FUNDAMENTOS INFORMÁTICOS

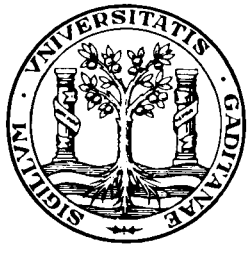
José Galindo Gómez
Pedro J. Sánchez Sánchez
Andrés Yáñez Escolano
María José del Jesus Díaz
José Joaquín Aguilera García
José María Rodríguez Corral
Alfredo Sánchez Navarro
José Fidel Argudo Argudo

2ª EDICIÓN

SERVICIO DE PUBLICACIONES
UNIVERSIDAD DE CÁDIZ

FUNDAMENTOS INFORMÁTICOS

José Galindo Gómez
Pedro J. Sánchez Sánchez
Andrés Yáñez Escolano
María José del Jesus Díaz
José Joaquín Aguilera García
José María Rodríguez Corral
Alfredo Sánchez Navarro
José Fidel Argudo Argudo



UNIVERSIDAD DE CÁDIZ
Servicio de Publicaciones
1996

Copyright: Universidad de Cádiz. Servicio de Publicaciones
Diseño y Maquetación: Creasur
Imprime: Imprenta Repeto - Cádiz
I.S.B.N. 84 - 7786 - 371 - 7
Depósito Legal: CA: 669/96

Índice General

Índice de Figuras

Índice de Tablas

Prólogo

En la primera lectura, este libro, me produjo una impresión inquietante, más todavía, cuando mis compañeros tuvieron la gentileza, con la que me siento muy honrado, de pedirme que les escribiese un pequeño prefacio. Digo impresión inquietante, por el hech o, sin duda, algo sorprendente de tener ocho autores, ésto me trajo a la memoria unas palabras de Descartes que me habían llamado la atención hace mucho tiempo; de su Discurso del Método, consulté, las palabras eran éstas; "... vemos que los edificios que un solo arquitecto ha comenzado y terminado suelen ser más bellos y mejor ordenados que aquellos que distintos arquitectos han tratado de recomponer". Comprenderá el amable lector, que con la autoridad de estas frases mi recelo fuese a mayores, inicié, p or tanto, una segunda lectura, con más calma, con más afilado y atento lápiz. Y quedé sorprendido, muy agradablemente sorprendido, la duda sembrada en mí por las palabras de Descartes se desvaneció, el libro tenía un tono holístico, contemplaba 'el todo' muy bien, de manera muy satisfactoria, lo encontré redondo, terminado y completo, sin saltos de autor a autor, discurriendo la materia de modo grácil página a página.

La ciencia computacional, es amplia, y cada vez más diversa y más cambiante, por ello, el reunir de forma coherente el conocimiento iniciático "para hoy"no es tarea sencilla, sobre todo teniendo plena consciencia de que "para mañana"ya habrán cambiado m uchas cosas, lo de hoy será obsoleto y una ola de nuevas ideas, de nuevos conceptos la invadirán mañana. Por eso, quizás, tiene gran mérito para mí, escribir un libro así, de fundamentos, de casi lo eterno en informática, esto me hace pensar que posiblem ente el poeta Virgilio tenía mucha razón cuando dijo: Audaces fortuna juvat, la fortuna favorece a los audaces. Os quiero animar para que prosigáis la andadura de escribir con audacia, seguro que vuestros alumnos serán receptivos y os lo agradecerán. Por mi parte un abrazo y deseos de mucho éxito.

IGNACIO PÉREZ BLANQUER
Prof. Tit. del Área de Lenguajes y Sistemas Informáticos

Introducción al libro

Fundamentos Informáticos es el sucesor de **Apuntes de Informática**, que surgió en el 95 fruto de la colaboración de una serie de profesores que impartíamos asignaturas de introducción a la informática en los Planes de Estudios de diversas Escuelas Técnicas y Facultades de la Universidad de Cádiz. Surgió ante la necesidad de crear unos apuntes para impartir la teoría de nuestras asignaturas, ya que, aunque la bibliografía en lo referente a temas de introducción a la Informática es muy amplia, no existía ningún libro que se adaptara plenamente a nuestras necesidades comunes.

En **Fundamentos Informáticos** pretendemos *tocar* todos los aspectos básicos de la informática (el ordenador, los periféricos, representación de la información dentro del ordenador, algoritmos, programas, etc.). En el libro se ofrece más información de la que se puede exponer en un curso académico, sin embargo, esto no es un error cometido por nosotros, sino que es intencionado. Al ofrecer toda esa información, el profesor que use este libro para impartir su asignatura, podrá seleccionar y *navegar* entre el mar de información que se le ofrece. Y el alumno podrá utilizar este libro tanto de libro de texto como de consulta.

A continuación vamos a indiciar brevemente el contenido de cada uno de los ocho temas que actualmente constituyen **Fundamentos Informáticos**:

- **Introducción: ¿Qué es la Informática?.**

  En este tema introducimos al lector en el mundo de la Informática. Como siempre que empezamos en algo nuevo, hemos de aprender nueva terminología y nuevas definiciones: informática, computador, hardware, software, personal informático, ... También es interesante conocer la evolución histórica de la informática y su aplicación en los distintos campos de la vida diaria.

  Finalmente, la automatización del tratamiento de la información, que es el fin que persigue la informática, plantea dos problemas: la confidencialidad y la seguridad de los datos. En la sección 1.5 de este tema se tratan ambos problemas.

- **El Ordenador.**

  Veremos individualmente cada una de las unidades o módulos funcionales en que se descompone un ordenador (A.L.U., U.C., Memoria Principal, ...). También conoceremos los dispositivos con los que el ordenador se comunica con el exterior: los periféricos. Posteriormente veremos una clasificación de los ordenadores, dentro de la cual encontraremos el tipo de ordenador más ampliamente difundido en la actualidad: el P.C. Y acabamos el tema con algunos consejos sobre ergonomía y ecología en el uso de los computadores.

- **Representando la Información.**

Una vez conocida la estructura funcional de un ordenador y los dispositivos con los que se comunica con el exterior, no está de más conocer cómo se representa internamente la información dentro del computador. En este tema se presta especial interés a la representación de la información numérica (binario, octal y hexadecimal), a cómo pasar un valor numérico de un un modo de representación a cualquier otro y las operaciones básicas que podemos hacer sobre dichos valores numéricos.

- **Sistemas Operativos.**

  Un sistema operativo es un programa o conjunto de programas de control cuyo objetivo es facilitar el uso del computador y que éste se use eficientemente. El sistema operativo es imprescindible para poder utilizar el ordenador y actúa de intermediario entre el ordenador y el resto de los programas. En este tema veremos: funciones que desempeña un sistema operativo, su evolución a lo largo de la historia de la informática, los módulos que lo componen, posibles arquitecturas, ... Finalmente veremos las características de algunos sistemas operativos concretos que están teniendo gran éxito en la actualidad.

- **Software de Aplicación.**

  Por aplicación entendemos un programa destinado a satisfacer una necesidad concreta: redacción de documentos, realización de cálculos más o menos complejos, dibujar en dos o tres dimensiones, ... En **Software de Aplicación** conoceremos las características de las aplicaciones más comunes: procesadores de textos, hojas de cálculo, ...

  Debemos resaltar el apartado 5.4 (*Software de Comunicaciones*), en el cual hacemos referencia a la famosa INTERNET.

- **Metodología de la Programación.**

  A la hora de desarrollar *programas*, hemos de seguir una serie de fases. Aquí conoceremos esas fases. También profundizaremos en un concepto clave a la hora de diseñar un programa: el *algoritmo*. Pretendemos enseñar al alumno los conceptos básicos de programación.

- **Lenguajes de Programación.**

  Un programa es una secuencia de pasos a seguir para resolver un problema. Esta secuencia ha de ser interpretada por el ordenador, por lo que debe estar escrita en un lenguaje que él entienda: el lenguaje máquina. Sin embargo, el lenguaje máquina es muy engorroso de usar debido a que está formado por secuencias de ceros y unos, por lo que surgen una serie de lenguajes más fáciles de utilizar, que se parecen más al lenguaje del hombre, y que pueden ser fácilmente traducidos a lenguaje máquina. Aquí surgen los programas traductores.

  Se presentan dos clasificaciones de los lenguajes de programación según dos criterios distintos: según la proximidad respecto al lenguaje máquina y según el comportamiento (paradigma) de los lenguajes.

  Por último, veremos que lenguajes son los más adecuados para desarrollar ciertas aplicaciones.

- **Redes de Ordenadores**

Estamos en una sociedad en la que la información es muy importante. En dicha sociedad, un ordenador aislado del mundo no es interesante. Por dicho motivo se tiende a conectar los ordenadores entre sí formando pequeñas redes de ordenadores locales (LAN) que, a su vez, se agruparán formando redes más extensas (WAN) que abarcan países, continentes e incluso el mundo entero. Aquí veremos ciertos aspectos de una red de ordenadores: su utilidad, arquitecturas, tipos de canales físicos que se usan para la comunicación, ... También haremos referencia a una red (o mejor dicho, a una red de redes) muy conocida actualmente: INTERNET.

Junto a los ocho temas que componen actualmente el libro, aparecen un glosario de conceptos (que servirá de diccionario informático al alumno) y dos índices, uno de figuras y otro de tablas, que permitirán al lector localizar fácilmente los gráficos y las tablas que aparecen en este libro.

Fundamentos Informáticos, como toda obra que desee preservar su interés, está abierta a sugerencias para modificaciones y mejoras, por lo que, desde aquí, animamos a los lectores a que nos hagan llegar todas las sugerencias que consideren oportunas para que, entre todos, hagamos un libro útil y que se adapte cada vez mejor a nuestras necesidades. Este fue el motivo que nos impulsó a escribir **Apuntes de Informática** y el que nos ha empujado a continuar con **Fundamentos Informáticos**.

LOS AUTORES

Capítulo 1

Introducción: ¿Qué es la Informática?

Desde sus orígenes, el hombre ha tenido necesidad de la información. Desde las antiguas civilizaciones se vio la necesidad de la información como la manera de acceder a noticias o experiencias no conocidas directamente. Así los viajeros, cuando regresaban de sus andanzas por tierras y mares desconocidos, traían datos interesantes para el comercio, las ciencias, las artes... Esta información, que en principio se recogía de forma oral, con el surgimiento de la escritura, comenzó a almacenarse en medios que evolucionaron desde las tablillas hasta el papel, pasando por los papiros y los pergaminos. También los medios de transmisión de la información han ido evolucionando desde la transmisión oral, buena para distancias cortas, hasta la transmisión a grandes distancias por cables utilizando código Morse o la propia voz mediante el uso del teléfono, pasando previamente por medios de comunicación tales como el envío de señales acústicas con instrumentos (por ejemplo el tam-tam) o visuales (por ejemplo las señales de humo o los destellos con espejos).

Como vemos, se puede decir que el tratamiento de la información es tan antiguo como el hombre y se ha ido potenciando y haciendo más sofisticado con el transcurso del tiempo hasta llegar a la era de la electrónica. El hombre no ha parado a lo largo de la historia de crear máquinas y métodos para procesar la información. Para facilitar esta tarea, en especial en el mundo actual, donde la cantidad de información que se procesa a diario es ingente, surge la **informática**. Con la aparición de los primeros ordenadores en la década de los años 50 y su gran difusión en los años 80 con la creación de los primeros PCs, se extiende una nueva manera de tratamiento de la información que desarrolla el tratamiento de la misma hasta límites antes insospechados.

1.1 Definición y origen de la Informática

El término **informática** es un neologismo creado en Francia en 1962 por Philippe Dreyfus. El término francés es **INFORMATIQUE** y está formado por la contracción de las palabras **INFOR**mation auto**MATIQUE**. Este término fue aceptado por los otros países. En España se tradujo por **INFORMÁTICA** (**INFOR**mación auto**MÁTICA**), aunque en los países anglosajones se utiliza el término de **Computer Science** (Ciencia de las Computadoras).

Existen muchas definiciones posibles de informática. La Academia Francesa de la Lengua la define en 1966 como *la ciencia del* **tratamiento racional**, *por medio de* **máquinas automáticas**, *de la* **información**, *considerada ésta como soporte de los conocimientos humanos y de las comunicaciones, en los campos técnico, económico y social.* La definición que nos da

la Real Academia Española de la Lengua nos dice que la informática es *el conjunto de conocimientos científicos y técnicas que hacen posible el tratamiento automático de la información por medio de* **computadoras** *electrónicas.*

De ambas definiciones podemos obtener una tercera, más completa, que nos define la informática como *la ciencia que estudia el* **tratamiento automático y racional** *de la* **información** *mediante el uso de* **computadores** *electrónicos.*

En la definición anterior hablamos de **tratamiento automático** porque son las propias máquinas las que realizan las tareas de captura, proceso y presentación de la información, y también hablamos de **tratamiento racional** porque todo el proceso está regulado a través de una secuencia de instrucciones (que aparecen registradas en lo que llamamos **programa**) y que *siguen el razonamiento humano.*

Para comprender con más claridad la definición anterior de informática, hemos de conocer el significado de otros dos términos claves: **información y computadora.**

1.1.1 Información

Hemos indicado que la informática nos permite el tratamiento automático de la información. Pero la pregunta entonces es *¿qué es la* **información***?.*

De una manera informal, podemos considerar la **información** como un conjunto de **datos** ordenados que nos aportan conocimiento sobre las cosas. Esta definición puede pecar de ser demasiado superficial para nuestros intereses, así que posteriormente daremos otra definición más formal, pero para ello debemos definir antes dos nuevos conceptos: **carácter y dato.**

Carácter es cualquier símbolo numérico, alfabético o especial que se emplea en la escritura y en el cálculo:

- *numéricos:* 0, 1, 2... , 9.
- *alfabéticos:* a, b, c... , z, A, B, C... , Z.
- *especiales:* *, ^, /, +, #, etc.
- *control:* Retorno de carro, Fin de Fichero (EOF)...

Dato es cualquier conjunto de caracteres (puede ser un único carácter). Existen tres tipos de datos:

- *numéricos*, que están formados exclusivamente por dígitos numéricos. Por ejemplo: 4, 21, 40.038[1].
- *alfabéticos*, que están formados exclusivamente por letras del alfabeto. Por ejemplo: x, ella, Juanillo.
- *alfanuméricos*, que están formados por la combinación de caracteres numéricos, alfabéticos y especiales. Por ejemplo: A4#10, Vx1, ejemplo1.

Ahora se podemos definir la **información** de una manera más formal como *el conjunto de datos (numéricos, alfabéticos o alfanuméricos)* **ordenados** *con los que se representan convencionalmente hechos, objetos o ideas.* En esta definición aparece una idea ausente en las

[1]Para la separación de la parte entera y decimal de un número se utiliza el carácter punto (.).

definiciones de **carácter y dato**: el orden. En la información es importante el orden de los datos, ya que un conjunto de datos empleados sin orden alguno nos aportaría una información diferente a la deseada o incluso podría no aportarnos ninguna información. Esto lo podemos ver en el ejemplo siguiente:

... esto es un libro básico de informática ...	Es correcto
... esto de informática es un libro básico ...	No es correcto porque no expresa la información deseada

1.1.1.1 Pasos en el tratamiento de la información

El tratamiento de la información no es más que operar o procesar un conjunto de datos iniciales o datos de entrada, y, como resultado de ese procesamiento, obtener un conjunto de datos finales o de salida. Como podemos ver, el **procesamiento de datos** está constituído por tres actividades básicas:

- **captura de datos de entrada:** Los datos deben ser registrados antes de poder procesarse. Este registro puede ser realizado sobre *documentos de papel*, y luego introducimos esos datos en el computador, o por un *dispositivo de entrada directa* (como puede ser un lector de códigos de barras) de modo que la máquina ya los puede tratar directamente.
- **manipulación de los datos:** Sobre los datos capturados podemos realizar las siguientes operaciones:
  - **agrupación:** consiste en organizar o clasificar elementos similares por grupos o clases.
  - **cálculo:** consiste en la manipulación aritmética de los datos.
  - **clasificación:** consiste en el ordenamiento de los datos agrupados según una secuencia lógica, como puede ser del más grande al más pequeño, del más antiguo al más reciente...
- **manejo de los resultados de salida:** Una vez han sido manipulados los datos de entrada, podemos realizar con ellos las siguientes operaciones:
  - **almacenamiento y recuperación:** Con el proceso de *almacenamiento* conservaremos los datos para consultarlos en el futuro. Para realizar la consulta utilizaremos el proceso de *recuperación*.
  - **comunicación y reproducción:** El proceso de *comunicación* de datos consiste en la transferencia de los mismos de un lugar a otro, donde serán utilizados o se procesarán de nuevo. Este proceso continúa hasta que la información llega hasta el usuario. Cuando la recibe, puede necesitar copiar o duplicar la información (por ejemplo, sacar por impresora un listado de las ventas del último mes) y esta tarea de *reproducción* la realiza una máquina.

1.1.1.2 Operaciones para el tratamiento de la Información

En el tratamiento de la información aparecen implicadas algunas o todas las operaciones elementales que exponemos a continuación:

- **lectura:** Consiste en adquirir la información que después utilizará el resto de las operaciones elementales.

- **almacenamiento:** Consiste en almacenar la información durante el tiempo que sea necesario para hacer uso de ella cuando se precise.

- **clasificación:** Permite ordenar la información guardada usando la operación anterior de *almacenamiento*. De este modo podemos acceder cuando sea necesario a parte o a la totalidad de dicha información.

- **cálculo aritmético y lógico:** Este tipo de operación elemental nos permite procesar la información realizando sobre ella operaciones aritméticas y lógicas.

- **copia:** Consiste en poder transcribir información a un soporte dado de forma automática.

- **escritura:** Consiste en mostrar la información de una manera clara y ordenada sobre un soporte dado. La diferencia con respecto a la *copia* radica en que, en el caso de tratarse de un tratamiento automático de la información, la información es mostrada de manera que sea *inteligible* para las personas, mientras que en la *copia* dicha información se encuentra escrita en un *código* inteligible por el computador.

Una vez vistas las operaciones elementales que intervienen en el tratamiento de la información, y recordando que la *informática* es el *tratamiento automático de la información*, nos podemos preguntar, *por qué se automatiza este tratamiento.* A esta pregunta intentaremos dar respuesta en la sección siguiente.

1.1.1.3 ¿Por qué se automatiza el tratamiento de la Información?

Las razones que han llevado a la automatización del tratamiento de la información son fundamentalmente cuatro:

- la realización de funciones que el hombre por sí sólo no puede llevar a cabo: comunicaciones a larga distancia, el radar, etc.

- la realización de funciones que, aunque el hombre puede llevarlas a cabo por sí mismo, su ejecución tardaría mucho tiempo, de modo que ya no se alcanzarí a el fin perseguido con la realización de dichas funciones. Por ejemplo, los cálculos complejos necesarios para el seguimiento y control de un proyectil dirigido o de una nave espacial.

- la obtención de seguridad en algunas tareas, como las que implican la repetición de una serie de pasos, en las que el hombre es más propenso a cometer errores. Sin embargo, las máquinas, una vez que se les ha *enseñado* como realizar las tareas correctamente, repiten el proceso una y otra vez sin cometer ningún error.

- la sustitución del hombre para tareas monótonas. Este tipo de tareas no implican el desarrollo de su actividad intelectual, con lo que, al automatizarlas, el hombre puede dedicar su esfuerzo a funciones más decisivas e importantes.

1.1.2 Computador

En la definición de informática indicábamos que el tratamiento automático de la información se realizaba con unas máquinas que llamamos *computadoras*. En este apartado pretendemos dar una visión global sobre qué es un computador y qué lo diferencia del resto de las otras máquinas que también procesan información.

Una **computadora**[2] es una *máquina útil y específica para el tratamiento de la información que acepta unos datos de entrada, efectúa con ellos operaciones aritméticas y lógicas*[3] *y muestra los resultados a través de un medio de salida. Todo este proceso no lo controla un operador humano, sino un programa que previamente ha sido almacenado en la propia computadora.*

La diferencia fundamental del computador respecto a otras máquinas que realizan un tratamiento automático de la información se encuentra en la propia definición: el proceso lo controla un programa y no un operador humano. Sin embargo, existen otras diferencias, ya que un computador posee también las siguientes características adicionales:

- Puede realizar las seis operaciones elementales de tratamiento de la información.
- Gran velocidad de tratamiento de la información.
- Gran potencia de cálculo aritmético y lógico.
- Puede *guardar* los programas y datos necesarios para la resolución de cualquier problema.
- Puede comunicarse con las personas y con otras máquinas para recibir y emitir datos.
- Puede tratar los datos en **tiempo real**, es decir, el tiempo de respuesta debe ser adecuado dependiendo del tipo de programa que está siendo *ejecutado*[4] en el computador en ese momento. Así por ejemplo, si se trata de un programa de un sistema de defensa, el tiempo de respuesta debe ser muy inferior al de un programa de gestión de un cajero automático.

1.2 Elementos básicos en la Informática

Los tres elementos básicos en los que se sustenta la informática son:

- el elemento físico o **hardware**.
- el elemento lógico o **software**.
- el elemento humano: **personal informático**.

[2] Sinónimos de computadora son computador y ordenador.

[3] Entendemos por operaciones lógicas aquellas que suponen comprobación, selección y copia de símbolos numéricos o no numéricos.

[4] Este verbo, en la jerga informática, es sinónimo de seguir paso a paso o interpretar las instrucciones de un programa.

1.2.1 Hardware

El **hardware** o **soporte físico** de un computador es la máquina en sí: los teclados, los monitores de vídeo, los dispositivos de almacenamiento de memoria, los tableros de circuítos... Por extensión también se considera hardware todo lo relacionado con la máquina, como son las disciplinas o materias relacionadas con el diseño, construcción y gestión del hardware.

El hardware se puede descomponer, de manera global, en dos elementos:

- **Computador Central**: Es el cerebro del computador y tiene como misión la coordinación y la realización de todas las operaciones que tengan lugar en el computador.
- **Periféricos:** Son los dispositivos a través de los cuales el computador se comunica con el mundo exterior. Dentro de los periféricos también incluímos los sistemas de almacenamiento secundario.

En temas posteriores se estudiarán con más detalle cada uno de los componentes del hardware.

1.2.2 Software

El **software** o **soporte lógico** de una computadora es el conjunto de programas *ejecutables* por la computadora, así como los datos manejados por los programas. Por extensión también se considera software a los documentos y materias relativos al funcionamiento, el diseño y la construcción de programas.

Dentro del software podemos hacer una clasificación:

- **Datos:** Son los datos que usan los programas para trabajar y operar con ellos. Por ejemplo: Los nombres de alumnos de la Universidad, sus notas, cursos, asignaturas...
- **Programas:** Instrucciones para operar con los Datos. Dentro de los programas podemos a su vez hacer otra división:
  - **Sistemas Operativos** (S.O.): Programas básicos para manejar el hardware del ordenador. Los veremos más ampliamente en otro tema.
  - **Lenguajes de programación:** Sirven para crear programas ejecutables de cualquier tipo. Básicamente se dividen en compiladores e intérpretes, que también estudiaremos en otro tema.
  - **Aplicaciones:** Programas que pueden servir para multitud de utilidades distintas. De hecho, podríamos considerar a los dos apartados anteriores como dentro de este mismo, ya que al fin y al cabo TODO son programas para distintas aplicaciones. Dentro de este apartado tenemos: programas de gestión de bases de datos, procesadores de textos, programas de dibujo artístico y lineal, hojas de cálculo, agendas electrónicas, programas de correo electrónico (e-mail)...

1.2.3 Personal Informático

El elemento humano es el más importante de los tres elementos de la Informática. Aunque los ordenadores reducen o reemplazan el trabajo del hombre en determinadas actividades, un ordenador sin un ser humano es un elemento inútil. El ser humano le es necesario por varios motivos:

- un ordenador necesita de un programa para hacer algo y estos programas deben de ser diseñados, codificados y mantenidos durante todo su período de vida útil por un ser humano.

- los distintos componentes del hardware de un ordenador evolucionan con el tiempo, haciéndose cada vez más potentes y eficaces. El diseño y construcción de estos componentes debe ser llevado a cabo por seres humanos.

- cada empresa necesita un equipo informático de acuerdo a sus necesidades. El estudio del equipo informático necesario, la instalación y el mantenimiento del mismo sólo puede ser llevado a cabo por el hombre.

Como vemos, el ser humano es imprescindible en la Informática, aunque muchos crean que el ordenador acabará sustituyendo al hombre en todas las actividades en las que actualmente es irremplazable.

Por **personal informático** entendemos el conjunto de personas que realizan funciones relacionadas con el uso de las computadoras en una empresa. Este personal informático lo podemos clasificar en:

- **personal de dirección**: El encargado de dirigir y coordinar el *Departamento de Informática* o el *Centro de Proceso de Datos* de la empresa para tener un buen rendimiento de los recursos (hardware, software y personal informático) disponibles.

- **personal de análisis:** El encargado de:

  - diseño y creación de algoritmos[5], que son el paso previo a la creación del programa.

  - estudiar los sistemas operativos y sus posibles modificaciones para una mayor eficacia.

  - dar apoyo técnico a los usuarios de las aplicaciones.

- **personal de programación:** El encargado de pasar el algoritmo realizado por el personal de análisis a un determinado lenguaje de programación y la prueba de dichos programas, para comprobar que su funcionamiento es correcto.

- **personal de explotación y operación:** El encargado de:

  - ejecutar los programas existentes y dar los resultados obtenidos
  - el mantenimiento del sistema informático.

No se considera como perteneciente al personal informático al **usuario** del ordenador, es decir, a la persona que utiliza el computador y una aplicación como una herramienta más para realizar o facilitar su trabajo.

[5]De una forma muy general, podemos decir que un algoritmo es una secuencia de pasos que se siguen para la resolver un problema.

1.3 Evolución Histórica

Los ordenadores son unas máquinas muy jóvenes, ya que nacen en los años cuarenta. Como casi todo, no surgieron de la noche a la mañana, sino que existieron otras máquinas que pueden ser consideradas como las precursoras de los mismos y que veremos en la sección siguiente. Posteriormente veremos como han evolucionado desde su aparición.

1.3.1 Hasta el Primer Ordenador

El primer aparato conocido que se utilizó para facilitar los cálculos fué el ábaco. Fundamentalmente se empleó en China (desde antes del año 470 a. de J.C.) y en las civilizaciones precolombinas, y aún se sigue usando en muchos países de Oriente.

Bastantes siglos después, Blaise Pascal construye la primera calculadora mecánica para sumar en 1642. Consistía en una serie de ruedas dentadas numeradas de cero a nueve. Cuando en una rueda se pasaba del nueve al cero (tras dar una vuelta completa) se producía un giro de una posición en la rueda situada a su izquierda. Se puede considerar como una máquina con programa interior único, es decir, que siempre realiza lo mismo (en este caso, sumar). Sin embargo, esta máquina no puede considerarse automática, porque necesita la intervención constante del que la maneja para realizar las maniobras que exige cada operación.

Treinta años más tarde, Leibniz idea una máquina que realiza las cuatro operaciones aritméticas. Ya no es una máquina con programa único (se puede elegir entre varias opciones: sumar, restar, multiplicar o dividir), sin embargo, no es tampoco automática.

Posteriormente, Charles Babbage (1792-1871) ideó un proyecto muy ambicioso, la máquina analítica, capaz de realizar cualquier operación matemática sin precisar la intervención humana durante el proceso de cálculo. Pero esta máquina no llegó a terminarse debido a que su complejidad mecánica era muy superior a las posibilidades existentes en su época. Sin embargo, su contribución fué decisiva para los ordenadores actuales. La máquina de Babbage utilizaba el sistema de tarjetas perforadas de Jacquard[6] para introducir en la máquina tanto el programa como los datos del problema. Así surgen los primeros ingenios con programa exterior.

Hasta ahora todos los ingenios vistos se han orientado a facilitar el cálculo, pero a lo largo del siglo XIX, el gran desarrollo industrial provoca un aumento de la complejidad en la organización de la sociedad y surge la necesidad de manejar grandes cantidades de información. Es entonces cuando Herman Hollerith (1860-1929), que era un funcionario de la Oficina de Censos de los Estados Unidos, ante la necesidad de mecanizar los datos del censo de 1890 y teniendo conocimiento de las tarjetas perforadas de Jacquard, tiene la idea de representar las respuestas a las preguntas como perforaciones o no en la tarjeta (si la respuesta es SI, se perfora, y si es NO, no se perfora). Surge así la codificación binaria de la información.

Tras los logros obtenidos a finales del siglo XIX y principios del XX, y gracias a las condiciones sociales existentes en el primer tercio del siglo XX, surge el primer ordenador.

1.3.2 Hasta la actualidad

En 1944, Howard Aiken (1900-1973), profesor de la Universidad de Harvard, diseñó y terminó el primer ordenador, el MARK I, basándose en las ideas de Babbage. Este ordenador se

[6] Jacquard fue un mecánico francés (1752-1834) que inventó un telar automático en el que se introducían programas mediante tarjetas perforadas.

construyó a base de elementos electromecánicos, con lo que la máquina era muy lenta debido, entre otras razones, a la inercia de sus partes móviles.

Entre 1943 y 1946 se construye el ENIAC, que fué la primera calculadora electrónica. Todas las operaciones, excepto las de entrada y salida, se realizaban mediante circuítos electrónicos a base de válvulas de vacío. Era más rápido que el MARK I, aunque para cada problema exigía una configuración de los circuítos. Como anécdota diremos que ENIAC disponía de 18.000 tubos de vacío y, cada vez que se ponía en funcionamiento, se producía un brusco descenso en las luces de la ciudad de Filadelfia.

En 1946, Von Neumann (1903-1957) enuncia los principios de funcionamiento de un ordenador en el que no hubiese que modificar los circuítos internos para cada programa. Surge así el concepto de **programa interno** o **programa almacenado**. La idea consiste en guardar en memoria tanto los datos de entrada, de salida e intermedios (algo que hacían ya las máquinas anteriores) como el propio programa (que en las máquinas anteriores se introducía mediante el cambio de la configuración de los circuítos). Esta idea de almacenar las instrucciones del programa junto con los datos supone un aumento de la flexibilidad y potencia en el uso de la computadora en dos sentidos:

- las instrucciones pueden cambiarse sin necesidad de cambiar las conexiones de los cables.
- como las instrucciones se almacenan como los datos, pueden procesarse como tales, haciendo posible la modificación automática de las mismas y la alteración de su secuencia.

En 1951 aparece en el mercado en primer ordenador construído según las ideas de Von Neuman, el UNIVAC I. Fue el primero de lo que se llamó ordenadores de la **primera generación**, que se caracterizaban por utilizar válvulas de vacío y poder ejecutar unas mil instrucciones por segundo. Su uso fundamental fué en el campo científico y en el militar.

En 1956 se descubre el transistor y en 1960 se incorpora a los ordenadores en lugar de las válvulas de vacío, comenzando así la **segunda generación** de ordenadores. Debido a que el transistor es más pequeño, barato y fácil de producir en grandes cantidades, se inicia la producción en serie de ordenadores más pequeños, potentes, económicos y veloces (unas 10^3 veces más rápidos que los ordenadores de la primera generación).

En 1965 se decide reunir en una sóla pastilla de silicio los transistores individuales mediante el uso de técnicas especiales. Surgen así los circuítos integrados y la **tercera generación** de ordenadores. Con los circuítos se consigue mayor velocidad de cálculo y mayor potencia.

En 1970 las técnicas de integración alcanzan tal desarrollo que dan lugar al primer **microprocesador**, que consiste en incluir todo lo principal de un ordenador en un sólo circuíto integrado. Algunos hablan de **cuarta generación**, ya que esto dió lugar a los microordenadores de bajo precio y consumo y gran velocidad de cálculo.

En 1981, países punteros en tecnología, como Japón y EE.UU., anunciaron una nueva generación de ordenadores (la **quinta generación**), que se caracterizaba, entre otras cosas por:

- el uso de componentes a muy alta escala de integración,
- tener Inteligencia Artificial,
- usar lenguaje natural para comunicarse con el usuario,
- multimedia (integración de sonido, imagen y voz),
- interconexión con todo tipo de dispositivos, etc.

1.4 Aplicaciones de la Informática

Son pocas las activadades humanas que no tienen nada que ver con la informática. Aunque no nos demos cuenta, la informática está presente en casi todos los lugares: supermecados, oficinas... e incluso en nuestro propio domicilio (por ejemplo, el videotexto o los robots de cocina).

Algunos campos de aplicación de la informática son los siguientes:

- **Investigación científica y humanística**

  Se usan las computadoras para resolución de cálculos matemáticos, recuentos numéricos, etc. Algunas de las operaciones realizadas son:

  - Resolución de ecuaciones y problemas matemáticos.
  - Análisis de datos de medidas experimentales, encuestas, etc.
  - Análisis automático de textos (búsqueda del número de apariciones de un vocablo, etc.).
  - Simulación. Se pretende simular el comportamiento de un sistema, estudiando los aspectos o características de éste que nos sean necesarios. Por ejemplo: para el diseño de naves espaciales, una vez diseñado el artefacto, se somete a una serie de pruebas mediante un simulador de vuelo que nos permite conocer las cualidades de la nave para volar, sus ventajas e inconvenientes. Lógicamente el coste de esta simulación es muy inferior al coste que supondría construir la nave y probarla, y permite que la investigación avance mejor y más rápidamente.
  - Cálculo o análisis numérico.

- **Aplicaciones técnicas**

  Usa la computadora para facilitar diseños de ingeniería y de productos comerciales, trazado de planos, etc. Algunas de las operaciones realizadas son:

  - Análisis y diseño de circuítos de computadora.
  - Cálculo de estructuras en obras de ingeniería.
  - Minería.
  - Cartografía.
  - Trazado de carreteras. CAD (Computer Aided Design): diseño asistido por computadora.
  - CAM (Computer Aided Manufacturing): fabricación asistida por computadora.

- **Documentación e información**

  Es uno de los campos más importantes para la utilización de las computadoras. Estas se usan para el almacenamiento de grandes cantidades de datos y la recuperación controlada de los mismos en *bases de datos*. Ejemplos de este campo de aplicación son:

  - Documentación científica y técnica. Existen bases de datos con referencias de publicaciones, artículos en revistas, etc., a las que se puede acceder según diversos criterios (materia, autor, año, etc.).

- Archivos automatizados de bibliotecas para la gestión de las mismas. Bases de datos con historias clínicas.
- Bases de datos jurídicas.
- Sistema de teletexto o videotexto. Están constituidos por bases de datos con información diversa a la que se puede acceder, en el primer caso, seleccionando los datos requeridos con un teclado conectado por la red telefónica y con una pantalla convencional de TV, y, en el segundo caso, pasivamente, leyendo la información que va apareciendo (sin control del usuario) en la pantalla de TV.

- **Gestión Administrativa**

  Automatiza las funciones de gestión administrativa típicas de una empresa. Existen programas que realizan las siguientes actividades:

  - Contabilidad.
  - Facturación.
  - Control de existencias.
  - Nóminas.
  - Control de producción y de productividad.
  - Investigación de mercado.
  - Control de proveedores y clientes.
  - Gestión de entidades bancarias.
  - Programas integrados de oficina electrónica u ofimática . Están constituídos por una serie de programas que intercambian información y facilitan la gestión administrativa. Entre estos programas encontramos: hojas de cálculo. correo electrónico, agenda electrónica, procesador de textos, gestión de archivos y/o bases de datos, aplicaciones gráficas...

- **Inteligencia artificial** Las computadoras se programan de forma que emulen el comportamiento de la inteligencia humana. Los programas responden como previsiblemente lo haría una persona inteligente. Podemos encontrar aplicaciones como:

  - Sistemas expertos: Se pretende que un programa actue, ante preguntas de un campo específico, como lo harían expertos en dicho campo.
  - Reconocimiento del lenguaje natural.
  - Programas de juegos complejos (ajedrez, damas, etc.).

- **Instrumentación y control**

  Aplicaciones:

  - Instrumentación electrónica.
  - Electromedicina.
  - Robots industriales.
  - Control de tráfico, polución industrial, etc.

- Control de vehículos (aviones, automóviles, barcos, helicópteros, etc.).

- **Otras aplicaciones**

  Algunos campos de aplicación de las computadoras no vistos anteriormente son:

  - El campo pedagógico
    - CAI (Computer Assisted Instruction).
    - CAL (Computer Aided Learning).
  - Juegos con computadora (video-juegos, etc.).
  - Aplicaciones en el arte:
    - composición de cuadros.
    - composición musical.
    - películas de dibujos animados, etc.
  - Procesamiento de imágenes

1.5 Informática y derecho

Nuestros datos, aunque nos pueda parecer extraño, pueden estar almacenados en muchos sistemas de ordenadores que pueden ir desde los ordenadores del sistema tributario hasta los de la empresa editorial que nos vendió la última enciclopedia, pasando por los ordenadores de los bancos, hospitales, agencia de seguros... Todo esto crea el potencial peligro de que los datos pasen a manos de personas o entidades que no son de nuestro interés o, aún peor, que no son deseables.

Aunque los métodos manuales de almacenamiento de información hace mucho tiempo que han estado en funcionamiento, el peligro surge con el uso masivo de los ordenadores, ya que estos permiten la transferencia de millones de datos en cuestión de escasos segundos, lo que puede dar lugar a la entrega no autorizada de datos a gran velocidad. Este problema tiene dos aspectos, que trataremos brevemente en la secciones posteriores:

- la entrega deliberada de datos a terceros
- el robo o destrucción de estos datos por personas ajenas a la organización que los almacena legítimamente.

El primer aspecto del problema se intenta solucionar mediante la implantación de una serie de legislaciones que regulan la **confidencialidad de los datos**. La única solución al segundo aspecto del problema consiste en aumentar la **seguridad de los datos**.

1.5.1 Confidencialidad de los datos

Desde siempre han existido grandes ficheros referentes a las personas (Registro Civil, Historiales médicos, ficheros bancarios, fiscales, etc.), sin embargo, con la aparición de los ordenadores, junto con sus soportes de almacenamiento, han permitido aumentar en proporciones antes impensables la cantidad de información sobre cada persona. También, con la llegada de estos *engendros electrónicos* se alcanzaron unas posibilidades de intercomunicación y difusión de

ficheros antes no conocidas, ya que podemos intercambiar millones de datos en cuestión de segundos.

Para conseguir la confidencialidad de los datos almacenados en los sistemas informáticos, los países occidentales utilizan una serie de legislaciones con unas normas que regulan el almacenamiento y la entrega de información a terceros. Estas legislaciones pueden obligar a los usuarios de sistemas informáticos destinados al tratamiento de información personal a declarar el uso que se le va a dar a la información y los individuos a los que se refieren los datos tienen el derecho de examinarlos.

Estos bancos de datos empezaron a desarrollarse en los años sesenta y se convirtieron rápidamente en un gran potencial industrial.

El primer país que tuvo una ley dedicada a la protección de datos fue Suecia (11 de mayo de 1973). Esta ley promulga que sólo determinados organismos de la Administración pueden recoger datos relativos al tratamiento de alcohólicos, internamientos psiquiátricos, informaciones relativas a la enfermedad o al estado de salud de una persona. Para ello, estos organismos deben ser autorizados.

Tras Suecia, el Congreso de los Estados Unidos aprobó la **Ley de la Intimidad** (Privaty Act) en 1974 por la que ponía límites al Gobierno, protegiendo la libertad de las personas. Esta ley autoriza a las personas a inspeccionar la información contenida en los archivos de cualquier organismo y que les afecte, así como a recusar, corregir o rectificar dicha información. Sin embargo, existen ciertos archivos que son una excepción a la ley: los de la CIA (Central Intelligence Agency), los del Servicio Secreto y ciertos registros gubernamentales delicados.

El 27 de Enero de 1977 se promulga la **Ley Federal sobre Protección de Datos** de la República Federal Alemana y el 14 de Julio de ese mismo año se promulga la **Ley de Derechos Humanos del Canadá**, en la que su parte IV se titula **Protección de la Información Personal.**

En Francia se aprueba el 6 de enero de 1.978 la **Ley sobre "Informática, Ficheros y Libertades"**. Unos años antes (1974) se había creado la **Comisión "Informática y Libertades"** cuya misión era la de proponer al Gobierno una serie de medidas para garantizar que el desarrollo de la Informática en el sector público o privado se realizara con el respeto a la vida privada, a las libertades individuales y a las libertades públicas.

En 1978 surgen Leyes de Protección de Datos en Dinamarca, Noruega y Austria, y en 1979 en Luxemburgo.

En septiembre de 1980 se firmó en Estrasburgo una *convención para la protección de las personas frente al tratamiento automático de los datos de carácter personal.* La importancia de este hecho radica en que fue el primer texto internacional adoptado en el terreno de la protencción de la intimidad de los datos personales. Aunque esta convención se hizo bajo el patrocinio del Consejo de Europa y está firmada por 30 países, su adhesión está abierta a países no europeos. Este texto estipula que todo ciudadano tiene el derecho de inspeccionar las informaciones de los ficheros informatizados que le conciernen. Si detecta errores, debe poder rectificar las informaciones. Además, se prohibe todo tratamiento de datos en que aparezcan el origen racial, las opiniones políticas, las convicciones religiosas o la vida sexual. Se imponen restricciones a la circulación internacional de datos, ya que técnicamente hablando, no hay nada que impida conservar en el extranjero ficheros relativos a la población de un país, pudiéndose crear en estos países extranjeros verdaderos *paraísos informáticos* si están desprovistos de toda legislación que limite los usos de la informática.

En España y Portugal, la Constitución protege al ciudadano contra los daños causados por el ordenador. La Constitución de la República Portuguesa (1976) indica en su artículo

35 que todos los ciudadanos tienen el derecho a conocer lo que se cuenta acerca de ellos en registros mecanográficos, así como el fin a que se destinen las informaciones, pudiendo exigir la rectificación de los datos y su actualización. Así como también indica que la informática no podrá ser usada para el tratamiento de datos referentes a convicciones políticas, religiosas o vida privada, excepto cuando se trata de datos no identificables con fines estadísticos. La Constitución Española, en su artículo 105 b) indica con carácter programático [7] que la Ley regulará el libre acceso de los ciudadanos a los Archivos y Registros, siempre que no afecte a la seguridad y defensa del Estado, la averiguación de los delitos y la intimidad de las personas. Más concretamente relacionado con la informática, el apartado 4 del artículo 18 indica que se limitará el uso de la informática para garantizar el honor y la intimidad personal y familiar, sin embargo, el análisis de este artículo lleva a diversas interpretaciones. El 5 de mayo de 1982 se aprueba la Ley de Protección Civil del derecho al honor, a la intimidad personal y familiar y a la propia imagen. Aunque esta ley no se refiere a la protección de los derechos de intimidad frente a la informática, es la que reguralará dicha protección mientras no se promulgue la normativa prevista en el artículo 18 4°.

1.5.2 Seguridad de los datos

Al hablar de la seguridad de los datos nos enfrentamos fundamentalmente a dos problemas:

- los **hackers** o **piratas informáticos**: Con el surgimiento y desarrollo de las grandes redes de ordenadores aparece la figura del *pirata informático*. El *hacker* es un experto en informática doméstica que se introduce en *sistemas informáticos*[8] que presuntamente son seguros y contienen material confidencial. Este acceso ilegal puede tener muy diversos fines que pueden ir desde el deseo de experimentar y divertirse hasta el afán de lucro (venta de la información obtenida).

  Alrededor del pirata informático ha surgido una subcultura con clubs, revistas, libros técnicos... e incluso literatura.

  El método empleado por un hacker para penetrar en una red de ordenadores suele ser el siguiente:

  1. localizar algún fallo del sistema de seguridad. Este fallo se suele detectar tras varios experimentos e intentos de entrada al sistema.
  2. montar una estrategia de ataque aprovechándose de los fallos descubiertos.
  3. *desmenuzar* las llaves o claves de acceso para poder moverse de un ordenador a otro.

  Como esta tarea es bastante monótona, los piratas informáticos suelen utilizar ciertos programas especiales que le ayudan en su tarea y reciben el nombre de **gusanos**.

  Aunque la actividad de un pirata informático es ilegal, es muy raro que destruya la información de los sistemas informáticos que invade, ya que su objetivo es penetrar todo lo que le sea posible en un sistema y cualquier actividad destructiva podría facilitar su detección.

[7] es decir, con intención de desarrollarse posteriormente con una ley.

[8] Por sistema informático entendemos al computador junto con todos los elementos necesarios para crear y utilizar aplicaciones informáticas.

- los **virus informáticos**: Un *virus* informático es un *programa de ordenador* con capacidad de autorreproducción y que efectúa acciones nocivas contra el sistema informático. Estas acciones nocivas pueden ser de muy diverso tipo:

  - borrado de todos los ficheros que tratemos de ejecutar,
  - aparición de objetos molestos en pantalla,
  - caída de los caracteres hacia la zona inferior de la pantalla,
  - aparición de mensajes de protesta,
  - formateado de disquetes o del disco duro, ...

  Existen también una serie de programas nocivos, que no son virus, aunque se suelen confundir con ellos por tener alguna de sus características:

  - **conejos:** El conejo fué la primera criatura que surgió para tormento de los usuarios del ordenador. Nació en un ordenador multiusuario de una universidad americana. En estos ordenadores existen múltiples usuarios conectados por terminales a un computador central y cada uno de los usuarios tiene una prioridad. El ordenador ejecutaba los programas de los distintos usuarios de acuerdo a su prioridad y al tiempo que llevaban esperando. Fue entonces cuando surgió el conejo, que era un programa que se situaba pacientemente en la cola de espera y, cuando llegaba su turno de ejecutarse, creaba un par de copias de sí mismo y las ponía a la cola. Este programa no destruía los ficheros, pero, al poco tiempo, la memoria del ordenador se saturaba con miles de copias del programa conejo y se bloqueaba el sistema. Entonces, al encargado no le quedaba más remedio que borrar todas las copias y restaurar el funcionamiento del ordenador.
  - **gusanos:** Un gusano es un programa que se desplaza por la memoria de los ordenadores. Posee las siguientes características:
    * tiene identidad propia, es decir, no se camufla en el código de otros programas ni los utiliza como medio para su propagación.
    * busca zonas de memoria desocupadas en donde realiza copias sucesivas de sí mismo hasta desbordar la memoria.
    * los hijos mantienen comunicación con los padres.
    * proliferan en redes públicas y de área local, donde utilizan el correo electrónico para propagarse.
  - **caballos de troya:** Son programas que ocultan, tras la apariencia de un programa interesante y útil, una sección de código nocivo. Las características a destacar de este tipo de programas son:
    * no se autorreproducen, con lo que el código maligno que portan sólo se podrá ejecutar una vez, sin embargo se propagan de un ordenador a otro gracias a la copia de disquetes.
    * se utilizan fundamentalmente para destruir sistemas informáticos de empresas.
    * suelen ser introducido en disquetes disfrazado de publicidad o de información de utilidad.

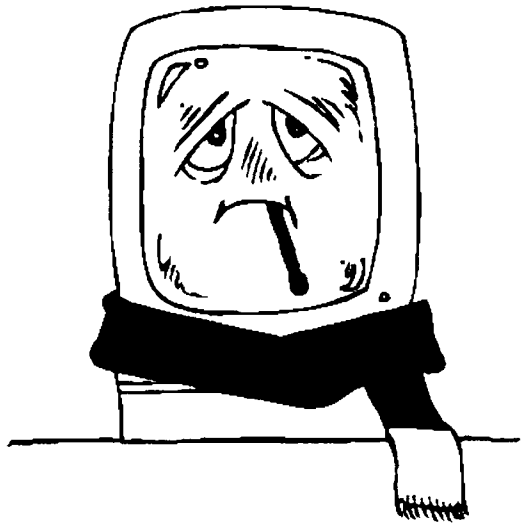


Figura 1.1: Un ordenador infectado.

- **bombas lógicas:** también llamada **bomba de tiempo** es un segmento de código nocivo que permanece en estado de latencia hasta que se produce algún suceso predeterminado, dando lugar a la activación de la bomba. Estos sucesos pueden ser de muy diversa índole: llegada de una fecha determinada, pulsar una secuencia de teclas, etc.

Los **virus informáticos** son programas, o mejor dicho, segmentos de código que poseen las siguientes características, algunas de las cuales heredan de las *criaturas* anteriormente mencionadas:

* son capaces de generar copias de sí mismos, de forma homogénea o en partes discretas, en un fichero o en disco. Esta es una acción que también realiza el *gusano*, sin embargo, este último realiza la función de autocopia sobre la memoria principal y no sobre el soporte magnético.
* modifican los programas ejecutables, a los que se adosan consiguiendo una ejecución parasitaria. Así, cuando el usuario ejecuta inocentemente un programa infectado, el código del virus se ejecuta también. En esto se diferencia del gusano (y se asemeja al *caballo de Troya*), ya que el gusano es un programa independiente y no necesita que se ejecute ningún programa infectado para comenzar su ejecución.
* para que un virus realice sus acciones nocivas se tiene que ejecutar y permanecer en memoria para obtener el control permanente de la C.P.U.
* las fases en la vida de un virus son tres:
  - **Infección:** En esta etapa el virus llega al ordenador a través de un programa contaminado, generalmente contenido dentro de algún disquete de dudosa procedencia. El usuario ejecuta el programa infectado y el virus se ejecuta, tomando el control del ordenador. Sin embargo, no afectan al funcionamiento normal de los programas, por lo que el usuario no sospecha de la existencia del virus.

- **Latencia:** El virus ya se ha instalado dentro del ordenador y tiene en todo momento control sobre lo que ocurre en el sistema. Entonces comienza a infectar todos los programas ejecutables que se ponen a su alcance. Si copiamos alguno de estos programas infectados en otro ordenador, el virus contaminará los ficheros no infectados aún de ese nuevo ordenador. Esta es la etapa más característica de un virus, ya que si no fuera por esta etapa, el efecto destructivo del virus delataría demasiado pronto su existencia, con lo que apenas se propagaría.
- **Activación:** Hasta el momento, el virus se ha limitado a autorreproducirse e infectar ficheros con la intención de expandirse, pasando desapercibido a los ojos de los inocentes usuarios. Sin embargo, una vez que se dan determinadas circunstancias (la llegada de una determinada fecha, se ha alcanzado un número de autocopias determinado, etc.), el virus se activa y comienza su actividad destructora. En esta etapa el virus se asemeja a una bomba lógica.

1.5.2.1 Características del terrorismo informático.

Podemos considerar un **terrorista informático** a aquel individuo que se introduce en sistemas informáticos de terceros con la intención de destruir (o chantajear con destruir) la información que contienen.

Este tipo de terrorismo tiene una serie de características:

- es necesario tener una serie de conocimientos y manejo de elementos técnicos que no todo el mundo domina.
- tienen la ventaja del tiempo y el espacio, es decir, son sólo necesarios milésimas de segundo para realizar la tarea destructiva y no es necesario la presencia física de los autores, sino que esta tarea de destrucción la pueden efectuar desde un ordenador remoto.
- pueden provocar serias pérdidas personales y materiales.
- en estos delitos es difícil de probar la autoría material.
- es sumamente sofisticado.

1.5.2.2 Mecanismos de control para preservar la seguridad

Distinguimos dos tipos de mecanismos que nos pueden ayudar a la hora de preservar la seguridad de un sistema informático:

- **mecanismos extrajurídicos:** consiste en utilizar todo tipo de mecanismo técnico, administrativo, psicológico... con el fin de impedir o mitigar los efectos de cualquier atentado terrorista. Entre estos mecanismos podrían estar:
  - realizar examenes psicológicos a personas que quieran ocupar puestos estrátegicos en el área de informática con el fin de conocer si se trata de potenciales terroristas.
  - encriptar (codificar) los datos mediante un código.
  - usar contraseñas (*passwords*) para poder acceder al sistema...

- **mecanismos jurídicos:** consiste en la incorporación de nuevos tipos de delitos en la legislación penal a nivel nacional e internacional, introduciendo la figura de los **delitos informáticos** y, dentro de ellos, el **terrorismo por computadora**.

Capítulo 2

El ordenador.

"Dios no juega a los dados." – Albert Einstein –

De similar manera, un ordenador no presenta un comportamiento aleatorio, aunque a veces lo parezca.

En este tema veremos los fundamentos básicos de funcionamiento de estas máquinas, que cada vez nos invaden más y más... y han venido para quedarse.

Hemos pretendido centrarnos en las generalidades de funcionamiento, sin olvidar los últimos avances tecnológicos, por eso a lo largo del tema iremos explicando brevemente estas novedades.

Por desgracia para esta obra, y por suerte para la humanidad, la técnica avanza a pasos agigantados, y cada vez más, en progresión exponencial, por lo que es posible que cuando leas estas líneas ya haya muchos datos anticuados, pero el funcionamiento básico del ordenador no cambiará tan deprisa, y tener una visión histórica del pasado nos ayuda a comprender mejor el futuro.

Además, si un libro de informática no se queda obsoleto en poco tiempo es porque no es demasiado bueno.

Hay que tener en cuenta, además, que un ordenador se queda anticuado en 2 ó 3 años. Por esto, a la hora de adquirir una de estas máquinas, es recomendable adquirirla con una configuración superior a la que necesitemos, pues los requisitos (de memoria, disco duro...) de las aplicaciones (programas) van aumentando de una versión a otra.

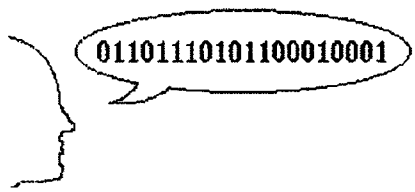


Figura 2.1: Los ordenadores sólo entienden código binario.

Código Binario	Símbolo a codificar
00	0
01	1
10	2
11	3

Tabla 2.1: Ejemplo de un código de 2 bits.

2.1 Introducción a la Codificación.

La codificación la podemos definir como una transformación de los elementos de un conjunto en elementos de otro conjunto siguiendo un método determinado, de tal forma que posteriormente se pueda efectuar el proceso inverso de decodificación.

En el interior de las computadoras la información se almacena y se transfiere de un sitio a otro según un código que utiliza solo dos valores (código binario) representados por 0 y por el 1. Cada uno de esos valores recibe el nombre de **BIT** (del inglés BInary uniT, o BInary digiT). En la Entrada y Salida de la computadora se efectúan automáticamente los cambios de código para que la información sea comprendida por los usuarios humanos.

Por ejemplo, si queremos sumar dos números (el 2 y el 7) lo primero que hacemos es darle al ordenador esos números. Primeramente, el ordenador codifica esos valores, los pasa a código binario (por ejemplo como 0010 el 2 y como 0111 el 7) y luego efectúa la operación deseada en el nuevo código, dando un resultado también en ese código binario (por ejemplo el 1001). Finalmente efectúa el proceso de decodificación para que entendamos el resultado final (esto es, decodifica 1001 en el valor 9 comprensible por nosotros los humanos).

Así, la información (datos e instrucciones) se transfiere a través de impulsos que la codifican, o sea, que los ceros (0) y los unos (1) no son más que impulsos eléctricos con una determinada tensión o voltaje (por ejemplo 3.3 Voltios para el 1 y 0 Voltios para el 0). Y son esos cambios de tensión los que circulan por los circuitos y chips del ordenador.

Obviamente, para codificar un alfabeto con más de 2 símbolos necesitamos más de un bit (0 ó 1). Por ejemplo, para codificar un alfabeto con 256 símbolos (por ejemplo las letras mayúsculas y minúsculas, números y otros símbolos) necesitamos un total de 8 bits por carácter, ya que $2^8 = 256$. Si necesitásemos codificar 257 símbolos necesitaríamos por fuerza 9 bits como mínimo, aunque con 9 bits ya podemos codificar un máximo de $2^9 = 512$ caracteres.

Supongamos que queremos codificar 4 símbolos, entonces necesitamos como mínimo 2 bits, ya que $2^2 = 4$. De esta forma, cada posible combinación de 2 bits (ceros o unos) corresponderá a un carácter o símbolo. En la tabla 2.1 vemos un ejemplo de esto.

Como hemos visto, el **BIT** es la unidad más pequeña de información que se puede almacenar en un ordenador. Es tan pequeña que normalmente se usa una unidad mayor: el **BYTE**. Un byte es el conjunto de bits necesarios para almacenar un carácter, que como hemos visto depende del número de caracteres que vayamos a representar. Normalmente, un byte suele ser igual a 8 bits. De hecho, los franceses no usan la palabra inglesa byte, sino que usan la palabra *octete*, refiriéndose a su habitual tamaño de 8 bits.

Los bits y los bytes, se pueden usar como una unidad de medida de la cantidad de información o como unidad de medida de capacidad de almacenamiento. Por eso, podemos decir que un determinado fichero informático tiene 1024 bytes (cantidad de información que

Unidad			N° Bytes		Otras equivalencias		
1 KByte	(KB)	=	2^{10} Bytes	=	1.024 Bytes		
1 MegaByte	(MB)	=	2^{20} Bytes	=	1.048.576 Bytes	=	2^{10} KB
1 GigaByte	(GB)	=	2^{30} Bytes	=	1.073.741.824 Bytes	=	2^{10} MB = 2^{20} KB
1 TeraByte	(TB)	=	2^{40} Bytes	=	2^{10} GB = 2^{20} MB	=	2^{30} KB

Tabla 2.2: Múltiplos del Byte.

contiene) o que un determinado ordenador puede almacenar hasta 1.048.576 bytes (cantidad de información que puede almacenar o capacidad de almacenamiento que tiene).

El byte sigue siendo una unidad bastante pequeña para medir algunas cosas (capacidad de la memoria de un ordenador o capacidad de almacenamiento de un disco, por ejemplo), por eso se utilizan unos múltiplos de esta unidad, que podemos ver en la tabla 2.2.

Existen multitud de códigos, para representar la información en binario, como el MORSE por ejemplo (que en vez de 0 y 1 usa puntos y rayas). Dentro de los códigos binarios informáticos podemos citar el EBCDIC, FIELDATA, AIKEN, GRAY, BCD y el más famoso de todos, el código **ASCII** (*American Standard Code for Information Interchange*), que codifica los caracteres más usuales, como son los signos alfabéticos (mayúsculas y minúsculas), numéricos, de puntuación y especiales (%, &, $, #, \, [,], {, } ...) que se usan en la escritura y el cálculo.

En un principio el código ASCII era de 7 bits, pero posteriormente, se le añadió un bit más. En total contiene, por tanto, 256 símbolos.

Ejemplo: Supongamos que codificamos la letra "P", siguiendo el famoso código ASCII, el cual le asigna el número 80 en decimal y 50 en hexadecimal, que corresponde a la siguiente sucesión de ceros y unos del código binario: 0101 0000. Para transmitir esa letra habría que transmitir los siguientes impulsos:

0 1 0 1 0 0 0 0

Por último, comentar que un procesador, trabaja siempre con un determinado número de bits, a la vez. Si decimos que un procesador es de 32 bits (como el procesador 486 de Intel) nos referimos a que sus operaciones las hace de 32 en 32 bits. Entonces se dice que tiene una longitud de **PALABRA** de 32 bits, concepto que veremos más detenidamente en otro apartado posterior.

Es obvio que cuanto mayor sea la longitud de palabra de un procesador más rápido procesará la información, ya que puede trabajar con más información (más bits) a la vez.

2.2 Esquema funcional de un ordenador.

De una manera general, una computadora u ordenador, no es más que una máquina capaz de aceptar unos datos de **ENTRADA**, efectuar con ellos operaciones lógicas y/o aritméticas (*procesarlos*) y proporcionar la información resultante a través de un medio de **SALIDA**, todo ello bajo el control de un programa o conjunto de instrucciones ordenadas, previamente almacenado en la memoria del ordenador.

Una simple calculadora se puede parecer a lo anteriormente descrito, pero la calculadora sólo hace operaciones aritméticas, y en todo caso, no sigue un programa enlazando automáticamente las operaciones una tras otra, sino que actúa exclusivamente de forma interactiva con un operador humano.

El esquema funcional de un ordenador está representado en la 2.2, y representa en módulos distintos las funciones principales de un ordenador o computadora (según el esquema de **Von Neumann**).

Hay que señalar, que aunque distinguimos entre datos e instrucciones, en realidad, ambos se pueden considerar como datos.

Se podría haber incluido en el esquema funcional la **Memoria Masiva**, auxiliar o externa (discos duros, disquettes, cinta...), pero no lo hemos hecho por considerarla como periféricos de Entrada y Salida.

Periféricos de Entrada/Salida.

Se conoce como periféricos aquellos dispositivos que usa el ordenador para comunicarse con el exterior. Nos detendremos más adelante en este tema, pero veamos ahora unas generalidades básicas.

Por Periféricos de ENTRADA se conocen aquellos dispositivos usados para introducir información (datos e instrucciones) en el ordenador. Ejemplos típicos de periféricos de entrada son el teclado (para introducir caracteres), el ratón (para introducir posiciones en la pantalla), un scanner (para introducir imágenes), un micro (para introducir sonidos)... y muchos más.

Por Periféricos de SALIDA se conocen aquellos dispositivos usados para extraer información (principalmente los resultados de los programas ejecutados) del ordenador. Ejemplos típicos de periféricos de salida son la pantalla o monitor (para visualizar caracteres o gráficos), una impresora (para imprimir resultados), un plotter (para dibujar imágenes y dibujos de gran calidad), un altavoz (para reproducir sonidos)... y muchos más.

Existen periféricos que no son exclusivamente de entrada o de salida, sino que son de Entrada y Salida (E/S) . Los más típicos son los dispositivos de almacenamiento o soportes de información (Memoria masiva, auxiliar o externa), como discos duros, disquetes, cintas... en los que se puede escribir información mediante un código preestablecido (código binario) y se puede leer la información que previamente se ha escrito. Son usados para almacenar datos y programas para su uso posterior. Son como armarios donde podemos meter, almacenar y sacar información.

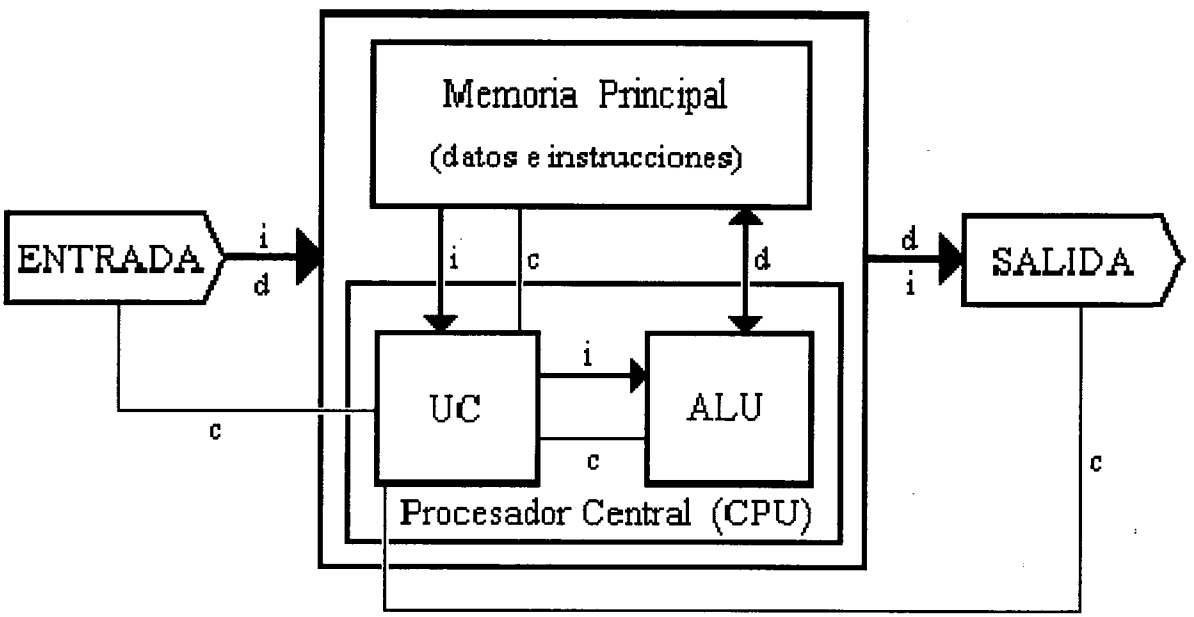
Ya indicamos en el tema 1 que el hardware de un computador se podía dividir en **ordenador central** (es el ordenador en sí) y los **periféricos** que comunican el ordenador con el exterior o almacenan información de forma indefinida.

2.3 El ordenador central.

En esta sección estudiaremos las distintas unidades que podemos encontrar en el ordenador central. Por ordenador central entendemos el ordenador propiamente dicho, sin periféricos de Entrada/Salida.

2.3.1 La Memoria Principal.

Para que el ordenador pueda ejecutar un programa es necesario tener cargado en memoria principal tanto el programa a ejecutar como los datos que necesite para su ejecución. La



i	Circulación de instrucciones. (Bus de instrucciones)
d	Circulación de datos. (Bus de datos)
c	Circulación de órdenes de control. (Bus de control)
UC	Unidad de Control (Control Unit). Controla todos los componentes: memoria, ALU,... Obsérvese que el bus de control llega a todos.
ALU	Unidad Aritmético-Lógica (Arithmetic Logic Unit). Realiza las operaciones aritméticas y lógicas.
ENTRADA	Periféricos de entrada: teclado, ratón, ...
SALIDA	Periféricos de salida: pantalla, impresora, ...
Memoria Principal	Memoria interna del ordenador: RAM y ROM.

Figura 2.2: Esquema funcional de un ordenador.

Figura 2.3: Ordenador central con el teclado y la pantalla como periféricos.

UC, como veremos, será la encargada de leer de la memoria principal, una a una, todas las instrucciones del programa a ejecutar.

La Memoria Principal está formada por circuitos electrónicos integrados, chips, que son capaces de almacenar valores binarios (cero o uno) en cada elemento o celda de memoria. Cada celda de memoria puede entonces estar en dos estados, a cero (0) o a uno (1).

Estas celdas se reúnen en las llamadas **palabras de memoria** que son el menor conjunto de celdas de memoria que se pueden leer o escribir simultáneamente, es decir, una **palabra** es el conjunto de bits que se leen o escriben en memoria de una vez. Normalmente, la longitud de palabra coincide con un número exacto de bytes.

Cada palabra de memoria tiene su **dirección de memoria**, que es un número que la identifica de forma única, de forma que cuando queremos leer o escribir en la memoria principal debemos decirle en qué dirección o posición queremos hacerlo. Por eso, se suele decir que la memoria principal es una memoria de acceso directo, ya que accedemos de forma directa al dato que necesitemos sin más que dar su dirección. Por tanto, el tiempo de acceso a cualquier palabra de la memoria es independiente de la dirección o posición a la que se accede.

En la figura 2.4 vemos una representación de la memoria en la que a la izquierda de cada palabra está representada su dirección. En el ejemplo, las palabras son de 1 byte (8 bits) y como tenemos 1024 palabras, tenemos 1024 bytes, o sea 1 KByte.

Veamos un ejemplo real: ¿Cuantas palabras de memoria tendrá un ordenador equipado con un procesador Pentium y con 8 MB de memoria?.

Para resolverlo es necesario conocer que ese procesador tiene una longitud de palabra de 64 bits (8 bytes):

$$TotalPalabras = \frac{TotalMemoria}{LongPalabra} = \frac{8MB}{64bits/plbra.} =$$

$$\frac{8 * 2^{20}Bytes}{8Bytes/plbra.} = 2^{20}Palabras = 1MPalabra.$$

Direcciones	Palabras de Memoria
0	0 1 0 1 1 1 1 0
1	1 1 0 1 0 0 1 1
2	1 0 0 0 1 1 1 0
3	0 0 1 1 1 0 0 0
4	1 1 1 0 1 0 1 1
5	0 1 1 1 1 1 0 0
.	.
.	.
.	.
1023	0 1 1 0 0 0 0 1

Figura 2.4: Memoria de 1KByte.

2.3.1.1 Tipos de Memoria Principal: RAM, caché y ROM.

La memoria principal se divide en dos tipos básicos:

- **R.A.M.** (*Random Access Memory*).
- **R.O.M.** (*Read Only Memory*).

Memoria RAM.

La memoria **RAM** (*Random Access Memory*, Memoria de acceso aleatorio), se llama de acceso aleatorio porque el usuario y sus programas pueden acceder a ella a cualquier posición (acceso directo) para leer o escribir (es una memoria de lectura y escritura). La ROM solo es de lectura.

Ese acceso aleatorio es también propio de la ROM, por lo que el nombre de RAM no es muy adecuado (quizás fuera mejor llamarle memoria de lectura y escritura). El tiempo de acceso a un dato no depende de su posición. Ese tiempo se mide en nanosegundos (nsg.) y en memorias RAM (dinámicas) suele rondar los 60 nsg. (pero cada vez serán más rápidas).

La RAM es un tipo de memoria **volátil** (la ROM no), que se borra al desconectarle la alimentación, o sea, que al apagar el ordenador se pierde todo lo que hubiera en ella. Debido a esto, antes de apagar el ordenador, es bueno salirse del programa en ejecución y guardar los datos que necesitemos.

Su función principal es la de contener información para que la use el procesador. La memoria por sí es hardware, ya que son circuitos físicos, pero lo que contiene es software. El software que contiene puede ser de dos tipos:

1. El código binario del **programa o programas** en ejecución. Además del programa que el usuario desee ejecutar hay muchos otros programas especiales, que son suministrados por el fabricante, como es el Sistema Operativo (S.O.) y diversos programas residentes (TSR) controladores de dispositivos (como el ratón, vídeo...).

2. Los **datos** que el programa necesita, obviamente codificados en binario.

Lógicamente, cuando escribimos un dato en una posición de la RAM, se pierde el dato que hubiera antes en esa misma posición, es decir, el contenido de la RAM puede variar, por lo que se las ha llamado *memoria viva*. Eso mismo no pasa al leer, pues un dato almacenado podemos leerlo cuantas veces queramos sin que este sea borrado.

Para que se mantegan los datos grabados en memorias RAM, hay que refrescarlos cada cierto tiempo (cada 4 ms. aprox., unas 250 veces/sg.)

Memoria caché.

Ya es muy frecuente incorporar un tipo de memoria mucho más rápida que la principal (y más cara) que se denomina **memoria caché** (del inglés cash). Esta memoria es de tipo estático (SRAM) y se pone en medio entre dos dispositivos, uno más rápido y otro más lento que la propia memoria, de forma que el rápido sólo accede a la caché que al ser más rápida que el dispositivo lento tardará menos en servirle y por tanto el dispositivo rápido esperará menos tiempo.

Lo más normal, por ejemplo, es poner memoria caché entre el procesador (la CPU) y la memoria RAM (ver figura 2.5), para que, la caché, de menor tamaño que la RAM, actúe de

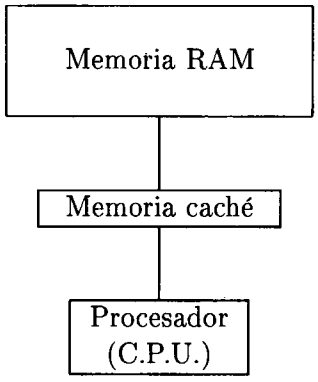


Figura 2.5: Posición de la memoria caché.

intermediaria. Así, cuando se va a usar un dato de memoria, se pasa de la RAM a caché, no solo el dato que se va a usar sino un bloque entero y se trabaja con ese bloque entero más rápidamente, ya que el procesador solo accede a caché, que es más rápida que la memoria principal. Cuando hace falta otro bloque, se graban en la RAM las modificaciones hechas en la caché (si se modificó algo) y de nuevo se pasa el bloque requerido, de la RAM a la caché.

Dependiendo del tamaño de la caché, esta puede almacenar varios bloques de la Memoria RAM.

La memoria caché acelera bastante los accesos a memoria y como esa función se usa muchísimo, el funcionamiento global del ordenador se ve aumentado. Tengamos en cuenta que el tiempo acceso a este tipo de memoria es del orden de 15 nanosegundos.

El rendimiento con memoria caché mejora debido a que cuando estamos usando un dato de memoria es muy probable que el siguiente dato a usar esté muy cerca (dentro de memoria) del dato que estamos usando. Si no fuera así, pudiera ser que la memoria caché no sólo no acelerara el sistema sino que incluso llegara a retardarlo. Imaginemos, por ejemplo, que la CPU necesitara en cada momento un dato de distinto bloque al que haya en caché, pues continuamente se estarían pasando bloques de RAM a caché y viceversa, con lo cual pudiera ser que se perdiera más tiempo que si se accediera directamente a la RAM. Afortunadamente, ese caso es muy improbable que ocurra.

Como es lógico, lo ideal sería que toda la RAM fuera de memoria caché, pero entonces saldría demasiado caro. Pensemos, por ejemplo, que el procesador 486DX2/66 de Intel, que puede manejar varios MegaBytes de RAM, incorpora dentro del mismo chip una pequeña memoria caché de 8 KB. Igualmente, los procesadores de Intel 486DX4/99 y Pentium, incorporan en el mismo chip una caché interna de 16 KB.

Pero hay que distinguir entre dos tipos de caché, la de *primer nivel* y la de *segundo*. La de primer nivel es la que se encuentra integrada dentro del chip del propio microprocesador y es a la que nos referíamos en el párrafo anterior. La de segundo nivel está fuera del chip y es por tanto menos efectiva, ya que la de primer nivel dispone de una vía de acceso directo. Normalmente, cuando se dice que un equipo tiene tantos Kbytes de caché (256, 512...) se está refiriendo a la de segundo nivel, ya que la de primer nivel es fija y más pequeña.

Para no complicar más el asunto no entraremos en discutir otra división de la caché en *write-through* y *write-back*. Baste indicar que la segunda es más efectiva, más compleja y más cara.

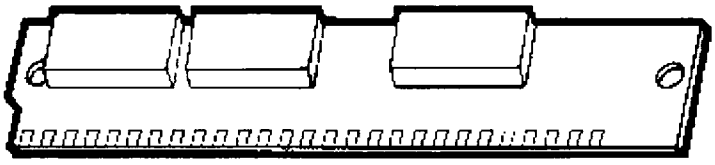


Figura 2.6: Módulo SIMM de Memoria.

Otro lugar muy usado para incorporar memoria caché es en los accesos a dispositivos de almacenamiento secundario (memoria masiva), como son discos duros y flexibles y otros similares. Así, se aceleran los accesos a disco, que suelen ser muy lentos. Este tipo de memoria caché suele ir incorporada en la tarjeta controladora de los dispositivos de almacenamiento en cuestión.

A veces, a la memoria caché se le denomina **memoria RAM estática** (**SRAM**), y a la memoria RAM convencional, **memoria RAM dinámica** (**DRAM**).

Otros aspectos de interés.

Otro concepto muy importante es el de **memoria virtual**, que consiste en una forma de hacer creer a los programas que se dispone de más memoria RAM de la que existe en el sistema. Para ello, lo que hace es almacenar una parte del programa en la RAM real y otra parte en la RAM virtual, que no es otra cosa que un medio de almacenamiento de acceso directo, como por ejemplo un disco duro. Usando este sistema (que debe ser gestionado por el sistema operativo), la limitación del tamaño de los programas y de su uso de memoria está limitada por el espacio libre disponible en el disco donde se simule la memoria virtual.

Si la memoria virtual es simular memoria RAM en disco, un **disco virtual**, consiste en simular un disco en memoria RAM. Es como si tuviésemos un medio de almacenamiento en disco, pero que realmente los datos se guardan en RAM. La utilidad de esto reside en que la memoria RAM es mucho más rápida que cualquier disco, por esto, si necesitamos ejecutar un programa que accede mucho a disco, podemos usar un disco virtual en RAM y así acelerar todos los accesos a disco. Pero ¡ojo!, si los ficheros de este disco nos interesan debemos copiarlos en un disco real (físico), ya que, como toda la RAM, se borrará al apagar el ordenador.

Actualmente la memoria se vende en los llamados módulos **SIMM** (ver figura 2.6), más eficaces que los antiguos módulos **SIPP**. Son unas pequeñas tarjetas en las que están implantados distintos chips de esta memoria. Estas pequeñas tarjetas se insertan en la placa principal (*motherboard*) perpendicularmente.

Existen dos tipos básicos de módulos SIMM, según el número de contactos (eléctricos) con la placa principal, de 30 y de 72 contactos, siendo estos últimos más rápidos y de mayores capacidades, que pueden llegar hasta los 32 MB por módulo.

Es interesante que la placa principal tenga zócalos de módulos SIMM de 72 contactos y que tenga varios libres, para poder ampliar la memoria sin tener que descartar los SIMM que ya tengamos instalados.

Poco a poco se están imponiendo las memorias **EDO** (*Extended Data Output*), que ofrecen mejores prestaciones en velocidad y fiabilidad. Su mejora consiste en precargar el valor de la siguiente posición de memoria en un pequeño *buffer* o *latch*. Por eso, el rendimiento de la memoria EDO es mucho mayor en sistemas sin caché.

Memoria ROM.

La memoria **ROM** (*Read Only Memory*, Memoria de solo lectura) es también de acceso directo , pero **sólo** se puede acceder a ella para **leer** lo que haya escrito. Su contenido es grabado por el fabricante y el usuario sólo puede leerlo, ni borrarlo ni escribir encima.

Es un tipo de memoria **no volátil**, esto es, que no se borra su contenido cuando se corta la alimentación. Sus datos permanecen siempre ahí invariables, por eso se han llamado también *memorias muertas*.

En este tipo de memoria vienen grabados programas o datos de gran interés para el usuario y sus programas. Por ejemplo, tenemos en ROM el programa para la puesta en marcha del ordenador, cómo cargar el S.O., control del teclado y muchas rutinas básicas para el funcionamiento (de Entrada/Salida, interrupciones...). Quizás lo más importante sea la **BIOS** (*Basic Input Output System*, Sistema Básico de E/S) que contiene los programas básicos que controlan todas las entradas y salidas de datos del computador.

Al encender un ordenador, la CPU empieza ejecutando un programa de arranque (**boot**) de la ROM, y es este el encargado de ver qué periféricos hay conectados, si funcionan, cuánta memoria hay disponible... en fin, hacer un chequeo del hardware. Posteriormente intenta cargar el S.O. desde disco (¡Ojo!, el S.O. no está en la ROM, ya que si fuera así no podríamos usar otro S.O.).

Las memorias ROM son más lentas que las RAM (suelen ser del orden de 200 nsg. de tiempo de acceso), esto es, que para leer un dato concreto se tarda más en leerlo de ROM que de RAM. Como la ROM contiene rutinas de uso muy común, como las rutinas de la BIOS por ejemplo, interesa tener esta rutinas en memoria rápida para no perder tiempo accediendo a ROM. Lo que se hace es copiar las rutinas de la ROM en memoria RAM e identificar esa zona de RAM con las direcciones propias de la ROM. Cuando se hace eso, a esa zona de la RAM se le llama **shadow ROM**, porque es como la *sombra* (*shadow*) de la ROM, reflejada en RAM.

Desde un punto de vista más hardware, existen distintos tipos de memorias ROM:

- **PROM (Programable ROM):** Son chips de memoria que se pueden programar una vez y no se pueden borrar, es decir, se puede guardar en ellos el programa o datos que queramos, pero una vez grabado ya no se puede cambiar.

- **EPROM (Erasable PROM):** Es una PROM que se puede borrar con rayos ultravioleta y volver a programar después. Los chips de EPROM suelen tener un adhesivo opaco pegado encima, de forma que no deje pasar la luz. Si le quitamos ese adhesivo, se descubre una ventanita por la que se puede ver la memoria. El adhesivo tapa dicha ventana para evitar que le de la luz, pues los rayos ultravioleta (del sol por ejemplo) la borran.

- **EEPROM (Electricaly EPROM, ó $\mathbf{E^2PROM}$):** Son EPROM pero que se borran eléctricamente, con lo cual son mucho más seguras y más útiles. A título orientativo, diremos que se pueden programar unas 100.000 veces y lo grabado dura más de 100 años, sin consumir batería. Dentro de esta podemos incluir la **memoria flash**, que es un tipo de memoria que se puede programar por software. Se usa, por ejemplo, en algunas BIOS o algunos modems (en los que los protocolos van grabados en esa memoria). Si queremos cambiar la BIOS o esos protocolos, respectivamente, basta con usar un programa que cambie el contenido de esa memoria flash.

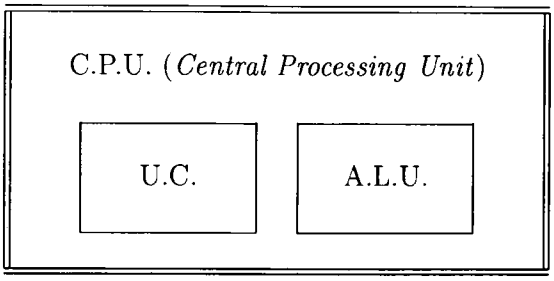


Figura 2.7: Elementos de la C.P.U.

2.3.2 La C.P.U., Unidad Central de Proceso.

La **C.P.U.** (*Central Processing Unit*) o **Unidad Central de Proceso** es el auténtico corazón del ordenador, es el elemento que controla y realiza la mayoría de las instrucciones y las más importantes. Se suele llamar también **procesador** y dependiendo de su tipo el ordenador será más o menos rápido y más o menos caro.

El procesador o CPU es como el director de una orquesta pero mucho más importante pues no solo se encarga de sincronizar y ordenar cuando se debe realizar cada operación, sino que además, es la CPU la que realiza la mayoría de las operaciones. Pero, sin los demás componentes no se puede hacer nada. De la misma forma que el director de una orquesta necesita a sus músicos, la CPU necesita elementos como la memoria principal o los periféricos de E/S.

El proceso de generación de cualquier chip no es simple, pero la generación de los chips procesadores requieren de la más avanzada tecnología. Su fabricación se hace en una **sala blanca** que no es una habitación pintada de blanco, sino una habitación perfectamente esterilizada y aislada para evitar que entre la más mínima mota de polvo.

En la sala blanca se manejan las *obleas*, que son placas de silicio, usadas para fabricar los chips. Dependiendo de la tecnología, puede contener desde 100.000 hasta más de 3 ó 4 millones de transistores. En algunos ordenadores muy potentes (como el CRAY), no se usa el silicio, sino el arseniuro de Galio.

En la figura 2.7 se puede observar que la CPU consta de dos componentes principales:

- **Unidad Aritmético-Lógica (ALU).**
- **Unidad de Control (UC).**

Algunos autores consideran que la CPU engloba también a la memoria principal, pero eso es una cuestión superflua. De todas formas, si nos atenemos al significado de CPU, la memoria no debe estar incluida, ya que la memoria no procesa nada, sólo sirve para almacenar información.

Dentro de la CPU existen unos elementos de memoria llamados **REGISTROS de la CPU**, que son usados como almacenamiento temporal de los datos que en cada momento son relevantes para el proceso.

Los registros de la CPU suelen ser del tamaño de una palabra de memoria, es decir, que cada vez que se lee o escribe de memoria se lee o escribe tantos bits como quepan en un registro de la CPU. A ese tamaño de bits se le llama tamaño de palabra de memoria. Así, se dice que el procesador 486 de Intel es de 32 bits, refiriéndose a su tamaño de palabra.

Una característica muy importante de los registros es su velocidad, es decir, las operaciones de lectura/escritura se realizan mucho más rápido que en memoria principal, e incluso más rápido que en memoria caché.

2.3.2.1 La Unidad Aritmético-Lógica (A.L.U.).

La **A.L.U.** realiza las operaciones de tipo aritmético (sumas, restas, multiplicaciones...) y de tipo lógico (comparaciones, operaciones del algebra de Boole, como NOT, AND, OR, XOR...). A veces sus operaciones se potencian con un **coprocesador matemático** que acelera estas operaciones. Este coprocesador es muy recomendable si se van a usar programas que requieran muchos cálculos matemáticos. En los chips modernos (como el 486DX, Pentium...), el coprocesador se incluye en el mismo chip que la CPU.

Básicamente la **A.L.U.** consta de:

- **Circuítos operacionales:** Son los circuitos digitales que hacen las operaciones.
- **Registros:** Donde almacena temporalmente los datos de E/S de los circuitos operacionales. Los registros principales son:
  - **Registro Acumulador:** Registro muy usado como almacenamiento temporal de algún operando y lugar de almacenamiento del resultado de la operación antes de llevar a memoria el resultado definitivo. Se le suele llamar AX.
  - **Registros de Operandos:** Donde se pueden almacenar otros operandos para ser usados. Se suelen llamar BX, CX, DX...
  - **Registro de Estado:** Un registro especial que indica el estado de la última operación. Cada bit indica algo en particular (si ha habido overflow, si se ha hecho una división por cero, si el resultado es negativo, o si es cero...).

La UC es la que determina, en cualquier caso, la operación que debe realizar la ALU, como veremos a continuación.

2.3.2.2 La Unidad de Control (UC).

Es el auténtico director de la orquesta que supone un ordenador (La ALU sería el virtuoso solista). La UC dirige las operaciones más importantes del ordenador, estableciendo la comunicación entre la ALU, la Memoria principal y el resto de componentes.

Además, detecta las señales eléctricas de estado (a través del *Bus de control*) procedentes de los distintos módulos del ordenador, que le indican el estado, situación o condición de funcionamiento de cada módulo, pudiendo así obrar en consecuencia.

La UC es principalmente la encargada de controlar la **ejecución de cada instrucción** del programa a ejecutar. Para ello, el programa en cuestión debe estar almacenado en la Memoria Principal (normalmente la RAM) del ordenador. Una vez cargado en memoria, la UC realizará las siguientes fases para ejecutar cada instrucción del programa en memoria, esto es, repetirá sucesivamente las siguientes dos fases hasta que el programa termine:

1. **Fase de captación de instrucción a ejecutar:** lee de memoria principal una a una las instrucciones del programa, empezando por la primera (obviamente). Para realizar esto, la U.C. tiene un **Registro Contador de Instrucciones** o **Contador de Programa**,

que contiene la dirección de memoria de la siguiente instrucción a ejecutar, esto es, su posición. Entonces, lee el contenido de ese registro, se va a memoria principal y lee la instrucción que haya en esa posición, es decir, lee el contenido de la dirección de memoria que indica el registro contador de instrucciones.

Ese contenido lo almacena en otro registro, llamado **Registro de instrucción en ejecución**, que contiene el código (en binario, en ceros y unos) de la instrucción a ejecutar.

El registro contador de instrucciones es como un índice, que nos dice la instrucción del programa cargado en memoria que se está ejecutando en cada momento. Por eso, al terminar esta fase se debe cambiar el valor de este registro, (sumándole una posición a la dirección), de forma que la instrucción a ejecutar sea la siguiente (la situada en la posición siguiente en memoria). Pero no siempre sucede así, ya que hay instrucciones que implican un salto en el orden de ejecución, de forma que al ejecutar la siguiente fase alterará el contenido del registro contador para que contenga la dirección de la instrucción que se debe ejecutar a continuación.

2. **Fase de ejecución de instrucción:** decodifica el código de la instrucción a ejecutar (almacenado en el registro de instrucción) y genera las señales de control convenientes para la ejecución de dicha instrucción.

   La UC dispone de un **decodificador** formado por circuitos especiales encargados de determinar qué se debe hacer, teniendo en cuenta el código de la instrucción a ejecutar y las señales de estado de los dispositivos que deban intervenir. O sea, la UC mira el código de instrucción y el estado de los dispositivos y genera las señales de control pertinentes para la ejecución de la instrucción.

   Estas señales de control encargarán una operación al dispositivo conveniente. Estos dispositivos pueden ser:

   - **ALU:** si la instrucción es una operación aritmética o lógica. Se trata de alguna operación con los registros de la ALU, dejando el resultado también en algún registro.
   - **Memoria principal:** si se desea leer o escribir datos. Esta transferencia será entre memoria y registros o viceversa.
   - **Canales de E/S:** si se desea efectuar alguna Entrada a Memoria desde algún periférico, o alguna Salida desde Memoria a algún periférico de salida. Esta transferencia se puede hacer pasando la información por los registros de la CPU, aunque es más rápido usar algún canal de acceso directo a memoria, es decir, sin pasar por la CPU. Esta última forma de acceso se denomina **DMA** (*Direct Memory Access*) y se encarga de realizarla algún chip especial. Por periféricos entendemos discos, cintas, impresoras, modem, tarjeta de sonido o vídeo...

2.3.3 El reloj.

Todas las operaciones del ordenador deben hacerse de forma sincronizada. No es posible que cada módulo (la memoria, la UC, la ALU...), actúe a su modo y realice las operaciones cuando quiera, de forma caótica. Para que todo funcione correctamente debe haber un elemento que

Figura 2.8: Un ordenador es más rápido que el hombre, en algunas operaciones.

sincronice las operaciones de forma que por ejemplo cuando la ALU vaya a operar ya tenga los operandos.

El elemento encargado de sincronizar todos los elementos y operaciones es el **RELOJ**, el cual es un circuito que emite un pulso a intervalos regulares de tiempo (ciclos), de forma que todas las operaciones se hagan dentro del pulso (ciclo) que les correspondan. A cada ciclo se le suele llamar *ciclo máquina* y es obvio que cada instrucción se ejecuta en un tiempo igual a un múltiplo de ciclos máquina (o sea, en un número entero de ciclos máquina).

La velocidad de *latido* del reloj, llamada **frecuencia** de reloj, se mide en **MegaHerzios (Mhz)**, de forma que 1 Mhz es igual a un millón de pulsos (ciclos máquina) por segundo. Es obvio que cuanto más deprisa vaya el reloj, cuanto mayor sea su frecuencia, más deprisa se ejecutarán las operaciones. Así, hay procesadores de microordenadores que van, hoy día, desde los 33 Mhz. de frecuencia de reloj hasta los 200 Mhz. o más.

Entonces... ¿Por qué no se puede coger un procesador y cambiarle el reloj por otro más rápido?. Pues por la sencilla razón de que si aumentamos demasiado la velocidad del reloj puede llegar un momento que los circuitos trabajen tan deprisa que se calienten demasiado y se queme el procesador. Cada procesador está pensado, y debe estar garantizado, para trabajar a una velocidad determinada.

Debido a ese sobrecalentamiento hay procesadores que incorporan un radiador encima de su chip para aumentar la disipación de calor, a otros se les pone un ventilador encima y hay algunos que hasta son metidos en un pequeño frigorífico de forma que puedan trabajar rápidamente sin sobrecalentarse.

Por supuesto, para medir la velocidad de un ordenador no solo basta saber la velocidad de su reloj, sino que cuentan muchas más variables, como el tamaño de palabra, si tiene coprocesador matemático, su juego de instrucciones (si es RISC o CISC)...

2.3.4 Los Buses.

Como hemos visto, un ordenador se divide en módulos independientes, cada uno con sus funciones claramente definidas. Para que todo funcione, además de todos los módulos, debe existir algún sistema para comunicarlos entre sí, de forma que operen todos los módulos en

armonía.

Como se ha visto, todos los módulos están conectados (y comunicados) por líneas de transmisión de datos, que no son más que un conjunto de cables (pistas o hilos conductores) que conectan los distintos módulos. Esos conjuntos de cables se llaman **BUSES** y suelen ser cables para los Periféricos de Entrada/Salida, e hilos conductores en una placa o tarjeta para conectar la Memoria Principal y el resto del Sistema.

Por cada cable (hilo o pista) se transmite un bit, en un momento determinado. Así, si tenemos que transmitir un bloque de muchos bytes, podemos transmitirlos todos los bits por un mismo hilo, transmitiendo un bit después de otro (**transmisión en serie**), o podemos transmitir de byte en byte, transmitiendo por 8 cables a la vez (**transmisión en paralelo**). En paralelo obviamente es más caro pues se necesitan más pistas, pero a cambio es mucho más rápido. Para cada tipo de transmisión existe un bus determinado.

Según el tipo de información que circula por ellos, se distinguen tres tipos de buses:

- **Bus de datos:** lleva datos de un lugar a otro para que se realicen operaciones con ellos o instrucciones para indicar qué operaciones se deben hacer con los datos.

- **Bus de direcciones:** lleva direcciones de memoria, es decir, indica dónde están los datos en la memoria.

- **Bus de control:** señales para controlar y sincronizar todos los componentes, así como para ver el estado de cada módulo. La UC (Unidad de Control) es la encargada del control, por eso, cada módulo debe tener un bus de control que lo comunique con la UC (véase figura 2.2).

El tamaño del bus de direcciones es muy importante, porque de él depende el máximo de memoria que podemos direccionar, es decir que podemos usar. Por ejemplo, si el bus de direcciones tiene 2 hilos (2 bits), solo podemos direccionar 4 palabras de memoria, es decir, la palabra 00, la 01, la 10 y la 11, y ya no podremos direccionar más, ya no podemos indicar otras palabras.

Si tenemos un bus de **n** hilos (**n** bits), podremos direccionar hasta $\mathbf{2^n}$ palabras o posiciones de memoria.

Esta es la razón por la que 1KB=1024 bytes, pues con un bus de direcciones de n=10, podremos direccionar hasta $2^{10} = 1024$ palabras. Para direccionar 1000 posiciones de memoria exactas, necesitamos igualmente 10 bits, por lo que estaríamos desperdiciando 24 posiciones, que podemos direccionar con el mismo costo del bus de direcciones.

Ya hemos visto que para que la CPU trabaje necesita recibir/enviar información (datos e instrucciones) al resto de los componentes, y que esto lo hace a través de los buses, los cuales son de anchura variable (número de bits), que depende del procesador.

Pues bien, en un ordenador podemos distinguir dos caminos que permiten este intercambio de información entre la CPU y el resto del sistema (dos tipos de buses dependiendo de las partes que conecten):

- El primer camino es entre la CPU (UC y ALU) y la memoria (**bus local**), donde los datos se transfieren a la máxima velocidad que permite el procesador (que es la frecuencia del reloj que comentamos en el apartado 2.2.3).

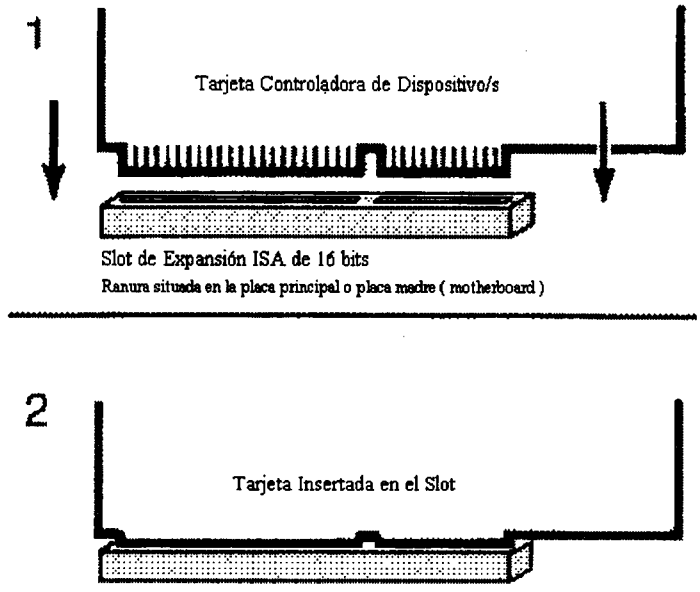


Figura 2.9: Conexión de una tarjeta en un slot ISA.

- El segundo camino es entre la CPU y el resto de componentes o periféricos, que se hace a través del llamado **bus de expansión** (pues permite que el ordenador crezca en tamaño y funcionalidad), o **bus del sistema** (*system bus*).

  Este bus de expansión puede ser de varios tipos (ISA, EISA, MCA...) y es lo que determina la llamada **arquitectura del ordenador**. La velocidad de este tipo de buses ya no es la máxima permitida por el procesador, ya que puede haber muchos periféricos conectados a ese bus y la CPU tiene que controlarlos todos. La memoria principal se conecta a otro bus más rápido debido a que se usa muchísimo y es muy importante que las operaciones de lectura/escritura en memoria se hagan lo más rápido posible (de ahí el nacimiento de la caché).

  Conectar un periférico al bus de expansión es tan simple como insertar la tarjeta controladora del periférico en una de las ranuras (más conocidas por **slots**) que existen en la placa principal (o placa madre, *motherboard*), donde están situados el procesador y la memoria, entre otros componentes (ver figura 2.9).

  La **tarjeta controladora** (ver figura 2.10) es una tarjeta de plástico duro donde se insertan y conectan diversos chips que realizan la función que sea, como controlar un disco duro o disquete, controlar el vídeo, transmitir por teléfono (modem), generación y tratamiento de sonidos...

2.3.4.1 Un poco de historia sobre buses.

El primer PC (1981, no hace tanto tiempo), empleaba un bus de expansión **ISA** (*Industry of Standard Arquitecture*), funcionando a 8 Mhz. y con un ancho de 8 bits. En principio esto parecía más que suficiente, ya que la CPU *corría* a 4,77 Mhz. y no solían conectarse muchos periféricos. Pero, los tiempos cambian, y los procesadores también. Con el nacimiento de

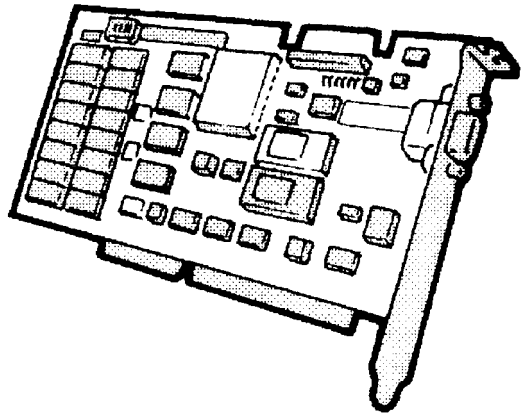


Figura 2.10: Una tarjeta controladora de dispositivo.

procesadores a 10 Mhz. ya se quedaba pequeña la velocidad de este bus ISA de 8 bits y lo modificaron para que funcionara con un ancho de 16 bits (véase figura 2.11)

Aún así, pronto se quedaron cortos esos 16 bits y el acceso a los periféricos se convertía en un cuello de botella por el que tenían que pasar muchos datos, lo cual ralentizaba todo el sistema. De hecho, a veces se afirma que un ordenador funcionará tan rápido como el más lento de sus componentes, ya que los más rápidos deberán esperar al más lento, perdiendo tiempo. Ante estos problemas, IBM inventó su bus **MCA** (*MicroChannel Arquitecture*), a 10 Mhz. (en algunos modelos 16 Mhz.) y 32 bits. Muy bonito y muy rápido, pero que era incompatible con las tarjetas fabricadas hasta ese momento, es decir, las tarjetas que servían para insertarlas en un bus de tipo ISA, no se podían insertar en un bus MCA, por lo que hacía falta una inversión económica mayor. Por eso, los principales fabricantes se unieron y crearon el bus de arquitectura **EISA** (*Extended ISA*), también a 32 bits, pero a 8,33 Mhz.

No acaban ahí los problemas. Nacieron ordenadores más rápidos y sobre todo crecieron las demandas. Por ejemplo, llegaron los interfaces gráficos (como Windows 3.1) que necesitaban transmitir al sistema gráfico más información que en una pantalla de texto normal, o sea, que necesitaba usar más y más deprisa el bus. Además, precisamente el sistema gráfico es uno de los más usados, pues el programa, normalmente, muestra sus resultados por pantalla (monitor) y es donde el usuario ve lo que está haciendo en cada momento. Por eso, mejorar la velocidad sólo del sistema gráfico redunda en una mejora del rendimiento general del sistema. A principios de 1993 se reunieron más de 120 compañías en el consorcio llamado **VESA** (*Video Electronics Standards Association*), y crearon una especificación para un bus especialmente rápido, denominado **VLB (VESA Local Bus)**. La idea era simple: hacer un bus que accediera directamente al procesador, como el que hay entre el procesador y la memoria. Se trata de implementar otro bus local al procesador, para poder *pinchar* ahí otros periféricos.

Para que el sistema VLB sea provechoso, las tarjetas deben estar específicamente diseñadas para ese sistema, es decir, no sirve cualquier tarjeta que se pinche (o inserte) en el slot de bus local VESA. Existen tarjetas VLB para multitud de utilidades pero las más usuales son para el sistema gráfico (por las razones antes aducidas) y como controladoras de discos.

El slot físico de VLB es una mezcla entre un conector ISA (de 16 bits) y otra parte de MCA (de otros 16 bits), un poco raro si no se sabe que el consorcio VESA intentó que los componentes que ya existían se pudieran seguir usando. Así una tarjeta ISA se puede usar en

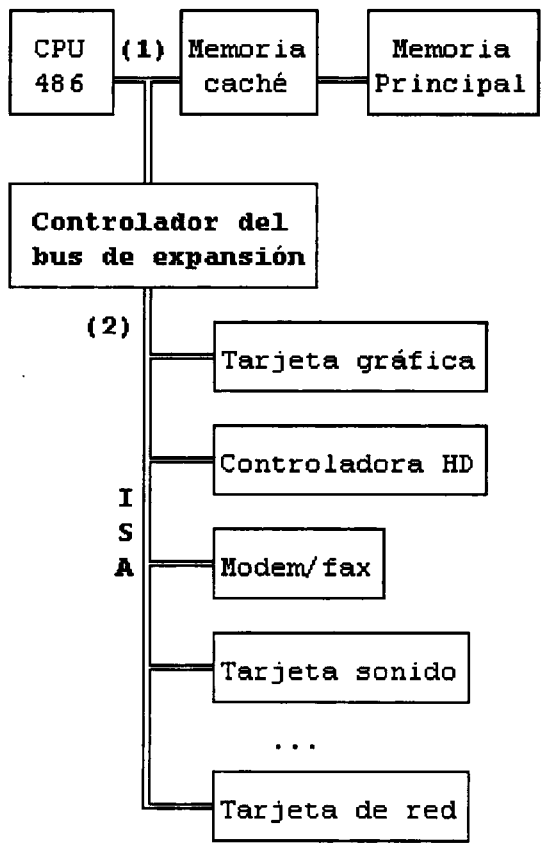


Figura 2.11: Interconexión de periféricos en un 486 con arquitectura ISA.

un slot VLB, obviamente sin sacar mejores rendimientos, ya que para ello, como hemos dicho, la tarjeta debe ser diseñada específicamente para VLB.

Pregunta: ¿Por qué no se conectan todos los periféricos al bus local y se gana en velocidad?. Sería muy fácil, pero no funcionaría, se cargaría tanto el procesador que se perderían las ventajas del bus local (además de otros problemas eléctricos como las capacidades parásitas que aparecen con altas frecuencias de funcionamiento). Esta es la razón por la que en la placa principal nunca se implementan más de 3 conectores de bus local, e incluso la eficacia de los 3 a la vez es bastante cuestionable.

