1. CONCEPTOS BÁSICOS DE COMPUTACIÓN

1.1. SISTEMA O PERATIVO

Un sistema operativo es un conjunto de programas que permiten utilizar los recursos de la computadora. Es decir, sirve como el intermediario entre el usuario de una computadora y el hardware de ésta.

Un sistema operativo tiene como objetivos básicos el poder ejecutar programas de usuario, ser amigable y garantizar una cierta eficiencia. Además proporciona servicios tales como:

- Asignar recursos de la Computadora a los programas (memoria, CPU, etc.).
- Proporcionar acceso a los dispositivos de la computadora y sus periféricos
- Proporcionar un sistema organizado de almacenamiento de datos
- Establecer comunicación interactiva con el usuario

Como ejemplo de sistemas operativos tenemos:

- MS-DOS
- DR-DOS
- OS/2
- UNIX (Solaris, IRIX, etc)
- LINUX
- Windows (NT, 2000, XP, Vista, 7, etc.)
- VMS (Vax)
- MAC-OS
- Novell (Sistemas Operativos de red)
- MOSIX, Amoeba, Mach (Sistemas Operativos Distribuidos)

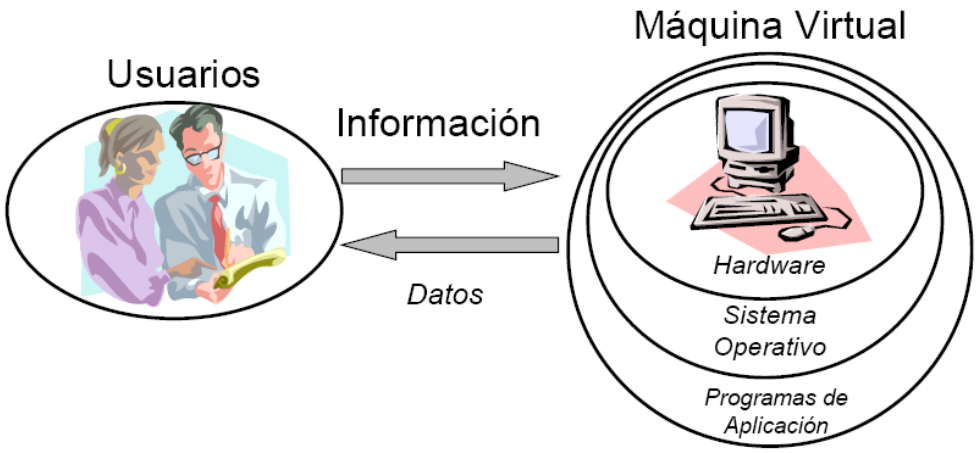
Un Sistema Operativo debe ser:

1. Eficiente, es decir, no debe desperdiciar tiempo útil y debe realizar sus funciones de una manera rápida
2. Fiable, ya que un fallo de él, puede causar que el sistema “se caiga”
3. Debe ser de tamaño pequeño

1.1.1. Funciones principales de un Sistema Operativo

Las funciones principales de un Sistema Operativo son tres:

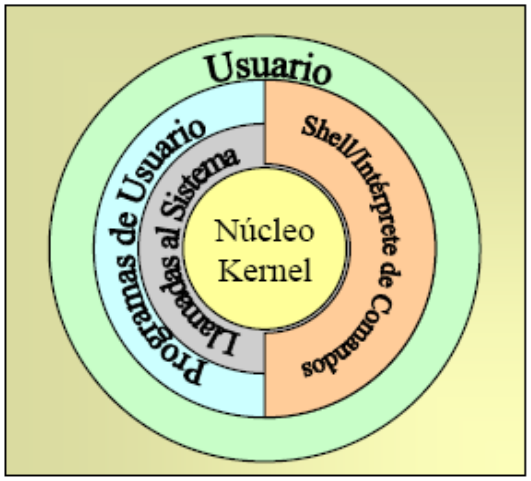
1. Asignación de recursos de la computadora al usuario
2. Contar con un programa de control para manipular el hardware de la computadora y la entrada y salida de datos
3. Facilitar la ejecución de tareas de los usuarios



