algunas aplicaciones adware populares son TopMoxie, 180 Solutions, 180SearchAssistant, Zango, Bonzi Buddy, ClipGenie, Comet Cursor, Cydoor, Daemon Tools, ErrorSafe, Gator Hotbar, PornDigger!, Smiley Central, WeatherBug, WhenU, WinFixer.

Muchas de estas consideradas también espías como Gator Hotbar.

Definición de Beerware: (Beer = cerveza, ware = software). Beerware es un término en broma utilizado para designar aquellas aplicaciones que pueden ser "compradas" si se le compra una cerveza a su autor (o, en otra variante, se bebe una cerveza en nombre de este).

Definición de Careware: (charityware, helpware, goodware - software de caridad, de ayuda o de bien). Careware es una forma de distribución de software que beneficia una entidad de caridad. Algunos careware son distribuidos gratuitamente y el autor sugiere que se done dinero a una caridad mencionada.

Por ejemplo, el editor de textos "vim" es un software libre, pero incluye un pedido de su autor para que sus usuarios donen dinero a una institución de ayuda de víctimas de SIDA en Uganda. Otro ejemplo actual es la aplicación MJs CD Archiver, un archivador de ficheros que sugiere una caridad a NACEF.

Definición de Crippleware: Tipo de software que se distribuye de forma gratuita, y que posee algunas de sus características importantes deshabilitadas, como grabar o imprimir. Para habilitar esas características se debe comprar una clave de registración.

Definición de Código abierto: (Open source). Denominación para aquellas aplicaciones que tienen su código fuente liberado. En general, los programas de código abierto suele ser libres. Aunque existen aplicaciones de código abierto que no son libres.

Open Source es utilizado también para hacer referencia a un nuevo movimiento de software, la Open Source Initative.

Definición de Donationware: Donationware es un modelo de licencia para software. El software donationware es completamente funcional, pero tiene la característica de que pide una donación al programador por el tiempo y esfuerzo que llevó desarrollar el programa.

Si el pago va hacia un software de código abierto, la licencia se considera ransomware.

Definición de Donateware: Donateware es una forma de distribución de un software, que se distribuye como freeware, pero estipula que el usuario haga una donación a

una causa de caridad (generalmente especificada por el autor del programa) para poder "registrar" el programa.

Por lo general, el autor del programa no puede verificar si la donación se ha concretado, pero la idea es que el programa recuerde a sus usuarios sobre la caridad.

Definición de Freeware: Free (gratis) + ware (software). Cualquier software que no requiere pago ni otra compensación (como adwares) por parte de los usuarios que los usan. Que sean gratuitos no significa que se pueda acceder a su código fuente.

Definición de Greenware: Greenware es un tipo de licencia de software que provee al usuario el derecho de usar un programa en particular, si hace algún esfuerzo para ayudar al medio ambiente. La licencia suele sugerir lo siguiente:
-Usar sólo papel reciclado de computadora.
-Reciclar el papel de computadora luego de usarlo.
-Usar el transporte público.
-Reemplazar el auto de 8 cilindros a uno de 4 cilindros.

Definición de Nagware: (software + nag - programa pesado). También llamado begware o annonyware. Nagware es un tipo de shareware que recuerda continuamente al usuario que debe registrarse o pagar por el programa.

La ventana que aparece pidiendo la registración o similar suele ser llamada pantalla nag (nag screen).

Algunos ejemplos de programas nagware son WinRAR, WinZip, mIRC, SmartFTP, etc.

Definición de Postcardware: De postcard + ware (software). Un software cuyos autores requieren como pago por parte de los usuarios el envío de una carta a ellos. Se trata de un tipo de licencia de distribución como lo son el freeware, shareware, etc.

Definición de Ransomware: (hostageware). Originalmente, la palabra ransomware hacía referencia a una licencia de distribución de software, donde su autor requería un monto determinado de dinero para liberar el código fuente del programa. En otras palabras, si se lograban las condiciones que el autor requería (por ejemplo, llegar a un determinado monto de dinero), el código del programa se liberaba. Esto permitía que los desarrolladores pudieran lograr el dinero suficiente como retribución al trabajo que les insumía hacer un programa.

Actualmente el término se utiliza también para hacer referencia a aquellos malwares que "secuestran" archivos y piden "rescate" en dinero por ellos. Por lo general estos programas malignos encriptan la información de algunos archivos considerados importantes para el usuario, y no entregan la clave para lograr desencriptarlos

Definición de Registerware: Registerware hace refencia a aquellos programas que requieren que sus usuarios llenen un formulario con sus datos, si quieren descargarlos o usarlos. Por lo general, estos programas suelen ser también freeware.

