CARACTERÍSTICAS DE LOS SISTEMAS OPERATIVOS PARA LOS MICRPROCESADORES AVANZADOS

4

4.1- CONCEPTOS BÁSICOS SOBRE SISTEMAS OPERATIVOS.

Definimos sistema operativo como un conjunto de programas implementados tanto en software como en firmware que hacen asequible el hardware, de forma que lo hacen disponible de la manera más adecuada al usuario y **aumentan su rendimiento total**, es decir, su rendimiento específico o **throughput** (volumen de trabajo por unidad de tiempo) y su disponibilidad (tiempo de respuesta mínimo).

Un ordenador no es una máquina dedicada sino que esta construido para un propósito general con el fin de ejecutar aplicaciones de diversa índole. En líneas generales el ordenador dispone además de una unidad central de proceso y de la memoria, de diversos periféricos (pantallas, impresoras, teclados, etc) que son compartidos en gran medida por todas las aplicaciones porque todas las aplicaciones de un computador utilizan los mismos recursos físicos y los mismos recursos lógicos.

Los **sistemas operativos nacen** porque sería extremadamente complejo que cada una de las aplicaciones soportadas por el ordenador contuviese todos los programas necesarios para manejar el hardware y el software que comparten muchas aplicaciones.

Además el sistema operativo hace de intermediario entre el hardware y el usuario creando un entorno adecuado que se adapte a los diferentes usuarios y aplicaciones.

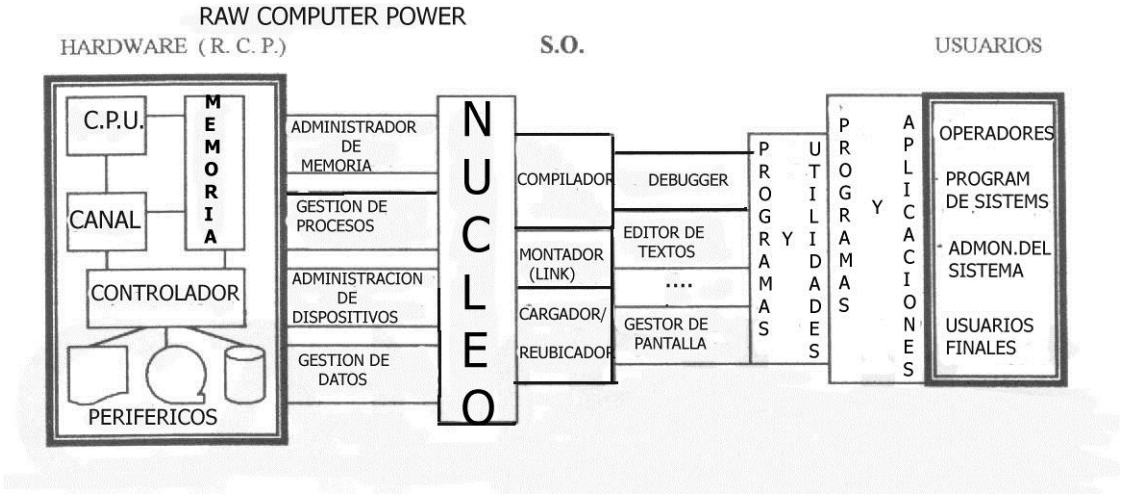

Figura 4.1. Descripción interna del S.O.

Por tanto **el sistema operativo** es un **gestor de recursos**, tratando de sacar de ellos el mayor rendimiento posible, por lo que debe poseer las siguientes **funciones**; **organiza** los archivos en diversos dispositivos de almacenamiento; **supervisa y gestiona** ejecuciones, **errores** hardware y la **pérdida** de datos; **coordina y manipula** el hardware de la computadora y tiene que **evitar** usos inadecuados además de **responder** a cualquier evento que se produzca.

4.1.1- Clasificación de los sistemas operativos.

Existen muchas clasificaciones diferentes de sistemas operativos dependiendo del criterio en el que se basen. Nosotros los clasificaremos basándonos en dos criterios:

1º- Según el punto de vista del usuario:

Existen los **SISTEMAS MONOUSUARIO** en los cuales la máquina virtual (la presentada por el S.O. al usuario) tiene un solo usuario y generalmente está dedicada a una sola función. Por tanto el interface del S.O. constará básicamente de un **gestor de ficheros** sencillo, utilidades que proporcionen **facilidades de E/S** y un intérprete de comandos sencillo, es decir un **OSCL**(Lenguaje de Control del S.O.); y sus **cualidades** son fiabilidad, eficacia, sencillez de uso y facilidad de extensión.

Se puede representar de las siguientes maneras según el modo en que se trabaje:

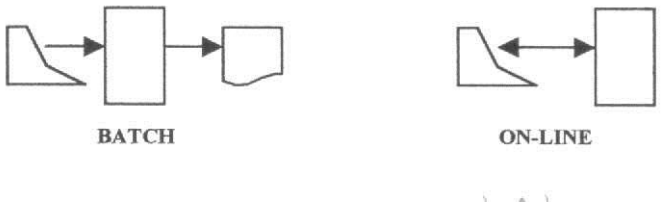

Figura 4.2. Representaciones Batch y On-Line de sistemas monousuario.

También existen los **SISTEMAS DE TIEMPO REAL** que tienen en común la necesidad de dar respuestas a unas entradas con un tiempo de proceso informático limitado**.** Por los que sus **funciones** principales serán interactuar con dispositivos externos (sensores, válvulas,...), reacción inmediata ante cualquier suceso externo, registrar la información (fichero registro), tener noción de tratamientos prioritarios y realizar una planificación eficiente, siempre teniendo en cuenta el tiempo físico (tiempo real)

Su **cualidad** principal debe ser la fiabilidad de forma que garantice la seguridad del sistema en cualquier caso (tolerancia a fallos) debiendo ofrecer unos servicios mínimos (disponibilidad) Los sistemas en tiempo real se utilizan en muchos campos como gestión de centralitas telefónicas, pilotaje de aviones, seguimiento de trayectorias, vigilancia médica, control de robots, ..., y control de procesos en general..

En general responden al siguiente esquema:

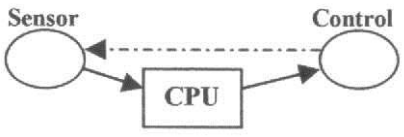

Figura 4.3. Esquema de un sistema de tiempo real.

Como evolución de los sistemas operativos aparecieron los **SISTEMAS TRANSACCIONALES** que se caracterizan por poseer las siguientes **funciones**; gestionar un gran volumen de información (Bases de Datos) desde distintos y numerosos puntos de acceso y de transacciones desarrollándose simultáneamente (de forma paralela → concurrencia) y por la ejecución de operaciones predefinidas, muy a menudo interactivas.

Sus **cualidades** son fiabilidad y disponibilidad.(servicios mínimos)

Son ejemplos de sistemas transaccionales los sistemas de reservas de plazas en líneas aéreas, gestión de cuentas bancarias, consulta de documentos, etc...

Más tarde aparecieron los **SISTEMAS TS/MULTIPROGRAMADOS**, donde **los sistemas Time-Sharing** se caracterizan por prestar servicio a un conjunto de usuarios simultáneamente, dividiendo el tiempo de utilización de la máquina en quantums para cada usuario.

Mientras que **los sistemas multiprogramados** se caracterizan por la ejecución simultanea de varios programas (procesamiento concurrente). En ambos sistemas el S.O. será el encargado de conmutar los recursos entre los diversos procesos y/o usuarios.

Por tanto las **funciones** que debe proporcionar el S.O. serán todas las de los sistemas monousuario y las de los sistemas transaccionales: definición de la máquina virtual a presentar a cada usuario, ofrecer la utilización compartida y asignación de los recursos físicos comunes (conmutación), gestión de la información compartida. Sus **cualidades** serán: disponibilidad, fiabilidad y seguridad, facilidad de extensión y adaptación a los diferentes usuarios, facilidad de uso y eficacia.

También existen los **SISTEMAS MULTIPROCESADOR** que **s**e caracterizan por la existencia de varios procesadores centrales compartiendo a veces memoria y periféricos en la ejecución de instrucciones en paralelo (concurrencia real).