Actualmente, VESA está desarrollando una especificación que trabaje a 64 bits (añadiendo un conector MCA de 32 bits). El panorama del bus local se complica cuando Intel saca su especificación **PCI** (*Peripheral Components Interconnection*, o Interconexión de componentes periféricos), otro sistema de bus local más optimizado que VLB, que está ya impuesto como un estándar y es el único aceptable para sistemas Pentium o mayores. La especificación PCI (véase figura 2.12) está pensada para que el bus funcione a 33 Mhz., en cambio, la VESA no, hasta que no saque una norma nueva, que podría ser VESA-2.

Existe otra tercera especificación de bus local, el **QuickRing** de Apple, el cual supera a VLB y a PCI (es más de 3 veces más rápido) pero, por ahora, es específico para cada procesador de la serie 680x0 de Motorola. Mención aparte requiere el bus **PCMCIA**, especialmente diseñado para ampliar memoria a un ordenador portatil, pero que ya se ha extendido para todo tipo de usos (memoria, modems, fax, tarjetas de red local, comunicaciones celulares, discos duros, tarjetas de sonido, procesadores de encriptación digital...).

Aunque se suele decir que sus siglas responden a *People Can't remember Computer Industry Acronyms* (la gente no puede recordar los acrónimos informáticos), lo cierto es que significan *Personal Computer Memory Card International Association* (Asociación Internacional para tarjetas de Memoria de ordenadores personales). En realidad, esta asociación, formada por unas 500 empresas, define los estándares para las llamadas tarjetas o dispositivos PCMCIA (PC Card) que son dispositivos de pequeño tamaño (algo más que una tarjeta de crédito), ideales, pero no exclusivas, para ordenadores portátiles.

Existen, por el momento, 3 tipos de tarjetas PCMCIA y sus respectivos tipos de zócalos (slots o ranuras) donde se conectan. Todas ellas tienen la misma longitud y anchura (85,6 x 54 mm.) e igual interfaz de 68 contactos. Su diferencia, básicamente está en su grosor:

- **Tipo I:** grosor hasta 3,3 mm. Se usa, principalmente, para tarjetas de memoria (RAM, EEPROM...). Los zócalos de este tipo sólo admiten tarjetas de este tipo.
- **Tipo II:** grosor de hasta 5,5 mm. Usados, principalmente, para modems (hasta 28.800 bps.), fax/modems, tarjetas de conexión a redes locales y otros tipos de dispositivos puramente electrónicos. Los conectores suelen ser del tipo usado para teléfonos (RJ11) o de tipo 10Base-T. Los zócalos de este tipo admiten tarjetas tipo I y tipo II.
- **Tipo III:** grosor máximo de 10,5 mm. Básicamente usados para almacenamiento de información (discos duros de más de 80MB y menos de 15 mlsg.). Los zócalos de este tipo admiten tarjetas de cualquiera de los 3 tipos.

Normalmente, los zócalos PCMCIA se conectan a un bus ISA, pero también existen implementaciones en PCI, SBus y NuBus.

La idea de PCMCIA fue el poder configurar líbremente el ordenador de una forma fácil y flexible. Su idea original de *Plug&Play* (conectar y utilizar) no funcionó muy bien al principio

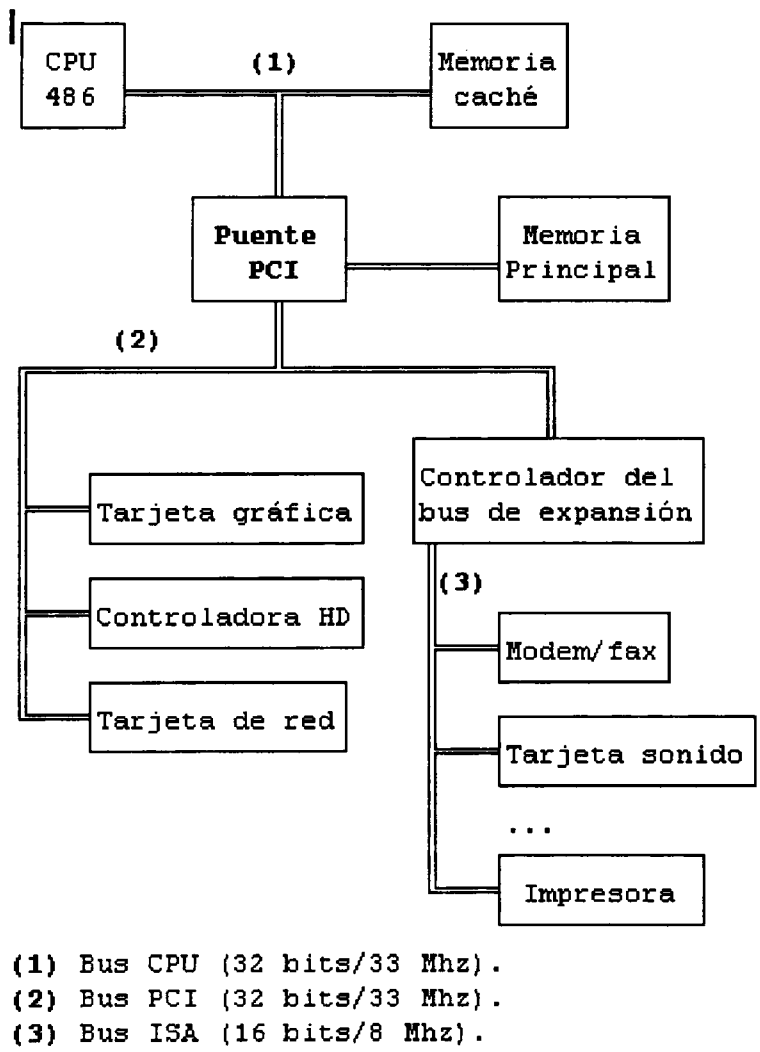


Figura 2.12: Interconexión de periféricos en un 486 con arquitectura de bus local PCI.

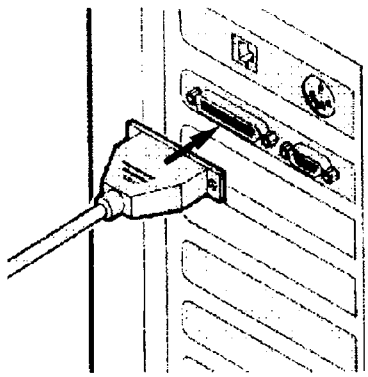


Figura 2.13: Puertos traseros de un ordenador, donde se pueden conectar periféricos.

y, de hecho, hoy día la mayoría de las PC Card funcionan en casi todos los sistemas, generalmente sin necesidad de reiniciar el ordenador cada vez que se conecta una de estas tarjetas, pero nadie garantiza que esto sea así.

El futuro está por determinar, pero pronto saldrá la especificación CardBus que aporta muchas prestaciones, entre ellas: bus de datos de 32 bits, 20 Mhz. de velocidad de funcionamiento, bajo consumo (máximo 3,3 V.), posibilidad de tarjetas multifunción (hasta 8 funciones en una sola tarjeta)...

La razón principal del poco uso de estos periféricos es la ausencia de soporte, para esta tecnología, en los sistemas operativos. Casi exclusivamente la aceptan sin problemas el S.O. de IBM OS/2, desde su versión 2.1. y casi seguro también funcionarán bien en el S.O. de MicroSoft, Windows'95.

La especificación *Plug&Play* (PnP) consiste en un conjunto de técnicas cuyo objetivo es conseguir que la instalación y configuración del ordenador resulte tan sencilla como conectar las tarjetas y los cables (eliminando la configuración de jumper, conmutadores DIP, interrupciones -IRQ-, puertos y direcciones base). Para ello, se necesita una señal especial en el bus, por lo que ya no sirve el bus ISA. Los bus EISA, PCI y VL-bus ya disponen de estas señales en sus *slots*.

2.3.5 La placa principal o madre.

La placa madre, principal o *motherboard*, es el componente sobre el cual se montan todos los demás. Es una superficie plástica donde se insertan todos los chips de todos los componentes. Es la placa de mayor tamaño en un ordenador. El rendimiento global del equipo depende mucho del diseño de esta placa y de el llamado *chipset* que esta tenga.

Los componentes principales de la placa madre son:

- Zócalo para el procesador: Es donde se insertará el procesador. Modernamente se han extendido los llamados zócalos ZIF (*Zero Insertion Force* o Fuerza de Inserción nula) que facilitan mucho la colocación y el cambio de procesador. El zócalo ZIF es un cuadro de color blanco con muchos agujeritos (para los pines del procesador) y una palanca para insertar o liberar el procesador.

- Zócalos para la memoria RAM: Es donde se insertarán los módulos SIMM de la memoria (ver figura 2.6).

- Zócalos para la memoria caché: Aunque a veces los chips de memoria caché van soldados directamente en la placa, otras veces van en un zócalo especial (CELP).
- Chips para la memoria ROM.
- El chipset: Es el conjunto de chips que se encargan de controlar la conexión con la CPU, control de la memoria RAM y caché, arbitrar los buses (PCI, ISA...). Es famoso el chipset Tritón de Intel, aunque hay otros fabricantes como SIS, Opti, UMC u Octek.
- Slots: Son las ranuras en las que se pueden insertar otras tarjetas para ampliar las posibilidades del PC (ver figura 2.9). En la actualidad se suelen incluir slots ISA (de 16 bits) y slots para bus local PCI.
- Diversos controladores: A veces se incluyen en la misma placa principal controladores para diversos periféricos como Discos Duros y CD-ROM (normalmente EIDE), discos flexibles, controladora gráfica e incluso controladores de sonido. Con esta tendencia se busca la reducción de costes y del número de tarjetas insertadas en los slots de la placa.
- Chips de control de puertos: Se incluyen también chips para el control de puertos serie (chips de UART), paralelo y para joystick. El más moderno chip UART es el 16550AFN.
- Zócalo VRM (*Voltage Regulator Module*): Se usa para controlar el voltaje al que debe funcionar el procesador. Podemos instalar un procesador en una placa y después, si queremos cambiar de procesador a otro mejor podremos hacerlo independientemente del voltaje con el que este trabaje.
- Chip controlador del teclado y conector para el mismo.
- Generador de reloj en tiempo real.
- BIOS y memoria CMOS: La BIOS (*Basic Input/Output System*) es, como ya hemos dicho, un conjunto de pequeños programas para controlar las entradas y salidas de datos en el sistema. Son rutinas, por ejemplo, para escribir en memoria de vídeo, manejar las unidades de disco, obtener información del estado del ordenador... Se suele incluir un programa llamado *setup* con el que se configuran muchos aspectos del equipo: Unidades de disco, fecha y hora, activación de la caché, control del bus PCI, modos de bajo consumo... Estos datos pueden variar, por lo que no podemos guardarlos en ROM, pero interesa que no se pierdan al apagar el equipo. Por tanto se almacenan en una pequeña memoria RAM de bajo consumo alimentada por una batería. Se le suele llamar memoria **CMOS** (*Complementary Metal Oxid Semiconductor*) por la tecnología con que está construida, pero a veces también se la llama como memoria *no volátil* (aunque también es no volátil la ROM). La batería dura unos 7 años aproximadamente. Otras características de la BIOS deben ser que permita discos duros de más de 1024 cilindros, auto-detección de discos duros, Green PC (la norma *Energy Star*) y que sea *Plug and Play.*
- *Jumpers* de configuración: Los *jumpers* son pequeños elementos que se usan para cerrar (juntar) un par de pines situados en la placa. Dependiendo de si estos están puestos o no pueden configurar diversos elementos: Tipo de microprocesador, voltaje, cantidad de memoria caché...

- Conexión para LEDs: De la placa suelen salir cables para diversas luces (LEDs, Diodos emisores de Luz) del panel frontal.

2.4 Los periféricos más típicos.

Los periféricos son aquellos dispositivos que nos ayudan a comunicarnos con el ordenador. Son dispositivos que se le añaden al ordenador para que este pueda comunicarse con el exterior, tanto para captar datos como para mostrarlos.

Se pueden clasificar en 3 tipos:

- **Periféricos de Entrada:** permiten al ordenador leer datos del exterior. Son como traductores que nos traducen nuestros datos o nuestras órdenes a su críptico lenguaje binario.

- **Periféricos de Salida:** permiten al ordenador escribir o mostrar datos, resultados... Básicamente, hacen lo contrario a los anteriores, traducir los resultados obtenidos en código máquina a un formato más humano.

- **Soportes de información, Medios de Almacenamiento o Memoria Masiva:** los periféricos de este tercer tipo son de entrada, pero también lo son de salida, y además no nos traducen la información, sino que su objetivo es almacenar la información para que la podamos recuperar en cuanto queramos.

Hay que decir también, que algunos periféricos tienen un uso dual y son tanto de Entrada como de Salida.

Visto todo lo anterior, podemos establecer una relación entre el ser humano y un computador:

- El Cerebro es como el computador central, cuyas tareas son:
  - Memorizar información (Memoria Principal).
  - Realizar cálculos y razonamientos (ALU).
  - Controlar todo: esos cálculos, la memoria y el resto del cuerpo (UC).

- Los 5 sentidos son los dispositivos (periféricos) de Entrada, por los que el hombre recibe toda la información que necesita y procesa (vista, tacto, olfato, oído y gusto).

- El resto del cuerpo puede verse como los periféricos de salida, por los que transmitimos información al exterior, principalmente con la voz, manos, pies, piernas, gestos...

La memoria masiva auxiliar puede ser una agenda donde guardamos:

- Información que no merece la pena retener en la cabeza (o memoria principal), por su poca importancia o por ser demasiada información.

- Información que deseamos tener duplicada por razones de seguridad.

2.4.1 Soportes de Información.

Justo ahí arriba hemos visto su utilidad, que consiste en soportar y guardar la información en código binario. Esta información, básicamente, se guarda en forma de ficheros. Un **fichero** o **archivo** es un conjunto de información al que el ordenador o el usuario se puede referir con un único nombre. Así, yo puedo tener multitud de ficheros cada uno con un nombre, como por ejemplo un fichero llamado CARTAMOR en el que guardo el texto de una carta de amor, o PAZ en el que guardar un dibujo de una paloma, o un fichero PATRI para guardar una foto de Patricia, un fichero CONTA que sea un programa que mantenga la contabilidad de una empresa o incluso puede ser un fichero HDS que guarde una canción de Héroes Del Silencio. En fin, las posibilidades son ilimitadas. Hay que destacar que algunos S.O. (Sistemas Operativos) establecen unas reglas más o menos rígidas para los nombres de los ficheros. Estos ficheros, cuando son tratados por el ordenador, se cargan, como hemos visto, en la memoria RAM, donde se almacenan temporalmente mientras se procesan (ver figura 2.14). Podría plantearse la pregunta: ¿por qué no almacenamos todos los ficheros siempre en RAM? La respuesta es simple:

- La memoria RAM es pequeña y cara, por lo que no cabrían todos nuestros ficheros. Los medios de almacenamiento son más baratos, capacidad casi ilimitada, aunque, eso sí, son más lentos.
- La memoria RAM es volátil, por lo que los perderíamos al apagar el ordenador... o tendríamos que dejarlo siempre encendido.
- Siempre hay necesidad de transportar información (ficheros) de un lugar a otro.

Los soportes de información o medios de almacenamiento también se conocen como **memoria masiva**, **auxiliar**, **secundaria** o **externa**, haciendo referencia a que es en teoría ilimitada (podemos guardar todos los ficheros que queramos), puede ayudar a la RAM cuando esta se queda pequeña y suele ser externa al ordenador.

Los medios de almacenamiento pueden ser de dos tipos: **Reutilizables** y **NO Reutilizables**. Los primeros están hechos de forma que una vez utilizados se pueden reutilizar, es decir, una vez que ya han sido grabados pueden borrarse y grabarse otra vez, cosa que no se puede hacer con los segundos.

Atendiendo a otro criterio, los medios de almacenamiento pueden ser **Removibles** y **NO Removibles**. Los removibles son aquellos soportes específicamente diseñados para ser trasladados de un sitio a otro (discos flexibles por ejemplo), y los no removibles son aquellos diseñados para permanecer dentro del ordenador y mantener ahí su información (como los discos duros, aunque también hay discos duros removibles).

Antes de seguir, vamos a hacer una distinción por si hubiera alguna duda: Una cosa es el soporte de información, es decir, donde físicamente se guarda la información, y otra cosa es el dispositivo que lee (o escribe) el soporte para comunicarse con el ordenador. Cada soporte diferente tiene su unidad o dispositivo diferente de lectura/escritura. En muchos casos, cuando son NO removibles, ambas partes forman un conjunto indivisible, pero cuando son removibles, en la mayoría de los casos, lo que es removible es el soporte en sí y no el dispositivo que los lee. De esta forma, grabamos la información en el soporte y para llevarla a otro lado es necesario que el ordenador de destino tenga una unidad lectora del soporte en cuestión.

Otra cosa importante es que cada dispositivo debe disponer de un conjunto de chips que lo controlen. Normalmente existen las llamadas **tarjetas controladoras**, que se insertan en

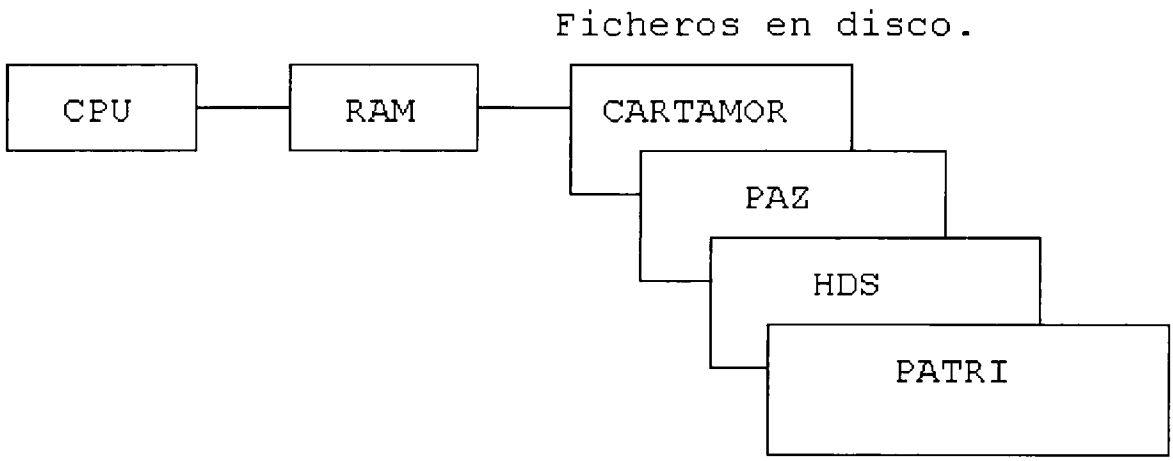


Figura 2.14: Acceso de la CPU a los ficheros.

un slot de expansión del ordenador y de aquí sale un cable (bus) que va directamente al dispositivo que debe controlar. Una tarjeta puede servir para varios dispositivos, ahorrando espacio y dejando así más slots libres para otros periféricos.

Veamos un esquema (figura 2.15) de cómo accede el procesador a los datos almacenados en un medio de almacenamiento cualquiera:

1. La CPU indica a la controladora qué dato quiere leer. Esto depende del dispositivo (en el caso de discos, tendrá que especificar de qué disco, qué cara del disco, qué pista y qué sector es el que quiere leer).

2. La tarjeta controladora toma esa orden y la traduce para que la cabeza lectora se posicione en su sitio.

3. La cabeza detecta las variaciones de flujo y las transmite a la controladora.

4. De la controladora pasa a memoria RAM el sector leído.

5. La CPU lee de la RAM el dato requerido.

A la hora de elegir un método de almacenamiento secundario hay que tener en cuenta los siguientes puntos:

1. **Portabilidad:** si nos interesa que sea o no removible.

2. **Capacidad:** sepende de lo que pensemos almacenar.

3. **Compatibilidad:** que sirva en muchos equipos.

4. **Interfaz (o interface):** que nos sirva en nuestro equipo.

5. **Rapidez:** depende de su uso, pero invertir aquí es rentable.

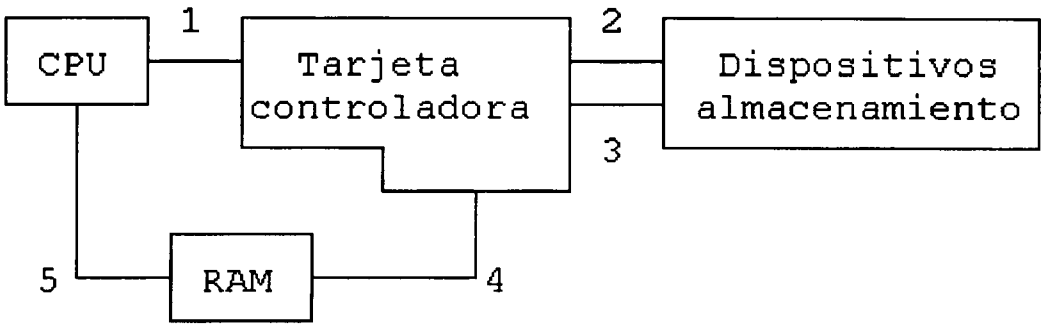


Figura 2.15: Acceso del procesador a dispositivos.

6. **Seguridad:** normalmente suelen ser muy seguros, pero...

7. **Precio:** depende de todas las características anteriores.

La velocidad de estos dispositivos se mide en dos factores: **tiempo de acceso**, que es el tiempo medio que tarda en llegar al lugar en donde se sitúa un dato concreto que se suele medir en milisegundos, y la **velocidad de transferencia**, que es el número de bytes que puede leer por unidad de tiempo (KB/sg).

La clasificación más normal es la que los divide según su tecnología de almacenamiento, **según la forma que tienen de almacenar la información**. Según esta tecnología, pueden ser:

- Soportes NI magnéticos NI ópticos.
- Soportes MAGNÉTICOS.
- Soportes OPTICOS.
- Soportes HÍBRIDOS (Magneto-Opticos, MO).

2.4.1.1 Soportes NI magnéticos NI ópticos.

Hoy día ya están obsoletos, pero en su día fueron una revolución, ya que hasta entonces había que darle al ordenador los datos *a mano,* es decir, introduciendo manualmente en el ordenador bit a bit todos los programas y datos. y estos soportes supusieron una mecanización del proceso. Fueron utilizados hasta finales de la década de los setenta.

Eran soportes NO reutilizables y de acceso lento. Se basaban en agujerear un papel o cartulina, de forma que una vez agujereado ya no había forma de reutilizarlo de nuevo. Los más usados fueron:

- Tarjetas perforadas.
- Cinta de papel perforada.

2.4.1.1.1 Tarjetas perforadas. Era una cartulina rectangular con una esquina cortada (para distinguir fácilmente su posición correcta y que no se dieran la vuelta accidentalmente).

Cada tarjeta tenía un número de columnas determinado (80 en las famosas tarjetas **Hollerith**), de forma que cada columna significaba un carácter. Cada carácter se codificaba con un número de bits determinado (12 en las Hollerith), de forma que en cada columna se perforaba en las posiciones que correspondieran a los bit a 1 y se dejaba sin perforar en las posiciones de los bits a 0.

Los códigos más usados para codificar los caracteres eran el código Hollerith (nombre de su inventor) y el ASCII.

Los sistemas lectores de tarjetas perforabas usaban una célula fotoeléctrica para detectar donde había agujero y eran dispositivos muy lentos (entre 200 y 2000 tarjetas por minuto).

2.4.1.1.2 Cinta de papel perforada. Consistía en un rollo de cinta, donde se perforaba de la misma forma que en las tarjetas. El código más usado fue el ASCII y, aunque eran más rápidas que las tarjetas, seguían siendo demasiado lentas.

2.4.1.2 Soportes MAGNÉTICOS.

Son siempre reutilizables. Constan de una base metálica o plástica y sobre esta, una capa de material magnético (óxido de hierro en muchos casos). Ese material influye en la velocidad de lectura de la información almacenada. En esta capa magnética se crean campos magnéticos que determinan la grabación y posterior lectura. El código binario representado por 0 y 1 es almacenado mediante cambios de orientación en los campos magnéticos (llamados transiciones de flujo), es decir, por su polarización (positivo-negativo, norte-sur).

La información, los bits, se almacenan físicamente en serie, es decir, en forma de *fila india*, un bit seguido de otro y así todos los bits.

El elemento encargado de grabar los cambios de polarización en el disco y de leer esos cambios para ver qué hay grabado en el disco se llama **cabeza** de lectura/escritura.

Estos tipos de dispositivos magnéticos no se deben acercar nunca a fuentes magnéticas (altavoces, imanes...), pues corren el riesgo de perder la información que tuvieran y quedar inutilizados. Además, nunca se debe tocar la superficie magnética con la mano u otro instrumento y se deben tener alejados de lugares polvorientos. La temperatura ideal es entre los 10-50° C.

En estos dispositivos la información puede leerse indefinidamente, si se tratan con cuidado y se usan unidades de lectura/escritura limpias. En las cintas el riesgo es mayor, ya que la información se sitúa en capas adyacentes y los campos magnéticos de una capa afectan a las adyacentes haciendo que la información se deteriore con el tiempo.

Los más importantes tipos de soportes de información magnéticos son los siguientes:

- Cinta magnética.
- Fichas o tarjetas con bandas magnéticas.
- Discos duros (Hard Disk).
- Discos flexibles (Floppy Disk).
- Discos Bernoulli.
- Discos SyQuest.

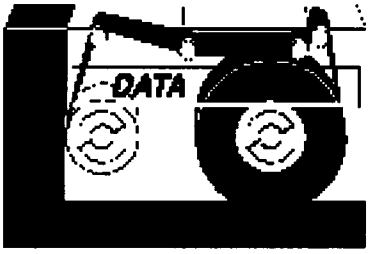


Figura 2.16: Una cinta magnética.

2.4.1.2.1 Cinta magnética. Es un sistema bastante antiguo pero que aún se sigue usando. Consiste en un rollo de cinta (tape) magnética en la que se graba la información secuencialmente, en posiciones contiguas y su recuperación o lectura se realiza en el mismo orden.

Hay distintos tipos, desde la típica y cotidiana cinta de cassette para música (que ya no se usa como soporte informático) hasta rollos de cinta industriales de más de 30 cm. de diámetro usados para almacenamiento masivo de información de poco uso y de acceso secuencial.

Los más usados son unas cintas en cartuchos, denominadas **streamer**, que se usan sobre todo en sistemas informáticos medianos o grandes para hacer copias de seguridad, es decir, para hacer una copia de toda la información, por si se estropeara poder volver a recuperarla. Es un sistema muy adecuado para copias de seguridad debido a su gran capacidad, y el poco espacio que ocupan (ver figura 2.16).

Existen distintos sistemas de cinta en cartuchos, como los llamados QIC y DAT (*Digital Audio Tape*), entre otros. Los segundos son más caros pero más rápidos.

La capacidad de una cinta depende lógicamente de la longitud del rollo y de la densidad de grabación. La densidad de grabación se mide en **bpi** (bit per inch, o bpp, bits por pulgada), es decir hace referencia al número de bits que se pueden grabar en un trozo de cinta de una pulgada de longitud.

Existen diversos formatos de cintas y de grabación que son muy universales, es decir, que usan muchas organizaciones (sobre todo bancos y grandes entidades).

Ventajas:

- Gran capacidad (en poco espacio físico).
- Bajo coste.
- Universalidad.
- Removibles.

Desventajas:

- Acceso secuencial.
- Baja velocidad de acceso.
- Poco fiables (sobre todo a largo plazo).

2.4.1.2.2 Tambores magnéticos. Aparecieron después de las cintas y han sido desplazados por los discos duros. Su interés es meramente histórico. Estaban formados por un cilindro vertical que gira alrededor de su eje central. La superficie curva del cilindro o tambor está recubierta de material magnético y se divide en pistas de igual anchura, de forma que para cada pista existía una cabeza de lectura/escritura que accedía a los datos de esa pista.

Su mayor innovación consistía en ser de acceso directo, es decir, ya no era necesario (como en las cintas) recorrer todo lo grabado hasta encontrar el dato que necesitábamos, sino que se accedía directamente a la posición de ese dato y se leía (o escribía).

Ventajas:

- Acceso Directo.

Desventajas:

- Grandes Dimensiones.
- Poca capacidad.
- No removibles.

Pronto fueron sustituidos por los discos duros, pero su imagen (la de un cilindro) sigue siendo usada en esquemas informáticos para referirse a los discos duros o a que determinados ficheros se almacenan en disco.

2.4.1.2.3 Fichas o tarjetas con bandas magnéticas. Estas tarjetas permiten almacenar información en la banda magnética que llevan pegada. Se utilizan en contabilidad y sobre todo en las tarjetas de crédito y de los cajeros bancarios automáticos, donde nada más con meter la tarjeta el cajero la lee e identifica al propietario de la misma.

Las ventajas son su facilidad de trasporte y las derivadas de su uso (sacar dinero en el cajero...). La desventaja es su pequeñísima capacidad, por lo que no son válidos como almacenamiento masivo de información.

2.4.1.2.4 Discos duros (Hard Disk). Fueron pioneros en este campo los discos con tecnología Winchester, pero hoy ya ha sido muy mejorada esta tecnología.

Suelen estar formados por varios (entre 2 y 10 aprox.) discos de una aleación de aluminio, recubiertos de un material magnético por sus dos caras, y que giran alrededor de su eje central (a unas 5000 rpm). Su tamaño oscila entre los 10 cm. de diámetro, pero se va reduciendo con el tiempo y los avances tecnológicos.

Estos discos se colocan uno sobre otro, dejando espacio entre ellos para que se desplace la cabeza de lectura/escritura, y poder leer o escribir en ambas caras de cada disco (ver figura 2.17). Es decir, si el disco duro tiene 5 discos, contendrá 10 cabezas de lectura/escritura, una por cada cara de cada disco. La cabeza se desplaza muy cerca de la superficie del disco, pero sin llegar a tocarla, debido a la altísima velocidad de giro. Un simple roce de la cabeza con la superficie podría dañar la superficie o, en el peor caso, la propia cabeza.

En la figura 2.18 podemos ver que cada cara de cada disco está dividida en **PISTAS** circulares concéntricas. El número de pistas depende de cada disco en particular. A su vez,

Figura 2.17: Interior de un Disco Duro.

cada pista está dividida en **SECTORES** angulares, y es en estos sectores donde se almacenan los bloques de información, de tamaño fijo (512 bytes para el S.O. DOS).

Resumiendo, es como si dividimos el disco por medio de círculos concéntricos (las pistas) y luego dividimos todas las pistas mediante varios radios, atravesando todas las pistas y de igual ángulo entre dos radios sucesivos. Cada porción de pista dividida entre dos radios consecutivos es un sector.

Para poder leer o escribir un sector del disco se deben de especificar los siguientes datos:

- Unidad: indica el disco que queremos leer/escribir.
- Superficie: indica la superficie dentro del disco.
- Pista: la pista concreta de esa superficie.
- Sector: el sector concreto de esa pista.

Pero estas cuestiones son transparentes al usuario, es decir, esas son tareas que hace Sistema Operativo automáticamente.

Podemos observar que los sectores interiores (más cercanos al eje) son de menor tamaño que los exteriores, sin embargo, en todos se almacena la misma cantidad de información, la misma cantidad de bits. Para comprender la causa de esto hay que tener en cuenta que el disco gira a velocidad constante, por lo que la cabeza lectora/escritora tardará lo mismo en leer un sector interno que uno externo, pero el sector externo pasará a mayor velocidad lineal (la velocidad angular, en revoluciones por minuto, es constante). Así, en los sectores internos, la densidad de grabación es mayor, es decir, hay los mismos bits en menos espacio.

Al proceso de crear, en un disco nuevo (*virgen*), las pistas y sectores se llama **formatear** y es una función que deben incorporar los Sistemas Operativos. Por eso, cada S.O. puede formatear discos de una forma. Veamos el caso del S.O. MS-DOS, como caso general: Al formatear, el DOS, graba al principio de cada sector una marca que define sus características (número del sector, número de pista, cara del disco y tamaño), y verifica que el estado de la superficie magnética de ese sector está en perfecto estado. Esto se hace grabando una secuencia de bits, volviéndola a leer y comparándolas. Al final del sector va grabado un

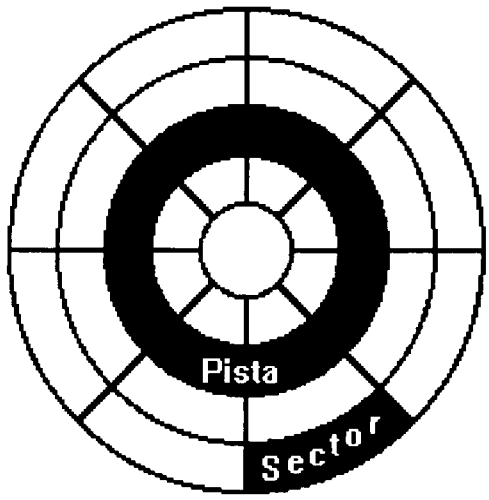


Figura 2.18: Pistas y sectores en la cara de un disco.

Código de Redundancia Cíclica (CRC) que sirve para comprobar si la información que se lee es correcta o si ha habido algún fallo. Además, crea los índices para acceder a los ficheros que se graben, la F.A.T. (*File Allocation Table*), que es una tabla que mantiene la posición de todos los ficheros en el disco, sectores erróneos, sectores libres...

Al conjunto de todas las pistas que están en la misma vertical de un disco duro se le llama **cilindro**. Es decir un cilindro lo forman todas las pistas a igual distancia del eje de giro. Tomemos por ejemplo la pista más exterior de cada disco, pues un cilindro será el formado por todas las pistas más exteriores de todos los discos (no hay que olvidar que cada disco tiene dos caras y por tanto dos pistas más exteriores que el resto). O sea, si el disco duro tiene 5 discos, cada cilindro tendrá 10 pistas. Un disco duro tendrá tantos cilindros como pistas tenga cada cara de cada disco.

También está el concepto de **cluster**, **grupo** o **unidades de asignación**, que lo forman varios sectores, pero esto depende mucho de cada disco y del sistema operativo. Un fichero tiene que ocupar como mínimo un grupo, de ahí que no cuadren las cuentas entre bytes grabados y espacio libre (si sumamos esos valores no saldrá el espacio total del disco).

La capacidad de cada disco aumenta con la técnica. Los primeros no llegaban a un MegaByte, pero actualmente están entre 860 MB y varios GB, aunque lo más normal es que no superen los 1,2 *gigas*. Su precio depende, lógicamente, de su capacidad.

Permiten acceso directo, ya que es posible ir directamente a la pista y sector que deseemos leer.

Suelen estar dentro del ordenador, por lo que no son fácilmente desplazables (no removibles), aunque existen **discos duros removibles** (docks), que se pueden colocar fácilmente en una unidad lectora y ser transportados de un ordenador a otro. El principal inconveniente de los discos removibles es que las cabezas tienen que ir dentro de la caja que contiene los discos, y para evitar que en los traslados las cabezas golpeen con los discos estropeándolos, los fabricantes construyen las cabezas directamente sobre el material magnético, lo cual repercute en una menor velocidad de giro y, por tanto, en una menor velocidad de lectura/escritura.

Existen otro tipo de discos duros removibles, los llamados **discos duros de bolsillo**, los

cuales son de pequeño tamaño y se conectan al puerto paralelo del ordenador. Los hay de hasta 1,8 pulgadas de diámetro y de más de 600 MB, y son ideales para su uso en portátiles. Casi exclusivos de los portátiles son los discos en tarjeta **PCMCIA** (Tipo III), pues se conectan en ranuras PCMCIA, típicas de los portátiles. Estos últimos son del tamaño de una tarjeta de crédito, pero algo más gruesos.

Los discos duros, en general, son muy delicados y se deben manejar con cuidado para no dañar la superficie de los discos y las cabezas. Son tan delicados que una simple mota de polvo puede estropear muchos sectores del disco, por lo que siempre van en pequeñas cajas herméticamente cerradas que no se deben abrir nunca. Los antiguos discos disponían de un programa para *aparcar las cabezas* (park, ship...), que situaba las cabezas en una zona del disco especial (llamada **Lzone**), sin datos, de forma que fuera más difícil que se estropearan en los traslados, pero los modernos no los necesitan porque siempre se aparcan automáticamente (auto-Park). Aún así, se debe tener cuidado al transportar un ordenador que contenga un disco duro.

Si durante el proceso de lectura/escritura de un disco duro se va la luz o se apaga el ordenador, la cabeza aterrizará brúscamente en la superficie de la pista que estuviera leyendo o escribiendo, y podría producir serios daños al disco. Es por esto por lo que no se debe apagar un ordenador mientras se está leyendo o escribiendo en el disco, lo cual se suele indicar mediante una luz (LED) en el frontal de la máquina.

La velocidad de lectura/escritura en este medio es bastante rápida y se suele medir por dos factores: la **velocidad de transferencia** (en bytes/segundo) y el **tiempo de acceso medio** (en milisegundos). La velocidad de transferencia mide la cantidad de información que pasa a la memoria (RAM) en un segundo y hoy día ronda los 10 MB/sg en los mejores casos. El tiempo de acceso medio es el tiempo que se tarda en acceder a un sector determinado, el cual depende de la velocidad con que se desplaza la cabeza a la pista y de la velocidad de rotación. También depende de la posición del sector a leer con respecto a la posición de la cabeza (pero no lo consideramos porque hablamos del tiempo medio no del tiempo absoluto). En un disco duro el parámetro más importante es el tiempo de acceso medio, ya que se tarda más en encontrar los datos que en transferirlos (en los disquetes es justamente al contrario). Actualmente los más rápidos son de menos de 9 ms. de tiempo de acceso, pero seguro que pronto bajará ese número. Eso quiere decir que puede que en algunas ocasiones tarde más de 9 ms. y en otras tarde menos, pero de media tardará unos 9 ms.

Igual que cualquier periférico, para que funcione un disco duro necesita un conjunto de chips que lo controlen, que suelen venir en una tarjeta controladora para ser insertada en un slot libre de la placa principal del ordenador. Hay muchos tipos de tarjetas controladoras, pero los interfaces más comunes son **IDE** (*Integrated Drive Electronics*) **EIDE** (*Enhanced* IDE)y **SCSI** (*Small Computer System Interface*). Existen tarjetas controladoras con memoria caché, que pueden hacer que los accesos al disco duro sean mucho más rápidos. Estas tarjetas disponen de BIOS propia y de zócalos (SIMM) para conectarles los módulos de memoria. En pruebas prácticas, se ha demostrado que con un caché de 2 MB el rendimiento es hasta 4 veces mejor que sin caché.

La técnica de caché llamada de buffers preventivos, almacena datos de los sectores que siguen a los datos pedidos, con la esperanza de que sean también solicitados a continuación (lecturas secuenciales), mejorando así considerablemente la velocidad de lectura del disco. Si la tarjeta es apta para ser insertada en un bus local, los accesos a disco se harán más rápidamente todavía.

Veamos cuatro casos prácticos y lo que sucede en cada uno de ellos:

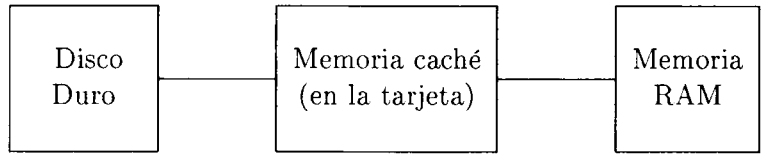


Figura 2.19: Transferencia disco-memoria con tarjeta caché (la transferencia es bidireccional).

1. Supongamos que tenemos una controladora normal (sin caché), instalada en un bus ISA: aquí, todo irá lento, tanto la comunicación disco-tarjeta como la comunicación tarjeta-memoria y las transferencias estarán limitadas por el dispositivo más lento (el disco o la controladora).

2. Supongamos ahora una controladora normal pero en un bus local: la comunicación con memoria es rápida, así que iremos a la máxima velocidad del disco duro.

3. Controladora caché en un bus ISA: sucede lo contrario, la comunicación disco-tarjeta va bien, estando el cuello de botella entre controladora y memoria.

4. Controladora caché y bus local: caso óptimo, ya que el caché de la tarjeta nos aisla de la lentitud del disco y el bus local se comunica rápidamente con la memoria.

Los discos duros son muy fiables, es decir, no suelen aparecer problemas en mucho tiempo (depende del uso, lógicamente). Su fiabilidad se mide por el tiempo medio entre fallos (**MTBF**, Mean Time Between Failures) que es más de veinte veces superior al de los disquetes. Suele ser superior a las 30.000 horas, pero lo mejor es hacer copias de seguridad de nuestro disco duro regularmente.

En un disco duro sólo se pueden almacenar ficheros creados por el S.O. que lo formateó. Si queremos usar varios S.O. con un mismo disco duro este se puede **particionar**, que consiste en partir el disco duro, de forma que cada partición es como si fuera un disco duro diferente. Nos referiremos a cada partición como si fueran discos distintos y podrán contener S.O. distintos. En principio, el usuario no puede saber si el disco está particionado o si son realmente varios discos (con la orden `fdisk` del MS-DOS se puede averiguar).

Hoy día, los discos duros con interfaz IDE o sus mejoras EIDE y Fast ATA se han quedado como los mejores para sistemas personales, ya que a mayor nivel existen otros sistemas mejores, como son **SCSI** y **SCSI-2**.

Los discos IDE no pueden ser de más de 528 MB (bajo DOS). Este límite ya lo superan los EIDE y el Fast ATA, pero para utilizarlos necesitará un PC con un BIOS (Basic Input/Output System) que admita una u otra.

En entornos **multiusuario** y **multitarea**, mejor interfaz que el IDE es el **SCSI**. Con SCSI se permite encadenar hasta 7 dispositivos de casi cualquier clase. IDE sólo permitía 2 y tenían que ser discos duros a la fuerza y EIDE permite 4 y pueden conectarse otros dispositivos (CD-ROM). Lo malo del SCSI es que es difícil de configurar, sobre todo si no cumplen los estándares ASPI (*Advanced* SCSI *Programming Interface*) o SCAM. Con el software ASPI es fácil configurar todo tipo de dispositivos SCSI (discos duros, CD-ROM...). Mejoras de este SCSI, son, de peor a mejor, SCSI-2 (5 Mbps), Fast SCSI-2 (10 Mbps) y Fast/Wide SCSI-2 (20 Mbps).

Para ahorrar energía muchos discos duros dejan de girar cuando no se están usando, algo fundamental para los portátiles y para cumplir los requisitos de bajo consumo de *Energy Star* de la EPA.

Antes de pasar al resumen de ventajas y desventajas de los discos duros, diremos que para ampliar la capacidad de un disco duro existen programas compresores/descompresores que comprimen la información al escribir en el disco y la descomprimen al leerla, de forma que los accesos son más lentos, pero se puede ampliar la capacidad del disco a casi el doble. También existen programas que simulan una caché de disco en memoria RAM, de forma que acelera los accesos a este dispositivo tan lento, en comparación con la RAM.

Ventajas:

- Acceso directo.
- Gran rapidez (y cada vez más), que depende de la velocidad de transferencia, del tiempo de acceso medio, de la controladora (si tiene caché, o si es para bus local)...
- Bastante capacidad (hasta 4GB).
- Pequeño tamaño (cada vez más).
- Muy fiables (MTBF).

Desventajas:

- Generalmente NO removibles.
- Caros (al menos más que los flexibles).
- Los removibles son más caros aún.
- Delicados.

2.4.1.2.5 Discos flexibles (Floppy Disk). También llamados disquetes y flopis (palabras aún no aceptadas por la Real Academia de la Lengua Española, que vienen de la españolización de las palabras inglesas diskettes y floppy disk respectivamente).

Son los típicos discos de los ordenadores personales usados para llevar información de un lado a otro (son removibles). Estos discos guardan la información igual que los discos duros, mediante pistas y sectores, y giran a menor velocidad (unas 300 rpm).

Los hay de diversos tamaños, y se miden por las pulgadas de su diámetro (1 pulgada = 2,54 cm.). Varían desde las 8 pulgadas (que ya no se usan), hasta los más usados, de 5 pulgadas y cuarto ($5_{1/4}$) y sobre todo los de 3 pulgadas y media de diámetro ($3_{1/2}$). Los más grandes están cubiertos por una funda de plástico blando con un agujero por donde lee la cabeza y hay que decir que ya están al borde de su desaparición. Los de $3_{1/2}$ pulgadas (ver figura 2.20) son más pequeños y están cubiertos de una funda de plástico duro con una puerta metálica que se corre hacia un lado para dejar al aire libre el disco, para que pueda ser leído.

Dentro de cada uno de estos dos tamaños más típicos hay dos tipos de discos, dependiendo de su densidad de grabación, pero en ambos se graba por las dos caras (double side):

- **DD** (Double Density, Doble Densidad).

Figura 2.20: Disco flexible de $3_{1/2}$ pulgadas y alta densidad, el cual tiene 1.44 MegaBytes en sus 2880 sectores.

- **HD** (High Density, Alta Densidad).

Los discos de HD usan mejor tecnología y admiten mayor número de pistas, por lo que tienen mayor capacidad. La capacidad de cada disco depende del sistema operativo que formatee el disco. En un sistema operativo tipo DOS (como MS-DOS), las capacidades se pueden ver en la tabla 2.3.

Los de $3_{1/2}$ disponen en su parte trasera de una pequeña compuerta que puede tapar o destapar un agujero. Si el agujero esta destapado indica que en el disco no se puede escribir, y la información que contenga sólo podrá ser leída. Si en el lado simétrico a ese agujero tiene otro agujero, el cual no se puede tapar, indica que el disco es de HD. Caso contrario será de DD.

Son mucho más lentos que los discos duros y, como dijimos al hablar de estos últimos, el parámetro más importante es la velocidad de transferencia, ya que al ser pequeños el tiempo de acceso medio es también pequeño.

Ventajas:

- Fácilmente transportables.
- Acceso Directo.
- Baratos.
- Muy extendidos (sobre todo $3_{1/2}$).

Desventajas:

- Escasísima capacidad.
- Demasiado lentos.

Tamaño en pulgadas	Densidad (Double o High)	Capacidad	Pistas/Cara	Sectores/Pista
$5_{1/4}$	DD	360 KB.	40	9
	HD	1,2 MB.	80	15
$3_{1/2}$	DD	720 KB.	80	9
	HD	1,4 MB.	80	18

Tabla 2.3: Capacidades de varios discos flexibles en MS-DOS.

- Delicados (se deben manejar con cuidado).

Después de todo lo visto anteriormente sobre los medios de almacenamiento de información más comunes podemos establecer un orden general en relación a su velocidad. De menor a mayor velocidad de acceso y de transferencia estarían: Cintas, Discos flexibles, Magneto-Ópticos, Discos duros, memoria ROM, memoria RAM, memoria caché y, por último, los registros de la CPU (los más rápidos y los más pequeños).

2.4.1.2.6 Discos Bernoulli. **Bernoulli**, un físico suizo del siglo XVIII, descubrió, con su teoría, lo que pasaría a ser el fundamento de la tecnología de almacenamiento removible de la compañía **Iomega**, en 1980. La principal ventaja respecto a los discos duros es la posibilidad de ser retirado como un disco flexible. Los primitivos discos Bernoulli resultaban frágiles, lentos y de escasa capacidad, por lo que su uso inicial no distaba mucho de la tarea de las unidades de cinta, es decir, la copia de seguridad.

Los Bernoulli cambiaron con los tiempos y actualmente son bastante rápidos (y siguen mejorando), y casi con tanta capacidad como un disco duro (y siguen aumentando). A veces incorporan memoria caché para aumentar su velocidad de acceso.

La teoría de Bernoulli es simple y dice que si tenemos un cuerpo sumergido en un fluido y se hace circular dicho fluido con mayor velocidad por una cara del cuerpo que por otra, éste experimenta una fuerza proporcional a la velocidad del fluido y hacia esta cara del cuerpo.

Puede parecer un poco lioso, pero no lo es. Veamos un ejemplo práctico: cojamos un folio de papel por sus dos esquinas superiores. Si nos acercamos el borde a la boca vemos que el folio se curva por la fuerza de la gravedad, pero si soplamos en su cara superior el folio tiende a subir. En este caso el fluido es el aire, que al circular con mayor velocidad por la cara superior del folio, éste tiende a ir hacia arriba.

Esta misma teoría se usa en la fabricación de aviones. El fluido vuelve a ser el aire y el objeto son las alas del avión, haciendo circular el aire más deprisa por la cara superior (por la forma del ala) hace que el avión se eleve más fácilmente.

Pero volvamos a los discos Bernoulli: estos están formados por dos discos flexibles separados por una placa. Tienen dos cabezales de lectura/escritura, uno para cada disco. Cuando el motor hace girar al disco se generan unas corrientes de aire internas, que hacen, por la teoría de Bernoulli, que cada disco abandone su estado normal, doblándose y elevándose, acercándose a la posición de la cabeza. O sea, que es el disco el que se acerca a la cabeza y no al revés.

Esto tiene una ventaja frente a los discos duros: en caso de producirse un corte eléctrico o cualquier golpe, el disco pierde velocidad de giro y abandona su posición, alejándose de la cabeza y evitando tanto que se dañe ésta como la superficie magnética del disco.

Ventajas:

- Acceso directo.
- Removibles (y bastante pequeños).
- Gran capacidad (más que los disquetes).
- Caso de corte eléctrico no sufren daños.

Desventajas:

- Demasiado caros (por ahora).
- Poco extendidos (por ahora).

2.4.1.2.7 Discos SyQuest. Son de parecido aspecto y tecnología que los Bernoulli. Contienen un disco de aluminio cubierto con una aleación metálica muy delicada. En el propio cartucho se incluyen las cabezas de lectura/escritura. Son menos fiables, más económicos y también removibles.

2.4.1.3 Soportes ÓPTICOS.

Los soportes ópticos llegaron de la mano del Compact Disk de música, cada vez más extendido. Tanto, que ya podemos decir (con más o menos pena para los bohemios) que los discos de música de vinilo han desaparecido.

Al igual que sus *hermanos musicales*, los soportes ópticos informáticos usan la tecnología del haz de luz o **rayo LASER**. Su éxito radica en ser de gran capacidad (más de 650 MB) y ser removibles, con lo que se han hecho el medio ideal para trasladar aplicaciones que requieran gran cantidad de *megas*. En síntesis, permiten almacenar más información en menos espacio que ningún otro medio.

La autoedición, el tratamiento de imágenes, el mundo MULTIMEDIA... son tareas todas ellas que demandan manejar gran cantidad de información con un tiempo de acceso reducido. Aunque por ahora los soportes ópticos son más lentos que los discos duros, se va ganando terreno en ese asunto.

Las aplicaciones **Multimedia** son, como su nombre indica, aquellos programas que usan *muchos medios*, es decir, programas que usan muchas imágenes (fotos o dibujos), animaciones, música y sonidos... En fin, aplicaciones estas que requieren mucho espacio para almacenar toda esa información. Para grabar y distribuir este tipo de programas se necesita un almacenamiento que sea grande y removible. Si se grabaran en disquetes se necesitarían un buen montón de ellos, lo cual sale más caro y es más problemático. La solución: un CD-ROM.

A la hora de adquirir uno de estos dispositivos debemos exigir que cumpla las especificaciones MPC (*Multimedia PC- marketing Council*), establecidas por un conjunto de fabricantes de hardware y software, que vienen a decir las características mínimas que debe tener cada dispositivo multimedia.

Dentro de la tecnología óptica, básicamente está el llamado CD-ROM, pero existen otros:

- CD-ROM (Compact Disk - Read Only Memory).
- CD-WORM (Write Once, Read Many).
- Tecnología de cambio de fase.

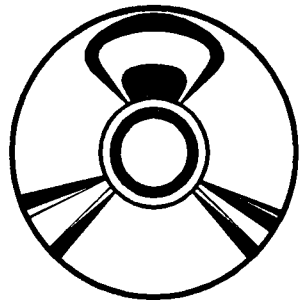


Figura 2.21: Disco de CD-ROM.

2.4.1.3.1 CD-ROM (Compact Disk - Read Only Memory). El CD-ROM es como un compact disk de música, pero donde se graban datos informáticos, no sólo música. La tecnología es idéntica, puesto que en los CD de audio, la música está grabada de forma digital, es decir codificada en ceros y unos (código binario). De hecho, la mayoría de los dispositivos lectores de CD-ROM pueden también leer un disco de música.

El formato básico, de 12 cm., fue definido por Phillips y Sony, pero existen otros formatos en CD-ROM:

- **CD-A:** 'A' de audio, es decir, los CD de música normales.
- **CD-ROM modo 1:** la especificación normal que veremos a continuación.
- **CD-XA (modo 2):** una especificación que permite grabar gráficos, textos y sonido en la misma pista, aumentando su capacidad.
- **PhotoCD:** un invento de Kodak para almacenar fotos en alta resolución. En un futuro cercano al revelar nuestras fotos nos darán un CD-ROM de este tipo de forma que las fotos las podremos ver en el monitor del ordenador o en la televisión. Será entonces fácil retocar fotos, cambiar colores... En un CD pueden caber más de 5000 fotos.
- **CD-I:** 'I' de Interactivo. Se pretende que sea un CD para manejo con la televisión (vídeos...). Aún no muy extendido.

Los CD-ROM, como los discos magnéticos, tienen pistas y sectores, pero de distinta forma. Un CD-ROM sólo tiene una pista de forma espiral de unos 34 Km. de largo. En dicha pista es donde se grabarán los datos (como los discos de vinilo) en bloques del mismo tamaño (los sectores).

La velocidad del CD-ROM varía según se encuentre el detector (el LASER) en la periferia o en el centro del disco.

La información se almacena gracias a un LASER de gran potencia que desgasta la superficie de aluminio de un disco virgen (nuevo) produciendo una serie de hendiduras que, posteriormente, serán interpretadas (leídas) por otro LASER de menor intensidad, mirando el reflejo de este haz de luz. El disco de aluminio está recubierto por una capa de plástico, que evita que se raye la superficie de aluminio.

Uno de los mayores problemas es que el polvo y la suciedad afecta negativamente al funcionamiento del LASER y del resto de lentes. Existen unas cajas de plástico (**caddy**) donde se

Carácterística		Medición
Velocidad de transferencia de datos	:	600-800 KBytes/seg. (Mínimo 4X velocidad)
Tiempo de acceso medio	:	300-100 mlseg.
Tamaño de buffer (o caché)	:	256-512 KBytes.
Interface Tarjeta controladora	:	SCSI ó SCSI-2.
Autolimpieza de lentes.		

Nota: Las especificaciones MPC, hoy día, son menos exigentes.

Tabla 2.4: Características *recomendables* para un lector de CD-ROM.

debe meter el aparato para evitar esto, pero aún así, no es posible evitarlo del todo metiéndolo en una caja hermética ya que si fuera así ¿Cómo metemos/sacamos los discos?.

Una vez grabado no es regrabable, y no los puede grabar un usuario normal, es decir, hay dispositivos que sirven como lectores de CD-ROM y otros como grabadores (estos últimos muy caros). Un usuario sólo los puede leer (de ahí su nombre de ROM), por lo que su uso principal es para distribución de todo tipo de software, ya que su gran capacidad evita tener que usar muchos disquetes. Pensemos, por ejemplo, que en sus más de 650 MB caben muchos disquetes, unas 150.000 páginas (una enciclopedia), más de 5000 imágenes...

Decir las características recomendables, hoy día, para un CD-ROM es muy arriesgado, pues en poco tiempo quedarán anticuadas, pero en la tabla 2.4 nos haremos una idea.

Hay otro tipo de interfaces para las tarjetas controladoras (IDE, AT o incluso por el puerto paralelo), pero esos dos son los mejores.

Existen además, lectores de doble (2X), triple (3X), cuádruple (4X), sextuple (6X), octuple (8X) velocidad, que son más veloces, en su velocidad de transferencia... y más caros. Los de simple velocidad van a 150 KBytes/segundo, que es lo necesario para oir un CD de música (CD-A).

Ventajas:

- Acceso directo.
- Gran capacidad.
- Baratos.
- Resistentes (longevidad).
- Cada vez más usados.

Desventajas:

- No reutilizables.
- No muy rápidos.
- El polvo/suciedad afecta a las lentes.

IBM ha anunciado que está investigando un CD-ROM multicapa que permitirá amontonar hasta 10 capas semitransparentes separadas por unos micrones de aire, cuyo resultado es un disco de poco más grosor que el convencional, que usará para su lectura un rayo láser convergente y un mecanismo que permita un enfoque altamente preciso. El CD-ROM de 2 niveles pronto se podría empezar a comercializar.

Además, también ya se habla del CD-ROM de alta densidad que podrá almacenar hasta 18 Gbytes en 1998.

2.4.1.3.2 CD-WORM (Write Once, Read Many). Es como un CD-ROM, pero permite grabar (escribir) una vez y leer todas las que queramos. Es un medio fiable, con las desventaja de que no podemos borrar o corregir algo que ya hayamos grabado y, lo peor de todo, es que es muy caro.

Ya hay unidades CD-WORM que pueden albergar hasta 6,56 GB, en 12 pulgadas, con 600 ms. de tiempo de acceso y 900 KB/sg. de velocidad de transferencia. Bueno, y existen otros que llegan hasta 187,2 GB.

2.4.1.3.3 Tecnología de Cambio de Fase (*Phase Change*). Es una novedosa tecnología de la empresa Matsushita, puramente óptica y de lectura-escritura.

Los discos, recubiertos de un material formado por una serie de cristales termosensibles de estructura regular, se calientan con un potente láser, consiguiendo que su estructura se desordene u ordene, pudiendo así grabar los ceros y unos. Si la estructura está desordenada, sigue siendo reflectante pero amorfo, por lo que al leerlo con un láser de menor potencia, la reflexión será distinta de la estructura ordenada. Pueden volver a escribirse hasta un millón de veces, siendo la velocidad de lectura igual que la de escritura.

Ventajas:

- Acceso directo.
- Gran capacidad (igual que un CD-ROM).
- Reutilizables.
- Gran seguridad.
- Su unidad lectora, también puede leer discos CD-ROM.

Desventajas:

- Tiempo de acceso lento.
- Velocidad de transferencia similar al CD-ROM.
- Poco extendidos.

2.4.1.4 Soportes HIBRIDOS (Magneto-Opticos, MO).

Consisten en una capa magnética sensible al calor, protegida por una cubierta de milímetro y medio. El proceso de escritura se produce en dos pasadas, una para borrar y otra para escribir. En la fase de borrado el LASER calienta a grandes temperaturas (unos 150°C, el

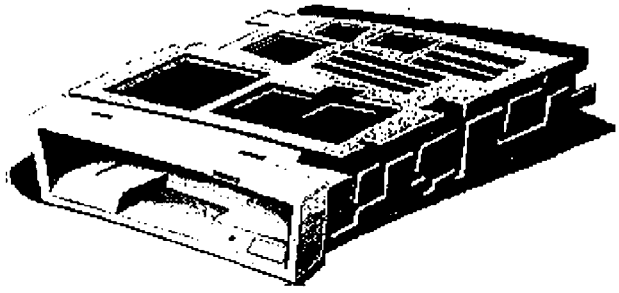


Figura 2.22: Unidad interna MO de Fujitsu.

Característica		Medición
Tamaño	:	3.5 pulgadas
Capacidad	:	230 Megabytes (MB)
Densidad de grabación	:	29.296 bits por pulgada (bpi)
		18.273 pistas por pulgada (tpi)
Velocidad de rotación	:	3.600 revol. por minuto (rpm)
Tiempo medio de acceso	:	30 milisegundos (mls)
Velocidad transferencia	:	2,1 MB por segundo (MB/sg)
Vida de los datos grabados	:	40 años a 25°C
Interface	:	SCSI-2

Tabla 2.5: Características de la unidad MO M2512A2 de Fujitsu.

punto Curie) la capa magnética para su polarización, al tiempo que una cabeza magnética orienta el medio caliente dándole valor nulo. Para la escritura se vuelve a polarizar el medio, mientras el cabezal magnético fija el sentido de orientación que va a representar el dato.

De esta forma, al suprimir el láser, la polaridad aplicada a éste queda fijada hasta que se repita el proceso, y los datos no pueden verse afectados por campos magnéticos externos, como ocurre con los almacenamientos magnéticos tradicionales.

Para la lectura no se usa el cabezal magnético sino que la polaridad de un punto se determina por el reflejo del rayo láser.

Existen sistemas que superan el gigabyte de capacidad, como el MiniDisc de Sony de 2,5 pulgadas que alcanza los 1,3 GB y se espera duplicar pronto esa capacidad. Como ejemplo, podemos comentar que la unidad MO M2512A2 de Fujitsu (figura 2.22) tiene las características que se detallan en la tabla 2.5.

Por sus características, podemos decir que el final de los discos flexibles está cerca.

Ventajas:

- Rápidos (casi como un disco duro).
- Muy fiables (No les afecta campos magnéticos externos ni golpes).
- Gran capacidad.
- Portables y de pequeño tamaño.

- De lectura y escritura.
- La relación Precio/MB sale muy ventajosa.

Desventajas: Sólo una:

- No muy extendidos (por ahora).

Existe otro tipo de soporte híbrido, llamado **flóptical**, el cual usa la tecnología magnética para almacenar la información y aprovecha la precisión de la tecnología óptica para posicionar la cabeza en la pista adecuada, consiguiendo así que quepan más pistas (y por tanto más información) y también que se reduzcan mucho los tiempos de acceso. Junto a cada pista magnética debe haber una pista que puede ser leída por un diodo LASER.

Los flópticas son una alternativa a los disquetes tradicionales, ya que son pequeños (3,5 pulgadas), mucho más capaces (más de 20 ó 30 MB) y hasta más de 3 veces más rápidos.

2.4.2 Periféricos de SALIDA.

A estas alturas, ya debe ser obvio que un periférico de salida es un aparato o dispositivo que sirve para **sacar** información del ordenador, o para que este muestre sus resultados. Aquí veremos los más importantes, sin olvidar que alguno de ellos puede ser que, además, también sea un periférico de entrada.

Los que veremos a continuación son:

- Perforadora de fichas y cinta de papel.
- Pantalla (monitor) y tarjetas gráficas.
- Impresoras:
  - De impacto:
    - De tipos.
    - Matriciales.
  - No de impacto:
    - Inyección (chorro) de tinta.
    - Térmicas.
    - Láser.
- Plotter.
- Conversor Digital/Analógico (sintetizador de voz).
- Displays.

2.4.2.1 Perforadora de fichas y cinta de papel.

Periféricos ya muy obsoletos. Consistían en una serie de punzones que se lanzaban por impulsos electromagnéticos y que perforaban la ficha o cinta, en aquellas posiciones que correspondiera al carácter a grabar y al código usado.

No diremos más de estos periféricos, ya que pasaron a la historia informática de los años 70 como uno de los primeros periféricos.

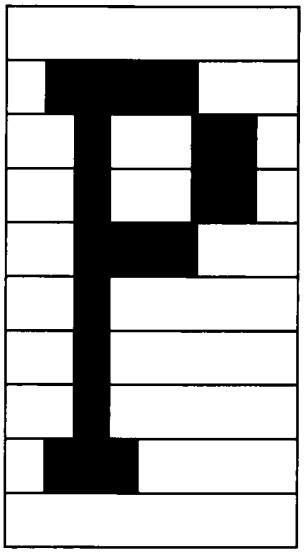


Figura 2.23: Matriz de píxeles del carácter "P".

2.4.2.2 Pantalla (monitor) y tarjetas gráficas.

Nos encontramos ante el dispositivo de salida más típico y más usado. La pantalla (display, screen, CRT o simplemente monitor), es, normalmente, usado como **salida estándar** en un sistema. En el apartado sobre el teclado (ambos son llamados consola), aclararemos qué se entiende por entrada/salida estándar.

En fin, es similar a una televisión, con su tubo de rayos catódicos (CRT, Cathode Ray Tube), el cual hace incidir un haz de electrones en la superficie interna de la pantalla, recubierta de un material fosforescente. Cada posición donde pueden incidir los electrones se corresponde con un punto o **pixel** (picture element) de la pantalla. La imagen se forma por multitud de estos píxeles, encendidos o apagados, y con diferentes colores (si es un monitor color).

Se puede usar un aparato de TV normal como monitor de un ordenador, pero hace falta un dispositivo adaptador especial (modulador RF).

El número de píxeles (puntos) de la pantalla (ancho × alto) se llama **resolución**, y lógicamente cuanto mayor resolución tenga la pantalla, más píxeles tendrá y se podrán representar gráficos y dibujos con mucha mayor exactitud. Según esto (y otras características) hay distintos tipos de pantallas (y controladoras gráficas) como son, por ejemplo MDA (Monochrome Display Adapter), CGA (Color Graphics Adapter), EGA (Enhanced Graphics Adapter), TIGA (Texas Instruments Graphics Arquitecture)... pero los más recomendables hoy día es la VGA (Video Graphics Array) y mejor aún, la SuperVGA (SVGA). Otra muy moderna, pero menos extendida, es la XGA (eXtended Graphics Array).

De esta forma, la pantalla se forma por una matriz (array) de puntos, y los caracteres y gráficos se forman encendiendo o apagando, con distintos colores, estos píxeles (ver figura 2.23).

Como todo periférico, necesita unos circuitos controladores, que en este caso suelen venir en una tarjeta aparte, denominada **tarjeta gráfica** o controladora gráfica. Esta controladora

Controladora	Resolución	Colores
CGA	320x200 ó 640x200	4 ó 2 resp.
Hércules monoc.	720x348	2 (blanco y negro)
VGA	hasta 640x480	256
SVGA	hasta 1024x768	256
XGA	1024x768	256
	ó 640x480	65.536

Tabla 2.6: Algunos tipos de controladoras gráficas.

debe ser de igual tipo que el monitor, para que éstos se entiendan, y la controladora pueda enviar y tratar una menor o igual resolución que la máxima aceptada por el monitor. En la tabla 2.6 se pueden ver las características de algunos de estos tipos, pero hay muchos más.

Para el control de la imagen que se visualiza en el monitor se tiene la llamada **memoria de vídeo** (o buffer de vídeo), que almacena lo que contiene la pantalla. El monitor es un periférico de los llamados *memory mapped* (mapeados por memoria). Esto quiere decir que no se accede a la pantalla mediante instrucciones de salida normales, sino que el procesador (manejado por el S.O.) escribe en una zona determinada de memoria, y el controlador lee esa zona y envía las señales correspondientes a la pantalla.

El término **VRAM** (Video RAM) implica un tipo especial de memoria RAM de doble puerto o de doble acceso (en contraposición de la DRAM que tiene un solo puerto). O sea, la VRAM permite que el procesador escriba por un puerto y la controladora lea por otro puerto, ambos accediendo a la vez a la memoria de vídeo.

Tengamos en cuenta que el monitor muestra entre 50 y 100 imágenes por segundo (depende del monitor y de la tarjeta). Así, si la memoria de la tarjeta no es VRAM, la CPU tiene que esperar a que la tarjeta (el DAC, *Digital to Analog Converter*) termine de leer para poder escribir en esa memoria y modificar así la imagen. Aquí, la CPU tiene menos prioridad que el DAC, para que la imagen no parpadee.

La imagen, para ser visualizada durante un tiempo, debe ser repetida o **refrescada** periódicamente al menos 25 veces por segundo (en las pantallas de barrido o raster scan), por lo que es muy importante que la controladora pueda acceder a la memoria de vídeo siempre que lo necesite (de ahí la utilidad de la VRAM).

La cantidad de memoria de vídeo necesaria para una placa de gráficos depende de la resolución y del número de colores. Para determinar la cantidad de memoria de vídeo requerida, hay que multiplicar el número de bytes por pixel (según número de colores) por el número total de pixels de resolución. Así, por ejemplo, si queremos una resolución de 1280x1024 en *true color* (color real), requeriría 3 bytes/pixel multiplicado por 1280x1024, es decir, 3.932.160 bytes, que son unos 4 MB (redondeando por exceso).

La imagen de la pantalla se forma, como hemos dicho, por multitud de píxeles, ordenados en filas (horizontales) y columnas (verticales). Para **refrescar** la imagen, se debe recorrer la memoria de vídeo e ir pintando, cada pixel, de su color correspondiente. Este recorrido, pixel a pixel, se hace siguiendo un orden determinado, que depende del tipo de monitor (normalmente línea a línea y de arriba a abajo).

Hay unos monitores llamados **sensibles al tacto**, o **pantallas táctiles**, que permiten introducir información (posiciones de la pantalla, para elección de opciones...), tocando una zona de la pantalla. Hay varias técnicas. Una de ellas trabaja con una rejilla de haces infrarro-

jos emitidos por delante de la pantalla, de tal forma que cuando los haces son interrumpidos en un punto cualquiera, genera dos valores, la posición horizontal y la vertical, que se introduce en el ordenador. En otros modelos, la rejilla es de hilos conductores, embebida o incrustada en la pantalla, de forma que al apretar en la pantalla se produce un cortocircuito que hace introducir la posición del cortocircuito.

Veamos a continuación las características más destacadas de un monitor:

- **Resolución:** número de píxeles máximo que puede visualizar (columnas x filas). Depende, lógicamente del tamaño de la pantalla del monitor.

- **Tamaño:** se refiere al tamaño total de la pantalla del monitor. Igual que las televisiones, se mide por las pulgadas de la diagonal, y lo más normal es que oscile entre 14 y 19 pulgadas (1 pulgada = 2,54 cm). Los monitores de 20 ó 21 pulgadas están reservados a profesionales de la imagen y los gráficos.

- **Frecuencia de barrido:** específica para cada controladora y cada monitor, por lo que ambas deben coincidir. Esta frecuencia se divide en frecuencia de barrido vertical (frecuencia de renovación de la pantalla total), y horizontal (frecuencia de refresco de cada línea). La horizontal se mide en KiloHerzios (Khz.), pero la más importante es la vertical, que se mide en Herzios y en los mejores monitores suele oscilar, entre los 80-90 Hz., esto es, refresca la imagen 80 ó 90 veces por segundo. A esta última se le suele llamar **frecuencia de refresco**, e indica, como hemos visto, la frecuencia con que se recompone la imagen en número de veces por segundo. A mayor frecuencia, se consigue que el monitor parpadee menos y por tanto, canse menos la vista, pero los componentes son más caros.

  El consorcio VESA descubrió que la velocidad de refresco de algunos focos de luz (fluorescentes) era de 50 Hz, que si coincidía con la frecuencia de refresco del monitor generaba un efecto estroboscópico incómodo para el usuario, por lo que recomienda frecuencias superiores a los 70 Hz.

  Hay monitores que admiten dos frecuencias (CGA y hércules) y son llamados Bisync, otras que son Trisync y los más modernos son los **Multisync** (o multiscanning), que admiten un amplio rango de frecuencias, y por tanto pueden trabajar con muchas tarjetas gráficas.

- **Entrelazado/NO entrelazado:** como ya hemos visto, la imagen se dibuja siguiendo un barrido, pintando pixel por pixel, todos los de la pantalla y volviendo a empezar. Este barrido se suele hacer por filas (llamadas raster), empezando por la fila de píxeles superior de la pantalla y recorriendo fila a fila, de izquierda a derecha, hasta la más inferior.

  Para poder usar una frecuencia menor se inventaron los monitores entrelazados (interlace), que refrescaban una fila sí y otra no, y a la segunda vuelta refrescaban las que no se habían refrescado en la vez anterior y luego vuelve a empezar. Esto funcionaba, pero se creaban unos parpadeos en la imagen que inducía mucho cansancio de la vista. Por tanto, lo mejor es que sea NO entrelazado.

- **Color o monocromo:** ya casi han desaparecido los monitores monocromos y son poco recomendables, ya que hay aplicaciones que requieren el color. En los monitores color, el

haz de electrones se divide en tres haces separados, controlando, los tres colores primarios de vídeo: rojo, verde y azul.

- **Tecnología:** la más usada es la de barrido (CRT) que hemos detallado más arriba, pero hay más. Las pantallas de cristal líquido (**LCD**, Liquid Crystal Display) o de cuarzo líquido, son muy usadas en portátiles, por su pequeño grosor. Ultimamente para portátiles se usan las pantallas de matriz activa TFT (*Thin-Film Transistor*, o transistor de película fina) que incorpora un transistor detrás de cada pixel (3 si la pantalla es color) y que incrementa la legibilidad de los gráficos y su uso resulta más agradable.

- **Separación entre píxeles (*dot pitch*):** esta separación es muy importante para que la imagen de una sensación de continuidad, y no se distingan los píxeles independientes. Se mide en milímetros y puede variar desde 0,26 hasta más de 0,31 mm., pero lo normal es 0,28 mm., aunque si se tiene un monitor con menor separación, se conseguirá mayor claridad en la imagen.

- **Mandos de control:** un buen monitor debe tener como mínimo los siguientes mandos de control:

  - Encendido/Apagado (On/Off).
  - Brillo.
  - Contraste.
  - Posición de la imagen (horizontal y vertical).
  - Tamaño de la imagen (Control de la anchura y la altura).

- **Capa antiestática:** cuando los electrones rebotan en el interior de la pantalla, se crea una carga electrostática y como consecuencia de ello se acumula polvo en su superficie, lo que reduce la calidad de la imagen. Esta capa reduce esa carga y el riesgo de descarga si se toca la pantalla.

- **Pantalla plana:** los monitores de pantalla plana producen menos distorsiones y menos reflejos que cansan la vista.

- **Ergonomía:** que sean monitores que no produzcan cansancio en el usuario. Por eso es recomendable un monitor de baja radiación, es decir, que no emita mucha radiación de alta frecuencia que cansan la vista. Es recomendable que cumpla el estándar sueco MPR-II, y en todo caso, también es recomendable usar un filtro, que es un cristal (polarizado) que se pone delante de la pantalla y que recoge las radiaciones y se las lleva por un cable de toma tierra que debe incorporar. Además, un filtro también evita reflejos que cansan mucho la vista. También es aconsejable que el monitor se pueda girar hacia los lados, así como inclinar hacia arriba o hacia abajo.

- **Cumplimiento de Energy Star:** el estándar de consumo de energía de la Agencia de Protección Ambiental (EPA) de EEUU exige que los monitores de PC no gasten más de 30 watios en modo espera, de modo, que los monitores deben desactivarse automáticamente cuando cesa la actividad en pantalla durante cierto tiempo.

n bits	colores	calidad
1	2	baja, monocromo
2	4	baja calidad, low color
4	16	baja calidad, low color
8	256	falso color, pseudo color
12	4.096	falso color, pseudo color
15	32.768	alto color, high color
16	65.536	alto color, high color
24	16.777.216	color auténtico, true color

Tabla 2.7: Número de colores según los bits empleados en la tarjeta gráfica.

- **Precio:** depende de los factores arriba mencionados y muchos más. Lo mejor, como siempre, es preguntar a muchos distribuidores, comparar la relación precio/prestaciones y luego elegir conforme al presupuesto, que suele ser el que decide.

Veamos ahora las **características** más importantes de una **tarjeta** o controladora, **gráfica** o de vídeo, amiga inseparable de todo monitor:

- **Resolución:** número de píxeles máximo que puede visualizar (columnas x filas). Normalmente podrán controlar varias resoluciones. Debe depender, lógicamente de la resolución del monitor. En este punto incluimos los estándares de los que ya hemos hablado, principalmente SVGA (es la que recomendamos, por ahora).

  Preferiblemente debe cumplir la normativa VESA. No nos referimos aquí a la de bus local, que también es recomendable, sino a unas especificaciones que permiten garantizar la compatibilidad en alta resolución con todo tipo de programas.

- **Número de colores:** los colores que una tarjeta puede mostrar simultáneamente se llama *paleta*. El color de cada pixel se guarda en forma de bits. Cuantos más bits tenga reservados para el color, más colores podrá representar. Así, si tiene **n** bits para el color, podrá tratar $\mathbf{2^n}$ colores. De esta forma, las tarjetas más comunes suelen usar los valores de la tabla 2.7.

  Las tarjetas, en realidad, almacenan valores de combinaciones de rojo, verde y azul (RGB, Red, Green, Blue). Entonces, por ejemplo, la tarjeta de 15 bits usa el formato 5:5:5, 5 bits para cada color, la de 16 bits usa el formato 5:6:5, 5 bits para el rojo, 6 para el verde y otros 5 bits para el azul, o también 6:6:4. Para los 24 bits se usa el formato 8:8:8.

  El elemento encargado de pasar el contenido de color de cada pixel (los n bits), a señales eléctricas para representarlo en el monitor, se llama LUT-DAC (Look Up Table - Digital Analogic Converter). El LUT (o a veces CLUT, de Color Look Up Table) se encarga de elegir un color de la tabla de colores disponible y el DAC se encarga de transformar ese valor digital (bits) a valores analógicos que pueda entender el monitor.

  Para trabajos de usuario normal (bases de datos, tratamiento de textos, hoja de cálculo...), con 256 colores se funciona bien, y más rápido. Si lo que prima son los gráficos

entonces será conveniente usar más colores, y si es para profesionales de la imagen, televisión, vídeo o fotografía es imprescindible el **true color**. Aunque este último sistema va ganando adeptos fuera de los profesionales y cada vez es más rápido.

- **Velocidad:** en casi cualquier programa, el sistema gráfico consume un mínimo del 10% del total del tiempo. Si trabajamos en un entorno gráfico, el procentaje se sube al 25% cómo mínimo (dependiendo de la resolución y el número de colores). Si encima nuestro programa usa mucho la representación gráfica en pantalla (que suele ser muy habitual), el porcentaje aumenta más aún.

  En modo texto, una pantalla puede ocupar unos 4 KB, pero en entorno gráfico (como OS/2 o Windows) y según la resolución, puede llegar a ocupar más de 250 KB, lo cual redunda en la velocidad de transferencia de la imagen de la pantalla.

  Por eso, el sistema gráfico es uno de los más importantes y de los que pueden llegar a ralentizar mucho un sistema. De casi nada nos sirve un potente procesador a muchos Mhz., si queremos trabajar en entornos gráficos y el sistema gráfico es lento. Por eso, los modernos ordenadores incluyen por defecto una tarjeta gráfica insertada en un bus local. Puesto que la salida gráfica es uno de los sistemas más usados, pongámoslo en el lugar más rápido.

  Otra opción para acelerar el sistema gráfico son las llamadas **tarjetas aceleradoras** que incorporan o un **chip acelerador** (no programable), o un **procesador propio** (más caro, pero programable), que descargan de trabajo al procesador principal, mejorando no sólo el sistema gráfico, sino el resto de operaciones del procesador.

  Como ejemplo, podemos asegurar que un 486DX/33 es más lento que un 386SX/20 con tarjeta aceleradora, trabajando ambos en entornos gráficos, con muchas salidas a vídeo.

- **Memoria:** la cantidad de memoria es fundamental para el manejo de grandes resoluciones, con muchos colores y si nos interesa ganar en velocidad. Lo que hoy día recomendamos es 1 ó 2 MB. Algunas tarjetas son ampliables, pero los chips de VRAM deben ser iguales y cuanto más rápidos mejor. No olvide que esta memoria no cuenta como memoria del sistema. Es sólo para vídeo y no vale para otra cosa.

  A veces se incluye memoria DRAM (monopuerto), más barata que la VRAM (de doble puerto), lo cual hace que haya conflictos a la hora de acceder a leer/escribir en esa memoria de vídeo. Para evitar (en parte) estos conflictos, a veces se separan las direcciones (posiciones) pares de la memoria de las impares, minimizando así la posibilidad de conflictos, con el objetivo de equiparar rendimientos con los de la VRAM.

- **Drivers:** elemento software (programas) que saca el máximo rendimiento a la tarjeta y que deben ir incluidos con la propia tarjeta en uno o varios disquettes. A veces también se incluyen en el Sistema Operativo. Quizás, consiguiendo la última versión de un driver (que suele ser gratuíta, en algún BBS, Compuserve...) se puede mejorar el rendimiento de nuestro sistema gráfico.

- **Precio:** obviamente, va íntimamente ligado al resto de características ya comentadas y a otras más o menos técnicas (marca...). Pero es muy importante no olvidar que, **no siempre lo más caro es lo mejor**, en este, y en general, en todos los productos.

Por último, decir que para conseguir un ordenador multimedia, para trabajar con este tipo de aplicaciones, necesitamos que tanto el monitor, como la tarjeta, sean de buena calidad.

2.4.2.3 Impresoras.

Un periférico de salida también muy típico. Se usan para imprimir información (datos, texto, dibujos...) sobre un papel (*hard copy*).

Hay impresoras de multitud de formas tipos y tecnologías, que pueden usar papel normal (en folios), papel continuo o que usan un papel especial... Aquí daremos una clasificación por su tecnología y sistema de impresión. Pero independientemente de esto, todas crean una imagen en un papel a base de pintar puntos. Cuanto menor sea el tamaño del punto, la calidad del dibujo (o texto) será mejor.

Debe existir un esquema común para determinar dónde colocar los puntos en cada letra o dibujo. Los esquemas más comunes son las fuentes bitmap y outline. Las fuentes bitmap poseen tamaño y anchura predefinida y las outline pueden ser escaladas (variar de tamaño) y asignarle distintos atributos (negrita...).

La impresión normalmente se hace en negro, pero existen impresoras que imprimen en color. Normalmente lo hacen combinando tres colores: cyan (azul), magenta (rojo) y amarillo. El negro se puede conseguir mezclando los 3 colores a partes iguales, pero a veces se incluye en cartucho aparte, por ser un color muy usado. Para ahorrar tinta, una impresora debe tener un sistema para imprimir en modo borrador (modo económico y ecológico).

Según su tecnología de impresión, se pueden clasificar en dos tipos:

2.4.2.3.1 Impresoras De Impacto: la impresión se realiza al poner delante del papel una cinta entintada y golpear sobre esta lo que se desee imprimir. Es similar a una máquina de escribir convencional. Debido a este impacto son bastante ruidosas. Podemos a su vez dividirlas en dos tipos:

- **De tipos:** son ya muy antiguas. Su técnica es tener un juego (conjunto) de caracteres (tipos) fijo, con el que golpear la cinta entintada que hay delante del papel, imprimiendo así el carácter deseado. Eran de gran calidad pero muy lentas, y sólo permitían imprimir caracteres muy limitados, sin posibilidad de imprimir gráficos.

  De este tipo son las de cilindro, bola, barra, cadena... pero la más famosa fue la impresora de margarita, en la que el juego de caracteres se instalaba en forma de flor de margarita en la que en cada pétalo estaba grabado un carácter. Primero giraba la margarita hasta posicionar el pétalo del carácter deseado en la posición correcta, y luego un pequeño martillo golpeaba el pétalo por detrás, imprimiendo dicho carácter. Se podía cambiar el juego de caracteres, cambiando la margarita.

- **Matricial (o de agujas):** han sido hasta ahora las más populares debido a su bajo precio, pero ahora pierden terreno por lo ruidosas que son y por su lentitud, en favor de las de inyección que ya están a precios muy asequibles.

  Los caracteres se forman con estructuras de puntos individuales que efectúan un impacto en el papel. Cada carácter tiene una estructura matricial de puntos que deben pintarse (bitmap).

  La cabeza escritora de la impresora consta de una hilera vertical de agujas. Cada aguja equivale a un pixel o punto de una fila de la matriz de cada carácter, y todas las agujas forman una columna de dicha matriz. La cabeza va pasando a lo largo de una línea y las agujas se van lanzando, imprimiendo columna a columna todos los distintos caracteres de esa línea. Luego, sube el folio, pasando a la línea inferior y continúa igual. La mayoría

de este tipo de impresoras imprimen en ambos sentidos, es decir, la cabeza escritora imprime de izquierda a derecha una línea, y la siguiente de derecha a izquierda, ganando así velocidad.

No suelen ser de gran calidad, pero permiten imprimir todo tipo de gráficos. Su calidad depende del número de agujas (normalmente 9 ó 24) y su velocidad oscila, en el mejor caso, los 200 cps (caracteres por segundo).

Además, a la hora de evaluar una impresora hay que tener en cuenta el costo de mantenimiento (costo/página), el cual es mínimo en estas impresoras que sólo necesitan una cinta entintada similar al de una máquina de escribir. Al ser de impacto se pueden sacar varias copias a la vez usando papel calco, reduciendo así los costes y el tiempo de impresión.

2.4.2.3.2 Impresoras No De Impacto: usan otras técnicas más silenciosas, que las dividen en:

- **Térmicas:** similares a las matriciales, pero más silenciosas. Utilizan un papel especial termosensible que se ennegrece al aplicar calor (unos 200° C). El calor se transfiere desde el cabezal por una matriz de pequeñas resistencias. Al pasar una corriente eléctrica por las resistencias, se calientan, formando los puntos en el papel.

  Suelen ser caras y de gran calidad, aunque no muy extendidas debido a que tienen que usar un papel especial. Esta técnica es muy usada en los fax.

- **Inyección (o chorro) de tinta:** son impresoras de gran calidad, silenciosas y bastante baratas, por lo que se están extendiendo cada vez más, como impresoras domésticas. Además, hay modelos que imprimen en varios colores. Lo malo es el precio de mantenimiento, ya que usa cartuchos de tinta bastante caros. De todas formas, existen empresas que rellenan de tinta los cartuchos vacíos, consiguiendo un ahorro considerable.

  Su sistema de impresión es, al menos, bastante original. El cartucho de tinta se inserta en la cabeza de impresión, que tiene unos minúsculos agujeros, por los que pasará la tinta. La tinta es calentada (a más de 450° C durante millonésimas de segundo) de forma que hierve formando una burbuja de vapor que crece hasta que empuja a la tinta a salir por el agujero, en forma de gotita (de unos 0,05 mm.).

  Este chorro de gotas de tinta están ionizadas, de forma que en su recorrido son desviadas por unos electrodos. El carácter se forma con la tinta que incide sobre el papel. Usualmente un carácter es formado por una matriz de unas 20×20 gotitas. Cuando no se debe escribir, las gotas de tinta se desvían hacia un depósito de retorno.

  Tomando la tinta de distintos cartuchos podemos obtener impresoras que impriman en colores.

  Su resolución puede variar, estando las resoluciones típicas entre 360×360 y 720×720. Además, la calidad de impresión de una foto en color depende del tipo de papel usado. Hay un papel especial que mejora este tipo de impresión.

- **Impresoras LASER:** son de elevada velocidad y calidad. Su precio ha hecho que no se extiendan mucho y están restringidas casi exclusivamente a empresas. Además, el mantenimiento es caro, pues requieren cartuchos de tóner (tinta en polvo) que son bastante caros.

Son de parecido funcionamiento que las máquinas fotocopiadoras. Lo más importante es su resolución, que puede variar entre un mínimo de 600 ppp (puntos por pulgada) y más de 1200 ppp.

Para imprimir, estas impresoras deben controlar cinco operaciones a la vez:

1. Interpretar las señales del ordenador.
2. Traducir esas señales a movimientos del LASER.
3. Controlar movimiento del papel.
4. Sensibilizar el papel para que acepte el tóner.
5. Imprimir en el papel.

Veamos más detalles. Inicialmente recibe la información del ordenador y estas son traducidas en apagar y encender convenientemente el haz de luz (LASER). Un pequeño espejo giratorio desvía la luz, y la hace incidir en un cilindro recubierto de material fotosensible (tambor). Los movimientos del espejo y el giro del tambor hacen que el LASER pueda llegar a cualquier parte de la superficie del tambor. O sea, una vez que el laser ha compuesto una línea de puntos (al incidir en el tambor), el tambor gira (unos 1/300 de pulgada, o menos) y pasa a la siguiente línea de puntos, donde por el mismo procedimiento compondrá de nuevo sus puntos.

Los puntos del tambor en los que dió la luz, quedan cargados de carga eléctrica (negativa) y corresponderán a puntos que luego se imprimirán en el papel. Las zonas donde no dió la luz quedarán como zonas sin imprimir (en blanco) en el papel.

Una vez compuesta la imagen, el tambor pasa por un depósito de tinta en polvo (**tóner**), cargado con carga opuesta a la del tambor (carga positiva), de forma que el tóner queda adherido a las zonas del tambor donde incidió la luz. Acto seguido pasa el papel por el tambor, que previamente se sensibilizó también eléctricamente, de forma que atraiga al tóner del tambor con más fuerza que éste, quedando así impreso el folio.

El tambor será limpiado eléctricamente para la siguiente hoja y el folio pasará por un sistema de fusionado, con el que por presión y calor unen el tóner permanentemente al papel.

El sistema descrito aquí es llamado de escritura negra (modo de impresión Canon). Existe también el sistema de escritura blanca (Ricoh), en el que las zonas donde incide el LASER en el tambor se cargan de igual carga que el tóner, por lo que el tóner se adhiere a las zonas donde no incidió la luz.

Tradicionalmente, a las impresoras láser se las ha calificado de antiecológicas, por tener elementos consumibles no reciclables (tambor, tóner, kit de revelado...). Además, suelen producir ozono (O_3), un gas que respirado continuadamente produce dolores de cabeza. Sin embargo, ya hay impresoras láser ecológicas, que reducen la emisión de ozono y tienen materiales reciclables o duraderos, como por ejemplo, tener un tambor de silicio, tan duro que sirve para toda la vida de la impresora.

2.4.2.4 Plotter.

El plotter, registrador o trazador gráfico es un periférico de uso principalmente industrial, ideado para realizar dibujos sobre papel, en varios colores y con una resolución y precisión

perfecta (planos de edificaciones, circuitos electrónicos, diseño de motores...).

Para dibujar usan distintas plumillas (a veces con distintos colores), que se desplazan por el papel con gran precisión y le permiten hacer curvas y rectas perfectas (continuas, no digitales).

Según su tamaño los hay de dos tipos:

- **De sobremesa:** constan de una tabla de tamaño fijo, donde se sitúa el papel. Por encima del papel se desplaza una barra (en el eje X) y a lo largo de la barra se desplaza una plumilla (en el eje Y).

- **De pie:** son de gran tamaño (y gran precio). Normalmente constan de un rodillo que hace mover al papel (en el eje X) longitudinalmente. El ancho del rodillo limita el ancho del papel, pero no tiene limitación de largo. Encima del rodillo se sitúa la plumilla que se desplaza a lo largo de este (eje Y). A veces en vez de una plumilla se usa una cabeza de inyección de tinta (como la de las impresoras).

2.4.2.5 Conversor Digital/Analógico (sintetizador de voz).

Existen computadoras cuya salida debe actuar sobre un dispositivo que es controlado por una señal eléctrica analógica. Esta señal actúa sobre otro aparato que transformará la señal eléctrica en otra de distinta naturaleza, como son, por ejemplo, movimiento de motores, emisión de sonidos, apertura de válvulas de conducción de fluidos...

La conversión del dato binario (digital) generado por el ordenador en una señal continua (analógica) se efectúa en un circuito denominado conversor D/A (Digital/Analógico).

Una de las aplicaciones quizás más sorprendentes son los sintetizadores de voz, que permiten que el ordenador hable. Usan un conversor Digital/Analógico, amplificador, altavoz... y sobre todo un software que controle todo. Permiten, por ejemplo, leer un fichero de texto.

A veces las palabras (fonemas e incluso frases) se tienen ya grabados y no hay que generarlos (con el conversor D/A), simplemente mandar la grabación a la salida del altavoz.

2.4.2.6 Displays.

Los Visualizadores o displays, son periféricos bastante poco usados y casi exclusivamente usados para aplicaciones específicas. Son pequeñas unidades de salida que permiten leer un dato. Los más típicos son los de 7 segmentos (como los usados en los relojes digitales o calculadoras) que sólo sirven para números y algunas letras.

Hay otros para letras, o con muchos LEDs (Diodos Emisores de Luz) o bombillas de colores que permiten poner mensajes o dibujos y que éstos se muevan (suban, bajen...), ideales para reclamos publicitarios.

2.4.3 Periféricos de ENTRADA/SALIDA.

Hay periféricos, que por su naturaleza, son dispositivos que permiten tanto la entrada como la salida de datos. Entre ellos, se encuentra la pantalla sensible al tacto, ya explicada anteriormente. Ahora veremos algunos otros dispositivos que se pueden englobar en esta categoría:

- Tarjetas de Sonido. Altavoces y micrófono.

- MODEM/Fax.

Figura 2.24: Con una tarjeta de sonido...

- Tarjetas de Red.
- Tarjeta de Vídeo y Sistema de Videoconferencia.
- Dispositivos de Realidad Virtual.

2.4.3.1 Tarjetas de Sonido. Altavoces y micrófono.

Son usados para salida/entrada de sonidos y **música**, de cierta complejidad, para los que no basta el pequeño altavoz incorporado en casi cualquier microordenador (ver figura 2.24). Ultimamente se usan mucho en aplicaciones Multimedia.

El tratamiento digital del sonido empezó con los Compact Disk (CD, o CD-A), pero no ha parado. Para digitalizar un sonido o una canción, hay que muestrear el valor del sonido, es decir, almacenar el valor del sonido en cada momento. La calidad del sonido almacenado (sonido digital) depende de la velocidad de muestreo, o sea, la velocidad entre distintas muestras del sonido, y del tamaño que podemos almacenar de cada muestra, su precisión (a mayor precisión, la muestra será mayor, ocupará más bits).

Sus principales utilidades son:

- Digitalizar sonidos (periférico de entrada) y almacenarlos en disco, para retocarlos o reproducirlos más tarde.
- La reproducción de sonidos (periférico de salida) previamente digitalizados.
- Compatibilizar un ordenador con los instrumentos musicales electrónicos a través del interfaz MIDI.

Para digitalizar un sonido lo que se hace es tomar una muestra de la *canción* cada cierto tiempo y almacenar ese valor, codificado en binario.

Es obvio, entonces, que lo mejor es que se muestree a mucha velocidad (gran frecuencia de muestreo) y de cada muestra se guarde mucha información (muchos bits).

La frecuencia de muestreo se mide en KiloHertz (o KiloHertzios, Khz.), y cada Khz. representa 1000 muestras por segundo.

Imaginemos una tarjeta de 8 bits y otra de 16. En la primera sólo podemos guardar un byte para cada muestra realizada, o sea, sólo podemos almacenar 256 (2^8) niveles de sonido distintos. En cambio, con la tarjeta de 16 bits, podremos guardar hasta 65.536 (2^{16}) niveles. La diferencia salta al oído.

Pero hay más características que hay que tener en cuenta a la hora de valorar una tarjeta de sonido:

- **Frecuencia de muestreo:** cuanta más mejor, pero para alcanzar la calidad de un CD de Audio (CD-A) basta con 44,1 Khz., aunque hay algunas a más de 45 Khz.
- **Tamaño de muestra:** igualmente, cuanto mayor sea el tamaño, mayor calidad. Un CD-A suele usar 12 bits, pero la mayoría de las tarjetas son a 16 bits.
- **MIDI** (*Musical Instruments Digital Interface*): interfaz digital para el control de instrumentos musicales. A través de este sistema podremos controlar desde el ordenador todos los instrumentos que tengan MIDI (teclados, sintetizadores, cajas de ritmos...). En algunos casos, utilizando conectores especiales, podremos conectar más de un instrumento. Las partituras se pueden almacenar en ficheros MIDI (`.MID`) para reproducirlas siempre que se quiera.
- **Número de voces:** número de instrumentos o sonidos que la tarjeta es capaz de emitir a la vez. Puede variar desde 20 a más de 50.
- **DSP** (*Digital Signal Processor*): procesador de señal digital, que permite manipular la señal para conseguir distintos efectos, como por ejemplo, un sonido más realista y nítido.
- **Técnica de generación de sonido:** hay dos técnicas básicamente:
  - **Sintetizador FM** de sonidos esterofónicos y monofónicos: existe un chip (casi siempre de la casa Yamaha) en el que se encuentran programadas las ondas de diferentes sonidos e instrumentos. Aquí se genera un sonido artificial, variando esas ondas.
  - **Síntesis por tabla de ondas** (*Synthesis Wave Table*): aquí no se genera la onda, sino que se reproduce de una grabación previa del instrumento real. Con esta técnica los sonidos son reales. Es obvio que no se graban todas las notas de todos los instrumentos, sino sólo una nota, y a partir de esta se generarán el resto variando la frecuencia de la nota grabada. Ejemplo: Digitalizamos de un instrumento la nota La 4 (440 Hz) y la almacenamos. Posteriormente podemos reproducirla al doble de velocidad (880 Hz) consiguiendo una nota de una escala superior (La 5). Este sistema es válido pero dentro de un intervalo por lo que en sistemas de mayor calidad existen varias digitalizaciones de distintas notas de cada instrumento.
    Muchas tarjetas disponen de memoria RAM en la que grabar nuevos instrumentos.
- **Software incluido:** la tarjeta debe incluir unos programas básicos (*drivers*), pero también puede incluir un secuenciador, programa para componer directamente todo tipo de partituras en un pentagrama (notación diastémica). También es posible el proceso contrario, tocar un instrumento y que el ordenador deduzca la partitura asociada. Así mismo, debe incluir librerías de funciones (en C por ejemplo), para programar la tarjeta.

- **Potencia de salida** para los altavoces: suele oscilar entre 2 y 5 Watios.
- **Volumen:** algunas permiten controlar el volumen con un mando manual que incorporan.
- **Efectos:** algunas incluyen efectos como el **Sonido 3D** (sorround), que permite escuchar sonido como si el foco emisor estuviese en cualquier punto situado en torno nuestro (arriba, abajo, izquierda, derecha, adelante o detrás). Otros efectos son la reverberación (sensación espacial) y chorus.

En el futuro, es muy posible que este tipo de tarjetas vendrán de serie en cualquier ordenador, para multitud de aplicaciones (correo electrónico por voz, teléfono, contestador automático...).

2.4.3.2 MODEM/Fax.

Un ordenador personal sin modem es como una casa sin biblioteca o un edificio sin cimientos. En la era de las comunicaciones, tener un ordenador desconectado es aislarse del mundo y aislarse de la información (no olvidemos que la información es poder... físico y psíquico).

Pero dejémosnos de filosofías y veamos qué es y para qué sirve un modem: Un modem es un aparato que principalmente sirve para comunicar ordenadores entre sí, usando una línea de comunicación analógica (normalmente la línea telefónica).

Ya sabemos que una **señal digital** es aquella señal que sólo puede tomar un conjunto de valores finito (en los ordenadores sólo toman 2 valores, cero o uno ¿recuerdan?), y una **señal analógica** es la que puede tomar infinitos valores dentro de un rango, como por ejemplo el sonido audible, que puede tomar distintos valores de frecuencia entre el rango audible por el hombre.

Para comunicar dos ordenadores, que usan señales digitales (código binario) , a través de la línea telefónica, ideal para transmitir voz y sonido (entre 300 y 3400 Hz.), hay que convertir una señal digital a una analógica -**MODULAR**-, transmitirla por la línea y al llegar al destino habrá que reconvertir la señal analógica recibida en una digital -**DEMODULAR**-. La señal analógica no es otra cosa que distintos pitidos en el rango de frecuencias que permita la línea telefónica.

Un **MODEM**, es el dispositivo encargado (de ahí su nombre), de **MODular** y **DEModular** señales, para hacer posible una comunicación de datos digitales (programas, ficheros de texto, ficheros de sonido, imágenes...) por las líneas analógicas ya existentes.

Nada más absurdo que un modem, si existiera ya la famosa **RDSI** (*Red Digital de Servicios Integrados*), que será como la red telefónica pero especial para señales digitales, por la que se retransmitirán canales de TV, música y radio, canales de datos, canales de voz... y muchas cosas más. Para la RDSI quedan, seguro, más de 10 años.

Veamos un resumen en la figura 2.25. Supongamos que en nuestra comunicación, se van a transmitir datos desde el ordenador A al B:

Pasos:

1. El ordenador A, transmite la información digital a su modem.
2. El modem A, MODULA la información.
3. La información modulada se transmite en forma de pitidos (señal analógica), por la red telefónica (RTC, *Red Telefónica Conmutada*).

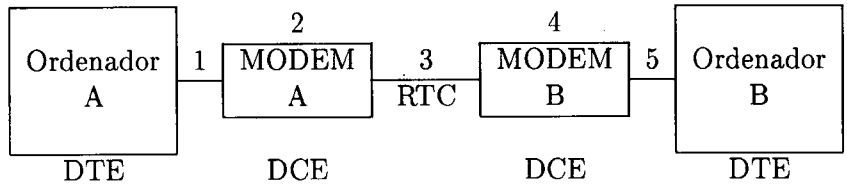


Figura 2.25: Comunicación por MODEM.

4. La señal analógica llega al modem B, que la DEMODULA y convierte a señal digital.

5. La señal digital es enviada del modem al ordenador B.

donde:

- **DTE** es *Data Terminal Equipment*, Equipo Terminal de Datos. El equipo que genera y/o utiliza los datos. El equipo terminal de la comunicación (donde empieza o termina la comunicación, origen o destino).

- **DCE** es *Data Communication Equipment*, Equipo de Comunicación de Datos. El dispositivo que transmite los datos y se encarga de las comunicaciones.

- **RTC** es Red Telefónica Conmutada. La red telefónica normal, pensada y usada para transmitir principalmente voz. Se llama conmutada, porque al realizar una llamada se conmutan unos interruptores de forma que siempre existe un enlace directo entre ambos extremos de la línea. Cuando todos los enlaces posibles están ocupados con distintas llamadas, e intentamos realizar una llamada más, esta última es imposible realizarse y nos sale una voz pregrabada que nos indica: *Por saturación rogamos vuelva a marcar pasados unos minutos* (¿A que les suena?).

Sabiendo todo esto, podemos decir que lo importante de un modem es su velocidad, ya que cuanto más rápido sea, menos tarda en transmitir la información y menos tendremos que pagar a la empresa telefónica, por la utilización de su línea, igual que ocurre en las llamadas de voz (cuanto menos tiempo hablemos, más barata nos sale la llamada).

La RTC permite un máximo de 2400 baudios (cambios de señal por segundo). Si tenemos 2 señales distintas, y asignamos a una el valor 0 y a otra el valor 1, podremos transmitir a 2400 bps (bits por segundo). Sin embargo, lo que se suele hacer es tener más tipos de señales y asignar a cada una más de 1 bits. Así, si asignamos 2 bits por señal, podremos transmitir a 2 bits por baudio, que son 4800 bps. Con 3 bits por baudio, llegamos a los 7200 bps, y con 12 alcanzamos los 28800 bps.

Como hemos dicho, a medida que aumentamos el número de bits por señal, se requieren mayor cantidad de señales distintas (frecuencias distintas). Así, para transmitir 12 bits por señal, necesitamos 2^{12} frecuencias distintas (4096). Tantas señales, con frecuencias distintas, existen, pero están tan próximas unas de otras que el más mínimo ruído hace que lo que antes era una señal ahora sea otra, produciéndose una transmisión incorrecta.

El CCITT (Comité Consultivo Internacional de Telegrafía y Telefonía), establece unos estándares para las normas que deben usar los modems, según su velocidad. Algunas de ellas se ven en la tabla 2.8.