Definición de Shareware: De share (compartir) + ware (software). Un tipo de software que es distribuido gratuitamente exclusivamente para ser probado, pero posee restricciones en su funcionalidad o disponibilidad.

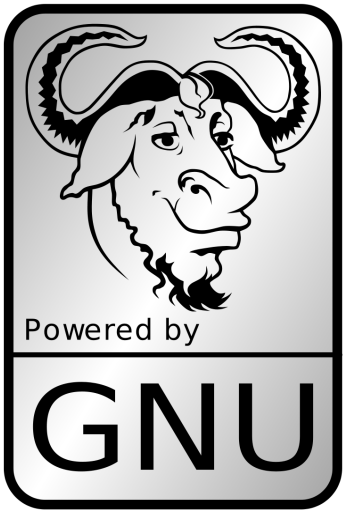
Por lo general son limitados a 30 días de uso, pero también algunos desactivan opciones como "Guardar", o tienen limitado el número de veces que pueden ejecutarse, etc.

En caso de que al usuario le convenza el software podrá comprarlo. El término fue acuñado por Bob Wallace.

Definición de Software de distribución libre: (Freely redistributable software o FRS, libre distribución). Tipo de software que se puede distribuir libremente por cualquiera. Pueden distinguirse dos tipos de software de distribución libre: aquellos que pueden modificarse y distribuirse gratuitamente, llamados software libre. Y aquellos que no pueden modificarse legalmente y pueden ser freeware, shareware o similares (software propietario).

De todas maneras, suele utilizarse más para hacer referencia a aquellos programas que tienen código abierto y pueden modificarse y distribuirse.

Definición de Software libre: Software libre es la designación de un grupo de programas que poseen ciertas libertades y obligaciones que incluyen: libertad de ser usado (tanto el programa como su código), copiado y distribuido por cualquiera. En el caso de la distribución, puede ser licencia tipo BSD (libertad de distribución a código cerrado) o GPL (distribución total, pero bajo las condiciones de tener el código abierto).



Software libre (en inglés ***free software***, aunque en realidad esta denominación también puede significar gratis, y no necesariamente libre, por lo que se utiliza el hispanismo **libre software** también en inglés) es la denominación del software que respeta la libertad de los usuarios sobre su producto adquirido y, por tanto, una vez obtenido puede ser usado, copiado, estudiado, modificado y redistribuido libremente. Según la *Free Software Foundation*, el software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software; de modo más preciso, se refiere a cuatro libertades de los usuarios del software: la libertad de usar el programa, con cualquier propósito; de estudiar el funcionamiento del programa, y adaptarlo a las necesidades; de distribuir copias, con lo cual se puede ayudar a otros, y de mejorar el programa y hacer públicas las mejoras, de modo que toda la comunidad se beneficie (para la segunda y última libertad mencionadas, el acceso al código fuente es un requisito previo)

El software libre suele estar disponible gratuitamente, o al precio de costo de la distribución a través de otros medios; sin embargo no es obligatorio que sea así, por lo tanto no hay que asociar software libre a "software gratuito" (denominado usualmente freeware), ya que, conservando su carácter de libre, puede ser distribuido comercialmente ("software comercial"). Análogamente, el "software gratis" o "gratuito" incluye en ocasiones el código fuente; no obstante, este tipo de software *no es libre* en el mismo sentido que el software libre, a menos que se garanticen los derechos de modificación y redistribución de dichas versiones modificadas del programa.

Tampoco debe confundirse software libre con "software de dominio público". Éste último es aquel software que no requiere de licencia, pues sus derechos de explotación son para toda la humanidad, porque pertenece a todos por igual. Cualquiera puede hacer uso de él, siempre con fines legales y consignando su autoría original. Este software sería aquel cuyo autor lo dona a la humanidad o cuyos derechos de autor han expirado, tras un plazo contado desde la muerte de este, habitualmente 70 años. Si un autor condiciona su uso bajo una licencia, por muy débil que sea, ya no es del dominio público.

Libertades del software libre

De acuerdo con tal definición, el software es "libre" si garantiza las siguientes libertades:

- **Libertad 0:** la libertad de usar el programa, con cualquier propósito.
- **Libertad 1:** la libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a tus necesidades.
- **Libertad 2:** la libertad de distribuir copias del programa, con lo cual puedes ayudar a tu prójimo.
- **Libertad 3:** la libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie.