1.1.2. Elementos de un Sistema Operativo

Un Sistema Operativo se compone de:

- Un Núcleo o Kernel que representa el corazón de éste.
- Un Programa llamado intérprete de comandos o Shell que se encarga de ejecutar las llamadas al sistema hechas por el usuario o por otros programas en ejecución
- Los programas de usuario y de aplicación que son con los que trabaja el usuario de una computadora



COMPONENTES

1. Gestión de Procesos
2. Gestión de Memoria Principal
3. Gestión de Archivos
4. Gestión del Sistema de E/S
5. Gestión de Almacenamiento Secundario
6. Trabajo con Redes
7. Sistema de Protección
8. Sistema de Interpretación de Órdenes

SERVICIOS

1. Ejecución de Programas
2. Operaciones de E/S
3. Manipulación del Sistema de Archivos
4. Comunicaciones
5. Detección de errores
6. Asignación de recursos
7. Contabilización
8. Protección

LLAMADAS AL SISTEMA

1. Control de Procesos y Tareas
2. Manipulación de archivos
3. Manipulación de Dispositivos
4. Mantenimiento de Información
5. Comunicaciones

PROGRAMAS DEL SISTEMA

1. Manipulación de archivos
2. Información de estado
3. Modificación de archivos
4. Apoyo a lenguajes de Programación
5. Carga y Ejecución de Programas
6. Comunicaciones

Un sistema operativo debe contar además con programas de apoyo al usuario que permitan realizar operaciones como:

a) EDITAR

Durante el desarrollo de un programa o la escritura de un documento, por lo general, resulta necesario efectuar correcciones, agregar módulos, etc. Para evitar el tener que volver a teclear grandes porciones de texto o instrucciones de un programa, se acostumbra mantener el documento que se esta creando en almacenamiento secundario de la computadora (disco externo, memoria USB, disco duro, etc.). Al programa que se

utiliza para escribir un programa por primera vez, utilizando algún lenguaje de programación para ello, y en general cualquier texto como manuales, cartas, etc., y poder realizar correcciones en ellos, recibe el nombre de EDITOR.
Un editor nos permite efectuar operaciones en los documentos o programas que se escriben como: Eliminar parte, reemplazar partes, insertar partes, etc.

b) TRANSFERIR INFORMACIÓN

Un Sistema Operativo debe ser capaz de permitir transferir información de memoria principal de la computadora a memoria secundaria y viceversa. También debe ser capaz de sacar respaldos de información a discos u otros medios externos de almacenamiento.
La información en un disco se organiza mediante archivos. Un archivo es una colección de información relacionada entre sí y es comparable a una o varias hojas de papel en un archivero convencional. Todos los programas, textos, imágenes, etc., en un disco, residen en archivos.

Un sistema operativo debe ser capaz de desplegar los nombres de los archivos almacenados en un disco, así como sus contenidos. Debe permitir el borrado de archivos que ya no se utilicen. También deberá permitir el borrado de archivos que ya no se usen y permitir la impresión del contenido de un archivo, etc.

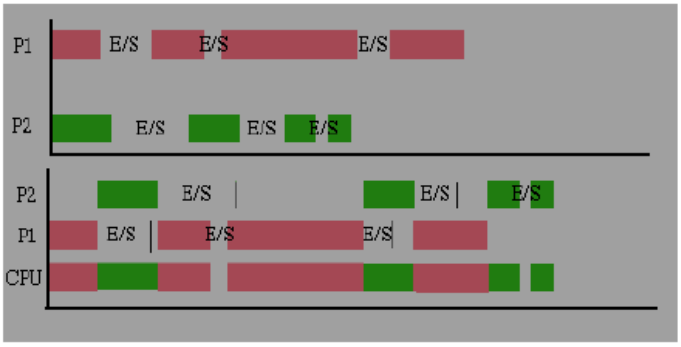
c) EJECUTAR PROGRAMAS

Mediante el Sistema Operativo debe ser posible ejecutar un programa que ya se encuentre traducido a lenguaje de máquina, pues los programas se crean utilizando lenguajes de programación entendibles para los programadores pero inicialmente no para la computadora. El programa se carga en la memoria principal de la máquina antes de ser ejecutado.