Norma	Especificación
V.21	hasta 300 bps (bits por segundo).
V.22	hasta 1200 bps.
V.22bis	hasta 2400 bps, nueva norma Ibertext
V.23	antigua norma Ibertext (1200/75)
V.32	hasta 9600 bps.
V.32bis	hasta 14.400 bps.
V.34	hasta 28.800 bps. (igual que V.Fast Class). Con compresión de datos llega a 115.200 bps.
V.42	norma de corrección de errores.
V.42bis	norma de compresión de datos.

Tabla 2.8: Normas de transmisión en un Modem.

Actualmente se está preparando una norma para transmitir a 38400 bps que deberá aprobar el ITU (*International Telecommunications Unit*) que engloba el CCITT.

Hay además otras normas para usar el modem como **fax** (V.17 entre otras). Otras normas nos aseguran que la información llega libre de errores (V.42, o MNP 2 a 4), ya que las líneas telefónicas no son todo lo buenas que debieran.

Por último comentaremos que hay protocolos para compresión de datos, es decir, que consiguen que el tamaño de los datos a transmitir sea menor, comprimiéndolos, de forma que el tiempo en transmitirlos sea menor (V.42bis o MNP5), y el ahorro sea mayor.

Los protocolos MNP no son del CCITT, sino de un fabricante americano, pero dada su popularidad se han convertido en estándar de hecho.

Características más destacadas de un MODEM:

- **Velocidad** y normas que cumple: básicamente las que hemos comentado, pero recomendamos un mínimo de 14.400 bps con V42 y V42bis.

- **Fax:** si incorpora normas para mandar faxes, como la CCITT v.29. Normalmente sí se incorporan, pues los circuitos para fax y para modem son prácticamente iguales. Ver tabla 2.8.

- **Corrección de errores y Compresión de datos:** si tiene normas para esas utilidades. Ver tabla 2.8.

- **Software** que incluye: para conexión a BBS, Ibertext... Lo importante es tener uno para conexión a BBS, luego, de una BBS podemos traernos otros programas que sean mejores. Los hay muy buenos y muy baratos.

- **Manuales:** instalar un MODEM no es excesivamente complicado, pero puede dar más de un quebradero de cabeza si no tenemos unos manuales detallados. En cualquier caso, no se debe olvidar aquel viejo refrán: *Con paciencia y buen vino... se anda el camino.*

- **Interface:** no tiene demasiada importancia, salvo que nuestro ordenador sea portátil, en cuyo caso existen unos modems PCMCIA que son poco más grandes que una tarjeta de crédito.

- **Externo/Interno:** los internos van insertados en un slot libre del ordenador y los externos conectados a un puerto serie por la parte trasera del ordenador.

  Para los externos es muy importante saber el chip UART (*Universal Asynchronous Receiver Tranceiver*, o Receptor Transmisor Asíncrono Universal), que incorpora el ordenador, ya que si este es malo no podrá transmitir el modem a 14400 bps o más.

  Los internos incorporan su propio chip UART, son más baratos (no tienen caja, ni luces de estado) y estorban menos, ya que están dentro de la caja del ordenador. Los externos usan la UART del ordenador, por lo que si ésta es mala no podrán correr a toda la velocidad que puedan.

  La única UART recomendable, dadas las velocidades actuales es la UART 16550 u otra mejor (la 8250 y la 16450 ya están anticuadas).

En un tema posterior se verá más detalladamente las utilidades del MODEM, pero resumiendo, básicamente son tres, además de la comunicación de dos ordenadores privados:

1. **Envío/Recepción de fax:** un sistema de comunicación básico en las empresas actuales. Consiste en la transmisión telefónica de documentos impresos. Con la incorporación de un fax en el ordenador, ya no es necesario imprimir el documento y mandarlo por fax, sino que *faxear* es como imprimir en un fax remoto. Mandar un documento por fax es como mandarlo a imprimir, pero que llegará a un fax remoto, y este lo imprimirá o no. El objetivo es conseguir la famosa *oficina sin papel.*

2. **Conexión con Videotext** (en España Ibertext): son unas bases de datos de todo tipo de información profesional o lúdica a las que se puede acceder. Por ejemplo, banco en casa, compras en grandes almacenes, participación en campañas de Amnistía Internacional, lectura de artículos de revistas... Se explicará más en detalle en otro tema posterior.

3. **Conexión a BBS** (*Bulletin Board System*), a FidoNet y otras redes: Todo esto se explicará más en detalle en un tema posterior, pero aquí daremos unas ideas básicas.

   Un **BBS** (sistema de tablón de anuncios) es un ordenador conectado las 24 h. que permite al que le llame llevarse ficheros, participar en conferencias o usar la mensajería privada:

   - **Ficheros:** el usuario de un BBS puede llevar o traer ficheros del BBS. Los ficheros pueden ser de todo tipo: ficheros de texto con información sobre multitud de temas, imágenes de todo tipo y formato, ficheros de música y sobre todo programas, programas y programas. Todo tipo de programas, pero nunca comerciales, sino los llamados programas *freeware* (gratuitos) y *shareware* (probar antes de pagar).
   - **Conferencias** o áreas de mensajes echo: están formadas por usuarios de distintos BBS que están interesados en un mismo tema, y cualquier interesado en ese tema puede conectarse a ese área o conferencia y leer y contestar los mensajes que ahí se escriban. La comunicación entre los BBS hace que los mensajes lleguen a casi cualquier lugar. Existen multitud de conferencias sobre multitud de temas, desde los más triviales e intrascendentes hasta los más serios: Astronomía, informática, electrónica, humor, ecología, historia, política, juegos, comics, libros, economía,

aviación, ofertas de empleo, ofertas de compra/venta, gráficos, fotografía, animales, plantas, deportes, telecomunicaciones, medicina, motor, ciencia, tecnología, cultura, música...

Todos esos son una pequeñísima muestra de los temas sobre los que podemos hablar, preguntar y responder y en definitiva **APRENDER**.

Si tenemos alguna duda sobre cualquier tema, basta con activarse el área más apropiada sobre el tema y preguntarla. No suele faltar alguien que responda nuestra pregunta, aunque esto no es la panacea.

Igualmente, si alguien pregunta algo y nosotros lo sabemos, no debemos tener inconveniente en responderle en todo o parte a su pregunta. Salvo raras excepciones los usuarios de FidoNet son muy amables. **FidoNet** es el nombre genérico que se le da a esta red formada por BBS de todo el mundo y que se explicará más en detalle en un tema posterior.

- **Mensajería:** de la misma forma que el correo convencional, a través de este medio podemos escribir cartas y mandarlas a usuarios de los BBS. Obviamente, cada usuario (o punto) de FidoNet tiene su dirección propia (que son del tipo 2:345/803.13).
- **Conexión a otras redes:** además de **FidoNet**, existen otras redes similares, y con un modem también es posible conectarse a la famosa red internacional **InterNet**, la cual se explicará en otros temas posteriores.

No hay que olvidar que no todas las BBS pertenecen a la red FidoNet, es decir, uno puede montar su BBS y no ser nodo de FidoNet. Igualmente, no hay que ser punto de una BBS (tener dirección), para conectarse ella. Para conectarse sólo hace falta lo siguiente: Un ordenador, un modem, un programa de comunicaciones y una línea telefónica que llegue al modem del ordenador. Naturalmente, ser punto de una BBS tiene sus ventajas, pero eso es otra historia.

Un punto muy importante es, que el mito de que por los BBS y FidoNet se contagian muchos virus, es totalmente falso. Incluso, por este medio podremos conseguir buenos y recientes antivirus.

2.4.3.3 Tarjetas de Red.

Son las encargadas de conectar un ordenador a una red de ordenadores a través de algún medio (principalmente un cable). Quizás su característica más importante sea su velocidad de transmisión/recepción. Las hay de muy diversos tipos dependiendo del tipo de red (véase el tema de redes de ordenadores más adelante en esta misma obra).

2.4.3.4 Tarjeta de Vídeo y Sistema de Videoconferencia.

Una tarjeta de Vídeo es un dispositivo ideado para manejar películas, vídeos o animación. Su principal función es comprimir las imágenes que le llegan desde un vídeo o una cámara y digitalizarlas para almacenarlas en un formato digital (`.AVI` u otros). Posteriormente descomprimirá las imágenes de una película de forma que se puedan mostrar a la velocidad adecuada en la pantalla.

El objetivo de un **Sistema de Videoconferencia** es conseguir transmitir imagen y sonido en una conversación entre dos puntos alejados. La información debe comprimirse y enviarse rápidamente para poder conseguir la simultaneidad de la comunicación. Por eso, no hay tiempo para introducir códigos de corrección de errores.

Debido a la cantidad tan enorme de información que se tiene que enviar en tiempo real, la RTC no resulta muy adecuada para ello, aunque se puede utilizar. Lo ideal es usar la RDSI. El problema es la escasa implantación geográfica y el coste que es muy superior a la RTC.

Un sistema de videoconferencia consta de una cámara de vídeo conectada a una tarjeta para la captura y compresión de la señal y un interfaz de comunicaciones (tarjeta de red, de RDSI, modem...) y por supuesto, un programa que coordine todos los elementos.

2.4.3.5 Dispositivos de Realidad Virtual.

La Realidad Virtual (RV) es una tecnología por la que intentamos engañar nuestros sentidos con el fin de introducirnos en un mundo creado por el ordenador. El objetivo será que podamos movernos por ese mundo e interactuar con él. Se trata, en síntesis, de introducirnos en un mundo, como lo hacemos en un videojuego, pero de manera que nuestra relación con ese mundo no sea sólo a través de un mando (joystick) y de una pantalla, sino que sintamos realmente que estamos inmersos en un mundo tridimensional con todo tipo de sensaciones que eso conlleva. En un videojuego sólo existen 2 dimensiones. En la realidad virtual se pretende que existan las mismas 3 dimensiones espaciales que en la vida real.

El término Realidad Virtual (RV) puede parecer contradictorio, pero está muy bien elegido. Algo *virtual* es algo que no es *real*. La RV intenta que parezca **real** un mundo **virtual**.

El periférico más importante es el llamado **casco o visor de RV**. Consta de un casco con 2 pequeñas pantallas, una para cada ojo y dos auriculares (uno para cada oido). Así podemos obtener una visión tridimensional y un sonido estereofónico.

En cada pantalla se muestran distintas imágenes para dar la sensación de que estamos en un mundo tridimensional. Ambas imágenes serán generadas por un potente ordenador según un programa previo.

El casco, debe ser también un periférico de entrada de datos a la computadora, ya que debe incorporar sensores que indiquen en todo momento la posición del mismo. Así, si el usuario del casco de RV mira hacia la derecha, hacia la izquierda, hacia arriba o hacia abajo, el ordenador debe capturar esos datos en tiempo real y cambiar las imágenes de forma coherente con dichos movimientos.

El **guante digital** o **guante de datos** es un guante que integra cables de fibra óptica, para detectar perfectamente los movimientos de la mano. El objetivo es poder interactuar con el mundo virtual, con la mano. Así podremos coger objetos del mundo, cambiarlos de lugar, abrir puertas y ventanas virtuales y un sinfín de posibilidades.

Igualmente, los guantes de datos deben ser receptores de las sensaciones que se produzcan. Así, si cogemos un objeto virtual, el guante debe ejercer sobre la mano la presión suficiente para simular que el objeto está en nuestra mano.

En el futuro, se pretende crear un traje de datos, de forma que detecte todos nuestros movimientos y no solo los de la mano (como en la película "El cortador de cesped").

Naturalmente, para instalar un sistema de RV se necesita una máquina capaz de procesar miles de datos al segundo. Con ordenadores Pentium se puede hacer algo, pero si queremos que la realidad virtual sea más real deberemos usar estaciones Silicon Graphics de varios millones de pesetas. Aún así, esta técnica tiene mucho camino por recorrer.

2.4.4 Periféricos de ENTRADA.

Como bien se sabe un periférico de entrada es un aparato o dispositivo que sirve para **meter** información en el ordenador. Aquí veremos los más importantes, teniendo en cuenta que alguno de ellos puede ser, además, un periférico de salida y no exclusivamente de entrada.

Los que veremos a continuación son:

- Lectora de tarjetas perforadas y de cinta de papel.
- Teclado.
- Dispositivos apuntadores:
  - Ratón.
  - Lápiz óptico.
  - Trackball.
  - Joystick.
  - Puntero táctil (*Track Pad*).
  - Pantalla sensible al tacto.
  - Tableros gráficos.
- Scanner.
- Lectores ópticos.
- Detector de caracteres magnetizables.
- Sensores de señales analógicas.
- Dispositivo reconocedor de voz.

2.4.4.1 Lectora de tarjetas perforadas y de cinta de papel.

Como dijimos al hablar de las tarjetas perforadas y de la cinta de papel, los dispositivos lectores incorporaban unas células fotoeléctricas que detectaban cuando había y no había agujero. Había otros sistemas pero no vamos a verlos aquí, porque estos periféricos, igual que las perforadoras, pasaron ya a la historia.

2.4.4.2 Teclado.

El teclado (keyboard) es, sin duda, el dispositivo de entrada por excelencia y además, es uno de los más antiguos (ver figura 2.26). Prácticamente todos los ordenadores incorporan este periférico, y seguirá siendo así hasta que consigamos que estas máquinas nos entiendan verbalmente todo cuanto les podamos decir, lo cual es la entelequia informática.

Los primeros, se les llamaba teletipos e incluían también una pantalla o monitor (dispositivo de salida por excelencia). Ambos periféricos son llamados **consola** y suelen ser la entrada y salida estándar, respectivamente, de casi todos los sistemas.

Por entrada/salida estándar se entiende a aquellos dispositivos de los que, por defecto, los programas leen/escriben su información. Es decir, si un programa lee datos de su entrada

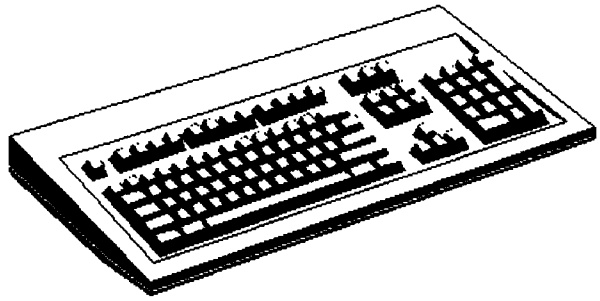


Figura 2.26: Teclado expandido convencional.

estándar, los leerá del teclado, esto es, esperará a que el usuario/operador introduzca los valores que el programa espera. Igualmente, si un programa saca datos por su salida estándar, lo que hace es visualizarlos en pantalla.

Pero... estas entrada y salida estándar se pueden cambiar, de forma que un programa que toma los datos de su entrada estándar y saca los resultados en su salida estándar, puede usar el teclado y el monitor, pero también puede tomar su entrada de un fichero y guardar los resultados en otro fichero u otro dispositivo (impresora...). Las funciones de redirigir las E/S estándar son propias del S.O.

Volvamos al teclado. Al pulsar o presionar una tecla se cierra un conmutador que hay debajo de cada tecla que hace que unos circuítos codificadores generen el código correspondiente al carácter seleccionado. Según su conmutador hay dos tipos: de tacto duro (mecánicos), que hacen un clic al pulsar cada tecla y de tacto suave (de membrana).

El más difundido es el teclado extendido, el cual se divide en las siguientes partes, las cuales pueden variar sensiblemente según el fabricante:

- **Teclado convencional** de una máquina de escribir, tipo QWERTY (llamado así por las primeras letras de la fila superior). Arriba tiene los números y otros caracteres. Este módulo de teclas tiene una serie de teclas especiales:
  - *Tabulador:* Tecla a la izquierda, etiquetada con dos flechas hacia izquierda y derecha con una rayita vertical.
  - *Bloq. Mayús:* Tecla para bloquear el uso de mayúsculas. Al darle se encenderá o apagará una luz en la parte superior derecha del teclado.
  - *Shift/Mayús:* Tecla de desplazamiento a las mayúsculas esporádicamente. A veces está etiquetada con una flecha hacia arriba.
  - *Control/Ctrl:* Usada para códigos especiales que dependen del programa en ejecución. Por ejemplo, usar Ctrl-C, significa pulsar la tecla Ctrl y sin soltarla, pulsar, simultáneamente, la tecla C. Esta combinación, en concreto, suele abortar el programa en ejecución, volviendo al S.O. Por ejemplo, Ctrl-S lo que hace es parar el programa (no lo aborta) de forma que para que siga basta con pulsar una tecla cualquiera.
  - *Alt:* Alternativa. Su uso es igual que Ctrl, pero sus funciones serán distintas, dependiendo del programa. Uno de los usos más estándares es usarla para obtener un carácter, sabiendo el valor decimal de su código ASCII. Ej. la letra 'A' tiene el

código ASCII 65. Para obtenerla podemos pulsar la tecla Alt, y sin soltarla pulsar en el teclado numérico el número 65. Al soltar la tecla Alt, aparecerá en pantalla una 'A'.

- *Alt Gr:* Alternativa gráfica. Se usa para conseguir algunos caracteres gráficos especiales, como \, @, |, # y otros.
- *Retroceso:* También etiquetada con una flecha hacia la izquierda, y situada en la esquina superior derecha de este bloque de teclas. Se usa para borrar el carácter que está justo a la izquierda del cursor.
- *Return/Intro:* También etiquetada con una flecha que viene de arriba y tuerce hacia la izquierda, o como Transmit/Xmit. Se usa como retorno de carro al final de una línea, para decirle al ordenador que ejecute YA la orden dada o para seleccionar opciones de un menú... Es una de las teclas más importantes.
- *Teclas de Windows95:* Algunos teclados incluyen dos teclas especiales que sólamente funcionan bajo el sistema operativo Windows95.

- **Teclas de función:** en la parte superior del teclado y etiquetadas como F1, F2, F3 ... hasta F12. Su uso depende de cada programa. Un estándar, de hecho, es usar F1 como tecla para solicitar ayuda o consejo al programa en ejecución.

- **Tecla de escape, ESC:** su uso depende del programa, pero normalmente se utiliza para salir de la opción, parar la ejecución o salir del programa.

- **Flechas:** Cuatro teclas con cuatro flechas (arriba, abajo, izda. y dcha.). Se usa para mover el cursor o la opción activa.

- **Teclas de edición:**

  - *Insert/Ins:* Para poner el teclado en modo inserción (para insertar algo) o sobreescritura (para no insertar).
  - *Supr/Del:* Para borrar algo, normalmente, el carácter que está justo en la misma posición del cursor, corriéndose una posición a la izquierda los caracteres que haya a la derecha. La combinación Ctrl-Alt-Supr hace que el ordenador se apague y se vuelva a encender (como reset).
  - *Inicio/Home:* Pone el cursor al principio de algo (un texto...).
  - *Fin/End:* Pone el cursor al final.
  - *Re Pág:* Retrocede una página.
  - *Av Pág:* Avanza una página.

- **Teclado numérico:** son teclas redundantes, es decir, se pueden prescindir de ellas, pero se incorporan para dar facilidad a la hora de meter números u operar con ellos. Tiene una tecla llamada Bloq.Num que hace que los números se conviertan en las teclas de flechas y en las de edición. Al darle a esta tecla se encenderá/apagará una luz en la parte superior derecha del teclado.

- **Otras teclas:**

  - *Impr Pant:* Manda a impresora el contenido de la pantalla actual.

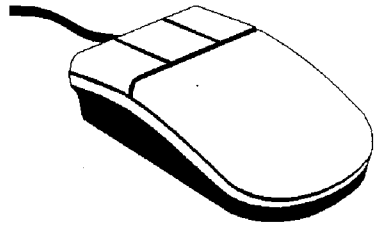


Figura 2.27: Ratón de 3 botones.

- *Bloq Despl:* Bloqueo de Desplazamiento. Igual que las teclas de bloqueo de mayúsculas y bloqueo numérico, también suele tener una luz que avisa de su estado.
- *Pausa:* Detiene la ejecución del programa. Para volverlo a activar hay que pulsar cualquier otra tecla.

El teclado suele tener un driver o programa residente para controlar los caracteres propios de cada país, como por ejemplo en España, la Ñ, la ñ, los acentos...

Es tan normal tener un teclado como periférico que normalmente se conecta directamente a la placa base, es decir los circuitos controladores vienen ya en la placa principal, y la conexión es mediante clavijas DIN o mini-DIN.

2.4.4.3 Dispositivos apuntadores.

Sirven para apuntar en la pantalla del ordenador, o sea, para introducir posiciones (x,y) de la pantalla. Un ejemplo, en la pantalla aparecen tres opciones y el programa espera que elijamos una. Esto lo podemos hacer moviendo el cursor (normalmente una flecha) con un dispositivo apuntador, situándonos encima de la opción a escoger y validándola, dándole a un botón del dispositivo. Otra utilidad es, por ejemplo, para dibujar, traduciendo los movimientos en el dispositivo apuntador por líneas en la pantalla.

Estos dispositivos se pueden anular, realizando todas las funciones con el teclado, en especial con las teclas de flechas, pero las últimas tendencias en aplicaciones, son, precisamente, minimizar el uso del teclado, pudiéndose hacer prácticamente todo, con una mano encima de uno de estos dispositivos, principalmente el ratón.

2.4.4.3.1 Ratón: Es el dispositivo apuntador más extendido y estandarizado. Un ratón (o mouse) consta de una pequeña cajita plástica (de menor tamaño que un puño) con 2 ó 3 botones por encima y una bola que gira por debajo (ver figura 2.27). Los movimientos que se hagan desplazando este dispositivo, harán rodar la bola, y se traducirán en movimientos en la pantalla del ordenador.

Sus orígenes se remontan a las investigaciones realizadas en los laboratorios de Xeros en Palo Alto y su primera comercialización se hizo con el Apple Lisa, antepasado de los actuales Macintosh.

La bola inferior es de un material con mucho rozamiento, para que no resbale al moverla por la mesa. Aún así, muchos ratones incluyen una pequeña alfombrilla para facilitar el movimiento del *roedor*.

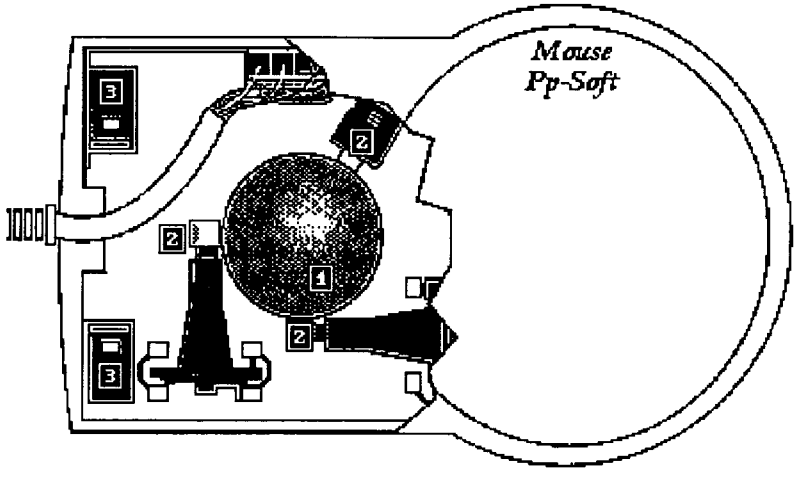


Figura 2.28: Interior de un ratón.

Al moverse la bola por acción de la mano, hace que se muevan dos rodillos en posiciones perpendiculares entre sí, los cuales detectan los movimientos en el eje X y en el eje Y, respectivamente. Para ver estos rodillos basta con darle la vuelta al ratón, girar una pequeña tapadera (con un orificio por donde asoma la bola), sacar la bola y mirar en su interior.

O sea, que si movemos el ratón con la mano a través del eje X, por ejemplo, el rodillo del eje Y no se moverá y el del eje X se moverá tan deprisa como lo hagamos nosotros. Igualmente podemos desplazar el ratón por el eje Y, o en diagonal, haciendo que se muevan ambos rodillos simultáneamente.

De esta forma, movemos el cursor por la pantalla y cuando esté en el sitio deseado, validamos la posición, pulsando un botón (normalmente el izquierdo).

Veamos el interior de un ratón en la figura 2.28. Básicamente consta de:

1. **Bola del ratón:** Esta rodará encima de una superficie lisa.
2. **Rodillos:** Se moverán según el movimiento de la bola. En realidad sólo son necesarios dos. Normalmente son usados los dos que son perpendiculares entre sí.
3. **Botones:** Estos transmitirán una señal de estado indicando si están pulsados o están sin pulsar. Puede haber 2 ó 3 botones.

Estos dispositivos deben incluir sus propios *drivers*, o programas manejadores, de forma que los programas (aplicaciones) se limiten a comprobar la posición del ratón y el estado de sus botones.

Existen otros ratones con tecnología óptica. Estos, son más caros y más precisos (permiten apuntar con mayor precisión). Su funcionamiento es similar. Tienen debajo un haz de luz (roja, en muchos casos) en vez de una bola y es obligatorio moverlo encima de una alfombrilla metálica con unas rayas horizontales y verticales, formando una cuadrícula, donde los cuadrados más pequeños tienen menos de 1 mm. de lado. Al moverse por esta alfombrilla la luz detecta el paso de estas rayas y lo traduce en movimientos del cursor, dependiendo de si pasan rayas horizontales, verticales o ambas. Esta tecnología no tiene los problemas que ocasiona la suciedad que se acumula en el interior de un ratón de fricción.

2.4.4.3.2 Trackball: Usa la misma tecnología que el ratón. Es prácticamente igual que un ratón invertido, en el que no hay que mover el aparato entero, sino hacer mover, con la mano, directamente la bola. Con un trackball se puede ser más preciso que con el ratón, ya que se tiene más precisión moviendo el dedo pulgar que moviendo la mano entera. Además, no cogen tanto polvo como los ratones al arrastrarse por la mesa. Es muy útil en espacios donde no haya una mesa libre para poder deslizar el ratón. Por ejemplo, se usa mucho en los ordenadores portátiles, ya que no se sabe en qué lugar pueden llegar a usarse estos pequeños ordenadores.

Existe también el trackball óptico. Logitech, con la llamada tecnología *Marble Sensing*, ha eliminado el rozamiendo de estos dispositivos, evitando así el desgaste y la pérdida de precisión por la suciedad acumulada, y aumentando la vida del dispositivo.

Esta técnica se basa en la detección del movimiento de la bola por elementos opto-electrónicos. La bola, y no por azar, es de color rojo con puntos negros de diverso tamaño (que ofrecen un gran contraste bajo luz infrarroja). La bola se ilumina por unos LEDs, y un sensor registra los movimientos sin necesidad de contacto directo. La imagen de la bola es analizada más de 1000 veces por segundo y un microprocesador calcula el movimiento, con gran precisión, en los ejes X e Y.

2.4.4.3.3 Lápiz óptico: Tiene la forma de una pluma o lápiz grueso con uno o varios botones en un lateral, de cuyo extremo sale un cable que va a la unidad controladora.

Su uso es más simple que el ratón: Para señalar algo en la pantalla basta con apuntar con el lápiz en ella, como si quisiéramos dibujar encima.

Para entender su funcionamiento hay que recordar el modo de trabajo de la pantalla o monitor, ya que este periférico está intimamente relacionado.

En la punta del lápiz hay una pequeña abertura por la que puede pasar la radiación luminosa de la pantalla, la cual genera una señal eléctrica (usando un fotodetector) que es transmitida a la tarjeta gráfica si se pulsa el botón.

Como recordaremos, la imagen de la pantalla se ve periódicamente refrescada o redibujada, para evitar que se pierda, por un haz de electrones que recorre la pantalla, de una determinada forma y velocidad, pasando por todos los puntos (píxeles).

En el momento de incidir este haz de electrones, que recorre todos los píxeles de la pantalla, justo en el pixel que apunta el lápiz óptico, se genera un pulso eléctrico, por el que se obtiene el tiempo que ha tardado el haz de electrones en recorrer desde el primer pixel de la pantalla hasta ese pixel. Sabiendo ese tiempo, la velocidad del haz de electrones y el orden en recorrer los píxeles, se pueden hallar las coordenadas exactas del pixel donde estaba situado el lápiz óptico.

2.4.4.3.4 Joystick: El Joystick, palanca de control o palanca de juegos, es una palanca, que sale, hacia arriba, de una cajita con varios botones.

El movimiento de esta palanca, inclinándola hacia los lados, se traduce en movimiento del cursor (arriba, abajo, derecha, izquierda, diagonal...).

Es un dispositivo ideal para juegos de reflejos y habilidad manual, pues permite un rápido movimiento y perfecto control de un objeto (cursor) por la pantalla.

A veces, sobretodo en equipos portátiles, se incorpora un pequeño joystick entre las teclas G, H y B del teclado.

Existe otro periférico, muy poco usado en la actualidad, de características similares, llamado Paddle. Consiste en dos botones o diales con los que controlamos el cursor (uno para la horizontal y otro para la vertical).

2.4.4.3.5 Puntero táctil (*Track Pad* o *Point Pad*): Este novedoso dispositivo consiste en una pequeña superficie de pocos centímetros cuadrados. Para mover el puntero en la pantalla, se deberá deslizar el dedo sobre la superficie de contacto. Deslizando el dedo hacia un sentido, el puntero se moverá de igual forma. Utiliza un algoritmo de coordenadas relativas que le hacen funcionar de igual forma que un ratón.

Para hacer *clic* puede que existan unos botones (como en un ratón), o también es posible dar un ligero golpecito sobre la superficie de contacto.

Este dispositivo no tiene piezas móviles, por lo que no le afecta el polvo de la mesa, ni la humedad.

2.4.4.3.6 Pantalla sensible al tacto (o táctil): Ya lo hemos visto en los periféricos de salida, pero cuando es sensible al tacto es también un periférico de entrada para introducir posiciones de la pantalla. Naturalmente, no permite mucha precisión.

2.4.4.3.7 Tableros gráficos. Los tableros gráficos, tabletas digitalizadoras o digitalizadores constan de una superficie especial con marcas por la que se mueve un ratón muy sensible. Para poder apuntar exactamente a un punto determinado, el ratón suele incorporar una pequeña regla, con una cruz que indica el lugar exacto que está siendo apuntado. Esto permite una precisión excelente, por lo que se usa sobre todo en aplicaciones de diseño industrial (CAD), para transferir al ordenador gráficos, figuras, planos, mapas... Es como realizar la operación inversa al plotter.

A veces se sustituye este ratón por un lápiz con el que señalar en la tableta gráfica.

Encima del tablero se coloca el plano o dibujo, y con el puntero (lápiz o ratón) se pasará manualmente por el dibujo, como si se estuviera calcando. Automáticamente se van transfiriendo las coordenadas (x,y) de los puntos por los que pasa el puntero, quedando registrados en el ordenador.

En la figura 2.32 podemos ver un intrépido usuario con un tablero gráfico en su mesa de trabajo.

2.4.4.4 Scanner.

El scanner (o escáner) es un aparato de muy diversas formas, que sirve, básicamente para digitalizar imágenes. Es decir, para pasar a un formato (digital) que pueda tratar un ordenador, todo tipo de imágenes, fotos, dibujos, textos...

Hay algunos que son de sobremesa, similares a una fotocopiadora, otros son más manuales y se parecen más a un ratón, pero lo importante es su resolución y el número de colores que es capaz de distinguir. La resolución mide el número de puntos que es capaz de distinguir en una pulgada (ppp, puntos por pulgada o en inglés dpi). Por encima de los 1200 ppp los precios son demasiado elevados.

Con respecto al color, hay algunos que sólo distinguen distintos niveles de gris (blanco y negro) y otros distinguen también más de 16 millones de colores y tonalidades distintas (incluso más que el ojo humano). Los mejores, asignan 24 bits a cada pixel, de forma que se pueden conseguir los 16,4 millones de colores (que sólo cubren la cuarta parte de los colores

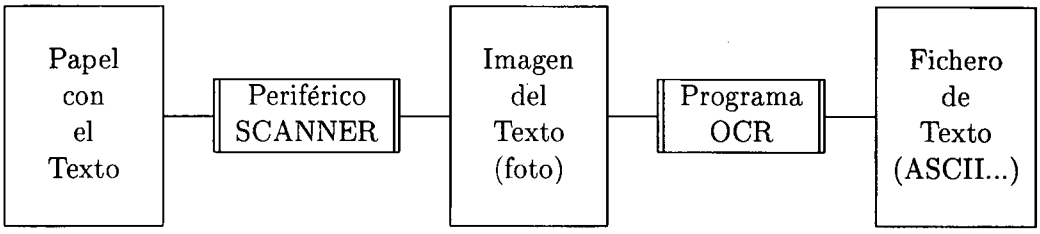


Figura 2.29: Esquema de trabajo de un OCR (Reconocedor óptico de caracteres).

reales). Estos escáneres miden la proporción de cada uno de los 3 colores básicos que posee el color original, en una escala de 0 a 255 (8 bits por cada color básico).

Hay aplicaciones (programas, software), llamadas **OCR** (*Optical Character Recognition*) de Reconocimiento óptico de caracteres, que se encargan de pasar a un formato de texto una imagen de un texto leída previamente por un scanner. Hay que diferenciar claramente lo que es una imagen de un texto (un conjunto de píxeles de distintos colores), y el texto en sí (un conjunto de caracteres distintos). En el formato imagen se almacenan los colores de todos los píxeles (texto o no) y cuando se pasa a texto por un OCR, obtenemos un fichero de texto, que se guarda almacenando uno a uno todos los caracteres. El esquema lo podemos ver en la figura 2.29:

El reconocimiento automático de caracteres manuales es un campo muy investigado y con pocos frutos prácticos, ya que no sólo cada persona escribe de distinta forma cada carácter, sino que una misma persona escribe cada carácter de muchas formas distintas.

2.4.4.5 Lectores ópticos.

Estos periféricos se usan para leer determinadas marcas (**OMR**, *Optical Mark Reader*). Son muy usados últimamente en multitud de aplicaciones y su tecnología es similar a la del scanner. Hay de muchos tipos. Los más usados son:

- **Detector códigos de barras:** Muy extendidos, sobre todo en supermercados y grandes almacenes, para el control de existencias (stock). Cada producto tiene su código de barras, que son unas líneas verticales negras en fondo blanco, por las que, según su posición y grosor, se codifica un número, que indica algunas características del producto (tipo, procedencia, fabricante...). Al salir o venderse, se le pasa este lector que informa a la base de datos que debe darlo de baja. De esta forma la persona encargada de la caja no tiene que teclear el precio de cada producto, minimizando el riesgo de error.

- **Detector de marcas:** Detectan las marcas hechas en unas zonas especiales del papel. Muy usados en quinielas, loterías y para corrección de exámenes tipo test en el que se marca la respuesta correcta. En muchos casos, para un óptimo funcionamiento, se aconseja que las marcas sean hechas con un lápiz blando del número 2.

2.4.4.6 Detector de caracteres magnetizables.

Interpretan unos caracteres especiales escritos en tinta magnetizable. El detector, lo primero que hace es magnetizar esta tinta y luego lee las zonas que están magnetizadas. Estos carac-

teres suelen ser con forma de códigos de barras. Se usan casi exclusivamente en aplicaciones bancarias (cheques...), y cada vez menos.

2.4.4.7 Sensores de señales analógicas.

Son dispositivos sensibles a la magnitud a medir, y dependen de la naturaleza de ésta: Temperatura (termómetro, termostato), sonido (micrófono), luz (células fotoeléctricas), humedad, nivel de agua de un pantano...

Se usan unos circuitos llamados conversores Analógico/Digital, para digitalizar las señales analógicas leídas y el programa adecuado obrará en consecuencia.

Ya se usan en multitud de aparatos como el de aire acondicionado. Están también los llamados edificios inteligentes, que detectan el nivel de luz de sus habitaciones y hacen subir/bajar las persianas o encender/apagar las luces. También usado en farolas callejeras en algunas ciudades, las cuales se encienden cuando son realmente necesarias.

2.4.4.8 Dispositivo reconocedor de voz.

Hoy día esto es un buen campo de investigación para electrónicos e informáticos, pero se han conseguido grandes avances, como el reconocimiento de algunas palabras específicas (números...).

El problema radica en que tienen que solucionar muchos problemas que los humanos hacemos con naturalidad: Separar fonemas y palabras, distinguir tonos de voces distintas, entonación...

Estos dispositivos tratan de identificar palabras dentro de un repertorio normalmente muy limitado. Lo que hacen es, al detectar un sonido, extraen ciertas características del mismo y lo comparan con ciertos patrones de palabras previamente memorizados.

Hay dos tipos básicos. El primero es dependiente del usuario y consta de una fase de aprendizaje en la que aprende a reconocer ciertas palabras de un usuario concreto. El segundo tipo es más general, es independiente del usuario y el vocabulario que reconoce suele ser más limitado.

Lo básico para trabajar en este mundo, es una tarjeta de sonido y un micrófono.

La utilidad de esto es inmensa, baste pensar que algún día podremos darle órdenes al ordenador verbalmente: "Escribe una carta a la sección de Ventas que diga..." ó "Imprime un listado de los clientes que...".

2.5 Clasificación de los Ordenadores.

Actualmente es difícil dar una clasificación general de las computadoras pues los avances tecnológicos han permitido clasificar como ordenador personal máquinas con más capacidad de proceso que antiguas macrocomputadoras.

Podemos dar una clasificación a nivel intuitivo, pero dejando claro que los límites de esta clasificación son difusos y a veces puede no ser fácil clasificar un ordenador en uno de los siguientes apartados. La siguiente clasificación está hecha, básicamente, de mayor a menor tamaño, precio, velocidad y fiabilidad.

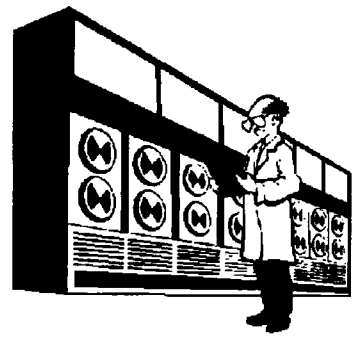


Figura 2.30: Un mainframe puede ocupar muchos m^2 de superficie.

2.5.1 MAINFRAMES.

Los mainframes, supercomputadoras o macrocomputadoras, son máquinas de gran capacidad de proceso, gran velocidad, gran fiabilidad y gran tamaño. Su precio es de bastantes millones de pesetas por lo que no son ordenadores al alcance de la mayoría de los particulares.

Son **multiusuario** por lo que puede haber cientos de usuarios conectados y trabajando a la vez con el mismo ordenador. Cada uno de estos usuarios puede trabajar en un microordenador (que tiene a su vez capacidad de proceso) o simplemente en un *terminal tonto*, que es un aparato (consola) sin capacidad de proceso, destinado simplemente a enviar y recibir los mensajes a el mainframe.

Son ordenadores típicos para la gestión de grandes empresas como grandes bancos y cajas de ahorro, donde suelen tener un equipo de muchos programadores trabajando a la vez en el mismo mainframe. Unos ejemplos de estos ordenadores son el IBM 3090S o el UNISYS 2200.

2.5.2 MINIORDENADORES.

Los miniordenadores o superminicomputadores (o más simplemente minis), son máquinas muy potentes pero de menor potencia, tamaño, velocidad, fiabilidad y precio que los mainframes. Son también multiusuario pero su número de usuarios máximo suele ser menor de 150 aproximadamente, aunque como hemos dicho los límites entre esos ordenadores y los mainframes son difusos.

Son ordenadores para empresas medianas, como lo podrían ser algunas cajas de ahorro de pequeño volumen de negocios. Unos ejemplos pueden ser el VAX-8350 o el MV-10000, aunque el más extendido es el AS/400 de IBM. No obstante, este tipo de máquinas está perdiendo adeptos debido a su alto coste y al gran rendimiento que están demostrando las últimas generaciones de PCs.

Las **WorkStation** (estaciones de trabajo) son un tipo de ordenadores que están a medio camino entre los minis y los microordenadores. Son ordenadores potentes y generalmente usados para trabajar bajo el S.O. UNIX (marca registrada de los laboratorios Bell). Una de sus más destacadas características es su conectividad (fácil de conectar a redes...). Son famosas las WorkStation de la familia SUN microsystems.

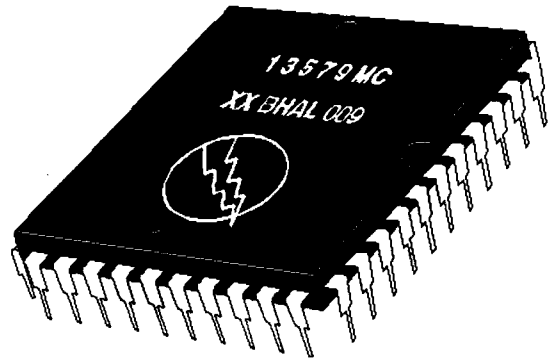


Figura 2.31: En un chip de pocos cm^2 caben millones de transistores.

2.5.3 MICROORDENADORES o PC's. Su historia.

Son ordenadores de pequeño tamaño y precio y cada vez más potentes y fiables. Sus características han hecho posible que la informática se introduzca casi como electrodoméstico imprescindible y de ahí su nombre de ordenadores personales (**PC**, *Personal Computer*).

Los avances tecnológicos han conseguido que estos ordenadores tengan, a veces, más capacidad de proceso que algunos antiguos mainframes y cada vez esto será más una realidad.

Aunque pueden usarse multiusuario, por sus características son normalmente monousuario y con un S.O. apropiado se consiguen que sean **multitarea**, es decir pueden ejecutar varios programas a la vez. Un S.O. ideal para sacar el máximo rendimiento a uno de estos ordenadores es el UNIX (hay una versión gratuita, de dominio público, llamada LINUX) o el OS/2 Warp Connect de IBM y Windows95 de MicroSoft (véase capítulo sobre Sistemas Operativos en esta misma obra). Mientras, las distintas versiones del DOS se quedan muy pequeñas para estos procesadores de grandes prestaciones.

2.5.3.1 Su historia.

A principios de los años 80 nació el **IBM PC** (más exactamente en 1981), que revolucionó el mundo informático por muchas cosas, entre ellas por el estándar que supuso. Se trataba de un pobre ordenador (pobre hoy, en su día era bien caro: ¡más de un millón de pesetas de aquellos años!) dotado con el procesador **8088** de la empresa **Intel**, trabajando a 4,77 Mhz. con el S.O. **MS-DOS** de la empresa **MicroSoft**.

A partir de entonces *todos* los procesadores intentarían ser compatibles con este primero pero trabajando a mayor velocidad. Y entonces surgieron una serie de procesadores clónicos **compatibles** con el de Intel, es decir, que podían ejecutar los mismos programas.

Más adelante salió el procesador **Intel 8086** y ambos fueron llamados XT. Luego salió el **Intel 80286** (conocido entonces como AT) que abreviadamente se le llamaba i286 (la i es de Intel), que se vió rápidamente sustituido por el **i386DX** un procesador de 32 bits trabajando a 16 Mhz. con unos 275.000 transistores que le daban una potencia de 4 **MIPS** (***Millones de Instrucciones Por Segundo***). Más adelante salieron versiones más rápidas a 20, 25 y 33 Mhz., este último llegaba a los 11,4 MIPS. El **i386SX** salió después como opción más barata y más lenta pues estaba limitado por un bus de datos externo de 16 bits.

El 386 permitía direccionar hasta 4 GB de RAM, aunque raramente encontraremos placas en las que se pueda instalar más de 128 MB (incluso con los procesadores sucesores al 386).

Procesador Intel Pentium	Indice iCOMP
60 Mhz	510
66 Mhz	567
75 Mhz	610
90 Mhz	735
100 Mhz	815
120 Mhz	1.000
133 Mhz	1.110
150 Mhz	1.176
166 Mhz	1.308

Tabla 2.9: Prestaciones de los chips Pentium a distintas velocidades.

No sólo la empresa Intel fabricaba procesadores, sino que le salieron competidores, como AMD con su procesador Am386DX a 40 Mhz. y Cyrix con otra buena oferta de procesadores.

En Abril de 1989 se presentaba el **486DX** que fue un gran avance pues a igualdad de frecuencia de reloj (33 Mhz.) conseguía muchas más prestaciones que el 386DX. De hecho, llegaba a los 40,7 MIPS e integraba 1,2 millones de transistores, coprocesador matemático y memoria caché de 8 Kbytes. Dos años después surgía el **486SX** básicamente igual que su hermano mayor DX pero sin coprocesador matemático y por tanto más barato, un truco de Intel para aprovechar los procesadores DX a los que les fallaba el *copro* por un fallo de fabricación.

El invento de Intel de duplicación de velocidad supuso el nacimiento del procesador **486DX2** a **66 Mhz.**, consiguiendo que el chip del procesador trabajara a doble velocidad que el resto del sistema consiguiendo unos 54 MIPS. También existen los 486DX2 a 50 Mhz., duplicación del reloj de los que iban a 25 Mhz.

En Marzo de 1993 llegaron los **Pentium** (los que debían ser los 586, a los que Intel le dio un nombre no numérico para evitar que otros fabricantes pudieran presentar chips con igual nombre). Se han sacado versiones del Pentium a 60, 66, 90, 100, 120, 133, 150 Mhz. y 166 (y cuando escribimos estas líneas están preparándose otras versiones más aceleradas). En la tabla 2.9 ofrecemos una comparativa de las prestaciones de las distintas versiones del Pentium. Todos tienen arquitectura *superescalar*, de 64 bits (tamaño de palabra), 16 Kbytes de memoria caché, con una gran densidad de integración (3,3 millones de transistores usando tecnología de 0,6 micras) y funcionando a menor tensión eléctrica de lo habitual, 3,3 voltios, lo que redunda en una disminución del consumo eléctrico y la disipación del calor producido, evitando así sobrecalentamientos de los chips, lo cual es muy perjudicial.

Más adelante salió el **486DX4**, que incluía una triplicación de la velocidad de reloj, consiguiendo llegar a los 99 Mhz. (de los antiguos a 33 Mhz., o llegando a los 75 Mhz. de los antiguos a 25 Mhz.) y tanta caché como el Pentium (16 Kbytes). Su potencia es menor que la del Pentium, pero en determinados casos puede que no haya gran diferencia, ya que para sacarle el máximo partido al Pentium debe usarse con programas y S.O. especialmente diseñados para tal fin, lo cual no es así en la mayoría de los programas comerciales.

Mención aparte merecen los chips **OverDrive** de Intel, con los cuales se puede actualizar el procesador de un ordenador sin tener que cambiar toda la placa. Por ejemplo, existe el chip

Intel OverDrive DX4-100 (con 16 KB de memoria caché), que es compatible con el patillaje de un 486DX convencional, de forma que basta con cambiar un chip por otro para gozar de la triplicación de la velocidad de reloj, llegando a los 99 Mhz. (lo de los 100 es pura campaña comercial). También existen OverDrive para actualización a Pentium, aunque no llegan a igualar las prestaciones de un Pentium *pata negra*.

Hemos contado aquí la historia de los ordenadores compatibles de la familia PC con procesador Intel. Sería injusto no mencionar a otra gran familia de menor difusión, la de **Apple Macintosh** con su familia de procesadores 68000 de **Motorola**. Ordenadores muy potentes y en muchos casos por delante de los de Intel, pero que no gozan, ni han gozado, de tanta popularidad. El último microprocesador de Motorola es el llamado **PowerPC** (con arquitectura RISC), algo más potente que el Pentium pero sin grandes diferencias.

De la alianza entre IBM, Apple y Motorola, surgieron la familia de procesadores **PowerPC** (**P***erformance* ***O****ptimized* ***W****ith* ***E****nhanced* ***RISC*** o rendimiento optimizado con RISC mejorado) de arquitectura RISC (en contraposición a la familia de Intel), en un único chip. En Octubre de 1994 se presentaba el más revolucionario de todos, el PowerPC 620, de 64 bits por registro, 133 Mhz., bus de datos de 128 bit y bus de direcciones de 40 bits (que le permite acceder a 1 terabyte de memoria), 7 millones de transistores, trabajando a 3,3 voltios, dos cachés separadas de 32 KB. y *superescalar* con 6 unidades de ejecución independientes (3 para enteros, una para FPU, otra para carga/almacenamiento y otro para bifurcaciones). Parece claro que el futuro está en arquitecturas **RISC** (*Reduced Instruction Set Computer*, o computador de conjunto reducido de instrucciones), basadas en la idea de que es mejor tener pocas instrucciones simples que sean muy rápidas, que tener muchas instrucciones muy complejas que tardan más en ejecutarse, como las de las arquitecturas **CISC** (*Complex Instruction Set Computer*, o computador de conjunto de instrucciones complejo).

De todas formas tampoco es seguro que las tecnologías RISC dominen el mundo de los microprocesadores. De hecho, Intel y HP están actualmente investigando en su tecnología **VLIW** (*Very Large Instruction Word*, o Palabra de Instrucción Muy Larga), que consiste en que el compilador empaquete en una sola instrucción varias instrucciones que el procesador pueda ejecutar en paralelo. De esta forma, cuando el procesador lee esa instrucción, ejecuta más instrucciones a la vez, ya que el compilador le da hecho el trabajo de ordenar las instrucciones para ejecutarlas paralelamente. Esta tecnología pretende reducir la complejidad del hardware y poder aumentar así la velocidad de reloj, pero necesitarán compiladores muy complejos. Los primeros frutos del VLIW no aparecerán hasta 1997 o 1998.

Aún no sabemos qué procesador gobernará mayor cota de mercado, pero lo que es seguro es que un pequeño fallo encontrado en el Pentium de las primeras versiones (ya corregido) ha hecho plantearse a muchos sobre el futuro de su plataforma. Por curiosidad diremos que el fallo consistía en que en una división, cuando se daban unas determinadas condiciones muy precisas y poco normales, daba un resultado erróneo entre el cuarto y décimo dígito decimal.

Fuera de Intel y Motorola también hay vida, es decir, existen más fabricantes de procesadores y la competencia entre ellos es cada día más inesperada.

2.5.3.2 Siguiente generación.

La siguiente generación de microprocesadores viene encabezada por el **Pentium Pro** de Intel (inicialmente llamado P6) a 150 Mhz y trabajando a sólo 1,5 V, anunciado el 13 de Noviembre de 1995. Ya existen versiones a 180 y 200 Mhz. Se espera que para mitad de 1996 llegue a

los 231 Mhz. Su rendimiento estimado es de 220 SPECint92 y 215 SPECfp92. Integra caché secundaria de 256KB (que llegará hasta los 512 KB en otras versiones), que se comunica con la CPU a la misma velocidad del procesador, microarquitectura superescalar con reordenación de instrucciones, superpipelines incluidos para permitir altas velocidades de reloj. Los problemas de este procesador no son pocos, en principio: Demasiado consumo de energía y disipación para ordenadores portátiles, alto precio de fabricación y lo peor de todo es que ejecuta peor que un Pentium programas con código de 16 bits. Para optimizar la ejecución de código de 32 bits, Intel prescindió de ciertas características que reducen drásticamente el rendimiento en programas de 16 bits (un 22% más lento que un Pentium 133). Aunque son muchas las aplicaciones de 16 bits, para cuando el Pentium Pro sea asequible económicamente (1997) serán muchas las aplicaciones de 32 bits. El Pentium Pro competirá con estaciones de trabajo RISC y servidores, aunque puede ser visto también como un ordenador personal, de gama alta.

AMD (*American MicroDevices*) ha sido una empresa que siempre ha ido detrás (en tiempo y tecnología) de Intel, pero para el verano de 1996 planea sacar su procesador **K5**. La letra K viene de la mítica Kriptonita, sustancia capaz de destruir a Superman -lease Intel-. Su arquitectura será también *superescalar*, como el Pentium de Intel, pero mucho más optimizada, lo que le permitirá ser, según AMD, más de un 30% más rápido. Además, K5 tendrá el doble de caché que el Pentium para instrucciones (16 KB, frente a los 8 KB del Pentium). El problema lo tiene en la escasa velocidad conseguida (en principio sólo 75 Mhz.). Su rendimiento estimado está entre 109 y 115 SPECint92.

Con un núcleo RISC, el K5 divide las largas instrucciones CISC (que le hacen compatible x86) en operaciones RISC (R-ops) que se ejecutan independientemente. Esta tecnología híbrida CISC/RISC le hace gozar de algunas de las ventajas RISC, sin renunciar a la compatibilidad CISC de la familia x86. La próxima familia de procesadores Intel (proyecto P6) también gozará de esta mezcla de tecnologías. Por su parte AMD, también ha anunciado su futuro procesador K6. Que gane el mejor.

Escapa de las pretensiones de estas líneas dar una detallada relación de características del K5, pero es interesante saber que, igual que el Pentium, tiene dos pipelines de enteros (puede ejecutar dos operaciones con enteros, a la vez). Pero además, el K5 también puede ejecutar a la vez, una instrucción en coma flotante (con reales), una instrucción de carga/almacenamiento y una instrucción de bifurcación, cosas que no puede efectuar el Pentium. El Pentium le gana en la **FPU** (*Floating Point Unit*, Unidad de Coma Flotante o coprocesador matemático), pero las operaciones en coma flotante no son muy importantes en el software normal.

Otro competidor es el procesador **T5** de **MIPS Technologies**. En este caso la T viene de Terminator -una clara referencia a la película *Terminator 2: el Juicio Final* (1991) ganadora de un Oscar por sus efectos especiales, creados con WorkStations Silicon Graphics basadas en procesadores Mips-. La base del T5 es, una tecnología *superescalar* RISC, de quinta generación, similar a la del Mips R8000, presentado a principios de 1994. El T5, con su reloj interno de 200 Mhz. y sus más de 6 millones de transistores, se ha diseñado para que funcione igual de bien bajo Windows NT que como lo hace en estaciones de trabajo bajo UNIX o en servidores multiprocesador para bases de datos transaccionales. Se trata de un chip muy tolerante a fallos, lo que le convierte en la base ideal para servidores de bases de datos. Este procesador de Mips es claramente **Superescalar**, es decir, es totalmente paralelo (en cada ciclo, T5 puede cargar 4 instrucciones de 32 bits de su caché de instrucciones de 32 KB).

No hay que olvidar que Mips, con su R2000, fue la primera empresa que comercializó un chip RISC, en 1985, y uno de los pioneros en tecnología RISC fue uno de los fundadores de

Mips, J.L. Hennessy.

Otra técnica del T5 es la llamada **ejecución desordenada**, que consiste en ordenar las instrucciones de forma que se aprovechen mejor los recursos del microprocesador. Es muy importante la velocidad de reloj, la anchura de los buses, el número de pipelines... pero con todo esto, se llega un momento en el que algunas unidades funcionales se paran cuando tienen poco trabajo. Los compiladores optimizados intentan solucionar este problema reordenando las instrucciones de forma que se puedan aprovechar al máximo los requisitos de cada CPU, pero la ejecución desordenada constituye una sustitución parcial de esta compilación optimizada, ya que encarga este ordenamiento de las instrucciones a la CPU (no al compilador o programador en lenguaje ensamblador). O sea, la CPU se encarga de reordenar las instrucciones de forma que concuerden con los recursos disponibles en cada momento, de forma dinámica durante la ejecución y no de forma estática durante la compilación (como hacen los compiladores optimizados).

Intel tiene aún más competencia: **Cyrix** tiene su procesador **Cx6X86** de 120 Mhz., con prestaciones similares a un Pentium/150, y **NexGen** desearía comercializar su **Nx686** en 1996. Además **SUN** ha anunciado su **UltraSPARC-II** para mediados de 1996, ideal para el tratamiento de imagen y vídeo (estaciones multimedia). Estará disponible con cerca de 300 Mhz. y unos 400 SPECint92.

DEC tiene a su chip prodigioso **Alpha 21164**, al que le sucederá el **Alpha 21164A** uno de los chips más poderosos de todos los tiempos. Con sus 9,3 millones de transistores y sus más de 300 Mhz. previstos llegará a los 500 SPECint92 y 750 SPECfp92. Es un procesador ideal para estaciones de trabajo científicas y de ingeniería y para servidores Windows NT y UNIX. Lo peor es el precio: Si el 21164 es el más caro del mercado (unos 3000 dólares), seguro que más lo será el 21164A. Para 1997 se esperan conseguir más de 1000 SPECint92.

La empresa Cray Research ha presentado un innovador superordenador, el T3E, que puede albergar entre 16 y 2048 procesadores DEC Alpha.

No hemos pretendido con estos párrafos dar una detallada lista de procesadores y sus características, sino más bien dar una ligera idea de la complejidad del mercado de procesadores, en el que ya se confunden ordenadores llamados personales con *WorkStations* (estaciones de trabajo) y servidores.

El futuro no es fácil de predecir pues los que nos dedicamos a la informática no dejamos de sorprendernos día a día. Pero lo que es seguro es que en un futuro cercano predominarán dos palabras: **conectividad y portabilidad**. **Conectividad**, para que se puedan conectar y comunicar máquinas de distintas marcas y modelos (Sistemas Abiertos) y **portabilidad**, para que cualquier programa funcione en cualquier máquina con cualquier S.O. (Sistema Operativo).

La conectividad es tan importante que ya deja de tener sentido un ordenador aislado del mundo exterior. En un futuro cercano todo ordenador estará integrado en una Red de Comunicación que conecta ordenadores de todo el mundo, por ejemplo la red internacional **InterNet** (ver capítulo sobre Redes de Ordenadores en este mismo libro).

Lo mínimo exigible a un ordenador personal de uso doméstico es un **MODEM** con el que conectarlo, a través de la red telefónica a la red internacional **FidoNet** o a **InterNet**. Por el precio de un Modem... ya no hay excusa.

A través de estas redes internacionales de ordenadores podemos comunicarnos con gente de todo el planeta, conseguir programas para los más diversos fines y resolver nuestras dudas sobre cualquier tema por alejado que parezca de la informática, pero esto, es ya otra historia.

2.6 Ergonomía y ecología informática.

Desde 1992, la palabra ergonomía está incluida en el diccionario de la Real Academia Española (la que *"limpia, fija y da esplendor"*), que la define como "Estudio de datos biológicos y tecnológicos aplicados a problemas de mutua adaptación entre el hombre y la máquina".

Referidos a la informática, la **ergonomía** es la ciencia que estudia la comodidad y la salud en el trabajo con un ordenador y la **ecología** la ciencia que estudia cómo aprovechar mejor los recursos de forma que un ordenador contamine y consuma lo menos posible.

La Presidencia del Consejo Superior de Informática del Ministerio para las Administraciones Públicas (MAP), publica en el BOE 252, del 21 de Octubre de 1994, en la página 32950, la resolución 23082, por la que dicho organismo ha resuelto adoptar las recomendaciones y medidas emanadas del modelo general de adopción de pautas del **proyecto MABER**, proyecto para el establecimiento de pautas medioambientales y ergonómicas en la adquisición de bienes y servicios en tecnologías de la información.

Aquí daremos una serie de normas elementales, pero muy importantes, para facilitarnos el trabajo con estas cada vez más frecuentes máquinas. A la hora de comprar un equipo informático es muy importante tener siempre presente los siguientes consejos, que nos facilitarán tanto el trabajo diario como la modificación de su configuración.

La ya famosa ley de las **Tres Erres** se debe aplicar siempre y no sólo en material informático, desde un frigorífico a una bombilla:

- **REDUCIR:** Consumir lo menos posible reduciendo el gasto de recursos no renovables (papel, envoltorios, plásticos...). Sobre todo reducir el consumo de energía.
- **REUTILIZAR:** Los objetos de *usar y tirar* sólo aumentan la basura y se despilfarran recursos.
- **RECICLAR:** Si no se puede reducir el consumo de algo, ni reutilizarlo, que al menos se puedan reciclar sus materiales para producir otros productos.

2.6.1 Diseño.

Primero, podríamos decir que los **ordenadores** deben estar diseñados de forma que faciliten el trabajo para el que están pensados. O sea, que tengan a mano los interruptores, las disqueteras... y todo lo que sea de uso frecuente. También interiormente deben estar diseñados de forma que facilite su manipulación, para cambiar, añadir o quitar tarjetas controladoras, reparar averías, incluir un disco duro...

Igualmente, las **impresoras** deben ser de fácil cambio de papel y otros fungibles (tóner...), así como fácil de desatascar caso de que se atasque el papel. El **teclado** debe tener teclas de 19 mm. y tener dos alturas regulables con unas patas en su parte inferior, y si lo usamos mucho, es recomendable añadirle un reposa-muñecas. Ya hay teclados ergonómicos que se adaptan mejor a la forma natural de las manos y cansan menos, pero sólo son necesarios si somos mecanógrafos de profesión. También existen teclados con una distribución de letras más cómoda para según qué idioma. Y si copiamos a menudo, debemos situar el original de forma que no tengamos que variar el ángulo de inclinación de la cabeza para mirar al papel original y al monitor (usar un atril). Una Comisión Europea para la Salud y la Seguridad, dictó unos requisitos mínimos para garantizar un puesto de trabajo digno a las personas que se pasan el día delante del **monitor**: caracteres en pantalla bien definidos, sin parpapadeos

Figura 2.32: El ordenador no muerde pero hay que aprender a usarlo correctamente.

ni reflejos, con brillo y contraste ajustable y monitor con una peana para ajustar fácilmente su orientación. Un **ratón** también puede ser más ergonómico y cómodo de usar. Los hay que se adaptan mejor a la forma de la mano, pero... depende del tamaño de cada mano. También los hay sin cables, que tienen más libertad, pero son más caros. También incluimos aquí el diseño de **programas** (no sólo de hardware), ya que es muy importante que estén bien diseñados para facilitar su uso. Los programas deben ser fáciles de usar, intuitivos, agradables al uso, con mensajes de estado en cada momento, con opción de ayuda para cada opción... y siguiendo siempre un estándar general (si existe) o propio para nuestro programa (si no existe).

2.6.2 Consumo eléctrico.

Un PC de configuración media gasta unos 300 watios. Se debe conseguir que cuando no se esté usando se desconecte automáticamente usando mucha menos potencia, tanto el ordenador como el monitor. Para ello también ha dado unas normas la asociación VESA (que permite ahorrar sólo 8 w cuando no se usa el ordenador). La **EPA** (*Environmental Protection Agency*, Agencia norteamericana de Protección del Medio Ambiente) fija unos requisitos que deberían cumplir todos los fabricantes. Entre ellos dice que cuando el equipo no se está usando durante un período de tiempo, el máximo consumo debería ser 60 watios (30 por el monitor y otros 30 por el propio ordenador). La llamada compatibilidad **Energy Star** permite reducir la factura de la luz considerablemente (permitiendo así mandar faxes, realizar llamadas u otro tipo de tareas por la noche, sin que nos asuste la factura de electricidad).

La serie de **procesadores SL** de Intel ha demostrado que se pueden construir procesadores de bajo consumo. Lo que hacen, entre otras cosas, es reducir el voltaje de funcionamiento, llegando a funcionar a **3,3 V**, en vez de a los 5 V. habituales. Además, así se reduce el calentamiento del chip y se puede aumentar en velocidad. Los nuevos chips comienzan a trabajar a 1,5 V. Existen también **discos duros** que dejan de girar cuando no los usamos, ahorrando mucho consumo eléctrico.

Esto es un problema más serio de lo que pueda parecer. No sólo importa pagar poco a fin de mes en la factura de electricidad, sino que ese ahorro se traduce, multiplicado por los meses de uso y por todos los ordenadores, en muchos millones de toneladas de dióxido de carbono (CO_2)que se dejan de emitir a la atmósfera.

No olvidemos que aunque la electricidad es una fuente de energía limpia, no suelen ser límpias las formas de obtenerla. Por esto, se deben de potenciar las energías renovables (eólica, solar, mareas, geotérmica, biomasa...) frente a las gravemente contaminantes (atómica, térmica, combustión de residuos fósiles...), sobre todo si tenemos en cuenta el tremendo aumento de CO_2 que ha sufrido la atmósfera en los últimos años, lo que redunda en un calentamiento global, efecto invernadero, cambio climático, sequía...

Por eso, si nuestro ordenador va a estar encendido todo el día y no va a usarse todo el tiempo, esa opción ecológica es casi una obligación. Si nuestro ordenador no tiene esa opción, no olvidemos que conviene apagarlo si vamos a estar un buen rato sin usarlo, y si no queremos perder los datos, al menos, apagar el monitor.

Para **portátiles** resulta casi imprescindible que sean de bajo consumo para que su independencia supere las 3 horas.

2.6.3 Consumo de materiales.

Procurar no imprimir *a ver que sale* (se consume mucho papel), sino saber qué debe salir. Usar el papel por ambas caras, al menos si es en sucio, y por supuesto, siempre **papel reciclado**.

Usar la opción de borrador para gastar menos tinta si la copia no es definitiva. Hay que tener en cuenta que el tóner de las impresoras (LASER e inyección de tinta) es muy tóxico y muy contaminante. Por eso, los cartuchos vacíos no se deben tirar a la basura, sino que se deben reciclar. Hay empresas que se dedican a llenar de nuevo los cartuchos vacíos, que además sale más barato que comprar cartuchos nuevos. Igualmente se hace con las cintas de impresoras matriciales.

Hay que tener en cuenta que es más caro el cartucho que contiene el tóner que el tóner mismo, y que si acaba en un vertedero liberará gases tóxicos y cancerígenos, por lo que su reciclado es obligatorio desde el punto de vista moral y económico. El ahorro que produce al usuario es superior al 40%.

Entre otras medidas, el anteriormente citado **proyecto MABER**, concreta en el reciclado de los cartuchos de tóner. Esto significa que todas las oficinas estatales tendrán que almacenar y establecer un sistema de recogida para evitar que se tiren los cartuchos a la basura. Además, esto será obligatorio para todos los ciudadanos comunitarios en breve, según una directiva comunitaria.

Recientemente, la marca de impresoras Kyocera, ha sacado sus modelos con la característica **Ecosys**, en las que el cartucho no necesita reciclarse, ya que sólo es un continente de plástico, y que además ahorran un montón de consumibles.

Ya hay empresas (como Siemens Nixdorf, Digital, IBM o Philips) que recicla ordenadores viejos recuperando hierro y otros metales, plásticos, cables... con el consiguiente ahorro para su empresa y el planeta.

Otro punto que deben tener en cuenta los fabricantes es no usar el material plástico PVC en sus ordenadores, ya que contamina muchísimo en su fabricación. Es mucho más ecológico usar PET.

Ya hay muchas empresas que elaboran todos los manuales y el cartón de los embalajes en papel reciclado. Recordemos que de media, cada tonelada de papel equivale a más de 15

árboles.

Las baterías de los portátiles suelen ser muy perjudiciales para el medio ambiente (sobre todo las de cadmio). Deben ser de materiales no contaminantes, reciclables y por supuesto NUNCA tirarlas a la basura.

2.6.4 Ruido.

El ruido es otro factor importante, sobre todo en impresoras, aunque no se puede olvidar el ruido del ventilador de la fuente de alimentación del ordenador. De hecho, algunos ordenadores llevan en la carcasa una placa de un material que absorbe la radiación y el ruido (para comprobarlo basta con quitar la carcasa y comprobar el ruido). Existen las llamadas **fuentes silenciosas** (hacen poco ruido) y las **fuentes inteligentes** (que sólo funcionan cuando realmente se necesita). En cuanto a las **impresoras**, lo mejor es no tenerlas cerca y si son compartidas mejor, ya que puede que estén más alejadas y saldrán más baratas. Se aconseja que no superen los 60 decibelios (Db). De todas formas esto es importante, sobretodo, en impresoras matriciales, las cuales parece que van desapareciendo por su ruido y lentitud. El resto de tecnologías de impresión no suelen superar los 55 Db.

El ruido, además de provocar distracciones y repercutir en el rendimiento laboral, también influye en la salud: Aumento del número de pulsaciones, variación del ritmo respiratorio, presión arterial, tensión muscular, estres... Por eso, evite los lugares ruidosos e intente no producir ruido.

2.6.5 Características de los dispositivos.

Los **monitores** deben ser de baja radiación para evitar el daño en la vista, aunque realmente no son tan perjudiciales como a veces los pintan. Un monitor de baja radiación y/o un buen filtro serán suficientes para preservar nuestra salud ocular.

Además, deben ser **no entrelazado**, para evitar el parpadeo que ocasiona mucho cansancio.

Las **Impresoras** deben usar cartuchos de tinta/tóner reciclables y en caso de las láser que no emitan demasiado ozono (O_3, gas que produce mareos y jaquecas si se respira frecuentemente).

2.6.6 Buen uso.

No hay que olvidar que además se debe hacer un buen uso de todos los dispositivos. Por ejemplo, se debe hacer un descanso de al menos 10 ó 15 minutos por cada hora de trabajo delante de la pantalla de un ordenador.

El diseño del puesto de trabajo también es muy importante: Debemos estar en una silla cómoda y que nos permita tener la espalda recta, apollada en el respaldo. La parte superior del monitor a la altura de los ojos y el teclado a la altura de las manos, doblando los codos unos 90° (ver figura 2.33).

Si hay ventanas, nunca situarlas delante (nos deslumbrarán) o detrás (producirán reflejos en el monitor). Lo mejor es al lado izquierdo si somos diestros y al derecho si somos zurdos.

La lectura de documentos en papel requiere un alto nivel de iluminación puesto que presenta un contraste positivo (caracteres oscuros en fondo claro). Sin embargo, la lectura en pantalla exige una iluminación más baja y debe ser con caracteres claros sobre fondo oscuro.

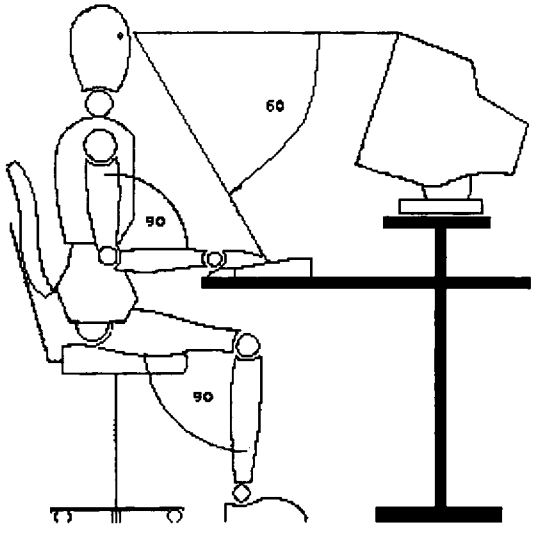


Figura 2.33: Posición ideal al trabajar con un ordenador.

El brillo del monitor se debe regular para que se vea correctamente, pero en general es mejor dar poco brillo a la pantalla.

El lugar de trabajo debe ser un sitio bien ventilado y preferiblemente con luz natural. También es importante la **temperatura** ambiente del lugar de trabajo, que debe estar entre los 19 y los 24 grados. No olvidemos que todos los aparatos eléctricos, en general, producen calor. La **humedad** relativa, además, debe estar entre el 40 y el 60%.

Capítulo 3

Representando la Información

En este capítulo se pretende presentar la forma en que se representa la información dentro de un ordenador. También veremos las operaciones básicas que se realizan con los datos como sumas, restas, operaciones lógicas, etc.

Un ordenador es una máquina que procesa información, y esa información tiene una forma de almacenarse dentro del ordenador. En el ordenador se almacenarán tanto los datos como las instruciones. Nosotros sólo nos preocuparemos en este capítulo de como se representa la información no de como debe almacenarse de forma física.

Para empezar, y para que podamos comunicarnos con el ordenador y éste con nosotros tendremos que ponernos de mutuo acuerdo para utilizar un conjunto de símbolos común. Este conjunto de símbolos se divide en los siguientes grupos:

Caracteres alfabéticos Son las letras tanto mayúsculas como minúsculas del alfabeto inglés:

a,b,c,...,x,y,z,A,B,C,...,X,Y,Z

Caracteres numéricos Estan compuestos por las diez cifras decimáles:

0,1,2,3,4,5,6,7,8,9

Caracteres especiales Son todos los símbolos no incluidos en los grupos anteriores. Este grupo de caracteres se utiliza para que se pueda presentar cualquier tipo de información necesaria. Además algunos caracteres incluidos en este grupo pueden variar de unos países a otros dependiendo de las raíces culturales específicas de cada país como la ñ en España, etc.

) (* / + . ; , ñ Ñ ¿ ? ¡ ! " # .....

Caracteres de control Estos caracteres también se llaman *Caracteres no Imprimibles*, es decir que no se pueden visualizar directamente en pantalla o por una impresora. Estos caracteres son utilizados por el ordenador para realizar algunas tareas necesarias de control, tales como: pasar de línea, sincronización en una comunicación, emitir un pitido, borrar un carácter de la pantalla, etc.

De esta forma ya sabemos como se divide el conjunto de caracteres que vamos a poder utilizar con un ordenador, ahora nos queda ver como el ordenador resuelve el problema de representarlo para que pueda realizar las operaciones que se esperan del él.

Actualmente la tecnología que se ha utilizado para el diseño de ordenadores se basa en dos estados posibles, es decir, un ordenador trabaja sólo con 0's y 1's. Lo que a continuación vamos a ver es como se representan los distintos tipos de datos con tan sólo dos estados posibles, es decir, veremos cuales serán las combinaciones necesarias de 0's y 1's para poder hacer que el ordenador trabaje de una forma eficiente y fiable.

3.1 Sistemas de Numeración Usuales en Informática

Como el ordenador trabaja con dos estados posibles, 0 ó 1, diremos que el ordenador trabaja en base dos, que para abreviar denominaremos **binario natural**, y cuando no se preste a confusión lo denominaremos **binario**, si más. Cualquier número que se represente en el ordenador sera una secuencia de 0's y 1's. Este sistema de numeración es el que siempre utilizará el ordenador, ahora presentaremos otros tipos de numeración que se utilizan en informática:

3.1.1 Base n

Cuando nosotros representamos los números utilizamos diez cifras distintas, esto es, utizamos un sistema de numeración en base 10. Cuando el ordenador representa los números de manera interna sólo utiliza dos cifras, utiliza una base 2. En general si nosotros utilizamos n cifras distintas para representar cualquier número se dice que nosotros estamos trabajando en base n.

Una secuencia de cifras de un número en base n viene a expresar el peso que le correspone a cada potencia de la base, es decir, cada cifra indicará el peso que tiene cada potencia de la base empezando por cero:

$$c_k * n^k + c_{k-1} * n^{k-1} + ... + c_1 * n^1 + c_0 * n^0$$

siendo c_k una cifra en base n.

Ejemplo:

Si tenemos el número 3234 y nos dice que esta en base 10 podemos verlo según lo expuesto anteriormente como:

$$3 * 10^3 + 2 * 10^2 + 3 * 10^1 + 4 * 10^0$$

Si por el contrario nos dicen que se encuentra en base 5 lo que tenemos es:

$$3 * 5^3 + 2 * 5^2 + 3 * 5^1 + 4 * 5^0$$

Por lo que hemos visto hasta el momento basta con sumar la expresiónes anteriores y obtenemos la representación en base 10 (decimal) de cualquier número expresado en base n. De la misma forma con las expresiones anteriores se desprende que basta con dividir por m sucesivamente hasta que ya no se pueda dividir más y las cifras serán el último cociente y los restos empezando por el último. Con esto se consigue pasar de un número en base n a base m.

A continuación lo que se presentan son aquellas bases más habituales a la hora de trabajar en informática.

3.1.2 Decimal

Es el sistema de numeración que estamos acostumbrados a utilizar desde el colegio cuando empezó nuestra educación. Este sistema tambien se le llama sistema en base 10, se utilizan diez cifras diferentes las cuales son las siguientes:

0,1,2,3,4,5,6,7,8,9

Por ser este sistema el más utilizado por nosotros es el método el cual quisieramos utilizar para poder dar datos de tipo numérico al ordenador. Y éste debiera responder con el mismo sistema de numeración para que nosotros interpretemos los datos de forma inmediata.

3.1.2.1 Paso de Decimal a Binario

Nuestro próximo paso consiste en poder identificar cualquier número en la representación usual para nosotros, es decir en decimal, a la representación usual del ordenador, es decir a binario.

Para realizar esta tarea efectuaremos los siguientes pasos:

1. Lo primero que se necesita es un número en el sistema decimal.
2. Dividimos el número entre 2 y guardamos el resto y el cociente.
3. Tomamos el cociente anterior como nuestro nuevo número y si se puede dividir pasamos al paso 2. En caso contrario sólo falta escribir adecuadamente los restos almacenados y el último cociente.
4. Para formar el número en binaro basta con poner primero el último cociente obtenido de las sucesivas divisiones y seguidamente todos los restos empezando por el último.

Ejemplo:
Vamos a pasar el número 35 a su representación binaria:

1. Dividimos 35 entre 2. El resto R1=1 y el cociente=17.
2. Como se puede seguir dividiento, dividimos 17 entre 2. Resto R2=1, cociente=8.
3. Dividimos 8 entre 2. Resto R3=0, cociente=4.
4. Dividimos 4 entre 2. Resto R4=0, cociente=2.
5. Dividimos 2 entre 2. Resto R5=0, cociente=1.
6. Como ya no podemos dividir más sólo queda componer el número con el último cociente y los restos tomados en orden inverso, esto es, cociente-R5-R4-R3-R2-R1. El número en binario es: 100011.

Ejercicios propuestos
Pasar a binario los siguientes números decimales: 127, 256, 1024, 666.

3.1.2.2 Paso de Binario a Decimal

Ahora nuesto problema es el contrario, es decir, partimos de un número que viene representado en binario (en una secuencia de 0's y 1's) y queremos obtener el número en decimal que lo representa. Actuando de la misma manera que antes debemos seguir los siguientes pasos:

1. Partimos de un número en representación binaria.
2. Numeramos las cifras de la representación en binario de derecha a izquierda empezando por 0.
3. Ahora, de cada cifra obtenemos la potencia utilizando como exponente su posición.
4. Por último sumamos aquellas potencias de dos cuya cifra binaria sea un 1. Con esto tenemos el numero decimal correspondiente.

Ejemplo:
Vamos a obtener el número decimal correspondiente al siguiente número binario 10111011:

1. Numeramos las cifras de derecha a izquierda empezando por 0:

NÚMERO	1	0	1	1	1	0	1	1
POSICIÓN	7	6	5	4	3	2	1	0

2. Ponemos las potencias de dos correspondientes a la anterior numeración

NÚMERO	1	0	1	1	1	0	1	1
POSICIÓN	7	6	5	4	3	2	1	0
$2^{posicion}$	128	64	32	16	8	4	2	1

3. Por último sumamos aquellas potencias que se correspondan con una cifra distinta de 0: $128 + 32 + 16 + 8 + 2 + 1 = 187$

Ejercicios propuestos
Pasar de binario a decimal los siguientes números: 1001001, 10000000101, 11111111, 1010101010.

3.1.3 Octal

Este sistema utiliza base 8, es decir utiliza ocho cifras diferentes para expresar cualquier número. Las cifras a utilizar son las siguientes:

0,1,2,3,4,5,6,7

El sistema de numeración octal es uno de los sistemas de numeración intermedios, es decir, por ser la base potencia de dos tendrá una sencilla conversión de binario a octal y viceversa. También tiene como ventaja que los números representados en octal són mucho mas pequeños que los escritos en binario por lo que són más faciles de utilizar para las personas.

OCTAL	BINARIO
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Tabla 3.1: Correspondencia entre las cifras octales y su representación binaria

3.1.3.1 Paso de Octal a Binario

Como ya he dicho anteriormente al ser la base una potencia de dos el pasar de octal a binario es inmediato. Primero veremos una tabla donde tendremos el equivalente entre las cifras octales y las binarias:

Como se ha hecho en el epígrafe anterior voy a detallar los pasos necesarios para pasar cualquier numero octal a su correspondiente representación binaria:

1. Tomamos cualquier número en octal.
2. Debajo de cada cifra octal ponemos su correspondiente transformación a binario con las mismas cifras con que aparece en la tabla 3.1 y ya tenemos su representación en binario.

Ejemplo:

Vamos a tranformar el número octal 7563 en su correspondiente binaria por lo que vamos a empezar:

1. Colocamos la correspondencia binaria de cada una de las cifras octales que tenemos:

OCTAL	7	5	6	3
BINARIO	111	101	110	011

2. Con lo que el número binario correspondiente es el 111101110011.

Ejercicios propuestos

Pasar los siguientes números de su representación octal a su correspondiente representación binaria: 10, 210, 3444.

3.1.3.2 Paso de Binario a Octal

Ahora lo que nosotros tenemos es un número en representación binaria y queremos obtener cual es su representación octal. Procediendo de la misma manera que en las veces anteriores:

1. Tenemos cualquier número escrito en binario.
2. Vamos agrupando las cifras binarias de tres en tres empezando por la derecha.

3. Si el último grupo que se formara, es decir, el que se encuentra más a la izquierda no tuviera tres cifras se le añaden tantos ceros como sea necesario.

4. Cada grupo de tres cifras bianrias lo pasamos a su correspondiente cifra octal segun la tabla 3.1.

5. Con eso ya tenemos la representación octal del numero.

Ejemplo:
Vamos a convertir el número binario 1000101100 a su correspondientes representación octal:

1. Agrupamos en grupos de tres cifras binarias empezando por la derecha:

BINARIO	1	000	101	100

2. Como el grupo más a la izquierda sólo tiene una cifra binaria se le añaden dos ceros para completarlo:

BINARIO	**00**1	000	101	100

3. Tomamos la representación octal de cada uno de los grupos de tres cifras binarias.

BINARIO	**00**1	000	101	100
OCTAL	1	0	5	4

4. Con esto ya hemos conseguido el número octal que es 1054

Ejercicicios propuestos
Pasar de binario a octal los siguientes números: 1001, 1100001, 0011, 10000110101.

3.1.3.3 Paso de Decimal a Octal

Si nosotros quisiéramos pasar un número decimal a octal podemos hacerlo de dos formas:

1. Utilizamos el mismo método que para pasar de decimal a binario salvo que hay que dividir por 8 en vez de por 2.

2. Pasamos primero de decimal a binario y luego de binario a octal.

3.1.3.4 Paso de Octal a Decimal

Ahora si lo que queremos es pasar de octal a decimal, también podemos hacerlo de dos formas:

1. Hacemos lo mismo que cuando pasábamos de binario a decimal, con la excepción que tomamos potencias de 8.

2. Pasamos primero a binario y luego de binario a decimal.

Ejercicios propuestos
Pasar de decimal a octal los siguiente números: 128, 255, 1024.
Pasar de octal a decimal los siguientes números: 77, 666, 13.

HEXADECIMAL	BINARIO	HEXADECIMAL	BINARIO
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Tabla 3.2: Correspondencia entre las cifras hexadecimales y su representación binaria

3.1.4 Hexadecimal

Este es otro sistema de numeración cuya base también es una potencia de 2. En este sistema de numeración consta de 16 cifras distintas. Estas cifras son:

0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Tiene las mismas ventajas que el sistema de numeración octal y además por tener un número mayor de cifras para poder representar cualquier número implica que la cantidad de cifras necesarias para representar cualquier número sea la menor de todos los sistemas de numeración vistos hasta el momento. Éste es el sistema de numeración más usado en informática junto con el decimal.

3.1.4.1 Paso de Hexadecimal a Binario

Como en el caso anterior el paso entre hexadecimal y binario es inmediato y como en el caso anterior empezaremos mostrando en una tabla las equivalencias entre las cifras hexadecimales y sus correspondientes binarias:

Procediendo como en las veces anteriores vamos a presentar los pasos necesarios para pasar cualquier número en hexadecimal a su correspondiente binario:

1. Primero partimos de un número en su representación hexadecimal.
2. Debajo de cada cifra hexadecimal ponemos su correspondiente transformación a binario con las mismas cifras con que aparece en la tabla 3.2 y ya tenemos su representación en binario.

Ejemplo:

Vamos a pasar el número hexadecimal A39F a su correspondiente representación binaria siguiendo los pasos descritos anteriormente:

1. Colocamos la correspondencia binaria de cada una de las cifras hexadecimales según la tabla que presentamos anteriormente:

HEXADECIMAL	A	3	9	f
BINARIO	1010	0011	1001	1111

2. Una vez colocados los correspondientes unos y ceros ya tenemos la representación binaria del número, con lo que el número es: 1010001110011111

Ejercicios propuestos
Pasar de hexadecimal a binario los siguientes números: A001, BCF, 100.

3.1.4.2 Paso de Binario a Hexadecimal

Ahora lo que nosotros tenemos es una sucesión de ceros y de unos que forman un número binario concreto y nosotros queremos obtener su correspondiente representación hexadecimal. Procediendo como hasta el momento:

1. Tenemos un número cualesquiera es su representación binaria.
2. Vamos agrupando las cifras binarias de cuatro en cuatro empezando por la derecha.
3. Si el último grupo que se formara, es decir, el que se encuentra más a la izquierda no tuviera cuatro cifras se le añaden tantos ceros como sean necesarios.
4. Se sustituye cada grupo de cifras binarias por el correspondiente dígito exadecimal utilizando la tabla 3.2.
5. Con esto ya tenemos la representación en hexadecimal del número.

Ejemplo:
Vamos a convertir el número binario 100101001011000101 a su correspondiente representación hexadecinmal:

1. Agrupamos las cifras binarias en grupos de cuatro empezando por la derecha:

BINARIO	10	0101	0010	1100	0101

2. Como el grupo más a la izquierda sólo tiene dos cifras binarias se le añaden dos ceros para completarlo y así llegue a tener las cuatro que se necesitan:

BINARIO	**00**10	0101	0010	1100	0101

3. Tomamos la representación hexadeimal de cada uno de los grupos de cuatro cifras binarias:

BINARIO	**00**10	0101	0010	1100	0101
HEXADECIMAL	2	5	2	C	5

4. Con esto ya tenemos el correspondiente número hexadecimal el cual es: 252C5.

Ejercicios propuestos
Pasar de binario a hexadecimal los siguientes números: 100, 1101101, 110001101, 101101.

3.1.4.3 Paso de Decimal a Hexadecimal

Si nosotros quisieramos pasar ahora un número en representación decimal a su correspondiente hexadecimal tendíamos dos alternativas:

1. Utilizamos el mismo método que para pasr de decimal a binario salvo que hay que dividir por 16 en vez de por 2.
2. Pasamos primero de decimal a binario y luego de binario a hexadecimal.

3.1.4.4 Paso de Hexadecimal a Decimal

Si lo que deseamos hacer es lo contrario, es decir, queremos pasar un número hexadecimal a su correspondiente representación decimal, también tenemos dos formas de hacerlo:

1. Hacemos lo mismo que cuando pasabamos de binario a decimal, con la excepción que tomamos las potencias de 16
2. Pasamos primero de hexadecimal a binario y luego de binario a decimal.

Ejercicios propuestos
Pasar de decimal a hexadecimal los siguientes números: 12, 100, 666, 1024, 255.
Pasar de hexadecimal a decimal los siguientes números: 10, 255, AF8, 20F.

3.1.5 Código BCD

BCD viene de las siglas, Bynary Coded Decimal, en castellano es decimal codificado en binario. Esta forma de representar un número pretende aprovechar dos cualidades que hemos descrito anteriormente en la presentación de las distintas formas de representar un número:

1. Que se pueda pasar de una representación cualquiera a la binaria de forma inmediata, como pasa en el octal y en hexadecimal, y viceversa.
2. Que nos resulte lo más familiar posible, caso del decimal.

Si integramos estas dos cualidades surge el BCD. Ya no se trata de un cambio de base o paso de una representación a otra de forma matemática simplemente, sino que lo que estamos haciendo es codificar, es decir, a una secuencia de ceros y unos les hacemos corresponder aquello que a nosotros nos interesa más (en este caso las cifras decimales).

Con esta codificación se pretende que utilizando las diez cifras decimales y dado un numero cualquiera decimal poder obtener de forma inmediata una sucesión de ceros y unos para que el ordenador pueda entenderlos, y viceversa. Lo primero que haremos es presentar la equivalencia entre las diez cifras decimales con su correspondiente codificación BCD.

3.1.5.1 Paso de Decimal a BCD

Vamos a seguir el ejemplo de las conversiones anteriores, es decir, vamos a indicar los pasos necesarios para poder pasar de un número decimal a su correspondiente representación BCD.

1. Partimos de un número decimal cualquiera.

DECIMAL	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Tabla 3.3: Codificación de las cifras decimales en BCD

2. Debajo de cada cifra decimal colocamos su correspondencia BCD segun la tabla 3.3.

3. Con eso ya hemos conseguido la representación BCD del número decimal.

Ejemplo:
Vamos a obtener la representación BCD del siguiente número decimal 3276.

1. Colocamos debajo de cada cifra decimal su equivalencia BCD:

DECIMAL	3	2	7	6
BCD	0011	0010	0111	0110

2. La representación del número en BCD es 0011001001110110

Ejercicios propuestos
Pasar los siguientes números en decimal a su correspondencia BCD: 10, 255, 666, 1024.

3.1.5.2 Paso de BCD a Decimal

Los pasos a sguir en este caso son los siguientes:

1. Partimos de cualquier representación válida en BCD, es decir, que si agrupamos los 0's y 1's de derecha a izquierda en grupos de cuatro ninguno de esos grupos representa un número mayor de 1001 o lo que es lo mismo 9.

2. Agrupamos de derecha a izquierda en grupos de cuatro.

3. Debajo de cada grupo ponemos la cifra decimal correspondiente.

4. Con esto ya hemos conseguido la representación decimal que deseábamos.

Ejemplo:
Como lo que necesitamos es una representación válida no podriamos utilizar la siguiente:

1000 1100 0011 0100

pues nos encontramos que uno de los grupos representa un número mayor que 1001 o 9.
Vamos a pasar la representación 1000100100010000 en BCD a su correspondiente decimal.

1. Agrupamos de derecha a izquierda en grupos de cuatro.

BCD	1000	1001	0001	0000
DECIMAL	8	9	1	0

2. La representación decimal que perseguimos es: 8910.

Ejercicios propuestos
Pasar de BCD a decimal los siguientes números: 100110000101, 011010010001, 10, 10111.

3.2 Operaciones básicas

Una vez que ya hemos presentado las distintas representaciones de los números en informática, el siguiente paso a seguir es ver como quedan las operaciones más usuales realizadas en binario, es decir, como son las operaciones con sólo 0's y 1's.

3.2.1 Suma Binaria

La primera operación que vamos a presentar es la suma, primero veremos como se opera con sólo un dígito binario y luego presentaremos un ejemplo con más cifras. La suma con sólo una cifra binaria en juego es:

PRIMER SUMANDO	SEGUNDO SUMANDO	RESULTADO	SUMAR AL SIGUIENTE
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

En el caso de sumar 1+1 el resultado sería 2, pero como estamos en binario sería 10 pero al sólo disponer de una cifra, el 1 que nos sobra debemos sumárselo a la siguiente cifra binaria (acarreo) para no perder información cuando realicemos la suma. Para que todas las operaciones que describamos se realicen de la misma manera, cuando haya que sumar se sumará siempre a la cifra correspondiente del segundo número ya sea sumando como restando.

Ejemplo:
Vamos a sumar los siguientes dos números: 10010110 + 01101100

$$\begin{array}{rr} & 10010110 \\ + & 01101100 \\ \hline & 100000010 \end{array}$$

Ejercicios propuestos
Sumar los siguientes números:
1001 + 10001, 100010011+10110, 11011+11110, 1100111+1100110+10110

3.2.2 Resta Binaria

Esta es otra de las operaciones básicas que vamos a ver, como en el caso anterior primero veremos como se realiza la operación con sólo una cifra binaria y posteriormente presentaremos un ejemplo con más cifras para ilustrar un ejemplo más complicado.

La resta binaria con sólo una cifra es:

MINUENDO	SUSTRAENDO	RESULTADO	SUMAR AL SUSTRAENDO
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

En este caso la operación conflictiva que se nos da es 0-1 lo que daría -1, pero como en binario no tenemos número negativos para saber cual es el número positivo que le corresponde deberíamos hacer la operación de 2-1 = 1 (10 - 1 = 1) y como no podemos perder información debemos sumar a la siguiente cifra del sustraendo un 1 o lo que es lo mismo restar a la siguiente cifra del minuendo un 1.

Ejemplo:

Vamos a restar los siguientes dos números: 10010110 - 01101100

$$\begin{array}{rr} & 10010110 \\ - & 01101100 \\ \hline & 00101010 \end{array}$$

Ejercicios propuestos

Restar los siguientes números: 1001-11, 100010010-11001, 11000111-1010.

3.2.3 Multiplicación Binaria

Esta operación en binario se realiza de una forma muy sencilla pues sólo se multiplica por 0 o por 1, sólo se debe poner atención a la hora de sumar pues la suma puede ser un poco laboriosa por la gran cantidad de ellas que pueden llegar a realizarse. Directamente pondré un ejemplo que ilustra como se realiza la multiplicación.

Ejemplo:

Vamos a multiplicar los números siguientes: 10010 x 101

$$\begin{array}{rrrrrrrr} & & & 1 & 0 & 0 & 1 & 0 \\ x & & & & & 1 & 0 & 1 \\ \hline & & & 1 & 0 & 0 & 1 & 0 \\ + & 1 & 0 & 0 & 1 & 0 & & \\ \hline & 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{array}$$

Ejercicios propuestos

Multiplicar los siguientes números: 10110110x10110, 11000111001x111, 1001001x0101.

Como nos encontramos en base dos cuando nosotros multiplicamos cualquier número en binario por una potencia de dos sólo hay que añadir tantos ceros por la derecha como veces multipliquemos por dos, a esa operación en informática se le conoce como desplazamiento a la izquierda de un número.

3.2.4 División Binaria

Al igual que la multiplicación, la división es una operación simple pues cuando queremos ver el dígito correspondiente al cociente sólo tenemos que mirar si se trata de un 0 o un 1, es decir, si el número que queremos dividir *cabe* ponemos un 1 en caso contrario ponemos un cero y bajamos la siguiente cifra binaria del dividendo. Se trata de dividir como nos enseñaron en el colegio. Para ilustrar el problema pondremos un ejemplo.

Ejemplo:

Vamos a dividir los siguientes números binarios: 10010101 : 11

```
1 0 0 1 0 1 0 1 | 1 1
- 1 1           |-------
-------         | 1 1 0 0 0 1
0 0 1 1         |
  - 1 1         |
  -------       |
    0 0 0 1 0 1 |
      - 1 1     |
      ----------|
        0 1 0   |
```

Ejercicios propuestos

Dividir los siguientes números: 1000010001:1101, 1110101110:10011, 100111101101:1101101.

Al igual que con la multiplicación si dividimos por una potencia de dos sólo hay que desplazar la coma decimal hacia la izquierda tantas posiciones como veces dividimos por dos. Si consideremos sólo los números enteros dividir por una potencia de dos es igual que desplazar hacia la derecha el número, es decir, introducimos ceros por la izquierda y despreciamos las cifras decimales que nos salen.

3.2.5 Complemento a 1

Esta es una operación muy simple, lo que hace es cambiar los 1's por 0's y los 0's por 1's. La idea general de esta operación es ver lo que le falta a un número de n dígitos binarios para alcanzar al número de n dígitos binarios siendo todos ellos 1.

Ejemplo:

Si tenemos el número 100110 y queremos obtener su correspondiente complemento a 1 el resultado seía 011001 o lo que es lo mismo:

```
  111111
- 100110
--------
  011001
```

3.2.6 Complemento a 2

Ahora lo que se pretende es ver lo que le falta a un número binario de n bits para alcanzar al número binario que tiene un 1 y n 0's, o lo que es lo mismo, primero hacemos el complemento a 1 del número de n bits y al resultado le sumamos 1.

Ejemplo: Vamos a calcular el complemento a 2 del número 100110:

```
                      100110  |   1000000
COMPLEMENTO A 1       011001  | -  100110
+                          1  |---------
-----------------------------  |   0011010
                      011010  |
```

Con la operación de complemento a dos también podemos realizar la operación de resta binaria con solo sumar, es decir, la operación de restar se transforma en coger el minuendo y sumarle el complemento a 2 del sustraendo despreciando el dígito binario más a la izquierda.

Ejemplo: Vamos a realizar la resta del los siguientes números: 10010110 - 01101100. Primero tomamos el complemento a dos de 01101100 que es 10010100 y sumando queda:

$$\begin{array}{rr} & 10010110 \\ + & 10010100 \\ \hline & 100101010 \end{array}$$

Despreciando la cifra más a la izquierda el resultado es: 00101010 que es el mismo que en ejemplo de la resta binaria.

Ejercicios propuestos

Tomar los ejercicios del epígrafe de resta binaria y realizar la operación anteriormente expuesta y comprobar que el resultado es el mismo.

3.2.7 Suma Lógica: OR

Las operaciones que hemos definido anteriormente son las operaciones aritméticas, es decir, las que operan sobre el conjunto de todos los bits que forman el número, las que vamos a definir a continuación son las que operan individualmente sobre cada bit, es decir, el resultado de un bit (dígito binario) no influye en el siguiente.

Esta primera operación tiene la siguiente tabla de operación, también conocida como tabla de verdad:

PRIMER OPERANDO	SEGUNDO OPERANDO	OR
0	0	0
0	1	1
1	0	1
1	1	1

Como puede observarse cuando hay un 1 el resultado es 1 y sólo da 0 cuando los dos son 0.

Ejemplo: Vamos a calcular el OR de 1001101 y 1100101

$$\begin{array}{rr} & 1001101 \\ \text{OR} & 1100101 \\ \hline & 1101101 \end{array}$$

Ejercicios propuestos

Realizar la suma lógica de los siguientes números: 10011 OR 11001110, 1001110 OR 1010, 1100100110 OR 11000111001.

3.2.8 Producto Lógico: AND

Esta es la segunda de las operaciones lógicas que vamos a definir. Su tabla de verdad es la siguiente:

PRIMER OPERANDO	SEGUNDO OPERANDO	AND
0	0	0
0	1	0
1	0	0
1	1	1

Como puede verse sólo se como resultado 1 si los dos son 1, cero en caso contrario.

Ejemplo: Utilizando los dos números anteriores vamos a calcular su AND:

$$\begin{array}{rr} & 1001101 \\ \text{AND} & 1100101 \\ \hline & 1000101 \end{array}$$

Ejercicios propuestos

Realizar el producto lógico de los siguientes números: 11001 AND 11000110, 11110000 AND 101010110, 1111 AND 10110010011.

3.2.9 Negación Lógica: NOT

Esta operación lógica tiene el mismo efecto que el complemento a 1.

3.2.10 Suma Lógica Exclusiva: XOR

Esta va a ser la última operación lógica que definamos. Su tabla de verdad es la siguiente:

PRIMER OPERANDO	SEGUNDO OPERANDO	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Como puede observarse si los dos son iguales es cuando da 0, 1 en caso contrario. Con esta operación lo que quiere darse a entender es que sólo puede existir una alternativa.

Ejemplo:

Como en el caso anterior vamos a utilizar los mismos números que en los ejmemplos anteriores:

$$\begin{array}{rr} & 1001101 \\ \text{XOR} & 1100101 \\ \hline & 0101000 \end{array}$$

Ejercicios propuesto

Realizar la suma lógica exclusiva para los siguientes números: 111 XOR 111, 10000 XOR 010110, 1010101010 XOR 1011011010.

1B DE VERIFICACIÓN	2B DE ZONA	4B DE POSICIÓN

Tabla 3.4: Estructura del código de Entrada/Salida BCD

3.3 Códigos de Entrada Salida típicos

Ya se indicó en la introducción de este capítulo que los códigos de entrada/salida (E/S o I/O) o códigos externos son códigos que asocian a cada carácter una determinada combinación de 0's o 1's. El número de bits necesarios dependerá de cuantos caracteres distintos se quieran representar, es decir, si tenemos **n** bits sólo podemos representar 2^n caracteres distintos. En el caso general que queramos representar **m** caracteres distintos se debe de verificar que $\mathbf{n} \geq \log_2 \mathbf{m}$, o lo que es lo mismo $2^n \geq \mathbf{m}$ siempre teniendo en cuenta que **n** es un número entero.

A continuación se describirán algunos de los códigos de E/S más utilizados en informática y sus características. Al final se presentará una tabla donde aparecen:

3.3.1 BCD de intercambio normalizado

Usualmente este código utiliza n = 6 bits, con lo que con él se pueden representar m = 64 caracteres. A veces se añade a su izquierda un bit adicional para verificar posibles errores en la transmisión o grabación del código, de forma que en este caso cada carácter queda representado por n = 7 bits, tal como se muestra en la tabla 3.4.

Las cuatro posiciones de la derecha se denominan bits de posición y para los caracteres numéricos de 1 a 9 el código de los bits de posición coincide con la representación en binario natural de dichos números (código BCD para dígitos decimales) y para el 0 con la representación del 10; es por este motivo por lo que se denomina BCD. Los dos siguientes bits (bit 4 y 5) se denominan bits de zona, siendo estos 00 para los caracteres numéricos. El último bit, que es un bit opcional, es el bit de verificación.

3.3.2 EBCDIC

Extended Binary Coded Decimal Interchange Code.

Este código utiliza n = 8 bits para representar cada carácter, con lo que el significado se indica más en la tabla 3.5.

Este tipo de codificación permite codificar hasta m = 256 símbolos distintos, es decir, posibilita que se representen una gran variedad de caracteres; incluye las letras minúsculas y mayor número de caracteres especiales. También es posible, y se hace con las combinaciones que empiezan por 00, codificar caracteres de control que suministran órdenes o señales de control, por ejemplo, para la impresora, pantalla o para codificar las transmisiones de información.

3.3.3 ASCII

American Standard Code for Information Interchange.

Este código utiliza 7 bits y hoy en día es uno de los más usuales. Se puede decir que la mayor parte de las transmisiones de datos entre dispositivos se realizan en esta codificación. Usualmente se incluye un octavo bit para detectar posibles errores de transmisión o grabación.

4 bits de Zona		4 bits de Posición
00 Control	00 A-I	
01 Especiales	01 J-R	
10 Minúsculas	10 S-Z	
11 Mayúsculas y Números	11 Numérico	

Tabla 3.5: Código EBCDIC

En la tabla que a continuación se presenta se puede ver cuales son los caracteres representados por este código.

Hoy en día el octavo bit no se utiliza en algunos ordenadores (como son los PC's) para detección de errores, sino que se utiliza para poder representar el doble de caracteres. Al código resultante se le conoce como ASCII extendido, con lo que se pueden llegar a representar hasta un total de 256 caracteres distintos.

3.4 Detección de errores. Paridad

En el apartado anterior hemos podido comprobar que cantidad de bits son necesarios para representar una cantidad fija de caracteres distintos. También se ha visto que no siempre se utilizan todas las combinaciones, es decir, hay combinaciones de 0's y 1's que no se corresponden con ningun carácter. Cuantas menos combinaciones se desperdicien más eficiente es el código. La **eficiencia de un código** (τ) se define como *el cociente entre el número de símbolos que se representan realmente, m, dividido por el número, $\overline{m}$, de símbolos que en total pueden representarse*; es decir, con códigos binarios en los que $\overline{m} = 2^n$, se tiene

$$\tau = \frac{m}{\overline{m}} = \frac{m}{2^n}$$

con

$$0 \leq \tau \leq 1$$

Obviamente, cuanto más eficiente sea el código, τ será mayor.