Sus **funciones** son todas las de los sistemas TS/Mutiprogramados, las de los sistemas de tiempo real y además las de gestión de los 'n' procesadores: asignación, conmutación, comunicación, etc... Por otro lado, si estos procesadores están a cierta distancia y se comunican por las líneas de transmisión de datos (sistemas distribuidos) tendrán como función la gestión teleinformática.

Sus **cualidades** serán todas las de los tipos anteriores, pero haciendo especial hincapié en la fiabilidad.

2º- Según la arquitectura del computador:

Existen los **SISTEMAS MONOLÍTICOS** que están diseñados como un conjunto de rutinas (rutinas de servicio y rutinas de utilidad o auxiliares) compiladas por separado, que pueden llamarse entre sí y que se montan para formar el S.O., que será la rutina principal (gestor de interrupciones).

El **principal problema** de los sistemas operativos monolíticos es su diseño poco apropiado para sistemas operativos grandes, ya que es muy difícil de especificar, codificar, probar y depurar. La solución sería introducir estructura en el S.O.

Como ejemplos de S.O. monolíticos están DOS/VS 360 y UNIX.

También se desarrollaron los **SISTEMAS POR NIVELES O CAPAS** en los cuales los sistemas operativos están diseñados como una jerarquía de niveles, en la que cada nivel hace uso de las facilidades que le proporcionan el hard y los niveles inferiores a él, sin necesidad de conocer los detalles de operación de dichos niveles.

Por tanto **el S.O**. está constituido por una serie de **módulos pequeños** y fáciles de manejar, cada uno de ellos especializado en una única función y con interfaces con otros módulos.(**ocultamiento de información**) Son ejemplos de S.O. por niveles MINIX y THE :

La principal ventaja de estos sistemas es su facilidad de mantenimiento que incrementa la fiabilidad del sistema, y su principal problema reside en la dificultad de elegir las funciones que deben asignarse a cada nivel.

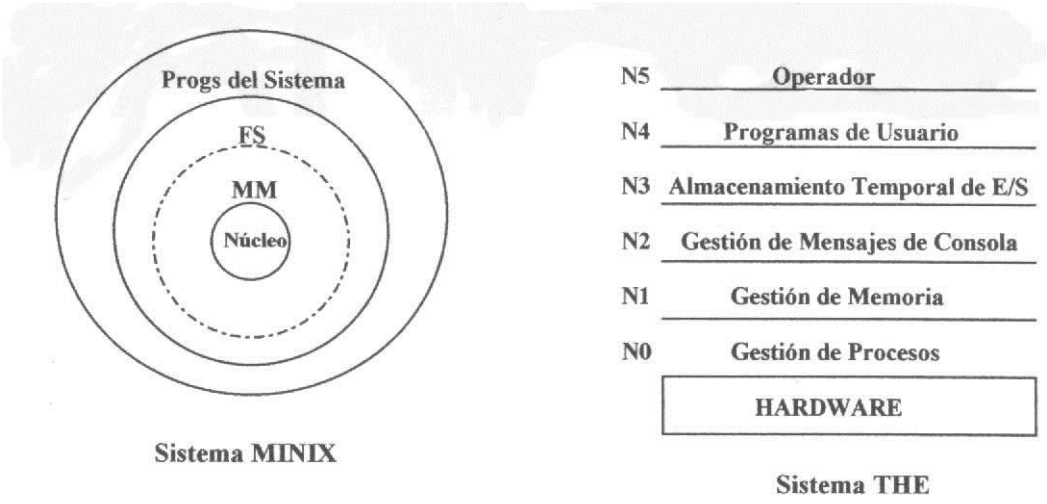

Figura 4.4. Representación gráfica de un sistema por capas

Como extensión del enfoque de capas surgieron los **SISTEMAS DE MÁQUINA VIRTUAL** en los cuales se separa la parte del S.O. que proporciona la máquina virtual (la máquina adaptada al usuario) de la parte que posibilita la compartición de recursos.

Se pueden instalar diferentes S.O.s sobre las máquinas virtuales, lo cual proporciona gran flexibilidad al sistema y la posibilidad de experimentar y reducir tiempos de desarrollo con nuevo S.O.s sin interferir con el trabajo habitual.

Ejemplo : VM de IBM

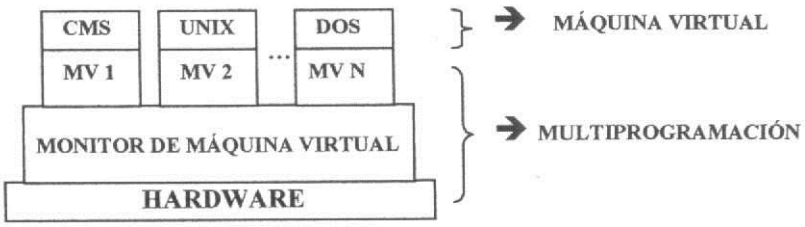

Figura 4.5. Representación gráfica de un sistema máquina Virtual VM de IBM

El **principal problema** de estos sistemas es la dificultad de implementación.(proporcionar duplicados exactos de las máquinas)

Por último también existen los **SISTEMAS BASADOS EN MICROKERNELS** que incluyen en el núcleo sólo las funciones esenciales del S.O., estando el resto de funciones implementadas en módulos a nivel de usuario sobre el núcleo, que hace de base común para los procesos de usuario y el resto de servicios del S.O.

Por tanto estos sistemas tienen un núcleo mínimo pero con un número suficiente de mecanismos eficientes, que le capacitan básicamente para gestionar mensajes y controlar los accesos al hardware.

Las **principales ventajas de este tipo de S.O.s** son su fácil adaptabilidad a los sistemas distribuidos; su alta fiabilidad (por tener un núcleo pequeño); su facilidad de idear; implementar y mantener; además de su facilidad con la que los servicios implementados pueden modificarse, recompilarse y cargarse sin afectar al resto.

La **principal dificultad** consiste en decidir qué funciones deben incluirse en el núcleo: las que necesiten rapidez o deban ejecutarse muy a menudo. En general habrá que lograr un equilibrio entre servicios rápidos e inflexibles, ya que cuanto menos se sobrecargue el núcleo más fácil será su mantenimiento.

4.1.2-Requisitos del sistema operativo multitarea.

Los sistemas operativos que soportan la multitarea manejan una lista de tareas pendientes de ser procesadas, que se ordena y organiza en función de diversos criterios, o bien, como sucede en los sistemas operativos de tiempo real, de acuerdo con la aparición de factores externos.

Para ello debe poseer **ciertos requisitos**. Un requisito imprescindible para la multitarea es la asignación de una zona exclusiva de memoria a cada tarea, en la que se almacenan su código y sus datos. Dicha zona de memoria recibe el nombre de **área local.**

Por otra parte, también deberá existir una zona de memoria común y compartida por todas las tareas, a la cual se denomina **área global.**

Cada tarea tendrá **derecho a acceder** a su propia área local y al área global; pero tendrá prohibido el acceso a las áreas locales de las restantes tareas.

En cada momento, el sistema operativo pone a disposición de la tarea en curso los recursos de la máquina, incluyendo los espacios de memoria adecuados.

Como se ha mencionado anteriormente, cada tarea tiene un **tiempo asociado para su ejecución ("quantum")**, pasado este tiempo, el sistema operativo provoca una conmutación de tareas poniendo la máquina a disposición de otra tarea.