Las libertades 1 y 3 requieren acceso al código fuente porque estudiar y modificar software sin su código fuente es muy poco viable.

Ciertos teóricos usan este cuarto punto (libertad 3) para justificar parcialmente las limitaciones impuestas por la licencia GNU GPL frente a otras licencias de software libre. Sin embargo el sentido original es más libre, abierto y menos restrictivo que el que le otorga la propia situación de incompatibilidad, que podría ser resuelta en la próxima versión 3.0 de la licencia GNU GPL, causa en estos momentos graves perjuicios a la comunidad de programadores de software libre, que muchas veces no

pueden reutilizar o mezclar códigos de dos licencias distintas, pese a que las libertades teóricamente lo deberían permitir.

En el **sitio web oficial de OSI** está la lista completa de las licencias de software libre actualmente aprobadas y tenidas como tales.

El término software no libre se emplea para referirse al software distribuido bajo una licencia de software más restrictiva que no garantiza estas cuatro libertades. Las leyes de la propiedad intelectual reservan la mayoría de los derechos de modificación, duplicación y redistribución para el dueño del *copyright*; el software dispuesto bajo una licencia de software libre rescinde específicamente la mayoría de estos derechos reservados.

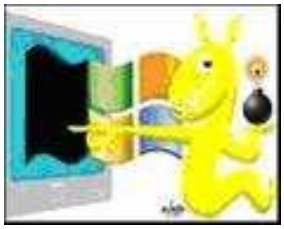
La definición de software libre no contempla el asunto del precio; un eslogan frecuentemente usado es *"libre como en libertad, no como en cerveza gratis"* o en inglés *"Free as in freedom, not as in free beer"* (aludiendo a la ambigüedad del término inglés "*free*"), y es habitual ver a la venta CDs de software libre como distribuciones GNU/Linux. Sin embargo, en esta situación, el comprador del CD tiene el derecho de copiarlo y redistribuirlo. El software gratis puede incluir restricciones que no se adaptan a la definición de software libre —por ejemplo, puede no incluir el código fuente, puede prohibir explícitamente a los distribuidores recibir una compensación a cambio, etc—.

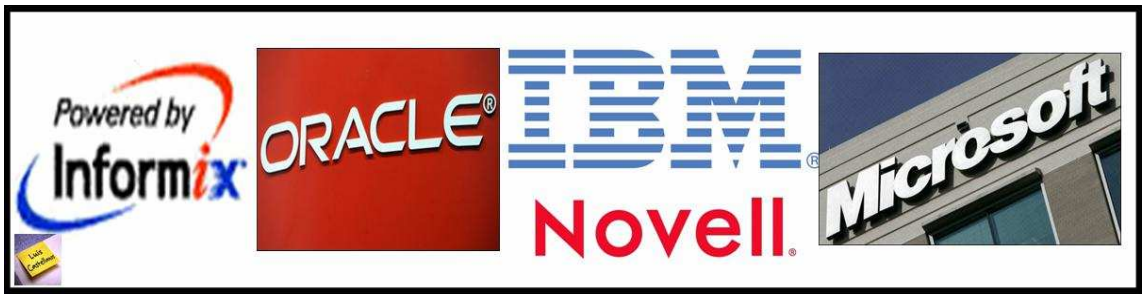
Para evitar la confusión, algunas personas utilizan los términos "libre" (*software libre*) y "gratis" (*software gratis*) para evitar la ambigüedad de la palabra inglesa "free". Sin embargo, estos términos alternativos son usados únicamente dentro del movimiento del software libre, aunque están extendiéndose lentamente hacia el resto del mundo. Otros defienden el uso del término *open source software* (software de código abierto, también llamado de fuentes abiertas). La principal diferencia entre los términos "open source" y "free software" es que éste último tiene en cuenta los aspectos éticos y filosóficos de la libertad, mientras que el "open source" se basa únicamente en los aspectos técnicos.

En un intento por unir los mencionados términos que se refieren a conceptos semejantes, se está extendiendo el uso de la palabra "FLOSS" con el significado de "Free – Libre – Open Source Software" e, indirectamente, también a la comunidad que lo produce y apoya.

El software libre tiene dueño y no es lo mismo que el software de dominio público ni que el freeware.
GPL: General Public License (Licencia Pública General). *Licencia creada por la Free Software Foundation y orientada principalmente a los términos de distribución, modificación y uso de software libre.*
BSD: (Berkeley Software Distribution). *Es una licencia de software libre permisiva como la licencia de OpenSSL o la MIT License*

Ejemplo de Licencia GPL: http://gugs.sindominio.net/licencias/gples.html
En Idioma Ingles: http://www.gnu.org/copyleft/gpl.html





Definición de Software propietario: (proprietary software, software no libre, privativo, privado, con propietario o de propiedad). El software propietario es aquel que posee restriciones en el uso, copia o modificación o cuyo código fuente no está disponible (código cerrado).