Ejemplo:

Supongamos que utilizamos el código ASCII, sin bit de paridad, para representar 95 símbolos. La eficiencia de esta codificación será:

$$\tau = \frac{m}{2^n} = \frac{95}{2^7} = 0.742$$

Si introdujésemos un bit adicional, de paridad, la nueva eficiencia sería:

$$\tau = \frac{m}{2^n} = \frac{95}{2^8} = 0.371$$

Un código que es poco eficiente se dice que es redundante, es decir, transmite poca información útil.

Se define **redundancia** como $R = (1 - \tau) * 100$; observamos que se da en tanto por ciento.

Ejemplo:

Las redundancias del ejemplo anterior son:

$$R = (1 - 0.742) * 100 = 25.8\%$$

ALFABETO	CÓDIGO I	CÓDIGO II
a	000	0000
b	001	1001
c	010	1010
d	011	0011
e	100	1100
f	101	0101
g	110	0110
h	111	1111

Tabla 3.6: Dos códigos distintos para representar el mismos grupo de símbolos.

y

$$R = (1 - 0.371) * 100 = 62.9\%$$

A veces las redundancias se introducen deliberadamente para poder detectar posibles errores en la transmisión o grabación de información. Así si se desearan transmitir 8 símbolos y no deseasemos redundancias se necesitarían n = 3 bits. En este caso si al transmitir o al grabar se alterase alguno de los bits la información variaría y además no se podría detectar el posible error. Si se utilizase un código redundante, como añadiendo un bit más al código anterior, existirían algunas posibilidades de detectar errores. Así si se transmite el símbolo h, esto es, 1111, y por un error en la transmisión cambiase el primer bit, esto es, recibiéramos 0111, se detectaria como error al no coincidir con ninguno de los códigos válidos y por tanto no tiene asociado ningún carácter.

Las redundancias se introducen de acuerdo con algún criterio predeterminado; de esta manera los códigos pueden ser verificados automáticamente por circuitos de la computadora o de periféricos especializados en este objetivo.

Uno de estos algoritmos añade al código inicial de cada carácter un nuevo bit denominado bit de paridad. Existen dos criterios para introducir este bit:

Bit de paridad, criterio par: se añade un bit (0 ó 1) de forma tal que el número total de unos del código que resulte sea par.

Bit de paridad, criterio impar: se añade un bit (0 ó 1) de forma tal que el número total de uno del código que resulte sea impar:

En la tabla 3.6 se ha utilizado un bit de paridad con criterio par, en el caso que el criterio fuera impar sólo habría que cambiar el primer bit del Código II, es decir, cambiar los 1's por 0's y los 0's por 1's.

El bit de paridad se introduce antes de transmitir o grabar la información. Por ruido o interferencias en la transmisión o defecto del soporte de grabación puede eventualmente cambiar un bit. Al escribir o al leer la información se comprueba la paridad, se detectaría el error, ya que el número de unos dejaría de ser par o impar. Obviamente, si se produjese el cambio simultáneo de dos bit distintos no se detectaría el error de paridad; ahora bien, esta eventualidad es menos probable y por tanto no justificaría una redundancia mayor en el código con la consiguiente perdida de eficiencia.

SÍMBOLOS	CÓDIGO CUENTA FIJA
a	0011
b	0101
c	1001
d	0110
e	1010
f	1100

Tabla 3.7: Ejemplo de una codificación utilizando cuenta fija

Existen otros sistemas de introducir redundancias para poder detectar errores. Uno de ellos es el denominado de verificación de cuenta fija. En este tipo de sistema todos los códigos correspondientes a los caracteres tienen un número fijo de unos y ceros, siendo la posición relativa de los mismos la que determina el carácter a que corresponden. Por ejemplo, si quisiésemos codificar 70 símbolos distintos podríamos utilizar 8 bits de los cuales siempre 4 deben ser unos y los otros 4 ceros. Los circuitos de verificación tendrían que detectar en escritura o en lectura si el número de unos de cada carácter coincide con el número prefijado.

Ejemplo: Vamos a ver como se deberían codificar seis símbolos, lo haremos con 4 bits y siendo siempre el número de unos y ceros igual a 2.

Hay otros tipos de códigos para detección de errores como son los de paridad vertical y los polinómicos que son muy utilizados en transmisión de datos, estos son códigos mas elaborados y además de detectar un número mayor de errores que estos últimos, también pueden llegar a corregir algunos de estos errores.

Capítulo 4

Sistemas Operativos

"Dadme un punto de apoyo y levantaré el mundo"

La misma importancia que tiene el punto de apoyo para levantar el mundo, la tiene el S.O. para gobernar la máquina.

Como ya sabemos, un ordenador no entiende otra cosa que no sea código binario (ceros y unos). Toda la información y datos que procesa deben estar y están en este código bivaluado, o código máquina. Así, para comunicarnos con él, sería necesario que conociéramos ese código y tradujésemos a él todos los datos e instrucciones, para que el ordenador nos entendiera y ejecutara nuestras órdenes.

Afortunadamente, hace unos cuantos años se inventó un programa (software) que liberaba al hombre (usuario) de tener que usar el código binario para comunicarse con el ordenador. Este software, que hace de **intermediario** entre el hombre y la máquina, se llama **Sistema Operativo (S.O.)**, software de sistema o software funcional (ver figura 4.1).

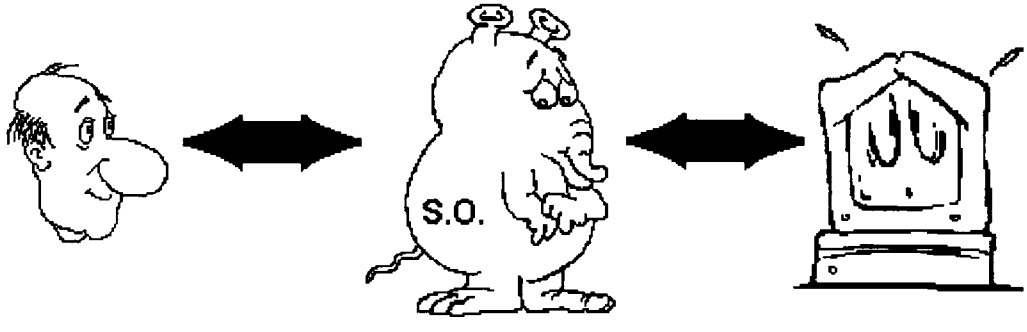


Figura 4.1: El S.O. es un intermediario entre el Usuario y el Hardware.

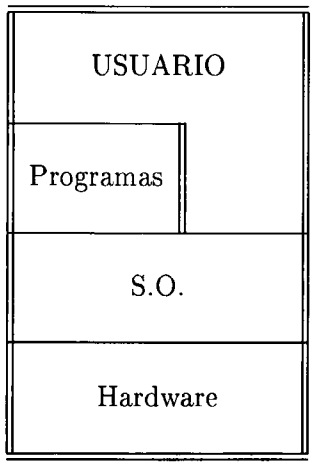


Figura 4.2: Posición del S.O.

El Sistema Operativo (S.O.) trabaja como un **traductor**, traduciendo primero nuestras órdenes a código máquina, las cuales ejecuta el ordenador, y luego traduce los resultados obtenidos de código máquina a un formato comprensible por el hombre. De esta forma, podemos decir, que el S.O. (Sistema Operativo) que se sitúa entre el usuario humano y el hardware físico (el ordenador), como vemos en la figura 4.2. Pero a veces, el humano no se comunica directamente con el S.O., sino que se comunica con otros programas, y son estos los que se relacionan con el S.O. para conseguir hacer funcionar al hardware. También es posible que haya programas que manejen directamente el hardware, sin pasar por el S.O., pero eso es más raro y no lo consideraremos.

Pero además, el S.O. hace otras tareas que, por su naturaleza, son complicadas de realizar por el hombre, de forma que nos facilita el uso y control de estas *maquinitas*, para que así, podamos aprovechar mejor, todas y cada una de las funcionalidades de una computadora.

Entre las tareas básicas de un S.O. podemos encontrar las siguientes, pero dejando claro que no todos los S.O. las tienen todas, sino que dependiendo del S.O. concreto, implementará todas o varias de las siguientes tareas:

- **Carga del programa en memoria:** Decir en qué zona libre de la memoria se debe cargar.
- **Planificar y supervisar la ejecución de los programas:** Controlar qué programas (procesos) se deben ejecutar en cada momento y sus posibles acciones.
- **Control de operaciones de Entrada** de datos al ordenador.
- **Control de operaciones de Salida** de datos del ordenador.
- **Tratamiento de posibles errores:** Controlar las acciones de cada programa y evitar que estas interrumpan o bloqueen a otros programas. Debe tener siempre un **registro de estado** en el que anotar la situación de cada programa, cada periférico, cada instrucción...
- **Traducir** datos y órdenes entre usuario y ordenador.

<table>
<tr><td></td><td colspan="4">USUARIO</td></tr>
<tr><td>(1)</td><td>Dibujo</td><td>Música</td><td>H. Cálculo</td><td>Juegos</td></tr>
<tr><td rowspan="2">(2)</td><td>Editores</td><td>Shells</td><td>Compiladores</td><td>Linkers</td></tr>
<tr><td colspan="4">Sistema Operativo</td></tr>
<tr><td rowspan="3">(3)</td><td colspan="4">Lenguaje Máquina</td></tr>
<tr><td colspan="4">Microprograma</td></tr>
<tr><td colspan="4">Dispositivos Físicos</td></tr>
</table>

(1) Software de Aplicaciones.
(2) Software de Sistema.
(3) Hardware.

Figura 4.3: Hardware y Software de un ordenador

- **Facilitar el trabajo de programadores:** librándolos de la complejidad que supone programar para un hardware concreto, de manera que los programadores hacen sus programas diciendo **qué** debe hacer el hardware, pero **no cómo** hacerlo, que es algo muy dependiente del dispositivo concreto del que se disponga y de lo que se encargará el S.O.

Los programas de un S.O. se pueden separar en tres, según su utilidad:

- **Programas interfaz hombre-máquina:** Programas que hacen que el trabajo (comunicación) con la máquina sea más cómodo y esta sea fácil de usar, (y ahora más con los S.O. con interfaces gráficos, como OS/2, Windows...). También son llamados **Intérpretes de Comandos** (*shells*).

- **Programas de control de recursos:** Programas que automatizan y mejoran el rendimiento del ordenador, aprovechando al máximo sus componentes (memoria, CPU, periféricos...).

- **Programas de ayuda a la programación:** Los que facilitan el trabajo de programadores (editores, compiladores, traductores, librerías, enlazadores...). No son fundamentales para el S.O., pero se suelen incluir con este.

Mención aparte merecen los **programas de aplicación**, los cuales son los que resuelven los problemas de los usuarios (procesadores de texto, de cálculos matemáticos, Hojas de cálculo, programas de diseño, dibujo, bases de datos...), que a veces se incluyen con el propio S.O., pero que no son imprescindibles para el funcionamiento de este.

En la figura 4.3 veremos más en detalle el funcionamiento de un ordenador y la función de un S.O.:

En la parte superior se encuentra el **usuario**, que es el que se encarga de usar el ordenador para los más diversos fines, ejecutando los **programas de aplicación**. Por debajo está el **software de sistema**, con los programas que antes hemos visto. Perteneciente ya a la parte **hardware**, nos encontramos con el **lenguaje máquina**, al que se traducen todas las órdenes y datos que debe procesar el computador y que depende de cada computador en concreto. Suele tener entre 50 y 300 instrucciones distintas, dependiendo de cada procesador (CISC, RISC...)

y son del tipo de instrucciones usadas en lenguaje ensamblador, como ADD (sumar datos), MOVE (mover de memoria o a memoria), JUMP (saltar a una instrucción determinada)...

Así, por ejemplo, para leer un dato de un disco, en lenguaje máquina, hay que decir que vamos a leer (también podría ser escribir), de qué disco (dirección), qué cara, pista, sector, número de bytes a leer, dirección en memoria principal donde almacenar lo que leamos... y luego controlar si ha habido algún error. Afortunadamente de todo eso se encarga el S.O. automáticamente. Nosotros lo único que tenemos que decir es **qué** información queremos leer, pero **no cómo** leerla.

Si bajamos más de nivel, nos encontramos con el llamado **microprograma**, que desglosa las operaciones del lenguaje máquina (ensamblador) en instrucciones más simples y ejecutables de forma directa por los **dispositivos físicos**, como indicar a la ALU la operación a realizar, activar los buses y registros o hacer girar el disco y mover las cabezas de lectura/escritura... en el ejemplo anterior.

Este tipo de microinstrucciones suelen estar grabadas en memoria ROM (puesto que no se deben borrar) y son básicas para el funcionamiento del ordenador. De hecho, tanto el lenguaje máquina como el microprograma en sí, son componentes software, pero se suele incluir como hardware debido a su cercanía física y lógica.

Existe un nivel intermedio, de lógica digital, donde los dispositivos implementan las operaciones lógicas y aritméticas. La base de los circuitos son las puertas lógicas, que están construídas a base de transistores.

Por último, los **dispositivos físicos**, son el hardware propiamente dicho, y están formados por chips, transistores, cables, buses, fuentes de alimentación, motores, cabezas de lectura/escritura, memoria...

Hemos visto que hay distintos tipos de software, y que hay software cuya ejecución es más importante y más de bajo nivel que otros programas. Por eso, el software, dependiendo de su importancia, se ejecutará en uno de los dos modos siguientes:

- **Modo privilegiado o supervisor:** Modo de ejecución propio de las rutinas del S.O., de forma que está protegido (por hardware) de posibles intromisiones por parte del usuario u otros programas, para que nadie efectúe ninguna operación ilegal, que podría llevar al colapso del sistema. Pensemos por ejemplo que un programa pudiera acceder a la zona de datos del sistema operativo para cambiarlos o asignarse más prioridad en la ejecución. Este tipo de acciones no deben pasar desapercibidas por parte del S.O., quien debe evitar que alguien efectúe operaciones que no debe hacer (como acceder a zonas de la memoria que están reservadas a otros programas). En general, las operaciones que se ejecutan de este modo están muy ligadas al hardware de los distintos componentes del ordenador.

  En este modo de ejecución de programas se puede, en definitiva, acceder a todas las posiciones (direcciones) de la memoria, ejecutar todas las instrucciones máquina del procesador y modificar el valor de cualquier registro.

- **Modo usuario:** Forma de trabajo del resto de programas, que el usuario puede cambiar por otros o programarse los suyos propios. Hay determinadas operaciones que no pueden realizar ellos mismos y para ello deben efectuar una llamada al sistema operativo para que éste efectúe la operación en cuestión.

En las primeras generaciones de ordenadores y S.O., los programas se desarrollaban para una máquina concreta (no eran portables), de forma que al cambiar de ordenador se debía

renovar todo el software o adaptarlo a la nueva máquina. Esto suponía una pérdida importante en tiempo, dinero y material. En la década de los 70 surgen las familias de ordenadores compatibles (como la famosa familia x86 de Intel). Entre ordenadores compatibles se pueden intercambiar programas sin necesidad de modificarlos.

En la actualidad los S.O. tienden a simplificar la vida de los usuarios y permitir que estos puedan aprovechar al máximo los nuevos y potentes procesadores, con S.O. de 32 bits (¿para cuando de 64?), de una manera fácil de aprender y cómoda (con gráficos y ratón), con los que poder ejecutar varios programas simultáneamente (multitarea), con fácil configuración de periféricos (Plug & Play), acceso a ordenadores remotos a través de redes internacionales (FidoNet, Internet,...), posibilidad de ejecución de programas para distintos S.O. (cosa, hasta hace poco, impensable)... y un buen montón de utilidades más.

4.1 Funciones Básicas de un S.O.

Ya hemos visto en la introducción un resumen de las funciones básicas de un S.O., pero aquí las veremos agrupadas en dos funciones tan básicas como generales. Así, un S.O. (según Andrew S. Tanenbaum) se puede ver como una **máquina virtual** -que facilita su uso- o como un **administrador de recursos** - para conseguir un uso eficiente de todo el sistema-, dependiendo de como interese verlo en cada momento.

4.1.1 El S.O. como Máquina Virtual.

Cómo ya hemos visto, un ordenador o computadora sólo trabaja con código binario difícil de entender y manejar por nosotros los humanos. Para poder elaborar programas complejos sería necesario un profundo conocimiento de cada elemento del ordenador (cómo manejarlo, funciones, códigos de error...) y por si fuera poco, cualquier cambio que se produjera en el hardware, podría afectar al funcionamiento de estos programas.

Tanto el programador como el usuario necesitan una herramienta que les libere de toda la complejidad que supone manejar el hardware mediante código máquina. Esta herramienta es el S.O., el cual, puede ser visto como una **máquina virtual** (no real), a la que hay que decirle lo qué queremos que haga, pero no cómo hacerlo.

Por ejemplo, para ver el contenido de un fichero de disco, un programador debe abrirlo en modo lectura, leerlo y cuando termine cerrarlo. No debe preocuparse de que los motores del disco giren, de que la cabeza se posicione en la pista y sector adecuado, de posibles errores (disco erróneo, fichero no existe...). De todas esas acciones y muchas más se encarga el S.O., que nos da una serie de herramientas para abrir un fichero, cerrarlo, leerlo, escribirlo, crearlo, borrarlo, añadir información al final, modificarlo... en fin, que estamos viendo al S.O. y a la máquina física como si fuera una máquina fácil de manejar, pues nos da las herramientas necesarias para trabajar con ella, eliminando las complejidades.

4.1.2 El S.O. como Administrador de Recursos.

Un ordenador, como hemos visto en temas anteriores, está formado por un montón de módulos con funciones claramente definidas (la CPU, la memoria, buses, periféricos...). Cada uno de estos módulos o recursos son necesarios para que se ejecuten los programas, pero no es posible dejarlos para que los programas los usen libremente, ya que entonces cada programa intentaría usar el recurso de una forma y sería imposible ponerlos de acuerdo.

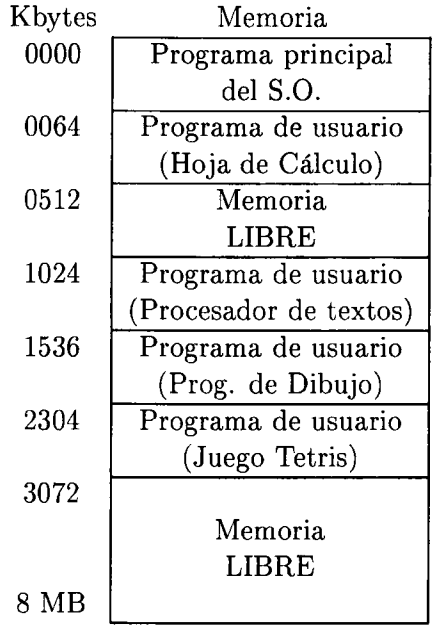


Figura 4.4: Asignación de memoria por el S.O.

El S.O. es visto, entonces, como un policía de tráfico que dirige y coordina todas las actividades del ordenador. Debe decir, qué programa se debe ejecutar en cada momento, qué parte de memoria asignarle, qué periféricos puede usar, cómo usarlos...

Sabemos, por ejemplo, que para que se ejecute un programa debe primero cargarse en la memoria principal (la RAM). Si al mandar ejecutar un programa cada usuario lo colocara en la zona de memoria que quisiera, pudiera darse el caso que borrásemos datos o instrucciones de otro programa, perdiéndose el trabajo realizado por este. Por eso, el S.O. es el encargado administrar la memoria, ver si hay memoria libre y qué parte está libre para cargar ahí el programa que se desee ejecutar. Además, debe asegurarse de que ningún programa ni usuario acceda a una parte de la memoria que no le corresponda, que no le haya sido asignada a él directamente, (sobre todo en sistemas multiusuario).

Otro ejemplo, pensemos que varios programas envían sus trabajos a imprimir a la misma impresora. El resultado podría ser unas líneas o caracteres del primer programa, luego unas líneas o caracteres del segundo... de forma que el listado final no fuera útil para nadie. Por esto, debe ser el S.O. el encargado de asignar por turnos el uso de la impresora a todos los programas (o usuarios) que deseen usarla. Una forma de hacer esto es que el S.O. guarde todos los trabajos a imprimir en una cola de impresión, de forma que conforme vayan llegando se van poniendo al final de la cola, mientras la impresora imprime del principio de la cola. De esta forma, los programas que quieran imprimir algo, sólo tienen que mandarlo a imprimir sin preocuparse de si la impresora está o no libre.

Es necesario, como hemos visto, que haya un programa (el S.O.) que se encargue de controlar, gestionar y administrar, todos los dispositivos del sistema, para que estos sean usados conveniente y óptimamente.

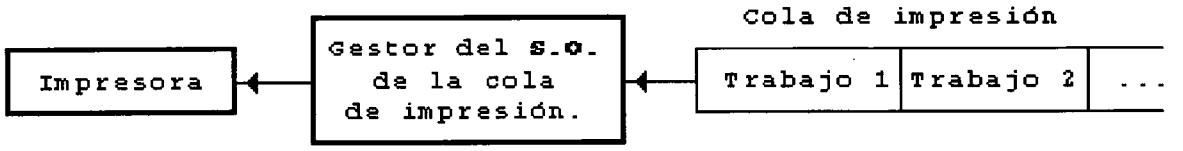


Figura 4.5: Gestor de la cola de impresión de una impresora.

4.2 Evolución de los S.O.

Históricamente, los S.O. han estado íntimamente ligados a la arquitectura de los computadores en que se ejecutaban, ya que siempre que se desarrollaba o se desarrolla un S.O. se hace para un tipo (plataforma) de máquina concreta.

O sea, que el S.O. de un Macintosh no funciona en un PC, ni viceversa, y el de un PC no funciona en una máquina con el poderoso procesador Alpha de DEC (de Digital), salvo que se modifiquen convenientemente.

Por eso, la evolución de los S.O. la veremos siguiendo la evolución de las generaciones de computadores, dejando claro que las distintas generaciones están formadas por distintos S.O. y que por tanto los límites cronológicos entre ellas son difusos.

Como primera máquina digital, fuera de lo que hoy conocemos como computadoras, podemos nombrar a la llamada *máquina analítica* del matemático Charles Babbage (1792-1871). Su diseño puramente mecánico y la tecnología de su tiempo hicieron a Babbage perder su fortuna en algo que no llegó a funcionar del todo bien. Esta máquina no disponía de S.O., sino que los datos debían ser introducidos en código binario.

4.2.1 Primera generación (1945-1955): En busca del S.O. perdido.

A mediados de los 40, se empezaron a construir las primeras máquinas de calcular, utilizando **tubos de vacío** (como la de John Von Neumann, Howard Aiken, Konrad Zuse...) Estas máquinas eran inmensamente grandes (varios pisos de un edificio) y eran muchísimo más lentas que un ordenador doméstico de hoy.

El equipo de personas que construía estos *ordenadores* era el que se encargaba de su mantenimiento y toda su programación se realizaba exclusivamente en lenguaje máquina. Era necesario cablear (conectar) los circuitos convenientemente, para que funcionaran adecuadamente en cada programa.

Los programadores de esta época no conocían lenguajes de programación (ni ensamblador), ni por supuesto nada que se pareciera a un S.O. Los programadores solicitaban la máquina por un período de tiempo y cuando les era concedida insertaban sus conexiones en un panel frontal, y sus datos (en binario todo), y esperaban a que el programa funcionara, sin que se estropeara ninguno de los 20 ó 25.000 tubos de vacío.

La idea de Von Neumann (1903-1957), expuesta en 1946, de insertar las instrucciones en la memoria, junto con los datos, en forma de **programa almacenado** revolucionó la informática, ya que hasta entonces los programas se introducían en las computadoras estableciendo manualmente las conexiones entre las diferentes unidades (en el llamado panel frontal de conexiones). Hasta entonces, la memoria se utilizaba exclusivamente para los datos. Es decir,

su funcionamiento era muy similar a las calculadoras de bolsillo más simples.

A principios de los 50 se mejoraron mucho las velocidades para introducir datos e instrucciones, con el invento de las tarjetas perforadas. Ya se podían escribir programas e introducirlos, junto con los datos, mediante lectores de tarjetas perforadas, pero aún así, todo el proceso se hacía en código binario sin ayuda de ningún Sistema Operativo.

4.2.2 Segunda generación (1955-1965): La era del transistor. S.O. en Batch.

A principios de los 50 se inventó el **transistor** (palabra que viene del inglés transfer resistor, resistencia de transferencia), por Bardeen, Brattain y Shockley, los cuales fueron galardonados con el premio Nobel de física de 1956.

No fue hasta mediados de los 50, cuando el transistor cambió radicalmente el panorama informático. Los computadores se volvieron lo suficientemente fiables como para venderse a clientes, sin que dejasen de funcionar por un período considerable de tiempo.

Fue entonces cuando se establecieron jerarquías laborales informáticas: **diseñadores** (diseñaban la máquina), **fabricantes** (desarrollaban sus piezas y las ensamblaban), **personal de mantenimiento** (encargados del buen funcionamiento de todos los componentes), **operadores** (controlaban el trabajo con el ordenador y las operaciones de E/S), y **programadores** (escribían los programas -instrucciones- del ordenador). Los **programadores**, escribían su programa (en Fortran o Ensamblador), luego perforaban las tarjetas según este programa y por último le daban el lote de tarjetas a un **operador**, para que este las pusiera en el lector cuando le tocara el turno.

Obviamente, entre el cambio de tarjetas se perdía mucho tiempo, en el que el computador estaba parado. Teniendo en cuenta el precio de estas máquinas, salía muy caro el tiempo que estuviera sin hacer nada. Para ahorrar tiempo (y dinero), se pensaron los llamados **sistemas de procesamiento por LOTES** (ver figura 4.6), o **sistemas BATCH** (lote), en los que los trabajos, en tarjetas (1), se leían con una máquina lectora y se pasaban a una cinta, todos seguidos, en lote (2), y acto seguido, un programa (el primer sistema operativo), se encargaba de leer el primer trabajo de la cinta y ejecutarlo (3) en el ordenador principal (por ejemplo un IBM 7094). Cuanto este terminaba, el programa leía el siguiente trabajo y lo ejecutaba, ejecutando así uno tras otro todo el lote de trabajos de la cinta.

Las salidas también se hacían en una cinta (4) que posteriormente sería mandada a impresora (5) por otro ordenador (o a otras tarjetas).

Las operaciones anteriores se podían realizar en paralelo, es decir, mientras se estaban pasando a cinta las tarjetas, se podían estar ejecutando programas e imprimiendo los resultados (desde la cinta), todo a la vez. Normalmente se usaban ordenadores pequeños para pasar datos a las cintas y para imprimir datos desde las cintas, y el ordenador más potente es el que ejecutaba los programas. Nótese que mientras el ordenador más potente realizaba una lectura/escritura de/en cinta no podía seguir ejecutando el programa hasta que no terminara esa operación de lectura/escritura (problema que se soluciona en la siguiente generación).

Los S.O. más típicos de esta segunda generación fueron el FMS (Fortran Monitor System) e IBSYS, el sistema operativo de IBM para su ordenador 7094.

Los S.O. en Batch, también se llamaron sistemas de administración de la CPU por **monoprogramación**, en contraposición a los de multiprogramación que veremos en la siguiente generación.

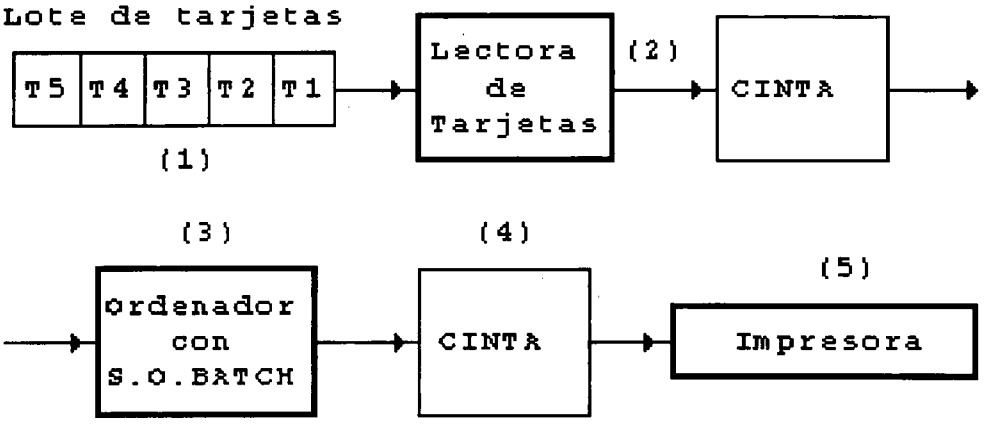


Figura 4.6: Ejecución de programas en Batch.

4.2.3 Tercera generación (1965-1980): Multiprogramación y Tiempo Compartido.

En esta época, los ordenadores seguían siendo, por su precio, exclusivos de grandes empresas, universidades... las cuales, en muchos casos, compraron un ordenador que al poco tiempo se les había quedado pequeño y necesitaban otro que ejecutara los mismos programas (compatible software) pero más rápidamente.

Por aquellos años se empezaron a usar los circuitos integrados (IC) de pequeña escala de integración, que abarataron los costes e incrementaron la calidad.

Se creó el concepto de *familia de ordenadores*, dentro de los cuales cualquier programa que funcionara en uno, podía funcionar en el resto, diferenciándose entre ellos, en potencia, memoria, tamaño, precio... pero no en arquitectura y juego de instrucciones.

La idea fundamental era que todo software pudiera funcionar en todos los modelos, incluido el S.O. (ya bastante mejorado), pero que como se pretendía que el S.O. sirviera para todos los modelos y para todas las funciones y periféricos, resultó un S.O. (OS/360) inmenso y terriblemente complejo. Tenía millones de líneas en lenguaje ensamblador, escritas por miles de programadores, y que contenían miles de errores, que al corregirlos generaban otros nuevos. Pero a pesar de esto el S.O. funcionaba y de manera razonablemente aceptable.

Este S.O. mejoró mucho las técnicas de la generación anterior. Se dieron cuenta, que en un sistema en Batch (por lotes), se perdía mucho tiempo cuando un proceso quería efectuar una operación de E/S (Entrada/Salida). Cuando un proceso quería escribir en cinta o impresora, leer un dato de la consola... el procesador se quedaba parado mientras se efectuaba esta operación. En operaciones científicas esto no era muy grave, ya que el número de E/S respecto a operaciones del procesador es mínimo, pero en aplioaciones de gestión, pueden llegar a suponer más de un 80% del tiempo total, por lo que la pérdida de tiempo era tremenda. La solución fue lo que se conoció por gestión de la CPU por **multiprogramación**, esto es, dividir la memoria del ordenador en varias partes (particiones) y poner trabajos distintos en cada una (Ver figura 4.4). Se empezaba ejecutando el primer trabajo hasta que terminaba o llegaba a una instrucción de E/S. Mientras este ejecutaba su E/S (que podía tardar mucho,

dependiendo de si el dispositivo estaba libre o no), el siguiente programa (cargado en otra partición) se empezaba a ejecutar en iguales condiciones. Cuando un programa terminaba, su partición quedaba libre para ser cargado el siguiente programa de la cola (en batch).

Con la técnica de la multiprogramación, y teniendo una memoria suficientemente grande como para albergar suficientes trabajos, se garantizaba que el procesador estuviera funcionando todo el tiempo, ya que cuando un proceso no lo usaba, pasaba a usarlo el siguiente, para más tarde volver a ejecutar el proceso anterior, por donde se quedó. El módulo del S.O. encargado de asignar la CPU (o administrarla) al programa que corresponda se llamó dispatcher (despachador), o planificador de prioridades de la CPU.

Al tener varios trabajos al mismo tiempo en memoria, era necesario proteger cada uno, para que ninguno de los otros pudiera usar su partición de memoria. Normalmente esta protección se hacía por mecanismos hardware. La técnica de **SPOOL** (*Simultaneous Peripheral Operation On Line*, operación simultánea de periféricos conectados en directo) también data de esta generación. Consistía en pasar las tarjetas perforadas a un disco, de forma que cuando un trabajo terminara y liberara su partición, esta fuera rápidamente rellenada con un programa leído del disco. Este sistema también se usó para la salida a los distintos periféricos (impresoras...). La idea principal es que los periféricos funcionaran simultáneamente a la CPU. Así, mientras la CPU estaba ejecutando un programa, la impresora podía estar imprimiendo un trabajo grabado en disco.

Con este sistema de multiprogramación se corría más que en batch, pero aún así, para recoger los resultados se tardaban un buen montón de horas y a veces, un fallo tonto en el programa obligaba a corregirlo e iniciar de nuevo todo el proceso. Los programadores necesitaban tiempos de respuesta más cortos, sistemas de trabajo interactivos con el ordenador... trabajar como si hubiera un ordenador para cada uno (una utopía por aquellos tiempos).

Con estas ideas nacieron los sistemas operativos con otros tipos de multiprogramación mejores:

1. **Multiprogramación clásica:** Es la primera que surgió y que hemos visto anteriormente. Se trata de asignar un programa a la CPU, cuando el que la esté usando no necesite la CPU, por estar efectuando una operación de E/S.

2. **Tratamiento paralelo:** Una variante de la multiprogramación en la que cada usuario tiene un terminal conectado al computador y este los atiende un determinado tiempo a cada uno, si tiene algo que hacer. Ese tiempo se llama **quantum** y varía de un sistema a otro, pero suele ser de fracciones de segundo. El ordenador ejecuta cada programa durante un quantum de tiempo, y luego pasa al siguiente, así sucesivamente, hasta que le vuelve a tocar al mismo programa de nuevo.

3. **Tiempo compartido** (*Time sharing*): Con el tratamiento paralelo de antes, pueden existir aún tiempos muertos de CPU, ya que si un proceso inicia una E/S durante su quantum de tiempo asignado, el resto de ese quantum, queda el procesador parado. Así, con el sistema de administración de la CPU, llamado por *Tiempo Compartido*, la CPU pasa de estar asignada a un programa a estar asignada a otro, si ocurre alguna de las tres condiciones siguientes:

   (a) El programa con la CPU agota su quantum de tiempo.

   (b) El programa con la CPU inicia una instrucción de E/S.

(c) El programa termina su ejecución.

Así, si hay 10 usuarios conectados, y 5 están con otras tareas, el computador atiende a los otros 5, y aún puede quedar tiempo para ejecutar otros programas en batch. De esta forma los usuarios tienen la impresión de disponer en exclusiva del computador, por estar trabajando con el de forma interactiva (conversacional).

Naturalmente, cuantos menos usuarios haya conectados más rápido se procesarán los requerimientos de los usuarios conectados, pudiéndose dar el caso, de que si hay muchos usuarios conectados, y todos ellos procesando mucha información, puede dar la sensación de que el procesador va muy lento, o incluso se ha parado.

El primer S.O. serio de tiempo compartido fue el CTSS, desarrollado en el MIT y tuvo tanto éxito que el MIT, los laboratorios Bell y General Electric decidieron unir fuerzas para desarrollar un macro-proyecto, el sistema **MULTICS** (MULTiplexed Information and Computing Service, servicio multiplexado de información y procesamiento). Se pretendía dar servicio (en tiempo compartido) a cientos de usuarios conectados a una red, como la red de distribución eléctrica, en toda la ciudad de Boston.

El proyecto MULTICS no funcionó como se pretendía, pero se usó como entorno de desarrollo en el MIT, e introdujo muchas novedades.

Los ordenadores, se fueron extendiendo, como el PDP-1 de DEC en 1961, que con sus 4K palabras de 18 bits se vendió muchísimo por su *alta* productividad. La familia de ordenadores PDP llegó hasta el PDP-11. Ken Thompson, un experto programador de los laboratorios Bell, que había trabajado en MULTICS, encontró un pequeño ordenador PDP-7 que nadie usaba, y decidió escribir una versión reducida de MULTICS par un solo usuario. Brian Kerninghan le llamó UNICS (UNiplexed, en contraposición a MULTICS), y pronto cambió su ortografía a **UNIX** (marca registrada de los Laboratorios Bell de AT&T). Posteriormente el sistema se trasladó a un PDP-11/20 donde funcionó bien y tuvo bastante éxito. Dennis Ritchie, otro experto de Bell, se unió con Thompson para reescribir el UNIX en un lenguaje de alto nivel, denominado **C**, diseñado y desarrollado por Ritchie. Los laboratorios Bell regalaron licencias de UNIX a las universidades, extendiéndose rápidamente y convirtiéndose en el S.O. que se ha trasladado a más computadores de la historia, y su uso aún se extiende día a día, entre universidades y empresas.

4.2.4 Cuarta generación (1980-1995): Ordenadores Personales. Redes.

Desde el punto de vista hardware, se desarrollaron circuitos integrados **LSI** (Large Scale Integration, escala de integración grande), que permitían tener miles de transistores por centímetro cuadrado de silicio. Esto permitió bajar precios y aumentar velocidades, con lo que ya empezó a ser posible que una sola persona tuviera su ordenador, para uso personal o doméstico. Nació el primer PC de IBM (8088), que se convertiría en el primero de una saga (8086, 80286, 386) que dura hasta nuestros días (486, Pentium, Pentium Pro...).

La filosofía fue que fueran ordenadores **fáciles de usar**, y que no hiciera falta saber informática para manejarlos. Sus capacidades interactivas y sus posibilidades gráficas fueron revolucionarias para su época. Realmente sólo dos S.O. fueron ampliamente extendidos:

- **MS-DOS** (MicroSoft-Disk Operating System): de Microsoft, el S.O. de los primeros PC's de IBM y que su última versión, la 6.2, aparecida en 1993 y que será usada hasta más de 1995. No era un buen S.O. pero su facilidad de uso y su compatibilidad con la familia x86 hizo que fuera muy extendido.

- **UNIX:** S.O. multitarea y multiusuario que dominó ampliamente el área de grandes computadoras, y que aún hoy es ampliamente usado.

A mediados de los 80 se empezaron a conectar ordenadores entre sí, dando lugar a las llamadas **redes** de ordenadores, que ejecutaban dos tipos de S.O.:

- **S.O. de red:** En el que cada máquina tiene su propio S.O. local y cada usuario puede copiar ficheros o mandar mensajes desde una máquina a otra, usando las funcionalidades de este S.O. de red.

- **S.O. distribuidos:** Aquí sólo hay un S.O. que está distribuido entre todos los ordenadores de la red. Los usuarios no saben, ni tienen que saber, en qué procesador se están ejecutando sus programas o en qué lugar se encuentran sus ficheros, lo cual puede variar. Ellos ven al sistema como un ordenador monoprocesador, aunque realmente esté formado por muchos. Con esta filosofía nació el **multiproceso**, de forma que un programa, o varios, se pueden ejecutar simultáneamente en varios procesadores, minimizando así el tiempo total de ejecución.

Se deben hacer S.O. **tolerantes a fallos**, es decir, que sigan funcionando incluso si parte del hardware se estropea, de forma que el diseño de estos S.O. se complica considerablemente.

4.2.5 Quinta generación (1995-20XX): Ordenadores Superescalares. Redes internacionales.

El nacimiento de procesadores **superescalares**, que permiten la ejecución de varias instrucciones en paralelo y el desarrollo de su tecnología de fabricación (que permite tener casi de 4 millones de transistores en el mismo chip), ha permitido el nacimiento de S.O. monousuario, personales, pero que sean **multitarea** (se puede comenzar a ejecutar un programa sin que haya terminado el anterior) y **multiproceso** (que consiste en uno o varios procesos ejecutándose a la vez en varios procesadores).

El futuro no está nada claro, pero parece que crecerán las plataformas formadas por procesadores Pentium o su sucesor el P6 de Intel, el K5 de AMD, el T5 de MIPS, el PowerPC de Motorola, Apple e IBM y a un nivel superior, el procesador Alpha de DEC.

En cuestión de S.O. personales, hay que hablar básicamente de tres, sin olvidar el ya viejo DOS que sigue siendo ampliamente utilizado:

- **OS/2 versión Warp 3 Connect**, de IBM: S.O. multitarea y multiproceso que permite conexión a las redes internacionales de ordenadores como InterNet.

- **Windows95**, de MicroSoft: Nacido en el verano de 1995, fue, quizás, el primer S.O. en el que se gastó más en publicidad que en su desarrollo y que, aunque no supone un nuevo y revolucionario S.O., sí es una nueva versión mejorada del interfaz Windows 3.11. A pesar de sus inconvenientes, parece ser que ha venido para ser un estándar.

- **UNIX** para PC: Encabezado por el famoso **LINUX**, cada vez va teniendo mayor éxito este tipo de S.O. principalmente por su potencia y robuztez, pero también por la incorporación de interfaces gráficas (GUI, *Graphics User Interface*). Además de ser muy barato, para LINUX existen multitud de programas para cualquier cosa, que, además del propio LINUX, son muy fáciles de conseguir por las redes informáticas (InterNet, BBS...).

Parece ser, que esos dos S.O. dominarán el mercado de ordenadores domésticos, mientras que a nivel empresarial, están, básicamente el **Windows NT** y el **UNIX** con interfaz gráfico **X-Windows**. El primero requiere para ejecutarse, como mínimo un ordenador Pentium con 32 MB de RAM, por lo que ahora resulta inaccesible para ordenadores domésticos.

La idea de MicroSoft al sacar el Windows95, es que sirva de puente para que en un futuro pueda ser Windows NT un estandar. Así, para que a un programa se le conceda la licencia de compatible Windows95, requiere que sea también compatible con Windows NT, de forma que al ampliar el parque de programas para el 95, se amplía, a la vez, el parque de programas para el NT, que será la continuación natural del 95, cuando los usuarios tengan acceso económico a máquinas más potentes.

Pero lo más importante de esta generación es que prácticamente ningún ordenador estará desconectado de las redes internacionales. En esta generación se conseguirá que cualquier ordenador, por pequeño o doméstico que sea, pueda conectarse a redes como **InterNet** o **FidoNet**, aunque sea usando un modem desde su casa. Esto permite obtener información, datos y programas para los más diversos fines y desde cualquier punto del planeta, así como usar el correo electrónico para comunicarse con cualquier persona del mundo, en cuestión de segundos... la revolución acaba de empezar.

4.3 Conceptos básicos de Sistemas Operativos.

Como ya hemos visto, el S.O. comunica el hardware con el usuario y sus programas. La forma de hacer esto es a través de un conjunto limitado de instrucciones que es posible darle al Sistema Operativo para que realice estas tareas. Estas instrucciones se llaman **llamadas al sistema**, y es como si se llamara al S.O. para que ejecute alguna acción. Este tipo de llamadas o servicios, varían de un S.O. a otro, pero los conceptos básicos son iguales, y serán los que veremos, apoyados en un S.O. de tipo UNIX.

Así, podemos decir que las llamadas al sistema, en un S.O. se suelen dividir en dos, las que se encargan de gestionar **procesos** y las que se ocupan del control y manejo de **ficheros**.

4.3.1 Llamadas al sistema para PROCESOS.

Un **proceso** es un **programa en ejecución**, esto es, un programa que ha empezado a ejecutarse en un ordenador. Para ser más precisos, se suele llamar proceso a toda la información necesaria para ejecutar (o seguir ejecutando) ese programa, como son el código ejecutable del programa (sus instrucciones), el contador de programa (que indica por donde vamos ejecutando), el puntero de pila (para no perder los datos), la pila (donde se almacenan datos de interés), los datos, valor de los registros de la CPU...

Pensemos, por ejemplo, en un S.O. a tiempo compartido, en el que cada vez que se produce un cambio de programa en ejecución (porque haya consumido ya su tiempo de CPU), es necesario guardar toda esa información, para poder seguir ejecutando ese programa cuando le vuelva a tocar el turno. La información sobre el estado de los procesos, se suele llevar en una **tabla de procesos**, donde figuran todos los procesos que hay ejecutándose en cada máquina bajo el S.O. que sea.

Las llamadas al sistema para control de procesos pueden ser muchas, pero las más importantes son:

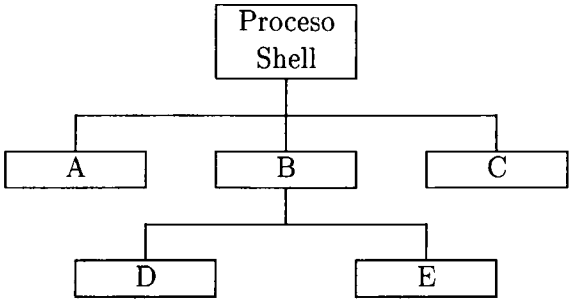


Figura 4.7: Árbol de procesos: creación de múltiples procesos hijos

1. Crear proceso (hijo).
2. Fin de proceso (vuelta al padre).
3. Esperar a un proceso (hijo).
4. Pedir memoria.
5. Liberar memoria.
6. Mandar mensajes (señales).

Veamos un ejemplo típico: Tenemos un proceso llamado **intérprete de comandos** o **shell**, ejecutándose en un ordenador. Este programa lo que hace es esperar órdenes del usuario, analizarlas y **crear** los procesos necesarios para la ejecución de esas órdenes de usuario. Los procesos que son creados por otro se llaman **procesos hijos**, y al que los creó **proceso padre**.

Un proceso puede crear más de un proceso hijo, y a su vez, cada proceso hijo, puede crear varios procesos, y así sucesivamente, hasta crear un **árbol de procesos**, como el de la figura 4.7.

Cuando un proceso termina, efectúa una llamada al sistema de **fin de proceso**, para indicar que ha terminado y que lo elimine de la tabla de procesos.

Otras llamadas al sistema muy típicas son para **esperar a que termine un proceso hijo** para continuar, o para **pedir más memoria**, donde el programa pueda almacenar más datos, para **liberar memoria** que previamente fue asignada, de forma que esta memoria pueda usarse por otros procesos, o para **mandar mensajes** (*señales*) a otros procesos en ejecución para indicarles cualquier dato o situación. El S.O. debe asegurar que estas señales son recibidas por el proceso destino, y en caso contrario avisar. Las señales se usan, por ejemplo, para indicar a un proceso que queremos que termine su ejecución y que deje de ejecutarse. Normalmente, cada usuario en un S.O. de este tipo tiene asociado un **uid** (*identificación de usuario*), que es un identificador que identifica a cada usuario. Para entrar en el sistema suele ser imprescindible indicar qué usuario es, y una clave de acceso o *password* (solo conocida, en teoría, por dicho usuario), que indica si efectivamente es el usuario que dice ser. Cuando un proceso se crea, a este se le asigna el **uid** de la persona que lo creó o que lo mandó ejecutar. Los procesos hijos, obviamente, tienen el mismo uid que el padre. Existe un uid llamado **superusuario** o **administrador del sistema**, que lo puede todo en el sistema. Este superusuario es el encargado de que el sistema funcione correctamente y de corregir los

problemas que puedan surgir, por lo que tiene poderes para hacer casi todo en el sistema. Además, cada proceso tiene un **pid** (*identificador de proceso*), que suele ser un número, y que identifica cada proceso independientemente. Cada proceso tiene un pid distinto del resto, de forma que sabiendo este pid podemos, por ejemplo, mandarle señales al proceso que queramos.

Como hemos visto, existe un programa llamado **shell** o **intérprete de comandos**, que aunque no es propiamente parte del S.O., se suele adquirir con él. Este programa hace dos operaciones importantes:

- Hace de interfaz entre el hombre y el S.O., de forma que el hombre se comunica con el S.O. a través de él.
- Traduce los comandos (órdenes) del hombre a las llamadas al sistema pertinentes para ejecutar dichos comandos.

El shell, al arrancarse muestra un signo de atención o **prompt**, que indica que está esperando órdenes. Cuando el usuario escribe alguna orden, el shell la analiza y crea el proceso hijo necesario para ejecutar dicha orden.

El usuario, desde el shell, puede dirigir la salida de un comando a un fichero, o leer un fichero, o conectar la salida de un programa con la entrada de otro (con las **tuberías** o **pipes**) y multitud de cosas más. También puede ejecutar programas en **background** (*en segundo plano*), de forma que mientras se ejecuta el programa, el shell devuelve de nuevo del prompt para que el usuario pueda ejecutar más órdenes. De esta forma, un usuario puede tener varios procesos ejecutándose a la vez.

Para pasar un proceso a background, en un S.O. tipo UNIX, es necesario añadir al final del comando el símbolo &. También es posible pasar un programa de background a primer plano (**foreground**), con la orden `fg`.

En el famoso S.O. MS-DOS, el intérprete de comandos se llama COMMAND.COM, que se ha visto mejorado con los nuevos interfaces gráficos de usuario (como Windows 3.1).

4.3.2 Llamadas al sistema para FICHEROS.

De una forma muy general, podemos definir un fichero (archivo o *file*) como un conjunto de información que identificamos mediante un nombre. En esta sección nos ocuparemos de las posibles acciones que se pueden llevar a cabo con este tipo de información.

El S.O. debe ser capaz de ocultar al usuario/programador la complejidad que supone el almacenamiento en disco u otro soporte, de los ficheros. Debe ofrecer, por tanto, una abstracción sencilla para efectuar al S.O. las llamadas pertinentes para las distintas acciones, que son, básicamente, las siguientes:

1. **Abrir un fichero:** Se trata de la primera operación que hay que hacer con un fichero. Lo primero es abrirlo, indicando el tipo de operación que queremos realizar con él.
2. **Cerrarlo:** Por añadidura, esta es la última operación que se debe hacer con un fichero. En muchos lenguajes de programación, al finalizar el programa, si hay ficheros sin cerrar, se cierran automáticamente, pero, por si acaso, es mejor cerrarlos. Un fichero sin cerrar puede originar pérdidas importantes de información.
3. **Leerlo:** Operación para *ver* la información que haya escrita en un fichero. En muchos casos hay que especificar a partir de qué posición queremos leer el fichero, y cuanta

información (en bytes). Normalmente, la lectura de los ficheros es secuencial, de forma que leemos un dato, y al volver a leer, leeremos el siguiente, sin necesidad de especificarlo explícitamente. Esto se hace, porque para cada fichero existe un índice o *puntero* que nos indica (o apunta) a una posición, que será la posición actual. Si leemos un dato, nos da el dato de esa posición y el puntero pasa a la siguiente posición.

4. **Escribir, modificar o añadir información:** Operación para modificar información, o añadir información al final, de un fichero que ya exista.

5. **Crearlo y escribir en él:** Igual que la lectura, se efectúa la escritura, pero en sentido inverso. También hay un puntero que indica dónde escribiremos el siguiente dato. Si abrimos un fichero para escribir en él y este no existe, normalmente, el S.O. se encargará de crearlo. Al crear un fichero, el S.O. debe apuntar quien lo creó, de forma que este usuario será el propietario de dicho fichero, y podrá decir quien puede leerlo, verlo, modificarlo...

6. **Borrarlo:** Cuando un fichero no nos interesa, lo que debemos hacer es borrarlo, eliminando así el acceso a ese fichero, y liberando el espacio en disco que ocupara, para que pueda ser usado por otros ficheros. Esta orden se incluye siempre en el S.O. como comando, por ejemplo, en el MS-DOS es `del` (delete) y en UNIX es `rm` (remove). Hay que tener mucho cuidado al borrar ficheros, pues un fichero borrado puede ser irrecuperable (en algunos sistemas existe la orden `undelete`).

El conjunto de ficheros en un disco duro, u otro dispositivo de almacenamiento no se suele (y no se debe) hacer almacenando o grabando todos los ficheros *caóticamente*, sino que se debe hacer creando unas carpetas o apartados especiales, llamados **directorios**, en los que se meterán todos los ficheros de una misma aplicación, o relacionados entre sí. Se trata de agrupar los ficheros afines, para que sea fácil localizarlos y trabajar con ellos.

Así, existirá un directorio principal, más conocido como **directorio raíz**, y dentro de este podrán existir ficheros y otros directorios **hijo** del raíz. Dentro de cada uno de esos directorios hijos del raíz, puede haber más ficheros y más directorios, creando una estructura jerárquica de directorios, como la mostrada en la figura 4.8.

No se debe confundir esta jerarquía de ficheros, con la jerarquía de procesos. Una cosa es la organización de los ficheros en un disco duro, por ejemplo, y otra cosa es la organización de los programas que se están ejecutando en cada momento en la máquina.

Como vemos, el nombre de un directorio se forma añadiendo una palabra al nombre del directorio padre, separada por el símbolo / (*slash*). De esta forma, dado el nombre completo de un directorio, también llamado **nombre de ruta** o **path name**, podemos llegar a él fácilmente, sin posibilidad de error, ya que nos dice la ruta (*path*) de directorios que hay que recorrer. A veces, para abreviar, y si no existe equivocación posible, podemos referirnos a un directorio sin nombrar el padre. De esta forma, si el directorio empieza por el símbolo /, sabemos que el nombre del directorio está de forma **absoluta**, es decir, referido al directorio raíz (con el path completo).

Podemos, por ejemplo, meter en el directorio PAISES, que es hijo (depende o cuelga), del directorio /TODOS, los ficheros que estén relacionados con el tema de *paises*.

Podemos también decir, que el directorio /PAZ, está formado por todos los ficheros que contenga este directorio, más todos los ficheros que contengan sus directorios hijos AMOR y LIBERTAD.

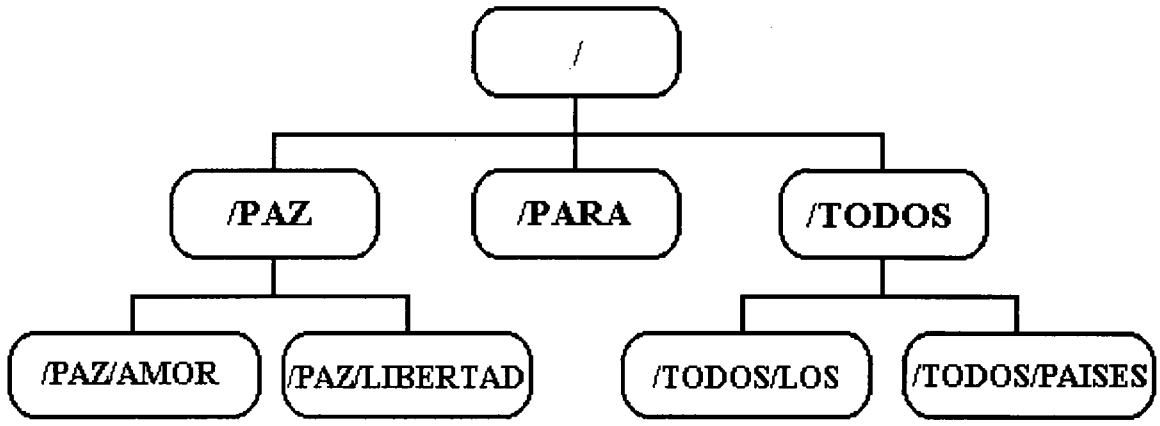


Figura 4.8: Estructura jerárquica del Sistema de Ficheros (*File System*).

Un S.O. debe incluir comandos para crear directorios (`mkdir`), borrar directorios (*rm*) y moverse de un directorio a otro (`cd`). De forma que siempre habrá un **directorio actual** o **directorio de trabajo**, que será el directorio en el que nos encontremos en ese momento. Esto indica que todas las acciones que hagamos (crear ficheros, modificarlos...) se harán dentro de ese directorio, salvo que indiquemos otro, explícitamente.

O sea, supongamos que existe un fichero llamado `PATRI`, dentro del directorio /PAZ/AMOR y otro fichero, con el mismo nombre en /PAZ/LIBERTAD. Si nuestro directorio actual, en el que estamos situados, es el directorio /PAZ/AMOR y modificamos el fichero `PATRI`, se modificará el fichero `PATRI` del directorio actual, y no el de otros directorios (como /PAZ/LIBERTAD) que también tengan un fichero con ese nombre.

Nota: En un S.O. tipo MS-DOS, la barra usada para separar directorios es la barra inversa, o sea \ ***(back-slash)***.

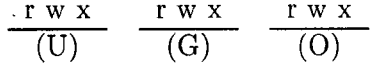
Cuando, en algún comando del S.O., nos queremos referir al **directorio actual**, nos podemos referir con un punto sólamente (.), y para referirnos al **directorio padre** del directorio actual, nos podemos referir con dos puntos, uno tras otro (..). De esta forma, podemos dar el nombre de un fichero o directorio de forma **relativa** (al directorio actual). Por ejemplo, si suponemos que nuestro directorio actual sigue siendo el /PAZ/AMOR, son equivalentes los nombres de directorio dados de forma relativa y absoluta de la tabla 4.1.

En un S.O. multiusuario, lo que se suele hacer es asignar a cada usuario un directorio de trabajo, donde pueda trabajar con sus ficheros. Este directorio de trabajo asociado a cada usuario se suele denominar **directorio HOME** (hogar) del usuario en cuestión.

Un S.O. serio (como UNIX), debe tener un sistema de **protección** de ficheros y directorios, de forma que cualquier usuario no pueda acceder libremente a la información allí contenida y leerla o modificarla a su libre antojo. En un S.O. tipo UNIX, esta protección se hace, asignando a cada fichero y a cada directorio 9 bits de protección. Estos 9 bits se dividen en 3 campos de 3 bits cada uno, refiriéndose el primer campo a los permisos del **propietario** (el que creó el fichero), el segundo campo se refiere a los permisos de un **grupo** de usuarios, al que debe pertenecer el propietario (normalmente en una empresa, cada usuario pertenece a uno o varios grupos, departamentos...), y el tercer campo se refiere a los permisos del **resto**

Ruta Relativa	Ruta Absoluta	
.	/PAZ/AMOR	porque es el directorio actual
..	/PAZ	porque es el padre del directorio actual
../LIBERTAD	/PAZ/LIBERTAD	porque es el directorio hijo del padre del directorio actual
../..	/ (dir. raíz)	porque es el padre del padre del directorio actual
../../PARA	/PARA	porque es hijo del directorio raíz

Tabla 4.1: Ejemplos de rutas absolutas y relativas (a /PAZ/AMOR).



Permisos para:
(U) - El Propietario del fichero (User).
(G) - El grupo de usuarios del propietario (Group).
(O) - El resto de usuarios (Other).

Figura 4.9: Permisos de un fichero o directorio.

de usuarios del sistema.

Cada campo posee 3 bits, uno para **lectura (r, read)**, otro para **escritura (w, write)** y otro para **ejecución (x, eXecution)**. Estos 3 bits se suelen representar por **rwx**, y cada fichero o directorio tendrá o no tendrá estos permisos (cuando no se tienen, se indica con un guión), para cada campo (propietario, grupo y resto), como muestra la figura 4.9.

Para ficheros, estos bits indican:

- r permiso para leer el fichero, ver su contenido, copiarlo...
- w permiso para escribir en el fichero o borrar información de él, o sea, modificarlo.
- x permiso para ejecutar el fichero, caso de que este sea un programa ejecutable.

Así, por ejemplo, un fichero con las siguientes protecciones `rwx r-x --x`, indica que el propietario puede tanto leer, como escribir o ejecutar su fichero, los miembros de su grupo pueden leerlo o ejecutarlo, pero no modificarlo (escribir en él), y el resto de usuarios sólo pueden ejecutarlo.

El propietario del fichero, puede, en cualquier momento, modificar esos 9 bits de protección, con el comando `chmod` (también los puede modificar el superusuario). Por ejemplo, supongamos que el propietario los cambia a `r-x r-- ---`. Esto indica que el propietario no puede escribir o modificar el fichero. Esto lo puede hacer para evitar que él mismo, accidentalmente, pueda borrar el fichero. Los miembros de su grupo solo pueden leer el fichero, de forma que si desean ejecutarlo, pueden copiarlo a su directorio (home) y ejecutarlo desde allí, ya que al copiarlo, pasan a ser propietarios de la copia, y pueden cambiarle los permisos. El resto de usuarios no pueden acceder al fichero para nada (no tienen ningún permiso).

Para **directorios**, los permisos varían su significado:

r Indica si podemos ver qué ficheros hay en dicho directorio (leerlo o listarlo).

w Indica si podemos crear, borrar o mover ficheros en el directorio. Este permiso no tiene sentido si no tenemos el permiso x en ese directorio.

x Indica si podemos acceder a ese directorio (cd). Se suele llamar permiso de búsqueda, y para modificar un fichero es necesario tener permiso x en todos los directorios, hasta llegar a dicho fichero (en todos los directorios de su path). Este permiso es necesario para leer, escribir y ejecutar los ficheros del directorio en cuestión (además de los permisos de cada fichero).

Visto esto, podemos aclarar que de nada nos sirve tener los permisos rwx en un fichero si no tenemos el permiso x en el directorio que lo contiene.

Es obvio, que para copiar un fichero, necesitamos tener el permiso de lectura en el fichero, ya que para copiarlo, hay primero que leerlo y también necesitamos el permiso x en el directorio.

Como ya hemos visto más arriba, la primera operación que se debe hacer con un fichero es abrirlo, que es cuando se comprueba si el usuario que intenta abrir el fichero tiene los permisos suficientes para hacerlo. Si el acceso no está permitido, al abrir el fichero nos dará un error, y si estamos autorizados nos devolverá un valor denominado **descriptor de fichero**, el cual se utilizará, necesariamente, para todas las operaciones siguientes, ya que este descriptor es como una puerta por la que acceder al fichero.

Hay un tipo de **ficheros especiales**, llamados **ficheros de dispositivo**, que sirven para que los dispositivos o periféricos de E/S parezcan y se comporten como ficheros. Hay dos tipos:

- **Ficheros especiales de bloques:** Representan dispositivos en los que se accede a ellos a través de bloques de información de un tamaño fijo de bytes. Por ejemplo, tenemos las unidades de disco, lectores de CD-ROM, modems...

- **Ficheros especiales de caracteres:** El trasvase de información entre estos dispositivos se realiza carácter a carácter. El ejemplo más típico lo forman los terminales de un ordenador (pantalla de texto y teclado), pero también están algunas impresoras, interfaces a redes de comunicaciones...

Estos ficheros especiales se comportan, exactamente, como ficheros, teniendo también sus 9 bits de protección, y para leer o escribir en ellos es tan fácil como leer o escribir de un fichero. Por ejemplo, para mandar un fichero a la impresora, lo que hay que hacer es copiar el fichero en cuestión en el fichero del dispositivo de la impresora (y tener permiso r en ese fichero de impresora).

Los ficheros especiales tienen un número de dispositivo principal, que indica el tipo de dispositivo (disco duro, impresora, modem...) y un número de dispositivo secundario, que indica a qué periférico de ese tipo nos referimos (pues en el sistema puede haber varias impresoras, varios modems...). Los periféricos con igual número de dispositivo principal, comparten las

Descriptor de fichero	Dispositivo especial	Nombre técnico
0	Entrada estándar	stdin
1	Salida estándar	stdout
2	Salida de errores	stderr

Tabla 4.2: Ficheros de E/S estándar.

mismas rutinas manejadoras de dispositivo, es decir, son manejados por los mismos programas. Lo que variará es el periférico concreto sobre el que actúen esos programas (que es lo que indica el número de dispositivo secundario).

Normalmente, los ficheros de dispositivo suelen estar en un directorio específico llamado `/dev` (*device*).

Hay unos descriptores de fichero especiales para unos dispositivos especiales (o estándares), conocidos como Entrada/Salida estándar (ver tabla 4.2).

En el tema de periféricos (cuando se habló del teclado y del monitor) ya se explicó lo que eran la entrada y salida estándar, las cuales se refieren a los dispositivos donde el ordenador (y sus procesos) leen y sacan información, por defecto, es decir, mientras no se indique lo contrario.

Por tanto, la entrada estándar suele ser el teclado, y la salida estándar la pantalla o monitor. Existe un tercer dispositivo especial para la salida de mensajes de error (fichero especial de salida estándar de errores), que normalmente será también el monitor, pero que puede cambiarse a una impresora determinada, u otro periférico.

El S.O. debe tener mecanismos para que la entrada y salida estándar, se puedan cambiar, de forma que podamos hacer que un proceso no lea de teclado, sino que tome sus datos de un fichero previamente creado, y las salidas no las dé en el monitor sino en un fichero que se deberá crear. Este tipo de operaciones se suelen hacer, con las **redirecciones**, usando los símbolos de dirección > (para redirigir las salidas) y < (para las entradas). Por ejemplo, si queremos que un proceso (`prog`) lea de un fichero (`FichEnt`) como si fuera la entrada estándar, y escriba en otro fichero (`FichSal`) como si se tratara de la salida estándar, podemos poner:

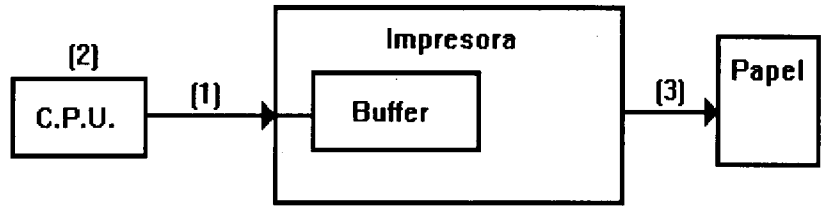
```
prog < FichEnt > FichSal
```

Podemos usar la redirección doble >> para redirigir la salida estándar a un fichero y evitar que se borre el contenido actual. Con la redirección normal (>), se borra el contenido del fichero y se introduce la salida del comando. Con la doble redirección (>>), la salida producida se **añade** al final del fichero indicado. Por ejemplo, si deseamos ejecutar de nuevo el fichero anterior, con una nueva entrada, pero añadiendo la salida al mismo fichero anterior, podemos poner:

```
prog < FichEnt2 >> FichSal
```

También el S.O. debe permitir conectar la salida estándar de un proceso con la entrada estándar de otro proceso, a través de las llamadas **tuberias** o tubos (*pipes*). Una tubería o tubo, podemos verlo como una especie de fichero en el que escribe un proceso (`procesoA`) y del que lee otro proceso (`procesoB`), como si fueran la salida y la entrada estándar respectivamente. Esto se suele hacer con el símbolo de tubería |, de la siguiente forma:

```
procesoA | procesoB
```



(1) La CPU manda el trabajo al buffer de la impresora.
(2) La impresora imprime.
(3) La CPU sigue con su trabajo.
Gracias al buffer, los pasos (2) y (3) se realizan en paralelo (a la vez).

Figura 4.10: Funcionamiento del buffer de la impresora.

4.4 Módulos de un S.O.

Como sabemos, un S.O. se encarga de gestionar el sistema informático, para que todo funcione de la mejor forma posible. A grandes rasgos, un S.O. se puede dividir en 3 módulos, dependiendo de la parte que se encarga de gestionar:

- Módulo de gestión de datos.
- Módulo de gestión de procesos (jobs o trabajos).
- Módulo de gestión de recursos.

4.4.1 Módulo de gestión de datos.

Controlan el movimiento de datos entre todos los componentes de ordenador, principalmente hacia o desde los periféricos, pero también entre memoria, CPU... Para los periféricos suelen existir unas áreas de memoria temporales o buffers, donde se almacena temporalmente la información que se ha leído del periférico, o que se debe escribir en él. Así, la CPU escribe (o lee) en el buffer, que al ser memoria es más rápido que si tuviera que escribir (o leer) directamente en el periférico, y el periférico lee (o escribe) la información del buffer a la máxima velocidad que pueda hacerlo.

De esta forma, por ejemplo, para imprimir un listado por impresora, la CPU escribe en el buffer de la impresora, y mientras la impresora se pone a leer del buffer e imprimir, la CPU puede realizar otras operaciones (ver figura 4.10). En ocasiones, el buffer de la impresora es controlado por la propia impresora.

También, al leer un dato del disco, no sólo se lee el dato deseado, sino que tiene, forzosamente, que leer un bloque completo, que lo almacenará en el buffer de disco en memoria RAM. Si acto seguido el programa tiene que leer otro dato y coincide que está en el mismo bloque, no tendrá que leer de disco para leerlo, sino que sólo leerá del buffer.

Este módulo es el encargado, también, de gestionar el sistema de ficheros (**File System**), de la forma que hemos visto en el apartado anterior. Además, debe de controlar el modo de acceso físico a cada uno de los ficheros, de forma que debe mantener una tabla, llamada **FAT** (*File Allocation Table*) (en sistemas tipo DOS) o zona de **I-nodos** (en sistemas tipo UNIX), en el que aparece la posición física de cada fichero en disco. Cuando un programa o usuario

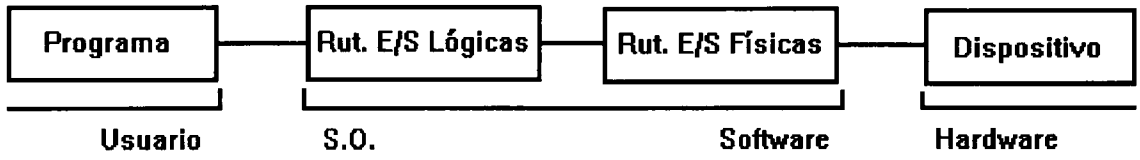


Figura 4.11: Control de E/S de un sistema.

accede a un fichero, el S.O. consulta esta tabla primero, y luego va a la zona del disco que indique, para operar con el fichero.

Podemos esquematizar, diciendo, que un programa o usuario dice al S.O. qué quiere hacer, este llamará a unas rutinas de E/S lógicas que indican lo que se desea hacer, pero no cómo hacerlo. Estas rutinas ejecutarán unas rutinas de E/S físicas, que indican, a más bajo nivel, cómo se deben efectuar las operaciones, cómo comunicarse con cada periférico (con el hardware). Esto se muestra en la figura 4.11.

4.4.2 Programa Monitor (gestión de trabajos y recursos).

El programa monitor es el encargado de gestionar la realización de operaciones básicas del sistema, controlar los trabajos en ejecución y los recursos disponibles. Debido a la importancia de estas tareas, este programa suele residir total o parcialmente en la memoria principal (la RAM) del ordenador, ocupando un trozo de esta memoria, para controlar el resto del sistema. En muchos casos a este programa Monitor se le suele llamar como si fuera por sí mismo el S.O.. También se le llamar **núcleo** o **kernel**, por ser, en todo caso, el corazón básico de todo S.O.

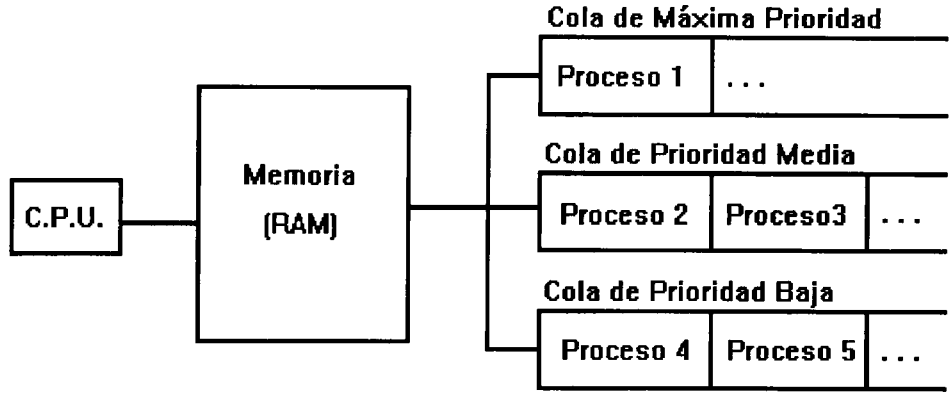
Podemos dividir este programa en dos módulos:

4.4.2.1 Módulo de gestión de trabajos.

La planificación y control de los trabajos, *jobs*, programas en ejecución o procesos, depende siempre del S.O. empleado. Existen multitud de técnicas, de las cuales, las más importantes ya las hemos visto en el apartado de *Evolución de los S.O.*

En cualquier caso, la planificación de trabajos pretende obtener la utilización más eficiente del sistema. Para esto, se suelen asignar prioridades a cada proceso, de forma que los procesos con mayor prioridad se ejecuten antes que los de menor prioridad. Según esto, hay varios modos de llevarlo a cabo, pero antes de verlos es importante aclarar que en el computador, puede haber varios procesos ejecutándose a la vez (cargados en memoria), usando algún sistema (como multiprogramación, tiempo compartido...) y lo que se trata ahora es de ver qué procesos empiezan a ejecutarse antes:

- **Secuencial:** Todos los procesos tienen igual prioridad, por lo que se ejecutará antes el que llegue antes a la cola de espera.
- **Con colas de prioridad:** En este caso, tenemos varias colas de trabajos, cada una con una prioridad, de forma que se ejecutarán antes los trabajos de la cola con mayor prioridad. Cuando no existan trabajos en esa cola, se ejecutarán los de la cola siguiente en prioridad, y así sucesivamente (ver figura 4.12).



NOTA: El número del proceso indica el orden de ejecución, suponiendo que no llegue ningún otro proceso a ninguna cola.

Figura 4.12: Colas de prioridad en la ejecución de procesos.

Si durante la ejecución de un proceso, llega uno con mayor prioridad que precisa ser ejecutado, podemos optar por dos sistemas:

- Esperar a que algún proceso en ejecución termine.
- Almacenar el estado de algún proceso (de menor prioridad), para seguir con él posteriormente por donde nos quedamos, y dejar paso así al proceso con mayor prioridad.

- **Prioridad con limitación:** Podría ocurrir, que un proceso con baja prioridad no se ejecutase nunca, si siempre hubiera procesos con mayor prioridad. Con este sistema, cuando un proceso supera un tiempo establecido de espera máximo, se eleva su prioridad, pasando así a la siguiente cola de prioridades, donde ocurrirá igualmente, si no es ejecutado en el tiempo máximo establecido para esa cola.

- **Prioridad con limitación y ejecución automática:** Es una variación del método anterior. Aquí, cuando un proceso espera su tiempo máximo, se le asigna la máxima prioridad, de forma que pasará a ejecutarse inmediatamente.

- **Prioridades por uso de recursos:** El S.O. asigna una prioridad a cada proceso en función de los recursos que usa (impresora, discos, cintas, CPU, coprocesador...) y del tiempo de uso de estos, de forma que se ejecutará antes el que use menos recursos, o los use menos tiempo, y aquel que los recursos que use estén libres en ese momento. De esta forma, se consigue el mejor aprovechamiento de los recursos disponibles. Este sistema es complicado, porque suele ser difícil saber *a priori* cuales recursos se van a usar y por cuanto tiempo.

4.4.2.2 Módulo de gestión de recursos.

Una labor fundamental del S.O. es la gestión y asignación de los recursos del ordenador a los procesos en ejecución, de forma que esta asignación sea coherente y no se produzca

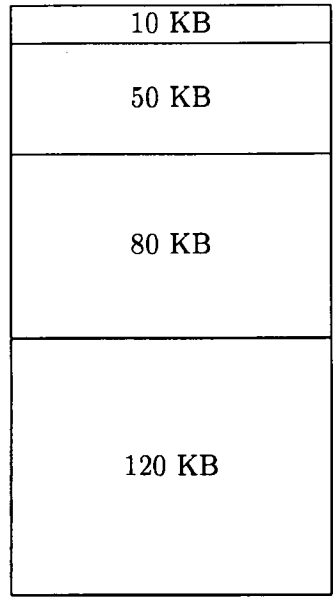


Figura 4.13: Disposición de la memoria, por particiones fijas

un interbloqueo. Imaginemos que un proceso tiene asignada la impresora y para continuar necesita que se le asigne el acceso al lector de CD-ROM, que lo tiene asignado otro proceso, esperando a que le asignen la impresora para liberarlo. Este tipo de situaciones, sin solución, deben ser evitadas a toda costa.

Hay S.O. en los que no se ejecuta un trabajo si no están todos los recursos que este necesita libres. Esto provoca una pérdida tremenda de tiempo de CPU, por lo que es mejor hacer una **asignación dinámica de recursos**, de forma que un recurso se asigne a un proceso sólo y exclusivamente en el momento de ser utilizado. De esta forma, mientras el recurso puede ser usado por un proceso, la CPU será utilizada por otro proceso.

Normalmente, el recurso más caro y por el que los procesos compiten más duramente es la **memoria**, ya que es necesario que un proceso esté cargado en la memoria para que se ejecute. Por tanto, para ejecutar un programa, es necesario que exista tanta memoria libre como el tamaño de dicho programa, cargarlo en memoria, ejecutarlo y cuando termine, la memoria se devolverá al sistema, quien la podrá asignar a otros procesos. El método de asignación de memoria puede ser muy diverso, pero los métodos más usuales son:

- **Por particiones fijas:** La memoria global se divide en trozos o particiones fijas de distintos tamaños. Cuando un programa se va a ejecutar, se le concede la partición más pequeña que lo pueda contener, y que esté libre.

  Ejemplo: Supongamos la partición de memoria que aparece en la figura 4.13.

  Si en esta situación de particiones, se quisiera ejecutar un proceso de 52 KB de tamaño, habría que asignarle la partición más pequeña que puede contenerlo, es decir, la de 80 Kbytes, desperdiciando así 28 KB de memoria principal.

- **Montón o poll de memoria:** La memoria se mantiene intacta, sin particionarla, y a cada proceso se le asigna la cantidad de memoria necesaria para contenerle. Este sistema es muy bueno, porque no se desperdicia memoria, pero los problemas de protección, de la

zona de memoria de un programa, son más complejos, ya que los límites de las particiones varían con el tiempo.

- **Segmentación de páginas (paginación):** Aquí la memoria se divide en **páginas** de igual tamaño, y pequeña dimensión. Igualmente, al cargar un programa para su ejecución, se divide este en páginas del mismo tamaño, y sólo será necesario que estén en la memoria principal, aquellas páginas del programa que se estén ejecutando en cada momento. Este método no desperdicia memoria, y permite tener muchos procesos cargados en memoria, ejecutándose a la vez. El inconveniente es que cada vez que se termina una página (y esta no está en memoria), hay que leer de disco la siguiente, por lo que las lecturas a disco son frecuentes, y para evitar retardos muy grandes, el disco (u otro almacenamiento secundario) debe ser de acceso muy rápido.

- **Memoria virtual:** Ya se definió este concepto en el tema que se habló sobre la memoria. La idea fundamental es, puesto que la memoria RAM es cara y escasa, podemos usar memoria de disco, simulando que ésta es memoria RAM. De esta forma, se reserva un espacio del disco para tal fin, y la memoria principal, parecerá más grande de lo que realmente es. El disco debe ser de acceso muy rápido, pero, aún así, la ejecución de los procesos utilizando memoria virtual es más lenta, ya que el acceso a un disco es siempre más lento que el acceso a la RAM.

4.5 Arquitecturas típicas de los S.O.

Con este nombre, nos queremos referir a los tipos de S.O., según el modo en el que se hayan programado. No se trata de las operaciones que realicen los S.O., sino del modo de trabajo que tengan, de su estructura interna.

4.5.1 S.O. Monolíticos.

En este tipo de S.O. no hay una clara estructura, sino que hay un buen montón de rutinas (o procedimientos), en los que cada uno puede llamar al resto cuando quiera. Debido a su falta de estructura su modificación y depuración son bastante complejas.

Las llamadas al sistema (**trap**) o llamadas al núcleo, las realiza un programa en ejecución (dejando los parámetros en algún lugar conocido por el S.O., registros, pila...). En ese momento, la máquina cambia de modo usuario a modo privilegiado (o supervisor) y pasa el control al S.O., quien examina esos parámetros y determina el número de la rutina (o servicio) del sistema a ejecutar. Acto seguido mira en una tabla indexada por el número de llamada, la posición de dicha rutina y se va a ejecutarla. Cuando termina su ejecución se devuelve el control al programa de usuario que efectuó la llamada, para que continúe.

Por tanto, aunque hemos destacado la falta de una estructura clara en este tipo de S.O., es básico que debe existir:

1. Un programa principal, que captura las llamadas al sistema y realiza las llamadas oportunas.

2. Un conjunto de procedimientos que implementan todas las llamadas (o servicios) al sistema posibles. Algunos, podrán efectuar llamadas a otras rutinas de este mismo nivel.

NIVEL	Procedimientos
0	Asignación del procesador, multiprogramación básica. A partir de aquí, ya no es necesario que los procesos sepan que hay varios ejecutándose en la misma máquina.
1	Gestión de memoria principal. Elimina de los procesos la complejidad que supone el control de la memoria.
2	Comunicación entre procesos.
3	Control de dispositivos de E/S y sus transferencias. El objetivo era que los procesos vieran a los dispositivos como fáciles de manejar, haciendo transparentes sus características hardware. Los dispositivos se verán como si fueran ficheros, con características especiales, como hemos visto.
4	Formado por los programas de usario.
5	Proceso del operador del sistema.

Tabla 4.3: Niveles del S.O. THE.

3. Un conjunto de procedimientos básicos (o auxiliares), que son utilizados por las llamadas anteriores e incluso por otros procedimientos básicos de este nivel.

4.5.2 S.O. en Niveles.

Se basa en la estructura citada del anterior, pero llevada más coherentemente. Se trata de construir una jerarquía de niveles, cada uno de los cuales está construido sobre el anterior, de forma que llama a procedimientos del nivel anterior y sus rutinas son llamadas por procedimientos del nivel posterior.

Un ejemplo, es el S.O. THE, fabricado en Holanda por Dijkstra (1968), un S.O. ya arcaico, con 6 niveles, que están detallados en la tabla 4.3.

Otro S.O. más famoso, también en Niveles, fue el MULTICS, ya comentado anteriormente.

4.5.3 Los S.O. como Máquinas virtuales.

Un grupo de científicos del centro de investigación de IBM, produjo un S.O., que IBM adoptó como S.O. en tiempo compartido y que aún hoy se emplea. Se llamó CP/CMS, pero hoy se conoce como VM/370 (1979). Este original S.O. se basaba en la idea de separar las dos funciones básicas de un S.O. en tiempo compartido: por un lado multiprogramación y por otro una máquina virtual, con un interfaz fácil de manejar.

El llamado **monitor de la máquina virtual**, se ejecuta directamente sobre el hardware y se encarga de la multiprogramación. De esta forma ofrece al nivel superior no una, sino varias máquinas virtuales. Estas máquinas no son extensiones de la máquina real, sino que

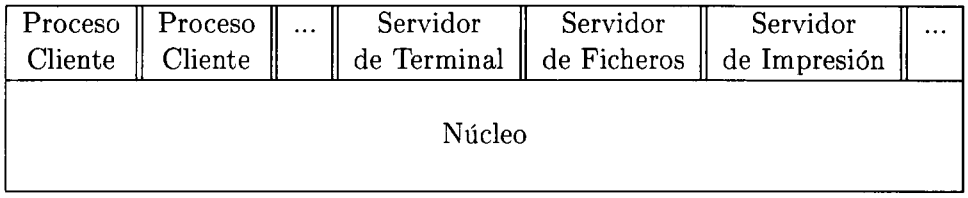


Figura 4.14: Estructura de un S.O. con Modelo Cliente-Servidor

son como si realmente hubiera varias máquinas físicas (con sus modos normal y privilegiado, E/S, interrupciones...), exactamente igual que la máquina que realmente existe. Por tanto, cada una de estas máquinas (virtuales) puede tener un S.O. distinto.

Cuando se produce una llamada al sistema en una de estas máquinas virtuales, la llamada es recibida por el S.O. de esa máquina virtual, quien la pasará a VM/370, que la ejecutará como parte de su simulación del auténtico hardware físico de la máquina virtual.

4.5.4 El Modelo Cliente-Servidor.

Para que el S.O. en sí, sea simple, y por tanto se ejecute eficientemente, se suelen pasar partes del sistema a niveles superiores, dejando reducido el S.O. a su mínima expresión, quedando muchas funciones del S.O. como procesos de usuario. Para solicitar cualquier servicio, estos procesos de usuario, o **clientes**, envían peticiones al proceso **servidor**, que realiza la función solicitada, devolviendo su respuesta. Dependiendo del tipo de servicio requerido se solicitará la ejecución de uno u otro proceso servidor.

De esta forma, la única función del **núcleo** (kernel) del S.O. es comunicar a los procesos clientes (o de usuario) con los procesos servidores (o del S.O.). Por tanto, un cliente obtiene servicio enviando mensajes, a través del núcleo, a los procesos servidores. De la misma forma, un proceso servidor del S.O. puede ser, a la vez, cliente de otro servidor distinto.

Como se ve en la figura 4.14, el S.O. queda dividido en servidores de ficheros, de procesos, de terminales, de memoria, de impresión... consiguiendo que cada parte sea más pequeña, más manejable y más fácil de realizar, modificar o cambiar. Así puede fallar uno de los servidores, pero la máquina seguir funcionando en el resto de sus funciones, ya que el único módulo que se ejecuta en modo privilegiado es el **núcleo** (que no debe fallar).

De todas maneras, algunas funciones del S.O. son difíciles o imposibles de hacer en programas trabajando en modo usuario, por lo que algunos servidores críticos (manejadores de dispositivos de E/S...), podrían ejecutarse en modo privilegiado, lo que les da un acceso completo al hardware. Otra solución es incluir en el núcleo alguna forma (mandando un determinado mensaje, por ejemplo), para que estos servidores críticos pudieran mandar al núcleo que ejecutase ciertas cosas que sólo son posibles en modo privilegiado.

Una ventaja de este tipo de S.O. es su fácil adaptación a **sistemas distribuidos**, en los que cada parte del S.O. (cada servidor) puede ejecutarse en una máquina distinta, conectadas, todas ellas, en una red. A un proceso cliente le da igual que sus servidores estén ejecutándose en la misma máquina o en otra, siempre y cuando este le sirva sus peticiones (aunque sea a través de una red de ordenadores).

Como vemos, no es necesario que un servidor se ejecute exclusivamente en una máquina, sino que en una máquina pueden coexistir varios procesos, clientes o servidores, indistintamente.

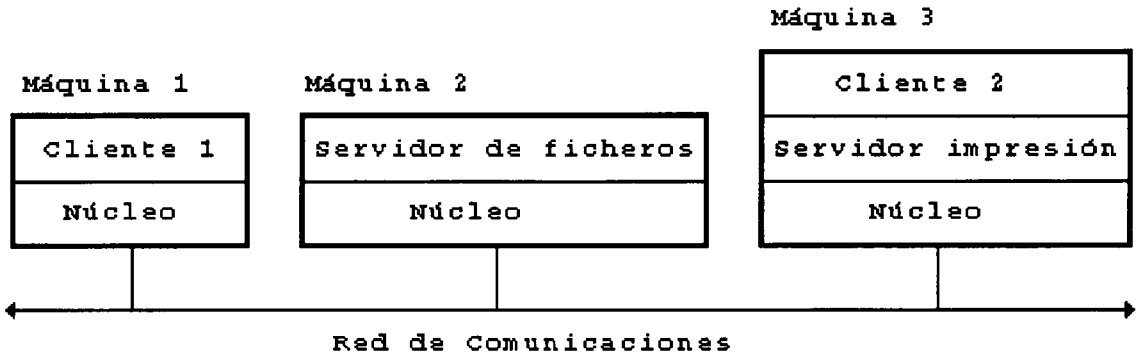


Figura 4.15: Modelo Cliente-Servidor en un sistema distribuido.

4.6 Algunos S.O. comerciales.

La bajada de precios de los ordenadores personales, ha facilitado no solo la expansión de éstos, sino la proliferación de muchos S.O. en busca de un lugar en el mercado. Son muchos: MS-DOS, OS/2, UNIX, VMS, MacOS (el S.O. de los Macintosh), Windows95, Windows NT... Aquí veremos, muy brevemente, algunos de los S.O. más extendidos, y algunas de sus características:

- El DOS.
- UNIX y Linux.
- El OS/2.
- Windows95 y NT.

Todos los S.O. tienen sus ventajas y sus desventajas y no debemos olvidar que aún no se ha construido un S.O. a prueba de todo tipo de fallos, por lo que cualquier S.O. lo podremos llevar a los llamados *fallos generales de protección* (GPF) o *bombas del sistema* (los famosos *cuelgues*), y es que... no se puede tener todo en la vida.

También hay que indicar que existen multitud de programas simuladores que nos permiten ejecutar programas para un S.O. en otro S.O.

De todas formas, es curioso observar que los Sistemas Operativos están desfasados con respecto al hardware que va saliendo. Por ejemplo, podemos invertir mucho dinero en un ordenador PC muy potente con más de 4 procesadores Pentium con muchos megaherzios (que ya existen en el mercado), pero ningún S.O. sabrá sacarle el máximo partido. También es cierto que, en general, los programadores no suelen optimizar muy bien su código para conseguir programas rápidos y que requieran poco espacio en disco y memoria, por lo que muchos programas, para que se ejecuten suficientemente bien requieren varias generaciones de procesadores posteriores a la actual.

Quizás una de las características que más preocupa al usuario a la hora de instalar un S.O. es la cantidad de espacio que este necesita, tanto de espacio en disco duro como de espacio de memoria RAM mínima para funcionar. Naturalmente, el espacio en disco duro depende de si instalamos todas o parte de las aplicaciones que se incluyen siempre con el

Sistema Operativo	Espacio mínimo en disco duro	Espacio máximo en disco duro	Memoria mínima
MS-DOS	0.6	5	0.5
Windows95	20	40	4
OS/2	20	50	4
NT Workstation	60	80	12
NT Server	70	90	16
Linux	20	3 GB	4

Tabla 4.4: Espacio (en MB) de disco y memoria que necesitan algunos S.O.

sistema operativo. El caso más exagerado es el del S.O. Linux ya que existen multitud de aplicaciones que podemos instalar con él (hasta más de 3 GB). En cuanto a la cantidad de RAM, en general, cuanto más mejor, sin olvidar que depende del uso que vayamos a dar al sistema. En la tabla 4.4 exponemos un resumen de cuánto espacio necesitan algunos de los S.O. más usuales.

4.6.1 El DOS.

Bajo este nombre, que viene de las siglas *Disk Operating System*, se engloban unos S.O. encabezados por el MS-DOS, de MicroSoft (de ahí la MS), el cual, nació a principio de los años ochenta, y fue el S.O. de los primeros PC's de IBM, con procesador 8088, por lo que es un S.O. **monousuario**, esto es, pensado para que se ejecute en un ordenador personal, atendiendo las órdenes de un único usuario.

Como casi todos los programas informáticos, han ido apareciendo sucesivas versiones, y fue mítica, por su enorme expansión la versión 3.3. Actualmente la última versión es la 6.22. y parece ser que será la última, debido a que este S.O. tiene unos problemas, de difícil solución, ya que la idea de cada versión es que fuera compatible con las versiones anteriores y con los procesadores anteriores.

El principal defecto de este S.O. es su pésima gestión de la memoria principal, que utiliza los primeros 640 KB, la llamada **Memoria Convencional**, como la única en la que se pueden ejecutar programas (en teoría, porque en la práctica se pueden usar ciertos trucos), luego 384 KB de **Memoria!Superior**, inmanejable por los programas y que queda reservada para controladores de dispositivos. El resto, a partir de ese megabyte (1024 KB) es la llamada **Memoria extendida** y que este S.O. no controla bien, pudiéndose dar el caso de tener 4 MB (3 de extendida), y no poder cargar varios programas en memoria por no caber estos en la memoria convencional.

No vamos a meternos en más detalles, sólo comentar que además de monousuario es **monotarea**, que indica su imposibilidad de trabajar con varios programas a la vez (hay formas de hacerlo, pero son complicadas para el usuario).

Están formados por dos conjuntos de programas:

- **Programas residentes en memoria:** Se cargan en memoria al arrancar el ordenador y permanecen siempre allí (como extensión del propio S.O.). Tienen los programas más usados por los usuarios. El más famoso es el intérprete de comandos, bajo el nombre de

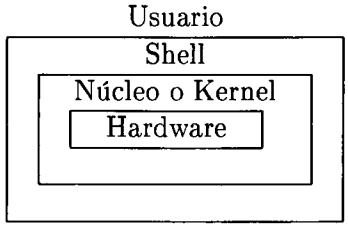


Figura 4.16: Capas del S.O. UNIX

COMMAND.COM, que ejecutará los llamados comandos internos, que son comandos básicos y muy usuales (`dir`, `cd`, `md`, `rd`, `del`, `copy`, `ver`, `time`, `date`...).

- **Programas no residentes:** Son el resto de programas que se facilitan con el S.O. No pueden cargarse todas en memoria porque, de ser así, no habría memoria libre para los programas de usuario. Se suelen almacenar en un directorio llamado DOS, en forma de ficheros de programas ejecutables (.EXE o .COM) y contienen órdenes para manejo de ficheros, discos, controladores de dispositivos (.SYS)... y en las últimas versiones tienen duplicadores de la capacidad del disco, optimizadores de memoria, programas antivirus, reparadores de discos estropeados, programas para conexión de ordenadores y otros programas de gran utilidad. Ejemplos de estos comandos son: `format`, `diskcopy`, `diskcomp`, `xcopy`, `msav`, `msd`, `attrib`, `dblspace`, `msbackup`, `scandisk`, `interlnk`, `intersvr`...

Hay otros S.O., aparte del MS-DOS, que son compatibles con este, y su diferencia principal es que son de distinto fabricante: PC-DOS, de IBM y DR- DOS de Digital Research.

MicroSoft desarrolló un shell de este S.O., en entorno gráfico, fácil de manejar usando un ratón, el **Windows**, del que la versión más extendida es la 3.1, pero no hay que olvidar que este no es un S.O. sino un interfaz gráfico, con algunas funciones de S.O., como la gestión de la memoria, que gestiona mejor que el propio MS-DOS.

4.6.2 UNIX y Linux.

En la tercera generación ya vimos su nacimiento, y a lo largo del tema hemos ido explicando muchas de sus características. Es un S.O. potente, **multitarea** y **multiusuario**, ideal para ejecutarse en grandes ordenadores, atendiendo a la vez a cientos de usuarios.

Es un S.O. en Niveles, en los que básicamente tenemos los siguientes niveles (ver figura 4.16):

El **núcleo** o **kernel**, es la parte del S.O. que trabaja directamente con el hardware del ordenador. Sus funciones básicas son la gestión de memoria, control de acceso al sistema, mantenimiento del sistema de ficheros, manejo de las interrupciones, asignación de recursos, control de errores y gestión de entradas y salidas.

El **shell** es, como ya dijimos, el intérprete de comandos, y se encarga de leer las órdenes de los usuarios e interpretarlas (creando procesos hijos...), interactuando con el núcleo, para ejecutarlas. Además, dispone de un potente lenguaje de programación. Hay muchos shells diferentes, por lo que cada usuario escogerá el que más le guste.

Existen multitud de versiones de este S.O., de distintos fabricantes, siendo la más usada la versión 4 de System V. Además, hay versiones que se ejecutan en un PC de la familia

x86, a partir del 386, como el Coherent, el XENIX de Santa Cruz Operation (SCO), y el más difundido, el **LINUX**, que es un S.O. de dominio público, es decir, gratuito.

El **LINUX** es un S.O. robusto en el que han trabajado desinteresadamente muchos ingenieros y programadores aficionados, coordinados por Linus Torwald (sobre todo en la programación del Kernel). Es, sin duda, el mejor S.O. de 32 bits que se puede adquirir por unas 3000 pesetas (el precio del soporte). Su éxito va en aumento, sobre todo en entornos universitarios, los cuales, suelen ser bastante exigentes a la hora de decidirse por un sistema u otro. Además, para LINUX se pueden encontrar miles de *megas* de programas, para todo. La mayoría de las distribuciones LINUX se venden en 4 CD-ROM llenos de software de todo tipo. Lo peor de este S.O. es que es un poco dificil de configurar y al principio suele dar algún quebradero de cabeza, no apto para impacientes. ¡Ah! y también incluye un sistema gráfico para X-Windows, y su emulación de Windows (`wine`) está en fase beta.

4.6.3 El OS/2.

Un S.O. muy potente, que debería haber sido el sucesor de MS-DOS, pero que no ha tenido demasiado éxito comercial, y es que... no siempre lo mejor es lo más vendido.

Una de sus últimas actualizaciones, llamada **OS/2 Warp 3 Connect**, aparecida a finales de 1994, añade multitud de ventajas a las anteriores. Es muy superior al MS-DOS, pensemos que este último se distribuye en unos 4 discos (de 1,4 MB) y el OS/2 en 36 discos formateados a 1,8 MB (14 del propio S.O., 7 de drivers, y 15 del *BonusPak*).

Este es un S.O. **multitarea** (permite tener varios programas cargados en memoria a la vez) y **multiproceso** (permite que varios programas se puedan ejecutar a la vez (con un sistema de prioridades) sin que sea de forma interactiva con el usuario.

Aunque normalmente no hay confusión posible, algunos autores, consideran que para que se llame multiproceso deben existir varios procesadores, así que, cuando aquí decimos multiproceso se debe entender como tiempo compartido, con prioridades, ya que para que haya dos procesos ejecutándose exáctamente en el mismo tiempo es necesario tener al menos dos unidades de ejecución (dos CPU, aunque a veces se integran en la misma CPU varias unidades de ejecución).

Así, con este S.O. se puede formatear un disco, mientras mandamos un fax, visualizamos una película en una ventana y jugamos a un juego en otra.

Incluye un buen paquete de utilidades, llamado *BonusPak* que contiene: Programa de comunicaciones por modem, programa de acceso a Compuserve, programas de acceso a **Internet** (correo electrónico, ftp para traernos ficheros y acceso a las *news* –noticias– de todo el mundo), soporte multimedia y el paquete IBM Works con una hoja de cálculo, procesador de textos, manejo de gráficos, base de datos y hasta generador de informes.

4.6.4 Windows95 y NT.

Antes de la salida de Windows95, OS/2 Warp Connect era el mejor medio para ejecutar, en multitarea los programas Windows de 16 bits. El problema es que no puede ejecutar aplicaciones Windows95 que son las que están siendo las últimas en salir.

Windows95 tiene sus fallos, como cualquier otro sistema operativo. Nada es perfecto. Quizás, la causa que más críticas le haya ocasionado es que MicroSoft vendiera la imagen de sistema operativo perfecto, cuando no lo es, ni de lejos. Los típicos errores de protección general de versiones anteriores siguen existiendo, aunque es más difícil que ocurran.

Windows95 requiere un mínimo de 8 MB de RAM, en un 486, para que pueda ejecutarse con cierta soltura. Es multitarea y soporta multienhebramiento (*multithread*) y con su sistema de archivos VFAT, admite nombres de archivos largos. Puede comunicarse con redes de muy variada filosofía (TCP/IP, IPX/SPX...), soporta correo electrónico, acceso telefónico a redes...

Para funcionar realmente bien requiere aplicaciones de 32 bits, que, aunque en principio pueda parecer demasiado restrictivo, en realidad, no lo es, ya que incluso la siguiente generación de procesadores (Pentium Pro), requieren aplicaciones de 32 bits para funcionar optimamente.

El problema radica en que aún arrastra código de 16 bits, lo cual arrastra consigo algunos de los problemas de la versión anterior (la 3.11). Quizás, la culpa de esto fue darse demasiadas prisas en comercializarlo, para que los usuarios no migraran al S.O. de IBM, OS/2 Warp.

Además, el sistema Plug and Play no funciona como debería, y sigue teniendo el problema de la mala administración y gestión de la memoria, por lo que resulta relativamente simple *colgar* el sistema: Para ello basta con abrir y cerrar aplicaciones hasta que se cuelgue. Al abrir una aplicación, el S.O. le asigna unos recursos (principalmente memoria), que al cerrarla no libera totalmente, por lo que tras varias de estas operaciones el sistema se queda sin recursos suficientes y muere.

Lo mejor es su interface de usuario (similar al Macintosh) y que permite ejecutar en multitarea aplicaciones de 32 bits, y es compatible con los programas existentes de 16 bits, aunque no al 100%.

El futuro está puesto en Windows NT, que es un S.O. mucho más robusto y potente, pero que los requisitos hardware son demasiado elevados, por ahora, para un ordenador de tipo personal. Por eso, actualmente está reservado exclusivamente como servidor de tipo empresarial. Quizás, dentro de algunas versiones del 95, los usuarios dispondrán de suficiente máquina para pasarse a NT.

Por tanto, el 95 puede verse como un S.O. de transición hacia NT. Si Microsoft no hubiera sacado está versión temporal, hubiera perdido mucha cuota de mercado en favor de OS/2.

Algunas de las características de Windows NT son las siguientes: Es un S.O. completo y autónomo, totalmente 32 bits, Compatibilidad (con programas DOS y Windows en cualquier versión), Multiplataforma, es decir, Transportable a cualquier máquina (por eso está programado casi integramente en C), Fiable (para que no se caiga el sistema si falla una aplicación), Distribuible (para distribuir tareas en una red) y Ampliable (para que no hiciera falta recompilarse nada cuando se quiera ampliar) e interconectable (con todo tipo de redes y ordenadores).

Windows NT, requiere 16 MB de RAM como mínimo, aunque se recomiendan 32, y si vamos a ejecutar bastantes procesos, mejor 64 MB. Actualmente hay dos versiones: *Advanced Server* y *Workstation*, que son básicamente iguales, salvo que la primera tiene funcionalidades para hacer de servidor de aplicaciones cliente/servidor, las cuales están adquiriendo cada vez más importancia.

Capítulo 5

Software de Aplicación

Para encuadrar el tipo de aplicación de este capítulo vamos a establecer algunas clasificaciones del software que puede utilizarse en una computadora. En la primera los programas que forman parte de una computadora pueden agruparse en tres apartados:

1. *Software de control* o *sistema de explotación*

   Es el conjunto de programas que controlan el funcionamiento de los programas que se ejecutan, y administran los recursos hardware facilitando el uso de la computadora de la forma más eficiente posible. Dentro de este apartado se incluye el sistema operativo de la computadora.

2. *Software de tratamiento*

   Incluye los programas con los que los usuarios utilizan los recursos de la computadora (hardware, sistema operativo, ...) para resolver sus problemas o realizar sus aplicaciones. Estas aplicaciones pueden ser cálculos científicos, aplicaciones de gestión administrativa, ..., etc.

   El software de tratamiento se puede dividir en dos tipos más: *sofware de programación* y *software de aplicación*. El primero es el utilizado por los desarrolladores de aplicaciones e incluye programas o utilidades que facilitan la construcción de éstas, con herramientas como intérpretes, compiladores, editores de texto, ..., etc. Y el segundo está formado por aquellos programas creados para el usuario final, donde se incluyen *procesadores de textos*, *hojas de cálculo*, *sistemas para la administración de bases de datos*, *programas de cálculo científico*, ..., etc.

   Dentro del este grupo existe una gran cantidad de paquetes comerciales para muchas aplicaciones que han ido adquiriendo una gran importancia por su difusión.

3. *Software de mantenimiento*

   En este grupo están los programas utilizados por las personas responsables del mantenimiento y puesta apunto de un sistema informático (hardware+software). Con ellos se consigue localizar averías en los circuitos de un ordenador, o encontrar las causas del mal funcionamiento de algún módulo del sistema operativo. Cuando por ejemplo, se detectan errores en el sistema operativo, es necesario rehacer los módulos que se ven afectados, y para esto se utiliza el software correspondiente de generación del S.O., que está incluido dentro de este grupo también.