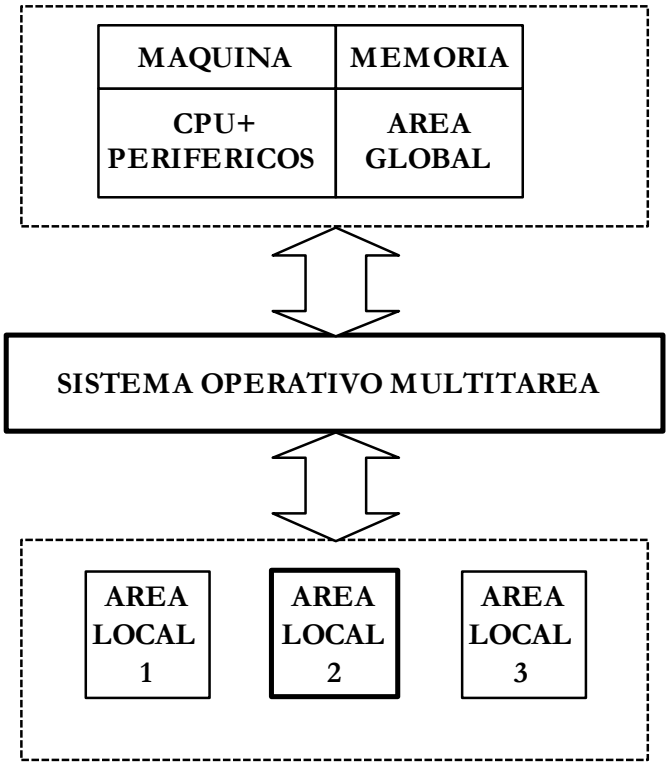

Figura 4.6. Representa un sistema operativo multitarea.

4.2-DÉSARROLLO HISTÓRICO DEL HARDWARE DE LOS SISTEMAS OPERATIVOS

4.2.1- FASE 0: AMBIENTE MANUAL E INTERACTIVO DIRECTO

Inicialmente los ordenadores no tenían S.O. por lo que cada vez que se quería que el hardware hiciese algo, el "usuario" debía reservar previamente su tiempo de uso de ordenador (**método de reservas de solicitud**), luego tenía que codificar todas las instrucciones del programa (en forma de 0s y 1s), cargarlo mediante interruptores y seguir la ejecución mediante lucecitas en la consola **(ejecución paso a paso**).

Evidentemente esto suponía muchísimo trabajo y una altísima probabilidad de cometer errores, por lo que se hacía necesario reducir las diferencias entre el lenguaje máquina y el lenguaje humano. La solución vino con el desarrollo de un software básico (componentes) y de nuevos dispositivos.

4.2.2- FASE 1: DESARROLLO DE COMPONENTES

Esta fase se caracteriza por el desarrollo de componentes como ensambladores, compiladores y herramientas que faciliten la puesta a punto de los programas (depuradores, etc...). Así los programas eran más inteligibles y el número de errores disminuyó considerablemente.

Luego, para evitar tener que codificar u montón de instrucciones similares cada vez que había que hacer una operación de E/S, codificaron unas rutinas de E/S (*device/driver*) con parámetros que se cargaban en memoria con los programas, los cuales sólo tenían que llamarlas pasándolas los parámetros adecuados. Estas rutinas de E/S se encargaban de sincronizar la E/S de datos con la CPU, de la conmutación automática de Ioáreas y detectaban algunos tipos de errores; Por tanto gestionaban buffers, flags, registros, bits de control, bits de estado, etc...

Otro componente que se desarrolló en esta fase fueron los programas de ayuda, como generadores de listados, cargadores de programas, programas de volcado de memoria (DUMP), combinadores de programas,(MERGE)... . Además surgió la idea de utilizar una biblioteca central de programas.

Con todo esto se consiguió aumentar un poco el rendimiento y facilitar la programación, pero la operación era muy compleja (**modo operativo de puerta abierta**) y se seguía haciendo un uso antieconómico de un material muy costoso, por lo que la siguiente fase tendría como principal objetivo aumentar el rendimiento reduciendo el tiempo de preparación.("Set-up time")

4.2.3.- FASE 2: OPERADOR / MONITORES RESIDENTES

Lo primero que se hizo para **aumentar el rendimiento** fue separar los tipos de trabajos naciendo así la figura del **operador**, que era un profesional especializado en operar con el sistema, por lo que era más rápido y cometía menos errores, desapareciendo así la puerta abierta y la inactividad por método de reserva de ordenador. Además, el operador solía clasificar los trabajos para aumentar la productividad (proceso por lotes (*BATCH*)).

No obstante se observó que la diferencia entre el tiempo de ejecución y el de preparación era abismal, así que se desarrolló **un programa que hiciera el trabajo del operador** (el secuenciamiento automático de trabajos) llamado **programa de control**, para el cual se desarrollo un lenguaje de control de trabajos (*OSCL*) con su correspondiente intérprete.

En torno al programa de control se agruparon los componentes desarrollados en la fase anterior, dando lugar a los *primeros sistemas operativos* conocidos como **monitores de encadenamiento o residentes**, responsables de gestionar y proteger los recursos máquina, y cuyas funciones eran supervisar las operaciones de E/S, proteger la memoria reservada al S.O., limitar el tiempo de ocupación de la CPU por los programas (par lo cual tenían que gestionar un reloj), asegurar el flujo continuo de trabajo, etc...

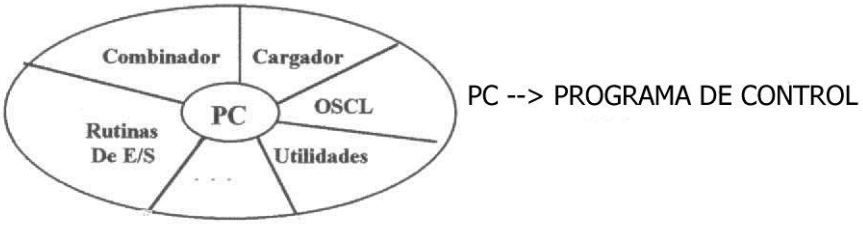

Figura 4.7. Representación gráfica de un monitor de encadenamiento

4.2.4.-FASE 3: DESARROLLO DE LA TECNOLOGÍA HARDWARE

El objetivo inicial en esta fase es disminuir el tiempo que la CPU emplea en las operaciones de E/S, ya que los periféricos son mucho más lentos al trabajar a velocidades electromecánicas.

- **FASE 3a**

Se desarrollan procesadores satélites u OFF-LINE, que eran unos dispositivos específicos (aparte del procesador principal) que permitían la **operación en OFF-LINE** al reemplazar las lectoras de tarjetas y las impresoras de líneas por unidades de cinta magnética.

Para conseguir una cierta **independencia de los dispositivos**, los programas trabajaban con dispositivos lógicos, siendo el S.O. el encargado de determinar los dispositivos físicos mediante el OSCL. Surge así la posibilidad de multiprocesamiento y la posibilidad de usar múltiples sistemas de grabación en cinta y de descarga a impresora.

Además se desarrollaron unos procesadores autónomos especializados en la transferencia de información entre memoria y periféricos, llamados **canales**, que se ocupaban de las operaciones de E/S. Así la CPU queda libre durante las operaciones de E/S, por lo que pensaron en cómo ocuparla, y así surgieron las E/Ss con **memorias intermedias** (buffers) de los que la CPU cogería los datos (leídos masivamente de antemano) para depositarlos en memoria intermedia también tras su proceso. De esta manera se conseguía solapar la E/S de los datos con su procesamiento.

Pero el *buffering* presentaba una serie de problemas como la dificultad de programarlo, la posibilidad de colapsar la memoria, no servía para trabajos intensivos de E/S y seguía sin poderse mantener ocupados a la CPU y a los dispositivos permanentemente, en parte por las limitaciones de la época (cintas) que eran secuenciales.

Aprovechando el nacimiento de los discos (**DAAD**) se encontró la solución a estos problemas trabajando con una memoria a dos niveles: técnica de SPOOL, que consistía en utilizar memoria secundaria como un gran buffer, evitando así que se colapsara la memoria central y consiguiendo un solapamiento de actividades.

Dado que quien tenía que gestionar esta memoria a dos niveles era el S.O., los monitores de encadenamiento tuvieron que extenderse dando lugar a los **Ejecutores** o **Ejecutivos**, los cuales, además de gestionar varios tipos de memoria, facilitaban el uso eficiente de los canales e incluían mecanismos completos de sincronización y manejo de interrupciones.