Que un software haya liberado su código (código abierto) no implica necesariamente que sea un software libre, sino que puede ser también un software propietario.

Elementos a Considerar para la Adquisición de un Software.

- **Procedimientos establecidos formalmente.**
- **Ajustarse a los requerimientos.**
- **Definir que tipo de Software se requiere.**
- **Tipo: (Mono, Multi, Web).**
- **Lenguaje: (Software Libre y Propietario).**
- **Contratos de Mantenimiento y Desarrollo de nuevos requerimientos.**

Software Libre:

- **Disponibilidad Presupuestaria.**
- **Infraestructura de Hardware y Software adecuado.**
- **Unidad o Departamento de Informática funcional.**
- **Recurso Humano capacitado.**

Software Propietario:

- **Adquisición de Programa Objeto y/o Programa Fuente.**
- **Garantía.**
- **Contratos de Mantenimiento y Desarrollo de nuevos requerimientos.**
- **Recurso Humano capacitado.**

Software de programación

Programa: Colección de instrucciones que un computador puede entender y efectuar; por lo cual debe estar hecho en un lenguaje para el mismo.

Instrucción: Es la parte mas pequeña de un programa, que un computador puede ejecutar. Ejemplo: Read A

Lenguajes de Programación: Lenguaje artificial que puede ser usado para controlar el comportamiento de una máquina, especialmente una computadora. Estos se componen de un conjunto de reglas sintácticas y semánticas que permiten expresar instrucciones que luego serán interpretadas.

Debe distinguirse de “lenguaje informático”, que es una definición más amplia, puesto estos incluyen otros lenguajes como son el HTML o PDF que dan formato a un texto y no es programación en sí misma.

El programador es el encargado de utilizar un lenguaje de programación para crear un conjunto de instrucciones que, al final, constituirá un programa o subprograma informático.

Los lenguajes de programación pueden clasificarse según el paradigma que usan en: procedimentales, orientados a objetos, funcionales, lógicos, híbridos, etc.

Son ejemplos de lenguajes de programación: php, prolog, ASP, ActionScript, ada, python, pascal, c, basic, JAVA, JavaScript, etc.

Tipos de Lenguaje según nivel:

- **Lenguajes de Bajo Nivel:** Se expresan en forma relativamente próxima a aquello que el computador puede ejecutar directamente.
- **Lenguajes de Alto Nivel:** Orientados al Programador.
  (high-level language). Tipo de lenguajes de programación que permite al programador escribir programas (algoritmos) que son más o menos independientes de un tipo particular de computadora (del hardware). Estos

lenguajes son considerados de alto nivel porque son más parecidos al lenguaje natural humano y más lejano al lenguaje de las máquinas.

En contraste, los lenguajes ensamblador son considerados lenguajes de bajo nivel porque están muy cerca al lenguaje que manejan las máquinas.

La principal ventaja de los lenguajes de alto nivel sobre los de bajo nivel, es que son más fáciles de leer, escribir y mantener por humanos. Al final, los programas escritos en alto nivel deben ser traducidos en un lenguaje máquina específico empleando un compilador o un intérprete. De esta manera pueden ser ejecutados por una máquina específica.

Los primeros lenguajes de programación de alto nivel fueron diseñados en los 50. Actualmente existen cientos de lenguajes de este tipo como Ada, Algol, BASIC, COBOL, C, C++, FORTRAN, LISP, Pascal, Prolog, etc.

Traducción de los Lenguajes:

Compiladores e intérpretes.

Compilación: Proceso de traducción de un código fuente (escrito en un lenguaje de programación de alto nivel) a lenguaje máquina (código objeto) para que pueda ser ejecutado por la computadora. Las computadoras sólo entienden el lenguaje máquina. La aplicación o la herramienta encargada de la traducción se llama compilador.

Compilador: (compiler). Los compiladores son programas o herramientas encargadas de compilar. Un compilador toma un texto (código fuente) escrito en un lenguaje de alto nivel y lo traduce a un lenguaje comprensible por las computadoras (código objeto).

Básicamente, existen dos grandes formas de ejecutar programas: programas compilados (previamente pasados por un compilador) y programas interpretados (necesitan pasar por un intérprete para ejecutarse en tiempo real).