1.1.3. Clasificación de los Sistemas Operativos

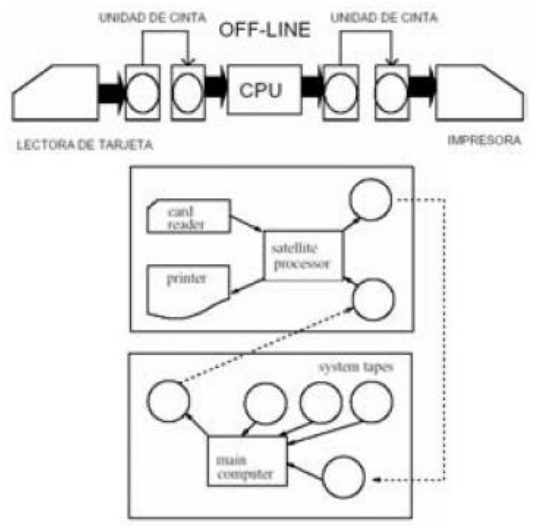
Existen además de los Sistemas Operativos Monousuario que ejecutan Una tarea a la vez atendiendo a un solo usuario a la vez, otras categorías de Sistemas Operativos como son:

1. **Sistemas Operativos Multitarea o Multiprogramación**, los cuales son capaces de ejecutar más de un programa a la vez. Estos se basan en técnicas de multiprogramación y son los más extendidos en la actualidad.

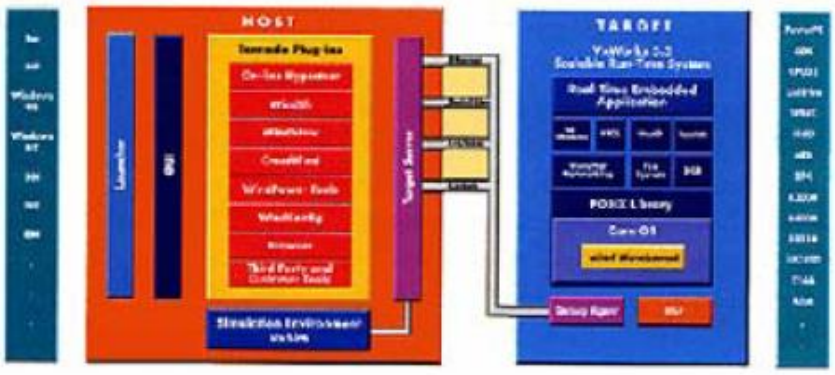


2. **Sistemas Operativos Multiusuario**, los cuales son sistemas que permiten que más de un usuario acceda al sistema de manera simultánea. Naturalmente dicho sistema debe ser también multitarea ya que cada usuario podrá ejecutar varios programas a la vez. UNIX es un ejemplo de este tipo de Sistemas Operativos.

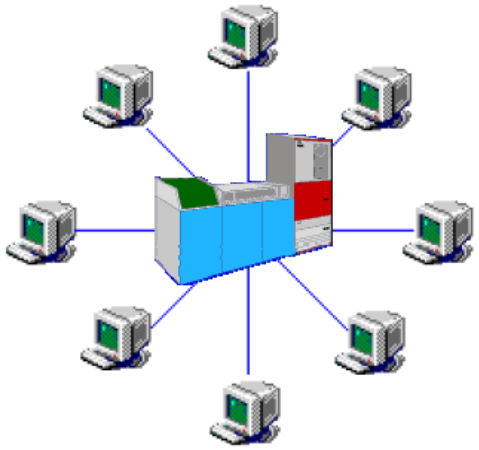
3. **Sistemas operativos por lotes**, Procesan una gran cantidad de trabajos con poca o ninguna interacción entre los usuarios y los programas en ejecución.