- **FASE 3b**

El objetivo de esta fase sería reducir los tiempos muertos de la CPU para aprovecharla al máximo y aumentar el rendimiento, y la solución vino con la **multiprogramación**: se reparte la memoria entre varios procesos, de forma que cuando el proceso en ejecución no podía seguir se conmutaba a otro proceso y así sucesivamente. Esta labor de conmutar el procesador central entre los diversos procesos la llevaba a cabo el **planificador de bajo nivel** o "dispatcher".

De esta manera se incrementaba considerablemente el rendimiento específico e incluso la disponibilidad, pero hubo que aumentar también la memoria y ampliar el S.O. para que incluyera al dispatcher y se ocupara de la gestión, protección y compartición de la memoria y de la asignación y ubicación de recursos que hicieran posible la concurrencia.

Paralelamente a la multiprogramación, surgieron las compañías de líneas aéreas con unas necesidades de tratamiento informático nuevas: básicamente necesitaban tratar un fichero maestro de plazas de forma ON-LINE desde varios puntos y con un tiempo de respuesta aceptable. La solución a este problema vino dada por una combinación de hard y soft: En cuanto al hardware se desarrollaron CPUs más rápidas, nuevos periféricos (terminales) y sistemas de comunicación; y en cuanto al software se desarrolló un programa maestro capaz de atender a varias peticiones de información simultáneas (procesado concurrente).

Y así nacieron los **Sistemas Transaccionales**:

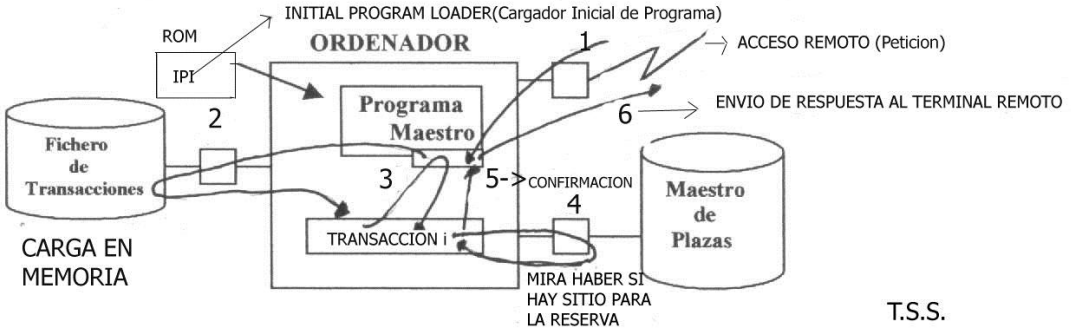

Figura 4.8. Representación de un sistema transaccional y su funcionamiento

4.2.5.-FASE 4: CRECIMIENTO DE LAS PRESTACIONES DE LOS MICROPROCESADORES

En esta fase se desarrollaron técnicas de transmisión de datos (*teleinformática*) y la progresiva integración de la función de comunicación en los sistemas (*telemática*).

De esta manera surgen las necesidades de compartir recursos y de adecuar la estructura de los sistemas a las aplicaciones que tratan (*descentralización*). La solución vino con el desarrollo de las **redes de teleinformática** (S.Distribuidos), de las **redes locales** y de los **sistemas multiprocesador** que permiten el procesamiento en paralelo (concurrencia real y aparente).

Para soportar las redes se desarrollaron los Servidores como módulos incluidos en el S.O. encargados de controlar las comunicaciones, obteniéndose S.O.s multiformes de propósito general.

DIAGRAMA DE LA EVOLUCIÓN HISTÓRICA

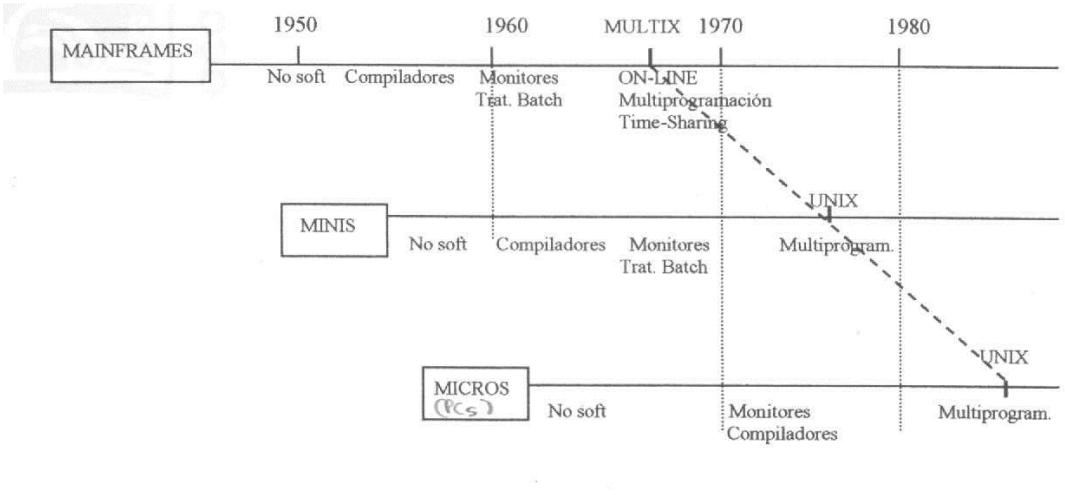

Figura 4.9. Esquema de la evolución histórica de los S.O.

Ahora las tendencias actuales son las siguientes:

• **USER-FRIENDLY**: Se tiende a que la máquina sea lo más "amistosa" posible, por lo que el interface presentado por el S.O. proporciona un acceso fácil y un manejo sencillo y guiado, vía menús, mensajes de ayuda e iconos, unido a una homogeneidad en el aspecto de las diferentes aplicaciones (*look & feel*).

• **MÁQUINA VIRTUAL**: El usuario no necesita conocer la máquina física sino la que le presenta el S.O.

• **PROCESAMIENTO DISTRIBUIDO**: Varios computadores independientes interconectados entre sí, cooperando e intercambiando información y con su propio procesamiento local (tolerancia a fallos, redundancia,...), S.O.s dispersos, disminución del coste de comunicación y mayor velocidad de transmisión.

• **PROCESAMIENTO PARALELO**: Aumento de la escala de integración y de la capacidad de procesamiento y almacenamiento, máquinas masivamente paralelas, lenguajes concurrentes y arquitecturas que distribuyen el control entre varias CPUs.

4.3 CARACTERÍSTICAS ESPECÍFICAS DE LOS MICROPROCESADORES AVANZADOS

Los microprocesadores avanzados provocaron un aumento del rendimiento en el procesamiento que se debe en gran parte a su configuración de su estructura interna también influyó la base de registros y a buses de 32 bits, además a que la mayoría de la cascada de instrucciones (pipeline) y por supuesto al aumento de las frecuencias de trabajo y el uso de memorias ultrarrápidas.

Estas **mejora**s se relegan a un segundo plano debido a los avances que se producen en la arquitectura de microprocesadores avanzados y la utilización de multitarea; aunque exigiendo la multitarea **aportaciones** de ciertos requisitos como que la CPU sea muy veloz sumándole una gran capacidad de memoria para soportar la multitarea y por último pero muy importante un **sistema de protección** que restringa los accesos privilegiados a las tareas y evite errores de protección de tareas tanto en área global como local.

Para obtener memoria de gran capacidad se emplea la técnica de memoria virtual. Que explicaremos a continuación pero que hace referencia a que el programador o el usuario piense que posee más memoria de la que realmente posee y puede utilizar.

El sistema de protección para soporte multitarea exige la inclusión en la arquitectura del procesador de dispositivos capaces de proporcionar al sistema operativo multitarea los requerimientos que precisa de la máquina.

Por lo que esto se refiere a que es más importante que el aumento de potencias, velocidad,..el trabajar en entornos multitarea, multiusuario y en tiempo real, basándose estas prestaciones en cuatro aspectos:

1. **Memoria Virtual:** Método de organización y gestión de la memoria que proporciona al programador de aplicaciones un espacio mucho mayor de que dispone físicamente, da la ilusión a la CPU de que puede manejar mucha más memoria de la que la DRAM tiene a su disposición.

2. **Sistemas Multitarea:** Dispone de los recursos físicos(registro de tarea, TSS) para producir un cambio o conmutación de tareas rápido y seguro.