Características de un compilador:

Generalmente un compilador se divide en dos partes:

* Front End: parte que analiza el código fuente, comprueba su validez, genera el árbol de derivación y rellena los valores de la tabla de símbolos. Parte que suele ser independiente de la plataforma o sistema operativo para el que funcionará.

* Back End: parte en donde se genera el código máquina exclusivo para una plataforma a partir de lo analizado en el front end.

Por lo general el resultado del back end no puede ser ejecutado directamente, se necesita pasar por un proceso de enlazado (linker).

Existen varios tipos de compiladores: Compiladores cruzados, Compiladores optimizadores, Compiladores de una sola pasada, Compiladores de varias pasadas, Compiladores JIT (Just In Time).

Intérprete: Es un programa de computadora que ejecuta o lleva a cabo instrucciones escritas en un lenguaje de programación.

La interpretación es una de las formas de ejecución de los programas de computadoras, la otra es la compilación.

El término "intérprete" puede hacer referencia al programa que ejecuta el código fuente que acaba de ser traducido a una forma intermedia, o puede hacer referencia al programa que lleva a cabo tanto la traducción como la ejecución.

Intérpretes vs compiladores

Cualquier lenguaje puede ser ejecutado tanto vía intérprete o vía compilador, pero algunos lenguajes suelen asociarse más a una vía que a la otra, y por esto son llamados "lenguajes interpretados" o "lenguajes compilados" respectivamente.

También puede darse que un programa contenga partes que son implementadas vía intérprete y otras vía compilador.

También existen intérpretes que incluyen cierta "compilación" en el medio. Son aquellos que compilan a un código intermedio llamado bytecode, que es más eficiente de ejecutar que hacerlo directamente desde el código fuente.

En general, la principal desventaja de los intérpretes, es que cuando un programa es interpretado, suele ejecutarse más lento que si el mismo programa estuviese compilado. Esto se debe a que el intérprete debe analizar cada sentencia en el programa en cada ejecución (un análisis en tiempo real). También el acceso a variables es más lento en un intérprete, porque mapear los identificadores para almacenar las localizaciones debe hacerse repetidas veces en tiempo real.

Algoritmo: Conjunto ordenados de pasos que se deben seguir para llegar a resolver un problema o efectuar un proceso.

Es un Método para resolver un problema mediante una serie de pasos precisos, definidos y finitos. Un algoritmo es una serie de operaciones detalladas, en otras palabras un algoritmo es un conjunto de reglas para resolver una cierta clase de problemas y se puede formular de muchas formas con el cuidado de que no exista ambigüedad.

Características

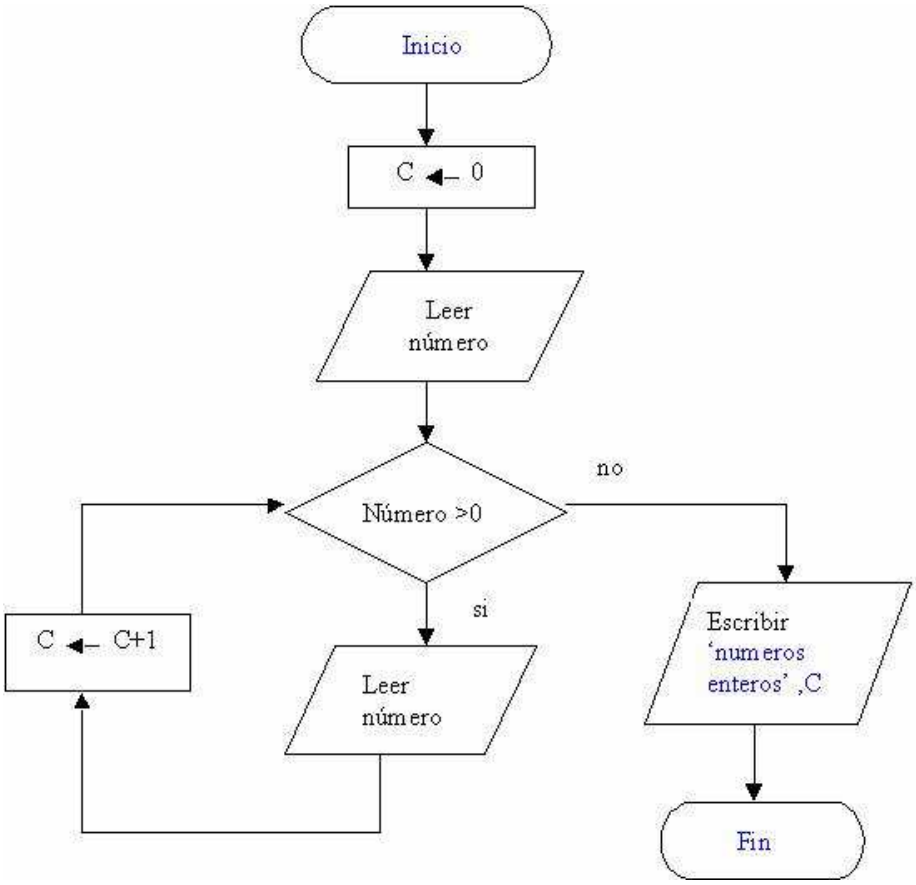
* Preciso (debe indicar el orden de realización en cada paso y no puede tener ambigüedad).
* Definido (si se sigue dos veces, obtiene el mismo resultado cada vez)
* Finito (tiene fin; un número determinado de pasos).
* Debe ser Sencillo, Legible.
* Modular.
* Eficiente y Efectivo.
* Se ha de desarrollar en el menor tiempo posible.
* Correcto.
* Todo Algoritmo debe tener cero ó mas entradas.
* Debe tener al menos una salida y ésta debe ser tangible.