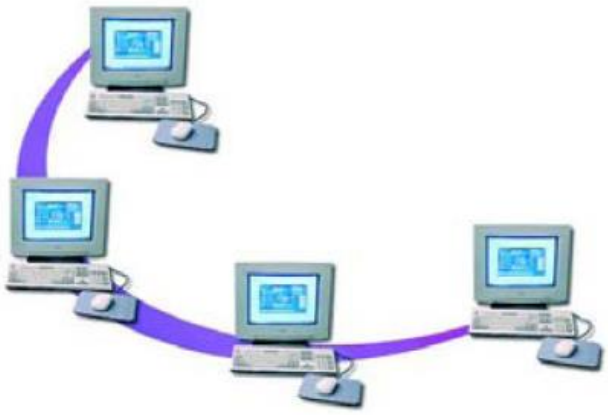
4. **Sistemas Operativos de Tiempo Real**, Son aquellos en los cuales no tiene importancia el usuario, sino los procesos. Por lo general, están subutilizados sus recursos con la finalidad de prestar atención a los procesos en el momento que lo requieran. se utilizan en entornos donde son procesados un gran número de sucesos o eventos.



5. **Sistemas Operativos de Tiempo Compartido**, Permiten la simulación de que el sistema y sus recursos son todos para cada usuario. El usuario hace una petición, se procesa tan pronto como le es posible, y la respuesta aparecerá en terminal.

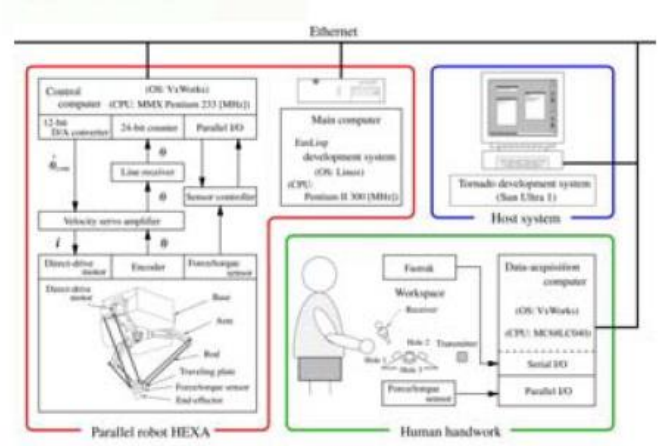


6. **Sistemas Operativos Distribuidos**, Permiten distribuir trabajos, tareas o procesos, entre un conjunto de procesadores. Puede ser que este conjunto de procesadores esté en un equipo o en diferentes, en este caso es trasparente para el usuario.



7. **Sistemas Operativos de Red**, Son aquellos sistemas que mantienen a dos o más computadoras unidas a través de algún medio de comunicación (físico o no), con el objetivo primordial de poder compartir los diferentes recursos y la información del sistema.

8. **Sistemas Operativos Paralelos**, Se pretende que cuando existan dos o más procesos que compitan por algún recurso se puedan realizar o ejecutar al mismo tiempo.



9. **Sistemas Operativos Multiprocesador**. Existen sistemas que tienen dos o más procesadores interconectados, trabajando simultáneamente. En este caso, el Sistema Operativo debe ser capaz de administrar el reparto del trabajo entre los distintos procesadores para sacar provecho del paralelismo existente. Ejemplos son LINUX, UNIX, WINDOWS-NT, MAC-OS.

1.2. COMPONENTES DE UNA COMPUTADORA TIPICA (Unidades de Almacenamiento)

La **Unidad de Control (CU)** es el componente básico de un sistema de cómputo, y con mucho el más importante, ya que esta encarga del control de la operación de todos los demás componentes.

El segundo componente básico es la **unidad aritmético-lógica (ALU)**, que es en donde, como su nombre lo indica, se efectúan todas las operaciones aritméticas, como la suma, resta, etc., y las operaciones lógicas, como por ejemplo la comparación de dos valores.