3. **Sistemas de protección:** Es un sistema que controla el acceso de las tareas a memoria. Que los segmentos de datos y de código sólo puedan acceder a segmentos accesibles por ellos, SÓLO de la misma tarea y del área global.

4. **Subsistemas de memoria caché:** Es una memoria ultrarrápida para las CPUs que trabajan a frecuencias medidas en GHz, normalmente oculta.

4.4 MEMORIA VIRTUAL

Definimos memoria virtual como conjunto de programas que tienen el sistema operativo que hacen creer a las CPU y por consiguiente, a los usuarios/programadores que pueden manejar directamente los discos aunque en la realidad sólo pueda acceder a la memoria electrónica (DRAM).

Nace la memoria virtual porque la CPU sólo es capaz de acceder directamente a una memoria principal o física cuya capacidad está limitada por el tamaño del bus de direcciones, y por ello cuando el procesador intenta acceder a memoria es realmente poco para el procesador.

Hay que tener en cuenta que en la memoria principal residen los datos y las instrucciones que manipulan la CPU directamente, a través de los buses, entre los dos elementos básicos del ordenador y que dicha memoria debe estar constituida con una tecnología similar a la de la CPU para que los tiempos de trabajo(**búsqueda de instrucciones y ejecución de CPU**) sean similares, (circuitos integrados CMOS rápidos)siendo el rendimiento de la máquina óptimo.

Las memorias RAM y ROM siguen siendo caras si las comparamos con dispositivos de almacenamiento magnéticos.

La CPU está limitada por el tamaño del bus de direcciones aunque éste haya evolucionado como aquí se indica:

	8085	Pentium
Número de bits	8	32
Líneas de bus direcciones	16	32(mínimo)
Tamaño memoria ppal.	64KB	4GB

Para **utilizar la memoria virtual** el procesador deberá realizar una serie de comprobaciones y pasos, que son los siguientes:

1. Genera la dirección del objeto que necesita, un mecanismo de gestión de memoria, denominado Unidad de Gestión de Memoria (MMU) comprueba si el objeto se encuentra en memoria principal

2. Si está en memoria principal accede a él normalmente.

3. En caso contrario, comunica el hecho al sistema operativo que pone en marcha la rutina encargada de localizar el objeto en memoria virtual (disco) y transferirlo a la memoria principal para que la CPU acceda normalmente a él.

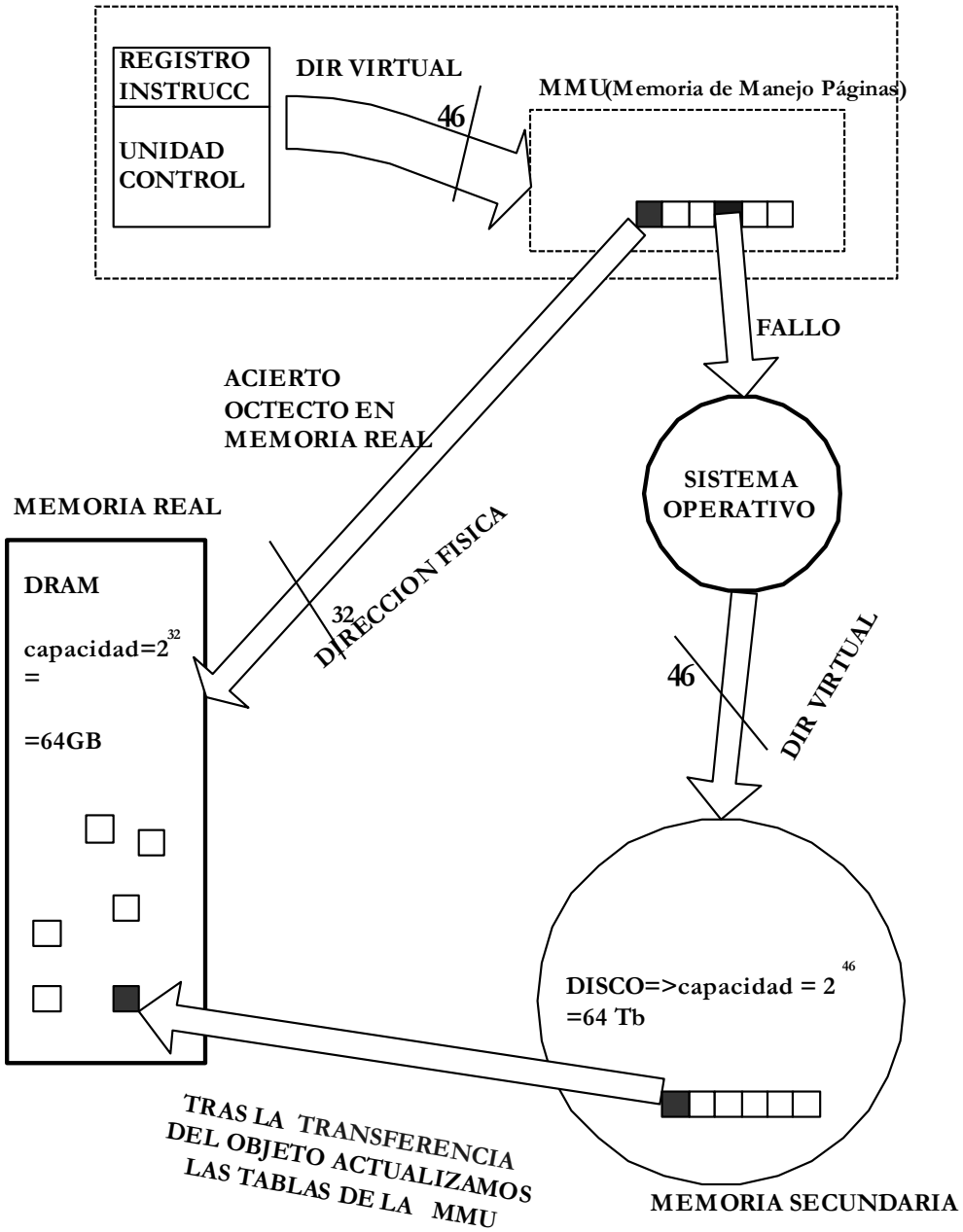

Figura 4.10. Representación de una traducción en memoria virtual

Como la memoria virtual es mucho mayor que la principal, existirá un constante flujo de transferencia entre ambas.

La memoria virtual permite que las aplicaciones ocupen mucho más espacio que el disponible en la memoria principal. Además si los **algoritmos de transferencias** son buenos esto implicará un aumento de accesos directos a la CPU implica **aumento de velocidad**; debido a que las posiciones que precisa la CPU se anticiparán.

La implantación de la memoria virtual conlleva la incorporación de hardware que configura la MMU que **facilita la comunicación con el sistema operativo** que debe disponer de las siguientes **tres funciones**.

La **primera** de ellas la posesión de una lista completa con la descripción de todos los objetos residentes tanto en memoria virtual como en la principal. La **segunda** de ellas es la carga de objetos en al memoria física en tiempo de ejecución.

Y por último, la **tercera**, el manejo de los espacios libres de la memoria física si no hay espacios libres decidir con un algoritmo cuál sustituimos en la memoria principal, para insertar la posición necesitada. Además al sacar la posición de memoria principal habrá que comprobar si ha sido modificada para antes de sustituirla modificarla en la memoria virtual o si no sustituirla directamente.

Existen dos formas de organizar la memoria virtual.(**Segmentación** y **Paginación**)

Una de ellas, la **segmentación** que es la división de la memoria virtual en "trozos"(segmentos) de tamaños variables. Los distintos segmentos pueden crecer o reducirse en forma independiente sin afectar a los demás, esto hace más sencilla la administración de las estructuras de datos que crecen ó se reducen, si cada procedimiento ocupa un segmento independiente con la posición inicial cero el ligado independiente de los procesos compilados es mucho más sencillo.

Otra forma de división es la **paginación** que es la división del espacio de direcciones de cada proceso en bloques de tamaño uniforme llamados páginas(1K,2K,4K ó4MB), los cuales se pueden colocar dentro de cualquier página marco disponible en memoria.(Se desaprovecha memoria), cuando las tablas de páginas son muy grandes se puede utilizar un esquema de paginación de varios niveles que las páginas se paginen a sí mismas.