Ejemplo: Contar los números enteros positivos introducidos por teclado. Se consideran dos variables enteras NUMERO y CONTADOR (contará el número de enteros positivos). Se supone que se leen números positivos y se detiene el bucle cuando se lee un número negativo o cero.

Pseudocódigo:

```
Inicio
 contador 0
  Leer (numero)
 Mientras numero > 0 hacer
     contador   contador+1
    Leer (numero)
  Fin_Mientras
 Escribir(\'El número de enteros positivos es : \', contador)
  Fin
```

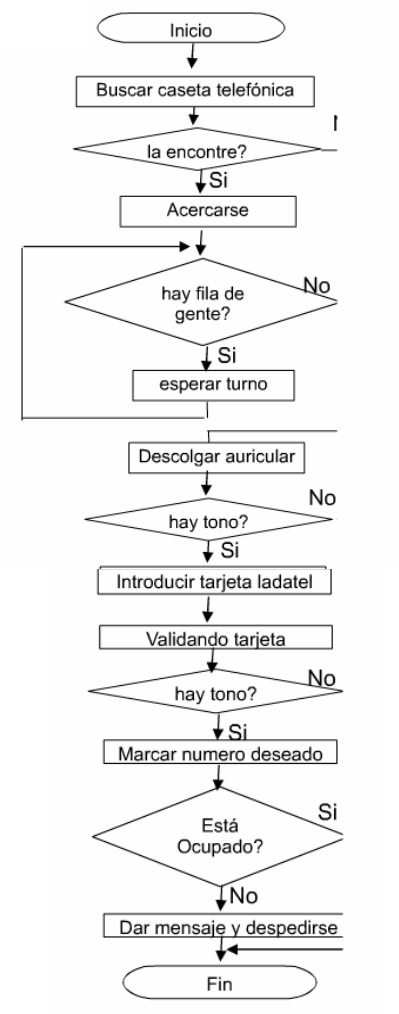
Diagrama de Flujo



Diseñar un algoritmo para hacer una llamada telefónica

Pseudocódigo

1. Buscar una caseta telefónica
2. La encontre? Si paso 3 No paso15
3. Acercarse
4. Hay fila de gente? Si paso 5 No paso 7
5 esperar turno
6 ir al paso 4
7 descolgar auricular
8 hay tono? Si paso 9 No paso15
9 introducir tarjeta ladatel
10 validando tarjeta
11 tiene crédito? Si paso 12 No paso 15
12 marcar numero deseado
13 esta ocupado? Si paso 15 No paso 13
14 dar mensaje y despedirse
15 Fin



Tipos de Lenguajes de Alto Nivel:

Lenguajes Estructurados: El lenguaje estructurado es un lenguaje natural limitado en palabras y construcciones, lo que le da más precisión y claridad, evitando ambigüedades (el lenguaje natural humano carece de precisión y es muy ambiguo).

El lenguaje estructurado puede utilizarse para especificar un algoritmo. Luego, para que la computadora pueda procesarlo, deberá transformarse o "traducirse" a un lenguaje de programación específico.

El lenguaje estructurado es una herramienta que puede utilizarse en la especificación de procesos, en el desarrollo de sistemas.

Lenguaje de marcas: (lenguaje de marcado, markup languaje). Tipo de lenguaje que combina texto con información extra acerca del texto. Esa información extra se entremezcla con el texto primario. El lenguaje de marcas más conocido en la actualidad es el HTML, que se utiliza en las páginas web.