A la combinación de la **CU** y el **ALU** se le conoce como **CPU (Central Processor Unit), Unidad de Procesamiento Central**. Por ejemplo, tenemos los procesadores de INTEL como el PENTIUM, Celeron, Xeon; los procesadores de Motorola como el 68000 y la familia de AMD como el Atlon, Duron, etc.

Todo sistema de cómputo debe contar además con dispositivos de entrada y salida que le permitan precisamente establecer contacto con el mundo exterior. Tenemos como ejemplos de estos al teclado, monitor, ratón, las impresoras, el CD-ROM.

Cada computadora tiene una determinada cantidad de almacenamiento interno denominado Memoria Principal. Esta memoria es la que opera a mayor velocidad. Para que un programa pueda ser ejecutado, éste debe ser almacenado en la memoria principal, la cual esta formada por multitud de celdas o posiciones (palabras de memoria) de un determinado número de bits y numeradas en forma consecutiva. A la numeración de las celdas se le denomina Dirección de Memoria y es mediante esta dirección que se puede acceder de forma directa a cualquiera de ellas. Decimos por ello que la memoria principal es una Memoria de Acceso Directo.

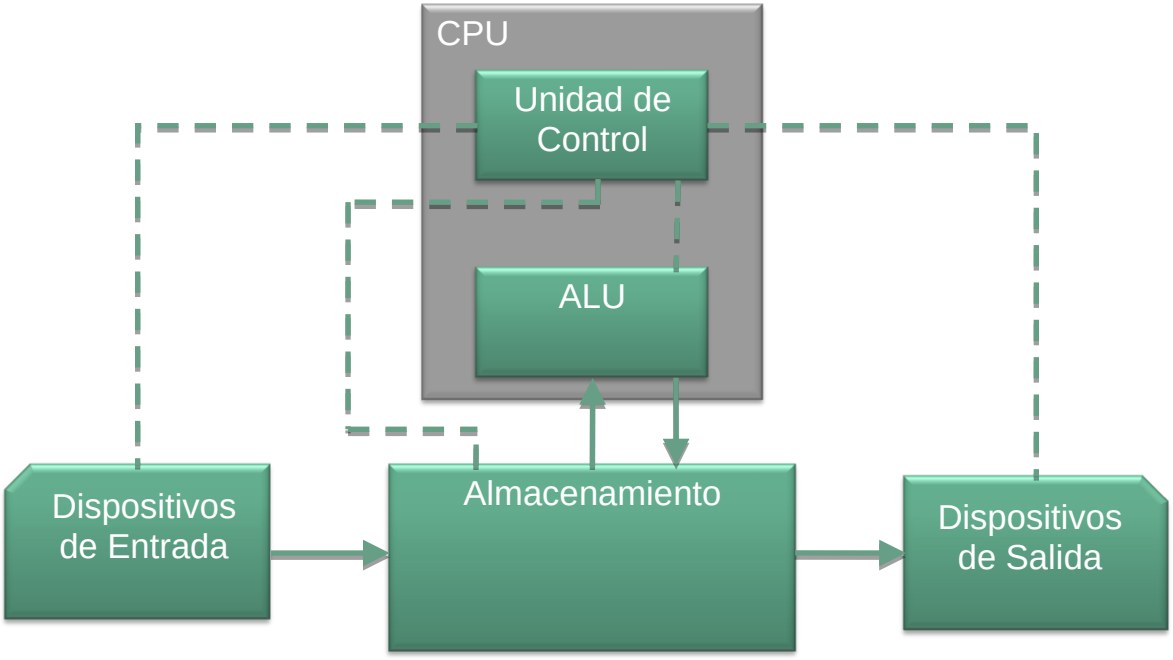
Existen tres tipos de memoria interna: la memoria **ROM (Read Only Memory)** en la que sólo se permite leer y es permanente, es decir, al apagar la máquina no se pierde la información que en ésta se tiene. En algunos casos el contenido de esta memoria esta permanentemente grabado desde que se fabricó; pertenecen a este grupo las memorias programables de sólo lectura **PROM (Programmable Read Only Memory)**, las cuales pueden borrarse y programarse de nuevo si se cuenta con el equipo indicado.