Otra clasificación del software atendiendo a las áreas de utilización del mismo es la siguiente:

- *Software de sistema*, sirve para dar servicio de ejecución a otros programas y se caracteriza por la fuerte dependencia del hardware que tiene.
- *Software de tiempo real*, mide, controla y analiza sucesos del mundo real. Se caracteriza por ser un software complejo en su construcción y en muchos casos para su utilización.
- *Software de gestión*, está centrado en aquellas aplicaciones que controlan grandes ficheros de datos en las áreas del trabajo administrativo. La mayor parte del parque de ordenadores que se utilizan hoy en día están dedicados a la ejecución de este tipo de software.
- *Software de ingeniería y científico*, en esta clase se consideran todos aquellos programas destinados a resolver problemas que requieren cálculo numérico. Ejemplos son los programas de cálculo de estructuras, diseño asistido por computadora, programación matemática, simulación, ..., etc.
- *Software de inteligencia artificial*, son programas cuyo funcionamiento está basado en el uso de métodos simbólicos para la resolución de problemas, intentando asemejar su funcionamiento al comportamiento humano (cuando es inteligente). Dentro de este área están los llamados *Sistemas Expertos*: grandes programas que se utilizan en áreas de conocimiento muy bien delimitadas, que son capaces de aprender reglas dictadas por un experto, y con estas actuar como lo haría éste.
- *Software empotrado*, se incluyen aquí aquellos programas que se aplican para el control de máquinas automáticas, y que suelen residir en memorias de sólo lectura. En esta clase están los que se utilizan en el control de ciertas funciones de los automóviles y electrodomésticos.
- ...

El resto del contenido de este capítulo va a estar centrado en hablar sobre una de los grupos de aplicaciones más extendidas como son las de *ofimática*. En la primera clasificación las aplicaciones de ofimática quedan encuadradas en el apartado de software de tratamiento y específicamente en lo que es el software de aplicación. Bajo la segunda, se considera que este tipo de aplicaciones están incluidas dentro del software de gestión. En definitiva las aplicaciones de ofimática engloban todos aquellos programas que convierten el ordenador en herramienta sustituta de los utensilios y técnicas que tradicionalmente se han usado en la oficina, como la máquina de escribir, calculadoras, fichas, ..., con el objetivo de aumentar en eficiencia y comodidad.

5.1 Procesadores de Texto

Los procesadores de texto han supuesto una gran aportación a la composión de textos, tanto a nivel profesional como para la gran mayoría de usuarios no profesionales. Con este tipo de programas se puede escribir desde los típicos documentos de poca extensión (una carta, informe, ...), hasta otros con mayor numero de páginas y de mayor complejidad, es decir documentos

en los que se utilizan notas a pie de página, cabeceras y/o pies de página, ilustraciones, tablas, distintos tipos de índices, ..., etc. Esto ha conseguido, junto con la mejor tecnología existente en lo que ha sistemas de impresión de documentos se refiere (impresoras de inyección de tinta, o impresoras láser), que la tradicional máquina de escribir haya sido sustituida, especialmente en su espacio tradicional: la oficina. Por estas circunstancias en este tema vamos a estudiar ciertas generalidades que es conveniente conocer a cerca de estas herramientas.

Asociados a los procesadores de texto, y como antecesores suyos están los *editores de texto*, sus funciones consisten en ofrecer las posibilidades de poder *introducir* texto en el ordenador, grabándolo para su posterior recuperación y *corrección* o *modificación*. Estas posibilidades, y en especial la última, son notables si pensamos que con una máquina de escribir tradicional, algo tan frecuente como corregir una palabra mal escrita, un acento olvidado, o cualquier tipo de falta suponía hacer un borrón en el papel o reescribir una hoja entera. Además de esto está la circunstancia de que son muchos los casos en los que a un documento es necesario añadirle nuevos trozos de texto como resultado de la evolución de un trabajo documentado, ante esto está claro que la ventajas que aportan los editores, a pesar de sus pocas prestaciones, son mayores.

Los procesadores de texto, añadieron a las funciones de los anteriores la posibilidad de *formatear* o *componer* adecuadamente el texto. Esta función puede consistir en asignar distintas clases de letra, definir márgenes para los párrafos, numerar las páginas del documento, insertar cabeceras de página, alinear texto de distintas formas, ajustar los caracteres de cada línea al ancho de página, ..., etc. Detalles que son difíciles de conseguir con una máquina de escribir, salvo en la imprenta, pero que gracias a los procesadores de texto están plenamente disponibles.

Existe otro tipo de programa dentro del conjunto de los que permiten introducir texto en un ordenador, *los programas de autoedición*. Hoy en día, y a medida que pasa el tiempo más, resulta difícil distinguirlos, y esto es debido a que unos van invadiendo el terreno de otros. Los programas de autoedición se han caracterizado por su mayor capacidad para la obtención de documentos complejos y de alta calidad en un ordenador, mayor que los procesadores de texto. Pero esto cada día es menos cierto dado que los procesadores de texto son más potentes: admiten la inserción de muchas figuras, más cantidad de presentaciones, mayor flexibilidad, ... El entorno de utilización de los programas de autoedición se centra en la confección y diseño de publicaciones: periódicos y revistas.

Los procesadores de texto actuales ofrecen una serie de características que vamos a enumerar a continuación:

- Combinación de texto y gráficos.
- Utilización de distintos tipos de letra.
- Edición de fórmulas matemáticas.
- Construcción de tablas.
- Presentación previa a la impresión: *WYSIWYG*, que permite ver el aspecto del documento antes de imprimirlo.
- Generación de distintos tipos de índices: materias, términos, ...
- Control y ajuste de funcionamiento de una amplia gama de impresoras.

- ...

En las secciones siguientes se van a tratar los conceptos más comunmente usados en cualquier procesador de textos. Antes de comenzar queda indicar que en ciertos aspectos se hace uso de lo que aportan dos procesadores de textos muy utilizados como WordPerfect y Word.

5.1.1 Estructura del documento

Podemos considerar que un documento está constituido por los siguientes componentes en una primera aproximación:

Texto: conjunto de caracteres distribuidos siguiendo una determinada estructura de acuerdo con una serie de reglas sintácticas.

Dentro de un texto existen otros componentes que son reconocidos con su entidad propia por parte del procesador de textos utilizando las marcas típicas que los distinguen:

- *Palabras*, conjuntos de caracteres que se distinguen por los espacios en blanco que separan unos de otros.
- *Frases*, conjuntos de palabras separados por signos de puntuación.
- *Párrafos*, son frases separadas por alguna línea en blanco.
- *Secciones*, agrupan una serie de párrafos que se pueden englobar en *partes*, *temas*, *capítulos*, ..., etc. Y se distinguen entre si mediante marcas especiales que establece el procesador para distinguirlos.

Ilustraciones: engloban a todos aquellos elementos de carácter pictórico, que no se pueden obtener utilizando los símbolos normales de un determinado alfabeto. Es decir, todo aquello que implica el uso de recursos gráficos, que no suele incluir el procesador de texto. Las *tablas* se pueden incluir en esta clase, aunque en casi todos los programas actuales forman parte de las herramientas que se pueden utilizar. Igualmente ocurre con las ecuaciones, aunque son menos los programas que tienen una herramienta específica para construir ecuaciones con un aspecto de calidad. Para incorporar gráficos a un texto, el procesador ha de disponer de la posiblidad de leer ficheros de gráficos, cuestión esta que puede ser problemática dada la cantidad de formatos existentes.

5.1.2 Atributos del texto

Los atributos del texto influyen en el aspecto de algunas partes del texto, pero no tienen porque afectar al conjunto del documento.

- *Justificación*, indica la distribución horizontal del texto entre los márgenes, puede ser a la derecha, izquierda, centrada o completa (como la del texto que compone este documento). Ver figura 5.1.
- *Márgenes izquierdo y derecho.*
- *Interlineado*, tamaño de la línea: puntos de la letra + puntos de ajuste. Los últimos garantizan que no se solapen las letras entre dos líneas.

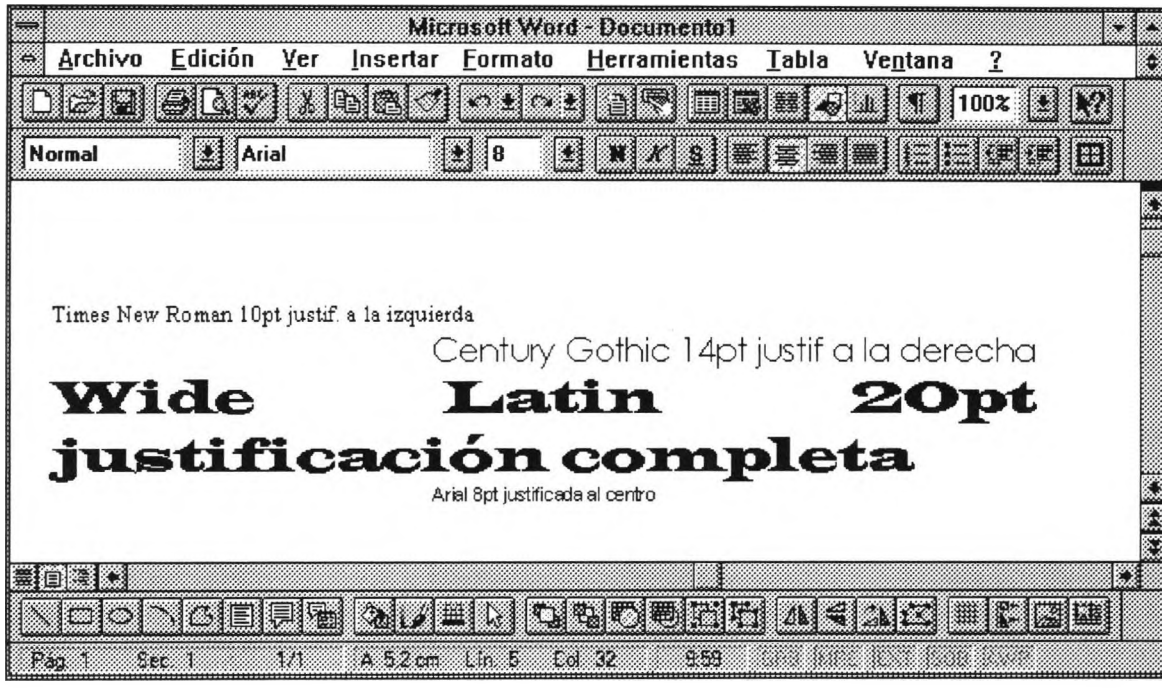


Figura 5.1: Justificación y tipos de letras

Figura 5.2: Tamaño en puntos de un tipo Helvetica

- *Espacio interlineal*, número de líneas que separan dos líneas con texto.
- *Tipo de letra*. También denominado *fuente*. El tipo de letra determina: el *conjunto de caracteres* que se pueden utilizar, el ancho y alto de cada carácter, el espacio entre cada carácter y, por supuesto, el diseño de los distintos caracteres, además de otras cuestiones.

  Un tipo de letra está diseñado en función de la *familia* a la que pertenezca, poseerá un *espaciado proporcional o fijo*, tendrá un tamaño de un número determinado de puntos, y opcionalmente su *aspecto* o *estilo* será alguno de los que se señala más adelante.

  - *Familias* existen muchas, y cada una se utiliza para un propósito determinado: en los periódicos se suele usar la Times, para los titulares la Helvética, los listados se realizan en Courier, ...
  - El *espaciado* fijo indica que todos los caracteres ocuparán un mismo espacio a lo ancho. Mientras que en el proporcional , éste será proporcional a la anchura "real" del carácter. Ejemplos: la familia Courier tiene un `espaciado fijo`, mientras que la Times es proporcional. En la figura 5.1 todas las fuentes tiene espaciado proporcional.
  - El *tamaño* del tipo viene determinado por el número de puntos que hay desde la parte superior del carácter mayúscula hasta el extremo inferior de los caracteres

que rebasan inferiormente la línea de texto, como son la 'g','j', ..., ver figura 5.2 . Teniendo en cuenta que un punto es 1/72 de pulgada.

Los tipos de letra suelen estar asociados a la impresora que utiliza el procesador, por esto el fabricante de ésta proporciona una serie de *controladores* para los programas más comunes que hay que instalar antes de poder imprimir. Estos controladores especifican a los programas de proceso de texto como utilizar los tipos de letra, tamaños, ..., correctamente. Muchas impresoras solo son capaces de usar tipos de tamaño fijo por sus características técnicas. Los llamados tipos *escalables* basados en una definición de tipo independiente del tamaño en puntos, ofrecen más posibilidades por su flexibilidad. Las impresoras láser son las que más tipos ofrecen, sobre todo las PostScript, el resto depende de las que estén definidas por hardware.

- *Tamaño*, algunos procesadores permiten especificar el tamaño de letra sin usar su expresión numérica, mediante unos tamaños predefinidos. Por ejemplo: Grande, Normal, Pequeña, Menuda, ...
- *Aspecto*, gracias a éste la diversidad de letras a utilizar aumenta: **negrilla**, subrayada, *cursiva*, ...

5.1.3 Atributos globales

Estos se establecen para afectar a todos los componentes del documento, o de una página:

- *Centrado de página*. Este en WP sólo afecta a una página a la vez.
- *Márgenes superiores e inferiores de una página*.
- *Numeración*, para indicar la posición de los números de página dentro de las distintas del documento.
- *Tipo de letra inicial*. Será la que se use en todo el documento, salvo que se indique otra cosa.
- *Tipo de papel*, para indicar características como: tamaño, orientación (apaisada, horizontal), ...
- *Resumen o sumario*, para mantener información sobre el documento como es: la fecha de creación, autor, tema, resumen, palabras claves, ...

5.1.4 Operaciones básicas de edición

5.1.4.1 Busqueda y sustitución

Son dos operaciones básicas con las que hay que familiarizarse cuando se trabaja sobre un texto de mucha extensión. La primera permite ir localizando sucesivamente la ocurrencia de una cadena de caracteres en el texto desplazandose sobre éste, o de otra manera, marcando las posiciones donde se encuentra la cadena de caracteres. La *sustitución* añade a la *busqueda* la capacidad de reemplazar las ocurrencias de una cadena de caracteres por otra.

En la mayoría de procesadores de texto se pueden especificar opciones de busqueda como: la distinción entre mayúsculas y minúsculas, la distinción del aspecto del texto (p.e. buscar las ocurrencias de la palabra *teorema* escritas en letra *cursiva*), ..., etc.

5.1.4.2 Bloques: Cortar, Copiar y Pegar

Los *bloques* son grupos de elementos seleccionados de un documento, por lo general adyacentes en cuanto a su posición. La selección del texto se puede realizar con comandos específicos del procesador o mediante algún dispositivo apuntador (como el ratón). Su uso está orientado a la aplicación de alguna operación sobre el texto seleccionado. Podemos destacar algunas de estas operaciones:

- Asignar un determinado aspecto a los componentes del bloque seleccionado.
- Convertir mayúsculas en minúsculas o viceversa.
- Alinear el bloque a derecha o izquierda en la página.
- Imprimir el contenido del bloque.
- Búsqueda/sustitución restringida al bloque.
- Eliminar la totalidad del contenido del bloque.
- ...

Quizás las operaciones más utilizadas con bloques de documento son las de *cortar*, *copiar* y *pegar*, que se utilizan combinadas dos a dos. Su funcionamiento es el siguiente:

Cortar: extrae un bloque de cierta posición del documento, eliminándolo de esta, y quedando "recordado" por el procesador de texto para utilizarlo en la siguiente operación de *pegar*.

Copiar: extrae un bloque de cierta posición del documento, sin eliminarlo de está, y quendando "recordado" por el procesador de textos para utilizarlo posteriormente en la siguiente operación de *pegar*.

Pegar: ésta culmina alguna de las dos anteriores y, previa selección de alguna posición en el documento, introduce en ésta el último bloque copiado o cortado.

El efecto de *cortar* y *pegar* equivale a mover el trozo de texto sobre el que se aplica desde una posici'on dada a otra diferente elegida para pegarlo. Cuando lo que se hace es *copiar* y *pegar* se duplica el texto objeto de estas dos.

5.1.5 Cabeceras y pies de páginas

Las cabeceras y pies de páginas están formadas por texto que se dispone en la parte superior e inferior, respectivamente, de cada página del documento. Suelen utilizarse, más las primeras, para indicar el título de un capítulo, nombre del autor, ..., etc. El texto que se introduce en cabeceras y pies puede tener su propio aspecto independiente del resto. Cuando se utiliza alguno de estos recursos el procesador de textos se encarga de ajustar el texto al espacio disponible de página.

5.1.6 Referencias cruzadas

Mediante este concepto se pueden definir relaciones entre distintos puntos de un texto. Pueden ser números de páginas, de parrafo, de nota a pie de página, de ilustración, ..., etc. Las referencias cruzadas mantienen la coherencia entre distintos elementos del documento a los que se alude, frente a modificaciones en el mismo.

5.1.6.1 Notas al pie

Son textos que suelen aparecer en distintas páginas situados en la parte inferior, a los que se hace referencia en un punto dado del resto del texto del documento por su contenido aclaratorio. Su presencia es frecuente en textos de jurisprudencia. Un ejemplo de nota al pie se puede ver en la nota [1].

Con un procesador de texto tenemos a disposición distintas opciones asociadas para especificar:

- Espaciado entre cada nota, y dentro de las mismas.
- Estilos de numeración, métodos (letras o caracteres), ..., etc.
- Separación entre las líneas de texto normal y las notas.
- Cantidad de nota a mantener unida cuando ésta ocupa más de una página, es decir junto con el resto del documento.
- ...

El procesador de texto se encarga de mantener las alusiones a cada nota (usando la numeración), para que esta se mantenga coherente si se introducen nuevas notas. Así como de ajustar el contenido de texto normal de cada página según las especificaciones de las notas.

5.1.7 Modelos

Los *modelos* (Microsoft WORD) proporcionan definiciones de formatos para distintos tipos de documentos o elementos del texto. Estas definiciones (tamaño de papel, tipo de letra, separación entre párrafos, márgenes, ...) se guardan en archivos independientes del texto, de manera que pueden ser utilizados en repetidas ocasiones. En esto radica su potencia y en el hecho de que cualquier cambio que se decida hacer en la definición del modelo afecta a los documentos asociados.

Parecida a la anterior existe otro concepto pensado para un fin similar, se llama *estilo* y como su nombre indica sirve para especificar un estilo en el documento, de forma similar al anterior. WP 5.1 es un procesador que los utiliza como herramienta de uniformización en la construcción de documentos, permiten especificar menos detalles, y suelen guardarse junto con el documento en el mismo fichero, aunque se puede especificar lo contrario para que se pueda uilizar en otros.

5.1.8 Procesos

Bajo este nombre se agrupan funciones de alto nivel que pueden ser aplicadas a un texto determinado:

- *Revisión ortográfica*, con éste se localizan y corrigen errores de los siguientes tipos: uso incorrecto de mayúsculas, palabras duplicadas y palabras incorrectamente escritas. Casi todos los procesadores disponen de un diccionario que les permite hacer esas correcciones y opcionalmente se puede especificar y construir el llamado diccionario suplementario en

[1] Nota al pie del apartado *Notas al pie*

el que se especifican palabras de un cierto argot. Asociado a la ortografía se puede usar también un diccionario de *sinónimos* que permite depurar el texto cambiando palabras con significado similar.

- *Fusión* o *correo personalizado*, consiste en combinar archivos diferentes para producir uno nuevo. P.e. para crear cartas modelo personalizadas se utilizaría un archivo que tuviese la carta modelo, y otro con una serie de nombres y direcciones de personas a las que se enviarían.
- *Clasificación*, para ordenar con algún criterio líneas de texto o párrafos.
- *Exportación/Importación*, posibilita la lectura/escritura de un fichero en un formato determinado. Estas operaciones tienen gran importancia tanto a la hora de poder leer ficheros de documentos creados con otros procesadores de texto, así como de poder integrar contenidos de ficheros de otro tipo de programas como hojas de cálculo o programas de dibujo.
- ...

5.1.9 Macros

Al igual que en otro tipo de programas, como las hojas de cálculo, en los que la introducción de secuencias de teclas es importante, los procesadores de texto ofrecen la posibilidad de mantener *macros* que realicen mediante la introducción de una secuencia corta de teclas, lo que sería una equivalente más larga.

5.2 Hojas de Cálculo

5.2.1 Descripción de la hoja de cálculo

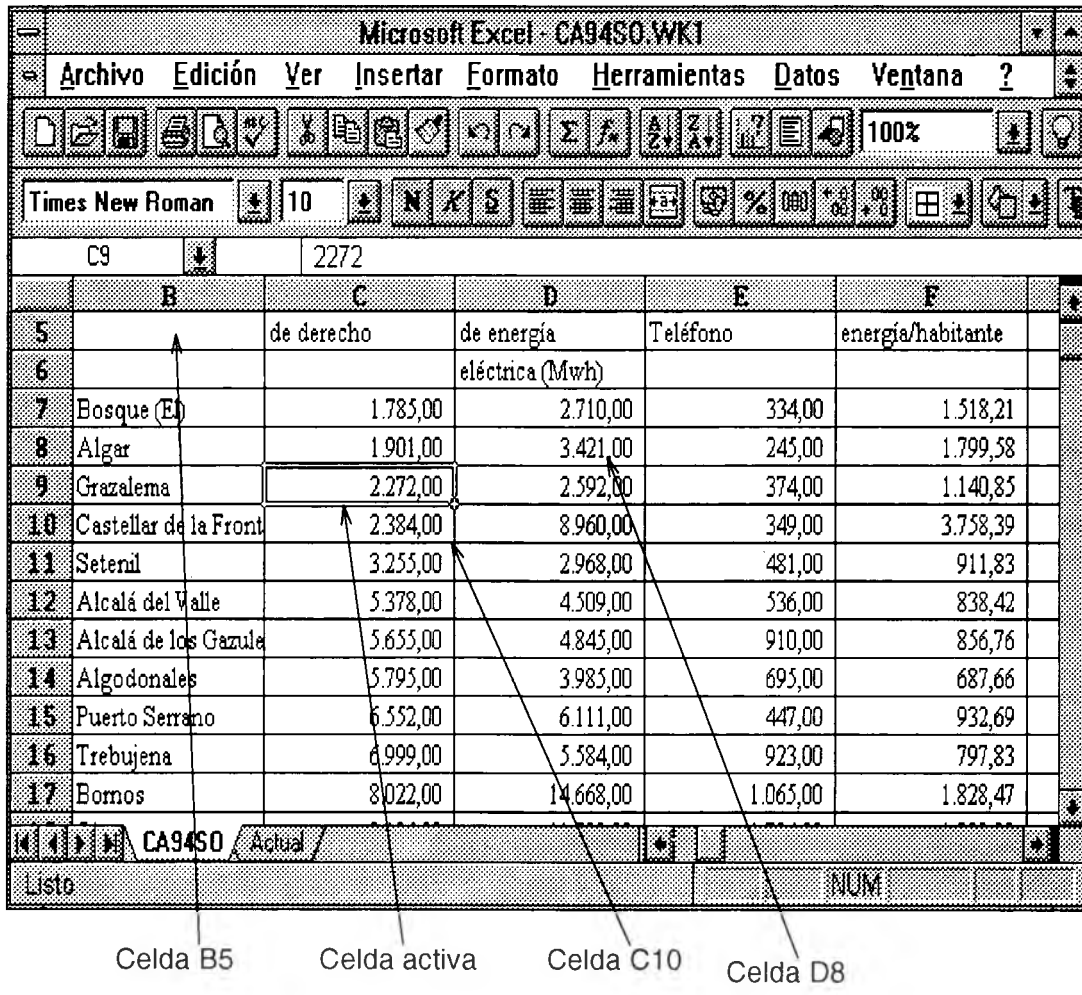
Se puede definir la *hoja de cálculo*, o estadillo como un agregado de objetos concurrentemente activos organizados por lo común en una matriz rectangular de casillas (parecida a las planas de cuadrículas que se usan en contabilidad), o desde el punto de vista de la Informática como un programa de tratamiento y manipulación de objetos activos, organizados en una matriz cuadrangular de casillas o celdas.

Fueron inventadas por Daniel Bricklin y Robert Frankston por la irritación que les producía utilizar los clásicos estadillos de papel pautado, siendo estudiantes de empresariales. Sus aplicaciones son infinitas, las principales están relacionadas con cálculos en el ámbito de las finanzas (beneficios, perdidas, gastos, ingresos por ventas, ...), también se puede organizar una pequeña base de datos de biblioteca, alumnos de un curso, ..., o de más trascendencia para realizar *simulaciones*.

Hay que hacer notar que la expresión *hoja de cálculo* se utiliza para denominar tanto a la propia hoja (estructura rectangular) donde se almacenan los datos, como al programa que ofrece la hoja visualmente y permite utilizarla.

5.2.2 Celdas

Una hoja de cálculo está formada por filas y columnas, y en las intersecciones de estas se conforman las *celdas*. La celda es la unidad mínima de dato de una hoja de cálculo. Los datos



	B	C	D	E	F
5		de derecho	de energía	Teléfono	energía/habitante
6			eléctrica (Mwh)		
7	Bosque (El)	1.785,00	2.710,00	334,00	1.518,21
8	Algar	1.901,00	3.421,00	245,00	1.799,58
9	Grazalema	2.272,00	2.592,00	374,00	1.140,85
10	Castellar de la Front	2.384,00	8.960,00	349,00	3.758,39
11	Setenil	3.255,00	2.968,00	481,00	911,83
12	Alcalá del Valle	5.378,00	4.509,00	536,00	838,42
13	Alcalá de los Gazule	5.655,00	4.845,00	910,00	856,76
14	Algodonales	5.795,00	3.985,00	695,00	687,66
15	Puerto Serrano	6.552,00	6.111,00	447,00	932,69
16	Trebujena	6.999,00	5.584,00	923,00	797,83
17	Bornos	8.022,00	14.668,00	1.065,00	1.828,47

CA94SO Actual
Listo NUM

Figura 5.3: Hoja de cálculo

de una hoja de cálculo se introducen en las distintas celdas que contiene (ver figura 5.3.

Asociado a este concepto se habla de *celda activa* cuando nos referimos a la única celda que admite alguna entrada en un momento dado. Cada celda tiene asociada una *dirección* o *referencia* dentro de la hoja que se forma con el nombre de la columna y la fila donde reside la celda, en la figura 5.3 se pueden observar algunos ejemplos. Dentro de una celda es posible introducir *valores* como en las celdas C10, D8, ..., etc., o *literales* (números o cadenas de caracteres) como en las celdas B8, C4, ..., etc., o *fórmulas*.

5.2.3 Tipos de datos

Se denomina *dato* a cualquier objeto manipulable por una computadora. Con un programa de hoja de cálculo se realiza un tratamiento de datos organizados a modo de matriz de celdas, cada una de estas almacenando un dato. El concepto asociado *tipo de dato* permite distinguir entre las clases de objetos (clases de datos) que puede contener una celda, y en particular saber si es posible la aplicación de una determinada fórmula a ciertos datos de un tipo dado introducidos en la hoja, así el programa puede diagnosticar el buen uso o no de la fórmula.

Tabla 5.1: Prioridades de los operadores de mayor a menor

Operador	Descripción
()	Paréntesis
^	Exponenciación
+ -	Signos
/ *	División y Multiplicación
+ -	Suma y Resta
= <> < > <= >=	Operadores de comparación
#NO#	Operador *no* lógico
#Y# #O# &	Operadores *y*, *o* lógicos y concatenación.

Los tipos de datos que maneja cualquier programa de la clase que nos ocupa son:

Número: es una cadena de caracteres numéricos que forman la parte entera y que opcionalmente, con el carácter (,) y más caracteres numéricos, tendrá también parte fraccionaria.

Fórmula: se compone de un conjunto de instrucciones en forma de signos o expresiones especiales, que indican la realización de un cálculo particular. Pueden ser:

- *Arimé ticas*, operaciones (operadores): +, -, *, / y ^. Admiten constantes numéricas, y referencias a celdas como operandos, produciendo resultados numéricos. El resultado de una fórmula depende del valor de los operandos, así al variar el valor de una celda operando, la fórmula se *recalculará* inmediatamente. Con la fórmula +(C8+C9)/(D10*0.5) se obtendría 2.84 según la hoja de la figura 5.3.
- *Lógicas*, operaciones (operadores) de comparación fundamentalmente: =, <>, >, >=, <, <=. No producen resultados numéricos, los resultados posibles son 1 (verdad) ó 0 (falso). Los operandos pueden ser constantes o referencias a celdas. También se utilizan los *operadores compuestos* : #Y#, #O#, #NO#. Cuando se combinan operaciones aritméticas con lógicas en una fórmula, se calculan primero las aritméticas. De la evaluación de la fórmula +D8>D9#Y#C8<C9 resultaría el valor 0.
- *con cadenas* (aparecen en la versión 2 de Lotus), permiten unir cadenas para crear elementos de datos para la hoja. Utilizan un operador &. El resultado de evaluar la fórmula "Hola, "&"mundo"se obtendría "Hola, mundo".

Los operadores de las fórmulas tienen una *prioridad en el orden de evaluación* de la misma. La prioridad de mayor a menor se muestra en la tabla 5.1.

Teniendo en cuenta las prioridades de la tabla 5.1 si especificamos la fórmula del ejemplo de fórmulas ariméticas sin paréntesis: +C8+C9/D10*0.5, el resultado difiere sensiblemente: 1785.37.

Asociado a las fórmulas hay otro concepto importante en la hojas de cálculo y es el *recálculo* . Cuando se modifican celdas referenciadas dentro de una fórmula en otra celda, la primera es recalculada automáticamente por el programa de la hoja, aunque se puede especificar que no sea así. Existen fundamentalmente tres formas de recálculo:

Tabla 5.2: Formatos de rótulos

Formato	*Contenido*	*Valor*	*Presentación*
`Izquierda`	'Enero	Enero	Enero
`Derecha`	''Enero	Enero	Enero
`Centro`	^Enero	Enero	Enero

- *Natural*, el que se usa por defecto, y que calcula primero las fórmulas de las que dependen de otras, es decir priemro aquellas cuyo resultado es utilizado por otras.
- Por *columnas*, evalúa las fórmulas columna a columna y dentro de estas de arriba a abajo.
- Por *filas*, igual que la anterior pero por filas.

En un programa de hoja de cálculo también es posible especificar que el recálculo de todas las fórmulas de una hoja sea manual, cosa que puede resultar útil cuando se trabaja con hojas complejas.

Rótulo: cadena de caracteres alfanuméricos precedida por el *prefijo de rótulo*, con el que se puede expresar la disposición del texto dentro de la celda.

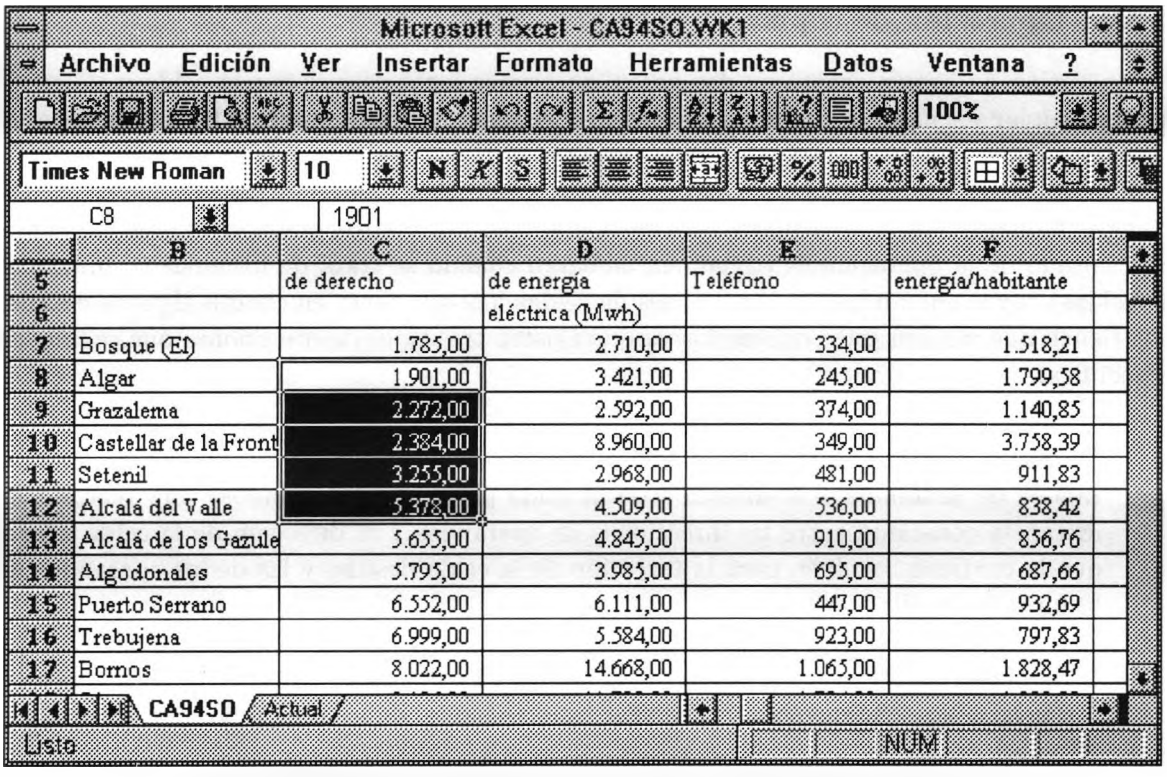
5.2.4 Formatos de presentación

El dato que almacena una celda de la hoja de cálculo se puede observar a tres niveles, que son: el *contenido* , el *valor* , y la *presentación* de la celda. En una primera aproximación a estos conceptos podemos decir que el primer nivel hace referencia al dato tal y como se introduce en la celda siguiendo las pautas de forma de la hoja en particular, a continuación el valor de la celda es lo que se obtiene al evaluar el contenido de la misma y finalmente la presentación por la que se distinguen varios formatos asociados al tipo de dato que se guarda en la celda. Para entender que significa esta segmentación vamos a ver que son esos niveles para los tres tipos de datos que conocemos.

- Cuando el contenido de una celda es un número, el resultado de la evaluación de éste es un número por lo que se dice que el contenido y el valor coinciden, mientras que la presentación puede variar según el formato. Asíen la figura 5.3 se puede ver como el contenido/valor de la celda activa es 2272, y su presentación es 2272.00.

- Para un rótulo , el contenido de la celda difiere de su valor y de su presentación, ya que suele ser una cadena alfanumérica precedida de un prefijo especial que la caracteriza (en algunas hojas de cálculo basta con que la cadena comience con una letra). Ese mismo prefijo define el formato de presentación como se ve en la tabla 5.2.

- Para las fórmulas, el contenido, valor y presentación también difieren mutuamente: el contenido de una celda donde se ha introducido una fórmula es la expresión de ésta, el valor es el resultado de la evaluación y la presentación es el formato aplicado a este resultado. Si el resultado de la evaluación es un número la presentación que se aplicará será la apropiada para los números, y de la misma forma para los rótulos. Teniendo en



	B	C	D	E	F
5		de derecho	de energía	Teléfono	energía/habitante
6			eléctrica (Mwh)		
7	Bosque (El)	1.785,00	2.710,00	334,00	1.518,21
8	Algar	1.901,00	3.421,00	245,00	1.799,58
9	Grazalema	2.272,00	2.592,00	374,00	1.140,85
10	Castellar de la Front	2.384,00	8.960,00	349,00	3.758,39
11	Setenil	3.255,00	2.968,00	481,00	911,83
12	Alcalá del Valle	5.378,00	4.509,00	536,00	838,42
13	Alcalá de los Gazule	5.655,00	4.845,00	910,00	856,76
14	Algodonales	5.795,00	3.985,00	695,00	687,66
15	Puerto Serrano	6.552,00	6.111,00	447,00	932,69
16	Trebujena	6.999,00	5.584,00	923,00	797,83
17	Bornos	8.022,00	14.668,00	1.065,00	1.828,47

Figura 5.4: Rango C8..C13

cuenta la figura 5.3 si en la celda $B6$ se introduce la fórmula +C5 & C6 el contenido de la celda será la fórmula anterior, su valor la cadena: `Poblaciónde derecho` y la presentación dependerá del tipo de justificaciôm que se asigne. En las celdas con fómulas es posible usar un formato especial que no presenta el resultado de su evaluación sino la expresión de la fórmula.

Además de los tipos de formatos que han sido nombrados hasta ahora muchos programas de hoja de cálculo ofrecen posibilidades de presentación similares a las del los procesadores de texto, por ejemplo la elección del tipo y aspecto de la letra, tamaño, alieneación vertical en la celda, ..., etc.

5.2.5 Rangos de celdas

Un *rango de celdas* es un conjunto de celdas adyacentes, que se define con la celda del límite superior izquierdo del conjunto, y la celda del límite inferior derecho, separados por dos puntos (.. ó :). Puede estar constituido por una celda, por una fila o columna o por una matriz rectangular (figura 5.4). Los rangos se pueden identificar mediante *nombres de rango* , éstos pueden servir para agrupar datos de un mismo tipo, y de cualquier modo ayuda a la lectura de la hoja cuando se utilizan funciones (que veremos más adelante) cuyos operandos se indican con rangos de celdas.

5.2.6 Operación de copiar

La operación de *copiar* consta de dos pasos, en primer lugar seleccionar la celda o rango de celdas a copiar y señalar la posición (celda) donde se va a duplicar el trozo de hoja seleccionado. Gracias a esta operación la construcción de grandes hojas de cálculo es más ágil y veloz cuando la hoja se diseña adecuadamente. Cuando se copian celdas de la hoja de cálculo que contienen rótulos, números, fórnulas o funciones con operandos/parámetros que son valores explícitos el resultado de la operación es trivial, sin embargo cuando se trata de fórmulas (o funciones en celdas) cuyos operandos son direcciones de celdas hay que tener en cuenta algunos detalles que tienen que ver con estos operandos ya que existen tres tipos de direcciones que se pueden especí ficar:

- Direcciones *relativas*, caso en el que al copiar se adaptarán las direcciones de los operandos de la fórmula a la dirección de la celda destino manteniendose una relación de distancia constante entre las direcciones de operandos y la dirección de la celda origen que la contiene, también para la dirección de la celda destino y las de los operandos de la fórmula en ella copiada.

- Direcciones *absolutas*, al realizarse la copia las direcciones de los operandos permanecerán invariables. Es necesario especificar las direcciones de los operandos con la notación ColFil.

- Direcciones *mixtas*, son un caso intermedio de las anteriores y especifican que fila o columna, $ColFil o Col$Fil, será relativa a la fila o columna de la nueva dirección donde se copie la fórmula. En este caso se mantiene constante la relación de distancia para las filas o para las columnas.

Ejemplos de cada uno de estos tipos de direcciones:

- A5 = +A4 * 4 se copia a la celda D5, con lo cual el contenido de esta será: +D2 * 4

- A5 = +$A$4 * B3 se copia a la celda D5, con lo cual el contenido de esta será: +$A$4 * E3

- A5 = +$A2 + A3 se copia a la celda D7, con lo cual el contenido de esta será: +$A4 + D5; si en A5 hubiesemos tenido: +A$2 + A3, en D7 se obtendría: +D$2 + D5

Mediante la copia (ver figura 5.5) es posible:

- copiar *el* contenido de una celda en *una* nueva dirección,

- copiar *el* contenido de una celda en *varias* direcciones, seleccionando un rango de celdas para que se copie la celda origen, y

- copiar *los* contenidos de distintas celdas en *varias* direcciones, para lo cual es necesario seleccionar un rango de celdas origen y una celda destino a partir de la que se reproducirán las originales.

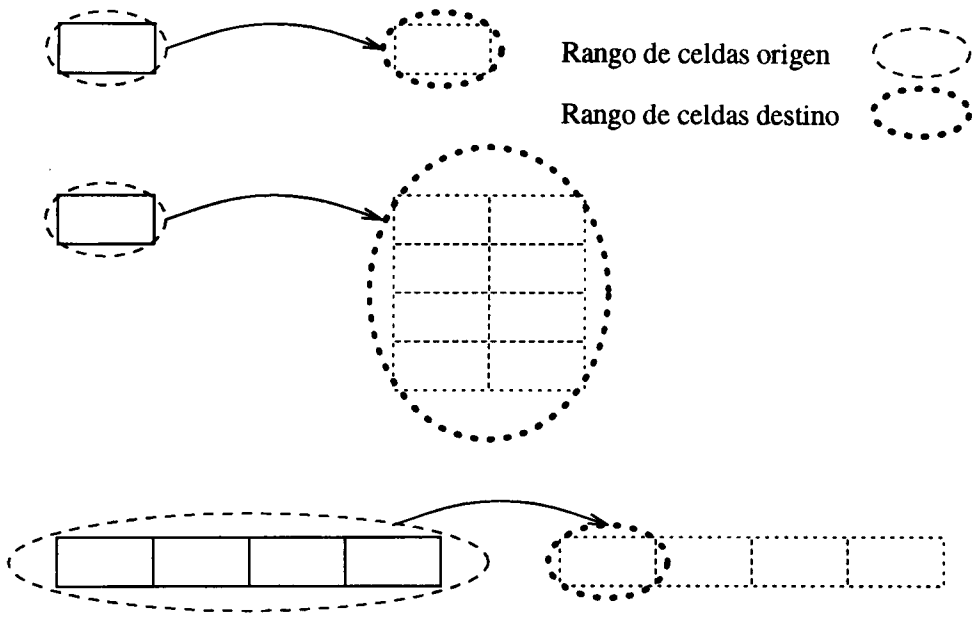


Figura 5.5: Operaciones posibles de copiar

5.2.7 Funciones

Las *funciones* son fórmulas definidas internamente en la hoja de cálculo, y por ello su utilización es similar a la de estas. El formato sintáctico genérico de una función es

NombreFunción(Argumento1, Argumento2, ..., ArgumetoN)

Para algunas hojas de cálculo el nombre de las funciones está precedido del símbolo @ y a continuación una secuencia de caracteres, por ejemplo : @SUMA(9,4,8,7).

Los argumentos, que suelen ir separados por ';' o ',', son los datos que utiliza la función para realizar el cálculo. Funciones diferentes requieren tipos de argumentos diferentes. Los tres tipos básicos de función son:

- las que no necesitan argumentos,
- las que esperan una lista de argumentos en cualquier orden,
- y las que esperan un número fijo de argumentos, en un orden determinado.

A su vez un argumento puede ser

- un valor,
- una dirección de celda, o
- un rango de celdas.

Los tipos de funciones que se pueden utilizar en un hoja son:

1. *Estadísticas*. Suelen esperar una lista de valores o celdas (rango) como argumento. Ejemplos: @MEDIA, @CUENTA, @MAX, @MIN, ...

2. *De cadena.* Permiten la manipulación de rótulos . Ejemplos: @IGUAL, @LONGITUD, @MINUSC, @VALOR, @CADENA, ...(Versión 2 de Lotus).

3. *Matemáticas.* Ejemplos: @ABS, @COS, @SEN, @ALEAT, @REDOND, @EXP, ...

4. *Financieras.* Sirven para utilizarlas en cálculos sobre inversiones de capital y en general para cálculos relacionados con la evolución del valor del dinero. Algunas de ellas son:

   - @AMORT(capital, interés, plazo del préstamo), calcula el importe de los pagos de un préstamo,
   - @PERIODO(pagos, interés, valor futuro), calcula el número de períodos necesarios para alcanzar un valor futuro,
   - @VALFUT(pago, interés, período), calcula el valor futuro de una inversión,
   - @TASA(valor futuro,valor actual,número períodos), calcula el tipo de interés necesario para alcanzar un valor futuro,
   - ...

5. *Lógicas.* Permiten realizar cálculos basados en valores de verdad. Devuelven valores lógicos como resultado de su evaluación: verdad, o falso. Ejemplos: @FALSO, @SI, @ESCADENA, @ESNUM, @TRUE, ...

5.2.8 Macros de teclado

En un programa de hoja de cálculo el trabajo se realiza utilizando menús de operaciones, y por lo general el acceso a menús se realiza a través de pulsaciones de teclado. Cuando se realizan muchas operaciones repetitivas sobre una hoja de cálculo puede ser interesante automatizarlas, para esto se utilizan las *macros de teclado.*

Las macros de teclado son una columna de rótulos que tienen asignado un nombre especial. En estos rótulos se introducen las pulsaciones a efectuar para realizar las operaciones que se implementen. En una macro se pueden grabar selecciones de un menú de operaciones, o pulsaciones de teclas que pueden ser a su vez de: movimiento, ({AR}, {AB}, {D}, ...), edición, ({DEL}, {INS}, {ESC}, {RT}), función ({AYUDA}, {EDICION}, ...

Existe un reducido lenguaje de órdenes que permiten mejorar la confección de las macros, {BIFURCAR celda,nombre de rango}, {SI condición}{BIFURCAR}, {?}, ...

Ejemplo:

```
@FECHA(
{ ? }
;
{ ? }
;
{ ? }
)~
/rfd1~
/hcf10~
```

Con esta macro se puede introducir la fecha comodamente. Los comandos { ? } permiten introducir los parámetros que necesita la función que ejecuta la macro. Las dos últimas líneas fijan el formato a fecha y cambian el ancho de la columna a 10. Este otro ejemplo extiende la macro anterior para introducir 10 fechas a lo largo de una columna:

```
            { DEJAR A1; 0 }
Principio   {SI A1 = 10} {BIFURCAR Final}
            { DEJAR A1; A1+1}
            @FECHA(
            { ? }
            ;
            { ? }
            ;
            { ? }
            )~
            {ABAJO}
            {BIFURCAR Principio}
Final       {SALIR}
```

5.2.9 Diseño de una hoja de cálculo

Para terminar de hablar de las hojas de cálculo vamos a ver algunos aspectos básicos que hay que tener en cuenta a la hora de plantearse utilizar un programa de hoja de cálculo para resolver un problema. Para llevar ésto a cabo nos vamos a servir de tres ejemplos sencillos de hojas que se pueden construir dados los tres problemas respectivos.

De cualquier manera, a título informativo es bueno atenerse a las siguientes indicaciones que pueden resultar de mucha utilidad.

- Realizar sobre papel antes de comenzar un esquema de la hoja:

  1. Identificar los datos de entrada y los resultados que se desean. Estos deben venir especificados en el planteamiento del problema y es conveniente reconocerlos para pensar en la posición que ocuparán dentro de la hoja.
  2. Establecer las relaciones entre los datos de entrada y salida y recolectar los conocimientos específicos necesarios para resolver el problema. Estos conocimientos suelen ser la llave para conocer las relaciones entre los datos.

- Para la presentación de la hoja es importante utilizar modelos conocidos existentes. Y siempre que sea posible:

  – Situar los datos de entrada o en filas solamente o columnas.

  – Separar los datos de entrada de los resultados.

  – Utilizar rótulos explicativos.

5.2.9.1 Ejemplo 1

El planteamiento para este ejemplo dice que sabiendo los datos de coste de una serie de materias primas (ver tabla 5.3) necesarias para construir sillas y mesas de comedor, se desea para mantener un control de la producción construir una hoja de cálculo mediante la cual

- se pueda saber cuál es el precio de una silla o una mesa en función de los datos sobre coste materias primas (tabla 5.3) y cantidad de estas materias necesarias para su construcción (tabla 5.4),
- por otro lado es interesante conocer el precio de los comedores en función del número de plazas (sillas), además del resto de parámetros que afectan a las materias primas,
- y por último pensando en un pedido de comedores de un determinado número de plazas se desea saber que cantidad de materiales es necesaria para su construcción.

Madera	5000	pts/m^3
Barníz	1000	pts/lt
Tornillos	5	$pts/unidad$

Tabla 5.3: Coste de materiales

Los *datos de entrada* se pueden observar en las tablas 5.3 y 5.4. Los *resultados* que se deben obtener según el enunciado son los *precios de las sillas y de las mesas*, el *coste de los comedores*, y la *cantidad de material* necesaria para construirlos.

El diseño elegido para este problema se puede apreciar en la figura 5.6. En ella se observa la distribución separada de los datos de entrada que influyen sobre los resultados y es fácil identificar las respuestas a los requerimientos del problema.

5.2.9.2 Ejemplo 2

En este ejemplo se trata de diseñar una hoja de cálculo mediante la cual se pueda analizar el importe de la cuota de un prestamo a amortizar mediante el método francés, en función de un determinado interés, sabiendo el importe del prestamo, el interés mínimo aplicable y el tiempo que se va a invertir en pagarlo. La idea es poder ver que cuota hay que utilizar para cada tipo de interés dentro de un rango que empieza por el interés mínimo anterior.

En este caso sin necesidad de conocer los valores concretos de las variables que aparecen en problema podemos identificar como *datos de entrada* los siguientes: importe del prestamo (C), interés (i) mínimo y tiempo de amortización (o número de periodos o pagos, n). El *resultado* será una tabla con una entrada para los intereses. Por otro lado es necesario conocer

Material	Silla	Mesa	*Unidad*
Madera	0.75	1.4	m^3
Barníz	0.2	2.5	*litros*
Tornillos	50	300	*unidades*

Tabla 5.4: Necesidades de materiales

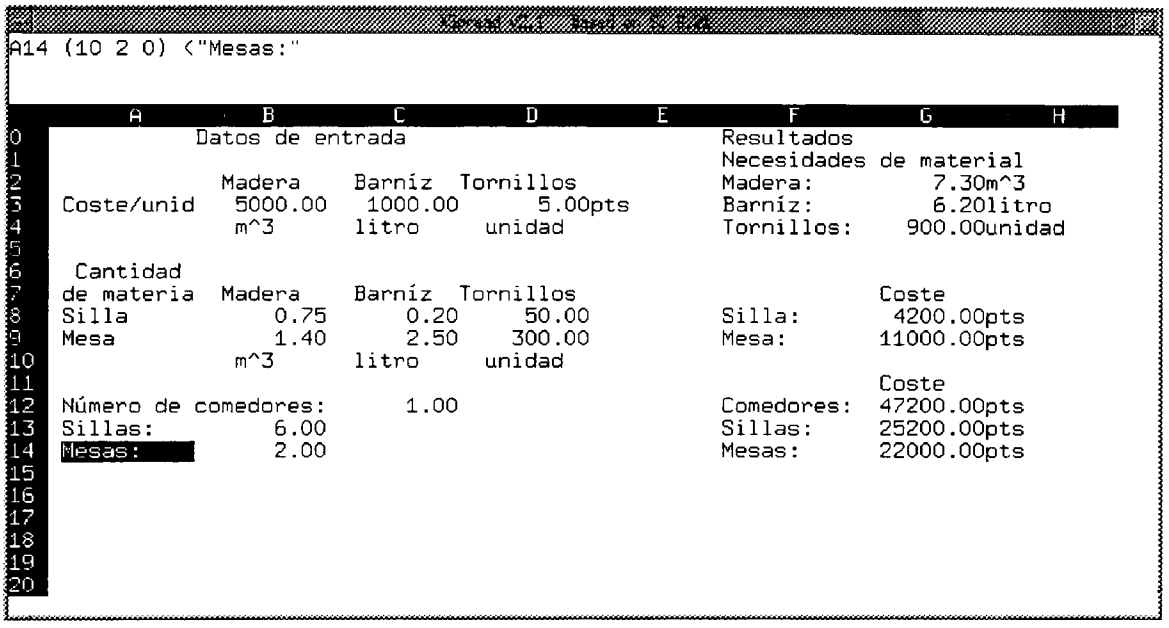


Figura 5.6: Resolución del ejemplo 1

algo sobre como calcular la cuota de pago para amortizar un prestamo con el método francés. La expresión que nos interesa es: $\frac{C \cdot i}{1-(1+i)^{-n}}$

Un posible diseño se puede apreciar en la figura 5.7. Sobre la resolución hay que señalar en primer lugar que se han introducido más variables, una es la llamada *Salto* mediante la cual se construye la entrada de *Interés* de la tabla incrementando sucesivamente el *Tipo anual* (interés mínimo). Otra variable es *Pagos/año* (*pa*), ya que se ha supuesto que los periodos de pago se pueden realizar un número determinado de veces al año, es por esto que la variable n de la expresión para el cálculo de la cuota se sustituye por $pa \cdot a$, siendo a el número de años.

5.2.9.3 Ejemplo 3

El problema que se plantea a continuación consiste en realizar un sencillo modelo de planificación comercial para cuatro periodos de tiempo. Las relaciones entre las variables que interviene son las siguientes:

- Los *gastos de ventas* representan el 10% de las *ventas.*
- Los *gastos de publicidad* representan el 5% de las *ventas.*
- Los *gastos de intereses* se calculan como la suma del 10% del *promedio de las deudas a largo plazo* y el 12% del *promedio de lasduedas a corto plazo.*
- La *previsión de impagados* se calcula como el 1% de las *cuentas a cobrar.*
- Los *impuestos* suponen un 48% de los *beneficios.*

A partir de la construcción del modelo se pretende conocer como *resultado* el valor de las variables indicadas anteriormente para cada periodo, y conociendo el *gasto total* presentar

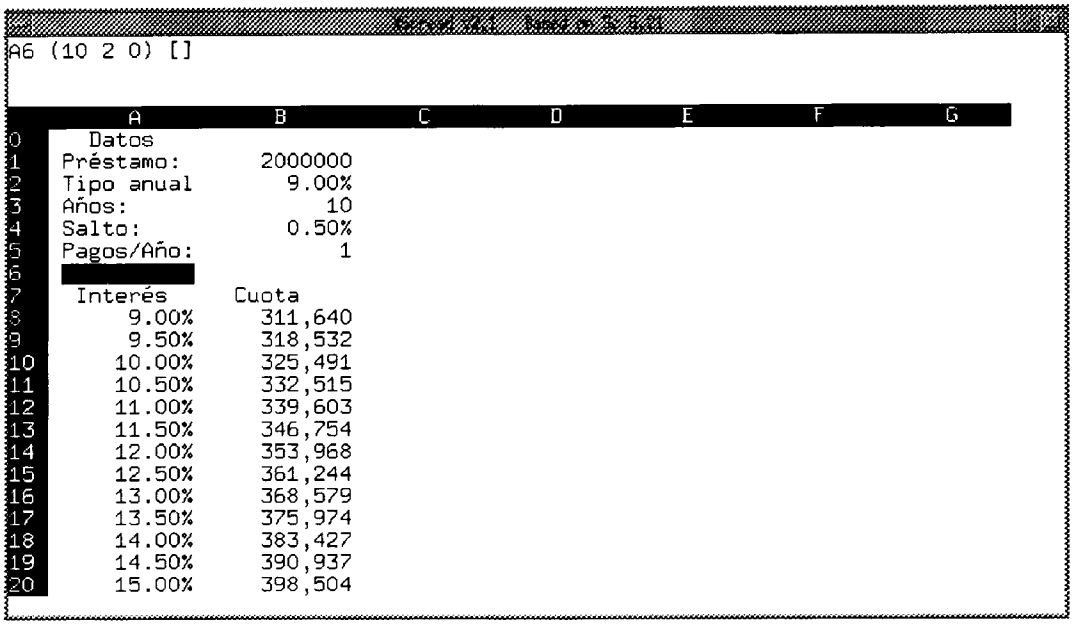


Figura 5.7: Resolución del ejemplo 2

Periodos	1992	1993	1994	1995
Ventas	110	120	130	150
Promedio deudas a largo plazo	100	100	100	100
Promedio deudas a corto plazo	10	15	10	20
Cuentas a cobrar	30	35	35	40
Gastos de administración	40	45	45	55

Tabla 5.5: Datos para el modelo de planificación

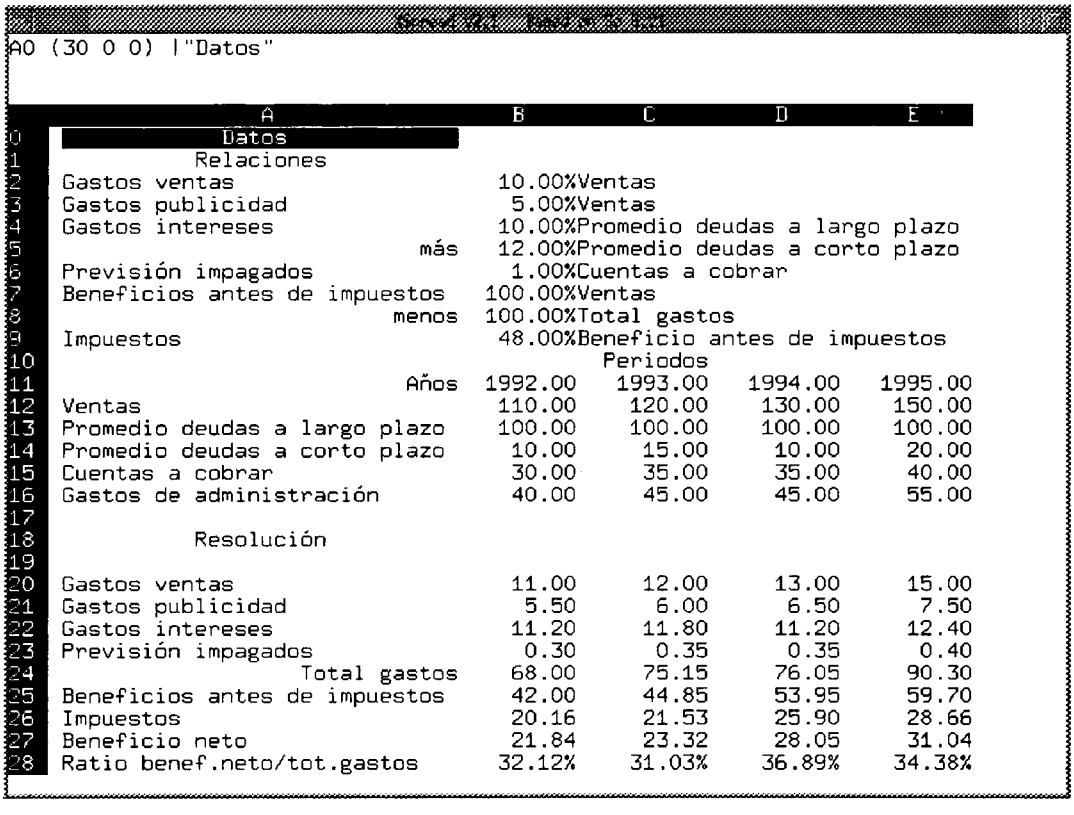
para cada periodo la relación entre el beneficio neto (después de impuestos) y el total de gastos.

Los *datos de entrada* de los periodos a analizar se pueden ver en la tabla 5.5.

La figura 5.8 muestra la solución adoptada para plasmar el modelo en la hoja de cálculo y como se puede observar el planteamiento del problema permite establecer una estructura clara para la disposición de los datos y los resultados.

5.3 Gráficos

Desde sus principios, los ordenadores han atraido a los usuarios en general por su alta capacidad de proceso y por las posibilidades que ofrece en cuanto al manejo de datos para poder representar eficientemente ideas. Al comienzo las posibilidades de realizar estas representaciones gráficamente eran pocas, debido a la velocidad limitada de los ordenadores para obtener buenos comportamientos, y por la tecnología existente en sistemas de vídeo y de impresión. El cambio progresivo de estas circunstacias hizo cambiar progresivamente el conjunto de aplicaciones de la informática, pasando la que nos ocupa a ser de las más sobresalientes.



A0 (30 0 0) |"Datos"

	A	B	C	D	E
0	Datos				
1	Relaciones				
2	Gastos ventas	10.00%Ventas			
3	Gastos publicidad	5.00%Ventas			
4	Gastos intereses	10.00%Promedio deudas a largo plazo			
5	más	12.00%Promedio deudas a corto plazo			
6	Previsión impagados	1.00%Cuentas a cobrar			
7	Beneficios antes de impuestos	100.00%Ventas			
8	menos	100.00%Total gastos			
9	Impuestos	48.00%Beneficio antes de impuestos			
10		Periodos			
11	Años	1992.00	1993.00	1994.00	1995.00
12	Ventas	110.00	120.00	130.00	150.00
13	Promedio deudas a largo plazo	100.00	100.00	100.00	100.00
14	Promedio deudas a corto plazo	10.00	15.00	10.00	20.00
15	Cuentas a cobrar	30.00	35.00	35.00	40.00
16	Gastos de administración	40.00	45.00	45.00	55.00
17					
18	Resolución				
19					
20	Gastos ventas	11.00	12.00	13.00	15.00
21	Gastos publicidad	5.50	6.00	6.50	7.50
22	Gastos intereses	11.20	11.80	11.20	12.40
23	Previsión impagados	0.30	0.35	0.35	0.40
24	Total gastos	68.00	75.15	76.05	90.30
25	Beneficios antes de impuestos	42.00	44.85	53.95	59.70
26	Impuestos	20.16	21.53	25.90	28.66
27	Beneficio neto	21.84	23.32	28.05	31.04
28	Ratio benef.neto/tot.gastos	32.12%	31.03%	36.89%	34.38%

Figura 5.8: Modelo de planificación

Por esto con el ordenador tenemos una herramienta nueva para realizar tareas que han requerido, hasta ahora, del uso del lápiz, escuadra, regla y demás herramientas de dibujo, y que son sustituidos por otros elementos que van desde el más simple ratón hasta la tableta digitalizadora o el escáner.

La capacidad gráfica de un ordenador se puede utilizar para estas cuatro tareas fundamentalmente:

- Representación de datos.
- Representación de ideas y dibujos.
- Diseño asistido por computadora.
- Tratamiento digital de imágenes.

Para cada una de estas existe el software apropiado, aunque algunos programas no se puedan encuadrar en un único tipo.

5.3.1 Software de representación de datos

Este tipo de programas está muy relacionado con aquellos en los que se utilizan muchos datos, y viene a aportarles una extensión gracias a la cual los datos pueden ser resumidos de alguna forma en una representación pictórica que ayuda a su comprensión. Este tipo de software nos lo encontramos en programas como las hojas de cálculo (figura 5.9), o en programas de cálculo científico, como ejemplos típicos, formando un sistema indivisible en el que la parte gráfica está perfectamente enlazada con el resto del programa, lo que facilita el paso a la presentación gráfica de cualquier conjunto de datos producidos como resultado de algún proceso.

Los tipos de representaciones gráficas (ver figura 5.10 que se pueden realizar son: diagrámas de barras, líneas, sectores, áreas, ..., en sus versiones 2D y 3D. A las que se pueden agregar rótulos informativos, leyendas sobre la distribución de los datos en el gráfico, ...

5.3.2 Software de representación de ideas y dibujos

Este tipo de programa engloba a toda una serie de herramientas orientadas a sustituir en la medida de lo posible al estuche de colores y al papel de dibujo. De tal manera que es típica la apariencia de estos programas con su *área de trabajo* en la que el usuario tiene a su disposición una serie de recursos, a modo de herramientas, para poder realizar un dibujo. Cada herramienta esta representada por un icono gráfico (figura 5.11) que ayuda a su identificación, como ejemplo el icono de un lápiz representa la posibilidad dé pintar puntos, o el de un bote de spray para poder puntar grupos grandes de puntos a la vez, el icono de una goma puede servir para poder borrar un trozo de dibujo, ..., etc. Las operaciones que se pueden realizar de dibujo vienen determinadas fundamentalmente por esas herramientas que se pueden ver de forma gráfica, y por los menús de opciones que también presentan estos programas como medio de comunicación con el usuario. Las operaciones más comunes son:

- Recorte de áreas de dibujo para copiar, o transformar.
- Dibujo de puntos, líneas y curvas.
- Dibujo de áreas circulares, rectangulares y poligonales.

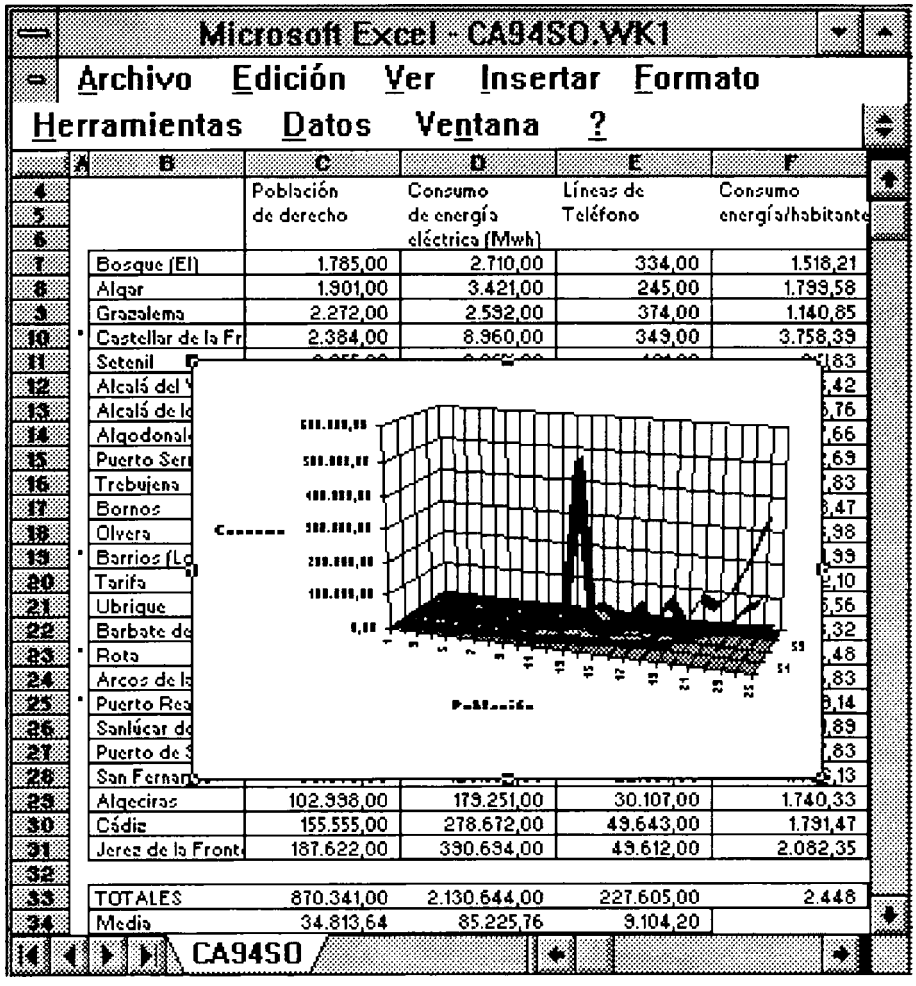


Figura 5.9: Hoja de cálculo

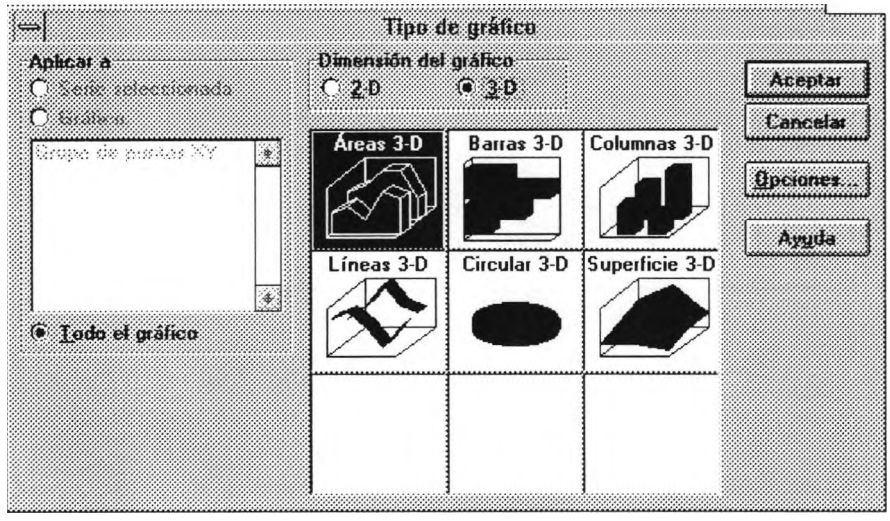


Figura 5.10: Ejemplo de tipos de representaciones de datos en una hoja decálculo

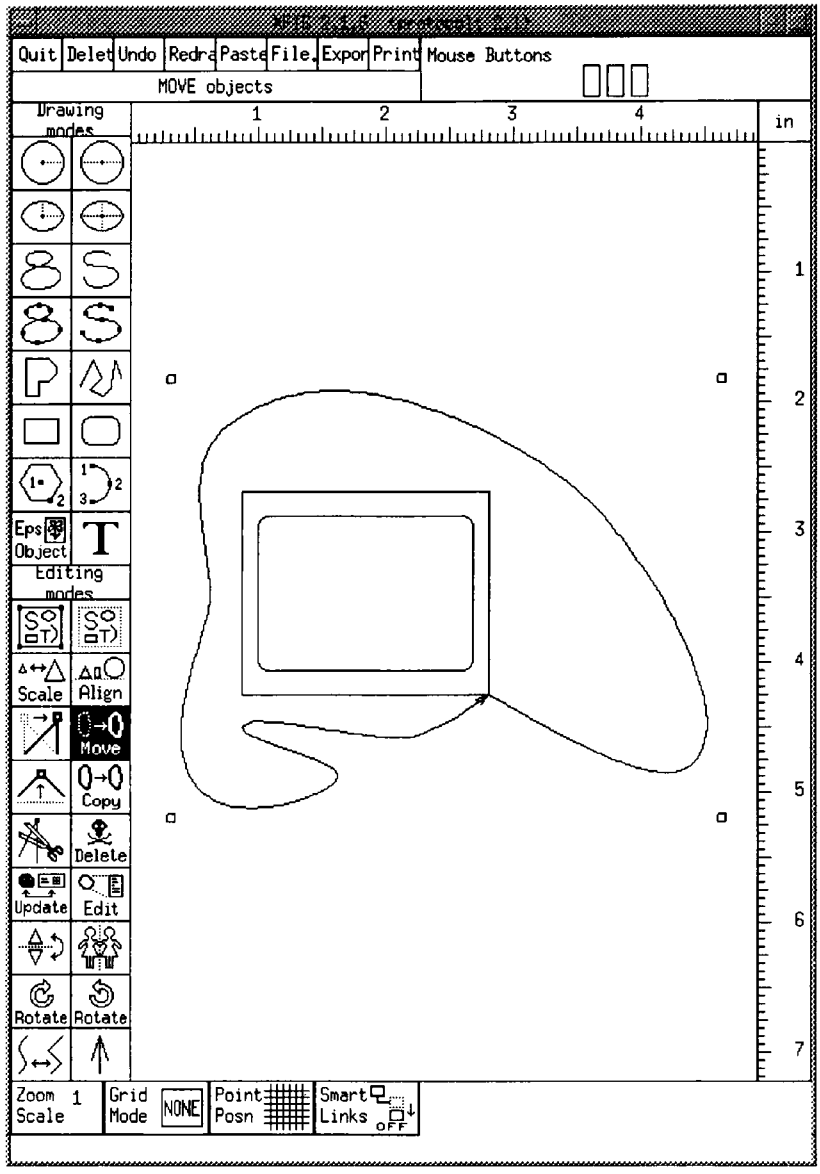


Figura 5.11: Aspecto del programa Xfig

- Cambio de pincel, tanto en tamaño como en color a utilizar.
- Introducción de texto, utilizando distintas fuentes y funciones de justificación.
- Importación de gráficos de otras fuentes.
- Transformaciones: rotaciones, reflexión, cambios de perspectiva, ...

El dibujo que se realiza con un programa de ordenador tiene una serie de características que hay que especificar:

- Tamaño del dibujo: en puntos (pixels), centímetros, pulgadas, ...
- Tamaño del papel en el que se va a situar el dibujo.
- Disposición del dibujo en el papel.
- Dibujo en colores o monocromático.
- ...

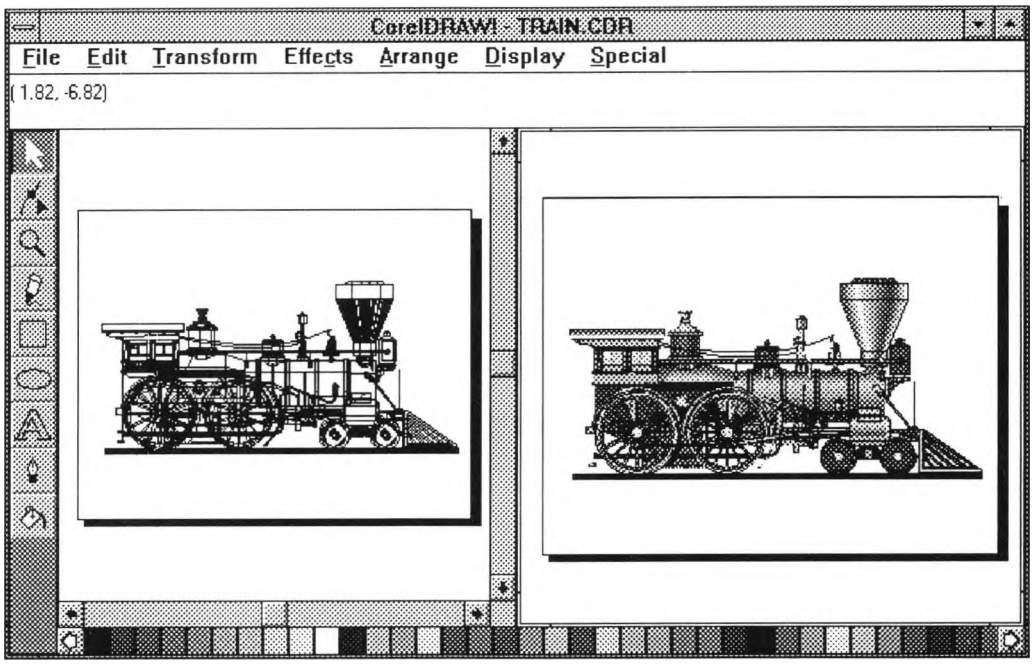
Podemos distinguir dos tipos de programas de dibujo según trabajen teniendo en cuenta un número determinado de puntos de imagen o no. La distinción es simple pero importante, ya que los que hemos puesto en primer lugar están limitados por ese número de puntos para ofrecer una calidad de dibujo, mientras que en el otro tipo la calidad sólo va a depender del soporte final que vaya a producir la salida impresa del dibujo.

La posibilidad de apreciar la calidad del dibujo producido depende mucho de la capacidad de colores del sistema de video del ordenador. Por lo general ésta viene dada por el tamaño posible de la pantalla en puntos, y por la memoria disponible para representarlos. Cuando la cantidad de memoria no es suficiente para representar una amplia gama de colores por pixels, los programas de dibujo ofrecen la posibilidad de simular colores utilizando *tramas* de colores. Las tramas se construyen combinando colores básicos en distintas proporciones para producir un efecto similar al de las combinaciones de los colores básicos que conforman la paleta hardware. Muchos programas permiten definir tramas nuevas además de las predefinidas. Se utilizan fundamentalmente para el rellenado de áreas.

En ciertos programas de dibujo, cuyas posibilidades resultan lo suficientemente complejas como para ofrecer la posibilidad de visualizar en pantalla el resultado de un dibujo de forma cómoda, se pierde el carácter WYSIWYG de los programas de dibujo en general y en su defecto se utiliza la opción de *preview* para ver como quedará el trabajo impreso, similar a lo que ocurre con ciertos procesadores de texto antiguos.

Evidentemente el trabajo que se ha realizado con el programa debe poder ser almacenado para algún posible uso posterior. El almacenamiento se realiza en forma de fichero con una representación de la información determinada por el programa. Basicamente se pueden considerar dos tipos de representaciones:

- Utilizando el *mapa de bits* de la imagen, es decir almacenando la información asociada a cada punto. En cada uno tendremos un color determinado durante la creación del dibujo, y en función del número de colores que pueda ser asociado a cada punto, se almacenará cierto número de bits por ese punto. Los dibujos así almacenados llegan a ocupar una gran cantidad de espacio, por ello se suele aplicar algún método de compresión, ya sea propio del programa, o alguno genérico como: GIF, JPEG, PCX, ...



- Utilizando la lista de elementos que componen el dibujo. Esta forma se asocia con aquellos programas, que por su mayor capacidad, estructuran las posibilidades de dibujo ofreciendo al usuario *primitivas* de dibujo con las que es posible realizar cualquier composición mediante combinación y transformación de las mismas. El concepto de primitiva se corresponde con la idea de herramienta que se ha nombrado anteriormente. Lo que se almacena en este caso son las primitivas usadas para construir el dibujo, junto con las transformaciones que se han realizado y los atributos asociados (color fundamentalmente). El espacio que ocupa esta representación suele ser menor.

Algunos programas de dibujo tienen la capacidad de poder concatenar una serie de imágenes o gráficos a modo de presentaciones, y mediante una pantalla especial y un retroproyector, permiten exponer "trasparencias animadas" gracias a los efectos que se pueden realizar con estos programas.

5.3.3 Software de diseño asistido por computadora

Son programas diseñados para funcionar en computadoras de gran potencia de cálculo y de representación de gráficos, en primer lugar por que se utilizan para aplicaciones de ingeniería, donde los problemas que se resuelven se caracterizan por su complejidad, y en segundo lugar porque casi todas las aplicaciones están orientadas a la representación del diseño. El diseño arquitectónico fue quizás la primera aplicación de este software, junto con el diseño de circuitos integrados, a todos ellos se les conoce bajo el nombre de programas *CAD*.

Como es típico en los programas de dibujo en general, el usuario se encuentra con una *mesa de trabajo* cuando utiliza el programa, en la que a cada herramienta disponible se le asigna el icono apropiado. La forma de trabajar con estos programas consiste en ir seleccionando una serie de objetos de los que hay a disposición, situarlos en algún lugar de la escena y transformarlos en función de ciertos parámetros y atributos aplicables. Una particularidad

de estos programas es la de poder construir nuevos objetos dados los existentes, pudiendo asignar a cada uno las características necesarias que lo definan. Por ejemplo una puerta puede ser un conjunto de áreas del espacio 3D dispuestas de una forma determinada, de manera que al hacer referencia al objeto puerta hacemos uso de la definición de otros objetos más básicos que son las áreas 3D. Algunos de los objetos que podemos encontrar en un programa CAD (en particular en Autocad para Windows) pueden ser: líneas, arcos, círculos, elipses, polígonos, puntos, texturas, cotas, trazos sólidos, formas (objetos gráficos definibles usando los símbolos del código ASCII), bloques (grupos de objetos gráficos), atributos (información textual asociada a los bloques), sólidos AME (objetos con identidad de cuerpo 3D), ...

Es característico encontrar en estos programas la capacidad de realizar *simulaciones* sobre los diseños. Un programa de diseño arquitectónico debe aportar la posibilidad de realizar trayectos por las construcciones modeladas, ya que esto puede ayudar a evaluar el producto que se va a construir en realidad. Hay que pensar que una vez construido un bloque de pisos es muy costoso hacer modificaciones sobre los defectos o errores que se detecten, salvo que impliquen un peligro suficiente como para requerir su demolición. En cambio en un ordenador esto es posible y mucho más sencillo de hacer, sin que el coste sea comparable. El programa Autocad para Windows permite realizar vistas aéreas globales, zooms, desplazamientos y localización detallada de objetos. Algo similar ocurre con los programas de diseño de circuitos, gracias a estos es posible comprobar el comportamiento de un diseño con suficiente realismo como para evitar desagradables sorpresas una vez construido, ya que existen muchos factores del entorno de funcionamiento del circuito que influyen sobre él.

Junto al mantenimiento de archivos de los diseños, con un programa de CAD existe la posibilidad de enviar datos a otros programas como las hojas de cálculo, para la presentación de informes relacionados con el diseño. Por otro lado usando lenguajes de bases de datos como SQL, se pueden mantener datos sobre facturas de material e inventarios, realizar estimaciones de costes, ... Autocad para Windows es capaz de utilizar datos de dBase III+ y IV, así como ODBC (Open DataBase Conectivity) para Oracle, Informix, ...

5.3.4 Software de tratamiento de imágenes

Este es un tipo de programa diferente de los anteriores en cuanto a la fuente para la obtención de imágenes. Los programas que hemos visto hasta ahora (excepto el primero) obtienen imágenes como resultado de la introducción de las mismas por el usuario utilizando sus facultades intelectuales y ciertas herramientas conectadas a la computadora que facilitan la interacción con la misma mientras que realiza esa labor. El que nos va a ocupar ahora utiliza medios de digitalización de imágenes en soporte externo a la computadora. La digitalización de las imágenes se realiza usando un escáner o una cámara conectados al ordenador con el interfaz adecuado. El monitor de vídeo que se utiliza con estos programas ha de ser de gran calidad de imagen, y por esto muchos programas precisan de la calibración del mismo para ofrecer un máximo rendimiento al usuario.

Una vez introducida la imagen el ordenador, utilizando el programa de tratamiento se pueden realizar transformaciones de los atributos de la imagen y operaciones como la alteración del cromatismo, brillo, contraste, cambios de prespectiva, mezcla de imágenes, ..., etc. Los programas de este tipo utilizados en aplicaciones específicas, con imágenes específicas, (en medicina, imágenes de tomografía) ofrecen más operaciones que tienen que ver con esas aplicaciones. Podemos incluir dentro de este grupo a los programas OCR , o de *reconocimiento de caracteres*. Con ellos y mediante un escáner, utilizando como fuente la imagen de un

texto digitalizado, es posible producir el mismo efecto que se obtendría con la introducción por teclado del mismo texto utilizando el teclado, pero de forma automática. Su uso es de indudable interés dado el ahorro de trabajo que se consigue, pero tienen sus limitaciones en función del tipo de letra del documento fuente, y de la calidad de la imagen digitalizada.

5.4 Software de comunicaciones

Cuando aparecieron los primeros ordenadores personales y comenzaron a utilizarse en las empresas y en las universidades, los programas eran simples y se diseñaban para un solo usuario, entonces no era tan importante la comunicación entre ordenadores. Los avances tecnológicos en hardware y software han hecho de éstos herramientas de trabajo cada vez más potentes: ha aumentado la velocidad de procesamiento y la capacidad de almacenamiento de datos y se han desarrollado programas multiusuario más completos. Lo cual ha llevado a que su utilización sea cada vez más extendida y ha provocado que la conexión entre ordenadores se convierta en una necesidad debido a sus ventajas. Las principales son la posibilidad de compartir recursos hardware y software entre los ordenadores conectados y el intercambio de información entre los usuarios.

La comunicación entre ordenadores se puede conseguir a través de redes , ya sean éstas públicas o corporativas, o a través de módems. Una empresa puede conectar los ordenadores de sus empleados mediante una red de área local, si todos los ordenadores están localizados en un solo edificio, por ejemplo, o instalando una red local en cada departamento e interconectándolas entre sí para formar un sistema en red que cubra una superficie más amplia. Si la empresa tiene centros extendidos por todo un país o incluso varios, se pueden utilizar las redes públicas de transmisión de datos (en España: Iberpac, Ibermic, ...) de las que son propietarias empresas de telecomunicaciones (en España, Telefónica). En definitiva, las redes conectan directamente a los ordenadores, ya sea a través de cables especiales (coaxial, par trenzado, fibra óptica) o de alguna forma de transmisión inalámbrica (ondas de radio, satélites), que pueden ser propiedad del usuario o de una empresa que cobra un precio por su utilización.

La forma más económica y menos compleja de conectar dos ordenadores es a través de un módem, aprovechando la red telefónica o utilizando conexiones celulares (del tipo que usan los teléfonos móviles). La necesidad de los módems surge debido a que las características de las líneas telefónicas no se adaptan bien al funcionamiento de los ordenadores, éstos tratan la información codificada mediante señales digitales, mientras que por las líneas telefónicas se transmiten señales analógicas. Por lo que es necesario transformar unas señales en otras para poder enviar información desde un ordenador a otro a través de la red telefónica. Por tanto, las funciones del módem son dos: modulación, proceso por el cual las señales digitales que se envían se transforman en analógicas y demodulación, proceso inverso a la modulación, que consiste en transformar las señales analógicas que se reciben en digitales. Un módem es un MOdulador/DEModulador, de ahí su nombre.

Los avances tecnológicos en telecomunicaciones se producen tan rápidamente que contínuamente se desarrollan nuevos productos que superan las capacidades de los ya existentes. Con el fin de evitar la proliferación de módems con diferentes características y permitir la compatibilidad entre los de diferentes fabricantes, el CCITT (Comité Consultivo Internacional Telegráfico y Telefónico) ha desarrollado un conjunto de normas (V.19, V.20, V.21, ...), que definen las características de los distintos tipos de módems tales como la velocidad de transmisión, tipo de transmisión, protocolo de corrección de errores o método de compresión

de datos. En el momento de adquirir un módem habrá que tener muy en cuenta las características del módem del ordenador remoto al que queramos conectarnos, de manera que ambos módems sean compatibles y la comunicación entre los dos ordenadores sea lo más eficiente posible, lo cual se traducirá en conexiones más económicas, sobre todo cuando se necesiten transmitir grandes volúmenes de información. Las características de los módems se explican con más detalle en el **Capítulo 2**.

5.4.1 Programas para módem

Tan importante como el módem mismo es el software de comunicaciones, encargado de proporcionar al usuario una herramienta sencilla de manejar que gestione el módem. Se puede decir que el principal factor para disfrutar de una buena conexión es el software que se emplee para realizarla. Existen sofisticados programas de comunicaciones, como WinCIM para acceder a CompuServe o WinGopher para disfrutar del servicio Gopher de Internet (en los siguientes apartados se explicará en qué consisten estos servicios), que son productos especializados para realizar una sola función. Pero hay otros programas que proporcionan funciones genéricas de conexión entre ordenadores que son programas de comunicaciones de propósito general.

Las funciones básicas de un programa de comunicaciones son:

- *Emulación de terminal* o *acceso remoto* a otro ordenador, que permite el control remoto del ordenador que se encuentra al otro lado de la línea telefónica. El ordenador que actúa como terminal realiza las funciones de entrada y salida de datos, enviando los comandos que introduce el usuario para que se ejecuten en el ordenador remoto y presentando los resultados del procesamiento en la pantalla del ordenador local. De esta forma se utilizan los recursos hardware y software del ordendor remoto, mientras que el ordenador local es el medio de acceso, a través del módem, a dicho ordenador.

- *Transferencia de ficheros*, que consiste simplemente en enviar un fichero de un ordenador a otro y permite el intercambio de información entre usuarios de ordenadores distantes.

Para realizar estas funciones los programas de comunicaciones instalados en los ordenadores conectados deben gestionar la transferencia de información entre ambos. Para ello deben seguir una serie de normas que aseguren la eficiencia entre el envío y la recepción de datos. Un conjunto de tales normas se llama *protocolo*. Si un ordenador envía los datos vía módem utilizando un protocolo determinado, el ordenador receptor debe trabajar con el mismo protocolo, de lo contrario los programas no se entenderían. Existe un gran número de protocolos, pero entre los más comunes se encuentran XMODEM (prácticamente todos los programas disponen de él), YMODEM, ZMODEM y KERMIT .

Estos programas realizan su principal tarea, la comunicación entre ordenadores, a través del lenguaje de control Hayes, que agrupa un conjunto de comandos para controlar el funcionamiento del módem. Los buenos programas presentan un aspecto amigable mediante menús de comandos que ocultan al usuario los detalles del lenguaje básico que utilizan los módems en la comunicación. Son programas que se encargan fundamentalmente de gestionar la E/S de datos y de informar al usuario sobre los datos transmitidos y los errores producidos y cuyas principales características incluyen:

- Capacidad de entenderse con cientos de módems, empleando cualquier estándar desde el V.21 al V.34 (desde 300 a 28.800 bps.).

- Opcionalmente pueden utilizar técnicas de compresión de datos o de detección de errores.
- Pueden transferir ficheros empleando una amplia variedad de protocolos.
- Ajuste automático de los parámetros de configuración del módem en cada conexión.
- Emulación de diferentes tipos de terminales.
- Posibilidad de realizar acciones previamente programadas en un fichero de conexión.
- Disponen de una agenda donde se pueden guardar los números telefónicos más usados junto con los parámetros de configuración para cada llamada.
- Permiten aprovechar todas las posibilidades que ofrecen los módems y son susceptibles de programar como, por ejemplo, el número de veces que ha de sonar el teléfono antes de que el programa responda automáticamente, el tiempo de espera de la respuesta, la hora a la que enviar un mensaje, etc.

En la actualidad, los programas de comuncicaciones van dirigidos a dos tipos de usuarios: personal del departamento de informática de una empresa y usuarios individuales que los utilizan con carácter personal. Los primeros tienen que desarrollar programas para los empleados, que se conecten al ordenador central de la empresa usando un terminal no inteligente, de manera que los empleados puedan trabajar desde centros alejados de la central o incluso desde sus propias casas. Esto significa que el personal de informática necesita herramientas de programación que le faciliten el desarrollo de tales programas, permitiéndole controlar el módem y emular cualquier tipo de terminal no inteligente, sin tener que preocuparse de los detalles de bajo nivel. Para ello, los buenos programas de comunicaciones poseen un lenguaje de programación interno con el cual se pueden crear macros para automatizar tareas más o menos complejas, de manera que los usuarios finales puedan realizarlas sin tener que ser expertos en el funcionamiento y configuaración de los módems. Algunos productos de comunicaciones incluyen, también, librerías de funciones que se pueden emplear para construir aplicaciones en un lenguaje de programación convencional como el C.

Los usuarios personales necesitan programas que les faciliten la conexión eficiente y sencilla a ordenadores remotos y a servicios de información. Estos programas ponen a disposición del usuario funciones de marcado y conexión automáticas, recepción rápida de ficheros y la posibilidad de disponer de gestores de mensajes *offline* para reducir el tiempo en el que hay que mantener activa la conexión y hacer que ésta sea más barata. Se necesitan también unas mínimas posibilidades de programación, de forma que se pueda capturar y almacenar información recibida y automatizar la introducción de respuestas. También disponen de un directorio telefónico y la posibilidad de que sea el propio programa el que decida los mejores parámetros para cada conexión. También hay programas de comunicaciones que incorporan funciones básicas de acceso a Internet. Algunos ejemplos de estos programas son Procomm Plus, BitCom, HyperAccess y Crosstalk.

Hasta ahora hemos dicho que los programas para módems permiten conectarse a ordenadores remotos y transferir información hacia y desde ellos, pero lo interesante es a qué ordenadores nos podemos conectar, o dicho de otra forma, qué servicios podemos obtener utilizando un módem y el software adecuado. En la actualidad, las posibilidades son muchas y siguen ampliándose día a día, es posible conectarse a miles de BBS extendidos por todo el mundo, suscribirse a un servicio electrónico de información y obtener información sobre los

temas más variados, o acceder a Internet, la mayor red de ordenadores de ámbito mundial, en la que podemos encontrar todo tipo de información y a través de la cual nos podemos comunicar con millones de usuarios. De todo esto hablaremos en siguientes apartados.

5.4.2 Programas para fax

Un facsímil o fax es una máquina que se utiliza para enviar o recibir documentos a través de la red telefónica. Un fax de sobremesa obtiene una imagen digital del documento en papel que se le introduce y la transmite por la línea telefónica. En el otro extremo de la línea, el fax que recibe la imagen digital la imprime en papel. Básicamente, un fax realiza las funciones de un scanner, un módem y una impresora.

Existen módems (norma V.29) con las características necesarias para enviar y recibir información hacia y desde un fax de sobremesa. Para utilizarlos es necesario un programa que los controle, pero los programas de comunicaciones por módem no incluyen funciones de envío y recepción de ficheros de documentos por fax, por lo que existen programas específicos para ello. Los programas para fax sirven esencialmente para enviar y recibir ficheros de documentos y, además, para mantener una agenda de direcciones y un fichero de los enviados y recibidos. El mayor inconveniente de los módem/fax es que sólo es posible enviar ficheros de documentos almacenados en el ordenador, no se puede enviar directamente un documento en papel. Este problema se puede solucionar si el ordenador tiene conectado un scanner.

Las características más importantes de los programas de gestión de fax engloban la posibilidad de enviar un documento hacia el fax desde cualquier aplicación como si se tratara de una impresora, enviar el mismo fax a una lista de destinatarios, programar la fecha y hora de transmisión y editar el texto de los fax enviados o recibidos. Algunos programas incorporan, también, un programa OCR (Reconocedor Optico de Caracteres) que permiten convertir y almacenar las imágenes recibidas en formato de texto para utilizar los documentos en otros programas. Entre los más completos están BitFax y WinFaxPro, ambos para Windows.

5.4.3 BBS (Bulletin Board System)

Una de las primeras aplicaciones con las que comenzó a hacerse popular el uso de módems y programas de comunicaciones fue la conexión a BBS, debido a la facilidad de acceso de los usuarios. Un BBS (Bulletin Board System, que podría traducirse como "tablón de anuncios electrónico") no es más que un ordenador en el que se ejecuta un programa que está pendiente del módem para atender las llamadas telefónicas que recibe, y al que los usuarios se pueden conectar para intercambiar información entre sí.

El intercambio de información entre los usuarios se puede llevar a cabo de diferentes formas, las cuales se corresponden con los objetivos de los BBS:

- Establecer foros de debate sobre ciertos temas a través de mensajes (mensajería pública). Cuando un usuario se conecta a un BBS puede leer los mensajes dejados por otros usuarios y contribuir con sus propias respuestas o ideas enviando otros mensajes que leerán los demás.

- Comunicación con otros usuarios a través de correo electrónico (mensajería privada). La idea básica es la misma que en los debates, pero en este caso el mensaje sólo lo puede leer su destinatario.

- Transferencia de ficheros en los que se almacenan documentos o programas de cualquier tipo. Se pueden encontrar ficheros con documentos de texto, imágenes digitalizadas, ficheros de sonido o música, de vídeo, de animaciones, etc. Los programas que se encuentran en los BBS pueden ser de libre distribución (*freeware*) o por los que, si interesa usarlos, hay que pagar un módico precio a su autor después de probarlo (*shareware*).

En definitiva, un BBS es un sistema al que pueden acceder los usuarios para recoger y dejar información por medio de mensajes y ficheros, para lo cual sólo es necesario un ordenador, un módem, un programa de comunicaciones y una línea telefónica conectada al módem.

El primer paso para entrar en una BBS, una vez establecida la conexión desde un programa de comunicaciones, consiste en identificarse con un nombre de usuario y una clave, salvo la primera vez, en la que se nos preguntarán más datos para que el SysOp o administrador del sistema nos dé de alta como usuario. A partir de ese momento podremos acceder a la información almacenada en el BBS.

Normalmente, la información está organizada en áreas temáticas, para que sea más fácil encontrar lo que se busca y, de la misma forma, sea igualmente sencillo colocar un fichero o un mensaje en un lugar donde los demás usuarios puedan encontrarlo rápidamente. Además de ficheros y mensajes hay también boletines, que son textos elaborados por colaboradores del BBS y que tratan temas de gran interés; en cierta forma son como artículos periodísticos y por ello tienen ese estatus especial de "boletín" que les asegura una permanencia en el sistema, al contrario que los mensajes, que cambian cada poco tiempo.

Una característica del programa de comunicaciones, que hay que tener muy en cuenta, es la capacidad de gestionar mensajes *offline*. Si no se dispone de esta posibilidad, nos conectaremos en modo de acceso directo para leer los mensajes mientras dura la conexión; en caso contrario, podemos establecer la conexión para que el programa descargue los mensajes en el ordenador local y después los leeremos. El atractivo de esta última alternativa está claro: el tiempo de conexión es menor y, por tanto, también el precio a pagar a la compañía telefónica. De igual forma, al enviar mensajes sucede lo mismo, y antes de establecer la conexión procederemos a preparar los mensajes de salida para enviarlos todos juntos.

Los BBS son sistemas cuyo ámbito de influencia suele ser local y el número de usuarios está limitado a cientos o como mucho a unos pocos miles de usuarios. Sin embargo, existe una cantidad inmensa de BBS por todo el mundo que si se conectan entre sí se pueden convertir en un gran sistema de distribución de información. Esto es precisamente Fidonet, una red que aprovecha la red telefónica para conectar miles de BBS de todo el mundo. Todos ellos comparten un formato de mensajes, un protocolo necesario para que un sistema entre en contacto con otro y puedan intercambiar datos y, un sistema de numeración para identificar a cada integrante de la red.

Fidonet está organizada de forma jerárquica en zonas (continentes), redes (países), regiones (autonomías o provincias), nodos (BBS) y puntos (cada terminal que se conecta a un nodo y tiene una dirección propia a la que se pueden enviar mensajes). Por ejemplo, 2:345/803.13 corresponde a la zona 2, Europa, red de BBS de España (34), región 5 (Andalucía), nodo (BBS) 803 y al punto 13 de ese BBS.

La transmisión de mensajes por la red se hace siguiendo su estructura jerárquica, de forma que se garantiza que todos los mensajes llegan a su destino. Los nodos utilizan programas para comunicarse entre sí, que se ejecutan de forma automática y regular para transportar los mensajes por toda la red. Habitualmente la transmisión se realiza en horas de tarifa reducida para minimizar el coste de utilización de la red telefónica.

La ventaja de Fidonet es que puede cubrir las necesidades de comunicación de muchos usuarios personales a un coste muy económico. Existen BBS locales de carácter gratuíto en muchas ciudades, lo cual quiere decir que el precio que hay que pagar por estar en contacto con el mundo es el coste de una llamada telefónica local de unos pocos minutos. Además, no hay que olvidar que muchos BBS permiten intercambiar correo electrónico con usuarios de Internet, lo cual amplía enormemente las posibilidades de comunicación.

No obstante, el carácter "casero" de los BBS (cualquiera con unos conocimientos de Informática no muy avanzados puede instalar un módem en su ordenador y convertirlo en un BBS) hace que el rendimiento global del sistema sea menor que el de un servicio electrónico de información comercial o el de Internet, que no utilizan sólo líneas telefónicas para las conexiones. Aparte, la cantidad y la calidad de la información que se puede encontrar en Internet o en un servicio de información es superior.

5.4.4 Servicios de información

Los servicios de información los proporcionan compañías privadas que cobran un precio por su utilización. Esencialmente, se paga una cuota de conexión al suscribirse al servicio, que da derecho a disponer de una cuenta en los ordenadores de la empresa, y una tarifa que depende del tiempo de conexión o de la cantidad de información transferida. Mediante el acceso a su cuenta, un usuario/cliente se conecta a la red (por lo general, una red WAN) formada por los otros usuarios y los ordenadores de la empresa, los cuales administran los servicios ofrecidos.

Un servicio de información proporciona una gran fuente de información y un medio de comunicación de ámbito mundial a sus usuarios. Sin embargo, aunque tienen una funcionalidad parecida, los BBS no se consideran verdaderamente un servicio de información ya que, suelen ser sistemas locales y no pertenecen a una empresa, además muchos de ellos son gratuitos y están especializados en temas concretos.

Los servicios de información se organizan en áreas temáticas formadas por mensajes y ficheros. En los ficheros se encuentran los documentos y programas relacionados con el tema y los mensajes pueden ser públicos o privados. Los primeros permiten mantener conferencias o debates entre los usuarios y los segundos se utilizan para intercambiar información de carácter personal. El contenido de las áreas temáticas es muy amplio, pudiendo encontrarse prácticamente de todo: informática, deportes, noticias de actualidad, cómics, juegos, salud, finanzas, negocios, literatura, cotizaciones bursátiles, etc. La mayoría de estas empresas permiten el intercambio de correo electrónico con usuarios de Internet y muchas de ellas ofrecen un servicio completo de acceso a Internet pagando un precio adicional.

Para acceder a las redes de estas empresas se puede utilizar cualquier programa de comunicaciones para módem, aunque casi todas, al contratar la conexión, proporcionan software propio, diseñado específicamente para acceder a sus servicios.

Los servicios de información más importantes actualmente por su amplio contenido temático de interés general, por el número de usuarios y por ofrecer acceso a la mayoría de los servicios de Internet son:

Compuserve Es el servicio de información más importante del mundo con aproximadamente 3 millones de usuarios. Ofrece el mayor número de áreas temáticas, clasificadas en básicas, extendidas y *premium*. El coste de acceso a cada área depende del grupo al que pertenezca: las básicas son gratuítas; las extendidas son la mayoría y tienen una tarifa por tiempo de conexión; y las áreas *premium* son bases de datos y servicios exclusivos

por cuyo acceso se cobra un precio por tiempo de conexión y un extra por el servicio realizado. Para trabajar con esta red se utiliza un programa propio de comunicaciones llamado WinCIM.

America on Line (AOL) Tiene más de dos millones de usuarios, casi todos ellos en EEUU. Mientras Compuserve es una red orientada a dar servicio a empresas, AOL ofrece servicios para el usuario doméstico con gran variedad de posibilidades de ocio y entretenimiento. En España no es posible obtener acceso directo a AOL, pero se puede conseguir a través de empresas intermediarias que cobran un precio extra por ello.

BIX Se trata de un servicio electrónico puramente profesional, ligado tradicionalmente a la revista de informática *Byte*. En él se puede obtener información sobre los últimos desarrolos y avances tecnológicos.

Servicom Es el servicio más importante de España con unos 10.000 usuarios. Originalmente era un BBS, pero hoy en día es una empresa con su propia red de comunicaciones y nodos en las principales capitales. Ofrece acceso completo a Internet y dispone de un variado contenido temático, aunque no tan amplio como el de Compuserve u otras compañías. El programa utilizado para acceder a Servicom es First Class.

Sarenet Lo más importante de este servicio no es la comunicación y el intercambio de información entre sus usuarios, sino la posibildad de conectarse con múltiples bases de datos de dominio público en todo el mundo con información muy diversa: bases de datos económicas, de tecnología, de empresas, de medicina, química, farmacia, de legislación, sobre patentes y marcas, de informática, investigación y ciencia, etc. Es importante señalar que Sarenet no es propietaria de esas bases de datos, sino que simplemente es un intermediario que actúa como centro de servicio desde el cual se puede acceder a ellas. Esto quiere decir que cada base de datos es independiente de las otras y no hay posibilidad de relacionar la información de varias. Además es necesario darse de alta en cada servicio que se vaya a utilizar y pagar un precio diferente por él.

A parte de estas empresas, existen otras muchas no pertenecientes al ramo de los servicios telemáticos que ofrecen información sobre sus productos y otros servicios a sus clientes a través de Ibertex.

Ibertex es el nombre que la compañía propietaria, Telefónica, da al servicio de videotex en España. Se proporciona a través de la red pública de transmisión de datos **Iberpac** y se puede obtener mediante un terminal especial o utilizando un módem V.32 (9.600 bps). Ibertex permite el acceso a servicios de información, gratuítos en algunos casos, ofrecidos por multitud de empresas y organismos oficiales. Se puede utilizar para consultar la guía telefónica o las páginas amarillas, bases de datos del BOE, catálogos de productos, para hacer compras desde casa y cargar el importe en una cuenta corriente o tarjeta de crédito, para realizar consultas y movimientos bancarios y hacer reservas en hoteles, restaurantes o de billetes de avión, trenes, etc.