Las marcas también están formadas de texto, pero que es interpretado cuando se muestra el documento, y suelen llamarse también etiquetas.

Existen tres clases de lenguajes de marcas, y pueden presentarse todas en un mismo documento.

*** Marcas de presentación:** estas marcas indican el formato-marco del texto. Su uso comienza a reducirse dado que es poco flexible, especialmente en grandes proyectos.

*** Marcas de procedimientos**: estas marcas se utilizan para la presentación del texto, interpretándose cada una en el orden que en aparecen. Por ejemplo, la marca que se agrega inmediatamente antes de un texto para que se vea en negrita. Luego debe existir la marca correspondiente que termine o cierre la negrita. Otras marcas de procedimientos pueden ser centrar texto, cambio de tamaño de fuente, cambios de estilos, etc.

Algunos lenguajes de marcas de procedimiento son nroff, troff, TeX, PostScript, HTML, etc.

*** Marcas descriptivas:** También llamadas marcado descriptivo, o semántico. Aquí se utilizan las marcas para describir fragmentos de texto sin especificar cómo deben representarse. Algunos lenguajes diseñados para esto son el SGML y el XML.

En los lenguajes de marcas descriptivas el formato está separado del contenido, permitiendo flexibilidad a la hora de reformatear un texto.

Un ejemplo es una marca que indique la hora actualizada, pero no indica como se representará.

Algunos ejemplos de lenguajes de marcas son:

Darwin Information Typing Architecture (DITA)
•DocBook
•Extensible HyperText Markup Language (XHTML)
•Extensible Markup Language (XML)
•Generalized Markup Language (GML)
•HyperText Markup Language (HTML)
•Lilypond (sistema para notación musical)
•Maker Interchange Format (MIF)
•Mathematics Markup Language (MathML)
•Microsoft Assistance Markup Language (MAML)
•Music Extensible Markup Language (MusicXML)
•Rich Text Format (RTF)
•S1000D (Especificación internacional para documentación técnica relacionada al área comercial y militar).
•TeX, LaTeX (utilizado generalmente en matemáticas y publicaciones académicas).
•Text Encoding Initiative (TEI). (formato XML para publicaciones digitales)
•Wireless Markup Language (WML), Wireless TV Markup Language (WTVML)
•XHTML Basic (subconjunto de XHTML para dispositivos portátiles, para reemplazar a WML, XHTML MP y C-HTML).

Lenguaje scripting (Scripting language, lenguaje de guión). Un lenguaje scripting es un tipo de lenguaje de programación que es generalmente interpretado.

Se los programas compilados, porque los programas son convertidos de forma permanente a un código especial antes de que puedan ejecutarse (proceso de compilación). En cambio los scripts permanecen en su forma original (su código fuente en forma de texto) y son interpretados comando por comando cada vez que se ejecutan. De todas maneras, los scripts pueden ser compilados también, aunque no es usual.

Características de los lenguajes scripting

* Los scripts suelen escribirse más fácilmente, pero con un costo sobre su ejecución.
* Suelen implementarse con intérpretes en lugar de compiladores.
* Tienen fuerte comunicación con componentes escritos en otros lenguajes.
* Los scripts suelen ser almacenados como texto sin formato.
* Los códigos suelen ser más pequeños que el equivalente en un lenguaje de programación compilado.

Tipos de lenguajes de scripting

*** Lenguaje de control de tareas y shells: Interfaz de Línea de Comandos**, Ejemplos: cmd.exe (Windows NT, Windows CE, OS/2), COMMAND.COM (DOS, Windows 9x), csh, AppleScript, sh, JScript vía Windows Script Host, VBScript, entre otros.

*** GUI Scripting**: son lenguajes de scripting diseñados para interactuar con los elementos de las interfaces gráficas como botones, ventanas, menús, etc. Se utilizan

para automatizar acciones repetitivas. También son llamados "lenguajes macro". ***Ejemplos:*** AutoHotkey, AutoIt, Expect, Automator, etc.

* **Lenguaje scripting de aplicaciones específicas**: en grandes aplicaciones a veces es necesario incluir un lenguaje de scripting que se adapte a ésta y dé más posibilidades a sus usuarios. ***Ejemplos:*** ActionScript en Flash, MATLAB, mIRC script, QuakeC, etc.