El segundo tipo de memoria es la **RAM (Random Access Memory)** o de acceso aleatorio, la cual, al momento de ser apagado el equipo pierde la información que almacenaba.

Y el tercer tipo es la **Memoria FLASH**, la cual se ha hecho popular en las memorias USB. En ella se puede escribir y leer de manera directa y no es volátil, es decir, al interrumpirse la energía en el sistema de cómputo, mantiene su contenido.

Por otro lado esta la memoria secundaria o memoria externa, la cual permite resolver las deficiencias de la memoria principal en cuanto a volatilidad y pequeña capacidad, ya que aunque la memoria interna es muy rápida, no tiene gran capacidad. Tenemos aquí los discos duros, los CDs, las cintas magnéticas, etc., la información almacenada en este tipo de memoria no es volátil y puede ser borrada por el usuario.

El CPU opera con las instrucciones de control que proporciona un programador, las cuales deben residir en memoria principal. El CPU es el encargado de hacer que los datos necesarios para la ejecución de un programa, sean leídos mediante los dispositivos de entrada, almacenados en memoria, llevados y operados en el ALU y mostrar los resultados en algún dispositivo de salida.



El CPU puede entender instrucciones, solamente en lenguaje de máquina, el cual es por excelencia binario, es decir, en términos de ceros y unos. Cada CPU tiene circuitos especiales diseñados para ejecutar ciertas instrucciones en particular. La dificultad para programar en lenguaje de máquina incentivó el desarrollo de estructuras más simples y accesibles que se conocen como Lenguajes de Programación.

1.3. TRADUCTORES E INTÉRPRETES

Para que una computadora pueda ejecutar un programa (escrito en un determinado lenguaje de programación), es necesario que dicho programa sea traducido a un lenguaje que la computadora entienda, es decir, a lenguaje de máquina el cual esta totalmente apegado a los circuitos (hardware) de la máquina y muy alejado del lenguaje que los seres humanos utilizamos para comunicarnos. Es por ello que existe un conjunto de aplicaciones llamadas Software de base que nos permite hacer dicha traducción sin necesidad de comprender como se conforma dicho lenguaje de máquina. Ejemplo de ello son los Traductores e intérpretes.

1.3.1. Ensambladores y Macro ensambladores

El lenguaje Ensamblador es un primer intento de sustituir el lenguaje de máquina por uno o más cercano a nosotros los humanos. Cuando se asocia un mnemónico a una instrucción de máquina, tenemos lo que se conoce como Lenguaje Ensamblador. En un lenguaje ensamblador el direccionamiento también es simbólico, es decir, en lugar de utilizar direcciones binarias absolutas para accesar a los datos, éstos pueden ser identificados

usando nombres como SUMA: X,A,B. Además se permite el uso de comentarios, lo cual hace que los programas sean más entendibles.

Un programa llamado ensamblador traduce las instrucciones en lenguaje ensamblador a lenguaje de máquina. A la entrada de un programa ensamblador se reconoce como programa fuente y a la salida como programa objeto.
Sin embargo, este tipo de lenguajes presenta la mayoría de los inconvenientes del lenguaje de máquina, ya que su conjunto de instrucciones es muy reducido y rígido. No hay portabilidad ya que hay una fuerte independencia con el Hardware de la computadora. La ventaja es que permite el uso óptimo de los recursos de la máquina.

Para resolver las limitaciones de este tipo de lenguajes se desarrollan unos ensambladores especiales, los macro ensambladores.