Los servicios de Ibertex están organizados en niveles dependiendo del coste de su utilización. Así en los niveles 030 y 031 se encuentran los servicios gratuítos, es decir sólo se paga por el tiempo de utilización de la red y no por el servicio obtenido, y en los niveles 032 a 034 están los servicios retribuidos.

Recientemente, Telefónica ha puesto en funcionamiento un nuevo servicio de comunicaciones, **Infovía**, con una funcionalidad muy parecida a Ibertex: proporcionar un medio de

comunicación entre los servidores de información y los usuarios. Sus características más destacadas son su bajo precio (el coste será el de una llamada local) y que está basada en el protocolo de comunicaciones TCP/IP. Esto último significa que se puede incorporar información en formato gráfico (en contraste con Ibertex, que sólo admite texto) y, por tanto se puede utilizar software de acceso a Internet, como Netscape o NCSA Mosaic, para navegar por Infovía.

5.4.5 Internet

Desde el punto de vista del usuario final, Internet constituye una inmensa fuente de conocimiento y recursos de información compartidos a escala mundial. Es también la vía de comunicación que permite establecer la cooperación y colaboración entre un gran número de comunidades y grupos de interés por temas específicos, distribuidos por todo el mundo.

Las posibilidades de Internet como fuente de información o como medio de comunicación son múltiples. En ella se pueden encontrar artículos y publicaciones sobre los temas más diversos, que se podrán leer y copiar en el ordenador del usuario, también se pueden intercambiar mensajes con otros usuarios a través de correo electrónico, estén ellos en el edificio de al lado, en otra ciudad o en otro país. En Internet es posible encontrar toda clase de software para una gran variedad de ordenadores y sistemas operativos, se puede establecer una conexión con alguno de los miles de ordendores de la Red y transferir ficheros hacia o desde él, mantener una conversación en tiempo real con otra persona. No solamente es posible obtener información o utilizar algún tipo de servicio. El usuario también puede ofrecerlos si lo desea. Por ejemplo, se puede participar en un grupo de noticias o en una lista de correo. Los artículos que allí se envíen serán distribuidos automáticamente entre todos los miembros de la lista, y éstos pueden ser miles repartidos por todo el mundo. Si una empresa, universidad, centro de investigación, organismo de la administración, etc., tiene algo que decir, mostrar o anunciar, puede instalar en uno de sus ordenadores conectados a Internet un servidor de información y se podrá acceder a ésta desde casi cualquier parte del mundo.

Internet es un grupo de redes de ordenadores extendidas por todo el mundo e interconectadas permanentemente, o una gran *red de redes*, que permite comunicarse de forma directa con millones de personas y acceder a información almacenada en ordenadores de todo el mundo. Es importante resaltar que no estamos hablando de una red de ordenadores en el sentido usual, sino de una red de redes, donde cada una de ellas es independiente y reponsabilidad exclusiva de su propietario, pero comparten un protocolo de comunicaciones (*TCP/IP*) para el intercambio de datos entre dos ordenadores. Internet tampoco es un servicio electrónico de información como los descritos en la sección anterior, pues no pertenece a una compañia ni tiene por sí misma una tarifa. Sin embargo, muchos BBS y empresas propietarias de servicios de información tienen acceso a Internet, aunque no forman parte de ella (no están permanentemente conectados), y sus clientes/usuarios pueden utilizar algunos de los servicios de la Red, como por ejemplo correo electrónico.

5.4.5.1 Servicios y aplicaciones

Ya hemos dicho que Internet permite comunicarse con otros usuarios y acceder a la información almacenada en los ordenadores conectados por ella, para lo cual la Red ofrece diferentes tipos de servicios: *correo electrónico*, *transferencia de ficheros*, *conexión remota*, etc. Cada servicio se puede obtener utilizando un programa apropidado para ello. Existen programas de acceso

a Internet que permiten obtener diferentes servicios y otros programas diseñados para uno solo. Los programas que se puedan utilizar dependen del hardware y del sistema operativo sobre los que se esté trabajando, pero hay algunos de los que existen versiones para distintos sistemas.

En la actualidad Internet está muy extendida y cada vez es más popular, todos los días se conectan nuevos usuarios y servidores de información, de manera que el volumen de información aumenta a un ritmo muy acelerado. Por ello, el software y los servicios que ofrece Internet están en permanente cambio y evolución. Se mejoran los servicios existentes y se crean nuevas aplicaciones que facilitan el uso de los recursos y la búsqueda de información. Teniendo en cuenta esta situación, a continuación describiremos aquellos servicios y las aplicaciones que son más importantes y que se utilizan más en el presente.

Correo Electrónico (E-mail) Es el servicio más ampliamente usado de la Red ya que permite la comunicación entre los usuarios de forma sencilla. Un usuario puede enviar y recibir mensajes de cualquier otro usuario, no sólo mensajes personales, sino cualquier información que pueda almacenarse en un fichero de texto: programas fuente de ordenador, anuncios, revistas electrónicas, etc. Los programas que más comúnmente se utilizan para enviar correo y leer los mensajes recibidos son mail, elm o pine para Unix y Eudora para Windows.

Haciendo uso del correo electrónico, un usuario se puede suscribir a una **lista de correo**, un sistema organizado en el que un grupo de personas reciben y envían mensajes sobre el tema particular que se trate en la lista. Los mensajes que envía una persona son recibidos por todos los usuarios suscritos a la lista. Estos mensajes pueden ser artículos, comentarios, preguntas o cualquier cuestión relacionada con el tema de la lista. Todas las listas de correo tienen una persona que se ocupa de gestionarlas, a la cual hay que enviar un mensaje para suscribirse o borrarse de la lista.

Conexión remota (Telnet) Permite establecer una conexión remota con cualquier computador de Internet. Una vez que se ha establecido la conexión, se puede utilizar este computador como en una sesión de trabajo normal: el ordenador local actúa como un terminal del remoto y por tanto se tiene acceso a todos los datos y programas de éste, los cuales se ejecutarán en él. Desde luego, es necesario tener una cuenta de usuario en el ordenador remoto, es decir, un nombre de usuario y una palabra clave (password). Existen ordenadores conectados a Internet que permiten a cualquier usuario conectarse utilizando el nombre especial *guest* (invitado).

Transferencia de ficheros (FTP) Un servicio muy utilizado es el servicio **FTP** (File Transfer Protocol, Protocolo de Transferencia de Ficheros), que permite copiar ficheros desde un ordenador a otro. Normalmente, FTP se utiliza para copiar un fichero desde un ordenador remoto al del usuario, pero también se puede utilizar para transferir ficheros desde nuestro ordenador a otro conectado a la red. Es necesario identificarse para acceder a un ordenador y transferir ficheros desde o hacia él, pero muchas organizaciones ponen a disposición de todo el mundo los ficheros de sus ordenadores. A éstos se puede acceder utilizando el nombre de usuario anonymous y como palabra clave la dirección de correo electrónico del usuario. Este, junto con el correo electrónico, es el servicio más importante de Internet ya que permite el intercambio de cualquier tipo de información entre los usuarios y además de forma gratuita.

Búsqueda de ficheros (Archie) Hay miles de servidores de FTP anónimo por todo el mundo ofreciendo una cantidad inmensa de ficheros, por lo que se dispone de un servicio de búsqueda de ficheros por los ordenadores de Internet que se llama **archie**. Existen una serie de ordenadores distribuidos por la red que mantienen información sobre los ficheros almacenados en todos los ordenadores que ofrecen acceso a través de FTP anónimo. Cada servidor de archie ofrece información sobre la localización, nombre, tamaño y otras características de los ficheros disponibles en determinados servidores de FTP anónimo. Hay servidores de archie que tienen la misma información que otros con el objeto de facilitar el acceso, a los cuales se les llama *mirror* (espejo). La búsqueda se realiza buscando la subcadena de caracteres indicada en los nombres de los ficheros de la base de datos del servidor de archie. El resultado es una lista de los ficheros en cuyo nombre aparece la palabra buscada, junto con sus características e incluyendo los servidores de FTP donde se encuentran.

Noticias (News de USENET) Este es otro de los servicios principales de Internet, un sistema de miles de grupos de discusión sobre los temas más diversos (informática, historia, cine, agricultura, deportes, chistes, ...) en el que artículos individuales se distribuyen por todo el mundo. En cada nodo de la red, el administrador del sistema decide qué grupos de Usenet quiere hacer públicos y cuáles quiere recibir. Por esta razón, este servicio no está disponible en todos los ordenadores e incluso en aquellos desde los que se pueda acceder pueden no estar los artículos de todos los grupos. La funcionalidad de este servicio es la misma que la de las listas de correo. Los artículos de un grupo se pueden leer o enviar con un programa lector de noticias como rn, trn, nn o tin para Unix y Free Agent para Windows.

Gopher Es un sistema que permite al usuario obtener ficheros y otros servicios de Internet navegando a través de un sistema de menús y submenús. Fue uno de los primeros sistemas desarrollados de búsqueda de información a través de la Red. Una vez conectado el usuario a un servidor de Gopher, se presenta en la pantalla un menú de opciones que son punteros a submenús o a otros servidores Gopher de Internet (figura 5.12). El hecho de los submenús y servidores puedan estar en cualquier parte del mundo es completamente transparente al usuario. Esto permite que el usuario navegue por los ordenadores que dan este servicio saltando de uno a otro y recuperando ficheros con información muy diversa de todos ellos. Cuando se selecciona un fichero, cuyo nombre aparece en un menú, se transfiere al ordenador local y entonces, opcionalmente, el usuario puede cargarlo en un programa apropiado para tratarlo. Por ejemplo, si se trata de un fichero de texto, se utilizaría un editor para visualizarlo y si es una imagen digital se podría utilizar cualquier programa de gráficos. Para buscar servidores Gopher que puedan contener información de nuestro interés existe el servicio Veronica, una base de datos que almacena el texto de los menús del sistema Gopher.

World Wide Web Es el servicio más atractivo y el causante directo del aumento exponencial de usuarios de Internet, debido a la facilidad de uso que ofrece a través de un entorno gráfico de ventanas que incorpora capacidades multimedia. WWW es un sistema que permite al usuario buscar y recuperar información navegando a través de documentos de hipertexto. En un documento hipertexto, al seleccionar una palabra o una frase resaltadas se accede a un nuevo documento y se visualiza en pantalla. Este otro documento hipertexto puede estar en el mismo ordenador que el primero o en otro distinto, de for-

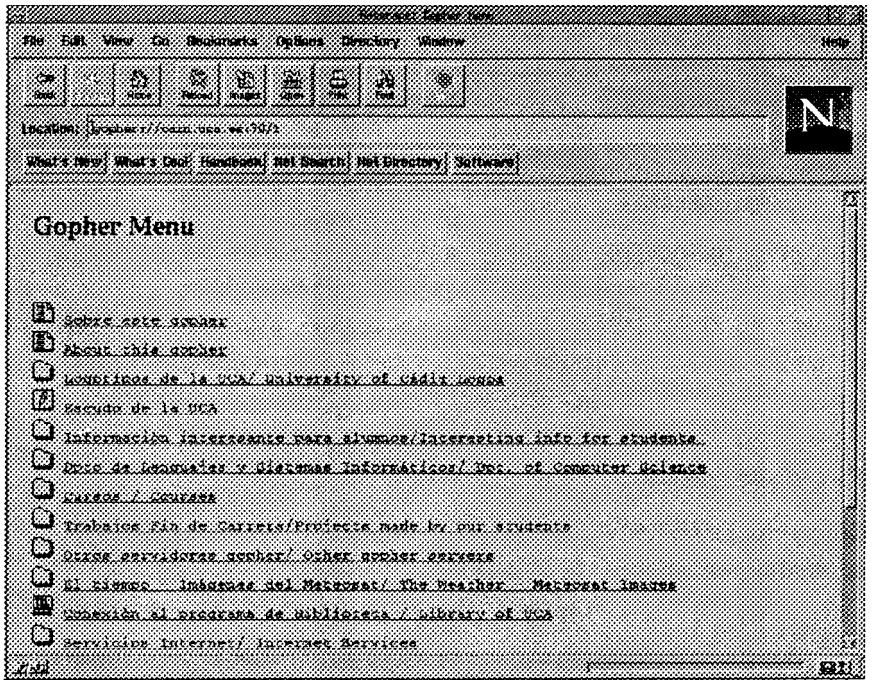


Figura 5.12: Un menú Gopher visto con el programa Netscape

ma que el usuario navega por la Red saltando de un ordenador a otro y visualizando documentos, que opcionalmente puede guardar en ficheros en su ordenador. Esto contrasta con los simples menús de texto de Gopher, ya que los documentos hipertexto son más ricos en información y pueden incorporar fuentes tipográficas y objetos multimedia como son gráficos, sonidos e imágenes en movimiento. Los programas más populares para conectar a servidores Web son Netscape (figura 5.13) y NCSA Mosaic.

Otros servicios Existen otros servicios orientados a la comunicación, como **talk** (para mantener una conversación entre dos usuarios) u otros orientados a la búsqueda de usuarios de la red, como Finger, Whois y X500 .

5.4.5.2 Conexión a Internet

Una conexión a Internet se puede obtener según las siguientes posibilidades:

- Se dispone de una cuenta como usuario autorizado en un ordenador multiusuario que está conectado a Internet (p.ej.en la Universidad, centro de investigación, departamento gubernamental o en una empresa que esté conectada a Internet).
- Se dispone de un ordenador personal con número IP asignado y conexión directa a Internet a través de una red local conectada, a su vez, a Internet (ejemplos: los mismos del caso anterior).
- Se tiene acceso a una cuenta en un ordenador multiusuario a través de un ordenador personal, un módem y una línea telefónica. En este caso, el ordenador personal sólo

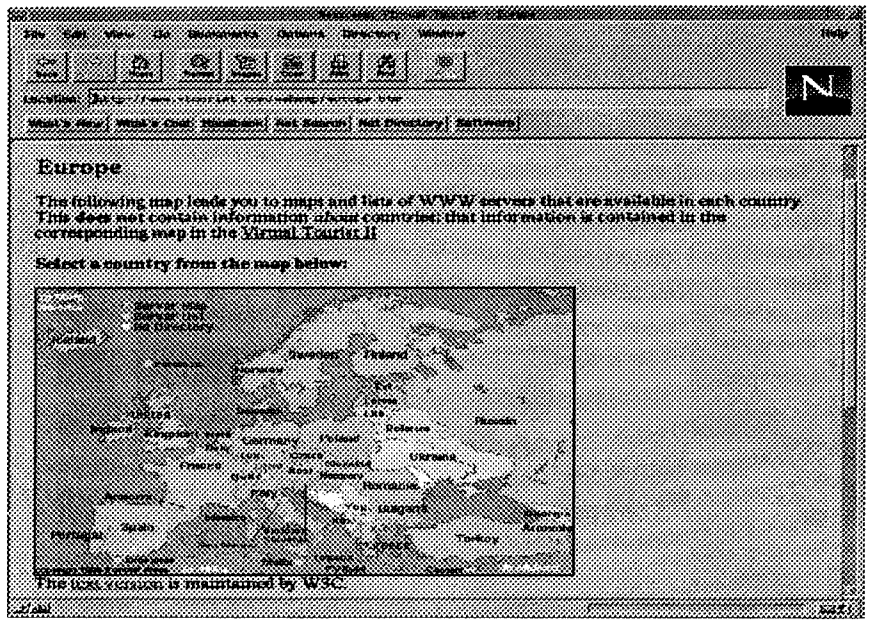


Figura 5.13: Página Web para acceder por países a todos los servidores de WWW de Europa

actuará como un emulador de terminal y no podrán usarse aplicaciones gráficas para navegar por Internet. El ordenador multiusuario puede pertenecer a una institución o a un proveedor comercial, el cual cobrará una tarifa por acceder a través de su ordenador a Internet.

- Se dispone de una conexión directa a Internet a través de una línea telefónica y un módem, mediante el protocolo PPP (Point to Point Protocol) o SLIP (Serial Line Internet Protocol). En este caso, el ordenador personal está directamente conectado a Internet y tiene asignado un número IP, pudiéndose usar aplicaciones gráficas. La conexión telefónica a través del módem puede establecerse con una institución o con un proveedor comercial de acceso a Internet.

En los dos últimos casos, la conexión vía módem se puede establecer directamente con el proveedor o con éste a través de Infovía.

Capítulo 6

Lenguajes de Programación

Para que un ordenador funcione, es necesario que tenga un programa almacenado en su memoria , ya que el programa le indica al ordenador lo que tiene que hacer. Este último, por su parte, se limita únicamente a ejecutar las operaciones que figuran en el programa.

Los programas, en general, están compuestos por secuencias de sentencias o frases. Estas sentencias pueden ser de dos tipos:

- **Sentencias imperativas** (o instrucciones). Estas sentencias indican al ordenador qué operaciones ha de llevar a cabo.

  A modo de ejemplo, la instrucción:

  $$y = 3 * x + 1$$

  - multiplica la constante 3 por el valor que la variable x tenga en ese momento.
  - suma la constante 1 al resultado anterior.
  - asigna el resultado final a la variable y.

- **Sentencias declarativas**. Proporcionan información sobre determinadas circunstancias del programa, tales como las definiciones de variables, las cuales obligan al ordenador a reservar cierta cantidad de memoria para las mismas. Este tipo de información es utilizada por los Traductores de Lenguajes , de los cuales se hablará en el apartado correspondiente.

Estos programas deben escribirse en un lenguaje de programación. Un Lenguaje de Programación puede definirse como *el conjunto de símbolos y reglas utilizados para construir un programa.*

Sin embargo, los ordenadores trabajan internamente en "lenguaje máquina" (en binario), y necesariamente tuvieron que nacer unos lenguajes intermedios que favorecieran la comunicación del hombre con la máquina. Estos son los lenguajes de programación. Uno de los primeros que apareció fue el Ensamblador (Assembler), muy cercano aún al lenguaje máquina. Años después aparecieron el COBOL y el RPG, por citar algunos, más lejanos a la máquina, pero más cercanos al hombre.

Figura 6.1: Ubicación del programa

6.1 Clasificación de los Lenguajes

La primera clasificación que se puede hacer con los lenguajes consiste en dividirlos en:

- Lenguaje Máquina.
- Lenguajes Simbólicos.

La programación en lenguaje máquina es difícil para el hombre, y por ello se han desarrollado los lenguajes simbólicos.

Los lenguajes simbólicos pueden clasificarse en lenguajes orientados a la máquina o de bajo nivel, y lenguajes orientados hacia el hombre o de alto nivel.

A su vez, puede establecerse una clasificación de los lenguajes de alto nivel atendiendo al modo de trabajo de los programas y a la filosofía con que fueron concebidos:

1. **Lenguajes Imperativos**. Utilizan las instrucciones como unidad de trabajo de los programas (COBOL, PASCAL, C, ADA).

2. **Lenguajes Declarativos**. Los programas se construyen mediante descripciones de funciones o expresiones lógicas (LISP, PROLOG).

3. **Lenguajes Orientados a Objetos**. El diseño de los programas se basa más en los datos y su estructura. La unidad de proceso es el **objeto**, y en él se incluyen los datos (variables) y las operaciones que actúan sobre ellos (SMALLTALK, C++).

4. **Lenguajes Naturales**. Se están desarrollando nuevos lenguajes con el principal objetivo de aproximar el diseño y construcción de programas al lenguaje de las personas.

6.2 Lenguaje Máquina

El lenguaje máquina utiliza exclusivamente el código binario. El microprocesador de cada ordenador posee su propio lenguaje máquina, el cual dispone de su juego particular de instrucciones.

Cada instrucción máquina ha sido diseñada por el fabricante del microprocesador. Por tanto, un microprocesador sólo puede reconocer los códigos de operación diseñados para él, y podrá interpretar las direcciones de los operandos sólo si están descritas en el modo que ha previsto el fabricante.

En lenguaje máquina una instrucción se compone de un **Código de Operación** (Operation Code) y, por lo general, de un máximo de tres operandos. El código de operación actúa

sobre los operandos. En los primeros tiempos de la informática, los programadores almacenaban el programa en la memoria del ordenador utilizando el único lenguaje que entiende el ordenador por si mismo: el Lenguaje Máquina.

El lenguaje máquina ofrece ciertas ventajas:

- Es directamente entendible por el ordenador.
- Es muy eficiente, ya que depende de la máquina, con lo cual se ejecuta muy rápidamente y permite aprovechar los recursos de ésta en su totalidad. No obstante, la eficiencia de un programa codificado en cualquier lenguaje de programación depende en gran medida de la habilidad del programador.

Sin embargo, presenta también algunos inconvenientes a considerar:

- Resulta complicado trabajar con el código binario.
- El programador debe conocer la arquitectura física del ordenador con cierto nivel de detalle. Fundamentalmente, las características del microprocesador (conjunto de registros y juego de instrucciones) y la programación de los diferentes dispositivos de entrada/salida.
- El lenguaje máquina no posee sentencias declarativas, sino que todas las sentencias son instrucciones.
- Es un lenguaje totalmente dependiente del microprocesador del ordenador. Si dos ordenadores tienen dos microprocesadores distintos, normalmente tendrán distinto lenguaje máquina. Por lo tanto, los programas escritos en este lenguaje son poco transportables de un ordenador a otro.
- No puede incluir comentarios que faciliten la legibilidad de los programas.
- Poseen un conjunto de instrucciones bastante reducido (aritméticas y lógicas, salto condicional e incondicional, carga y almacenamiento de operandos en memoria), en comparación con los lenguajes de alto nivel.

6.3 Lenguaje Ensamblador

Surge con el fin de solucionar los inconvenientes del lenguaje máquina. Las instrucciones escritas en este lenguaje guardan una estrecha relación con las instrucciones del lenguaje máquina, en que posteriormente serán traducidas: Cada instrucción en ensamblador equivale unívocamente a una instrucción en lenguaje máquina.

Las principales características de un programa escrito en este lenguaje son las siguientes:

- El código de operación es de tipo mnemotécnico (una palabra de pocas letras que indica la operación a realizar), a fin de facilitar su comprensión por el programador. Ejemplos: ADD (suma), SUB (substracción), MUL (producto), DIV (división), MOV (movimiento).
- Las direcciones de memoria de los operandos o datos pueden escribirse de forma simbólica mediante identificadores. Un identificador es una cadena de caracteres, construida según

determinadas reglas, que se utiliza para referenciar un elemento (dato, tipo de dato, subprograma) de un programa. Existen, por tanto, sentencias declarativas que asocian estos identificadores a direcciones de memoria. Ejemplo: CANT1 (129) y CANT2 (4). De este modo, en lenguaje máquina la instrucción "ADD CANT1 CANT2" se convierte en 00100011 10000001 00000100, suponiendo que el código de operación de la instrucción ADD es el número 35.

- Se pueden incluir comentarios.

Los lenguajes de este tipo suelen denominarse lenguajes Assembler o Ensamblador, al igual que los traductores correspondientes. Debido a que los diversos microprocesadores tienen su propio lenguaje máquina, los lenguajes tipo ensamblador son diferentes según el microprocesador de cada ordenador.

Sin embargo, un programador que trabaje en lenguaje ensamblador todavía se ve obligado a conocer los detalles de la arquitectura del ordenador en el cual se va ejecutar el programa, tal como sucede con la programación en lenguaje máquina.

6.4 Traductores de Lenguajes

Los Traductores son programas que, como su nombre indica, traducen un programa escrito en un lenguaje simbólico a su correspondiente en lenguaje máquina. Los traductores pueden clasificarse en Ensambladores, Compiladores e Intérpretes.

Mientras que los ensambladores se utilizan para traducir los programas escritos en lenguaje ensamblador, los compiladores e intérpretes son utilizados para traducir los programas escritos en lenguajes de alto nivel y lenguajes de cuarta generación.

Los traductores de algunos lenguajes se idearon inicialmente como intérpretes (BASIC, PROLOG, LISP, etc.) y otros como compiladores (MODULA-2, ADA, PASCAL, C, FORTRAN, etc.), sin embargo, actualmente pueden existir tanto compiladores como intérpretes para un mismo lenguaje.

6.4.1 Ensambladores.

Los Ensambladores se utilizan para traducir el programa escrito en lenguaje ensamblador a lenguaje máquina.

Una de las tareas que debe realizar el ensamblador consiste en sustituir los identificadores de las direcciones de memoria de los operandos por los valores concretos de tales direcciones durante el proceso de generación de instrucciones máquina.

Los Macroensambladores pretenden solucionar un problema que presentan tanto el lenguaje máquina como el lenguaje ensamblador, que es el reducido conjunto de instrucciones que ambos poseen. Para ello disponen de unas instrucciones especiales (macroinstrucciones), que equivalen a varias instrucciones máquina, y realizan operaciones más complejas.

6.4.2 Compiladores.

Traducen un programa escrito en un lenguaje de alto nivel o en un lenguaje de cuarta generación, denominado **Programa Fuente**, a un programa escrito en lenguaje máquina o en lenguaje ensamblador (que es fácilmente traducible a lenguaje máquina), denominado **Programa Objeto**. Este último puede almacenarse en memoria auxiliar o masiva para ser

Figura 6.2: Paso de Ensamblador a Lenguaje Máquina.

ejecutado posteriormente sin necesidad de volver a realizar la traducción. Una de las tareas propias del compilador consiste en asignar memoria a las variables y constantes que aparecen en el programa fuente.

Un compilador, además del proceso de traducción, realiza una serie de funciones que en su mayoría están enfocadas a la detección de errores en la escritura del programa fuente. Por lo general está constituido por los siguientes módulos:

1. **Analizador Lexicográfico o Scanner**. Examina el programa fuente para localizar las unidades básicas de información pertenecientes al lenguaje, denominadas **Unidades Léxicas o Tokens**. Un token es un elemento o cadena con significado propio en el programa fuente. Son tokens las palabras reservadas del lenguaje, los identificadores o los operadores. El analizador lexicográfico genera como resultado una Cadena de Tokens, que es la información que recibe el siguiente módulo del compilador.

2. **Analizador Sintáctico o Parser**. Recibe la cadena de tokens procedente del analizador lexicográfico y busca en ella los posibles errores sintácticos que aparezcan. Estos errores suelen deberse, entre otros motivos, a duplicidad de identificadores de distintas variables o a instrucciones escritas de manera incorrecta.

3. **Analizador Semántico**. Localiza los posibles errores de significado que aparezcan en el programa fuente como, por ejemplo, intentar sumar números con letras o combinar datos (constantes o variables) de distinto tipo dentro de la misma expresión.

   En cualesquiera de los tres analizadores, todo error detectado se comunica al programador por medio de un Listado de Compilación, en el que se indica el tipo de error y su localización dentro del programa fuente. En ocasiones, pueden existir determinados errores los cuales no perjudican al resto del proceso de compilación, e incluso permiten el funcionamiento del programa final. Los mensajes correspondientes a este tipo de errores se denominan **Advertencias o Warnings**.

4. **Generación de Código Intermedio**. Si no se han producido errores en alguna de las etapas anteriores, este módulo realiza la traducción a un código interno propio del compilador, denominado **Código Intermedio**, a fin de permitir la transportabilidad del lenguaje a otros ordenadores.

   Para un determinado lenguaje de alto nivel se hace común todo el proceso de análisis y generación de código intermedio, y es la generación del código objeto la que se particulariza para cada tipo de microprocesador.

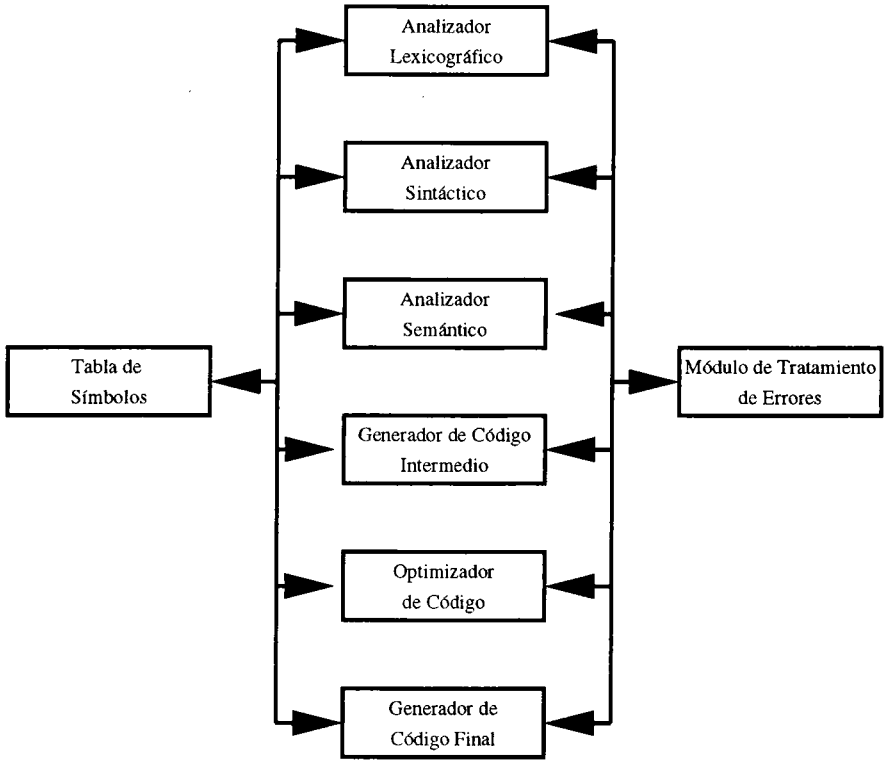


Figura 6.3: Partes de un compilador

5. **Optimizador de Código**. Su misión consiste en recibir el código intermedio y optimizarlo atendiendo a determinados factores, tales como la velocidad de ejecución o el tamaño del programa objeto.

6. **Generador de Código Final**. En esta última etapa se genera el código objeto mediante la traducción del código intermedio optimizado.

7. **Módulo de tratamiento de errores**. Facilita la detección, y en algún caso, la recuperación de los errores producidos en las distintas fases de la compilación. El compilador, cuando detecta un error, trata de buscar su localización exacta y su posible causa para presentar al programador un **Mensaje de Diagnóstico**, que será incluido en el listado de compilación.

8. **Tabla de Símbolos**. Es la estructura que almacena toda la información relativa a constantes, variables, estructuras de datos y otros elementos pertenecientes al programa que se está compilando. Esta información suele incluir el tipo de cada elemento, sus dimensiones y otras características. La tabla de símbolos está relacionada con todas las fases del proceso de compilación, y por tanto, es utilizada por cada uno de los módulos.

La figura 6.3 muestra la estructura general de un compilador, en la cual se pueden observar las interrelaciones entre los distintos elementos.

< INTERPRETE >

- I =1
- MIENTRAS no sea la última instrucción del Programa Fuente.
  - Leer instrucción (I) del Programa Fuente.
  - Traducir la instrucción (I) a Lenguaje Máquina.
  - Ejecutar las instrucciones máquina correspondientes.
  - I = I + 1
- FIN MIENTRAS

< FIN INTERPRETE >

6.4.3 Intérpretes

Los intérpretes, a diferencia de los compiladores, traducen el programa fuente instrucción a instrucción. Cuando leen una instrucción fuente ejecutan las instrucciones en lenguaje máquina correspondientes a la primera. Por lo tanto, los traductores de este tipo interpretan las instrucciones simbólicas del programa fuente. No generan ningún fichero con el programa objeto en memoria auxiliar para su posterior uso.

Los intérpretes poseen las siguientes características:

- Las sentencias que hay dentro de un bucle se interpretan tantas veces como se ejecute el bucle.
- La optimización, si existe, se lleva a cabo dentro de cada sentencia, no del programa completo, a diferencia de lo que sucede en el compilador.
- Cada vez que se desea ejecutar un programa, éste se debe volver a interpretar. Ello redunda en una considerable pérdida de tiempo.

Al igual que ocurría en el proceso de compilación, el intérprete debe asignar memoria a las variables y constantes que aparecen en el programa fuente.

A la hora de depurar un programa (corregir sus errores una vez escrito), el intérprete es mucho más cómodo, pues la traducción a lenguaje máquina se lleva a cabo DURANTE la ejecución del programa fuente. En caso de error, se modifica la instrucción fuente que lo ha producido y se vuelve a ejecutar el programa.

Sin embargo, si se trata de un programa compilado, en el caso de que se produzca un error hay que volver a compilar el programa fuente corregido antes de su ejecución. Entonces se lanza la ejecución del programa objeto, escrito en lenguaje máquina.

No obstante, un programa compilado se ejecuta mucho más rápidamente que un programa interpretado. Esto se debe a que en el intérprete, la ejecución de las instrucciones de un programa lleva consigo el proceso de traducción de las mismas. Esta traducción ya se ha realizado en su totalidad cuando se va a ejecutar el programa objeto que resulta de una compilación.

6.5 Lenguajes de Alto Nivel o Evolucionados

Las instrucciones de que consta un programa escrito en un lenguaje de alto nivel no tienen nada que ver con el funcionamiento interno del ordenador, en contraposición a los lenguajes de bajo nivel y los lenguajes máquina. Los lenguajes de alto nivel permiten escribir instrucciones orientadas al problema que se quiere resolver y no al ordenador que va a ejecutar el programa correspondiente.

Por este motivo, los lenguajes de alto nivel emplean una terminología mucho más comprensible y que se aproxima más al propio lenguaje humano (normalmente en inglés). Cada instrucción de programa escrito en lenguaje evolucionado se traduce en general a varias instrucciones máquina. Las instrucciones de un lenguaje de alto nivel son, por tanto, instrucciones muy potentes.

Asimismo, a diferencia del lenguaje Ensamblador, en un programa escrito utilizando un lenguaje de alto nivel se pueden definir las variables que se deseen, sin necesidad de especificar a qué posiciones de memoria se deben asignar. Esta tarea es realizada automáticamente por el traductor del lenguaje.

Así como los lenguajes máquina proliferan paralelamente al número de modelos diferentes de microprocesadores, los lenguajes de alto nivel pueden ser universales, ya que estos últimos no dependen de la arquitectura interna de la máquina. Es decir, los lenguajes de alto nivel más comunes pueden utilizarse para todos los ordenadores, siempre que estos últimos tengan el traductor correspondiente.

La compatibilidad de estos lenguajes en todos los ordenadores no se puede decir que sea total, pues existen excepciones en que, aun siendo básicamente iguales los lenguajes empleados, difieren ligeramente de unos a otros ordenadores.

Además de las ventajas de utilizar lenguajes de alto nivel, existen una serie de inconvenientes ligados a su utilización. Los programas objeto son generalmente menos eficientes en ocupación de memoria y en tiempo de ejecución, ya que en los lenguajes de bajo nivel el programador "sigue más de cerca" el trabajo de la máquina y puede aprovechar mejor todos sus recursos.

Con el fin de ilustrar la potencia de un lenguaje de alto nivel, se presenta un ejemplo con las instrucciones precisas para resolver un problema matemático en C, y comparativamente, las instrucciones que hacen falta en un Lenguaje Ensamblador determinado. Resolución del problema escrito en C:

```
x = ((a + b) * (c + d)) / (e + f);
```

Resolución del problema escrita en Lenguaje Ensamblador.

```
MOV   ax, dato_a
ADD   ax, dato_b   —   a + b
MOV   prod, ax

MOV   ax, dato_c
ADD   ax, dato_d   —   c + d
MUL   prod, ax     —   (a + b) * (c + d)

MOV   ax, dato_e
ADD   ax, dato_f   —   e + f
MOV   div, ax

MOV   ax, prod
DIV   ax, div      —   división
MOV   res_x, ax    —   resultado final
```

Nota: dato_a, dato_b, dato_c, dato_d, dato_e, dato_f, prod, div y res_x son identificadores de posiciones de memoria.

Sin embargo, el programador puede reorganizar el código de manera inteligente con el fin de realizar un mejor aprovechamiento de los recursos del sistema. En este ejemplo se consigue ahorrar una instrucción, debido a lo cual, el programa resulta más pequeño y se ejecuta con mayor rapidez (mayor eficiencia).

```
MOV   ax, dato_e
MOV   ax, dato_f   —   e + f
MOV   div, ax

MOV   ax, dato_a
ADD   ax, dato_b   —   a + b
MOV   sum, ax

MOV   ax, dato_c
ADD   ax, dato_d   —   c + d
MUL   ax, sum      —   (a + b) * (c + d)

DIV   ax, div      —   división
MOV   res_x, ax    —   resultado final
```

Todos los lenguajes simbólicos tienen, como todo lenguaje natural, un vocabulario (conjunto de símbolos) y unas reglas sintácticas y semánticas:

1. Palabras reservadas o palabras clave.
2. Identificadores (constantes, variables y subrutinas).
3. Declaraciones (constantes, variables y subrutinas).
4. Frases o sentencias.

5. Subrutinas o subprogramas (procedimientos y funciones).

6. Reglas sintácticas y semánticas.

A continuación se incluye una breve referencia de los lenguajes de alto nivel más extendidos.

6.5.1 FORTRAN.

El FORTRAN (FORmula TRANslator), el más antiguo de los lenguajes de alto nivel, ha sido durante mucho tiempo el primer lenguaje para programar problemas de cálculo numérico. Fue definido en el año 1955 en Estados Unidos por la compañía IBM, y en la actualidad se sigue utilizando.

Las sentencias FORTRAN se componen de variables, constantes y palabras reservadas. Los símbolos que se pueden utilizar son de tres tipos: caracteres alfabéticos, numéricos y especiales: "+", "-", "*", "/", etc.

Existen distintos tipos de variables y constantes: enteras, reales y alfanuméricas. Estas variables y constantes se definen mediante nombres (identificadores), construidos a partir de reglas bien definidas.

Los códigos de operación (operadores) pueden ser de tres tipos: aritméticos (suma, resta, producto, ...), lógicos (AND, OR, NOT, ...) y relacionales (GT, LT, EQ, ...).

Las diferentes sentencias se clasifican en:

- **Sentencias de control:** Se utilizan para alterar la secuencia del flujo del programa.
- **Sentencias aritméticas:** Sirven para la ejecución de cálculos aritméticos.
- **Sentencias de E/S:** Se encargan de la transferencia de datos entre la memoria principal y las unidades de E/S.

A lo largo de su existencia han aparecido diferentes versiones, entre las que destaca la adoptada en 1966 por el comité ANSI (American National Standarization Intitute), en la que se definieron nuevas reglas del lenguaje y se logró la independencia del mismo con respecto a la máquina. En 1977 apareció una nueva versión más evolucionada, denominada FORTRAN V o FORTRAN 77. En ella se incluyen instrucciones para el manejo de cadenas de caracteres y archivos, así como otras para la utilización de técnicas de programación estructurada. En la última normalización del lenguaje, FORTRAN 90, se incluyen características tales como la recursividad, tratamiento paralelo de tablas y utilización de memoria dinámica.

El FORTRAN permite utilizar subrutinas separadas. Cada una de ellas puede escribirse y compilarse de modo independiente.

6.5.2 COBOL.

Dentro de los lenguajes de gestión el más conocido es el COBOL. En 1959 un grupo de profesionales de la informática se reunieron para crear la conferencia CODASYL (Conference On Data System Language), y en la primera reunión acordó crear un lenguaje común para la resolución de los problemas comerciales.

El COBOL (Common Business Oriented Language) utiliza una sintaxis próxima a la de la lengua inglesa, y trata de evitar en lo posible el uso de símbolos especiales. No obstante, ya que es un lenguaje destinado al ordenador, es mucho más preciso que el inglés corriente. Es

autodocumentado y ofrece grandes facilidades en el manejo de archivos, así como en la edición de informes escritos.

Un programa COBOL se estructura en cuatro divisiones, que deben aparecer en un orden determinado. Cada división posee una subestructura de secciones, párrafos y cláusulas que es estándar para todos los programas COBOL.

A lo largo de su existencia ha sufrido diversas actualizaciones desde su primer estándar, aprobado por el comité ANSI en 1968. La última versión, COBOL ANSI-85, facilita el diseño estructurado de los programas.

Se puede afirmar que en la actualidad continúa siendo el lenguaje más utilizado en las aplicaciones de gestión.

6.5.3 LISP.

Fue creado a finales de la década de los cincuenta por John Mc-Carthy en el Instituto Tecnológico de Masachussets (MIT) y ha experimentado en los últimos años una gran evolución con la aparición sucesiva de nuevas versiones.

Se trata de un lenguaje específico para el procesamiento simbólico, en el cual, los elementos básicos a tratar son símbolos que representan objetos arbitrarios del campo o dominio de interés que se esté tratando. Estos símbolos se denominan átomos. El átomo constituye la unidad de información a procesar, y se compone de cadenas de caracteres.

El nombre LISP procede de la contracción de "procesamiento de listas" (LISt Processing), y es un lenguaje de programación fácil de aprender, del que existen compiladores e intérpretes para todos los ordenadores.

Dado que existen numerosos dialectos del lenguaje, han aparecido últimamente versiones estándar de LISP, con el objetivo de conseguir la transportabilidad de los programas entre distintos ordenadores. Pueden citarse como versiones estándar el Common LISP, el DG Common LISP de Data General y el LISP transportable estándar.

6.5.4 BASIC.

EL BASIC (Beginner's All-purpose Symbolic Instruction Code) es también un lenguaje de programación de propósito general, con instrucciones que recuerdan las fórmulas elementales del álgebra, con algunas palabras en inglés, como LET, GO TO, READ, PRINT, FOR, IF, THEN, etc.

Fue diseñado por los profesores John Kemeny y Thomas Kurtz del Dartmouth College (Estados Unidos) en 1965, con el objetivo principal del proporcionar a los principiantes un lenguaje fácil de aprender. Es muy útil para resolver problemas científicos, de matemáticas e ingeniería, aunque también se puede aplicar en otros ámbitos.

La amplia difusión de este lenguaje se debe por una parte a su facilidad, y por otra a que se trata de un lenguaje interpretado. El intérprete BASIC puede funcionar de dos modos: directo o calculador y programa. En el primer modo, el intérprete puede evaluar una sentencia o un conjunto de éstas sin que tengan que formar parte de un programa concreto. Funciona a modo de calculadora y permite la realización de cálculos sencillos. En el segundo modo, una vez elaborado el programa, el usuario escribe el comando "RUN" para comenzar la ejecución del mismo.

El éxito del BASIC se debe también, además de a su facilidad, a que es el lenguaje residente de muchísimos microordenadores personales. Existen multitud de intérpretes y compiladores

del lenguaje. En la actualidad se utiliza una versión avanzada denominada Visual Basic, la cual opera en ordenadores personales bajo el entorno MS-Windows.

6.5.5 PASCAL.

El lenguaje Pascal fue ideado originariamente con finalidades didácticas. Fue diseñado en 1970 por el matemático suizo Nicklaus Wirth, profesor del Instituto Politécnico de Zurich, el cual se basó en el lenguaje ALGOL, en cuyo diseño había participado en los años sesenta. El nombre del lenguaje proviene del filósofo y matemático francés del siglo XVII Blaise Pascal, que inventó la primera máquina de tipo mecánico para sumar. Se trata de un lenguaje estructurado, basado en un planteamiento formal, claro y riguroso que se inspira en los principios más modernos de la programación.

Los cánones de la programación estructurada pueden resumirse en tres:

- Definición de una serie de construcciones o estructuras estándar sobre los que articular, por bloques, la totalidad del programa: secuencia, selección (IF-THEN-ELSE) e iteración (WHILE-DO, REPEAT-UNTIL).

- Modularidad o Subdivisión de un problema complejo en subproblemas más sencillos. Es el diseño TOP-DOWN o descendente, basado en el principio "Divide y vencerás".

- Definición de unos tipos y estructuras de datos adecuadas para resolver el problema completo.

De todos modos, el PASCAL no deja de ser un lenguaje de propósito general, no especializado en resolver un problema concreto, a diferencia del FORTRAN o el COBOL. Su originalidad consiste en sus fines didácticos y la creación de buenos hábitos de programación en los usuarios de este lenguaje.

6.5.6 C

El lenguaje C fue creado por Dennis Ritchie en los laboratorios Bell Telephone en 1972, cuando trabajaba, junto con Ken Thompson, en el diseño del sistema operativo UNIX.

Es un lenguaje moderno, cuyo diseño hace que resulten naturales para el programador aspectos tales como la planificación escalonada, la programación estructurada y el diseño modular. Es muy eficiente, pues su diseño aprovecha las posibilidades del ordenador donde se está utilizando.

Además es un lenguaje portátil, pues la inmensa mayoría de los programas C desarrollados en un sistema pueden ejecutarse en otros sin ninguna modificación o modificaciones mínimas.

Otras características importantes son la flexibilidad y potencia, pues el C posee control sobre aspectos del ordenador asociados generalmente con lenguajes ensambladores. También dispone de una amplia biblioteca de funciones, concebidas para ser utilizadas en el desarrollo de una gran variedad de aplicaciones (p.e. creación de gráficos y manejo de cadenas de caracteres).

La utilización óptima de este lenguaje se consigue dentro de su entorno natural, que es el Sistema Operativo UNIX.

6.5.7 PROLOG.

Fue creado por J. Robinson en Francia durante la década de los setenta. Ha sido el lenguaje de programación para la Inteligencia Artificial más utilizado en Europa. Se trata de un lenguaje declarativo o de programación lógica (PROgramming LOGic), en el que se utiliza la teoría matemática de las relaciones.

En un programa escrito en PROLOG no se indica lo que hay que hacer, sino que se solicita lo que se necesita, habiéndose declarado previamente un conjunto de objetos (hechos y reglas) y sus relaciones. El lenguaje lleva incorporada la programación de operaciones y todo consiste en pedir, de tal forma que la máquina de inferencias busca en la base de conocimientos dando respuesta a las preguntas.

La principal ventaja de este lenguaje frente al resto es que permite el desarrollo de sistemas expertos a profesionales del diseño de programas y a no profesionales, puesto que no se necesita programar el algoritmo de búsqueda ni la comparación con patrones.

A diferencia de LISP, el lenguaje PROLOG se encuentra disponible en multitud de dialectos para todo tipo de máquinas, pero aún no existe ninguna versión estándar para dotarlo de la transportabilidad que hoy en día se hace necesaria.

6.5.8 MODULA-2.

El lenguaje MODULA fue diseñado en 1977 bajo la dirección de Nicklaus Wirth, creador también del lenguaje PASCAL, con el propósito de incluir las necesidades de la programación de sistemas y dar respuesta a las críticas recibidas respecto de las carencias del lenguaje PASCAL. En 1979 se realiza una nueva versión que pasa a denominarse MODULA-2 y que perdura en la actualidad.

Además de incluir las características de su predecesor, este nuevo lenguaje incorpora las principales carencias de aquél, como la posibilidad de Compilación Separada, creación de bibliotecas, Programación Concurrente, mejora del manejo de cadenas de caracteres, los procedimientos de entrada/salida y la gestión de memoria. Además posee grandes facilidades para la programación de sistemas.

También, debido a sus cualidades didácticas, ha sido ampliamente aceptado por la comunidad universitaria como herramienta idónea para la enseñanza de la programación.

6.5.9 ADA.

Es el último intento de obtener un único lenguaje para todo tipo de aplicaciones, e incluye los grandes avances en técnicas de programación. Su diseño fue encargado por el Departamento de Defensa de los Estados Unidos después de una selección rigurosa entre varias propuestas realizadas sobre una serie de requerimientos del lenguaje y de haber evaluado veintitrés lenguajes existentes.

De éstos se seleccionaron como base para la creación del nuevo lenguaje el PASCAL, el ALGOL y el PL/I. La estandarización se publicó en 1983 con el nombre de ADA, en honor de la considerada primera programadora de la historia, Augusta Ada Byron.

Entre las características del lenguaje se encuentran la compilación separada, los tipos abstractos de datos, programación concurrente, programación estructurada y libertad de formatos de escritura. Como principal inconveniente presenta su gran extensión.

6.6 Lenguajes de Cuarta Generación.

Se puede realizar una clasificación de los lenguajes de programación tomando como criterio las generaciones en las que han surgido. De este modo, los lenguajes máquina pertenecen a la primera generación, los lenguajes ensambladores a la segunda y los lenguajes de alto nivel a la tercera. Una característica que distingue los lenguajes de la tercera generación de sus predecesores es la independencia de los primeros respecto del hardware del ordenador. Los Lenguajes de Cuarta Generación se caracterizan fundamentalmente por su orientación al problema a resolver.

Los Lenguajes de Cuarta Generación (4GLs) aparecieron como respuesta ante la insatisfacción de los usuarios dedicados a los negocios al utilizar lenguajes convencionales como el COBOL. Tales usuarios eran a menudo programadores no profesionales que deseaban obtener resultados rápidos a partir de los datos almacenados en un ordenador. La popularización de los ordenadores personales ha tenido una gran influencia en cuanto al uso de los lenguajes de cuarta generación. Un director de negocios trabajando en su despacho ante un PC desea resultados inmediatos sin tener que realizar un gran trabajo de programación para el que, además, tampoco se haya preparado.

En los lenguajes de cuarta generación una instrucción equivale a más de una de un lenguaje de alto nivel. Son herramientas potentes que permiten crear de modo automático estructuras de datos, informes o incluso generar código. De este modo, pueden construirse programas potentes con un tamaño en instrucciones muy reducido. Sin embargo, la ejecución de los programas escritos en este tipo de lenguajes resulta ser más lenta que la de los programas codificados en lenguajes de alto nivel.

A continuación se presenta una posible clasificación de los lenguajes de cuarta generación:

6.6.1 Generadores de Aplicaciones.

Estos lenguajes generan soluciones para aplicaciones rutinarias. Las operaciones típicas son la entrada de datos (y comprobación de la validez de los mismos), la consulta de ficheros o bases de datos y la actualización de los mismos.

6.6.2 Lenguajes de Consulta (Query Languages).

Los lenguajes incluidos en esta categoría trabajan sobre bases de datos, y permiten al usuario hacer preguntas relativas a sus campos y registros. Asimismo, existen lenguajes más sofisticados, los cuales permiten al usuario no sólo consultar, sino actualizar las bases de datos sobre las que trabajan.

6.6.3 Lenguajes para la Toma de Decisiones (Decision-Support Languages).

La intención de los diseñadores de los lenguajes encuadrados en esta categoría consiste en ayudar al usuario a tomar decisiones mediante el suministro de una mayor cantidad de información. Estos lenguajes proveen al usuario de utilidades para construir bases de datos y realizar cálculos estadísticos. Normalmente están orientados a los negocios, y se usan típicamente en la etapa de planificación para analizar la inversión económica a realizar.

Pueden incluir capacidades gráficas para mejorar la presentación del material, las cuales varían entre las simples, pero efectivas, hojas de VISICALC y LOTUS 1-2-3 y los lenguajes con capacidades de cálculo estadístico sofisticado para asistir al usuario en la toma de decisiones.

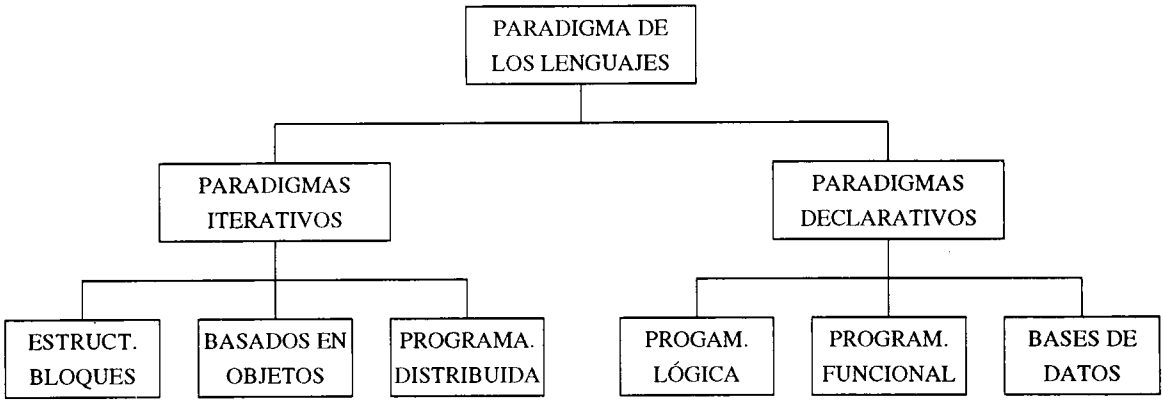


Figura 6.4: Paradigma de los lenguajes

La mayoría de ellos incluyen también el uso de ventanas y el desplazamiento de datos por la pantalla (data scrolling) como facilidades importantes.

En la práctica, los lenguajes de cuarta generación no están claramente divididos entre estas tres categorías. Mientras muchos de ellos incluyen generadores de informes y capacidades gráficas como parte del propio lenguaje, otros suministran tales capacidades en paquetes separados. Ciertamente, la habilidad de estos lenguajes para proveer un fácil acceso a los gráficos interactivos constituye un factor importante a tener en cuenta por parte del usuario dedicado a los negocios.

Los lenguajes de cuarta generación constituyen hoy en día el área más importante dentro del diseño de lenguajes de programación. Lo que no se están produciendo son lenguajes bien diseñados y adaptados a las necesidades del cliente, los cuales tengan en cuenta el hardware del ordenador actual y futuro.

6.7 Paradigmas de los Lenguajes de Programación

Una clasificación más moderna de los lenguajes de programación ha sido realizada por Peter Wegner en 1988, quien ha seleccionado unos Paradigmas como representantes de los lenguajes más usados en la actualidad por grandes grupos de programadores e investigadores. Wegner concibe los paradigmas como "patrones de pensamiento empleados para resolver problemas". En la práctica, son abstracciones creadas a partir de modelos de computación y características de diversos lenguajes de programación.

Wegner ha agrupado los paradigmas de los lenguajes dentro de dos grandes clasificaciones: Imperativos y Declarativos. Los lenguajes imperativos especifican **Cómo** se debe realizar un proceso de computación a través de secuencias de cambios en el almacenamiento de un ordenador, mientras que los lenguajes declarativos especifican **Qué** debe computarse.

6.7.1 Paradigma Imperativo.

Los Paradigmas Imperativos son aquéllos en los cuales el proceso de computación se lleva a cabo mediante **Cambios de Estado**. Un **Estado** puede definirse como la situación actual de la Memoria de Acceso Aleatorio (RAM) de un ordenador.

Cuando un programa se carga en memoria, sus datos se encuentran en una determinada situación (p.e.: una lista desordenada). Es tarea del programador especificar la secuencia de cambios sobre los datos almacenados, la cual conducirá al deseado estado final (p.e.: una lista ordenada). Junto con los datos que maneja el programa también se almacena el código del mismo, las tablas de símbolos, las pilas (stacks) utilizadas, el Sistema Operativo y sus estructuras de datos asociadas. El programa completo, los datos, e incluso la propia CPU pueden considerarse como parte del estado inicial.

6.7.1.1 El Paradigma de la Estructura de Bloques.

FORTRAN fue el primer lenguaje con Estructura de Bloques. Divide el estado en bloques, los cuales representan subrutinas y conjuntos de datos. La disposición de bloques del FORTRAN puede concebirse como un fichero plano (sin jerarquía) en el que cada bloque sigue a sus predecesores. Por esta razón, el FORTRAN no se considera ya un lenguaje con estructura de bloques.

El término **Estructura de Bloques** hace referencia a **Bloques Anidados**. Esto es, procedimientos que pueden estar anidados dentro de otros. El estado se representa mediante una pila con una referencia en su cima al bloque activo en ese momento. No se aplica restricción alguna en cuanto a la existencia de items duplicados en la pila, lo cual permite soportar la recursividad, la cual no es posible si se utiliza la estructura plana del FORTRAN. En un lenguaje con estructura de bloques, el **Procedimiento** (bloque) es el constructor principal de los programas. Ejemplos de lenguajes con estructura de bloques son ALGOL 68, PASCAL y C.

6.7.1.2 El Paradigma Basado en Objetos.

El paradigma basado en objetos engloba a los lenguajes que soportan objetos capaces de interactuar unos con otros. Un **Objeto** es un grupo de procedimientos que comparten un estado. Debido a que los datos forman parte también del estado, todos aquellos datos y procedimientos o funciones que hacen referencia al mismo estado pueden agruparse en un único objeto. Ejemplos de lenguajes basados en objetos son ADA, donde los objetos se denominan **Packages**, MODULA, donde los objetos se denominan **módulos**, y SMALLTALK, donde los objetos se denominan **Objetos**. Otro lenguaje más moderno de este paradigma es una extensión del lenguaje C denominada C++.

6.7.1.3 El Paradigma de la Programación Distribuida.

La **Programación Concurrente** se puede dividir en dos amplias categorías: **Sistemas Débilmente Acoplados y Sistemas Fuertemente Acoplados**. El término *distribuido* normalmente hace referencia a lenguajes que operan en sistemas débilmente acoplados, en los cuales un grupo de programadores trabajan simultáneamente sobre un programa particular y se comunican mediante mensajes que circulan a través de un canal de comunicaciones, como un enlace punto a punto o una red local. En un sistema distribuido débilmente acoplado, un lenguaje no necesita soportar la utilización de memoria compartida.

Un sistema fuertemente acoplado permite que más de un proceso en ejecución acceda a la misma posición de memoria. Los lenguajes asociados a estos sistemas deben sincronizar el uso de la memoria compartida, de modo que sólo un proceso puede escribir a la vez en una

variable compartida. Asimismo, un proceso debe esperar a que ciertas condiciones se cumplan antes de continuar su ejecución. La memoria compartida posee la ventaja de la rapidez.

La programación concurrente se asocia con la existencia de más de una CPU operando en paralelo, con o sin compartir datos. No es esencial para este paradigma la existencia de múltiples CPU. Lo fundamental es compartir (distribuir) el trabajo sobre un problema particular. ADA es quizás el lenguaje para programación concurrente más conocido. En él, dos o más procedimientos se ejecutan de manera independiente. El proceso de compartir los resultados se denomina *Rendez Vous*.

Recientemente, se ha trabajado sobre lenguajes que traspasan el límite entre paradigmas fuertemente y débilmente acoplados. PROLOG CONCURRENTE, LINDA y EMERALD son lenguajes que poseen características de ambos paradigmas.

6.7.2 Paradigmas Declarativos

Un **Lenguaje Declarativo** es aquél en el cual un programa especifica una relación o función. Cuando se programa en estilo declarativo, no se realizan asignaciones a las variables del programa. Es el intérprete o compilador para el lenguaje particular quien gestiona la memoria por nosotros. Estos lenguajes poseen un nivel *más alto* que el de los lenguajes imperativos. En los lenguajes declarativos el programador opera aún más lejanamente de la propia CPU.

Los tres Paradigmas Declarativos han sido tomados de las **Matemáticas:** *Lógica, Teoría de Funciones y Cálculo Relacional.*

6.7.2.1 Programación Lógica

La **Programación Lógica** se basa en un subconjunto del **Cálculo de Predicados**; las sentencias del programa se escriben como **Cláusulas de Horn**. El **Cálculo de Predicados** suministra reglas y axiomas (hechos), de modo que se pueden deducir nuevos hechos a partir de otros ya conocidos. Las cláusulas de Horn tan sólo permiten deducir un nuevo hecho a partir de una simple sentencia. Las sentencias expresadas como cláusulas de Horn hacen posible un particular método mecánico de prueba denominado **Resolución**. Una de las aplicaciones de este método es la **Demostración Automática de Teoremas**.

Un **Programa Lógico** consiste en una serie de axiomas o hechos, unas reglas o inferencias, y un teorema o pregunta a demostrar. La salida es verdadera si desde los hechos que se tienen se puede llegar a deducir el teorema y falsa en caso contrario. PROLOG es un ejemplo de lenguaje de **Programación Lógica**.

6.7.2.2 Programación Funcional

Los lenguajes puramente funcionales operan exclusivamente mediante funciones que devuelven un valor simple a partir de una lista de argumentos. No se permiten efectos laterales. De este modo, un programa es una función invocada con una serie de parámetros, cada uno de los cuales posiblemente invoque a otra función para generar un nuevo valor, que corresponderá a un parámetro efectivo. Las funciones en si mismas son **Valores de Primera Clase** que pueden pasarse a otras funciones. De este modo, la programación funcional provee al programa de la habilidad para modificarse a si mismo, es decir: **Aprender**.

En la práctica, existen muy pocos lenguajes que sean puramente funcionales, ya que los efectos laterales básicos, como la entrada y la salida son siempre deseables. LISP es el len-

guaje funcional más conocido. El LISP puro existe y ha sido desarrollado, pero las versiones comerciales de LISP incluyen muchas características no funcionales.

6.7.2.3 Lenguajes de Bases de Datos

Las propiedades que distinguen a los lenguajes designados para trabajar con bases de datos son la **Persistencia** y la **Gestión de los Cambios**. Las **Entidades** de una base de datos no desaparecen después de que un programa ha terminado, sino que perduran indefinidamente. Además, dado que una base de datos es permanente, estos lenguajes deben también soportar el cambio: Los propios datos pueden cambiar, y así las relaciones entre las entidades u objetos.

Un **Sistema de Gestión de Bases de Datos** suele incluir un **Lenguaje de Definición de Datos** (DDL) y un **Lenguaje de Manipulación de Datos** (DML) para interactuar con las bases de datos existentes.

Los lenguajes de bases de datos a menudo pueden incluirse en otros lenguajes de programación para conseguir mayor flexibilidad. También se pretende que sean fáciles de usar, de modo que los no programadores puedan utilizar estas herramientas para tratar automáticamente los datos del mundo de los negocios.

6.8 Aplicaciones de los Lenguajes de Programación

La diversidad de lenguajes existentes en la actualidad se debe a que determinados lenguajes son más apropiados que otros para desarrollar un cierto tipo de aplicaciones. Por lo tanto, a la hora de codificar una aplicación se debe elegir el lenguaje más adecuado.

Pueden distinguirse cinco tipos de aplicaciones:

- **Aplicaciones Científicas**. Predominan los algoritmos de cálculo numérico y matrices. Los lenguajes adecuados suelen ser PASCAL, FORTRAN o C.
- **Aplicaciones de Procesamiento de Datos**. Sobresalen las tareas relativas a la creación, mantenimiento, consulta y listado de datos. Estos datos se organizan en registros, ficheros y bases de datos. Pueden utilizarse COBOL, Dbase o Clipper.
- **Aplicaciones de Tratamiento de Textos**. Llevan a cabo la manipulación de textos en lenguaje natural. Son lenguajes válidos Snobol, C o PASCAL.
- **Aplicaciones de Inteligencia Artificial**. Constituidas por programas que emulan un comportamiento inteligente:
  - Juegos Inteligentes (ajedrez, damas, ...).
  - Comprensión del Lenguaje Natural.
  - Visión Artificial.
  - Robótica.
  - Sistemas Expertos.

  Los lenguajes más utilizados son LISP, PROLOG y C.
- **Aplicaciones de Programación de Sistemas**. Comprenden el desarrollo de módulos de un Sistema Operativo, traductores de lenguajes, etc. Tradicionalmente se ha utilizado el lenguaje ensamblador, pero en la actualidad se utiliza C, ADA o MODULA-2.

Capítulo 7

Metodología de la programación

Hemos visto cómo se relacionaban entre sí los distintos componentes hardware de una computadora. Pero todo este hardware resulta prácticamente inútil si no tiene software que le indique lo que tiene que hacer. Por tanto, necesitamos aprender técnicas de programación para utilizar la computadora como una herramienta para resolver problemas.

¿Qué ventajas aporta el uso del ordenador para la resolución de problemas? Un buen número de problemas conlleva cálculos complicados o manejo de grandes cantidades de datos. En el primer caso, el riesgo de equivocarse es grande, y en el segundo, el trabajo se convierte en pesado y rutinario. Mediante el uso de la computadora se eliminan estos inconvenientes debido a las capacidades de la misma, basadas en las siguientes características: **precisión**, **rapidez** y **memoria**.

Pero el ordenador por sí mismo no puede analizar un problema y dar directamente su solución. Es el programador quien debe encontrar la solución al problema y comunicárselo al ordenador. Es decir, el problema lo resuelve el programador, no el computador. Una vez resuelto, el programador deberá describir al ordenador con detalle y en una forma entendible para la máquina, todos los pasos que debe seguir para resolver el problema. Una descripción de este tipo es un *programa* y su objetivo es dirigir el funcionamiento de la máquina.

En este tema veremos en primer lugar los pasos que se siguen en la construcción de un programa y la importancia del seguimiento de los mismos para la obtención de un producto software de calidad.

El primer paso para conseguir un programa que resuelva un problema, es establecer de forma precisa la definición del mismo. Los principales aspectos de esta definición se explican en el apartado tercero.

Nos centraremos especialmente en el concepto de algoritmo y sus elementos, por la importancia de éste en el campo de la Programación. El proceso de diseño de un algoritmo se describe en el apartado cuarto.

El quinto apartado describe la transformación de un algoritmo en un programa, es decir, el proceso de codificación.

La técnica de Programación Estructurada que se debe utilizar para diseñar algoritmos correctos, fáciles de entender y de modificar en caso necesario, se estudia en el apartado sexto.

Niklaus Wirth dió la siguiente definición de algoritmo: *Un algoritmo está formado por dos partes esenciales, una descripción de las acciones que deben ser ejecutadas y una descripción de los datos manipulados por estas acciones.* Los datos pueden ser simples o estructurados. Los tipos de datos simples se ven en el apartado cuarto. Algunos conceptos básicos sobre tipos

abstractos de datos se describen en el apartado séptimo.

Una técnica que no podía faltar en un tema de Metodología de la Programación es la recursividad. Es importante por su potencia y porque muchos problemas presentan una solución inmediata si se adopta una perspectiva recursiva. Se describe en el apartado octavo.

Por último, en el apartado noveno, se destacan aspectos del diseño de un algoritmo que nos ayudan a conseguir algoritmos con un estilo correcto.

7.1 Ciclo de vida del software.

En general, en el proceso de creación de cualquier software se siguen las siguientes fases:

1. **Fase de definición:** Se estudia qué es lo que se va a realizar (por ejemplo, qué tipo de información va a manejar el software y qué tipo de operaciones va a realizar sobre esa información).

2. **Fase de desarrollo:** Se crean los programas y la documentación asociada a los mismos.

3. **Fase de mantenimiento:** En ella se hacen las mejoras y correcciones del software desarrollado, durante su tiempo de vida útil.

Podemos detallar más aún cada una de las fases anteriores descomponiéndolas en seis fases que constituyen el denominado **ciclo de vida clásico**:

- **Análisis del sistema**: El software que vamos a desarrollar suele ser parte de un sistema mayor formado por:

  - hardware,
  - software,
  - bases de datos, y
  - personas

  Hemos de estudiar las tareas realizadas por cada uno de esos elementos del sistema y cuáles de estas funciones se pueden automatizar. Por ejemplo, si deseamos realizar una aplicación para gestionar una empresa, hemos de ver qué funciones realiza la empresa, cuáles de éstas se hacen manualmente (las llevan a cabo personas) y cuáles están automatizadas (se llevan a cabo mediante ordenador). Se intentará automatizar el mayor número de funciones posible.

- **Análisis de los requisitos software:** Antes de comenzar a diseñar el software hemos de especificar ciertos aspectos del mismo:

  - funciones que debe realizar,
  - interacción con el resto de los elementos del sistema,
  - rendimiento,
  - fiabilidad, etc.

En cada una de las dos fases mencionadas se genera un conjunto de documentos que nos permiten fijar la estructura funcional de la organización y todos los requeremientos que se exigirán al software que se va a desarrollar. Los documentos, tanto del análisis del sistema como del software, se revisan con el cliente hasta que realmente reflejen la realidad de la organización y sus necesidades.

- **Diseño:** Debemos obtener un producto software que cumpla con los requisitos establecidos en las fases anteriores. Para ello, antes de codificar debemos diseñar -en esta fase- la estructura adecuada para el mismo.

- **Codificación:** Plasmamos el diseño de la fase anterior en programas escritos en el lenguaje de programación adecuado, dependiendo del tipo de aplicación. Si el diseño se ha hecho de forma detallada, la codificación puede realizarse mecánicamente.

- **Prueba:** Consiste en comprobar si hemos construido el software que se deseaba, es decir, comprobar que los programas:

  - se corresponden con el diseño,
  - realizan correctamente sus funciones, y
  - satisfacen los requisitos indicados

- **Mantenimiento:** El software sufrirá cambios después de que se entregue al cliente debidos fundamentalmente a:

  - la existencia de errores,
  - la necesidad de adaptar el software (nuevo Sistema Operativo, nueva máquina específica, etc.),
  - nuevas necesidades del cliente que requiere aumentos funcionales o de rendimiento.

En cada una de las fases comprendidas entre el diseño y el mantenimiento, se generan documentos que justifican y describen la forma de llevarlas a cabo y facilitan el uso del programa por parte del usuario. Al conjunto de documentos que llevan a cabo este último aspecto se le denomina **manual de usuario**.

Teniendo en cuenta el tipo de aplicaciones que se desarrollarán en este curso de introducción, la aplicación del ciclo descrito supondría un esfuerzo no justificado por la complejidad de los problemas que se abordarán. Es por esto por lo que diseñaremos programas siguiendo, al menos, los siguientes pasos:

1. Fase de definición del problema.

2. Fase de desarrollo.

   (a) Diseño de una solución general del problema (algoritmo).

   (b) Transformación de esta solución general (algoritmo) en una solución específica (programa).

Para resolver un problema con la ayuda de una computadora, es necesario un análisis completo del mismo que nos permita desarrollar un **algoritmo** (procedimiento paso a paso para obtener la solución de un problema). Una vez conseguido éste, se debe transformar a un lenguaje de programación concreto, convirtiéndose en un programa.

Generalmente estos pasos se incrementan para incluir:

- una prueba del algoritmo para detectar errores en el mismo y corregirlos antes de avanzar en el proceso,
- una prueba y verificación del programa que permita ver que el programa realmente hace la tarea adecuada de la forma correcta, y
- una etapa de mantenimiento en la que se realizarán mejoras en el programa y/o se corregirán posibles errores detectados en el uso del mismo.

7.2 Definición del problema.

Para definir con precisión un problema es necesario identificar los elementos de información del problema que sean útiles para obtener la solución, es decir, los datos de entrada. De igual manera se deben especificar los datos de salida. En definitiva, se debe responder a estas dos preguntas:

- ¿ Qué datos se necesitan para resolver el problema?
- ¿Qué información debe proporcionar la resolución del problema?

Es conveniente en este punto, intentar ver la relación existente entre las entradas y las salidas.

Ejemplo 1: Queremos realizar *un algoritmo que dado el radio de una circunferencia, calcule su longitud así como el área del círculo que determina.* Para este problema la entrada es el radio de circunferencia, el único dato necesario para realizar los cálculos. Respecto a las salidas, para este problema son: La longitud de la circunferencia y el área del círculo. La relación entre datos de entrada y de salida viene dada por las siguientes fórmulas:

longitud = 2 * 3.1416 * radio
area = 3.1416 * radio * radio

Ejemplo 2: Queremos realizar *un programa que calcule la desviación típica de un conjunto de datos y la muestre por pantalla.*

ENTRADAS:

Sabiendo que la formúla de la desviación es:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(\overline{X} - X_i)^2}{N-1}}$$

donde $\overline{X}$ es la media aritmética del conjunto de datos que constituye la muestra, $\mathbf{X}_i$ representa el elemento i de la muestra y **N** es el n de datos de la muestra, es evidente que para calcular la desviación típica necesitamos como información de entrada el conjunto de datos

que constituye la muestra. También necesitamos el número de elementos (**N**), pero éste es un dato que se puede obtener contando el número de datos que se dá como entrada. Por tanto, es una información que podemos tomar como información de entrada o bien calcular dentro del programa.

SALIDAS:

El resultado que debe mostrar el programa es el valor de la desviación típica.

A simple vista parece que el primer problema a que nos enfrentamos es la media aritmética ya que es un dato que no conocemos, necesario para el cálculo de la desviación. Para calcularla se usará la fórmula:

$$\overline{X} = \frac{\sum_{i=1}^{N} x_i}{N}$$

Una vez calculada ésta, el cálculo de la desviación típica se limita al seguimiento de la primera fórmula.

Es importante definir bien el problema para obtener una correcta solución del mismo.

7.3 Diseño del algoritmo.

7.3.1 Concepto de algoritmo. Características.

La noción de algoritmo es básica en programación de ordenadores, ya que para realizar un programa es necesario el diseño previo de un algoritmo. Por tanto, debemos empezar con un cuidadoso análisis de este concepto.

El término proviene del matemático persa Mohammed Al-khowârizmî que alcanzó gran reputación por el enunciado de reglas paso a paso para sumar, restar, multiplicar y dividir números decimales. Euclides, el matemático griego que inventó un método para encontrar el máximo común divisor de dos números, se considera el otro gran padre de los algoritmos.

Un **algoritmo** es un método para resolver un problema. Más explícitamente, es un conjunto finito de reglas que dan una secuencia de operaciones para resolver un tipo específico de problema. Debe cumplir las siguientes condiciones:

1. Ser finito, es decir, acabar siempre tras un número finito de pasos. Si el algoritmo nunca acaba, no obtendremos ninguna solución y, como se ha señalado, el objetivo principal de un algoritmo es obtener la solución de un problema.

2. Estar definido. Para ello debe estar compuesto por un conjunto ordenado de acciones, especificadas en cada caso rigurosamente y sin ambigüedad. De esta forma se cumplirá que si se sigue el algoritmo dos veces con los mismos datos de entrada, se obtendrán los mismos datos de salida.

¿Cuál es la diferencia entre programa y algoritmo? Para que una computadora resuelva un problema según un algoritmo éste tiene que ser convertido en un programa traduciéndolo a algún lenguaje de programación concreto. Los algoritmos no son directamente interpretables por la computadora, pero tienen la ventaja de que son independientes de cualquier lenguaje de programación.

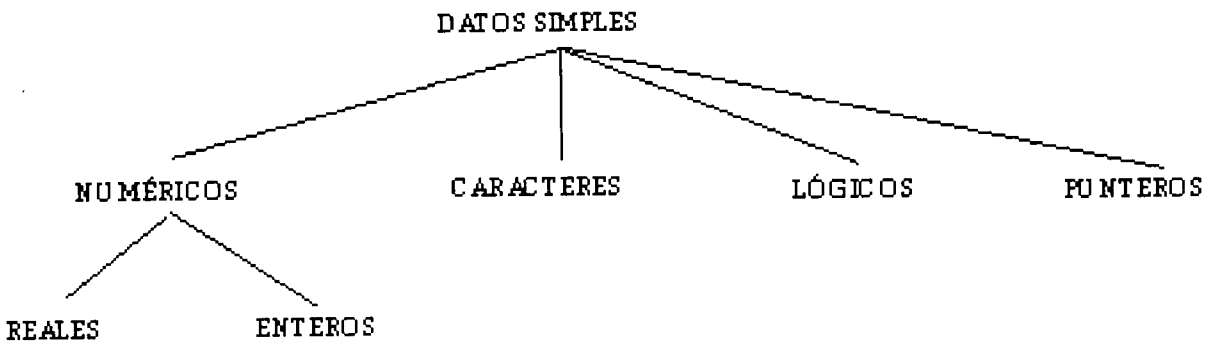


Figura 7.1: Tipos básicos de datos que pueden aparecer en un algoritmo

7.3.2 Elementos de un algoritmo.

En este punto del tema describiremos los elementos básicos que componen un algoritmo.

7.3.2.1 Datos, tipos de datos y operaciones primitivas.

El primer objetivo de todo ordenador es el manejo de información. A la información con que trabaja una computadora se le denomina **datos**. Es obvio que el manejo de información es importante en el trabajo de un ordenador y por tanto lo es también en un algoritmo (que especifica un modo de trabajo en el ordenador).

Los datos que se utilizan en un algoritmo, atendiendo a las propiedades que poseen y las operaciones que se pueden realizar con ellos, se clasifican en distintos tipos, de forma que cada dato pertenece a un tipo de datos concreto.

Podemos clasificar los tipos de datos en dos grandes grupos:

- Simples, no estructurados (que son los que vamos a estudiar en este apartado).
- Compuestos o estructurados, que están formados por agrupaciones de datos simples o no, relacionados entre sí (los estudiaremos en el apartado ocho).

Los tipos básicos de datos[1] que se manejan generalmente en un algoritmo son los siguientes (ver figura 7.1).

- **Numérico:** Un dato de tipo numérico se puede representar de dos formas distintas: Como real o como entero. Los enteros son números positivos o negativos, sin decimales. Los reales siempre tienen un punto decimal y pueden ser positivos o negativos. Un número real consta de un entero y una parte decimal.

[1]El conjunto de tipos básicos de datos no es igual en todos los lenguajes de programación, por lo que, según el lenguaje que usemos, un programa tendrá unos tipos básicos de datos u otros. Como se mencionó, los algoritmos son independientes del lenguaje de programación por lo que, de forma general, trabajaremos con estos tipos de datos como básicos, aunque -en algunos casos- tengamos que hacer modificaciones para codificar el algoritmo en el lenguaje de programación correspondiente.

- **Carácter:** Un dato de tipo carácter contiene un único carácter de los que la computadora reconoce: caracteres alfabéticos (a,..., A,..., Z), numéricos (0,...,9), especiales (@, #, $, -, %, etc.).

- **Lógico:** Es aquel que sólo puede tomar uno de los dos valores siguientes: *verdadero* o *falso*.

- **Puntero:** Un dato de tipo puntero es aquel cuyo valor es la dirección de memoria de otro dato. Como veremos en el apartado siete son útiles para construir estructuras de datos, ya que proporcionan los lazos de unión entre los elementos que constituyen dichas estructuras.

¿Qué operaciones podemos realizar con estos datos?

Con un dato de cualquier tipo podemos realizar operaciones primitivas o no.

Una **operación primitiva** es aquella que se realiza directamente en un lenguaje de programación concreto, es decir, que no tenemos que indicar cómo realizarla. Dentro del grupo de operaciones **no primitivas** se incluyen todas aquellas no proporcionadas por el lenguaje de programación que, previamente a su uso, tendremos que indicar cómo realizar a partir de las operaciones primitivas proporcionadas [2]

Para los **datos numéricos** se pueden considerar las siguientes operaciones básicas:

- suma (+),
- resta (-),
- multiplicación (*),
- división (/),
- exponenciación (^ó ↑). Así, por ejemplo, 2^3 sería 2^3),
- módulo, que proporciona el resto de una división entera (MOD). Así, por ejemplo: 15 MOD 2 = 1 y,
- la división entera, que devuelve el cociente de una división entera (DIV). Por ejemplo: 15 DIV 2 = 7.

Es importante destacar que el resultado de toda operación tiene el mismo tipo que los operandos implicados. Por ejemplo, si se suman dos números reales, se obtiene un número real; si se multiplican dos números enteros, el resultado es un número entero. Si se dividen dos números reales, el resultado también es un número real. Pero si se dividen dos números enteros, el resultado se redondea (o se trunca) para convertirlo a un número entero.

Ejemplo: 1/10*10=0; ya que 1/10 = 0 y 0*10=0.

[2]De nuevo, el conjunto de operaciones primitivas para cada tipo de datos no es igual en todos los lenguajes de programación. Para preservar la independencia del algoritmo respecto al lenguaje de programación en que posteriormente se codificará, en este apartado se van a describir una serie de opeaciones básicas que, aunque no estén disponibles en todos los lenguajes, sí podrán realizarse fácilmente con las ya existentes.

Operador A	Operador B	A Y B	A O B	No B
Verdadero	Verdadero	Verdadero	Verdadero	Falso
Verdadero	Falso	Falso	Verdadero	Verdadero
Falso	Verdadero	Falso	Verdadero	--
Falso	Falso	Falso	Falso	--

Figura 7.2: Los 3 operadores lógicos básicos.

Símbolo		Significado
$<$	$\Rightarrow$	menor que
$>$	$\Rightarrow$	mayor que
$=$	$\Rightarrow$	igual que
$\leq$	$\Rightarrow$	menor o igual que
$\geq$	$\Rightarrow$	mayor o igual que
$\neq$	$\Rightarrow$	distinto de

Tabla 7.1: Los operadores relacionales.

Afortunadamente, siempre es posible mezclar dos tipos diferentes de datos en las operaciones numéricas. Esto quiere decir que un operando puede ser real y el otro entero. En estos casos el resultado se expresa siempre como un dato del tipo que incluye a los demás (en el ejemplo anterior como un dato real).

Ejemplo: 1.0/10 * 10 = 1.0.

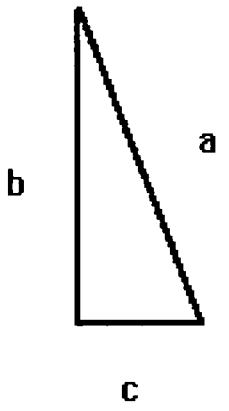
El **tipo** de dato **carácter** es un tipo muy limitado en cuanto a operaciones. De hecho, la operación que generalmente se realiza con los caracteres es la comparación de igualdad. (También se hacen comparaciones de orden con los operadores relacionales que se verán a continuación).

Las operaciones primitivas para datos de **tipo lógico** o booleano vienen determinadas por la *tabla de verdad* 7.2:

Otro tipo de operadores son los denominados *relacionales*, que describen la relación entre dos valores numéricos o dos caracteres. Los podemos ver en la tabla 7.1:

La razón por la cual se pueden comparar caracteres es la existencia de juegos de caracteres (como por ejemplo, los códigos ASCII y EBCDIC), que los representan en memoria como un número binario. Es por esto que los caracteres se pueden ver como números y, por tanto, se establece una relación de orden entre ellos. Así, el carácter 'A' tiene el valor 65 en el código ASCII y la 'H', el 72, por lo que 'A' < 'H'. En general, se verifica que:

'A'<'B'<'C'<...<'Y'<'Z'<'a'<'b'<...<'y'<'z'
'0'<'1'<...<'8'<'9'



$$a^2 = b^2 + c^2$$

Figura 7.3:

y que el número asociado con cualquier letra es menor que el asociado con cualquier dígito. Por último, sobre los datos de **tipo puntero** se realizan las siguientes operaciones:

- asignar una dirección a un dato de tipo puntero,
- *liberar* un puntero, es decir hacer, que no *apunte* a ninguna dirección.
- comparar las direcciones que representan dos datos de tipo puntero,
- ver el valor que se almacena en la dirección que representa,
- comparar los datos que se almacenan en las direcciones que representan dos datos de tipo puntero, etc.

El operador que muestra la información contenida en la dirección a que apunta un puntero, se denomina en notación algorítmica **INFO**. Es decir INFO(p) nos devolverá el dato contenido en la dirección que contiene **p**.

7.3.2.2 Variables, constantes y expresiones.

Una **variable** es un objeto de datos que posee un valor y que es conocido en un programa o en un algoritmo por un nombre (identificador de la variable). Desde otra perspectiva se puede considerar una variable como una zona de memoria identificada por un nombre.

El valor que contiene una variable puede ser modificado en el seguimiento de un algoritmo (al igual que en la ejecución de un programa).

En Matemáticas, el concepto es muy conocido. Por ejemplo, si se denotan los lados de un triángulo rectángulo mediante a, b y c, el teorema de Pitágoras proporciona la siguiente relación válida para los tres lados:

Ésta es una relación que se expresa de forma general utilizando variables y que puede aplicarse a cálculos específicos. Por ejemplo si a=5, y b=4, la fórmula determina que c=3. En este caso, 5, 4 y 3 son los valores específicos que tienen en un determinado momento las variables a, b y c respectivamente. También en un algoritmo, o en un programa de ordenador, el uso de las variables permite la especificación de una fórmula general de cálculo, y al igual que

en las fórmulas matemáticas, las variables de los algoritmos o de los programas, tienen nombres y toman diversos valores, pero en un instante concreto sólo tienen un valor determinado.

Existen algunas reglas simples en la denominación de las variables, aunque generalmente dependen del lenguaje de programación que se esté utilizando. En nuestro caso, para tener una regla general, se establecerá la restricción de que la variable comience con una letra y seguidamente vayan letras, dígitos numéricos y el carácter especial '_', en cualquier orden. No se permitirán espacios en blanco en el nombre de una variable.

Ejemplos:

- Identificadores de variables correctos:
  - hola
  - Total
  - Dia_Del_mes
  - W
  - r4
- Identificadores incorrectos:
  - 3j
  - Dos Palabras
  - x+y

En el proceso de diseño y construcción de un programa es preferible utilizar nombres descriptivos, para aumentar la legibilidad de los programas resultantes. En el caso del teorema de Pitágoras, podríamos cambiar el nombre a las variables, pasando a tener:

$$Hipotenusa^2 = Cateto1^2 + Cateto2^2$$

Otros elementos que podemos utilizar en un algoritmo son las **constantes**, las cuales contienen valores que no deben cambiar a lo largo del desarrollo del algoritmo. A la acción por la que las constantes toman el primer y único valor se denomina inicialización de la constante. Las reglas que rigen la forma de crear los identificadores y asignar valores a las constantes son las mismas que para las variables.

Un concepto relacionado con las constantes son los *valores constantes*: valores que aparecen explícitamente en un algoritmo y que no tienen un identificador asociado, por tanto, no pueden ser referenciadas más que por su propio valor. Así, -4.5667 es un valor constante real, 'Burgos' un valor constante de tipo cadena de caracteres, y Verdadero es un valor constante de tipo lógico.

Todas las variables, valores constantes y constantes utilizadas en un algoritmo son de un cierto tipo (entero, real, cadena, lógico, etc.). Una variable entera podrá tomar sólo valores enteros; una variable real podrá tomar sólo valores reales, y una variable lógica tomará también sólo los valores *verdadero* o *falso*. Por eso es conveniente que, al principio del algoritmo, se indique cada variable utilizada y su tipo.

	Orden	Operador	Significado
Mayor precedencia	1.	^	Potenciación
	2.	+, -	Signo
	3.	*, /	Producto y División
	4.	+, -	Suma y Resta
Menor precedencia	5.	DIV, MOD	División entera y Módulo

Tabla 7.2: Orden de precedencia de los operadores numéricos.

Una **expresión** es una combinación de variables, constantes, valores constantes, operadores, paréntesis y nombres de funciones especiales (raiz cuadrada, valor absoluto, etc.). Toda expresión tiene en todo momento un valor concreto que es el resultado de evaluarla de izquierda a derecha. Es decir, es el resultado de tomar el valor de las variables que intervienen en ella y realizar las operaciones que aparecen.

Ejemplo: Algunas expresiones numéricas:

$0.5 * 3/10$ $0.5 * a/20 + b$ c=a+b

El **orden de precedencia**[3] **de los operadores numéricos** se puede observar en la tabla 7.2.

Cuando se desee forzar la evaluación de una expresión en un determinado orden, independientemente de la precedencia de los operadores, se utilizarán los paréntesis. Así, en una expresión que contiene paréntesis, el orden a seguir para evaluarla es:

1. Las operaciones encerradas entre paréntesis se evalúan primero. Si existen diferentes paréntesis anidados, las expresiones internas se evalúan antes.
2. Las operaciones artiméticas dentro de una expresión se evalúan según el orden de prioridad expresado en el apartado anterior. Si coinciden varios operadores de igual prioridad, el orden a seguir es de izquierda a derecha.

Ejemplos: Hallar el valor de cada una de las siguientes expresiones e indicar el tipo de las mismas.

- 2*3+5/3⇒6+5/3⇒6+1⇒7
  Tipo: Entero
- (4.0+5)/(2+6)*4⇒9.0/(2+6)*4⇒9.0/8*4⇒1.125*4⇒4,5
  Tipo: Real
- A^3*4+B^((5+C)*2) con A=2; B=3; C=1
  ⇓
  A^3*4+B^(6*2)⇒A^3*4+B^12⇒8*4+B^12⇒32+B ^12⇒32+531441⇒531473
  Tipo: Real

[3]Tanto el orden de precedencia, como el de evaluación, puede variar en función del lenguaje de programación.

Mención aparte merecen, por su importancia, las expresiones lógicas que son las que ofrecen como resultado después de su evaluación un valor verdadero o falso.

Ejemplo: Son expresiones booleanas:

$$a \text{ O } b,\ a \text{ O (No } b \text{ Y } c),\ \text{No } (a \text{ O } b),$$

donde a, b, c son variables o expresiones boolenas. Dependiendo de los valores de estas variables, las expresiones de los ejemplos se podrán evaluar como verdaderas o falsas.

Se establece también un orden de precedencia con los operadores lógicos: El de mayor prioridad es No y el de menor prioridad es O, y el operador Y se encuentra entre los dos.

Ejemplo: Determinar el valor de las siguientes expresiones:

- $(3 < 6)$ Y $(5 < 4) \Rightarrow$Verdadero Y Falso$\Rightarrow$Falso
- (A=B) O (NO C) donde A=3,5; B=3,5; C=Falso

  $\Downarrow$

  Verdadero O (No C)$\Rightarrow$Verdadero O Verdadero$\Rightarrow$Verdadero

Ahora que sabemos lo que son las variables, vamos a ver la forma en que se les dá valores. Existen dos formas de hacerlo: una es leer un valor para la variable mediante una operación de entrada y la otra es darle un valor mediante una operación de asignación.

7.3.2.3 Operación de asignación.

La operación de asignación nos permite especificar que una variable tiene un valor dado; simplemente se le asigna dicho valor a dicha variable.

Ejemplo: Guardar en A el valor 3.

En este caso, estamos realizando una *asignación constante*, es decir, estamos diciendo que, a partir de ahora, A tiene el valor constante 3.

También podemos asignar a una variable una expresión, o lo que es lo mismo, el resultado de evaluar una expresión. En este caso, al realizar una asignación, estamos haciendo en realidad dos operaciones:

1. Evaluar la expresión.
2. Almacenar el resultado de la evaluación en la variable.

Ejemplo: Guardar en A , 3*B + 5

La operación de asignación es destructiva porque cualquier valor que tuviese la variable antes de la asignación se pierde y se reemplaza por el nuevo. Así, una secuencia de asignaciones como sigue:

guardar en A, el valor 1,
almacenar en B el valor 5,
guardar en A, 2*B,
guardar en A, A+1.
tiene como resultado A=11 y B=5.

La asignación implica siempre una transferencia o movimiento de datos en memoria.

7.3.2.4 Operación de entrada. Operación de salida.

Las operaciones de entrada / salida se utilizan para intercambiar información con un medio externo.

En una **operación de entrada** (o de *lectura*) se le asigna a una variable un valor dado desde el exterior (desde cualquier periférico de entrada).

Ejemplo: Si RESPUESTA es una variable de tipo carácter y queremos que almacene el carácter que se pulse en el teclado, una forma de indicarlo es:

Leer la tecla pulsada en el teclado y almacenarla en la variable RESPUESTA.

En una **operación de salida** (o de *escritura*) se transfiere el valor de una variable a un dispositivo de salida.

Ejemplo: Si quisieramos imprimir el contenido de la variable RESPUESTA del ejemplo anterior, realizaríamos así la operación de salida:

Imprimir el valor almacenado en la variable RESPUESTA.

7.3.2.5 Estructuras de control.

7.3.2.5.1 Secuencial. Dentro de esta descripción de los elementos básicos de un algoritmo, hemos visto datos, tipos de datos, operadores, variables, expresiones, las formas de asignar valores a las variables (con la operación de asignación y la de entrada) y la operación de salida. Con estos elementos podemos realizar operaciones básicas, que -si no se indica lo contrario- se realizarán de forma secuencial, es decir, una detrás de otra en el orden en que están escritas. A esto es a lo que se denomina **estructura de control secuencial**. Dicho de otra forma, una estructura secuencial es aquella en la que una acción sigue a otra sin romper la secuencia, sin saltos.

Ejemplo: En esta secuencia

A
⇓
B
⇓
C

la acción B se ejecuta después de la A y ninguna acción puede ejecutarse entre ellas.

7.3.2.5.2 Condicional. En determinados problemas necesitamos, en función de una condición, tomar una alternativa u otra. Para ello utilizaremos la estructura condicional.

En una estructura condicional se evalúa una expresión lógica y, en función del resultado de la misma, se realiza una opción u otra. Con ella podemos bifurcar en dos caminos el flujo de acciones.

Ejemplos:

- Si se cumple que $x \geq 0$, guardar en RAIZ, $x^{0.5}$; si no se cumple, almacenar en RAIZ el valor 0.
- *Algoritmo que lea dos valores enteros desde el teclado y escriba en pantalla el mayor.*

  Del enunciado del problema se deduce que, como información de entrada se tienen que recibir los dos números que se quieren comparar, obteniendo como salida del algoritmo aquel que sea el mayor.

Necesitamos dos variables enteras (PRIMERO y SEGUNDO) en las que almacenaremos los valores que leamos como entrada.

El algoritmo a seguir es:

Algoritmo MAYOR_DOS

Entrada: Dos números enteros.

Salida: El mayor de los dos.

Variables: PRIMERO, SEGUNDO: Entero

inicio

1.- Leer el primer valor dado por el teclado y almacenarlo en la variable PRIMERO.
2.- Leer el segundo valor dado por el teclado y almacenarlo en la variable SEGUNDO.
3.- **Si** el valor que contiene la variable PRIMERO es menor o igual que el contenido en la variable SEGUNDO, escribir en pantalla el valor de la variable SEGUNDO; **caso contrario**, escribir en pantalla el valor de la variable PRIMERO.

fin

7.3.2.5.3 Repetitiva. Una estructura repetitiva o ciclo es una estructura de control que indica la repetición de un conjunto de acciones cero o más veces. Estas acciones del ciclo se pueden repetir un número fijo de veces o bien el número de veces que imponga una determinada condición.

Ejemplos:

- *Diseñar un algoritmo que lea desde el teclado cincuenta números reales y escriba en pantalla la suma de todos ellos.*

  Algoritmo SUMA_50_NÚMEROS

  Entrada: Cincuenta números reales.

  Salida: Su suma.

  Variables: NUMERO y SUMA de tipo real.

  inicio

  1.- Asignar a SUMA el valor 0.
  2.- **Repetir** cincuenta veces las dos siguientes acciones:
    2.1- Leer desde el teclado un número real y almacenarlo en la variable NUMERO.
    2.2.- Almacenar en la variable SUMA el valor de la expresión SUMA+NUMERO
  3.- Escribir en pantalla el valor almacenado en la variable SUMA.

  fin

- *Diseñar un algoritmo que muestre por pantalla los números múltiplos de tres que existen entre el 1 y el 100.*

  Algoritmo MULTIPLOS_DE_TRES

Entrada: Ninguna.

Salida: Los múltiplos de tres menores de 100.

Variables: I de tipo entero.

inicio

1.- Asignar a I el valor 3.
2.- **Mientras** se cumpla que I es menor o igual que 100 repetir las dos siguientes acciones:
  2.1- Sacar por pantalla el valor que almacena en I.
  2.2.- Guardar en I el valor de la expresión I+3.

fin

7.3.3 Métodos de representación de algoritmos.

Podemos expresar un algoritmo, básicamente con dos tipos de métodos:

- **Método informal**, como por ejemplo, el lenguaje natural. De esta forma, describiremos un algoritmo como si contáramos a otra persona los pasos que ha de seguir para resolver un problema dado. Esta es la forma de descripción que se ha utilizado para explicar los elementos básicos de un algoritmo. Se ha hecho así para mostrar la ventaja y desventaja más importantes del uso del lenguaje natural para la descripción de los algoritmos: por una parte, el lenguaje natural, es fácilmente comprensible para todos, es la forma más intuitiva de describir un método para resolver un problema. Por otra, las acciones expresadas a través de él pueden no tener un significado preciso y por tanto el algoritmo correspondiente puede no estar definido, por lo que no es una forma de descripción aconsejable. Es preferible usar cualquiera de los métodos formales que se señalan a continuación.

- **Métodos formales**, entre los que tenemos:

  - Un lenguaje específico de descripción de algoritmos o **seudocódigo**. Su uso hace que el paso de un algoritmo a un programa sea relativamente fácil. Podríamos decir que se trata de un lenguaje natural limitado y sin ambigüedad. Por su limitación se consigue expresar el conjunto de pasos que resuelven un problema, en función de las estructuras de control básicas. Su no ambigüedad hace que el método sea preciso.
  - **Diagramas**, que pueden ser de dos tipos:
    * **Diagramas de flujo** u **organigramas.** Es un método de representación, que utiliza un conjunto de símbolos de forma que, cada paso del algoritmo se visualiza dentro del símbolo adecuado y el orden en que se realizan los pasos se representa mediante líneas de flujo que indican el flujo lógico del algoritmo.
    * **Diagramas de Nassi-Schneiderman (N-S)**. Con esta herramienta, los pasos sucesivos se escriben en cajas sucesivas -con distintas formas según la estructura de control que representen-. Se puede establecer una analogía entre un diagrama N-S y un diagrama de flujo en el que se omiten las líneas de flujo.

7.3.3.1 Seudocódigo.

Diremos que una notación es un seudocódigo si con ella podemos describir un algoritmo utilizando palabras y frases del lenguaje natural sujetas a determinadas reglas. No existe un seudocódigo totalmente estándar, ya que depende de quién lo utilice, pero la notación más estándar es la que se describe a continuación.

Las acciones simples y las sentencias de control se expresan en seudocódigo de la siguiente forma:

1. **Asignación:** Se pondrá en primer lugar el nombre de la variable a la que se va a asignar valor y después de una flecha apuntando hacia esta variable, el valor constante, la constante, variable o expresión que se le asigna.

   Ejemplos:

   - Guardar en A, el valor 23.
     `A←23`
   - Guardar en A, el valor de la expresión $B*(3+C)^2$.
     `A←B*(3+C)^2`.

2. **Entrada/Salida.**

   - Para cualquier **operación de entrada**, e independientemente del medio externo del que provenga la misma, se seguirá la siguiente notación: Se pondrá primero la palabra LEER y a continuación, el nombre de la variable a la que se quiere asignar el valor dado por el medio externo. Si son varias variables, se pondrá -después de la palabra LEER- los nombres de las variables separadas por comas.

     Ejemplo: Leer la tecla pulsada en el teclado y almacenarla en la variable Respuesta.

     ```
     LEER Respuesta
     ```

   - De igual forma, una **operación de salida** se expresará igual, independientemente del medio en el cual se escriba. Para ello se pondrá la palabra ESCRIBIR y a continuación, el valor constante, la constante, variable o expresión que se quiera dar como resultado.

     Ejemplos:

     - Escribir en pantalla el valor almacenado en la variable Respuesta.
       `ESCRIBIR Respuesta`

3. **Secuencia.** La estructura de control secuencial se expresa en seudocódigo, poniendo cada una de las acciones a realizar en una línea y una a continuación de otra.

   Ejemplo: Algoritmo que lee la base y altura de un triángulo y da como resultado su superficie.

```
Algoritmo SUPERIFICE_TRIANGULO
Entrada: La base y altura del triangulo.
Salida: Su superficie.
Variables: base, altura, superficie: Reales.
```

```
inicio
     LEER base
     LEER altura
     superficie ← base * altura/2
     ESCRIBIR superficie
fin
```

4. **Condicional.** La estructura condicional nos permite realizar un conjunto de acciones u otro en función de una determinada condición que se describe mediante una expresión lógica.

   El formato general de una sentencia condicional es el siguiente:

   SI *condición*
   ENTONCES *conjunto de acciones primero*
   SI NO *conjunto de acciones segundo*

   En determinadas circunstancias no aparecerá la parte **si no** por no existir acciones a realizar en el caso de que la condición sea falsa. En esta situación, la sentencia condicional quedrá como sigue:

   SI *condición*
   ENTONCES *conjunto de acciones*

   Ejemplo: *Algoritmo que lea dos valores y visualice por pantalla el mayor:*

```
Algoritmo MAYOR2
Entrada: valor1, valor 2: entero.
Salida: El mayor de los dos.
Variables: valor1,valor2: entero.

inicio
  LEER valor1
  LEER valor2
  SI valor1 ≤ valor2
  ENTONCES ESCRIBIR valor1
  SI NO ESCRIBIR valor2
fin
```

   En algunas ocasiones puede interesar que en una (o en las dos) de las alternativas de la estructura SI_ENTONCES_SINO se incluya otra estructura condicional. A esto se le denomina estructura condicional anidada.

   Ejemplo: *Algoritmo que lea tres números y visualice por pantalla el mayor:*

```
Algoritmo MAYOR3
Entradas: valor1, valor2, valor3: entero.
Salidas: El mayor de los tres.
Variables: valor1, valor2, valor3: entero.

inicio
  LEER valor1
```

```
  LEER valor2
  LEER valor3
  SI  valor1 ≤ valor2
  ENTONCES SI valor2 ≤ valor3
           ENTONCES ESCRIBIR valor3
           SI NO ESCRIBIR valor2
  SI NO SI valor1 ≤ valor3
        ENTONCES ESCRIBIR valor3
        SI NO ESCRIBIR valor1
fin
```

5. **Repetitiva.** Existen dos tipos de sentencias repetitivas: las que realizan un número fijo de iteraciones y las que repiten las acciones del bucle o ciclo un número de veces determinado por una condición.

   En el primero de estos tipos existe siempre una **variable de control del ciclo** que controla el número de veces que se repiten las acciones y a la que se le asignan automáticamente valores sucesivos durante la ejecución del ciclo. Se debe especificar de ella:

   - el valor inicial,
   - el valor final, y
   - el incremento o decremento.

   Ejemplo: *Algoritmo que escriba los números pares menores que 50.*

```
Algoritmo PARES
Entrada: Ninguna.
Salida: Los n\'umeros pares menores que 50.
Variables: i :Entero.

inicio
  PARA i=2 HASTA 50 CON INCREMENTO 2|
       ESCRIBIR i
  fin_para
fin
```

   En un bucle condicional hay una o varias acciones que se han de repetir y una condición que determina el número de repeticiones de las mismas.

   Dentro de las condicionales, dependiendo del momento de evaluación de la condición, se tienen dos tipos de sentencias:

```
MIENTRAS condición  HACER
     secuencia sentencias
fin_mientras
```

```
REPETIR
     secuencia sentencias
HASTA  condición
```

En el primer caso, el conjunto de sentencias se repetirá mientras la condición sea verdadera. En el momento en que sea falsa, continuará con la acción que exista a continuación de la sentencia repetitiva. En el segundo caso se repetirán las acciones hasta que la condición sea verdadera. En el momento en que sea verdadera continuará con la sentencia que existe a continuación de la sentencia repetitiva REPETIR...HASTA.

La diferencia fundamental entre una y otra estructura es que en la primera, por comprobarse la condición al principio, puede darse el caso de que las acciones del bucle no se ejecuten nunca. En la segunda al comprobarse al final, siempre se ejecutarán las acciones al menos una vez.