*** De programación web:** los lenguajes scripting para webs suelen servir para crear páginas dinámicas. De todas maneras, muchos de ellos son tan potentes como para poder crear programas de propósito general. Pueden diferenciarse en dos categorías:

- Del lado del servidor: PHP, ASP (Active Server Pages), Java Server Pages, ColdFusion, IPTSCRAE, Lasso, MIVA Script, SMX, XSLT.

- Del lado del cliente: JavaScript, JScript, VBScript, Tcl.

*** Lenguajes de procesamiento de textos:** muchos lenguajes de scripting comenzaron de esta manera. Permitían automatizar tareas al administrador cuando se trataba de configuraciones basadas en textos (usual en Unix por ejemplo). En el caso de Perl, comenzó como un lenguaje de generación de reportes, pero terminó siendo un completo lenguaje para desarrollar aplicaciones. **Ejemplos:** AWK, Perl, sed, XSLT.

*** Lenguajes dinámicos de propósito general:** Algunos lenguajes, como Perl, comenzaron como lenguajes de scripting para se desarrollaron como completos lenguajes de programación de propósito general. Otros lenguajes similares han sido descritos como "lenguajes scripting" por situaciones similares, aunque luego se hayan usado más para programar aplicaciones. **Ejemplos:** APL, Boo, Dylan, Ferite, Groovy, Io, Lisp, Lua, MUMPS (M), newLISP, Nuva, Perl, PHP, Python, Ruby, Scheme, Smalltalk, SuperCard, Tcl, Revolution.

Lenguaje Orientado a Objetos: Es un lenguaje de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de ordenador. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo y encapsulamiento. Su uso se popularizó a principios de la década de 1990. Actualmente son muchos los lenguajes de programación que soportan la orientación a objetos.

Hay un cierto acuerdo sobre exactamente qué características de un método de programación o lenguaje le definen como "orientado a objetos", pero hay un consenso general en que las características siguientes son las más importantes:

- **Abstracción:** Denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos y cuando lo están, una variedad de técnicas son requeridas para ampliar una abstracción.
- **Encapsulamiento:** Significa reunir a todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión de los componentes del sistema. Algunos autores confunden este concepto con el principio de ocultación, principalmente porque se suelen emplear conjuntamente.
- **Principio de ocultación:** Cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una interfaz a otros objetos que

específica cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas, solamente los propios métodos internos del objeto pueden acceder a su estado. Esto asegura que otros objetos no pueden cambiar el estado interno de un objeto de maneras inesperadas, eliminando efectos secundarios e interacciones inesperadas. Algunos lenguajes relajan esto, permitiendo un acceso directo a los datos internos del objeto de una manera controlada y limitando el grado de abstracción. La aplicación entera se reduce a un agregado o rompecabezas de objetos.

- **Polimorfismo:** comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre, al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica. Algunos lenguajes proporcionan medios más estáticos (en "tiempo de compilación") de polimorfismo, tales como las plantillas y la sobrecarga de operadores de C++.
- **Herencia:** las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo. Esto suele hacerse habitualmente agrupando los objetos en clases y estas en árboles o enrejados que reflejan un comportamiento común. Cuando un objeto hereda de más de una clase se dice que hay herencia múltiple.
- **Recolección de basura:** la Recolección de basura o Garbage Collector es la técnica por la cual el ambiente de Objetos se encarga de destruir automáticamente, y por tanto desasignar de la memoria, los Objetos que hayan quedado sin ninguna referencia a ellos. Esto significa que el programador no debe preocuparse por la asignación o liberación de memoria, ya que el entorno la asignará al crear un nuevo Objeto y la liberará cuando nadie lo esté usando. En la mayoría de los lenguajes híbridos que se extendieron para soportar el Paradigma de Programación Orientada a Objetos como C++ u Object Pascal, esta característica no existe y la memoria debe desasignarse manualmente.

Autogeneradores y Asistentes de Software: Wizard en inglés. Aplicación al servicio del usuario (especialmente inexpertos) que generalmente abrevia los pasos a seguir para realizar una tarea o, por lo menos, las explica muy bien. Los asistentes hacen más sencillas las tareas de instalar dispositivos, programas o realizar ciertas tareas.

Base de datos: (database). Almacén de datos relacionados con diferentes modos de organización. Una base de datos representa algunos aspectos del mundo real, aquellos que le interesan al diseñador. Se diseña y almacena datos con un propósito específico. Con la palabra "datos" se hace referencia a hechos conocidos que pueden registrarse, como ser números telefónicos, direcciones, nombres, etc.

Las bases de datos almacenan datos, permitiendo manipularlos fácilmente y mostrarlos de diversas formas.

El proceso de construir una base de datos es llamado diseño de base de datos.