En Intel la memoria virtual es de 64 Tb.

4.5 TIPOS DE MEMORIA VIRTUAL

Los distintos modelos de memoria virtual se diferencian por sus políticas de solape y por métodos que emplean en a organización de la memoria:

Como son la Memoria paginada, la Memoria segmentada, y la Memoria virtual con segmentos paginados. En estos tres casos la fracción de memoria que debe asignarse a un programa es variable en cada caso.

La **política de solape y compartición** debe tener en cuenta ciertas características internas de los programas que, determinan la construcción modular y estructurada de los mismos.

El sistema operativo debe tener en cuenta la fracción de memoria principal que se va a sustituir o cargar en disco así como las modificaciones que existan de lectura y escritura. Los criterios usados son los siguientes

Regla FIFO(First In – First Out): Se sustituye la fracción que más tiempo lleva en la memoria principal para dejar hueco a otra.

Regla LRU(Last Recently Used): La porción que lleva en la memoria más tiempo sin haber sido usada.

Regla LIFO(Last In – First Out): Se sustituye la fracción que menos tiempo lleva en la memoria principal para dejar hueco a otra.

Regla LFU(Last Frecuently Used): La porción que se accedido menos veces desde que se inició el proceso.

Regla RAND(Random): Se elige una porción al azar.

Regla CLOCK: Cuando se coloca un bit de uso en cada entrada de una cola FIFO y se establece un puntero que se convierte en circular. Es una aproximación al algoritmo LRU con una simple cola FIFO.

4.5.1-MEMORIA PAGINADA

Este modelo organiza el espacio(memoria) virtual y el físico en bloques de tamaño fijo llamados páginas.(de 1, 2 4 KB ó 4MB en los últimos Pentium). Los **S.O**. son muy **sencillos** porque los algoritmos de transferencia son muy sencillos, sabemos el tamaño de la sustitución, es decir el tamaño de la página. Pero esto implica que en una página no siempre entra toda la estructura del objeto y a veces el objeto es muy pequeño y la página no se llena, **desaprovechamiento de la memoria** es **clara** si hay **pocos datos** en la memoria. Está claro **no** se **optimiza la memoria.**

Ya que en la realidad como se trabaja con la memoria principal en lugar de con el disco, y se deben usar direcciones físicas se precisan de dos métodos para realizar la traducción virtual a física que son los siguientes:

4.5.1.1- MÉTODO DE CORRESPONDENCIA DIRECTA

El primero de los métodos para traducir la dirección física es el que se denomina método de correspondencia directa.

La dirección virtual para una página se divide en dos campos:

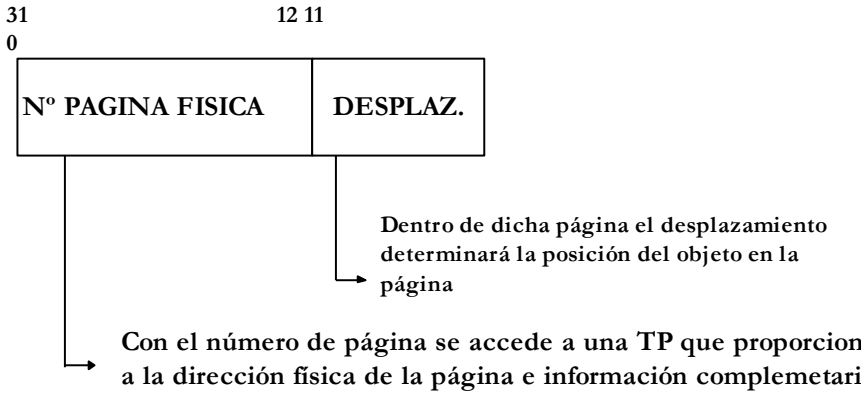

Figura 4.11. Representación de una dirección virtual de página.

La tabla de páginas (TP) tiene tantas posiciones como páginas de memoria virtual hay. La configuración de las **entradas de TP** se muestran en la figura y consta de los siguientes campos:

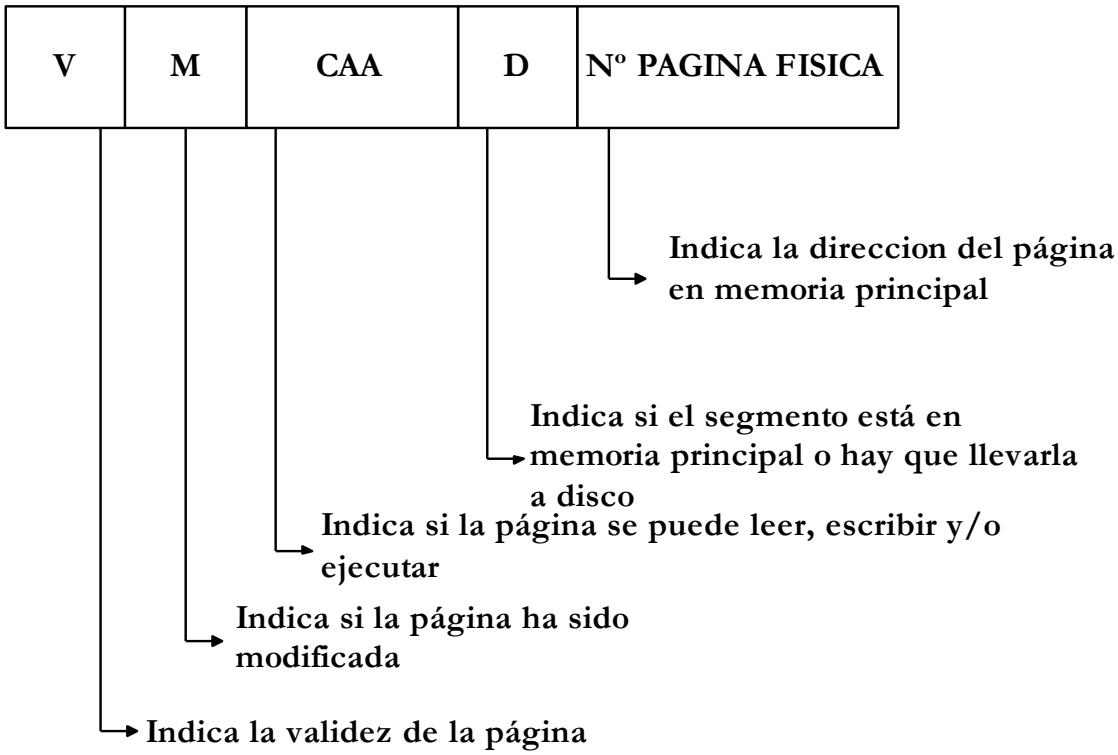

Figura 4.12. Representación de una entrada de la Tabla de Páginas.

La dirección de comienzo de la TP está almacenada en el Registro Base de la Tabla de Páginas(RBTP). Para acceder, a las entradas de la TP, se incrementa el valor correspondiente al número de página virtual (npv). Para calcular la dirección física en memoria se concatena DP con el desplazamiento (d) cuando la página en la memoria principal (Figura 4.13)