Ejemplos:

- *Algoritmo que escriba en pantalla los números pares menores de 50.*

```
Algoritmo PARES2
Entrada: Ninguna.
Salida: Los n\'umeros pares menores de 50
Variables : i: Entero.

inicio
  i ← 2
  MIENTRAS i≤50
       ESCRIBIR i
       i ←i+2
  fin_mientras
fin
```

- *El mismo algoritmo con un ciclo condicional del segundo tipo.*

```
Algoritmo PARES3
Entrada: Ninguna.
Salida: Los n\'umeros pares menores de 50.
Variables: i: Entero.

inicio
  i←2
  REPETIR
       ESCRIBIR i
       i←i+2
  HASTA i>50
fin
```

7.3.3.2 Diagramas de flujo u organigramas.

El diagrama de flujo es una herramienta gráfica de descripción de algoritmos muy simple. Utiliza un conjunto de símbolos para representar distintas actividades y procesos en el algoritmo.

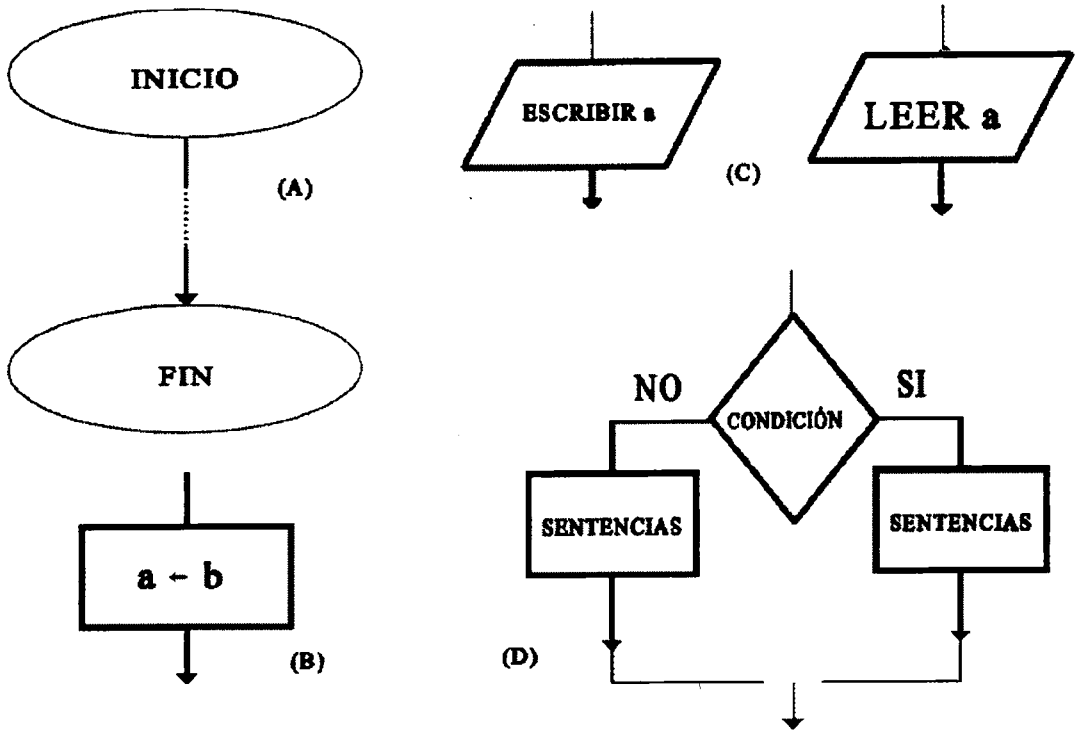


Figura 7.4: Símbolos más usados para la creación de organigramas.

Cada paso del algoritmo se escribe en el interior de estos símbolos y la secuencia en que se deben realizar se representa mediante flechas denominadas líneas de flujo.

Los símbolos más comunmente utilizados son los que aparecen en la figura 7.4.

En la figura A se señalan dos símbolos terminales que representan el inicio y final del algoritmo respectivamente. Entre ambos irán todas las operaciones que deban realizarse en el algoritmo para resolver el problema.

En la figura B se muestra el símbolo de operación o proceso. Es decir, en el se escribirá cualquier operación que se realice en el algoritmo (excluyendo las de entrada/salida).

En las dos figuras señaladas con la letra C se representan las operaciones de entrada y salida. Al igual que en seudocódigo, no se especifica de ninguna forma el medio externo del cual proviene la entrada o al que se dirige la salida.

En la figura D se representa la estructura de control condicional: la condición lógica se escribe en rombo central y a cada uno de los lados las acciones que se deben realizar si la condición es verdadera, y las que se deben hacer si la condición es falsa.

Con estos símbolos se puede escribir cualquier algoritmo. Las estructuras repetitivas –que tenían una forma específica de escribirse en seudocódigo– en diagramas se flujo se expresan en función de los símbolos anteriormente descritos.

Ejemplo: El algoritmo que escribía los números pares existentes entre 2 y 50, se describe en seudocódigo de la siguiente forma (ver figura 7.5):

Esta es, para este ejemplo, la forma de representar en diagrama de flujo la estructura

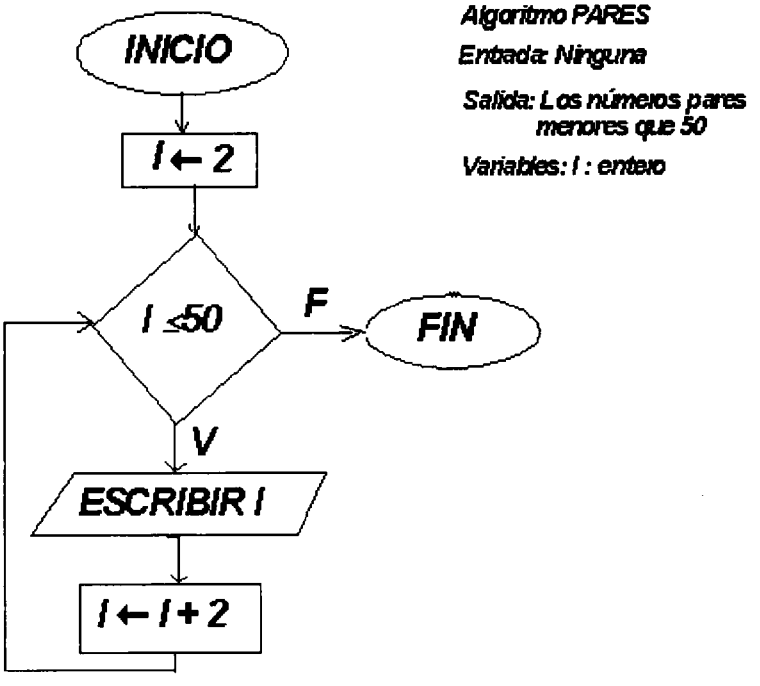
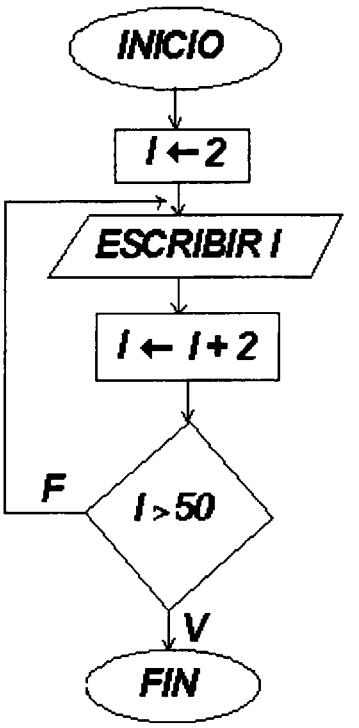


Figura 7.5: Algoritmo para escribir los números pares comprendidos entre 2 y 50.



Algoritmo PARES
Entrada: Ninguna
Salida: Los números pares menores que 50
Variables: I : entero

Figura 7.6: Algoritmo para escribir los números pares utilizando la estructura reperitiva REPETIR...HASTA QUE.

repetitiva MIENTRAS. También así se representa -para este caso- la sentencia repetitiva con un número fijo de iteraciones. Este último tipo de estrutura repetitiva, se describe en diagrama de flujo expresada como una estrutura repetitiva condicional.

Ejemplo: El mismo ejemplo anterior, pero utilizando la estructura repetitiva REPETIR_HASTA (ver figura 7.6):

Por último, sólo queda señalar que para representar llamadas a subalgoritmos –concretamente a procedimientos–, cuyo significado y diseño se verá a continuación, se utiliza el símbolo que muestra la figura 7.7.

7.3.3.3 Diagramas N-S.

Los diagramas N-S constituyen un método gráfico de descripción de algoritmos desarrollado por I. Nassi y B. Scheneiderman. En este tipo de diagramas el símbolo básico es el rectángulo, dentro del cual se describen las acciones que constituyen el algoritmo. Se podría decir que un diagrama N-S es un diagrama de flujo en el que se omiten las líneas de unión y las cajas o rectángulos son contiguos.

El aspecto general de un algoritmo expresado según esta forma de descripción es el que muestra la figura 7.8.

Las estructura básica secuencial de un algoritmo se representa como aparece en la figura

Figura 7.7: Símbolo utilizado en un organigrama para representar la llamada a un subalgoritmo.

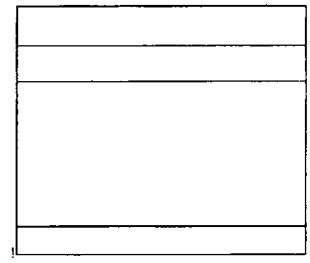


Figura 7.8: Los diagramas N-S están formados por un conjunto de cajas apiladas.

7.9.

De esta forma se representan todas las acciones que se realicen de forma secuencial. Es un conjunto de cajas en cada una de las cuales puede aparecer una o varias acciones.

La figura 7.10 muestra el aspecto de una caja que contiene una estructura condicional. En ella la condición ocupa la parte central y a cada uno de los lados se describen las acciones que se realizarán cuando la condición sea verdadera o cuando sea falsa. Dentro de los conjuntos de acciones correspondientes a la opción verdadera y falsa, se utilizará cualquier estructura de control: secuencial, repetitiva o condicional. Puede ocurrir que no se realice nada cuando la condición sea falsa.

De la forma que se indica en la figura 7.11 se representa el ciclo que en seudocódigo se denominaba bucle MIENTRAS. Describe un conjunto de acciones que se repetirán mientras la condición sea verdadera. Al igual que antes, en la zona donde se ha puesto *Acciones bucle* se detallarán las acciones a realizar con cualquier estructura de control.

La caja de la figura 7.12 representa la estructura repetitiva que realiza un conjunto de acciones hasta que una condición sea verdadera (en seudocódigo representa el ciclo REPETIR_HASTA). Se utilizará con las mismas consideraciones que la estructura repetitiva descrita anteriormente o que la estructura condicional.

Ejemplo: Algoritmo que escribe los números pares menores que 50 (ver figura 7.13).

7.3.4 Diseño descendente. Refinamiento por pasos.

En Programación, al igual que en otros contextos, la resolución de un problema se facilita notablemente si éste se divide en subproblemas. A continuación éstos se descomponen en subproblemas más simples, y así sucesivamente llegar a los problemas más pequeños y fáciles de resolver. Al método de diseño que obtiene la solución de un problema mediante la solución de sus subproblemas, se le denomina **diseño descendente**, y a cada uno de los algoritmos

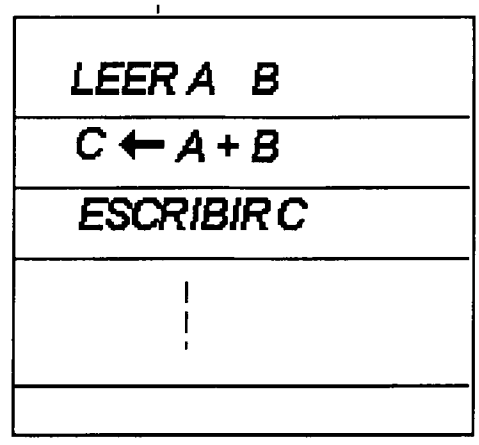


Figura 7.9: En este diagrama podemos ver una operación de lectura, una asignación y una operación de escritura.

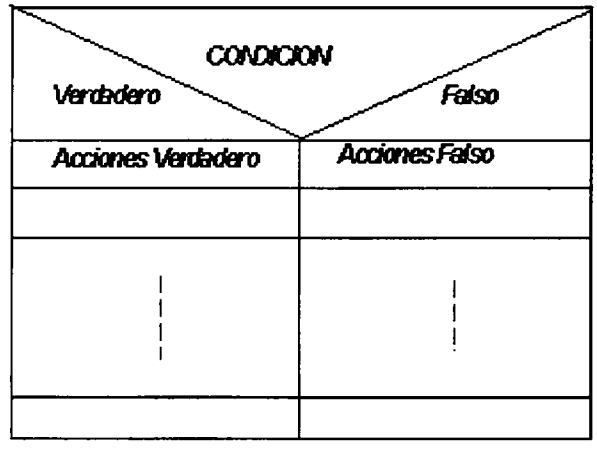


Figura 7.10: Caja de un diagrama N-S quecontiene una estructura condicional.

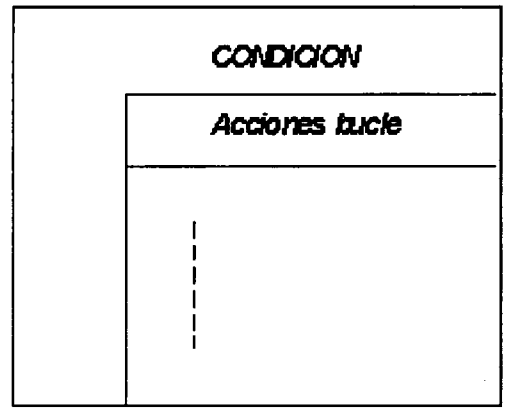


Figura 7.11: Caja de un diagrama N-S que contiene un ciclo condicional del tipo MIENTRAS condición...

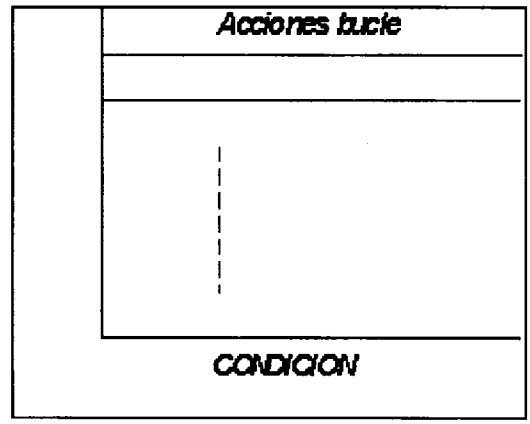


Figura 7.12: Caja de un diagrama N-S que contiene un ciclo condicional REPETIR HASTA condición.

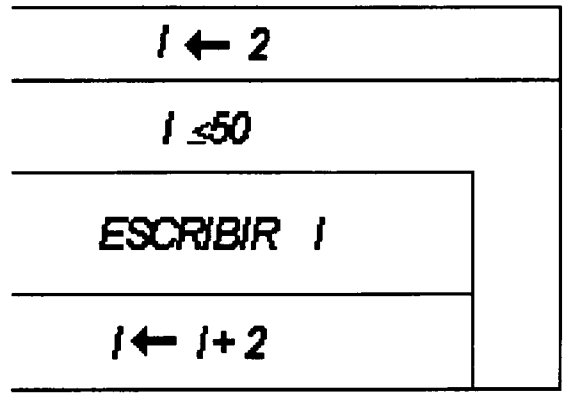


Figura 7.13: Representación del algoritmo que suma los números pares menores que 50 mediante un diagrama N-S.

que resuelven un subproblema, se les conoce como subalgoritmos.

En el diseño de un algoritmo (o de un subalgoritmo), inicialmente sólo se indicarán las tareas más importantes que se deben realizar para resolver el problema (o subproblema). Generalmente esta primera descripción es incompleta y debe ser refinada, por lo que este proceso se completará para indicar, de forma más precisa, los pasos a seguir. Algunos de estos pasos pueden requerir un refinamiento adicional que nos llevará a una tercera descripción del algoritmo. Este proceso, denominado **refinamiento por pasos**, continuará hasta que se obtenga un algoritmo claro, preciso y completo.

Por tanto, al diseñar algoritmos, aplicaremos las dos técnicas mencionadas anteriormente: diseño descendente (ya que se descompondrá el problema en subproblemas) y refinamiento por pasos (puesto que de forma sucesiva, iremos incrementando el nivel de detalle de las descripciones del algoritmo).

El diseño descendente proporciona varias ventajas al diseño de algoritmos y programas:

- Se simplifica el problema, algoritmo y programa resultantes.
- Las soluciones a los diferentes subproblemas pueden ser diseñadas de forma independiente e incluso por diferentes personas.
- El algoritmo y el programa final estará estructurado en módulos, lo que hará más fácil su lectura, comprensión, prueba, verificación y mantenimiento.

7.3.5 Subalgoritmos.

Para resolver un problema se diseñará un algoritmo principal y dentro de éste, para resolver cada uno de los subproblemas se diseñarán distintos subalgoritmos. El algoritmo principal cuando tenga que describir la solución de un subproblema, llamará al subalgoritmo correspondiente. Éste cuando termine de realizar su función, devolverá el control al algoritmo llamador.

A su vez, un subalgoritmo puede contener llamadas a otros subalgoritmos, correspondientes a un refinamiento del mismo, y el proceso que se sigue es el mismo.

7.3.5.1 ¿Cómo se diseña un subalgoritmo?

El proceso de diseño de un subalgoritmo es, en síntesis, el mismo que se sigue para un algoritmo. De hecho, en un subalgoritmo se pueden realizar las mismas acciones que en un algoritmo: realizar cálculos, recibir datos o devolver resultados a cualquier medio externo. Pero tiene una característica distintiva: como realiza una función para otro algoritmo o subalgoritmo, recibe datos de entrada y devuelve resultados al mismo. Esta comunicación entre subalgoritmo llamado y algoritmo (o subalgoritmo) llamante se realiza mediante las **variables de enlace** o **parámetros** y al proceso de emisión y recepción de datos y resultados mediante variables de enlace se denomina **paso de parámetros**.

Existen dos tipos de parámetros:

- Parámetros formales.
- Parámetros actuales.

Los primeros son las variables utilizadas por el subalgoritmo para la emisión y recepción de datos a o desde el algoritmo llamante. Los segundos son las variables, constantes o expresiones que se utilizan en cada llamada por el algoritmo llamante.

Ejemplo: Supongamos que hemos diseñado un *subalgoritmo que calcula el factorial de un número*. Este subalgoritmo recibe como dato de entrada el número del cual se quiere calcular el factorial, y tiene una variable de salida en la cual devolverá el resultado. El subalgoritmo escrito en seudocódigo es el siguiente:

```
subalgoritmo FACTORIAL (x,y)
Entrada: x: real.
Salida: y: real.
Variables:
          auxiliar: real.
          i: entero.

inicio
  auxiliar←1
  PARA i=2 HASTA x CON INCREMENTO 1
       auxiliar←auxiliar * i
  fin_para
  y← auxiliar
fin
```

Un algoritmo que calcule un número combinatorio dado por el teclado, llamará a este subalgoritmo de la siguiente forma:

```
Algoritmo COMBINATORIO
Entrada: a,b:enteros.
Salida: El n\'umero combinatorio.
```

```
Variables:
          a: entero.
          b: entero.
          num1: real.
          num2: real.
          num3: real.
          combi: real.

inicio
  LEER a
  LEER b
  factorial (a, num1)
  factorial (a, num2)
  factorial (a-b, num3)
  combi ← num1/(num2 * num3)
  ESCRIBIR ''El n\'umero combinatorio es '', combi
fin
```

En la definición del subalgoritmo factorial, aparecen dos variables para el intercambio de información entre módulo llamante y llamado, las variables x e y, que son los *argumentos formales* del subalgoritmo. Cuando el algoritmo Combinatorio llama al subalgoritmo, lo hace pasándole distintas variables, según el caso: la primera vez *a* y *num1*, la segunda *b* y *num2* y la tercera *a-b* y *num3*. A cada una de estas tres parejas se le denomina *argumentos actuales.*

Los argumentos actuales pueden cambiar en cada llamada al subprograma mientras que los parámetros formales permanecen fijos para cada subprograma.

La correspondencia entre parámetros actuales y formales se establece como sigue:

- debe existir el mismo número de parámetros formales que actuales.
- cada vez que se llama a un subalgoritmo, se establece una correspondencia uno a uno entre los parámetros actuales y formales del mismo.
- la correspondencia se establece por orden de aparición y no por el nombre de los parámetros.

El paso de parámetros puede realizarse de dos formas diferentes: **por valor y por referencia** (o por variable). Cuando un parámetro se pasa por valor, no se modificará en el subalgoritmo, ya que éste copia el valor en el correspondiente argumento formal para utilizarlo. Por el contrario, cuando un argumento se pasa por referencia, el algoritmo (o subalgoritmo) llamador le pasa al subalgoritmo llamado la dirección de la variable. Un parámetro por referencia sí podrá ser modificado en el subalgoritmo.

En el diseño de subalgoritmos, por defecto se entenderá que los parámetros se pasan por valor. Cuando se quiera indicar que el paso de algún parámetro se realiza por variable, se pondrá delante del mismo la palabra VAR.

Ejemplo: *Subalgoritmo que intercambie el valor de dos variables.*

```
Subalgoritmo INTERCAMBIO (x,y)
Entrada:
```

```
        x: entero.
        y: entero.
Salida:
       x: entero.
       y: entero.
Variables: auxiliar: entero.

inicio
  auxilar← x
  x← y
  y← auxiliar
fin
```

Como no se ha especificado nada, se entiende que el paso de los parámetros x e y se realiza por valor, por lo que el contenido de los parámetros actuales no se modifica, ya que al subalgoritmo sólo se le proporciona una copia del valor de los mismos y no las variables en sí. Es por esto por lo que en un algoritmo como el siguiente:

```
Algoritmo ORDENAR_CUATRO
Entrada: a, b, c, d: entero.
Salida: Los cuatro numeros introducidos como entrada, pero ordenados
        crecientemente.
Variables: i: entero.

inicio
  PARA i=2 HASTA 4 CON INCREMENTO 1
       SI a < b
       ENTONCES Intercambiar (a, b)
       SI b < c
       ENTONCES Intercambiar (b, c)
       SI c < d
       ENTONCES Intercambiar (c, d)
       ESCRIBIR ''Elementos ordenados: '', a, b, c, d
  fin_para
fin
```

Si seguimos este algoritmo veremos escribe los números en el mismo orden en que se introducen, ya que el subalgoritmo *Intercambio* no realiza ningún cambio en los argumentos al ser pasados éstos *por valor*. Sin embargo, si la cabecera del subalgoritmo fuese la siguiente:

Subalgoritmo Intercambio (var x, var y)

el subalgoritmo modificaría el contenido de los argumentos que se pasan y el algoritmo llamador ordenaría realmente las variables de forma decreciente.

Es necesario, en este punto, introducir dos conceptos que se han utilizado ya, aunque sin indicarlo explícitamente: **variable local** y **variable global**. Una variable local es aquella que un módulo necesita para sí mismo y que sólo tiene sentido en el entorno del mismo.

Ejemplo: Las variables *x*, *y* y *auxiliar* son variables locales y sólo son visibles (se conocen) en cualquier lugar interior al subalgoritmo.

Sin embargo, una variable global es aquella declarada en el algoritmo principal cuyo ámbito[4] se extiende al mismo y a todos sus subalgoritmos.

Ejemplo: El algoritmo *ORDENAR_CUATRO* tiene cinco variables: *i*, *a*, *b*, *c* y *d*. Estas cinco variables son globales y son visibles desde cualquier parte del algoritmo o de cualquier subalgoritmo. Desde cualquier punto se puede acceder a ellas y modificarlas, aunque es conveniente que siempre que se quiera modificar desde un subalgoritmo el contenido de una variable global, ésta se pase como parámetro del mismo.

Existen dos tipos de subalgoritmos: **funciones** y **procedimientos.**

7.3.5.2 Funciones.

Una función es un algoritmo que recibe unos valores de entrada y produce un valor denominado **resultado de la función** (valor de la función para los argumentos dados). Una función siempre toma un conjunto de parámetros de entrada y devuelve un único valor a través del nombre de la función. De forma intuitiva, una función se podría considerar como una expresión que devuelve un valor, teniendo en cuenta que el valor que devuelve la función se obtiene a través de un subalgoritmo.

Ejemplo: *Raizcuadrada(x)*, es una función que toma como dato de entrada un número y calcula (y devuelve) el valor de la raíz cuadrada de la entrada.

Todos los lenguajes tienen funciones propias incorporadas y permiten definir funciones al programador[5].

Una función se puede utilizar en el interior de una expresión de igual forma que una variable o constante.

Ejemplo: A←B + Raizcuadrada (9)

Sobre las funciones hay varios aspectos que considerar:

1. **Declaración de una función.** Una función se declara de forma similar a como hasta ahora hemos declarado de forma genérica un subalgoritmo, pero en lugar de escribir la palabra *Subalgoritmo*, se pondrá *Función* para indicar el tipo de subalgoritmo de que se trata.

   Dentro del conjunto de acciones de una función debe existir una única asignación de un valor al nombre de la función.

   Ejemplo: *Función que calcula el resto de una división entre dos números enteros.*

```
Funcion MODULO (dividendo, divisor)
Entradas: dividendo, divisor: entero
Salidas: MODULO que es el resto de la division entre dividendo y
         divisor. Es de tipo entero.
Variables: cociente: entero

inicio
  cociente← dividendo / divisor
```

[4]El ámbito de una variable es el lugar donde esa variable es conocida.

[5]De nuevo, para preservar la independencia del algoritmo respecto al lenguaje de programación, se evitará el uso de funciones incorporadas, y cuando esto no se consiga, el diseño de las mismas en los lenguajes de programación que no las contengan, es inmediato.

```
  MODULO ← dividendo - cociente * divisor
fin
```

2. **Llamada a la función.** Un algoritmo puede invocar a una función simplemente utilizando su nombre con una lista de argumentos actuales.

   Se realizará una llamada a función dentro de una expresión, de una operación de asignación, o de una operación de salida.

   Ejemplo: Un algoritmo podría utilizar la función MÓDULO anteriormente definida de cualquiera de las siguientes formas:

   - a←MODULO (10,3) + 7
   - `SI MODULO (n,3) = 0`
     `ENTONCES ESCRIBIR ''Divisible por 3''`
     `SI NO ESCRIBIR ''No divisible por 3''`
   - `ESCRIBIR MODULO(6,4)`

3. Una llamada a una función implica los siguientes pasos:

   (a) Se le asigna a cada parámetro formal el valor real de su correspondiente parámetro actual (si el paso es por valor) o la variable correspondiente (si el paso es por variable).

   (b) Se realiza el cuerpo de acciones de la función.

   (c) Se devuelve el valor de la función y se retorna al punto de llamada.

7.3.5.3 Procedimientos.

En muchas ocasiones necesitaremos que un subalgoritmo nos devuelva más de un valor. Para ello, las funciones no son adecuadas por lo que necesitamos disponer de otro tipo de subalgoritmo: **los procedimientos**.

Los procedimientos son subalgoritmos en los que se verifica que al nombre del procedimiento no está asociado ningún valor y que los datos de salida (al igual que los de entrada) se transfieren mediante los parámetros.

Un procedimiento no aparecerá en una expresión, ni en una operación de salida o de asignación. Aparecerá en el algoritmo llamador como una operación más que para realizarse necesita de un subalgoritmo que describa los pasos que se deben seguir.

Aspectos a considerar sobre los procedimientos:

1. **Definición de un procedimiento.** Es análoga a la de una función, pero se especificará la palabra *Procedimiento* al inicio de la misma. Generalmente se verificará que al menos alguno de los parámetros se pase por valor para devolver resultados a través de él. Esto no se cumplirá en el caso de que el procedimiento no devuelva valores al algoritmo llamador.

2. **Llamada al procedimiento.** Para llamar a un procedimiento sólo hay que indicar el nombre y la lista de parámetros actuales.

3. Una llamada a un procedimiento implica los siguientes pasos:

   (a) Se establece la correspondencia adecuada entre parámetros formales y actuales.

   (b) Se realiza el cuerpo de acciones del procedimiento.

   (c) No se devuelve ningún valor a través del nombre del procedimiento (aunque sí a través de los parámetros[6]), y se devuelve el control al algoritmo llamador.

Ejemplos:

- El subalgoritmo *Intercambio* de un ejemplo anterior, es en realidad un procedimiento ya que devuelve dos valores y ninguno de ellos asociado al nombre del procedimiento.
- El subalgoritmo **Factorial** planteado también en un ejemplo anterior, se definió como un procedimiento ya que, aunque devuelve un único valor, éste no está asociado al nombre del subalgoritmo. Se podría también haber planteado como una función.

7.4 Codificación

Una vez analizado el problema, diseñado el algoritmo y representado con alguno de los métodos mencionados, es necesario expresarlo según un lenguaje de programación para así convertirlo en un programa y ejecutarlo, obteniendo la solución al problema.

Un **programa** es un conjunto de sentencias que se dan a una computadora indicándole las operaciones que se desea que realice. Es un conjunto de sentencias que ha de procesar un ordenador con el objetivo de obtener unos resultados o datos de salida a partir de unos datos iniciales o de entrada.

Las **sentencias** son conjuntos de símbolos y se clasifican en dos tipos:

- **Sentencias imperativas** o **instrucciones**: Representan una orden para el ordenador.
- **Sentencias declarativas**: Proporcionan información sobre los datos que maneja el programa.

En el código fuente de un programa pueden también aparecer las llamadas **directivas**, que son órdenes al compilador para que haga algo antes de la compilación o durante la misma.

El proceso de transformación de un algoritmo en un programa se denomina **codificación** ya que al algoritmo transcrito a un lenguaje de programación específico se le denomina **código**. Este proceso es, en general, una tarea mecánica.

Tras la codificación el programa se ejecutará y se realizará la prueba y verificación del mismo para comprobar que el programa realiza la función adecuada y de la forma correcta.

7.5 Características de los programas. Programación estructurada.

Para un determinado problema se pueden diseñar distintos algoritmos y, consecuentemente, distintos programas. La elección del algoritmo más adecuado se puede basar en una serie

[6]Sólo en el caso de que los parámetros se pasen por variable.

de características que adquieren gran importancia a la hora de evaluar el coste de diseño y mantenimiento.

Las características generales que debe reunir un programa (y por tanto su algoritmo) son las siguientes:

1. **Legibilidad.** Es importante que todo programa sea fácil de leer y entender.
2. **Portabilidad.** El diseño del algoritmo debe permitir la codificación en cualquier lenguaje de programación, así como la instalación del programa en distintos sistemas.
3. **Modificabilidad.** Debe ser fácil realizar modificaciones y actualizaciones sobre el programa, es decir, debe ser fácil realizar el mantenimiento del mismo.
4. **Eficiencia.** Se deben aprovechar al máximo los recursos del ordenador, especialmente la memoria utilizada. De igual forma, se debe minimizar el tiempo de ejecución.
5. **Modularidad.** El programa debe estar dividido de una forma inteligente en módulos, cada uno de los cuales realice una subparte del proceso de resolución del problema.
6. **Estructuración.** El programa debe cumplir las reglas de la **Programación Estructurada**, para facilitar la depuración y mantenimiento del mismo.

7.5.1 ¿En qué consiste la Programación Estructurada?

Bajo el nombre de Programación Estructurada se incluye un conjunto de técnicas que aumentan la productividad de un programa, reduciendo bastante el tiempo requerido para escribir, verificar, depurar (quitar errores) y mantener los programas.

La programación estructurada utiliza un número limitado de estructuras de control que minimizan la complejidad de los problemas y por tanto reducen los errores. Se basa en el teorema de **BOHM-JACOPINI** que afirma que cualquier programa, por complejo que sea, puede escribirse utilizando tan sólo estas tres estructuras de control: **secuencial**, **selectiva** y **repetitiva**.

Un programa estructurado por tanto, no contendrá instrucciones de salto.

De las tres formas de representación estudiadas, las adecuadas para diseñar algoritmos que den lugar a programas estructurados son los **diagramas N-S** y el **seudocódigo**. Los organigramas no son adecuados ya que utilizan flechas para indicar el flujo de control (sin ninguna limitación) y al codificar el algoritmo en un lenguaje de programación, estas flechas no siempre pueden ser traducidas por una estructura secuencial, condicional o repetitiva, lo que obliga -en ocasiones- al uso de instrucciones de salto (y el programa resultante no sería estructurado).

7.6 Estructuras de datos

Hasta ahora hemos trabajado con el concepto de **dato de tipo simple**, que representa valores de tipo simple, como un número entero, real o un carácter. En muchas situaciones, sin embargo, se necesita procesar una colección de valores que están relacionados entre sí de forma que constituyen una unidad para su tratamiento. Por ejemplo, si se quiere manejar una lista de 50 nombres, es conveniente tratar este conjunto de datos de forma unitaria en lugar de utilizar 50 variables (una para cada dato de tipo simple). El procesamiento de estos conjuntos

de datos, utilizando datos simples, puede ser difícil y por ello, la mayoría de los lenguajes de programación incluyen **tipos abstractos de datos** y **estructuras de datos**.

Una **estructura de datos** es una colección de datos que puede ser caracterizada por la relación entre ellos. Algunos ejemplos de estructuras de datos son: **cadena de caracteres**, **arrays**, **registros**, **ficheros**, **conjuntos**, **listas**, **pilas**, **colas**, **árboles**, etc. Estas estructuras permiten el manejo de grandes, potencialmente interminables, volúmenes de datos mediante operaciones relativamente sencillas.

Las **estructuras de datos** pueden ser **estáticas** o **dinámicas**. Una **estructura de datos estática** es aquella en la que el tamaño ocupado en memoria se define antes de que el programa se ejecute y no puede modificarse durante la ejecución del programa. Las **estructuras de datos dinámicas**, mediante el uso del tipo de datos puntero, superan las limitaciones en el tamaño de memoria ocupada propias de las estructuras estáticas. En ellas el espacio ocupado crece o decrece a medida que evoluciona (se ejecuta) el programa correspondiente. Este último tipo de estructuras no son soportadas por algunos lenguajes, pero son muy útiles en aquellos que los ofrecen. Desde el punto algorítmico se estudia esta distinción entre estructuras de datos porque el tratamiento de las dinámicas es diferente al de las estáticas, ya que la forma de operar con el tipo de datos en que se apoyan (el puntero) es diferente a la del resto de los tipos básicos.

Cuando hablamos de una estructura de datos y del conjunto de operaciones que se definen sobre ella, nos estamos refiriendo a un **tipo abstracto de datos**. Dos tipos abstractos de datos se pueden diferenciar en la estructura de datos en que se apoyan y/o en el conjunto de operaciones que se definen sobre ella.

Con estos dos conceptos, podemos clasificar, de una forma general y sin exhaustividad, los tipos de datos que se pueden utilizar en un programa y por tanto en un algoritmo:

- **Tipos de datos simples:**
  - entero,
  - real,
  - carácter,
  - lógico,
  - puntero.
- **Tipos de datos abstractos:**
  - cadenas de caracteres,
  - arrays,
  - registros,
  - ficheros,
  - listas,
  - pilas,
  - colas,
  - conjuntos,
  - árboles,
  - grafos.

7.6.1 Cadenas de caracteres.

Una cadena de caracteres es una sucesión de caracteres representables por el ordenador que se almacenan en un área contigua de memoria. Generalmente, las cadenas de caracteres están delimitadas mediante comillas simples, dobles o apóstrofos (a los que se denomina delimitadores de cadena).

Ejemplos: Distintas cadenas de caracteres:

'Hola'
'¿'
'958 - 13 23 23'
'ñalsk123412 123asd'

Un tipo especial de cadena de caracteres es la denominada *cadena vacía* o *nula*, que es aquella que no tiene ningún caracter y, por tanto, su longitud es cero. La cadena nula se representa por ''. No se debe confundir con la cadena compuesta del caracter espacio en blanco (' '), ya que ésta última tiene de longitud el número de espacios en blanco que posea.

Las variables de tipo cadena se definen de la siguiente forma:

```
x: cadena(10)
y: cadena(100)
```

Entre () se indica el número máximo de caracteres que se pueden almacenar en la variable.

Las **operaciones** que se suelen realizar con este tipo abstracto de datos, son:

- determinar la longitud de la cadena, obteniendo el número de caracteres que contiene,
- concatenar dos cadenas, preservando el orden de los caracteres de cada una de ellas,
- comparar dos cadenas,
- extraer una subcadena de una cadena,
- insertar una cadena en otra,
- borrar una subcadena de una cadena,
- buscar una cadena en otra, es decir, determinar si una determinada cadena forma parte de otra más grande, indicándose en qué posición aparece la subcadena en la cadena mayor, etc.

El tipo de dato abstracto cadena de caracteres se puede definir con distintas estructura de datos -como en cualquier otro tipo abstracto de datos-, pero generalmente se define como un array de caracteres de una longitud fija. El tipo array se verá en el siguiente apartado.

7.6.2 Array.

Un **array** o matriz, es un conjunto *finito* y *ordenado* de elementos del mismo tipo. La palabra *ordenado* indica que el elemento primero, segundo, tercero, etc., de un array puede ser identificado mediante un índice que determina su posición dentro del array. Los elementos de un array pueden ser de cualquier tipo (simple o estructurado).

Los arrays se conocen también como **matrices** en Matemáticas, y tablas en cálculos financieros.

El tipo más simple de array es el array unidimensional o vector, y se declara de la siguiente forma:

```
X: matriz(1..5) de enteros
```

Dentro de una matriz, a cada elemento se le referencia mediante el nombre de la variable tipo matriz y a continuación el índice que indica la posición del elemento dentro de la matriz. Para la matriz declarada anteriormente, se referenciaría al primer elemento de la siguiente forma: `x(1)`.

Aquí se observa una característica importante que diferencia a los tipos de datos: los tipos de datos simples tienen como característica común que cada variable representa a un elemento, mientras en los tipos estructurados un identificador puede representar a múltiples datos individuales, pudiendo cada uno de estos ser referenciado independientemente.

Los arrays pueden tener más de una dimensión. Un array bidimensional por ejemplo, puede entenderse como una disposición de filas y columnas. Se requieren dos índices para localizar un elemento en un array bidimensional, uno para la columna y otro para la fila. Un array bidimensional se declara así:

```
Y: matriz(1..10,1..20) de reales
```

Se tendrá acceso al elemento situado en la fila tres y columna 15 mediante la siguiente expresión: `Y(3,15)`.

Las **operaciones** que se suelen realizar con un array son:

- asignar un valor a un elemento del array,
- borrar un elemento,
- buscar un elemento y ver su valor,
- ordenar el array, etc.

En general, las operaciones con arrays implican el tratamiento individual de los elementos del array.

Ejemplo: Para leer el vector anterior, tendríamos que hacer lo siguiente:

```
DESDE i=1 HASTA 5 CON INCREMENTO 1
   LEER X(i)
```

7.6.3 Registros.

Hasta ahora hemos tratado sólo dos tipos de datos compuestos o estructurados: la matriz -que incluye como caso particular el vector- y la cadena de caracteres. Tanto uno como otro tienen una importante restricción: Todos sus componentes son del mismo tipo de dato.

En una gran variedad de problemas reales, los datos que se almacenan referentes a un objeto concreto son de distinta naturaleza, y por tanto, de distinto tipo. Pensemos, por ejemplo en los datos que se almacenan sobre un alumno al matricularse (nombre, apellidos, edad, asignaturas, becas, ...). Necesitamos por tanto, un tipo de datos compuesto cuyas "celdas"no sean necesariamente todas del mismo tipo. Este tipo de datos es el registro.

Un **registro** es una colección de información, relativa a una misma entidad. Es un tipo de dato estructurado con un número fijo de componentes (no necesariamente del mismo tipo) a los que se accede por su nombre, no por su posición. Cada uno de los componentes recibe el nombre de campo.

En la declaración de un registro se especifica, además del nombre de dicho registro, el nombre y el tipo de cada componente (campo):

```
NombreCompleto: registro
                  nombre: cadena(20)
                  apellido1: cadena(20)
                  apellido2: cadena(20)
                fin_NombreCompleto
```

Ejemplo: Necesitamos almacenar información relativa a clientes de una empresa. La información que queremos guardar es: nombre, edad, NIF, teléfono, localidad y provincia de trabajo. Estos elementos de información son de diferente tipo, pero son relativos a una misma entidad: un cliente. Para guardar esta información utilizaremos el tipo abstracto de datos registro con las operaciones que necesitemos usar. Se declararía asi:

```
persona: registro
           nombre: cadena(60)
           edad: entero
           nif: cadena(9)
           telefono: cadena(10)
           localidad: cadena(25)
           provincia: cadena(20)
         fin_persona
```

Generalmente, sobre el tipo de datos registro se suelen realizar algunas de las siguientes **operaciones**:

- asignación de registros,
- acceso a los campos de un registro,

Para acceder a un campo concreto de un registro, se usa un selector de campo que no es, ni más ni menos, que la expresión usada para nombrarlo. El selector está formado por el nombre de la variable de tipo registro y el identificador del campo concreto al que se quiere acceder, separados por un punto.

Siguiendo esta nomenclatura y partiendo del ejemplo anterior, ejemplos válidos de selectores de campo serían:

```
persona.nif
persona.provincia
persona.edad
```

El campo se comporta como una variable del tipo concreto con el que fue declarado, por lo que sobre él, podrán aplicarse las operaciones propias del correspondiente tipo de dato.

7.6.4 Ficheros.

Al trabajar con memoria principal nos encontramos con dos limitaciones importantes: la capacidad de la misma y la volatilidad de los datos que se encuentran en ella. Esto nos hace ver la necesidad de guardar la información en dispositivos de almacenamiento secundario.

Un **fichero** es un conjunto de datos almacenados en memoria secundaria. Dicho de otra forma, es una secuencia de elementos, generalmente todos del mismo tipo, a los que se denomina **registros**. No hay que confundir los registros de un fichero -que pueden ser de cualquier tipo: entero, real, cadena de caracteres, array, registro, etc.- con el tipo de datos registro definido en el punto anterior.

Existen dos **tipos de acceso** a los datos almacenados en un archivo:

- el **acceso secuencial**, donde se accede a los registros según el orden en que estén almacenados, uno tras otro, y
- el **acceso directo**, donde se accede a un registro determinado sin que ello implique la consulta de los registros precedentes.

Otra característica de los archivos es su **organización**, que define la forma en la que los registros se disponen sobre el soporte de almacenamiento. Existen tres formas distintas de organización:

- **secuencial**, en la que los registros se almacenan consecutivamente sobre el soporte externo, lo que obliga a un acceso secuencial,
- **directa**, donde los datos se disponen de forma que se accede a ellos por su posición en el fichero, y
- la organización **secuencial indexada**, en la que los registros están situados en el soporte externo según un determinado campo de los registros al que se denomina **clave**.

Las **operaciones** que se suelen realizar con archivos son las siguientes:

- **Creacin**. Es la primera operación que sufrirá un archivo de datos. La creación exige determinar qué organización y estructura tendrá el archivo, así como reservar espacio en el soporte de almacenamiento.
- **Consulta** (lectura). Es la operación que permite acceder al fichero para conocer el contenido de sus registros.
- **Posicionamiento** del apuntador.
- **Actualización** (escritura).Esta operación permite tener al día el fichero, para lo cual se podrán insertar nuevos registros, eliminar registros existentes y/o modificar la información que contiene los registros.
- **Clasificación.** La clasificación u ordenación de un fichero consistirá en situar los registros en el fichero de acuerdo al valor que posean en su campo clave.
- **Destrucción o borrado del fichero.** Cuando se destruye un fichero éste ya no existe, y por tanto, no se puede utilizar: no se puede acceder a ninguno de sus registros.

- **Apertura.** Cuando se abre un fichero simplemente se prepara para ser utilizado. La diferencia con la creación es que en la apertura el fichero a abrir ya existe, mientras que en la creación no.

- **Cierre.** El objetivo de esta operación es cortar el uso del fichero.

7.6.5 Listas.

Una lista es un conjunto de elementos de un tipo dado que se encuentran ordenados según la posición que ocupa cada elemento en la lista.

Las **operaciones** que se pueden realizar con una lista son:

- insertar un elemento en una posición concreta de la lista,

- localizar un elemento,

- recuperar el elemento que se encuentra en una posición,

- suprimir un elemento que está en una posición determinada,

- devolver las posiciones siguientes y anterior a una dada, y

- ver si una lista está o no vacía.

Existen varios **tipos de listas** dependiendo de la estructura de datos que se utilice para su implementación:

- las **listas lineales**, donde los elementos se almacenan de forma contigua (en posiciones consecutivas de memoria), y

- las **listas enlazadas**, donde cada elemento de la lista contiene la posición del siguiente elemento de la lista.

7.6.6 Pilas.

Una **pila** es un tipo especial de lista en la que la inserción o borrado de elementos se realiza sólo por un extremo denominado cima o tope de la pila. Debido a que el último elemento que se pone en la pila es el primero que se puede sacar, a estas estructuras se le conoce por el nombre de listas **LIFO** (*Last In, First Out* = Ultimo en entrar, Primero en salir).

Las **operaciones** más usuales asociadas a las pilas son:

- meter en el tope de la pila un elemento,

- sacar el elemento que está en el tope de la pila, y

- determinar si una pila está o no vacía.

7.6.7 Colas.

Las colas son otro tipo especial de lista donde los elementos se insertan por el final de la lista y se eliminan por el principio. Las colas reciben el nombre de listas **FIFO** (*First In, First Out* = Primero en entrar, Primero en salir).

Las **operaciones** para las colas son análogas a las de las pilas, simplemente se diferencian en el lugar por donde se realizan la inserción y el borrado de sus elementos:

- meter por el final de la cola un elemento,
- sacar un elemento del principio de la cola, y
- determinar si la cola está o no sin elementos.

7.6.8 Conjuntos.

Un **conjunto** es una colección de elementos no repetidos. Esta es la diferencia fundamental con las listas.

En los conjuntos, los elementos pueden estar ordenados linealmente por la relación de precedencia "<" (se lee *menor que* o *precede a*), en cuyo caso el **conjunto** se denominará **ordenado**.

Las **operaciones** principales que se pueden realizar con conjuntos son:

- unir dos conjuntos,
- intersecar dos conjuntos,
- realizar la diferencia entre dos conjuntos,
- determinar si un elemento pertenece o no al conjunto (función de pertenecia),
- insertar un elemento en un conjunto,
- suprimir un elemento de un conjunto, y
- ver si el conjunto es el conjunto vacío.

7.6.9 Árboles.

Un **árbol** es una colección de elementos, llamados **nodos**, que poseen una relación de paternidad, la cual impone una estructura jerárquica sobre los nodos. Existe un nodo especial, denominado **raíz**, y es el nodo a partir del cual se derivan todos los nodos del árbol. Este nodo raíz es el único que no tiene padre. Otro tipo de nodo es el **nodo terminal** u **hoja**, que es aquel nodo que no contiene ningún hijo colgando de él. Los nodos de un mismo padre se denominan **nodos hermanos**.

Un **árbol general** es aquel en el que cada padre puede tener varios hijos y un **árbol binario** restringe el número de hijos a dos como máximo.

Las **operaciones** básicas que se pueden realizar con árboles son:

- conocer el padre de un nodo,
- conocer el hijo más a la izquierda de un nodo padre,

- conocer el hermano a la derecha de un nodo,
- conocer el nodo raíz,
- crear un árbol,
- colocar un subárbol en un nodo de un árbol.
- determinar si el árbol está o no vacío.

7.6.10 Grafos.

Matemáticamente un **grafo** está formado por un conjunto de nodos (también llamados **vértices**) y un conjunto de aristas (o **arcos**) que conectan diferentes nodos. Formalmente se define así: G=(V,A), donde V(G) es un conjunto no vacío de vértices y A(G) es un conjunto de aristas (pares de vértices). Cada arista se denota escribiendo las nombres de los dos nodos que une.

Existen dos **tipos de grafos**:

- los **grafos dirigidos**, donde la dirección de la línea es indicada por el nodo que se lista primero y
- los **grafos no dirigidos**, donde el par de vértices no está ordenado.

Dos **vértices** se dice que son **adyacentes** cuando existe una arista que los une.
De forma general, las **operaciones** que se podrían realizar sobre un grafo serían:

- Crear un grafo a partir -como mínimo- de un nodo.
- Insertar nodos o aristas en el grafo.
- Eliminar nodos y/o aristas en el grafo.

7.7 Recursividad

Es muy común en Matemáticas definir conceptos en términos del proceso usado para generarlos.

Ejemplo: Una descripción matemática del factorial de un número es:

$$n! = \begin{cases} 1, & \text{si } n = 0. \\ n * n(n-1)!, & \text{si } n > 0 \end{cases}$$

Esta definición del factorial de un número recibe el nombre de *recursiva*, ya que se define en términos de ella misma. Así, para calcular 3! tendremos que tomar la segunda rama de la función: 3! = 3 * 2!. Para calcular 3! hay que volver a considerar la función ya que ahora hay que determinar el valor de 2!. Tomando la segunda rama de nuevo: 2! = 2 * 1!. Nos encontramos en la misma situación, por lo que 1! = 1 * 0!. En este caso, la primera rama nos indica que valor de 0! es 1, por lo que podemos ir completando cada uno de los productos que se han quedado sin realizar por no conocer el valor de los diferentes factoriales:

$$1! = 1 * 1 = 1$$
$$2! = 2 * 1 = 2$$
$$3! = 3 * 2 = 6$$

Esta función se expresaría en seudocódigo de la siguiente forma:

```
Funcion FACTORIAL (n)
Entradas: n :entero.
Salidas: Factorial: entero.

inicio
  SI n = 0                                    (1)
  ENTONCES Factorial←1                        (2)
  SI NO Factorial←n * Factorial (n-1)         (3)
fin
```

Observando este algoritmo se introducirán los conceptos de **caso base** y **caso general**. Las líneas (1) y (2) determinan lo que se conoce como **caso base**, que representa el caso para el cual la solución puede establecerse en términos no recursivos. El **caso general** (mostrado en la línea (3)) es el caso para el que la solución se expresa en términos de una versión más pequeña de sí mismo, es decir, la función se vuelve a llamar a sí misma. Por tanto, se puede definir un **algoritmo recursivo** como una solución que se expresa en términos de instancias más pequeñas de sí mismo y de un caso base que finalizaría la recursividad. El subalgoritmo se estará llamando a sí mismo contínuamente (aunque con diferentes valores para los argumentos) hasta el momento en que se cumpla el caso base, a partir del cual se producirá la resolución de todas las llamadas anteriores hasta obtener el valor pedido.

Al igual que se pueden realizar funciones recursivas, también se pueden crear procedimientos recursivos, aunque en este caso los parámetros del procedimiento actúan como lugar en el que se devuelven los resultados de las llamadas recursivas.

Supongamos ahora que hemos creado una solución recursiva para un problema. Hasta que no lo codifiquemos utilizando un lenguaje de programación que soporte la recursividad y lo ejecutemos no sabremos realmente si el algoritmo desarrollado era correcto. Sería útil tener una técnica que nos ayudase a determinar si un algoritmo recursivo funcionará. Un método muy intuitivo y comúnmente utilizado es el **Método de las Tres preguntas:**

1. **Pregunta Caso-base:** ¿ Hay una salida no recursiva de algoritmo ? ¿ El algoritmo trabaja correctamente para ese caso base ?

2. **Pregunta Llamador más pequeño:** ¿ Cada llamada recursiva a la función se refiere a un caso más pequeño del original ?

3. **Pregunta Caso General:** Suponiendo que la llamadas recursivas funcionan correctamente, ¿ Funciona correctamente todo el algoritmo ?

Ejemplo: Veamos la respuesta de estas tres preguntas para el ejemplo de la función Factorial:

1. El caso base ocurre cuando n = 0, por lo que a Factorial se le asigna el valor 1, que es el valor correcto para 0!. La respuesta es sí.

2. Cada llamada envía un valor decrementado del parámetro original, es decir, la secuencia de llamadas sería: Factorial (n), Factorial (n-1), Factorial (n-2), ..., Factorial(0). En este caso, la respuesta también es afirmativa.

3. Suponiendo que la llamada recursiva Factorial (n-1) nos devuelve la solución correcta de (n-1)!, hacemos la asignación de n * (n-1)! a Factorial, por lo que tendremos la definición del factorial de un número positivo. Se verifica que el algoritmo funciona correctamente en el caso de que las llamadas recursivas también lo hagan.

En resumen, son dos las reglas fundamentales que hay que tener en cuenta a la hora de diseñar un algoritmo recursivo:

1. Determinar el caso base en el que el problema puede expresarse no recursivamente (nos servirá para parar la recursividad).

2. Resolver el caso general correctamente en términos de un caso más pequeño del problema.

Si se cumplen estas reglas se evitará que el algoritmo se llame recursivamente de forma indefinida y origine que, una vez codificado y ejecutándose en una máquina, agote el espacio de memoria reservado para llamadas.

Se utiliza una sentencia de selección para controlar las llamadas recursivas, en lugar de los diferentes tipos de bucles existentes, que son utilizados normalmente en las versiones iterativas del problema.

7.7.1 ¿Qué aporta la recursividad sobre la iteración?.

En general, la solución no recursiva (iterativa) es más eficiente en términos de tiempo de ejecución y de espacio, ya que la llamada recursiva origina que se asigne espacio para las variables locales por cada llamada, existiendo además un gasto adicional en tiempo por esa llamada. Esto puede originar, como ya se ha comentado, que el sistema no tenga suficiente espacio para obtener la solución recursiva de algunos problemas, o sea excesivamente lenta. Pero, en algunos casos, la solución recursiva es más simple y natural de escribir. Por ello, cuando la definición de un problema sea inherentemente recursiva, se debe considerar la solución recursiva.

Como guía general, si la solución no recursiva es más corta, o no mucho más larga, que la versión recursiva, se aconseja **no** usar la recursividad.

7.8 El estilo de programación

La legibilidad es uno de los criterios más destacados a la hora de decidir si un programa es bueno o no. El motivo es sencillo: la legibilidad es la llave para la comprensión de un programa y los programas que no se puedan comprender no podrán ser modificados ni se les podrá dar mantenimiento, por lo que tendrán poco valor. Incluso el mismo autor de un programa, tendrá dificultades para recordar lo que hacía un programa (y la forma de hacerlo) después de un intervalo de tiempo.

En el formato y la apariencia de un listado de un programa es donde más se puede hacer por mejorar la legibilidad del mismo. En general, el objetivo debe ser hacer el programa más legible, comprensible y fácilmente modificable. Para alcanzar este objetivo existen una serie de elementos de estilo que sirven de guía para producir programas de forma correcta. Estas consideraciones de estilo se deben tener en cuenta desde el diseño del algoritmo. Los principales elementos de estilo son:

- **Comentarios.** Constituyen el principal exponente en la documentación interna de un programa. Sirven para ayudar al lector a comprender el propósito y funcionamiento de determinadas secciones de código. En el uso de los comentarios hay que tener en cuenta varios aspectos:

  - Comentarios no adecuados pueden deteriorar un buen código mientras que buenos comentarios no pueden hacer mucho para mejorar una mala codificación.
  - La abundancia de comentarios aumenta la confusión.
  - Los comentarios y el código deben estar de acuerdo. Cuando se realiza una modificación en el código debe realizarse un cambio similar en cualquier comentario relacionado, evitando así incoherencias con el código.

  En general, es deseable utilizar comentarios en los siguientes lugares:

  - En las declaraciones de variables, comentando el uso de cada variable.
  - En las estructuras de control, explicando la función que realizan.
  - En las llamadas a subrutinas, para explicar su función y los efectos que realizan sobre los argumentos.
  - En la lista de parámetros de las subrutinas, comentando la función de cada uno de ellos y qué valores puede o no tomar.
  - En las zonas de código no obvio, es decir, en cualquier parte del algoritmo o del programa que sea confuso y/o complicado.

- **Código autodocumentado.** Los identificadores de un programa deben tener significado, es decir, su nombre debe ser representativo de los valores que almacenan o del procesamiento que realizan. Así, es mejor nombrar a una función que obtiene la nota media de las calificaciones de varios alumnos como Obtener_Nota_Media, que utilizar otro nombre como ONM, o simplemente A.

- **Formateado del código.** Se basa fundamentalmente en el uso de espacios en blanco para indentar sentencias dentro de las estructuras de control tales como bucles o bifurcaciones y el uso de líneas en blanco para separar unidades lógicas dentro del código. Esto aumentará la legibilidad del código.

- **Uso de constantes.** Permitirá ofrecer una mayor legibilidad al código, a la vez que facilidad de modificación. Por ejemplo, el valor constante 3.1415 en un código, probablemente no aporte información sobre lo que representa. Sin embargo el identificador PI sí. Además, si quisiéramos cambiar el valor 3.1415 en el proceso, tendríamos que buscar cada una de las ocurrencias del valor constante en el algoritmo o programa y modificarlas. Con la utilización de la constante PI en el algoritmo o programa, sólo será necesario cambiar el valor en la inicialización de la constante.

Capítulo 8

Redes de ordenadores.

Si miramos hacia atrás en la historia nos podemos dar cuenta de que cada uno de los tres siglos pasados ha estado dominado por una sola tecnología. El siglo XVIII fue la etapa de los grandes sistemas mecánicos que acompañaron a la Revolución Industrial. El siglo XIX fue la época de la máquina de vapor. Durante el siglo XX, la tecnología clave ha sido la recolección, procesamiento y distribución de la INFORMACIÓN. Entre otros desarrollos, hemos asistido a la instalación de redes telefónicas en todo el mundo, a la invención de la radio y de la televisión, al nacimiento y crecimiento sin precedente de la industria de los computadores, así como a la puesta en órbita de los satélites de COMUNICACIONES.

Todos estos acontecimientos no han pasado desapercibidos para nuestros paisajes urbanísticos, sino que se han ido reflejando en nuestros pueblos y ciudades. Así durante siglos, la religión jugó un papel muy importante en la sociedad. Los edif

icios más altos eran las iglesias y catedrales. Posteriormente se pasó a un mayor "culto" por el dinero y los gigantescos y monstruosos edificios bancarios se veían desde casi todas las partes del municipio. Hoy en día el tema de las comunic

aciones está en continuo desarrollo y rara es la ciudad que no cuenta entre sus edificios más altos con torres y antenas utilizadas para las comunicaciones (los famosos "pirulíes").

A medida que crece nuestra habilidad para recolectar, procesar y distribuir información, la demanda de más sofisticados métodos para procesar dicha información crece todavía con mayor rapidez.

Como es de suponer, todos estos avances han hecho posible que la COMUNICACIÓN entre los hombres sea cada vez más rápida, segura y fiable.

8.1 Introducción. Comunicación de sistemas.

En este primer apartado vamos a introducir la idea de algunos de los principios base en la **comunicación de datos**, además, la terminología y notación usada en este campo. Sin embargo, antes de ver explícitamente la comunicación de datos, examinaremos con más detalle el concepto de **comunicación**.

8.1.1 ¿Por qué comunicar sistemas?.

En general, podemos decir que los sistemas se comunican para *compartir información.*

Comunicar quiere decir pasar o transmitir algo (normalmente información). El término **Información** es usado aquí para indicar algo que conoce el emisor y no el receptor. A lo largo de la historia la capacidad de comunicación ha estado estrictamente limitada por la distancia. Aparte de los tradicionales esquemas de señales, tales como las señales de humo, banderas, linternas, etc... ; el único método para llevar información más allá de lo que la voz humana podía o alcanzaba implicaba el desplazamiento físico de una o más personas desde el emisor hasta el receptor.

La invención del telégrafo, en los primeros años del siglo XIX, permitió el intercambio de información de manera más rápida y segura, debido a que éste ya no estaba limitado por la velocidad a la cual el humano podía viajar.

En los últimos años del siglo XIX y durante el siglo XX, esta habilidad para poder comunicarse a través de largas distancias, en tiempos relativamente pequeños, condujo a un rápido desarrollo en todas las áreas investigación. De esta forma, tanto ingenieros como científicos, podían tener noticias unos de otros con más facilidad, debido a la existencia de caminos de información mejores y más estructurados.

Así mismo, todos estos adelantos ayudaron, con más velocidad, a la distribución de la información por todo el mundo, acelerando el proceso de cambio.

El desarrollo de los computadores durante los últimos cuarenta años ha sido uno de los mayores avances. Pero en un principio estos sólo podían comunicarse con el exterior a través de mecanismos poco eficientes. Aun cuando se inventó el microprocesador y su uso empezó a estar generalizado, la manera más común para introducir información en el computador era a través de las incómodas **tarjetas perforadas**. Paralelamente, obtener datos del computador sólo era posible mediante la impresión de éstos en papel, usando una **impresora**. La transferencia de información entre dos sistemas podía ser llevada a cabo utilizando **cintas magnéticas**, pero siempre y cuando fueran compatibles, es decir, debían ser de un tipo similar. Por otra parte, la información de interés podía encontrarse distribuida entre varios computadores en lugar de estar integrada en un lugar central. Por ello surgió la tendencia de centralizar el almacenamiento de la información. Los mecanismos para conectar máquinas empezaron a tener un uso extendido a principios de los años 80.

Precisamente por estos años, y debido a la facilidad de los computadores para intercambiarse información, tuvo lugar lo que se denominó la **Revolución Tecnológica de la Información**.

La capacidad de proceso de los computadores creció astronómicamente durante este período. Pero, principalmente, la capacidad de obtención de datos en una cantidad comparable con la velocidad de proceso fue la que produjo la explosión de las aplicaciones usando computadores enlazados entre si mediante cables.

8.1.2 ¿Cómo comunicar sistemas?.

En el proceso de comunicación de dos o más sistemas la información necesita algún medio de transporte físico por el cual desplazarse de un lugar a otro. Por ejemplo, al hablar, el medio físico por el que se transmiten las palabras es el aire, ya que es él quien transporta las ondas sonoras. En los sistemas digitales, el medio físico puede ser un cable de alambre, un cable de fibra óptica, etc...

El término más usado para denominar al sistema completo por el cual la información es llevada desde el emisor hasta el receptor es el de **canal de comunicaciones**. Aquí se incluyen tanto el medio físico como otros dispositivos.

En un sistema de comunicaciones electrónico el canal puede incluir algunos dispositivos tales como amplificadores o repetidores. Éstos sirven para aumentar la intensidad de la señal eléctrica a lo largo de su camino.

Por lo general, la información original debe ser modificada para que pueda adaptarse a las características de un canal determinado para su posterior transmisión. Usando otra vez el ejemplo del habla, esta adaptación consiste en convertir lo que una persona piensa, es decir, impulsos nerviosos, en ondas sonoras, utilizando para ello la boca y las cuerdas vocales. En un sistema digital, la transformación correspondiente se lleva a cabo en dos etapas: *codificación* y *modulación.*

La **CODIFICACIÓN** es el proceso por el cual la información digital es convertida de la forma en la que es producida a la forma en la cual podrá ser transmitida. Por ejemplo esto puede producir la adicción de uno o varios bits extra con el fin de permitir el control de posibles errores introducidos durante la transmisión por el canal (errores debidos a ruidos).

La mayoría de los medios físicos (la fibra óptica sería un ejemplo muy claro de ellos) no pueden aceptar directamente señales digitales, por tanto estas deben ser codificadas. La **MODULACIÓN** es el proceso de adaptación al medio de una señal para que pueda ser transmitida conteniendo toda la información (normalmente en una onda electromagnética de distinta frecuencia). En algunos casos, este paso de modulación es reemplazado por un paso más de codificación, adaptando además el dato digital. De este modo, éste puede ser transmitido directamente por el medio.

En todos los casos las propiedades físicas del medio usado para transmitir la información afectarán a los datos transmitidos. Por ejemplo, el aire atenuará las ondas sonoras, haciendo difícil o casi imposible que éstas se oigan a cierta distancia. El ambiente en el cual se habla también produce distorsión, ruidos y/o ecos en las ondas sonoras, haciendo que el mensaje no llegue todo lo deseadamente *limpio* al oído del receptor. Afortunadamente, el poder del cerebro es asombroso y puede interpretar, a veces, lo que se ha recibido con ruidos.

De manera similar, una señal digital puede recibir interferencias cuando se transmite de un lugar a otro. Si la distorsión es grave puede ocurrir que el receptor interprete mal el dato que fue transmitido, es decir, lo que se envió como un 1 lógico puede ser recibido como un 0, o viceversa. De esta forma uno o más errores estarán presentes en el dato recibido.

Varios métodos de codificación hacen posible que el receptor pueda detectar la presencia de errores en el dato enviado. En muchas circunstancias, si el receptor detecta un error, debe preguntar al emisor por el dato que fue transmitido. En términos del habla humana, si alguien no entiende lo que otra persona le está diciendo le pedirá que lo repita de nuevo. En la transmisión digital, más sofisticada, pueden ser usados esquemas de codificación para la detección y corrección de errores sin necesidad de preguntar al emisor por el dato enviado.

Haya o no haya error, en el receptor final debe existir algún mecanismo capaz de interpretar las señales o datos enviados. Un humano usa el oído para transformar las ondas sonoras emitidas a impulsos nerviosos eléctricos, para que el cerebro pueda interpretar lo que está oyendo. En un sistema digital los mecanismos encargados de realizar esta función son la *demodulación* y la *decodificación.*

La **DEMODULACIÓN** se usa para aislar o separar la información del resto de la señal transmitida. El proceso de **DECODIFICACIÓN** examina los pulsos digitales y los convierte de tal forma para que puedan ser usados por el receptor.

Este proceso puede involucrar la detección y corrección de algunos errores introducidos por el canal, siempre y cuando el esquema de codificación permita alguna de estas posibilidades.

Un sistema de comunicaciones digital debe tener, y además en este orden, los componentes

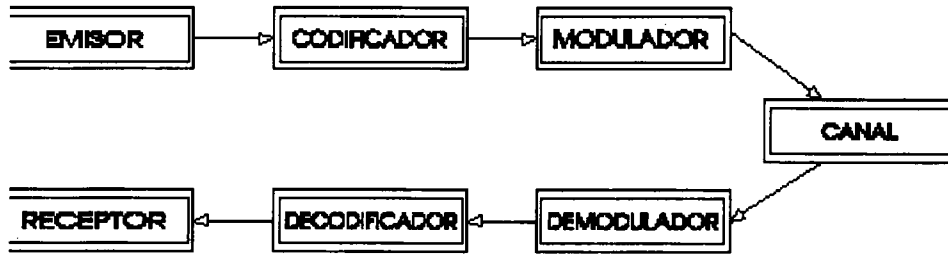


Figura 8.1: Componentes de un sistema de comunicación digital

que aparecen en la figura 8.1.

8.2 Comunicación de datos. Redes de Ordenadores.

El proceso de comunicación en el cual los computadores son usados como fuente y destino de la información digital es conocido como **comunicación de datos**.

Actualmente, en la comunicación de datos, lo más común es conectar entidades con un suficiente poder de procesamiento. De esta forma todas pueden ser tratadas iguales a la hora de la comunicación. Así un mainframe puede ser conectado con un PC y son considerados lógicamente iguales. Evidentemente el poder de procesamiento de estos no es igual pero, como dispositivos de comunicación, ambos son considerados con las mismas necesidades y habilidades.

Como ya hemos dicho, el viejo modelo de tener un solo computador para satisfacer todas las necesidades de cálculo de una determinada organización se está reemplazando con rapidez por otro que considera un número grande de computadores separados físicamente, pero interconectados, que efectúan el mismo trabajo. Estos sistemas se conocen con el nombre de **redes de ordenadores**. En éstas los archivos se almacenan en sistemas de archivos a los que se puede acceder fácilmente por otros usuarios, pero en el que cada computador ejecuta sus propios procesos.

En este campo los computadores son referidos por el nombre se **HOST** (patrón, huésped, anfitrión). Los hosts están conectados mediante una **subred de comunicación**. El trabajo de la subred consiste en enviar mensajes entre los hosts.

Por regla general una subred consta de dos componentes diferentes: **las líneas de transmisión**, por las que se desplaza la información y los **elementos de conmutación**, que son computadores especializados que se utilizan para conectar dos o más líneas de transmisión. Normalmente a los elementos de conmutación se les denomina **IMP (Procesadores de Intercambio de Mensajes)**.

Muchos diseñadores de sistemas construyen redes usando computadores personales, uno por usuario, con los datos guardados en una o más máquinas que funcionan como servidor de archivos compartido.

8.2.1 Componentes de una red.

Una red de ordenadores está compuesta tanto por hardware como por software.

- Por lo que respecta al **Hardware**, una red debe estar compuesta por:
  - **Servidor.** Que es el que ejecuta el S.O. y ofrece los servicios de red a las estaciones de trabajo (almacenamiento de archivos, gestión de usuarios, seguridad, gestión de colas de impresión, etc.).
  - **Estaciones de Trabajo.** Cuando un computador se conecta a una red, el primero se convierte en un nodo del último y se puede tratar como una estación de trabajo. Algunas de estas estaciones no tienen ningún tipo de disco, por lo que arrancan directamente desde el servidor, utilizando una rutina de arranque especial.
  - **Placas de Interfaz de Red.** Cada computador que se va a conectar a la red necesita un interfaz. Aunque el interfaz puede venir incorporado, en la mayor parte de los casos ha de añadirse como un elemento opcional. La placa ha de corresponder al tipo de red que se está utilizando. El cable de la red se conectará a la parte trasera de la placa.
  - **Líneas de Comunicación o Sistema de Cableado.** Está constituido por el cable utilizado para conectar el servidor con las estaciones de trabajo. Más adelante veremos los distintos tipos que se pueden utilizar.
  - **Recursos Compartidos y Periféricos.** Como por ejemplo los dispositivos de almacenamiento ligados al servidor, las unidades de disco óptico, las impresoras, etc.

- El **Software** es el conjunto de programas que hacen que todo el hardware funcione correctamente. Incluye los **controladores**, que son programas que se utilizan para gestionar los dispositivos periféricos y el **Sistema Operativo de Red**, el cual debe ofrecer una gran variedad de servicios. Hay dos tipos básicos de S.O. de red:
  - **Punto a punto**: permite a los usuarios compartir los recursos de sus computadores y acceder a los recursos compartidos de los otros computadores.
  - **Con servidor dedicado:** aquí uno o más computadores se reservan como servidores de archivos, no pudiéndose utilizar para nada más. Los usuarios acceden a los directorios y recursos de los servidores, pero no a los de otros sistemas.

Las características más importantes y necesarias de los S.O. de red avanzados son:

- **Seguridad**. Para impedir o limitar el acceso de un usuario a otros directorios o archivos. Así se evita que algún usuario se conecte a alguna estación de trabajo distinta de aquellas que le han sido asignadas, restricción de horario, etc...
- **Servicio de Archivos y Directorios**. Deberá ofrecer un alto nivel de fiabilidad y existencia de copias de respaldo y seguridad, ya que los usuarios acceden a programas y archivos que se encuentran en el servidor de archivos central.
- **Sistema Tolerante a Fallos.** Es un sistema para asegurar la supervivencia de la red en el caso de que fallen los componentes.
- **Optimización de Acceso al Disco (Disk Caching)**. Mejora el rendimiento del disco duro, utilizando una parte de la memoria del sistema como una zona en la que almacenar bloques de disco a los que se puede acceder de nuevo.

- **Sistema de Control de Transacciones.** Una transacción es un cambio en un registro o conjunto de registros de un archivo de base de datos. Esta utilidad se usa para evitar la falta de integridad en la base de datos, debida a fallos en una estación de trabajo o en el servidor.

- **Compartir Recursos.** De forma que cualquier usuario pueda utilizar teóricamente cualquier dispositivo desde un punto de la red.

- **Acceso Remoto.** Para poder conectarse con estaciones de trabajo y otras redes locales en puntos remotos.

- **Bridges (Puentes) y Routers.** Estos permiten que las redes se puedan interconectar con otras redes.

- **Gateways (Pasarelas).** Permiten conectar sistemas con distintos protocolos.

- **Adaptadores y Cables de Red.** Para admitir diversos tipos y marcas de placas de interfaz de red.

Para finalizar este punto decir que, durante la última década, el funcionamiento de las redes de ordenadores ha cambiado enormemente. Hace unos diez años éstas sólo se consideraban como herramientas extrañas para la investigación y sólo las utilizaban algunos especialistas. Hoy en día es más probable que los computadores, considerados en una escala que va desde los computadores personales a los supercomputadores, se encuentren como parte constitutiva de una red.

8.3 Razones para instalar una red de ordenadores.

Como ya sabemos, una red es un sistema de comunicación que conecta computadores. Las redes minimizan los problemas de distancia y comunicación y dan a los usuarios la posibilidad de acceder a información de cualquier punto de la red.

Las razones más usuales para instalar una red son:

- **Compartir programas y archivos.** Se pueden comprar versiones para red de muchos paquetes de software, con un ahorro bastante considerable si se compara con su coste al comprar copias con licencia individual. Además los programas y archivos de datos se almacenan en el servidor, de forma que cualquier usuario pueda acceder a ellos.

- **Compartir los recursos de la red.** Como impresoras, dispositivos de almacenamiento, etc. De esta forma nos evitamos que cada computador deba tener sus propios recursos.

- **Posibilidad de utilizar software de red.** El más utilizado es el software de gestión de bases de datos. También se está empezando a utilizar un nuevo tipo de software, el **groupware**, el cual está diseñado para grupos de usuarios que van a interaccionar entre sí en la red.

- **Correo Electrónico.** Se utiliza para enviar mensajes o documentos a usuarios o grupos de usuarios de la red. De este modo los usuarios pueden comunicarse más fácilmente entre

sí. El lugar de almacenamiento de los mensajes se denomina *buzón.* Algunos paquetes de correo electrónico y planificación (agendas) permiten llevar registro de los horarios y agenda de toda la empresa, permitiendo a los usuarios planificar sus actividades teniendo en cuenta las de los otros.

- **Creación de Grupos de Trabajo.** Los cuales tendrán unos derechos y unos privilegios especiales, que pueden que no sean accesibles a los restantes usuarios.

- **Gestión Centralizada.** La gestión de la red resulta fácil, ya que la mayoría de sus recursos se encuentran organizados alrededor del servidor.

- **Seguridad.** Los computadores sin disco pueden utilizarse para evitar que se extraigan o **pirateen** datos importantes mediante discos. Además los responsables pueden evitar que los usuarios trabajen fuera de unos directorios asignados, pudiendo aplicar también restricciones en la conexión.

- **Acceso a otros S.O.** De forma que se puedan conectar con otros sistemas que utilicen S.O. distintos y compartir recursos y archivos de idéntica forma que si fuera con un sistema que utilizara el mismo S.O.

- **Mejoras en la organización de la Empresa.** Las redes pueden suponer un cambio en la estructura administrativa de una organización, ya que permiten emular modos de trabajo en grupo, según los cuales, los departamentos sólo existen a nivel lógico dentro de una gestión computarizada y de una estructura de directorios.

Como resumen podemos decir que los objetivos principales de las redes son por un lado el de **compartir recursos** con el fin de hacer que todos los programas, datos y equipos estén disponibles para cualquiera de la red que así lo solicite, sin importar la localización física del recurso ni del usuario.

También decir que con las redes se proporciona al sistema una **alta fiabilidad**, al contar con fuentes alternativas de suministro. Además la presencia de múltiples CPUs significa que si una de ellas deja de funcionar, las otras pueden ser capaces de encargarse de su trabajo.

Otro objetivo es el **ahorro económico**. Los computadores pequeños tienen una mejor relación costo/rendimiento comparada con la ofrecida por las máquinas grandes. Éstas son, a grandes rasgos, diez veces más rápidas que el más rápido de los microprocesadores, pero su costo es miles de veces mayor.

8.4 Arquitectura de una red.

La arquitectura de una red define la estructura del sistema de cableado y de los computadores conectadas a él, además de las reglas utilizadas para transferir señales de unos a otros.

8.4.1 Topología de una red.

La podemos definir como la configuración formada por los nodos de la red y las interconexiones entre ellos, es decir, describe cómo va el cable desde un nodo a otro. Por tanto, dentro de la arquitectura de una red, se engloban los elementos que la componen, la forma en la que funcionalmente se unen a dicha red y su funcionamiento lógico de cara a la misma.

Figura 8.2: Topología lineal o en bus

Una red puede tener multitud de topologías distintas, en función de las necesidades a cubrir por la misma. Algunas topologías típicas son:

- **Topología lineal o en bus.** Se basa en una línea de comunicación que es compartida por todos y cada uno de los sistemas informáticos (llamados **nodos** y constituidos físicamente por equipos terminales o computadores) que se encuentran conectados a la red. Los distintos sistemas informáticos que están conectados al bus son recursos comunes que pueden ser compartidos por cualquier otro sistema en un momento determinado.

  Las **ventajas** que presenta el uso de esta topología son:

  - Medio de transmisión pasivo, es decir, un terminal en mal estado no influye sobre la transmisión de la información a lo largo del bus.
  - Fácil conexión a la red de nuevos dispositivos.
  - Facilidad de instalación.
  - Posibilidad de cubrir grandes distancias.

  Los **inconvenientes** principales son:

  - Facilidad para escuchar todos los mensajes de la red sin ser detectado.
  - Los terminales no inteligentes necesitan interfaces complejos.
  - El sistema no distribuye equitativamente los recursos a no ser que se instale un control central.

  En la figura 8.2 podemos ver el esquema de esta topología.

- **Topología en anillo.** Aquí todos los nodos están conectados formando un anillo entre ellos, es decir, un circuito cerrado en el que cada nodo está conectado solamente a dos nodos y nada más que a esos. Técnicamente se caracteriza porque aquí no existe ningún nodo que controle la red. La idea es distribuir la inteligencia entre varios dispositivos para lograr de esta manera una mayor seguridad frente a fallos, así como simplicidad y bajo coste.

  Entre las **ventajas** más destacadas podemos citar:

  - La no dependencia de un dispositivo central.
  - Facilidad de instalación y mantenimiento.

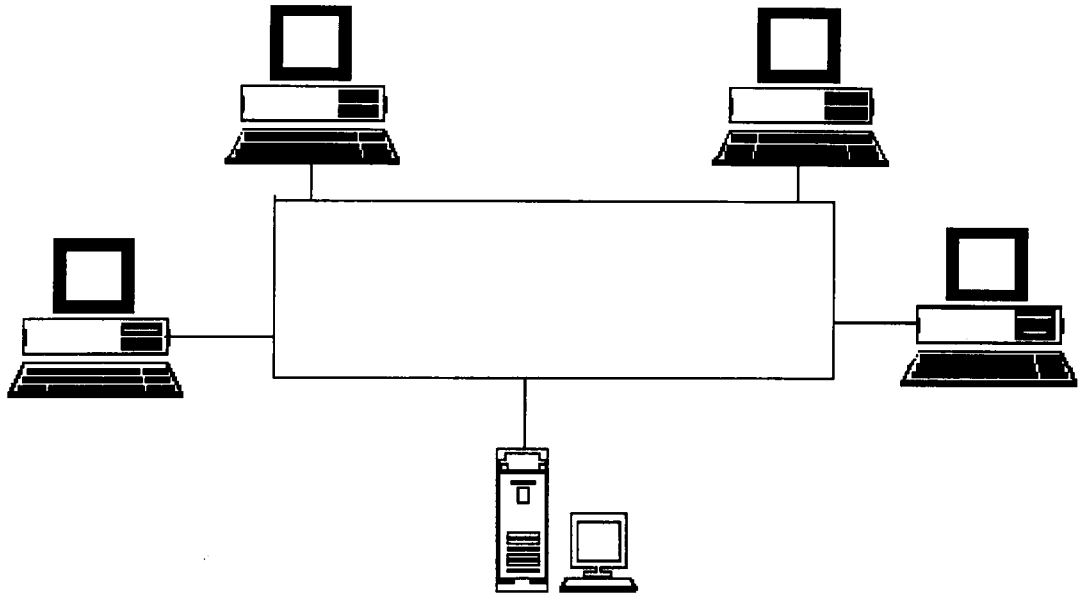


Figura 8.3: Topología en anillo

- Velocidades de transmisión muy elevadas.
- Coste no muy alto.

Entre sus principales **inconvenientes** están:

- Necesidad de contar en la práctica con un dispositivo monitor para monitorizar los datos erróneos o improcedentes.
- Es difícil ampliar el anillo una vez instalado.
- Se necesita interrumpir la transmisión cuando se conecta al anillo algún nuevo nodo.

La figura 8.3 muestra un esquema de esta topología.

- **Topología en estrella.** En esta todos los mensajes que circulan por la red pasan a través de un nodo central. En redes de datos, dicho nodo central suele desempeñar las tareas de nodo de conmutación, encargándose del encaminamiento de los distintos mensajes que llegan a él. La característica principal es que la práctica totalidad de la inteligencia de la red reside en el nodo central, siendo compartida por todos los dispositivos conectados con el mismo.

  La **ventaja** principal es que se proporciona una elevada seguridad en la transmisión y en el control de los errores, siendo fácil determinar la fuente de éstos.

  Dentro de los **inconvenientes** podemos decir que el principal de todos ellos y el más relevante es la posibilidad de avería en el nodo central, lo que provoca la paralización de la red.

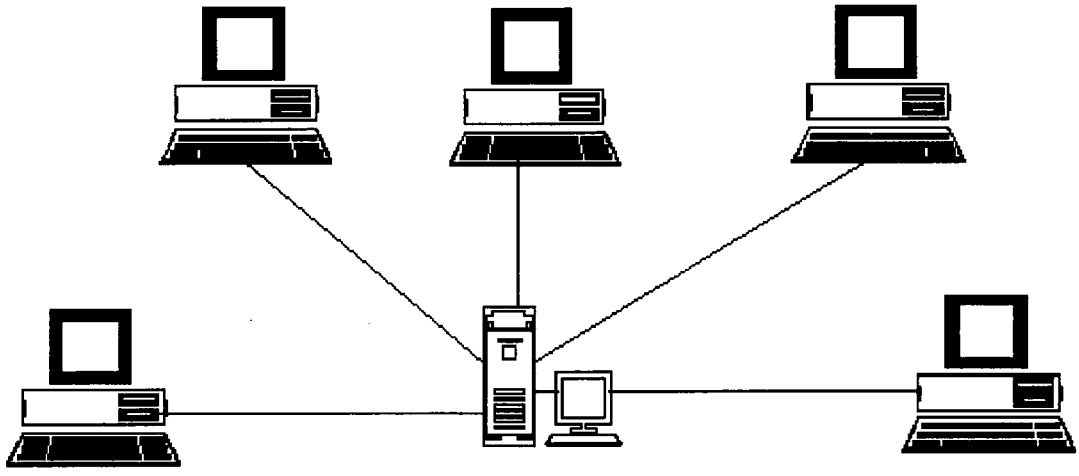


Figura 8.4: Topología en estrella

En la figura 8.4 se muestra el esquema de esta topología.

- **Otras Topologías:**
  - **Topología de Malla:** no posee ninguna estructura predefinida, sino que en cada caso concreto se adapta para obtener el mejor rendimiento.
  - **Topología de Interconexión Total:** aquí todos y cada uno de los nodos de la red se encuentran interconectados con todos los demás nodos que componen la red de comunicación de datos.
  - **Topología en Árbol:** consiste fundamentalmente en una serie de canales de distribución (buses) unidos entre si. Generalmente existe un bus de distribución que hace las veces de bus central al que se unen los demás buses, llamados complementarios.

Para finalizar, decir que la topología debe elegirse teniendo en cuenta algunos factores, como economía, eficacia de la interconexión entre todos los usuarios y los recursos, capacidad adecuada para entender las demandas de todos los usuarios, fiabilidad, tiempo de espera reducido, etc...

8.4.2 Métodos de acceso al cable.

Los métodos de acceso constituyen las técnicas que permiten a los usuarios finales acceder a los medios de transmisión de la red para depositar o capturar la información que necesitan, es decir, describen cómo accede un nodo o estación de trabajo al sistema de cableado. Una vez que la placa de interfaz de red consigue el acceso al cableado empieza a mandar paquetes de información (**bloques**).

Son muchos los métodos que pueden emplearse, pero nosotros sólo vamos a ver dos grandes grupos:

- **Métodos de Detección de Portadora o de Colisión.** Son empleados en los sistemas con topología lineal o en bus. Aquí la estación comprueba el estado del cable antes de transmitir, para ver si está siendo utilizado. De esta forma todos los nodos reciben la información, siendo éstos los que deben determinar si va dirigida a ellos o no. Si no lo fuera, el nodo devuelve la información recibida. Si dos nodos emiten al mismo tiempo se produce una colisión y se anulan ambas emisiones. En estos casos los nodos deben esperar un cierto tiempo aleatorio antes de volver a enviar la información.

  En este tipo de redes, a medida que aumenta el tráfico el rendimiento disminuye, debido a las colisiones que obligan a efectuar retransmisiones. El método de detección de portadora más usual es el **CSMA/CD** (**A**cceso **M**últiple por **D**etección de **P**ortadora / **D**etección de **C**olisiones). Dicho método es una mejora del **CSMA**, ya que el primero de ellos no sólo escucha el canal antes de empezar a transmitir (para ver si está libre o no), sino que continúa haciéndolo mientras transmite. Así cuando un nodo emisor escucha una señal que no se corresponde con la que él mismo está transmitiendo (se ha producido una colisión) cesa inmediatamente de transmitir, con lo que se ahorra un tiempo de utilización del canal de transmisión.

- **Métodos de Paso de Testigo.** Son empleados en los sistemas con topología en anillo o en estrella. Aquí se define una secuencia específica de caracteres (**testigo o token**). Cuando una estación de trabajo tiene información a transmitir debe esperar a que esté libre el testigo y apoderarse de él. De esta forma ya puede mandar la información. Una vez efectuada la transmisión libera el testigo. Así se evita que dos máquinas puedan utilizar simultáneamente el cable. Este sistema de paso de testigo envía los paquetes de una forma ordenada. Cada estación de trabajo examina la dirección del paquete para determinar si está dirigido a ella. Si no lo estuviera pasa el paquete a su estación vecina.

8.4.3 Protocolos de comunicaciones.

Son las reglas y procedimientos utilizados para establecer la comunicación entre los nodos que disponen de acceso a la red. En los protocolos se definen distintos niveles de comunicación. Las reglas de nivel más alto definen cómo se comunican las aplicaciones, mientras que las de nivel más bajo definen cómo se transmiten las señales por el cable. Una vez definidos y publicados los protocolos, los fabricantes pueden diseñar y producir productos para red que funcionen en sistemas con elementos de distintos fabricantes.

8.5 Cobertura de las redes.

En el mundo existen redes de todos los tamaños. La simple conexión de dos computadores a una sola impresora, utilizando un sistema de comunicación, técnicamente ya es una red. Pero la mayor parte de las redes están formadas por un mayor número de elementos y recursos compartidos. Dependiendo de la localización de éstos podemos hacer la siguiente clasificación.

- **Red de Área Local (LAN).** Es una red pequeña (de 3 a 50 nodos), normalmente localizada en un solo edificio o grupo de edificios pertenecientes a una organización.

- **Redes Interconectadas (Internetwork)**. Se pueden conectar dos o más redes locales para formar un sistema en red que cubra una mayor superficie. Normalmente suelen estar en las grandes empresas, en las que cada departamento tendría su propia red; estando estas redes interconectadas entre sí. Otras veces las redes más grandes se encuentran divididas en varios segmentos más pequeños con el fin de optimizar su rendimiento y su gestión.

- **Red Metropolitana (MAN)**. Se trata de un conjunto de redes interconectadas dentro de un área específica, como un campus universitario, un polígono industrial o una ciudad entera. Estas redes utilizan unas líneas de transmisión básicas de alta velocidad.

- **Red de Gran Alcance (WAN)**. Se trata de una red que cubre diversos países o incluso el mundo entero. Para ello se utilizan las líneas telefónicas o incluso los satélites de comunicaciones. Suelen utilizarse por las grandes empresas que poseen oficinas en todo el mundo, las cuales conectan sus redes de área local en una red de gran alcance. Por lo general, este tipo de conexiones son más lentas que las de las redes locales, pero también el tráfico que circula por ellas suele ser más reducido.

8.6 Estándares en las comunicaciones.

Veremos la división en **capas o niveles** que se puede hacer del **emisor** (lugar de donde parte el mensaje que se va a transmitir) y del **receptor** (destino final del mensaje transmitido). Dicha división ayuda a comprender más fácilmente el proceso de comunicación entre ambos.

Hace tan solo unos años el diseño de una red de ordenadores se consideraba como un arte de brujería. Cada fabricante de computadores tenía su propia arquitectura de red y en ningún caso existía compatibilidad.

Virtualmente, la industria informática en su totalidad, ha acordado una serie de Normas Internacionales para describir la arquitectura de las redes. De esta forma la mayoría de las redes se organizan en una serie de capas o niveles, con objeto de reducir la complejidad de su diseño. Cada una de ellas se construye sobre su predecesora.

El número de capas, el contenido y función de cada una varía de una red a otra. Sin embargo, en cualquier red, el propósito de cada capa es el de ofrecer ciertos servicios a las capas superiores, liberándolas del conocimiento detallado sobre cómo se realizan dichos servicios.

La capa n de una máquina se comunica con la capa n de otra máquina. Las reglas y convenciones utilizadas en dicha comunicación se conocen con el nombre de **protocolo de la capa n**. A las entidades que forman las capas correspondientes en máquinas diferentes se les denominan **procesos pares**. Según esto, son los procesos pares los que se comunican mediante el uso del protocolo.

En realidad no existe una transferencia directa de datos desde la capa n de una máquina a la de la otra, sino que cada capa pasa la información de datos y control a la capa inmediatamente inferior y así sucesivamente hasta que se alcanza la capa localizada en la parte más baja de la estructura. Debajo de esta capa está el medio físico, que es realmente por donde se transmite la información, es decir, a través del cual se realiza la comunicación real.

Entre cada par de capas adyacentes hay un **interfaz**, el cual define los servicios y operaciones primitivas que la capa inferior ofrece a la superior.

Al conjunto de capas y protocolos se le denomina **arquitectura de red**.

8.7 El modelo de referencia OSI.

Uno de los organismos involucrados en la creación de las normas es la **ISO** (Organización de eStándares Internacional). Esta es una organización no gubernamental, cuyos miembros representan a los cuerpos de estándares de la mayoría de los países del mundo. Desarrolla un gran número de normas relacionadas con muchas áreas.

Por lo que respecta a las comunicaciones, las normas se conocen como el **Modelo de Referencia OSI** (Interconexión de Sistemas Abiertos (Open)). En un futuro próximo, la mayoría de las otras arquitecturas de red desaparecerán y los computadores de un fabricante tendrán la capacidad de comunicarse sin mayor esfuerzo con los de cualquier otro fabricante, estimulando así, en forma más acentuada, el uso de las redes de ordenadores. Este concepto se conoce con el nombre de **interoperatividad**, es decir, la forma en la que el hardware y el software de distintos fabricantes pueden funcionar conjuntamente. A medida que crece la red es importante asegurarse de que todos los componentes que se añaden son compatibles.

El Modelo de Referencia OSI se basa en el principio enunciado por Julio Cesar, que dice: *Divide y Vencerás.* La idea consiste en diseñar redes como una secuencia de capas, cada una de ellas construida sobre la anterior. El proceso completo llega a ser más manejable, ya que se reduce el estudio de un todo al estudio de sus partes.

Este modelo tiene siete capas. Los principios aplicados para el establecimiento de este número fueron los siguientes:

- Una capa se creará en situaciones en donde se necesita un nivel diferente de abstracción.
- Cada capa deberá efectuar una función bien definida.
- La función que realizará cada capa deberá seleccionarse con la intención de definir protocolos normalizados internacionalmente.
- Los límites de las capas deberán seleccionarse teniendo en cuenta la minimización del flujo de información a través de los interfases.
- El número de capas deberá ser lo suficientemente pequeño para que funciones diferentes no tengan que ponerse juntas en la misma capa y, por otra parte, también para que su arquitectura no llegue a ser difícil de manejar.

Vamos a pasar ahora a ver, una por una, las **7 capas del modelo de referencia OSI**:

1. **La Capa Física.** Se ocupa de la transmisión de bits a lo largo de un canal de comunicación. Su diseño debe asegurar que cuando un extremo envía un bit con valor 1, éste se reciba exactamente como un bit con ese valor en el otro extremo y no como un bit de valor 0. También aquí se debe adaptar la señal a las necesidades del actual medio de comunicaciones; se debe decir todo lo concerniente a la velocidad de transmisión, cuántos segundos debe durar un bit, la posibilidad de realizar transmisiones bidireccionales de forma simultánea, la forma de establecer la conexión inicial y cómo interrumpirla cuando ambos extremos terminan su comunicación, el método de modulación usado, los conectores y cables usados y los niveles de voltaje usados para representar el 1 y el 0 lógicos.

2. **La Capa de Enlace.** Su tarea principal consiste en transformar un medio de transmisión común y corriente en una línea de transmisión sin errores para la capa de red.

Esta tarea se lleva a cabo al hacer que el emisor divida la entrada de datos en **tramas de datos**, las transmita de forma secuencial y posteriormente procese las **tramas de confirmación** devueltas por el receptor. Por tanto se deberá encargar de la creación o reconocimiento de los límites de la trama. Esto puede llevarse a cabo mediante la inclusión de un patrón de bits especial al inicio y al término de la trama. Si estos patrones de bits pueden aparecer entre los datos, deberá tenerse un cuidado especial para evitar cualquier confusión al respecto.

La trama puede destruirse por completo debido a una ráfaga de ruido en la línea, en cuyo caso el software de la capa de enlace perteneciente a la máquina emisora deberá retransmitirla. Sin embargo, múltiples transmisiones de la misma trama introducen la posibilidad de duplicar la misma (por ejemplo si se ha perdido la trama de confirmación enviada por el receptor).

La capa de enlace también ofrece diferentes clases de servicios a la capa de red, cada uno de ellos con distinta calidad.

Otro problema que surge aquí es el de cómo evitar que un transmisor muy rápido sature con datos a otro más lento. Para solucionar esto se deberá emplear un mecanismo de regulación de tráfico que permita que el transmisor conozca el espacio de memoria que en ese momento tiene el receptor.

3. **La Capa de Red.** Se ocupa del control de la operación de la subred. Un punto de suma importancia en su diseño es la determinación sobre cómo encaminar los paquetes del origen al destino. Las rutas podrían basarse en tablas estáticas que se encuentran "cableadas" en la red y que difícilmente podrían cambiarse. Otra posible forma de determinar las rutas sería al inicio de cada comunicación, por ejemplo en una sesión de terminal. Por último, podrían ser de tipo dinámico, determinándose de forma diferente para cada paquete, reflejando la carga real de la red.

   Si en un momento determinado hay muchos paquetes en la red ellos mismos se obstruirán mutuamente, dando lugar a un cuello de botella. El control de tal congestión dependerá también de la capa de red.

   El software de red deberá tener entre sus funciones alguna que contabilice el número de bits que se mandan, para tener un control a la hora de pagar por el uso de la red. Cuando un paquete cruza una frontera nacional, con precios distintos en cada lado, el cálculo de la cuenta puede llegar a complicarse.

   También pueden surgir otros problemas cuando un paquete tenga que desplazarse de una red a otra para llegar a su destino. El direccionamiento utilizado en la segunda red puede ser diferente al empleado en la primera. La segunda podría no aceptar el paquete en su totalidad, por ser demasiado grande, por tener protocolos diferentes, etc... La responsabilidad para resolver problemas de interconexión de redes homogéneas recaerá, en todos los casos, en la capa de red.

4. **La Capa de Transporte.** Su función principal consiste en aceptar los datos de la capa de sesión, dividirlos siempre que sea necesario en unidades más pequeñas, pasarlos a la capa de red y asegurar que todos ellos lleguen correctamente al otro extremo. Además todo este trabajo se debe hacer de una manera eficiente, de tal forma que aísle la capa de sesión de los cambios inevitables a los que está sujeta la tecnología del hardware.

En condiciones normales esta capa crea una conexión de red distinta para cada conexión de transporte solicitada por la capa de sesión. Si la conexión de transporte necesita un gran caudal, ésta podría crear múltiples conexiones de red, dividiendo los datos entre las conexiones de la red con objeto de mejorar dicho caudal. En definitiva, la capa de trasporte se necesita para hacer el trabajo de multiplexación transparente a la capa de sesión.

Aquí también se determinan qué tipos de servicio se deben dar a la capa de sesión y, en último término, a los usuarios de la red. El tipo de servicio se determina cuando se establece la conexión.

La capa de transporte es una capa del tipo origen-destino o extremo a extremo. Es decir, un programa en la máquina origen mantiene una comunicación con otro programa parecido que se encuentra en la máquina destino, utilizando las cabeceras de los mensajes y los mensajes de control. Los protocolos de las capas inferiores son entre cada máquina y su vecino inmediato y no entre las máquinas origen y destino.

Algunos hosts son multiproceso, lo cual implica que múltiples conexiones estarán entrando y saliendo en cada uno de ellos. Se necesitará, por tanto, alguna forma de decir qué mensaje pertenece a qué conexión. La cabecera de transporte es un lugar donde se puede colocar dicha información.

Aparte de todo esto, la capa de transporte debe ocuparse del establecimiento y liberación de conexiones a través de la red. También debe existir algún tipo de mecanismo para regular el flujo de información, de manera que un host muy rápido no pueda desbordar a otro más lento.

5. **La Capa de Sesión.** Permite que los usuarios de diferentes máquinas puedan establecer *sesiones* entre ellos. Una sesión puede permitir al usuario acceder a un sistema de tiempo compartido a distancia o transferir un fichero entre dos máquinas.

   Uno de los servicios de la capa de sesión consiste en gestionar el control del diálogo. Las sesiones permiten que el tráfico vaya en ambas direcciones al mismo tiempo, o bien, en una sola dirección en un instante dado.

   La administración del testigo es otro de los servicios relacionados con la capa de sesión. Para el caso de algunos protocolos resulta especial que ambos lados no traten de realizar la misma operación en el mismo instante. Para manejar estas actividades la capa de sesión proporciona testigos que pueden ser intercambiados. Sólo el extremo con el testigo puede realizar la operación crítica.

   Otro de los servicios de esta capa es el de la sincronización. Para ello proporciona una forma de insertar puntos de verificación en el flujo de datos, con objeto de que, después de cada caída, solamente tengan que repetirse los datos que se encuentren después del último punto de verificación.

6. **La Capa de Presentación.** Realiza ciertas funciones que se necesitan bastante a menudo. En particular y, a diferencia de las capas inferiores que únicamente están interesadas en el paso correcto de los mensajes de un lugar a otro, la capa de presentación se ocupa de los aspectos de sintaxis y semántica de la información que se transmite.

   Un servicio típico de esta capa es el relacionado con la **codificación de datos** conforme a lo acordado previamente. Los computadores pueden tener diferentes códigos

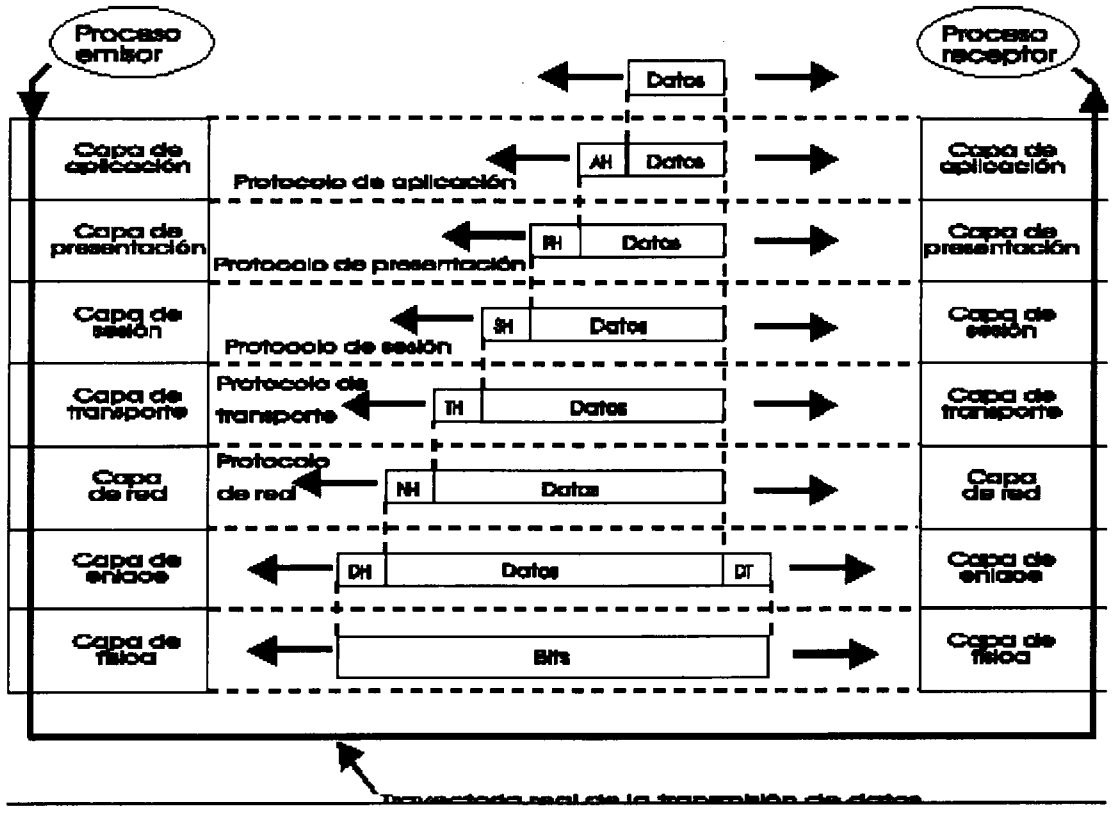


Figura 8.5: Capas del modelo OSI

para representar los caracteres, los enteros, etc... Para posibilitar la comunicación de computadores con diferentes representaciones, la estructura de los datos que se va a intercambiar puede definirse en forma abstracta, junto con una norma de codificación. El trabajo de manejar estas estructuras de datos abstractas y la conversión de la representación utilizada en el interior del computador a la representación normal de la red, se lleva a cabo a través de la capa de presentación.

Ésta también está relacionada con otros aspectos de representación de la información. Por ejemplo la compresión de datos se puede utilizar aquí para reducir el número de bits que tienen que transmitirse. El concepto de criptografía se necesita utilizar frecuentemente por razones de privacidad y de autentificación.

7. **La Capa de Aplicación.** Es la parte de los mecanismos de comunicación. Contiene una gran cantidad de protocolos que se necesitan frecuentemente. Otras funciones de esta capa son la de transferencia de archivos, el correo electrónico, la entrada de trabajo a distancia, el servicio de directorio y otros servicios de propósito general y específico.

En la figura 8.5 se muestra un dibujo de las diferentes capas del modelo OSI.

8.8 Canales físicos de comunicación.

Aquí veremos los canales físicos que se pueden usar para el transporte físico de la información. Después se verá cómo una señal eléctrica que lleva la información desde un punto a otro puede ser modificada para adaptarse al canal.