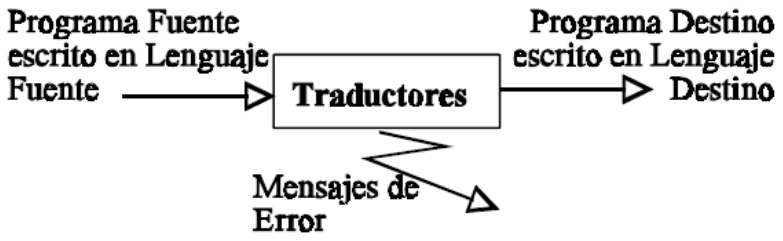
Al hacer un programa en lenguaje ensamblador se encuentra uno a veces con la necesidad de repetir algunas partes de código. Se llaman Macroinstrucciones o simplemente Macros, a las abreviaturas para un grupo de instrucciones. Una sola instrucción representa un bloque de código.

Un Macro ensamblador es un programa que traduce un programa de Macroinstrucciones a lenguaje de máquina.
Con el fin de hacer la programación independiente de la máquina en la que se esté programando, surgen los llamados lenguajes de programación de alto nivel. Estos usan frases relativamente fáciles de entender y están más apegados al lenguaje de los problemas que se requieren resolver. Un lenguaje de alto nivel es independiente de la arquitectura de la máquina y entonces se necesita de un Traductor que traduzca las instrucciones de un lenguaje de alto nivel a lenguaje de máquina.

En este caso una instrucción en lenguaje de alto nivel da lugar a varias instrucciones en lenguaje de máquina. Existe una gran cantidad de lenguajes de alto nivel y para su traducción se requiere de compiladores e intérpretes.

1.3.2. Compiladores

Es un programa que acepta un programa fuente en un lenguaje de alto nivel y produce su correspondiente programa objeto (programa ya en lenguaje de máquina).



Algunos compiladores traducen sólo programas completos, mientras que otros traducen partes de un programa. Se puede en este caso dividir un programa en módulos, donde un módulo es la unidad más pequeña de software que resuelve un subproblema de un problema general que se quiere resolver.

El programa principal controla todo lo que sucede y los submódulos o subprogramas se ejecutan, devolviendo el control al programa principal. Cada submódulo es independiente y solamente tiene acceso directo al módulo al que llama y sus submódulos. Cada módulo puede compilarse e incluso probarse de manera independiente. Se necesita entonces de un Ligador que una los módulos traducidos en un solo programa.

1.3.3. Intérpretes

Un lenguaje de alto nivel además de ser compilado, puede ser interpretado. Un intérprete es un programa que traduce, al igual que un compilador, programas escritos en un lenguaje de alto nivel a lenguaje de máquina; sin embargo en este caso, no existe independencia entre la fase de traducción y la de ejecución. Un intérprete traduce una instrucción o bloque lógico de un programa escrito en un lenguaje de alto nivel a lenguaje de máquina e inmediatamente se ejecuta. A continuación se ejecuta la siguiente instrucción o bloque de manera continua hasta llegar al final del programa fuente. Como ejemplo clásico tenemos los intérpretes del lenguaje BASIC.

Un intérprete no traduce el programa en un solo paso, sino que traduce y ejecuta cada instrucción o bloque lógico antes de traducir y ejecutar la siguiente.

1.3.4. Cargadores

Un cargador es un programa que carga (sube) un programa objeto a memoria principal y lo prepara para su ejecución.

1.4. LENGUAJE DE PROGRAMACIÓN

1.4.1. Definición

Un lenguaje de Programación es un conjunto de símbolos, junto con un conjunto de reglas para combinar dichos símbolos que se usan para expresar programas. Los lenguajes de programación, como cualquier lenguaje se compone de un léxico (conjunto de símbolos, instrucciones o vocabulario permitido) y una sintaxis (reglas que indican como realizar las construcciones correctas del lenguaje), y una semántica (reglas que permiten determinar el significado de cualquier construcción del lenguaje)

Un lenguaje de programación permite entonces especificar de manera precisa sobre qué datos debe operar una computadora, cómo estos datos deben ser operados o transmitidos y que acciones debe tomar bajo una variada gamma de circunstancias. Todo esto a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural.