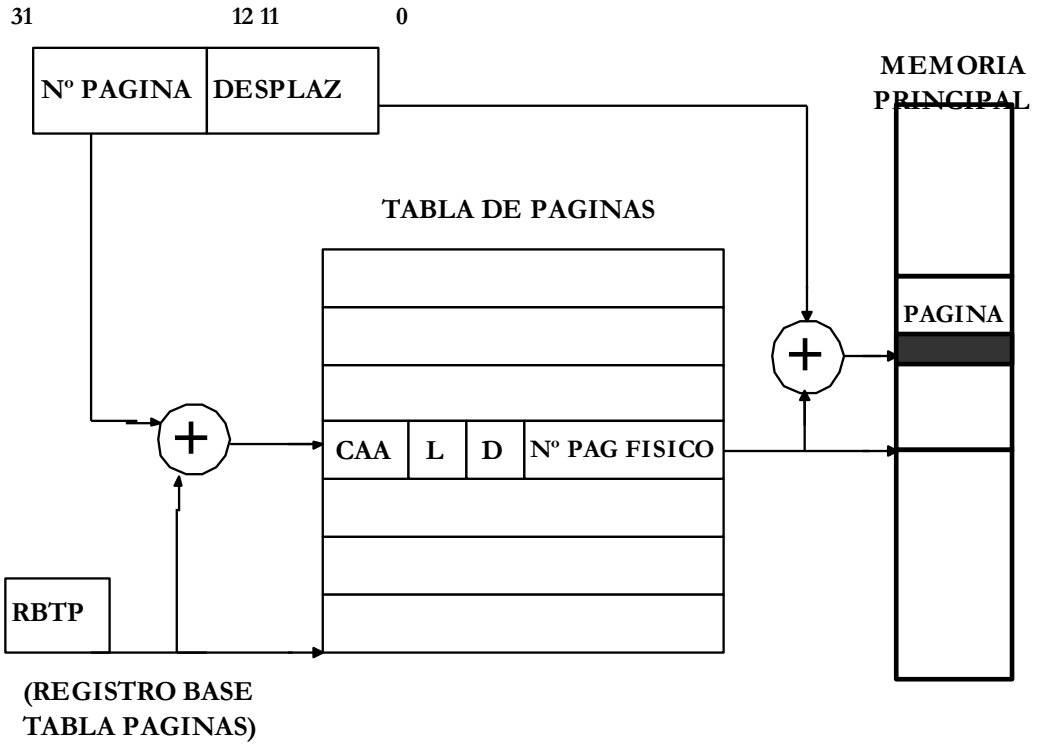

Figura 4.13. Representación del funcionamiento de búsqueda de una página con una Tabla de Páginas

Esta forma de trabaja plantea un **inconveniente** que es que el número de entradas de la TP debe coincidir con el de páginas virtuales, que es muy grande. Para **solucionarlo** se hace uso de la **tabla inversa**, ordenada por el número de página física, que reduce el número de entradas necesarias, pero requiere una traducción con una función de búsqueda. El proceso es muy lento.

4.5.1.2- MÉTODO DE CORRESPONDENCIA ASOCIATIVA

En este caso se dispone de una tabla inversa realizada con tecnología asociativa, esto es, memoria tipo CAM, mucho más rápida porque en lugar de tener la TP en memoria principal se carga en esa memoria especial llamada CAM, que se encarga del proceso de búsqueda a muy alta velocidad, suministrando el número de página física o indicación de que la dirección lógica no se encuentra en memoria. En este último se elimina una página de la memoria principal y se trae la nueva al hueco que deja.

En una memoria asociativa se puede acceder a la vez a todas las posiciones para identificar la que satisface cierto criterio de selección. Dado el elevado coste de las memorias asociativas, la tabla en CAM suele ser incompleta, albergando el conjunto de páginas activas en un momento determinado. Si se origina una falta en la CAM, hay que acudir a la TP para comprobar si está en la memoria principal y, en tal caso, actualizar la CAM. De lo contrario se procederá a un cambio de página.

La memoria CAM se divide en dos partes: Etiqueta y Datos.

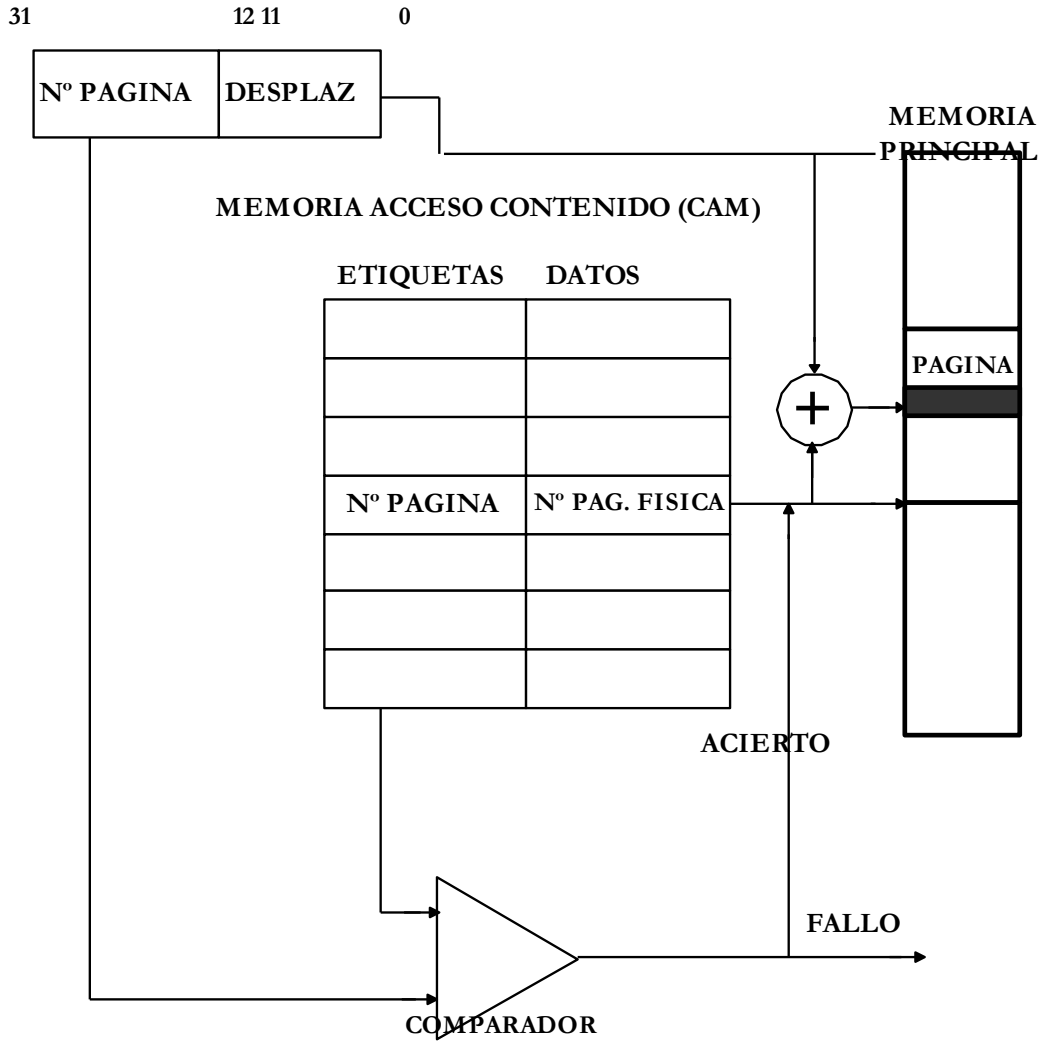

Figura 4.14. Búsqueda de una página mediante una memoria de acceso por contenido.

4.5.2- MEMORIA SEGMENTADA

La memoria se divide en trozos de tamaño variable llamados segmentos. Éstos tienen un tamaño diferente y se adaptan al tamaño del elemento siempre del mismo tipo para cada segmento que deben tener cada segmento; lo cual implica que la memoria virtual está muy bien estructurada.

Ello implica algoritmos de transferencia muy complicados por que no sabe le tamaño del segmento. Por ello necesita saber el tamaño (LIMITE) del segmento para saber si lo podemos introducir o no en la memoria real

Cada TS contiene los siguientes campos:

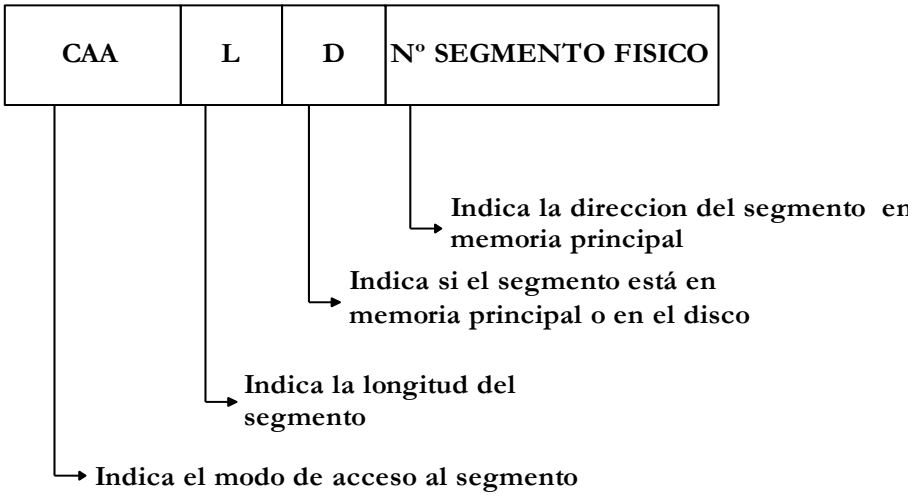

Figura 4.15. Disposición de los campos de una entrada de la TS.