Como es lógico, se pueden conectar dos dispositivos sin más que establecer un enlace físico entre ellos (cable, fibra óptica,...). Igualmente es posible conectarlos usando ondas de radio, microondas o vía satélite. En estos casos no existe dicho enlace físico.

Cuando se usa un cable, por ejemplo, puede ser que se necesiten amplificadores o repetidores para poder transmitir la señal. El receptor final debe ser capaz de reconocer la información que le llega a través de la señal, es decir, habrá de procesar la señal.

Es posible usar un canal de comunicaciones simple para conectar varias fuentes y destinos de información. Este proceso se denomina multiplexación. Existen varios métodos para dividir la capacidad disponible del canal entre las diferentes fuentes.

Otros aspectos importantes a tratar son la distancia entre los dos sistemas, la cantidad de datos que van a ser enviados entre ellos, la velocidad mínima aceptable en la transferencia de los datos y las características del entorno que hay entre los dos sistemas.

Otros factores, como la necesidad de transferir los datos en una sola o en dos direcciones (camino bidireccional), pueden influir en la elección del canal de comunicaciones.

8.8.1 Tipos de canales.

Dentro de los computadores, la mayor parte de la información se representa, en bytes (grupo de 8 bits). Otros lo hacen usando grupos de 16 ó 32 bits. Dentro del computador esta información se suele transferir de unos lugares a otros en **paralelo**, usando una línea de transmisión o cable para cada bit. Para llevar a cabo esta tarea es necesario el uso, además de las líneas de datos, de unas líneas de direcciones y de unas líneas de control.

Para comunicar el computador con el exterior no es posible hacer todo esto. Como es de suponer, la razón principal es el gran tamaño y costo en cables que supondría tener una línea para cada bit de información (además de las de control y direcciones) entre dos o más computadores. Por estas razones, lo más usual es convertir dichos datos a un formato **serie** y transmitirlos por un canal de comunicaciones simple, bit a bit. Debido a esto, la velocidad de transferencia es mucho menor, pero es necesario hacerlo por el ahorro que supone.

8.8.2 Dirección del flujo de datos.

Un enlace que es usado en una sola dirección se denomina **unidireccional**. Si el flujo de información puede ser en las dos direcciones, pero en un momento determinado sólo se puede estar usando en una determinada, el enlace se denomina **half duplex**. Si el flujo de información puede ser en ambas direcciones al mismo tiempo, el enlace se denomina **full duplex**. Esto último se consigue teniendo dos enlaces físicos o compartiendo el espacio del enlace.

8.8.3 Características de los canales de comunicación.

Como ya sabemos, la información puede transmitirse por medio de cables. Este proceso tiene lugar cuando se varían algunas de sus propiedades físicas, como su voltaje o la corriente que circula por él. Al representar el valor de este voltaje o corriente únicamente como una función

del tiempo f(t), entonces se puede modelar el comportamiento de la señal y analizarse en forma matemática.

A principios del siglo XIX, el gran matemático francés Jean **Fourier** demostró que cualquier función que se comporte de forma razonablemente periódica puede construirse mediante la suma de un número posiblemente infinito de senos y cosenos:

$$g(t)= \frac{1}{2} * c + \sum a_n \, sen(2\pi nft) + \sum b_n \, cos(2\pi nft)$$

en donde $f = \frac{1}{T}$ representa la frecuencia fundamental y a_n y b_n son las amplitudes correspondientes al seno y coseno de los armónicos n-ésimos. A esta descomposición se le conoce como **Serie de Fourier**.

Una señal, si es clasificada como analógica o como digital, puede ser descrita en términos del *dominio del tiempo* o del *dominio de la frecuencia.*

En el **dominio del tiempo** su representación es un gráfico de cómo la señal varía en el tiempo. Esto puede ser obtenido con un simple osciloscopio.

En el **dominio de la frecuencia** la representación muestra cómo la señal se crea variando las frecuencias.

Ambas representaciones son equivalentes. Para pasar de una a otra se usa la *Transformada de Fourier*, bien la *directa* para pasar del dominio del tiempo al de la frecuencia; o bien la *inversa* para el caso contrario. Las expresiones matemáticas de éstas son:

Directa: $X(f)=\int x(t)*e^{-j2\pi ft} \, dt$
Inversa: $x(t)=\int X(f)*e^{j2\pi ft} \, df$

La representación de una señal en el dominio de la frecuencia es importante a la hora de ver los efectos que se producen en el canal físico de transmisión cuando la señal pasa por él, ya que las características del canal usualmente se dan en términos de parámetros dependientes de la frecuencia. La frecuencia contenida en un tren de impulsos es muy importante en la transmisión digital para saber cómo la señal afectará al canal. La frecuencia contenida en una señal particular es comúnmente conocida con el nombre de **espectro en frecuencia.**

8.8.3.1 ¿Cómo afecta la transmisión a la señal?.

Ningún medio de transmisión puede efectuar la transmisión de señales sin dejar de perder potencia durante la realización de este proceso. Si todas las componentes de Fourier fueran igualmente disminuidas, la señal resultante se reduciría sólo en amplitud, pero no sufriría distorsión alguna. Desafortunadamente cualquier medio degrada varias componentes de Fourier en forma diferente, por lo que se produce distorsión. En general, las amplitudes se transmiten sin degradación de frecuencias en una escala que va desde 0 hasta **fc** (**frecuencia de corte** medida en Hercios o ciclos/seg), observándose que todas las frecuencias que caen por arriba de esta frecuencia de corte son fuertemente atenuadas.

De aquí obtenemos la definición de **Ancho de Banda, B_o**, de un canal, que es el rango de frecuencias que pueden pasar por él sin que se produzca atenuación en el nivel de potencia de la señal.

En otros casos este efecto se introduce intensionalmente en el circuito mediante un filtro, cuyo objetivo consiste en limitar el ancho de banda que se encuentra disponible para cada usuario.

El tiempo T que se necesita para transmitir un carácter depende del método de codificación usado y de la velocidad de la señal, definida como el número de veces que la señal cambia de valor en un segundo. El número de cambios por segundo se mide en **baudios.**

En 1924, H. Nyquist descubrió que existía una máxima capacidad de transferencia de datos por un canal. De esto derivó una ecuación que expresaba la velocidad máxima de datos a través de un canal sin ruido, con ancho de banda finito. En 1948, Claude Shannon llevó a cabo un trabajo más extenso sobre lo desarrollado por Nyquist y lo amplió para el caso de un canal sujeto a ruido aleatorio.

Nyquist demostró que si una señal arbitraria se hace pasar a través de un filtro paso-baja, con un ancho de banda H, la señal filtrada puede reconstruirse por completo mediante la obtención simple y sencilla de 2H muestras por segundo. El llevar a cabo un muestreo de la línea a una frecuencia superior de 2H no tiene ningún sentido porque las componentes de frecuencias más altas de dicho muestreo no pueden recuperarse, pues han sido filtrados. Si la señal consta de V niveles discretos, el teorema de Nyquist establece que:

$$\text{La velocidad máxima de datos} = 2H * \log_2 V \text{ bits/seg.}$$

Por ejemplo un canal sin ruido de 3 KHz no puede transmitir señales binarias a una velocidad que exceda los 6000 bps.

Hasta aquí sólo se han considerado canales sin ruido, de tal manera que si estuviera presente un ruido aleatorio la situación se llegaría a deteriorar rápidamente. La cantidad de ruido se mide por la relación que existe entre la potencia de la señal y la potencia de ruido, a la cual se la conoce con el nombre de **relación señal-ruido.** Si se denota por S a la potencia de la señal y por N (Noise) a la potencia del ruido, la relación señal- ruido es **S/N.** Pero este valor no se suele usar mucho, sino que lo que se indica es 10 $\log_{10}$ S/N, cuya unidad son los decibelios.

El resultado más importante del teorema de Shannon establece que la **velocidad máxima** de datos sobre un canal ruidoso, cuyo ancho de banda es H Hz y cuya relación señal-ruido es S/N, está dada por:

$$\text{número máximo de bits/seg} = H * \log_2 (1 + \frac{S}{N})$$

El resultado del teorema de Shannon se demostró mediante el uso de la teoría de la información y tiene una validez muy general.

8.8.4 Medios de transmisión.

Como ya sabemos, el propósito de la capa física consiste en transportar el flujo original de bits de una máquina a otra. Normalmente se usan varios medios físicos para realizar una transmisión. La forma más simple de conexión entre dos puntos es mediante un cable. Vamos a ver a continuación los medios de transmisión más usados.

8.8.4.1 Medio magnético.

Una de las formas más comunes para el transporte de datos de un computador a otro consiste en grabar dicha información sobre una cinta magnética o en discos flexibles y transportar físicamente la cinta o los discos hasta la máquina destino, para que después pueda leer ésta la información. Este método, aunque no tan sofisticado como aquéllos en los que se usan satélites

de comunicación geosíncronos, es bastante efectivo en coste, en especial en los casos en los que se necesitan anchos de banda grandes o en donde el coste por bit transportado representa un factor clave. Problema: el transporte o desplazamiento físico desde el emisor hasta el receptor.

8.8.4.2 Par trenzado.

Aunque las características del ancho de banda de una cinta magnética sean excelentes, las características de retardo son muy malas: su tiempo de transmisión se mide en minutos, en horas o incluso en días y no en milisegundos. En muchas aplicaciones resulta necesario tener una conexión en línea. El medio de transmisión más antiguo, y todavía el más utilizado, es el **par trenzado**.

Éste consiste en dos alambres de cobre aislados, en general de 1 mm de espesor. Los alambres se entrelazan en forma helicoidal, como en una molécula de ADN. La forma trenzada del cable se utiliza para reducir la interferencia eléctrica con respecto a los pares cercanos que se encuentran a su alrededor.

La aplicación más común del par trenzado es el sistema telefónico. La distancia que se puede recorrer con estos cables es de varios kilómetros sin necesidad de amplificar las señales, pero sí es necesario incluir repetidores en distancias más largas. Cuando hay muchos pares trenzados colocados paralelamente que recorren distancias considerables, éstos se agrupan y se cubren con una malla protectora. Dentro de estos agrupamientos los pares podrían sufrir interferencias mutuas si no estuviesen entrelazados.

Los pares trenzados se pueden usar tanto para transmisión analógica como digital y su ancho de banda depende del calibre del alambre y de la distancia que recorre. En muchos casos pueden obtenerse transmisiones de varios megabits/seg en distancias de pocos kilómetros.

Debido a su adecuado comportamiento, bajo costo y a la gran modularidad que se consigue con ellos, los pares trenzados se utilizan ampliamente y es probable que su presencia permanezca por muchos años.

8.8.4.3 Cable coaxial.

Hay dos tipos de cable coaxial que se utilizan con frecuencia. Uno de ellos es el cable de 50 ohms, que se usa en la transmisión digital. El otro tipo, el de 75 ohms, se emplea en la transmisión analógica.

Un **cable coaxial de banda base (50 ohms)** consta de un alambre de cobre duro en su parte central, constituyendo el núcleo, el cual se encuentra rodeado por un material aislante. Este material está rodeado a su vez por un conductor cilíndrico que frecuentemente se presenta como una malla de tejido trenzado. El conductor externo está cubierto por una capa de plástico protectora.

La construcción del cable coaxial produce una buena combinación de un gran ancho de banda y una excelente inmunidad al ruido. El ancho de banda que se puede obtener depende de la longitud del cable. Cuanto más largo sea el cable la velocidad de transmisión será menor. **Cable coaxial de banda ancha (75 ohms)**. Para transmitir señales digitales en una red analógica cada interfaz debe tener un dispositivo electrónico que convierta en señal analógica el flujo de bits de envío y otro para convertir la señal analógica que llega en flujo de bits. Dependiendo del tipo (y precio) de estos dispositivos electrónicos, 1bps puede llegar a ocupar un ancho de banda que va desde 1 a 4 Hz.

Normalmente los sistemas en banda ancha se dividen en varios canales, empleándose cada uno de ellos para cosas distintas.

Una diferencia clave entre los dos sistemas es que los de banda ancha necesitan amplificadores que refuercen la señal en forma periódica. Estos amplificadores sólo pueden transmitir las señales en una dirección. Para solucionar este problema se han desarrollado dos tipos de sistemas de banda ancha: el de cable dual y el de cable sencillo.

8.8.4.4 Fibras ópticas.

Los desarrollos recientes en el campo de la óptica han hecho posible la transmisión de la información mediante pulsos de luz. Un pulso de luz puede utilizarse para indicar un valor de 1; la ausencia de un pulso indicará la existencia de un bit de valor 0. La luz visible tiene una frecuencia de alrededor de 108 MHz, por lo que el ancho de banda de un sistema de transmisión óptico presenta un potencial enorme.

Un **sistema de transmisión óptica** tiene tres componentes: el medio de transmisión, la fuente de luz y el detector.

El **medio de transmisión** es una **fibra ultradelgada** de vidrio o silicio fundido.

La **fuente de luz** puede ser un LED (Diodo Emisor de Luz) o un diodo láser; cualquiera de los dos emite pulsos de luz cuando se les aplica una corriente eléctrica.

El **detector** es un fotodiodo que genera un pulso eléctrico en el momento que recibe un rayo de luz.

Al colocar un LED o un diodo láser en el extremo de una fibra óptica y un fotodiodo en el otro, se tiene una transmisión de datos unidireccional, la cual acepta una señal eléctrica, la convierte y la transmite por medio de pulsos de luz. Después reconvierte la salida en una señal eléctrica en el extremo receptor.

Este sistema tendría fugas de luz y prácticamente sería de muy poco uso si no existiera un interesante principio de la física: *cuando un rayo de luz pasa de un medio a otro, por ejemplo del silicio fundido al aire, se refracta en la frontera de ambos.* Es decir, un rayo incide con un ángulo α_1, saliendo formando un ángulo β_1. La cantidad de refracción dependerá de los índices de refracción de los medios. Para ángulos de incidencia que se encuentran por encima de un valor crítico, la luz se refracta y regresa al silicio; nada de ella se escapa al aire. Así el rayo de luz que incida por encima del mencionado ángulo crítico queda atrapado en el interior de la fibra y puede propagarse a lo largo de varios kilómetros sin tener, virtualmente, ninguna pérdida. A esta situación se la conoce con el nombre de **fibra multimodo**.

Sin embargo, si el diámetro de la fibra se reduce al valor de la longitud de onda de la luz, la fibra actúa como una guía de ondas y la luz se propagará en línea recta, sin rebotar, produciendo así una **fibra monomodo**.

8.8.4.4.1 Comparación entre fibra óptica y cable coaxial. Las fibras proporcionan un ancho de banda extremadamente grande y tienen una pérdida de potencia muy pequeña, razón por la que se emplean para distancias muy largas entre repetidores. Las fibras no se ven afectadas por alteraciones de voltaje o corriente en las líneas de potencia, por interferencia electromagnética o por productos químicos corrosivos dispersos en el aire, de tal forma que pueden utilizarse en ambientes industriales expuestos a condiciones muy severas, en las que los cables serían muy inadecuados. Las fibras son también muy delgadas, lo que representa un factor positivo muy importante para las compañías que tienen una gran cantidad de cables.

Del lado negativo se encuentra el hecho de que hay poca familiaridad con la tecnología de las fibras ópticas y requiere una habilidad que los ingenieros en redes aún no tienen. Otra desventaja es que el empalme o unión de dos fibras es difícil y todavía más su derivación. Las fibras ópticas son inherentemente unidireccionales y el costo de los interfases es mucho mayor que el de las respectivas interfases de tipo eléctrico.

A pesar de todo esto último, las ventajas de las fibras ópticas son tantas que el empeño y trabajo que se está dando para mejorar su tecnología y reducir su costo es muy grande e importante.

8.8.4.5 Canales de Radio y Microondas.

Aunque muchos de los sistemas de comunicaciones de datos utilizan cables de cobre o fibras para realizar la transmisión, algunos simplemente emplean el aire para hacer esta tarea. La transmisión de datos por **rayos infrarrojos**, **LASER**, **microondas** o **radio** no necesita ningún medio físico. Cada una de estas técnicas se adapta a la perfección a ciertas aplicaciones.

La comunicación mediante láser o infrarrojos es por completo digital, altamente directiva y, en consecuencia, casi inmune a cualquier problema de derivación u obstrucción. Por otra parte, la lluvia y neblina pueden ocasionar interferencias en la comunicación, dependiendo de la longitud de onda elegida.

Su uso se ha extendido, como una alternativa del cable coaxial, en aplicaciones para comunicaciones de larga distancia.

La ventaja de las microondas es que la construcción de dos torres resulta, por lo general, más económico que abrir, por ejemplo, una zanja de 100 Km de longitud sobre la cual se pueda depositar el cable o la fibra y posteriormente volver a cubrirla.

Por otra parte, las señales de una antena pueden dividirse y propagarse siguiendo trayectorias, ligeramente diferentes, hacia la antena receptora. Cuando estas señales que se encuentran desfasadas se recombinan, puede haber interferencia entre ellas, de tal manera que se reduce la intensidad de la señal. La propagación de las microondas también se ve afectada por las tormentas y otros fenómenos atmosféricos.

La transmisión mediante microondas se lleva a cabo en una escala de frecuencias que va desde 2 a 40 GHz, correspondiendo a longitudes de onda de 15 y 0.75 cm. Estas frecuencias se han dividido en bandas de portadoras comunes para aplicaciones de tipo gubernamental, militar y otras.

8.8.4.6 Comunicación por satélites.

Aquí se tienen uno o más dispositivos **receptor-transmisor**, cada uno de los cuales escucha una parte del espectro, amplifica la señal de entrada y después la retransmite a otra frecuencia para evitar los efectos de interferencia con las señales de entrada. El flujo dirigido hacia abajo puede ser muy amplio y cubrir una parte significativa de la superficie de la Tierra, o bien, puede ser estrecho y cubrir un área de cientos de kilómetros de diámetro.

Según la ley de Kepler, el período orbital de un satélite varía de acuerdo con el radio de la órbita elevado a la potencia de 3/2. Por tanto, cerca de la superficie terrestre el período es aproximadamente de 90 minutos. Los satélites de comunicaciones ubicados a esta altura no son muy convenientes, debido a que se encuentran a la vista de las estaciones terrestres un intervalo de tiempo demasiado corto. Sin embargo, a una altura de unos 36.000 Km por

encima del Ecuador, el período del satélite es de 24 horas, por lo cual giraría a la misma velocidad con que lo hace la Tierra.

Con objeto de prevenir un posible caos en el cielo se han establecido acuerdos internacionales sobre quién puede hacer uso de qué ranuras orbitales y de qué frecuencias. Las bandas de 3'7 a 4'2 GHz y 5'925 a 6'425 GHz se han designado como frecuencias de telecomunicaciones vía satélite para flujos de información provenientes del satélite o hacia el satélite. En la actualidad se encuentran superpobladas.

Las bandas superiores siguientes que se encuentran disponibles para la telecomunicación son las de 12/14 GHz, las cuales no se encuentran todavía congestionadas. Problema: la lluvia, ya que el agua es un excelente absorbente de estas microondas tan cortas. Afortunadamente las tormentas más fuertes pueden localizarse con facilidad.

Las bandas de frecuencias de 20/30 GHz también se han reservado para las telecomunicaciones, pero el costo del equipo necesario para utilizarlas es todavía muy elevado.

8.9 Redes Públicas de Transmisión de Datos.

En España la Compañía Telefónica ofrece una serie de productos a sus clientes, los cuales suministran una serie de servicios de comunicación para la transmisión de voz/sonido, datos e imágenes.

1. La Red de Conmutación de Paquetes **IBERPAC** suministra servicios de transmisión de datos en forma de *paquetes* (partes en que se divide un mensaje y a los que se añade un número determinado de bits de control antes de enviarlo por la red a su destino) con el protocolo X.25, a velocidades de 2.400, 4.800, 9.600 bps y 64 Kbps. Además ofrece servicios adicionales de videotex (Ibertex), datáfono, etc.

2. La red **IBERMIC** suministra servicios de transmisión de datos punto a punto en cualquier protocolo y velocidades de entre 2 Mbps a 140 Mbps. Los circuitos no están asignados permanentemente a un usuario, sino que pueden ser reasignados según las necesidades del momento.

3. La red **UNO** es una red dedicada para clientes con necesidades de transmisión y conmutación de grandes masas de información entre sus distintos centros (se recomienda un mínimo de 500 accesos) que opera a una velocidad de 64 Kbps. Per

   mite la gestión de la red por parte del cliente. No es una red propiamente dicha, sino un servicio que se ofrece a las empresas a través de la red IBERPAC.

4. Otra posibilidad de comunicación entre los diferentes centros de una organización la constituyen los **enlaces punto a punto**. Son líneas entre extremos fijos que permiten la comunicación de volúmenes considerables, pero sin posibilid

   ad de acceso a las redes públicas (lo que puede ser aconsejable por seguridad). Se caracterizan por su disponibilidad permanente y exclusiva y un coste fijo en función de la velocidad y la distancia.

5. **Red Digital de Servicios Integrados. (RDSI)**

   La red telefónica se diseñó para *transmisiones analógicas* de voz. Hoy en día es inadecuada para las necesidades de las comunicaciones modernas (*digitales*), como por ejemplo la transmisión de datos, documentos, imágenes, etc. Es

to ha propiciado el desarrollo de redes especializadas que utilizan técnicas distintas de transmisión, ofreciendo así diferentes servicios de transmisión de información.

La RDSI (o en inglés ISDN) es una red normalizada de transmisión de datos y voz dotada de múltiples servicios en un acceso único. Además constituye el último eslabón en la evolución natural de la red telefónica básica para convertirse

integramente en digital, desde un extremo a otro de la transmisión.

La tradicional red conmutada analógica empezó su digitalización mediante la RDI (Red Digital Integrada), la cual abarcaba tanto el cableado como los nodos de la red telefónica. Con la red RDSI se consigue llegar digitalmente hasta el domicilio del

abonado, incluyendo su propio teléfono.

La red está normalizada por el CCITT (Comité Consultivo Internacional para Telegrafía y Telefonía), que es el máximo organismo internacional para la estandarización de las telecomunicaciones y por su homólogo europeo ETSI.

La RDSI que actualmente se puede contratar también se denomina **RDSI-BE** (Banda estrecha) por contraste con el futuro desarrollo de la **RDSI-BA** (Banda Ancha) mucho más rápida. La RDSI-BE está disponible con velocidades que van de 64 Kbps

a 2 Mbps. Hasta ahora la velocidad máxima de transmisión de datos era de 28.800 bps. El usuario debía convertir los datos digitales de su ordenador en una señal analógica capaz de circular por la línea telefónica conmutada, utilizando pa

ra ello un módem. Con la RDSI-BA, que aun está en desarrollo y se basa en la tecnología MTA (Método de Transferencia Asíncrono) se podrán alcanzar velocidades cercanas a 622 Mbps. Actualmente Europa y Estados Unidos están cableando nodos

experimentales de este tipo de red que realmente constituirá lo que se ha venido a llamar las *superautopistas de la información.*

Veamos cómo está formada esta red. La RDSI dispone de una serie de canales digitales para la transferencia de la información, llamados **canales de acceso**. Existen tres tipos de canales: B, D y H. El canal B es de 64 Kbps y transporta la informa

ción proveniente del terminal del usuario. El canal D es de 16 Kbps o 64 Kbps y se utiliza para la señalización y control de llamadas. Este canal podrá ser habilitado para transmitir información a baja velocidad vía Iberpac. El canal H permi

te la transferencia de información de usuario a velocidades superiores de 64 Kbps en los servicios portadores en modo circuito. El canal del tipo H_0 alcanza 384 Kbps, el H_1 es de 1.536 Kbps y el de tipo H_2 de 1.920 Kbps. De las combinaciones de

los canales de acceso se derivan los llamados **accesos de abonado**, que pueden ser:

- *Acceso básico*: está al alcance de cualquiera. Está formado por dos canales B (64 Kbps) y uno D (16 Kbps). Por tanto, el ancho de banda máximo que puede utilizarse es de 144 Mbps (64 * 2 + 16). En el local del usuario debe realizarse un a instalación a cuatro hilos, dos para transmisión y dos para recepción, configurando un bus de datos denominado *bus pasivo*, que permite la conexión de hasta 8 equipos terminales.
- *Acceso primario*: es mucho más caro y selectivo, pero proporciona 30 canales de 64 Kbps, con lo que se puede alcanzar un gran ancho de banda.

La característica fundamental de la RDSI es la integración de servicios que antes eran caros y se encontraban separados. Su ventaja inmediata es la compatibilidad con los servicios analógicos clásicos, con lo que se facilita su introducción en

el mercado telemático. De esta forma, la implementación de la RDSI puede hacerse en cualquier momento con una mínima inversión y disfrutando de múltiples ventajas de manera inmediata, dejando para el futuro la incorporación de otros servici

os digitales más sofisticados.

En el aspecto de aplicaciones está el autentico desafío de la revolución digital de la RDSI. Cualquier organización puede obtener considerables ventajas de la RDSI para mejorar procesos organizativos. El impedimento principal no es tanto el prec

io, sino más bien las resistencias culturales. Es difícil imaginar soluciones telemáticas cuando no se está acostumbrado a ello. Mediante la RDSI se pueden ofrecer servicios empresariales de teletrabajo, telecompra, telemarketing, teleocio, tele

información, etc..., que implican cambios sociales de gran alcance. Podemos decir que existen dos grandes opciones para abordar la implantación de sistemas telemáticos: la orientada a la mejora de procesos organizativos existentes y la dirigida a cr

ear servicios nuevos. En todo caso el objetivo final y factible de la RDSI es automatizar procesos inter-empresariales, es decir, generar una auténtica redefinición de los procesos de comunicación entre empresas.

Todos los sistemas telemáticos complejos pueden descomponerse en dos tipos simples de servicios, que son transferencia de información y servicios interactivos. Oportunamente combinados y bien implementados pueden conformar complejos procesos organizat

ivos.

Para finalizar decir que la RDSI es sólo una tecnología básica para el usuario. Todo está por empezar y hay mucho por hacer. La RDSI es la chispa que permitirá convertir en realidad el sueño de las autopistas de la información y el mito de

la aldea global: acceso digital a cualquier teléfono u ordenador. La RDSI significa la generalización y mejora de la transmisión de datos. En el momento en que la mayoría de las empresas trabajen en RDSI la sociedad irá modificando sus hábi

tos de trabajo, compra y ocio. La revolución digital nos atañe a todos. Habrá que regular las aplicaciones sociales de los servicios basados en RDSI, la seguridad de los datos, los derechos de privacidad e información.

8.10 La red INTERNET.

La red Internet es la red de ordenadores más famosa que jamás se haya construido, o mejor dicho se trata de una red que ha ido firmemente evolucionando a partir de la conexión de otras redes.

Esta red nació en Estados Unidos a principios de los años 80 con la red Arpanet. En sus orígenes interconectaba centros académicos y de investigación, pero progresivamente se fueron conexionando redes de otros ámbitos geográficos hasta alcanzar un ámbito mundial. Su crecimiento ha sido vertiginoso en los últimos años, alcanzando a más de 85 países con

conectividad total, más de 3 millones de computadores conectados y con un número de usuarios que se dice superior a los 30 millones, incrementándose a razón de 160.000 al mes.

Internet está ahí, lista para que nos conectemos a ella. Existen diversas formas de conexión, cada una de ellas con distintos costes, distinta infraestructura que organizar y distintos niveles de seguridad.

Tres son los problemas que tiene que superar aquel que decida integrarse en el mundo Internet: comprender su estructura, conectarse a ella y sacarle el máximo partido. Para tener un acceso completo a Internet se necesita tener instalado el **TCP/IP** (**T**ransmission **C**ontrol **P**rotocol / Internetwork **P**rotocol), que es el **protocolo de comunicaciones** de la red. Nació inicialmente en el mundo Unix, siendo su principal característica la de permitir la conexión de computadores de distinta naturaleza. A pesar de que no es un ejemplo de efectividad y rendimiento, tiene la gran ventaja de funcionar correctamente en un mundo heterogéneo, donde cada computador que forma parte de la red no siempre responde a un mismo fabricante o a unas características determinadas. Una vez dado este paso, se puede acceder a Internet vía un servidor de LAN que actúe como nodo de la red, o bien vía modem.

Actualmente, en España, existe una cierta dificultad para acceder a todos los servicios de Internet, ya que son muy escasas las organizaciones que, por una tarifa, ofrecen la conexión a esta red. Pero este problema se está empezando a solucionar y están surgiendo empresas que, por un bajo coste (precio de una llamada telefónica local), permiten una buena conexión. Pero todavía es en la

Universidad donde probablemente se tiene un acceso más fácil a los servicios completos de Internet.

En una red que ha crecido espontáneamente, sin existir jamás una planificación centralizada, el asegurar un mecanismo de direccionamiento en el que cada usuario posea una dirección única no parece ser una tarea fácil. Pero afortunadamente este problema se ha resuelto satisfactoriamente y cualquier persona que utiliza Internet tiene su propia dirección. El esquema de direccionamiento de Internet se denomina *Sistema de Nombres de Dominio* (D.N.S.) y se basa en la combinación de información conceptual y geográfica. El aspecto que toma una dirección típica es el siguiente:

identif_usuario@dominio

- **identif_usuario:** es el nombre del usuario que se conecta a la red. Si se trabaja con un sistema Unix el identificador de usuario será el nombre que se utiliza como "login" para la conexión.
- **@ (arroba):** se utiliza para separar ambas partes. No debe haber espacios en blanco entre estas partes.
- **dominio**: está a su ves dividido en **subdominios**, separados por puntos. La forma de entender un dominio es mirar a los subdominios de derecha a izquierda.
  - El subdominio más a la derecha (de primer nivel) es la especificación más general. Se le conoce con el nombre de dominio geográfico. Está formado por dos letras, las cuales representan la abreviación de un **país**. A veces tambi

    'en puede indicar el tipo de **organización**. Por ejemplo: **at**-Austría, **au**-Australia, **ca**-Canadá, **de**-Alemania, **dk**-Dinamarca, **es**-España, **fr**-Francia, **jp**-Japón, **uk**-Reino Unido, **com**-Orga

nización comercial, **edu**-Institución educativa, **gov**-Institución gubernamental, **int**-institución internacional, **mil**-Organización militar, **net**-Gestión de red, **org**-Organización no comercial.

- Los siguientes subdominios que van apareciendo hacia la izquierda van siendo más específicos e indican el tipo de **organización o institución** o el **nombre de la organización**. Por ejemplo: **uca**-Universidad de Cádiz, **u**
  gr-Universidad de Granada,
- El subdominio de más a la izquierda suele ser el nombre específico del **computador**. Por ejemplo: hercules, cain, goliat, nevada, lucano, apolo,...

Ejemplos de direcciones de ordenadores conectados a la red Internet son:

- hercules.uca.es
- goliat.ugr.es
- archie.sura.net
- nic.funet.fi
- music.tuwien.ac.at

Actualmente, y por cuestiones de seguridad para evitar posibles "pirateos" o "ataques", se tiende a normalizar las direcciones utilizando el siguiente formato:

nombre-usuario.apellido@organización.pais

Pero en realidad los ordenadores no trabajan con direcciones como las desritas, sino que las direcciones reales de Internet son números. A estas direcciones numéricas se les llama **Direcciones IP** (Internet Protocol). La dirección de cada nodo d

e la red es un número de 32 bits, presentado en forma de cuatro bytes separados por un punto. Los dos o tres primeros suelen indicar la subred en la que se encuentra el nodo. Por ejemplo, todos los nodos de la UCA tienen direcciones de la forma 150.214.

7x.xxx, donde 'x' representa un dígito. Estas direcciones las asigna el **NIC** (Network Information Center).

Para un conocimiento más detallado de las aplicaciones de Internet consúltese el capítulo "Software de Aplicación".

Apéndice A

Glosario de Términos Informáticos.

Ábaco Termino que designa varios tipos de instrumentos simples, con bolitas que se pueden desplazar por un hilo, que permiten efectuar operaciones aritméticas elementales de modo inmediato. Se le considera como la primera calculadora, y el primer ordenador.

Abortar Interrumpir de modo más o menos drástico un proceso sin esperar a que termine su ejecución normal. Puede producirse por un fallo en el programa o en el sistema, o manualmente si queremos terminar la ejecución de un programa pulsando simultáneamente Ctrl-Pausa o Ctrl-C. En muchos casos el programa en ejecución no responderá a estas pulsaciones, entonces, o lo dejamos continuar o *reseteamos*, que es como apagar el ordenador, darle a la tecla Reset o pulsar simultáneamente las teclas Ctrl-Alt-Supr. No se debe hacer nunca, y sobre todo si se esta escribiendo o leyendo en algún disco (podría estropearse). Se debe elegir la opción de salida del programa, y luego apagar alegremente el sistema. Esta es una acción muy usada cuando un ordenador se queda *colgado*, es decir se queda bloqueado sin dar ninguna salida ni esperar ninguna entrada. Una prueba de que el ordenador se queda *colgado*, y que NO está operando internamente es pulsar la tecla Bloq-Mayús o Bloq Num para activar las letras mayúsculas o el teclado numérico, si pulsando esas teclas se apagan y encienden las luces del teclado que indican tales acciones es señal de que NO está *colgado*. Si por el contrario el teclado ignora tales acciones, podemos suponer que sí esta bloqueado.

Acceso al Medio Los métodos de acceso al medio constituyen las técnicas que permiten a los ordenadores de los usuarios finales acceder a los medios de transmisión de la red para depositar o capturar la información que necesitan. El tipo de medio puede ser diverso, pero en general son cables, fibra óptica, aire... Ver Red de Ordenadores, Arquitectura de una Red, Topología de una Red.

Acceso Directo Técnica de almacenamiento y recuperación de la información en la que se accede a los datos directamente. El tiempo de acceso a un dato NO depende de su posición. Debe ser en dispositivos que lo permitan: Memoria principal (RAM-ROM), discos magnéticos... Ver Acceso Secuencial, Disco, Memoria, Tiempo de Acceso.

Acceso Secuencial Técnica de almacenamiento y recuperación de información donde se almacenan todos los datos secuencialmente uno detrás de otro. El tiempo de acceso a un dato depende de su posición, pues para leerlo hay que leer antes todos los anteriores. Típico acceso de dispositivos de cinta, DAC... Ver Acceso Directo.

Acumulador Registro de la CPU para datos. Ver Registro, CPU.

Administrador de una red Encargado de que no falle nada en la red o en el S.O. Se encarga de asignar privilegios, dar de alta o baja a usuarios... Ver Red de ordenadores, LAN.

Agenda Electrónica Programa informático para gestionar las citas y acciones a realizar en fechas concretas, con un calendario, y almacenar teléfonos y direcciones. Suelen incorporar reloj, alarma, mapas... Ver Aplicaciones.

Algebra de Boole Sistema algebraico, creado por George Boole, que permite procesar valores de verdad, con dos posibilidades (Verdadero y Falso, 1 y 0 respectivamente), a través de operaciones lógicas sobre afirmaciones. Las operaciones elementales son AND, OR y NOT. A partir de estos se pueden construir otros operadores como NAND, NOR, XOR...

Algoritmo Conjunto bien definido de instrucciones o pasos que deben seguirse para la resolución de un problema (Ej: una receta de cocina, o un programa informático). Debe tener una sucesión finita de pasos no ambiguas. Su nombre deriva del nombre del matemático Al Khwarizmi. Las estructuras de control básicas en un algoritmo para un programa informático son la Secuencia, la Selección y los Bucles. Ver Programa, Selección, Bucle, Secuencia.

ALU *Arithmetic-Logic Unit*: Unidad Aritmético-lógica, parte de la CPU que se encarga de las operaciones aritméticas (suma,resta, división,..) y lógicas (AND, OR, NOT, XOR...). Ver CPU, Procesador, Coprocesador, Algebra de Boole.

Analista de Sistemas Persona responsable de analizar los problemas y diseñar las soluciones en un Centro de Proceso de Datos (CPD).

Analógica Ver Señales.

Aplicaciones Programas de utilidades, usados para sacarle rendimiento a un ordenador. Por ejemplo, ver procesadores de texto, hojas de cálculo, DBMS, agenda electrónica, e-mail... Ver también Programa, Software, ISO 9000, CAD, CAE, CAM, Agenda electrónica, Correo electrónico, Hoja de Cálculo, Procesador de Textos, DBMS, Compresor, Compilador, Intérprete.

Árbol Estructura de datos jerárquica cuyas características son que cada elemento posee algún elemento por encima de él (padre) y cero o más elementos por debajo (hijos). El único elemento que no tiene padre, el primero, se llama raíz del árbol, y los elementos que no tienen hijos, los últimos, se llaman hojas. . Ver Estructura de Datos.

Archivo o Fichero Colección de información agrupada bajo un sólo nombre y almacenada en un soporte de información (discos...). Puede contener datos o programas. Ver Base de Datos, Backup, AUTOEXEC.BAT, CONFIG.SYS, Batch, Comando, Oculto, Ruta, Extensión, Soporte de Información, EOF, Directorio.

Arquitectura de una Red Define la estructura del sistema de cableado y de los ordenadores conectados a él, además de las reglas utilizadas para transferir señales de unos a otros. Ver Red de Ordenadores, Topología de una Red, Acceso al Medio.

ASCII Ver Códigos.

AUTOEXEC.BAT Archivo batch (de procesamiento por lotes), de un Sistema Operativo tipo DOS, en el que se introduce un lote de instrucciones (un conjunto ordenado) que se ejecuta automáticamente cada vez que se enciende el sistema. Ver batch, CONFIG.SYS, COMMAND.COM.

Backup Acción de hacer copias de seguridad de algún archivo/s, directorio, disco... Se llama archivo de Backup al archivo que contiene la información generada en el proceso de backup, esto es, archivos de copias de seguridad. Hacer copias de seguridad, por backup, no es una simple copia de los archivos en cuestión, sino que el programa para backup (`msbackup` en el DOS) comprime toda la información de todos los archivos a copiar en un único archivo (o más si no cabe en un disco) ocupando así menos memoria (menos discos) que si se copian los archivos normalmente (con `copy`). Por eso, el backup también se usa para llevar mucha información de un ordenador a otro, aunque existen programas compresores que puede ser que compriman la información mejor. Ver compresor.

Base de Datos Conjunto de datos (archivos) relacionados entre sí, y organizados para proporcionar una base para su utilización efectiva: Recuperación de datos, modificación, creación de informes, inserción, borrado... Suelen ser jerárquicas, en red o relacionales. Ver Árbol, Red, Relacional, ODBC, DBMS.

Batch Proceso de ejecución por lotes, ejecutando ordenadamente varios programas. En el DOS es un archivo de texto, con extensión .BAT, que contiene una serie ordenada de comandos del S.O. y programas. Al ejecutarlo, ejecuta en lote (en batch, uno tras otro) todos los comandos especificados, en el orden descrito. Ver Comando, DOS, AUTOEXEC.BAT.

Baudio Unidad de velocidad de transmisión de información a través de una línea de comunicaciones modulada (como la línea telefónica...). Es el número de cambios de estado de una señal analógica por segundo. Puede corresponder a 1 bit por segundo, pero pueden ser más. Ver Bps, Señales Digitales.

BBS Nombre del programa de Tom Jenning (1983), siglas de *Bulletin Board System*, traducido como Sistema de correo electrónico o Tablón de anuncios electrónico. Este programa fue la base sobre el que se construiría la red internacional de conexión de ordenadores FidoNet. Actualmente, BBS se usa para designar un sistema que ofrece servicios avanzados de mensajería, boletines, ficheros y conversaciones sobre distintos temas, esto es, son Bases de Datos on-line y para acceder a ellas lo único que hace falta es un ordenador, un modem y línea telefónica. Ver Fidonet, Modem, Correo electrónico, SysOp.

Bernoulli Discos magnéticos actualmente no muy comunes, basados en la teoría del físico suizo del siglo XVIII con igual nombre. Tienen ventajas frente a los discos duros, gran capacidad (150 MB o más), y son intercambiables, es decir, podemos utilizar tantos discos como queramos utilizando tan solo una unidad de escritura/lectura. Ver Soporte de Información, Disco.

Bifurcación Sinónimo de Selección (`if`). Ver Selección.

BIOS *Basic Input/Output System*: Sistema básico de entrada/salida. Microcódigo que reside en la ROM de los microordenadores y es responsable de la realización de las operaciones de entrada y salida. Normalmente, el BIOS es llamado por el S.O., pero también lo pueden llamar directamente las aplicaciones, mejorando el rendimiento, pero produciendo una pérdida en la portabilidad. Ver S.O., ROM.

Bit BInary digiT. Unidad mínima de información en informática. Para representarla se le hace corresponder un dígito del código binario: 0 (cero) o 1 (uno). Ver byte, Código binario.

Bloqueado, ordenador Ver Abortar.

Bpi *Bits per inch*: Bits por pulgada: Indica número de bits que se graban en una pulgada de disco. Mide la densidad de grabación. Se usa sobre todo en unidades de cinta magnética.

Bps Bits por segundo. Es el número de bits de datos transferidos en un segundo, en una línea de comunicaciones. Mide la velocidad de esa línea (Velocidad de Transferencia). Puede ser entre dos modems, en el cable de una LAN... Ver Bit, Baudio, LAN.

Bucle Parte de un programa que se ejecuta reiteradamente (cero o más veces). Puede repetirse un número determinado de veces, o hasta que (o mientras que) se cumpla una determinada condición (estructuras del tipo `while`, `for`, `do-while`...). Ver Bifurcación, Secuencia, Algoritmo, Programa.

Buffer Banco de memoria donde se almacena temporalmente la información para compensar la diferencia de velocidad de trabajo entre el procesador y los periféricos. Por ejemplo, al mandar algo a la impresora, en vez de acomodarse el procesador a la velocidad de la impresora, este manda todo el trabajo al buffer y la impresora imprime de ahí mientras el procesador puede hacer otras tareas. Ver periféricos, Memoria.

Bulletin Board System Ver BBS.

Bus Conjunto de cables, filamentos, hilos o pistas, por el que se transmiten bits de información. Ver Bps, PCMCIA.

Byte Unidad para medir cantidad de información o capacidad de almacenamiento de información. Es el número de bits necesarios para codificar (en binario) un carácter. Normalmente son 8 bits. Ver Carácter, Código binario, bit, Kbyte, Megabyte, Gigabyte y Terabyte.

C Lenguaje de programación muy extendido últimamente, de gran flexibilidad para programadores y muy portable entre distintos sistemas. Su flexibilidad y potencia lo convierten en el lenguaje ideal tanto para aplicaciones de alto nivel como para determinadas aplicaciones de más bajo nivel, como sistemas operativos, compiladores, programas de Inteligencia Artificial... Su nombre viene de que deriva de otro lenguaje llamado B. Una extensión de este lenguaje es el C++, que es ya un lenguaje dirigido a objetos. Ver Lenguaje de programación, Algoritmo, Programa, UNIX.

Caché Memoria tipo RAM, pero ultrarrápida (de tecnología estática) que se incorpora entre un dispositivo rápido (la CPU) y un dispositivo de almacenamiento más lento (disco duro, memoria principal...) para evitar que la CPU pierda tiempo esperando lecturas

y escrituras en esos dispositivos lentos. El paso de información (datos) del dispositivo lento a la caché se hace por bloques. La CPU accede sólo a la caché y cuando necesita un dato que está en otro bloque, este se carga en caché. Ver Memoria, CPU, Registros.

CAD *Computer Aided Design*: Diseño asistido por Computadora. Programas que ayudan al hombre a diseñar objetos: coches, aviones, edificios... Ver CAE, CAM, Aplicaciones.

CAE *Computer Aided Engineering*: Ingeniería asistida por Computadora. Comprende el análisis y simulación con objetos de ingeniería. A veces, CAE, también son las siglas de *Computer Aided Education*: Educación asistida por Ordenador. Ver CAD, CAM, Aplicaciones.

CAM *Computer Aided Manufacturing*: Fabricación asistida por ordenador en el que la máquina controla la cadena de fabricación de un producto. Esta muy relacionado con el CAD y CAE. Ver CAD, CAE, Aplicaciones.

Campo Ver Registro (Record).

Carácter Cada símbolo que se usa al trabajar con el ordenador: letras, números, símbolos de puntuación (punto, coma, punto y coma, paréntesis, interrogaciones, exclamaciones...), símbolos de operaciones (+, -, /, *), caracteres de control (fin de linea –EOL–, fin de fichero –EOF–...). Ver Byte, EOF, Variable, Constante, Tabulador.

Caretos/caritas Ver Smiles.

CCITT Comité Consultivo Internacional de Telegrafía y Telefonía. Es la entidad encargada de establecer normas o recomendaciones en el área de las comunicaciones en general y para modem en particular. Consultar la tabla 2.8. Ver Modem, Baudio, Bps.

CD-ROM Compact Disk - Read Only Memory: Disco compacto de sólo lectura basado en tecnología óptica (no magnética), con rayo láser. Ver disco, pista.

CD-WORM Compact Disk - Write Once Read More: Disco compacto de tecnología óptica en el que se puede escribir una vez al principio, y leer muchísimas veces. Ver CD-ROM.

Circuito Impreso Placa de material aislante en la que se trazan, o graban, circuitos electrónicos.

Circuito Integrado (Chip) Dispositivo compuesto por transistores y otros elementos, construido sobre una pastilla de silicio y generalmente encapsulados en un aislante negro con pines metálicos (patas conductoras) que constituyen sus entradas y salidas.

Cluster En un disco, unidad física de almacenamiento que incluye uno o más sectores dependiendo de la densidad de grabación. Ver Sector, Pista.

Coaxial Tipo de cable de transmisión de datos y señales formado por un hilo central, rodeado de un aislante, rodeado a su vez de una malla de apantallamiento formada por muchos hilos conductores muy finos, que se rodean a su vez de un aislante externo.

Codificación Proceso por el cual la información es convertida de un formato original a otro con algún fin concreto, como transmitirla o procesarla. También se usa para indicar el proceso por el cual se pasa un algoritmo a un programa en algún lenguaje determinado.

Ver Lenguaje de programación, Código Fuente, Código Objeto, Decodificación, Modular, Demodular.

Código ASCII *American Standard Code for Information Interchange*: Conjunto estándar de codificación de caracteres reconocidos por el ordenador. Representa caracteres en un código binario. El número de caracteres ASCII es de 256 (8 bit por carácter, 2^8=256). En algunos sistemas de comunicación (BBS, FidoNet...) sólo se aceptan los 127 primeros, esto es, el ASCII con 7 bits. Ver carácter.

Código Binario Código con dos símbolos (0 y 1). Es un código de numeración en base 2 en el que sólo se usan esos dos símbolos. (Normalmente los humanos usamos una numeración en base 10). Ver Código máquina, Bit, Algebra de Boole.

Código Fuente Conjunto de instrucciones escritas por un programador humano en un lenguaje de programación (BASIC, COBOL, Pascal, C, C++, Clipper...). Ver C, Código objeto, Lenguaje, Programa.

Código Máquina Código binario, de dos estados (0 y 1), que usan los ordenadores para comunicarse, procesar la información y entender los programas escritos por los programadores. Ver Código binario, Bit, Algebra de Boole.

Código Objeto Programa obtenido del programa en código fuente, compilándolo. Ver Compilar, Código Fuente, Código Máquina, Codificación.

Constante Valor que no modifica su valor. En un programa informático suele llamarse constante a una sucesión de caracteres que identifican un dato invariable de un tipo concreto. Los Tipos de Datos simples más habituales son, entre otros, entero (por ejemplo la constante 6), real (por ejemplo 3.141592) y carácter (la constante 'K'). Ver Variable, Estructura de Datos, Programa, Registro.

Colgado, ordenador Ver Abortar.

COM1, COM2... Nombres asociados en un PC a los puertos o interfaces de comunicaciones asíncronas. Ver Comunicación, PC, Interface.

Comando Archivo que contiene instrucciones (en código máquina) para un determinado propósito. Puede tener las siguientes extensiones .COM, EXE o también .BAT, pero lo normal es el primero, llamando a los segundos como programas en general y a los terceros como archivos batch. Ver Batch, DOS.

Comandos internos Ordenes del S.O. (DOS) que no tienen un fichero ejecutable asociado, sino que las ejecuta el procesador de comandos COMMAND.COM. Son órdenes como `dir`, `del`, `cd`, `md`, `rd`, `copy`... Ver COMMAND.COM, DOS,

COMMAND.COM Fichero del S.O. MS-DOS que contiene el procesador de comandos del S.O. Se carga memoria al arrancar el sistema y permanece en ella durante todo el tiempo. Se encarga de los llamados comandos internos. Ver Comando, Comandos internos, AUTOEXEC.BAT, CONFIG.SYS, DOS.

Compatibilidad Capacidad de un ordenador de ejecutar los programas y usar los dispositivos diseñados para otro. Normalmente se establecen unas normas comunes, para que los fabricantes las tengan en cuenta, y todos los dispositivos sean compatibles.

Compilador Programa que traduce programas escritos por un programador, en algún lenguaje, normalmente de alto nivel (Pascal, Cobol, C, C++, Basic, Clipper...) a lenguaje máquina (ceros y unos) entendible por un ordenador. En el S.O. DOS, al compilar un programa obtenemos un fichero con extensión .EXE que ya es ejecutable. Ver Intérprete, Código máquina, Código fuente, Código objeto, Lenguaje, C, Aplicaciones.

Compresor Programa que comprime archivos de forma que estos ocupen menos espacio físico, menos memoria, y así se necesiten menos discos para llevarlos a otro ordenador, o menos tiempo de transmisión si se transmiten electrónicamente. Muy usados en FidoNet y BBS's para ahorrar tiempo de transmisión telefónico y por tanto dinero. Para ejecutar o ver los archivos comprimidos se deben descomprimir primero. Ver Backup, Programa, Aplicaciones.

Comunicación Asíncrona Es la comunicación en la cual la transmisión de cada carácter (byte) es independiente del resto de los caracteres que se transmiten durante la misma sesión. De forma que si se produce un error en el envío de un carácter y este es detectado cuando ya se han recibido sin fallos otros caracteres, vuelve a realizarse su transmisión. Ver Comunicación Síncrona, Protocolo de Comunicaciones

Comunicación Síncrona Sistema de transmisión de datos en que se produce una secuencia temporal sincronizada entre los datos transmitidos y los mecanismos de transmisión y envío de esos datos. Para transmitir, se deben sincronizar emisor y receptor, enviando una secuencia de bits de sincronismo. Por esa sincronización este tipo de comunicación es más lenta que la asíncrona, pero más fiable. Ver comunicación Asíncrona, Protocolo de Comunicaciones.

Conmutación de Circuitos Técnica de comunicación entre dos o más estaciones o teléfonos, de modo que se establece y mantiene la conexión durante el tiempo que dura la comunicación. Esa conexión se traduce en una línea directa y asignada en exclusiva. Ver RTC, CTNE, Conmutación de Paquetes.

Conmutación de Paquetes Técnica de transmisión de datos en la que el conjunto de datos se envía segmentado en paquetes. Cada paquete posee información de control, su destino y número de orden. Así, una vez enviados todos los paquetes del mensaje, el remitente corta la comunicación y la red se encarga de llevar cada paquete a su destino. Pueden llegar en distinto orden del que fueron emitidos y por distinto camino cada uno de los paquetes. El receptor los ordenará una vez recibidos todos. Esto permite usar la red por distintos emisores y receptores (usuarios) a la vez. Ver Iberpac, Conmutación de Circuitos, Paquetes de Información.

CONFIG.SYS Fichero del S.O. MS-DOS que contiene la configuración del sistema y que lee cada vez que se enciende o arranca el ordenador. Contiene cosas como el máximo número de ficheros que se pueden tener abiertos a la vez (files), características de los periféricos, drivers de control de periféricos... Ver DOS, AUTOEXEC.BAT, COMMAND.COM.

Consola Modo de referirse a la Entrada/Salida (E/S) estándar, usualmente el teclado (E) y la pantalla (S). Se suele referir como CON. Ver Periférico, Monitor.

Coprocesador Es un procesador específico para cálculos matemáticos auxiliando al microprocesador principal en estas tareas. Acelera notablemente los procesos con abundantes operaciones, por ejemplo, programas de CAD. Ver Procesador, ALU.

Correo electrónico e-mail (electronic mail). Intercambio de mensajes, información, ficheros, programas... a través de medios informáticos y telemáticos, sin necesidad de usar papel. Por ejemplo, a través de una LAN, por teléfono a una BBS o a través de FidoNet. Ver LAN, WAN, BBS, FidoNet, InterNet, Aplicaciones.

CPS Caracteres Por Segundo. Unidad de medida de la velocidad de una impresora de caracteres, o de una comunicación de datos (Por ejemplo entre dos modems). Ver PPM, Bps.

CPU *Central Processing Unit*: Unidad Central de Proceso. Corazón del ordenador, que realiza todos los procesos. Se encuentra en el microprocesador de la placa madre o principal del ordenador, y consta fundamentalmente de la UC (Unit Control) y la ALU (Arithmetic-Logic Unit). Ver UC, ALU, Procesador.

CTNE Compañía Telefónica Nacional de España. La que nos da un susto cuando nos manda la factura del teléfono, la responsable de todos los errores que se produzcan en las transmisiones telefónicas en España y la que se queda con las monedas que se atascan en las cabinas telefónicas. Ver RTC, Conmutación de Circuitos.

Cursor Símbolo en pantalla que indica una posición donde escribir o marcar. También, flecha o símbolo que se maneja con el ratón por la pantalla para señalar una posición. Ver Trackball.

Chip Circuito electrónico miniaturizado que contiene transistores, diodos... Ver Circuito integrado, Procesador.

DBMS *Data Base Management System*, Sistema Gestor de Bases de Datos. Conjunto de programas encargados de gestionar una base de datos. Ver Programa, Aplicaciones, Base de Datos, ODBC.

Decodificación Proceso en el cual se examinan unos datos (digitales) y se convierten a un formato que pueda usar el receptor. Ver Codificación, Modular, Demodular.

Demodular Lo contrario de Modular. Proceso por el cual se aisla o separa la información relevante del resto (señal portadora) de una señal. Es convertir una señal portadora modulada (analógica) en los datos digitales originales. Ver Modem, Modular, Señales Analógicas, Señales Digitales, RTC.

Depuración Eliminar fallos semánticos en un programa. Se suele hacer ejecutando el programa sentencia a sentencia y evaluando el estado del programa (variables...). Ver Variable, Programa.

Digital Ver Señales.

Dirección de Net (Red) Secuencia numérica (o no) utilizada en redes para describir el origen o destino de un mensaje. En la WAN FidoNet, cada BBS tiene una dirección de Nodo, y cada usuario *de a pie* tiene una dirección de Punto. Siempre, una dirección

debe ser distinta de todas las demás. Ver FidoNet, Punto, Red de ordenadores, InterNet, Router.

Dirección de memoria Secuencia de bits que identifica una única palabra o posición de memoria (normalmente memoria principal del ordenador). Ver Memoria, Palabra de memoria.

Directorio 1. Un directorio es un apartado, dentro del disco, donde meter ficheros generalmente de un mismo tema. Dentro de un directorio puede haber ficheros y directorios (también llamados subdirectorios), y dentro de esos subdirectorios, puede, igualmente, haber más ficheros y más directorios. Son usados para organizar la información dentro de un Soporte de Información. Ver Fichero, Soporte de Información, Ruta, Directorio raíz.

2. En un sistema operativo tipo DOS: Indice que el DOS establece en cada disco, para poder saber todos los ficheros que contiene dicho disco. La orden `DIR` indica de cada fichero, el nombre, extensión, tamaño, día y hora de su creación o modificación. Ver Extensión, DOS.

Directorio raíz Directorio principal de un disco, el que contiene todos los subdirectorios de ese disco. Normalmente el del disco duro se representa "`C:\`". Ver Directorio, Ruta.

Disco Soporte para el almacenamiento de grandes cantidades de información (memoria masiva). Puede ser magnético, óptico y magneto-óptico (MO). Los primeros van recubiertos de un material magnetizable y dependiendo de la polaridad del magnetismo se guarda la información (los bits). Los ópticos son grabados y leídos mediante LASER. En general, los magnéticos pueden ser rígidos, discos duros (HD, *hard disk*), normalmente internos al ordenador, no extraíbles y de gran capacidad, y flexibles (FD, *floppy disk*), extraíbles y de menor capacidad. Son soportes de acceso directo y en general su tiempo de acceso es del orden de milisegundos. Ver Pista, Sector, Bernoulli, Memoria, CD-ROM, Soporte de Información, Flóptical, Formatear, Tiempo de Acceso.

DOS *Disk Operating System*, Sistema operativo de disco. Familia de sistemas operativos (de 16 bits) para ordenadores personales (PC). El más extendido es el MS-DOS, pero hay otros como DR-DOS. Por sus características tan precarias están cayendo en desuso poco a poco. Ver MS-DOS, sistema operativo, UNIX.

DMA *Direct Memory Access.* Dispositivo que controla el acceso directo a memoria, de forma que se pueda acceder a la memoria, sin que los datos tengan que pasar por la CPU. Ver CPU, Memoria.

Dpi Puntos por pulgada (inch), también ppp. Mide la resolución de periféricos como impresoras LASER, scanneres... Ver LASER, Scanner.

Driver Programa que amplía las posibilidades del S.O. permitiendo que éste trabaje con un dispositivo o periférico concreto. Suelen ser programas que quedan residentes en la memoria del ordenador, captando las llamadas que se le hagan. Por ejemplo el programa driver para controlar el ratón, una impresora... Ver Residente, FOSSIL.

E-mail Electronic mail. Ver Correo electrónico, Aplicaciones, InterNet, FidoNet, BBS, Smiles.

EEPROM Electricaly EPROM: Unidad de memoria ROM que se puede programar muchas veces, borrándola por medios eléctricos. Ver EPROM.

EOF *End Of File*, Fin de Fichero. Señal o Marca que indica que se ha llegado al final de un Fichero de Datos. Suele verse como un carácter que va al final de la información de un fichero para indicar que no hay más información y que hay que dejar de leer. Ver Carácter, Archivo.

EPROM Erasable PROM: Unidad de memoria ROM que se puede programar una vez y que se puede borrar por rayos ultravioleta. Ver PROM.

Estructura de Datos Conjunto de información con algún formato específico. Algunas estructuras de datos típicas son las cadenas de caracteres, arrays, registros, ficheros, conjuntos, listas, pilas, colas, árboles... Ver Archivo, Árbol, Registros, Puntero, Constante, Variable.

Extensión Sufijo de 3 o menos caracteres que se añade al nombre del fichero separado con un punto, en el DOS, y que suele indicar su contenido:

- .EXE $\longrightarrow$ Programa Ejecutable.
- .COM $\longrightarrow$ Programa, COMando.
- .BAT $\longrightarrow$ Fichero de ejecución en BATch (por lotes).
- .TXT $\longrightarrow$ Fichero de Texto.
- .ASC $\longrightarrow$ Archivo de texto en código ASCII.
- .BIN $\longrightarrow$ Archivo con información en código Binario.
- .SYS $\longrightarrow$ Fichero para controlar algo del sistema o de configuración.
- .HLP $\longrightarrow$ Fichero con información de ayuda (Help).
- .INI $\longrightarrow$ Fichero con parámetros de inicialización de algo.
- .C $\longrightarrow$ Fichero de texto, conteniendo un programa fuente en lenguaje C.
- .CPP $\longrightarrow$ Programa fuente en C++.
- .PAS $\longrightarrow$ Programa fuente en Pascal.
- .OBJ $\longrightarrow$ Código objeto de un programa (útil para compilar).
- .TMP $\longrightarrow$ Fichero temporal, intermedio.
- .DOC $\longrightarrow$ Documentación sobre algo.
- .PIF $\longrightarrow$ Datos sobre un programa (Program Information File).
- .DBF $\longrightarrow$ Fichero de Base de Datos (de Dbase IV).
- .DBT $\longrightarrow$ Fichero de texto de Dbase IV (Data Base Text)
- .MDX $\longrightarrow$ Ficheros de índices de Dbase IV.
- .LBL $\longrightarrow$ Ficheros de etiquetas de Dbase IV (LaBeL).
- .FRM $\longrightarrow$ Fichero con un informe de Dbase IV (FoRM).
- .ARJ $\longrightarrow$ Fichero comprimido con el compresor ARJ.
- .ZIP $\longrightarrow$ Fichero comprimido con el compresor PKZIP.

- .DLL ⟶ Fichero con funciones o subprogramas que serán usados por otros programas. Biblioteca de enlace dinámico (*Dinamic Link Library*).
- .PCX .BMP .GIF .TIF .TGA .EPS ... ⟶ Fichero con una imagen en distintos formatos (hay muchos más formatos).
- .WAV .VOC ...⟶ Fichero con un sonido en diversos formatos.
- .MID ⟶ Fichero de música en formato MIDI.
- .AVI ⟶ Fichero de vídeo que compagina Audio y Vídeo.
- .MPG ⟶ Fichero de vídeo dígital en el estándar MPEG desarrollado por *Motion Pictures Experts Group.*

y muchísimos más...

Fichero Ver archivo.

FidoNet Red de tipo WAN, formada por BBS de todo el mundo que se organizan de forma jerárquica. Los asociados a ella (Nodos o Puntos) pueden intercambiar entre sí, desde cualquier punto del planeta, mensajes y ficheros. Ver Niveles FidoNet, WAN, Nodo, Punto, InterNet, Red de ordenadores.

Flóptical Unidades de almacenamiento masivo con tecnología híbrida, magnética y óptica. Consiguen introducir más pistas en el mismo espacio, aumentando así la capacidad, además de reducir los tiempos de acceso. Ver Disco, Soporte de Información.

Formatear Dar formato. Preparar un disco magnético para que pueda ser utilizado para el almacenamiento de información. Crearle las pistas y sectores. Si ya tuviera información, esta se perderá. En el DOS se hace con la orden `FORMAT`. Ver Pista, Sector, Soporte de Información.

FOSSIL *Fido, Opus, Seadog, Standard Interface Layer.* Driver que actua como interface (intermediario) software para la gestión de comunicaciones. Suele ser un programa residenteque se debe cargar en memoria antes de ejecutar un programa de comunicaciones para un modem... Hay algunos shareware muy buenos, como BNU o X00. Ver Driver, Modem.

Gateway Puente, enlace. A través de ese *puente* pueden comunicarse dos sistemas, dos redes LAN, para intercambiar servicios y mensajes. Dispositivo para comunicar dos redes distintas, encargado, entre otras cosas, de la conversión de los mensajes del protocolo de una red al de otra. Ver Router.

Gigabyte GB Unidad de medida de cantidad de información binaria o capacidad de almacenamiento, tamaño de una memoria. Equivale a 2^{30} bytes, 2^{20} Kbytes y 2^{10} Mbytes (Megabytes). Ver Byte.

Hacker Apasionado de los sistemas informáticos, cuya aficción al ordenador y a todo lo relacionado puede conducirle a un estado psíquico de carácter patológico que desemboca, a menudo, en la sociopatía. Características comunes: mente lógica, inteligente, individualismo, apatía ante convencionalismos sociales... A veces es llamado *pirata* informático, por introducirse sin permiso en sistemas informáticos ajenos.

Hardware Parte física, dura (hard) de un ordenador: Circuitos, chips, periféricos, discos, memorias... Ver Software, Chip, Periférico.

Hoja de Cálculo Programa que simula una hoja de trabajo típica en pantalla, y que permite incluir gran cantidad de datos y fórmulas interconectadas entre sí. Permite ver rápidamente las variaciones producidas en los resultados variando los datos iniciales. Normalmente se estructura en una tabla con filas y columnas, las cuales se pueden relacionar mediante complejas fórmulas matemáticas. Entre otras utilidades, destacan el cálculo de presupuestos, cálculo de viabilidad financiera, simulación de comportamientos respecto a unos datos iniciales... Ver Programa, Aplicaciones.

Host Ordenador grande (mainframe), en el que el sistema *hospeda* a otros sistemas más pequeños, usando una red de comunicaciones. Suele ser un ordenador encargado de la gestión de la Base de Datos de una gran empresa. Ver Mainframe, Red de Ordenadores.

Iberpac Red nacional de transmisión por paquetes de información (en binario). Servicio especial (de la CTNE) de teletexto proporcionado por Ibertex, mediante el que es posible conectar vía modem. Ver Videotex, Ibertex, Conmutación de paquetes, Modem.

Ibertex Servicio español de videotex desarrollado por CTNE. Se encarga de gestionar la comunicación entre las bases de datos de distintas compañías y el usuario. Este servicio se utiliza principalmente, para la compra desde casa, banco en casa, mensajería, juegos, ocio, información de Amnistía Internacional, party line... Ver Videotex e Iberpac.

Interfaz *Interface*: Circuitería (hardware) o programa (software) que se interpone entre dos sistemas informáticos que se desean conectar. Se encarga de pasar la información de un sistema al otro para que este la entienda. Lo ideal es que esté estandarizado y sirva para muchos sistemas. Ver Hardware, Software.

InterNet Red internacional (WAN) de comunicación de ordenadores. En los ordenadores conectados a dicha red existen miles de TeraBytes de información disponible para el que quiera acceder a ella. Ofrece multitud de servicios, entre los que destacan: e-mail (correo), ftp (transferencia de ficheros), conferencias y foros de debate (news), WWW (*World Wide Web*, Consulta de información en un hipertexto)... y multitud de posibilidades adicionales. Ver WAN, FidoNet, e-mail, Red de Ordenadores.

Intérprete Programa que traduce sentencia a sentencia un programa y las va ejecutando antes de traducir la siguiente sentencia. Un compilador las traduce todas de una vez y no las ejecuta, sino que deja el resultado en espera de ser ejecutado. Por este segundo sistema la ejecución de un programa es más rápida, ya que no tiene que ir traduciendo. Ver Compilador, Traductores.

ISO *International Standard Organization.* Organización internacional para la estandarización de sistemas informáticos. Creó el sistema de comunicación estándar OSI, y elabora normas para los más diversos fines. Ver OSI, ISO 9000.

ISO 9000. Estándar dictado por ISO para construir software de calidad. Ver ISO, Software, Programa.

Kbyte KB Unidad de medida de memoria que equivale a 2^{10} bytes. Ver byte, Gigabyte.

Kernel Núcleo del S.O. Parte del S.O. que está más en contacto con el hardware. Es la parte de más bajo nivel. Ver S.O., UNIX.

LAN Local Area Net: Red de area local. Red de ordenadores que normalmente abarca desde 2 ordenadores hasta varios edificios. Las principales utilidades son la transmisión de información (e-mail...) y compartir recursos (impresoras, discos, modems...). Las topologías más comunes son el bus, anillo, estrella, red... Ver WAN, MAN, Servidor, Repetidor, Router, Gateway, Red de ordenadores, Administrador de una red, e-mail.

LASER *Light Amplification by Stimulated Emitted Radiation.* Tipo de luz muy usada en las nuevas tecnologías de impresoras y dispositivos de almacenamiento. La luz está formada por un haz de radiaciones luminosas monocromáticas (de un solo color, de una única longitud de onda) y coherentes (de igual fase). Ver CD-ROM, Disco, Soporte de Información, Flóptical.

LED *Light Emitting Diode*: Diodo emisor de luz. Son como pequeñas bombillas, normalmente rojas, verdes o amarillas que llevan los teclados, ordenadores, modems externos... para indicar su estado. Ver Modem.

Lenguaje de programación Lenguaje con el que se le puede decir a un ordenador lo que queremos que haga, a través de un conjunto de órdenes con una sintaxis específica. Se denomina *de alto nivel* a aquel tipo de lenguaje que se aleja de la programación en código máquina y se acerca al lenguaje humano. Los programas escritos en estos lenguajes se deben compilar o interpretar para ser ejecutados. Ver Compilador, Traductores, Código fuente, C.

LINUX S.O. UNIX de 32 bits para PC's, de dominio público, es decir, gratuito (no comercial). Es un S.O. tremendamente potente y con multitud de programas para cualquier utilidad. Por su potencia, precio y prestaciones su uso se está extendiendo cada vez más. Ver UNIX, PC.

Mail Correo. Ver correo electrónico.

Mailer Programa dedicado al tratamiento de los mensajes que automatiza y optimiza el coste de la recepción y envío de mensajes (normalmente por teléfono). Ver Punto, FidoNet.

Mainframe Ordenadores de gran potencia, tamaño y precio. Puede ser multiusuario, gestionando miles de terminales y muchas operaciones simultáneamente. Ver host.

MAN Metropolitan Area Net: Red de Area Metropolitana, o urbana. Red de ordenadores que abarca una ciudad. Por ejemplo, una red entre hospitales de una ciudad, o entre las comisarías de policía... Ver Red de Ordenadores, LAN, WAN.

Megabyte MB Unidad de medida de memoria que equivale a 2^{20} bytes, 2^{10} Kbytes. Ver byte, Gigabyte.

Memoria Lugar donde almacenar información y programas por el ordenador. Se llama memoria Principal a la que maneja directamente la CPU. Esta puede ser RAM y ROM. Es de acceso directo y su tiempo de acceso es del orden de nanosegundos. Ver caché, RAM, ROM, CPU, Dirección de Memoria, Palabra de Memoria, Soporte de Información, Disco, Bit, Byte, Tiempo de Acceso, Von Neumann.

Milisegundo Milésima parte de un segundo. Unidad de medida usada, entre otras cosas, para medir el tiempo de acceso a los datos en los Soportes de información. Ver Nanosegundo, Soportes de Información, Tiempo de Acceso, Disco.

Modem Dispositivo que permite transmitir información entre dos equipos informáticos distantes, utilizando la red telefónica pública (RTC), transmitiendo la información en forma de pitidos. Su nombre procede de su función: MOdular/DEModular, convirtiendo la señal digital (ceros y unos) de los datos generados por un sistema informático, en señal analógica de audio (pitidos audibles por el oído humano). En el destino, otro modem hará lo contrario (demodular), convirtiendo la señal analógica audible en señal digital. Ver Señales, Bps, Baudio, CCITT, Periférico, Protocolo de Comunicaciones, Modular, Demodular.

Modular Proceso de adaptación al medio de una señal para que pueda ser transmitida conteniendo toda la información (normalmente es una onda electromagnética de distinta frecuencia). En general, lo que se hace es variar una señal (llamada portadora), según el modelo proporcionado por otra señal. En general la portadora es una señal analógica y la otra es una señal digital , con el objetivo de transmitir por una línea de comunicación analógica (como la RTC) la información contenida en la señal digital. Ver Modem, Demodular, Señales Analógicas, Señales Digitales, RTC, Codificación.

Monitor Dispositivo de salida más típico. Consta de una pantalla (similar a una TV), en la que el ordenador representa de forma gráfica (o textual) los datos. Normalmente está formado por un tubo de rayos catódicos (CRT). En los portátiles suele ser de tecnología TFM o LCD. Ver Paleta, Periférico, Consola.

Mhz Megaherzios Medida de frecuencia equivalente a un millón de ciclos (pulsos) por segundo. Se usa para medir la velocidad del reloj de la CPU, y por tanto, como medida de la velocidad de esta. No es significativo comparar dos procesadores distintos por la cantidad de Mhz que usan, pero sí si el procesador es de igual modelo pero con distintos Mhz. Ver Reloj, CPU.

MS-DOS MicroSoft - Disk Operating System: Sistema operativo (S.O.) de Disco de la compañía se Software MicroSoft Corporation. Es un S.O. pésimo por su gestión de memoria y por ser monotarea, entre otros problemas, acarreados por su regla de tener que ser compatible con el primer PC. La última versión comercial ha sido y será la 6.22. Va siendo lentamente sustituido por sistemas operativos gráficos (como Windows95). Ver S.O., DOS, AUTOEXEC.BAT, CONFIG.SYS, COMMAND.COM, Extensión, Directorio, Windows.

Multitarea Modalidad de uso de un ordenador que permite la ejecución de diversas actividades a la vez. Ver Multiusuario, UNIX, Sistema Operativo.

Multiusuario Modalidad de empleo de un ordenador que permite que esté a disposición de varios usuarios simultáneamente. Presupone ser Multitarea. Ver multitarea, UNIX, Sistema Operativo.

Nanosegundo Mil millonésima parte de segundo. Unidad de medida usada para medir el tiempo de acceso a dispositivos rápidos, como memoria RAM, ROM, caché... Ver Memoria, RAM, ROM, Caché, Tiempo de Acceso.

Net/Network Red/Red de trabajo: Red de comunicación entre ordenadores. Formadas por nodos que son los terminales. Ver LAN, MAN, WAN, FidoNet, Niveles FidoNet, InterNet, Red de Ordenadores, Nodo.

Niveles FidoNet La red FidoNet tiene una organización jerárquica y un sistema de comunicación concreto. La red abarca todo el planeta (primer nivel, es una WAN). En un segundo nivel se encuentran las Zonas (normalmente continentes), divididas en regiones (países), que se dividen también en Nets (autonomías, provincias...). Las Nets están formadas por Nodos, que son BBS. Por último, el nivel más básico de esta organización lo componen los Puntos, o usuarios cualificados de la BBS, del nodo al que pertenecen. Ver Nodo, FidoNet, Punto, WAN.

Nodo Terminal conectado a una red (LAN...) que normalmente será un ordenador. También, suele ser equivalente a BBS perteneciente a FidoNet, en comunicación por modem. Ver Niveles FidoNet, Red de ordenadores, Net.

Norma Definición disponible públicamente de un componente de hardware o software resultante de un acuerdo internacional, nacional o industrial. Ver CCITT, ISO, Protocolo de Comunicaciones.

Núcleo de un S.O. Ver Kernel.

Oculto, Fichero Archivo o Fichero oculto es aquel que no aparece en el listado del directorio (mediante la orden `dir` del DOS). Estos archivos no pueden ser borrados, copiados o afectados por comandos del S.O. Suelen ser archivos que usa exclusivamente el S.O. Ver Sistema Operativo, DOS, Archivo, Directorio.

ODBC Open Database Conectivity: Conectividad de Bases de Datos Abiertas. Acceso al interface de programación de Base de Datos. ODBC proporciona un lenguaje común para que las aplicaciones (Windows) interactúen con diversas Bases de Datos. Ver Base de Datos, DBMS.

Ordenador Máquina que procesa y almacena información. Capaz de ejecutar programas para obtener unos resultados a partir de unos datos de entrada. Se comunican con el exterior (con el hombre) a través de periféricos. Sinónimo de ordenador son computador o computadora. Ver Programa, Periféricos, CPU, PC, Mainframe.

OSI *Open System Interconection.* Sistema abierto de Interconexión. Un modelo estándar de ISO para la comunicación entre sistemas (redes). El proceso de comunicación lo divide en 7 niveles claramente diferenciados: físico, enlace, red, transporte, sesión, presentación y aplicación. Ver ISO, red.

Palabra de Memoria Número de bytes (o bits) que se pueden leer o escribir a la vez en memoria principal (RAM) de una sola vez. Indica el ancho (en bits) de la memoria principal. Cuanto mayor sea la longitud de palabra de un procesador, más rápido podrá operar. Así, los primeros PC's eran de 8 bits, luego aparecieron de 16, 32 y los actuales son de 64 bits (como el Pentium). Ver Byte, Memoria, Dirección de Memoria, Pentium.

Paleta Tabla de colores disponibles simultáneamente en una pantalla o monitor. Antiguamente el número de colores era de 2 (monócromo). Actualmente lo más habitual es

encontrarse con paletas de 16 ó 256 colores, aunque se puede llegar a más de 16 millones de colores (Color Real, *True Color*). Ver Monitor.

Paquetes de Información Partes en las que se divide un mensaje y a los que se añaden un número de bits de control (según el protocolo usado), antes de enviarlo por una Red de Ordenadores a su destino. Ver Red de Ordenadores, Protocolo de Comunicaciones, Testigo, Iberpac, Conmutación de Paquetes.

Password Clave: Cadena de caracteres solicitada por un sistema, para reconocer a un usuario concreto y permitirle acceder como si fuera dicho usuario. Debe ser secreto para todos menos para ese usuario. Normalmente se debe cambiar cada mes para evitar que otros la puedan averiguar y entren suplantando la personalidad de dicho usuario.

Path Camino, ruta. Ver ruta.

PC Personal Computer. Sistema de microordenadores muy extendido y casi estandarizado en la pequeña y mediana empresa, así como para uso doméstico y lúdico. Se le suele llamar PC a todo ordenador compatible con el primer ordenador personal de IBM. Ver Ordenador, Mainframe.

PCMCIA Personal Computer Memory Card International Association. Asociación de más de 300 fabricantes que crearon este estandar en tarjetas para ordenador. Son tarjetas de ampliación de un ordenador (modems, memoria, sonido, red...), ideales para portátiles por su pequeño tamaño (85,6x54x3,3 mm. en la primera versión). Por ahora hay tres tipos de diferentes grosores: Tipo I, II y III. Se pretenden que sigan la tecnología de *plug & play*. Ver Tarjeta, Plug and play, Bus.

Pentium Procesador de nombre registrado por la compañía Intel, superior a los anteriores 8088, 8086, 80286, 386 y 486. Existen versiones a distintas frecuencias de reloj (Mhz). Incorpora más de 3'3 millones de transistores y funciona sólo a 3'3 Voltios, reduciendo así su consumo. Tiene arquitectura de 64 bits. Ver Palabra de memoria, Pentium Pro, Procesador, CPU.

Pentium Pro Procesador superior al Pentium (también de Intel) en aplicaciones de 32 bits. En aplicaciones con código de 16 bits puede mostrarse con un rendimiento menor. Ver Procesador, Pentium.

Periférico Dispositivo que se acopla al ordenador para que este se comunique con el exterior. Pueden ser de entrada (teclado, ratón, sensores, scanner, modem...) o de salida (pantalla, impresora, plotter, modem...). Ver Consola, Soporte de Información, Disco, Monitor, Tarjeta, Modem.

Pista Cada uno de los círculos concéntricos de la superficie de un disco que contiene datos. Esta dividido en sectores. Los discos ópticos (CD-ROM) sólo tienen una pista en forma espiral. Ver sector, formatear, disco, CD-ROM.

Plotter Trazador gráfico. Equipo dotado con varias plumas de distintos colores, para dibujar gráficos, controlado por un ordenador. Ver Periférico.

Plug and Play Enchufar y funcionar: Tecnología de fabricación cuyo objetivo es simplificar al máximo la instalación de todo tipo de periféricos. Se trata de dado un periférico

(tarjeta...) enchufarlo al ordenador y que automáticamente se configuren todos los parámetros, de forma que el usuario sólo tenga que conectar la tarjeta y... usarla. Ver Tarjeta, PCMCIA.

Portabilidad Propiedad de los programas e instrumentos para ser transferidos sin modificaciones desde unos equipos a otros.

PowerPC Procesador de Motorola, nacido de la alianza entre esta empresa e IBM. Es muy potente, como el Pentium, pero no es compatible con este. Igualmente, trabaja con 64 bits. Ver Pentium, Pentium Pro, CPU, Procesador.

PPM Páginas por minuto. Unidad de medida de la velocidad de una impresora rápida, por páginas, de tipo láser, inyección... Ver CPS.

Procesador Unidad en la que tiene lugar el tratamiento de los datos. Puede estar integrada en un único chip o en varios. Ver Chip, CPU, Pentium, Pentium Pro, PowerPC, Procesador escalar, Procesador vectorial, Procesador front-end, Von Neumann.

Procesador escalar Procesador basado en un único conjunto de registros y de unidades de proceso. Ver procesador, procesador vectorial.

Procesador front-end Procesador utilizado para controlar el flujo de datos desde y hacia un procesador principal. Ver Procesador.

Procesador de Textos Programa informático que ayuda al tratamiento automático de textos: Centrar, sangrar, control de márgenes y tipos de letra (tamaño, negrita, cursiva...), impresión de documentos, almacenamiento y recuperación en ficheros... Sin ir más lejos, este libro se ha confeccionado utilizando uno de estos programas. Hay muchos y muy variados, pero dentro del mundo PC hay que destacar dos procesadores de textos comerciales, el Word-Perfect y el Word. En el ámbito científico y de autoedición es muy utilizado el LaTeX. Ver Programa, Aplicaciones.

Procesador vectorial Procesador con una arquitectura paralela basada en el uso de registros vectoriales y de grupos de circuitos. Ver Procesador, Procesador escalar.

Programa Secuencia de instrucciones ordenadas, pensadas para que un ordenador las ejecute en ese orden, para obtener un resultado concreto, como imprimir un texto, ejecutar operaciones, buscar un dato... Básicamente hay tres tipos: Sistemas Operativos, Traductores y aplicaciones en general. Ver Ordenador, Código fuente, Sistema Operativo, Traductores, Aplicaciones, Algoritmo, Depuración, ISO 9000, Von Neumann.

PROM Programable ROM: Unidad de memoria ROM que se puede programar (escribir) una vez y ya no se podrá borrar ni sobreescribir más. Ver ROM.

Prompt Indicación que el ordenador presenta al usuario para que este introduzca información o realice alguna acción en el S.O. Se suele llamar también símbolo de atención, y la línea donde se escribe el comando es llamada línea de comandos. En un S.O. tipo DOS suele indicar la unidad o disco en el que está y su directorio o subdirectorio (Ruta), y el símbolo >, seguido del cursor. Ejemplo: `C:\>_`. Ver Ruta, S.O., DOS, Directorio.

Protocolo de Comunicaciones Conjunto formal de reglas que gobiernan los formatos de datos, las temporizaciones, el control de secuencias y acceso, detección de errores... necesario para iniciar y mantener una comunicación entre sistemas o dispositivos. Ver Comunicación, CCITT, Red de Ordenadores, Topología de una Red, Norma.

Puntero En programación, una dirección de memoria que indica (o apunta) la posición, la dirección física de un dato concreto. Ese dato puede ser un registro con unos datos y otro puntero que apunte a otro registro igual, y así sucesivamente, formando una lista de registros. Ver Registros, Estructura de Datos.

Punto Usuario cualificado de una BBS de FidoNet, que colabora con el sistema a través de un mailer. Para ser punto de una BBS, debe solicitarlo al SysOp de dicha BBS, el cual le asignará una dirección de punto, su dirección. Por ejemplo 2:345/803.13 es la dirección del punto número 13 de la BBS 803 de Andalucía (5) en España (34) de Europa (2). Ver Niveles FidoNet, Mailer, Dirección de Red, FidoNet, Net, SysOp.

RAM *Random Access Memory*: Memoria de acceso aleatorio (directo). Memoria que usa la CPU para almacenar los programas en ejecución y sus datos. Su contenido varía en la ejecución de un programa y se pierde totalmente al desconectar el sistema. Ver memoria, ROM, byte.

RDSI Red Digital de Servicios Integrados. Red de comunicación digital. En el futuro será como la red telefónica pero no sólo para voz, sino para voz, datos, imágenes, TV, fax... Ver RTC. Red.

Red

1. Red de ordenadores (Net): Conjunto de ordenadores unidos por conexiones (cables, teléfono, ondas de radio, microondas, fibra óptica...) que comparten información y recursos (impresoras, discos...). Por ejemplo, los ficheros se almacenan en los llamados Sistemas de Ficheros a los que los usuarios pueden acceder fácilmente. En cada ordenador se puede ejecutar un proceso distinto, independientemente del resto. Ver LAN, MAN, WAN, FidoNet, InterNet, RDSI, RTC, Fichero, Arquitectura de una Red, Topología de una Red, Acceso al Medio.
2. Base de Datos: Sistema de organización de una base de datos en la que cualquier tipo de registro se puede relacionar con cualquier otro, formando una red de relaciones. Ver Base de Datos, Registros, Relacional.

Registros

1. Record: Son estructuras de datos simples usadas en programación, que hacen referencia a un mismo ente, a través de varios valores o datos. Ejemplo: Un registro de un cliente, contendrá los datos (campos) necesarios para identificar dicho cliente (DNI, nombre, dirección, teléfono...). A cada uno de esos datos elementales se le llama campo (field). Ver Puntero, Estructura de Datos.
2. Register: Dispositivo de almacenamiento de alta velocidad, utilizado para operaciones de importancia primaria y de frecuencia elevada. La CPU lo usa para almacenar los datos y resultados de las operaciones aritmético- lógicas en curso, además de los errores cometidos (registro de estado), control de operaciones, control de instrucción a ejecutar a continuación (registro contador de instrucciones)... Suele tener tantos bits como la palabra de memoria. Ver Palabra de Memoria, CPU.

Relacional Sistema de organización de una base de datos a través de tablas. Cada columna de la tabla se corresponde con un campo de un registro, y cada fila contendrá todos los campos de un registro completo. Las relaciones entre dos tablas se hacen por los campos que tengan en común, y que deben ser llaves (campos que distingan un registro de otro en una tabla, como el DNI por ejemplo). Ver Registros, Base de Datos.

Reloj Circuito que sincroniza las operaciones de la CPU. Emite un pulso a intervalos iguales de tiempo. Además, se usa para sincronizar la comunicación de datos entre la CPU y otros componentes (memoria, sistema de video, audio...). Ver CPU, Mhz.

Repetidor Alumno que, por suspender mucho, debe repetir el mismo curso al año siguiente. En telemática, aparato usado en las LAN para que no decaiga la señal de los mensajes por tener que recorrer estos demasiada distancia. Lo único que hace es amplificar la señal, de forma que pueda llegar más lejos. También son llamados Amplificadores. Ver LAN, Red de ordenadores.

Reset Re-establecer, Reinicializar un sistema estableciendo los valores iniciales que deba tener. Ver Abortar.

Residente Programa residente: Es un programa que no se elimina de la memoria al terminar, sino que se queda en ella, esperando a que suceda algún evento (pulsar una tecla, trasmitir por modem...) para actuar. También se llaman TSR (Termina y queda Residente). Se suelen usar como Driver de algunos dispositivos. Ver Driver.

ROM *Read Only Memory*: Memoria de sólo lectura. Memoria que usa la CPU para leer programas grabados de forma imborrable (no como la RAM), y que son básicos para el ordenador. Normalmente se leen cuando se arranca el ordenador, de forma que estos programas le dicen a la CPU qué cosas debe hacer al empezar (chequeo de disqueteras, de memoria, cargar el S.O., ...). Ver memoria, RAM, PROM, EPROM, EEPROM.

Router Dispositivo que conecta redes (LAN) o parte de una red (WAN). Es el encargado de enrutar los mensajes por el camino que les corresponda. Dependiendo de su destino, el router encaminará cada mensaje por un camino distinto, dependiendo de su Dirección de Red. Ver LAN, MAN, WAN, Gateway, Red de ordendadores, Dirección de Red.

RTC Red Telefónica Conmutada. La red de telefonía habitual, para voz, pero que también se usa para datos. Ver Conmutación de Circuitos, Modem, CTNE, Red de ordenadores.

Ruta Camino (path), lista de directorios, empezando por la letra de la unidad (disquetera), que definen la ubicación de un directorio o de un fichero. A veces la ruta se pone delante de un nombre de archivo para indicar el directorio en el que se encuentra dicho archivo. Por ejemplo: El archivo PAZ.EXE está en el subdirectorio PP del subdirectorio ONU del disco duro (unidad C:). Su nombre completo de archivo y su ruta completa sería: `C:\ONU\PP\PAZ.EXE`. Ver Archivo, Directorio, Unidad, Prompt.

Scanner Periférico para digitalizar imágenes (textos, fotos, gráficos...), esto es, facilita el pasar esas imágenes al ordenador para que este las pueda manejar, imprimir, modificar... Ver Periférico, Dpi.

Sector En discos magnéticos, segmento de una de las pistas concéntricas grabadas en la acción de formatear un disco. Cada pista se puede dividir en varios sectores de igual tamaño (normalmente 7-8 en un disco flexible). Ver Pista, Cluster, Formatear, Disco, Soporte de Información.

Secuencia Estructura de control en un programa (o algoritmo) por la cual se transfiere el control de una instrucción del programa a la siguiente. Las instrucciones (u órdenes) se ejecutan una tras otra en riguroso orden secuencial. Ver Bucle, Selección, Algoritmo, Programa.

Selección Estructura de control en un programa (o algoritmo) por la cual se transfiere el control de una parte de un programa a otra. Puede ser incondicional (salta a otra parte pase lo que pase) o condicional (salta sólo si se cumple determinada condición. Se trata de *seleccionar* una parte del programa a ejecutar de entre varias posibles. Se suele expresar, en los programas con la estructura `if`. Ver Bucle, Algoritmo, Programa, Secuencia.

Señales Analógicas Ondas de una frecuencia determinada que pueden tomar infinitos valores distintos, entre un rango posible. Suelen representarse por funciones sinusoidales. Ver señales digitales, modem.

Señales Digitales Ondas que consisten en una serie de pulsos de tiempo de subida y bajada muy pequeño (se considera nulo), teniendo en la práctica sólo varios valores posibles, normalmente 2: Estado alto (1) y estado bajo (0). Ver Señales analógicas, Modem, Modular, Demodular.

Servidor Gestor de recursos de un sistema informático en red. Normalmente es un ordenador dedicado (en exclusiva o no) a gestionar ficheros (servidor de ficheros), colas de impresión (servidor de impresión) o comunicaciones (servidor de comunicaciones). Ver LAN, Red de ordenadores, Unidad.

Sistema Operativo S.O.: Programas que coordinan el funcionamiento de todas las partes de un sistema informático. Gestión de memoria, de procesos activos, de entradas/salidas... Ver DOS, Windows, LINUX, UNIX, Multiusuario, Multitarea.

Slot Ranura que contienen algunos dispositivos electrónicos, capaz de alojar una tarjeta electrónica. La tarjeta se inserta en la ranura de forma que hagan contacto multitud de líneas de comunicación. De esta forma permite ampliar un dispositivo (ordenador). Cada Slot debe ir conectado a un bus. Dependiendo del tipo de bus hay distintos tipos de slots (ISA, PCI, EISA...) Ver Tarjeta, Bus.

S.O. Ver Sistema Operativo.

Software Elementos lógicos (soft=blando), no tangibles del ordenador, tan necesarios como el Hardware. Lo forman básicamente programas y datos. Ver Hardware, Programa, Sistema Operativo, Aplicaciones, ISO 9000.

Soporte de Información Dispositivo hardware utilizado para almacenar información en formato binario (0,1). La información se suele organizar en Ficheros y Directorios. Ver Archivo, Directorio, Disco, Bernoulli, Flóptical, Memoria, Tiempo de Acceso.

Smiles Caritas o caretos. Son un método expresivo muy extendido en BBS y FidoNet, que simbolizan una expresión facial relacionada con lo que se acabe de comentar en ese momento. Nacen de la necesidad de hacer más expresivas las frases escritas por e-mail. Se ven inclinando la cabeza hacia la izquierda. Por ejemplo, para indicar que se trata de una broma (y no un insulto), se podría escribir: "Es que estás loco... ;-) ". Hay tantos como imaginación. Los más usados, entre otros, son los siguientes. Ver InterNet, E-mail, FidoNet.

`:-)`	Sonrriendo	`:-(`	Triste	`:-|`	Serio
`:-D`	Riendo	`;-)`	Giñando	`:'-(`	Llorando
`8-)`	Con gafas	`B-)`	Otras gafas	`:-9`	Relamiéndose
`X-D`	Partiéndose de risa	`:-/`	Mosqueado	`}:-)`	Diablillo
`:-?`	Dubitativo	`O:-)`	Inocente (santo)	`:-O`	Sorprendido
`:-X`	Mudo	`=:-)`	Con cresta	`&:-(`	Cacao Mental
`:-[`	Drácula	`:-.`	Silvando (disimulando)	`%-O`	Asustado

SysOp *System Operator*, Operador del Sistema. Máximo responsable del mantenimiento y funcionamiento de una BBS. Ver FidoNet, Niveles FidoNet, Punto, Nodo, BBS.

Tabulador Carácter de control utilizado para ordenar datos en forma de tabla (tabular). En general es como si se escribiran varios espacios en blanco, pero teniendo en cuenta que el tabulador es un sólo caracter. En algunos lenguajes de programación (como C) se representa como `'\t'`. Ver Carácter.

Tarjeta Placa de plástico duro con diversos chips integrados, encargados de una función específica. Así, hay tarjetas de video, de sonido, de comunicaciones (de red, modems...), controladoras de discos, Controladoras del sistema gráfico... Estas tarjetas suelen ir insertadas en la placa principal del ordenador, en los llamados slots (ranuras) de expansión. Ver Chips, Slot, PCMCIA, Plug and play, Bus, Periférico.

Terabyte TB. Unidad de medida de memoria que equivale a 2^{40} bytes, 2^{30} Kbytes, 2^{20} Mbytes, 2^{10} Gbyte. Ver byte.

Terminal Ordenador o periféricos usados para conectarse a una red o a un ordenador central normalmente de mayor potencia, capacidad y velocidad. Ver red, Nodo.

Testigo (o token) Secuencia de caracteres utilizada para sincronizar la utilización de la línea de transmisión en las redes, de forma que no accedan al medio varios ordenadores a la vez. Ver Red de Ordenadores, Arquitectura de una Red, Topología de una Red, Acceso al Medio.

Tiempo de Acceso Tiempo empleado en leer/escribir un dato en un Soporte de Información, principalmente de acceso directo. En general la medida dada es una medida orientativa del tiempo de acceso medio. Ver Soporte de Información, Nanosegundo, Milisegundo, Disco, Memoria, Acceso Directo.

Topología de una Red Configuración formada por los nodos de la red y las interconexiones entre ellos. Describe cómo va el cable o línea de transmisiones desde un nodo a otro. Ver Red de Ordenadores, Arquitectura de una Red, Acceso al Medio, Testigo.

Trackball Dispositivo apuntador de igual utilidad que un ratón. Es similar a un ratón invertido. Consta de una bola que se mueve directamente con la mano y hace moverse un cursor en la pantalla. Se usa sobretodo en ordenadores portátiles. Ver Cursor, Periférico.

Traductores Ver compiladores e intérpretes.

UC *Unit Control*, Unidad de Control: Parte de la CPU que controla y sincroniza el correcto funcionamiento del sistema. Ver CPU, ALU.

Unidad Normalmente, unidad de disco: Dispositivo que maneja directamente los discos, bien para leer su información o para introducirla en archivos. En un S.O. tipo DOS se suelen nombrar por letras, seguidas de dos puntos. Normalmente, A: y B: son unidades de disco flexible y C: es unidad de disco duro. Si tuviera dos discos duros (o uno con varias particiones) el segundo sería D:, y así sucesivamente con otros discos duros o unidades de CD-ROM. Normalmente, cuando se trabaja en red (LAN), los discos de la red suelen tener nombres por encima de la E:. Ver Ruta, Directorio, Disco.

UNIX Sistema operativo apropiado para aplicaciones multiusuario, desarrollado (en principio y principalmente) en lenguaje C. Es mucho mejor (sin comparación) que el DOS pero requiere más potencia. Es un tipo de sistema operativo potente y robusto, ideado para facilitar el trabajo a los usuarios. Es siempre un sistema operativo de 32 ó 64 bits. Ver DOS, Multiusuario, Multitarea, XENIX, LINUX.

Usuario Persona que hace uso de un sistema informático u ordenador. Ver Ordenador, PC, UNIX, Multiusuario.

Variable Dirección de memoria que contiene un dato que puede variar a lo largo de la ejecución de un programa. Se usa en programación. Ver Programa, Depuración, Constante.

Versión X.xx Modificaciones sucesivas de un mismo programa, S.O., etc... Cuando la modificación es sustancial se le añade un número entero (la X, por ejemplo ver 5.0) y cuando la versión incluye pocas variaciones se le añade un decimal (las xx, por ejemplo, ver 5.1). Ver Aplicaciones.

Videotext Sistema de bases de datos con espectacular proliferación en el mundo bancario y financiero. Entre otras ventajas permite conocer en todo momento el estado de su cuenta corriente, compra-venta de acciones u otros productos, mensajería (e-mail)... Ver Ibertex e Iberpac.

Virus Programa, normalmente de pequeño tamaño, hecho por programadores expertos y sin escrúpulos que introducen en ordenadores para que actúen de forma anómala (borrar datos, sacar mensajes en pantalla,...). Pueden afectar a ficheros, discos duros, instalarse en memoria y duplicarse a sí mismos. Ver Programa, Software.

Von Neumann Científico americano que dió su nombre a la llamada Máquina o Modelo de Von Neumann que tiene las siguientes características.

- Unidades principales: Unidad de Control (UC), Unidad Aritmético-Lógica (ALU), Memoria y Periféricos de Entrada/Salida.
- Los programas y los datos comparten siempre la misma memoria. De ahí nace el concepto de Programa Almacenado (en memoria).

- Ejecución: La UC y la ALU (Procesador), determinan qué acciones ejecutar leyendo las instrucciones del programa de la memoria una tras otra, manteniendo un Contador de Programa que nos indica la instrucción siguiente a ejecutar.

Actualmente, la mayoría de los ordenadores son máquinas de Von Neumann. Ver Programa, Aplicaciones, CPU.

WAN Wide Area Net: Red de area amplia. Redes de ordenadores con cobertura mundial. Las más famosas son FidoNet (de aficionados) e InterNet (para investigación y comercio, sobre todo entre universidades). Ver FidoNet, Niveles FidoNet, LAN, MAN, Red de ordenadores, InterNet.

Windows Familia de S.O. de MicroSoft. El más famoso es el Windows95 multitarea de 32 bits que une todas las ventajas (y algunos inconvenientes) de MS-DOS versión 6.2, Windows 3.1, Windows para trabajo en grupo 3.11 y algunas cualidades de Windows NT 3.5. El Windows 3.1 ó 3.11 no es un S.O. propiamente dicho sino un interfaz gráfico para MS-DOS con algunas cualidades de un sistema operativo. Ver Sistemas Operativos, MS-DOS. Ver el tema de Sistemas Operativos para más información.

WYSIWYG What You See Is What You Get: Lo que tú ves es lo que obtienes. Filosofía de diseño de programas en los que lo que se ve en el monitor (pantalla) adopta la misma forma del producto que se obtendrá al final (al imprimirlo...). Ver Aplicaciones.

XENIX S.O. que respeta la definición del interface de UNIX y de ejecución en ordenadores personales. Hay otros como Coherent, LINUX... Ver UNIX, LINUX.

Bibliografía

[1] J.C. Torres A. Prieto, A. Lloris. *Introducción a la Informática.* McGraw-Hill, 1995.

[2] L.J. Aguilar. *Metodología de la Proramación. Diagramas de Flujo, Algoritmos y Programación Estructurada.* McGraw-Hill, 1988.

[3] D. Appleby. *Programing Languages, Paradigm and Practice.* Marymount College. McGraw-Hill, 1991.

[4] J.D. Hullman A.V. Aho, J.E. Hopcrogt. *Estructuras de Datos y Algoritmos.* Iberoamericana, 1990.

[5] P. Bihop. *Conceptos de Informática.* Anaya Multimedia.

[6] M. García E. Alcalde. *Metodología de la Programación. Aplicaciones en BASIC, COBOL y Pascal.* McGraw-Hill, 1988.

[7] M. García E. Alcalde. *Informática Básica.* McGraw-Hill, 1994.

[8] C.G. Guy. *Data Comminucations for Engineers.* MacMillan, 1992.

[9] R. Stout H. Hahn. *Internet. Manual de Referencia.* Osborne/McGraw-Hill, 1994.

[10] R. Martínez Tomás J.F. García de Sola. *Informática Básica.* Alhambra Longman, 1993.

[11] R.B. Bunt J.P. Trembley. *Introducción a la Ciencia de las Computadoras. Efoque Algorítmico.* McGraw-Hill, 1988.

[12] C. Weems N. Dale. *Pascal.* McGraw-Hill, 1989.

[13] S.C. Lilly N. Dale. *Pascal y Estructuras de Datos.* McGraw-Hill, 1989.

[14] J.S. Parrilla. *S.O. y Compiladores.* McGraw-Hill.

[15] R.S. Pressman. *Ingeniería del Software. Un Enfoque Práctico.* McGraw-Hill, 1990.

[16] F. Cuellar R. Ale. *Teleinformática.* McGraw-Hill, 1993.

[17] T. Sheldon. *Novel NetWare 386. Manual de Referencia.* McGraw-Hill, 1993.

[18] T. Sheldon. *Novel NetWare 4. Manual de Referencia.* McGraw-Hill, 1994.

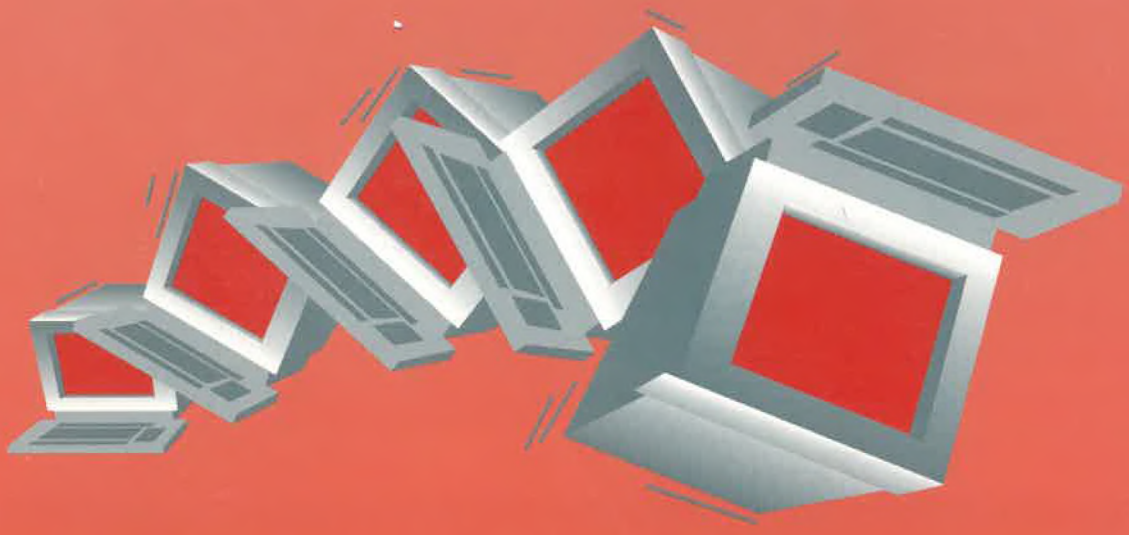
[19] Andrew S. Tanenbaum. *Computer Networks.* Prentice-Hall, 1991.

[20] A.S. Tanenbaum. *Sistemas Operativos: Diseño e Implementación.* Prentice-Hall, 1993.

[21] U.N.E.D. *Informática Básica.* U.N.E.D., 1992.

[22] N. With. *Algoritmos + Estructuras de Datos = Programas.* Ediciones del Castillo S.A., 1984.

Índice de Materias



SERVICIO·DE·PUBLICACIONES

UNIVERSIDAD·DE·CADIZ

1997

ISBN 84-7786-371-7

9 788477 863717