1.4.2. Clasificación

Los lenguajes de programación se pueden clasificar a través de varios criterios, como son:

- Según el nivel de abstracción
- Según la forma de ejecución
- Según el paradigma de programación que posee cada uno de ellos

SEGÚN EL NIVEL DE ABSTRACCIÓN:

- **Lenguajes de máquina:** Los lenguajes de máquina están escritos en códigos (código de máquina) directamente inteligibles por la máquina (computadora), siendo sus instrucciones cadenas binarias (0 y 1). Lenguaje de máquina hace referencia al lenguaje específico de una computadora, mientras que código de máquina hace referencia al modo en que se escriben los diferentes lenguajes de máquina.
- **Lenguajes de bajo nivel:** Son lenguajes de programación que se acercan al funcionamiento de una computadora. Los lenguajes de más bajo nivel son los lenguajes de máquinas. Ejemplo de este tipo de lenguajes es el lenguaje ensamblador el cual trabaja con los registros de la memoria de la computadora de forma directa.
- **Lenguajes de nivel medio:** En una pequeña medida se diferencian algunos lenguajes de nivel medio, como el lenguaje C, ya que tienen ciertas características que los acercan a los lenguajes de bajo nivel, como el uso de apuntadores de memoria y registros, pero con sintaxis, vocabulario y gramática de alto nivel.
- **Lenguajes de alto nivel:** Estos lenguajes se caracterizan por expresar los algoritmos de una manera adecuada a la capacidad cognitiva humana, en lugar de estar orientados a su ejecución en las máquinas. Los lenguajes de alto nivel requieren de conocimientos específicos de programación y del lenguaje concreto (vocabulario, gramática y sintaxis) para realizar las secuencias de instrucciones lógicas. Ejemplo de ello son los lenguajes C, y JAVA.

SEGÚN LA FORMA DE EJECUCIÓN:

Todo programa escrito en un lenguaje de alto nivel puede ser ejecutado de dos maneras:

- **Lenguajes compilados:** Antes de poder utilizarse un programa debe usarse un traductor llamado compilador que como ya se dijo antes, se encarga de traducir (compilar) el programa original (código fuente) al programa equivalente escrito en lenguaje de máquina o ensamblador (binario). Los binarios son los programas ejecutables y los únicos necesarios para el funcionamiento del programa.
- **Lenguajes Interpretados:** Cada vez que se usa el programa debe utilizarse un traductor llamado intérprete, que se encarga de traducir (interpretar) las instrucciones del programa original (código fuente) a código de máquina según van siendo utilizadas. Para el funcionamiento del programa siempre es necesario disponer del código original y del intérprete.

SEGÚN EL PARADIGMA DE PROGRAMACIÓN:

Un paradigma de programación representa un enfoque particular o filosofía para la construcción del software. Los diferentes paradigmas de programación son:

- **Algorítmico, imperativo o por procedimientos:** Describe la programación en términos del estado del programa y sentencias que cambian dicho estado. Los programas imperativos son un conjunto de instrucciones que le indican a la computadora como realizar una tarea. Lenguajes ejemplos son C, BASIC o PASCAL
- **Declarativo o Predicativo:** Basado en el uso de predicados lógicos (lógico) o funciones matemáticas (funcional). El objetivo es conseguir lenguajes expresivos en los que no sea necesario especificar cómo resolver el problema, sino qué problema se desea resolver. Los intérpretes de los lenguajes imperativos tienen incorporado un motor de inferencia genérico que resuelve los problemas a partir de su especificación. Ejemplo de un lenguaje lógico es PROLOG mientras que LISP es un lenguaje funcional.
- **Orientado a Objetos:** Paradigma cada vez más utilizado y en combinación con el imperativo. Se basan en la creación de objetos tratando de que su concepción sea lo más cercana a la realidad tangible. Utilizan técnicas de herencia, polimorfismo, encapsulación, modularidad, reuso, instanciación, ocultamiento, etc. Ejemplos de lenguajes orientados a objetos son: C++, JAVA y Python