Los elementos de un segmento se identifican con la dirección del segmento al que pertenecen y un desplazamiento dentro del mismo.

A semejanza con el modelo anterior existe una RBTS que direcciona el comienzo el comienzo de la Tabla de Segmentos (TS), de las que existe una por cada proceso activo.

Las TS funcionan de forma similar a las tablas de páginas (TP) antes descritas. Es decir con el número de segmento sumado al RTBS nos da la posición dentro de la tabla de segmentos.

Mientras no haya errores, en esa posición tendremos toda la información de ese segmento gracias a los campos de la tabla de segmentos.

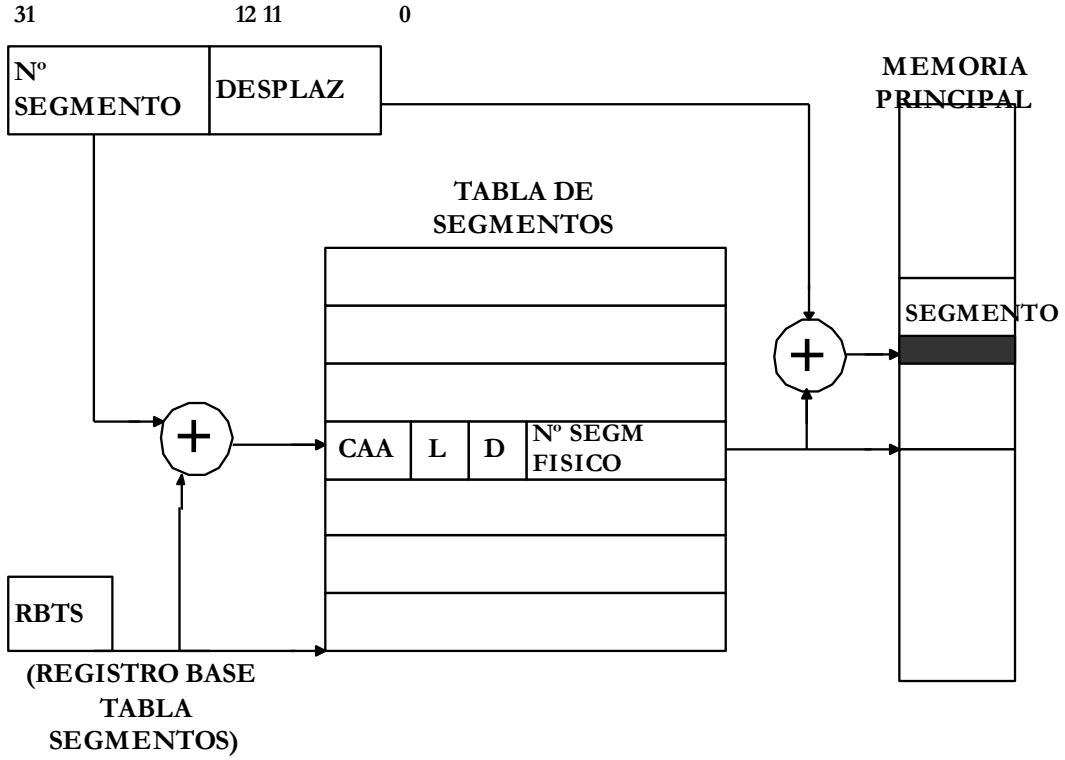

Figura 4.16. . Representación del funcionamiento de búsqueda de una página con una Tabla de Segmentos.

Siendo uno de ellos la dirección absoluta de comienzo de segmento que sumada al desplazamiento nos dará el segmento total y conseguiremos la información.

La utilización de la memoria segmentada implica mejor aprovechamiento de memoria ya que puede ser tan grande o tan pequeña como se quiera, lo que hace que no se desperdicie memoria y se estructure mejor y además proporciona un ajuste a la programación estructurada, utilizada por Intel.

Aunque esto también implica una complicación del sistema operativo, es más lento y con problemas.

Dado que la longitud de los segmentos es variable se precisa algún algoritmo que localice un espacio libre para que resida el segmento apropiado, ya que no es corriente encontrar un bloque continuo y vacío en memoria, para colocarlo por completo.

Estos algoritmos forman parte de un mecanismo que se pone en marcha cuando se detecta una falta de segmento, siendo los más relevantes: **De mejor ajuste:** Minimiza el espacio desaprovechado seleccionando el mejor hueco. **De peor ajuste:** Localiza el hueco maximiza el desperdicio. **De primer ajuste:** Localiza el primer hueco donde cabe el segmento empezando por la dirección más baja **Algoritmo Buddy:** Localiza técnicas de compactación de memoria, fusionando espacios útiles.

4.5.3- MEMORIA CON SEGMENTOS PAGINADOS

Este tipo de memoria implica que primero la memoria virtual se divide en segmentos que a su vez luego se paginan. Y quiere decir que para acceder a nuestro objeto primero accederemos a una TS y cada segmento de la TS tendrá su propia TP que será en ella donde encontraremos la página donde estará ubicado el objeto.

Por ello la dirección virtual se divide en tres como indica la siguiente figura:

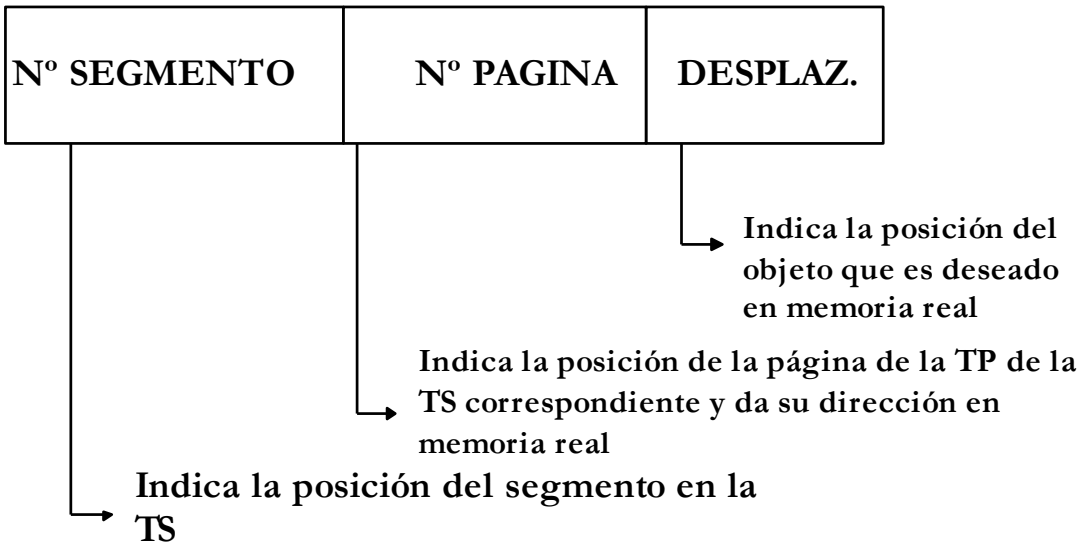

Figura 4.17. Representación de los campos de una dirección virtual

Ahora sólo se mueve páginas por lo que los algoritmos de transferencia son sencillos y además se aprovecha de la buena estructura de los segmentos.

Lo que en definitiva aprovecha las ventajas tanto de la segmentación y de la paginación en un intento de mejorar el rendimiento.

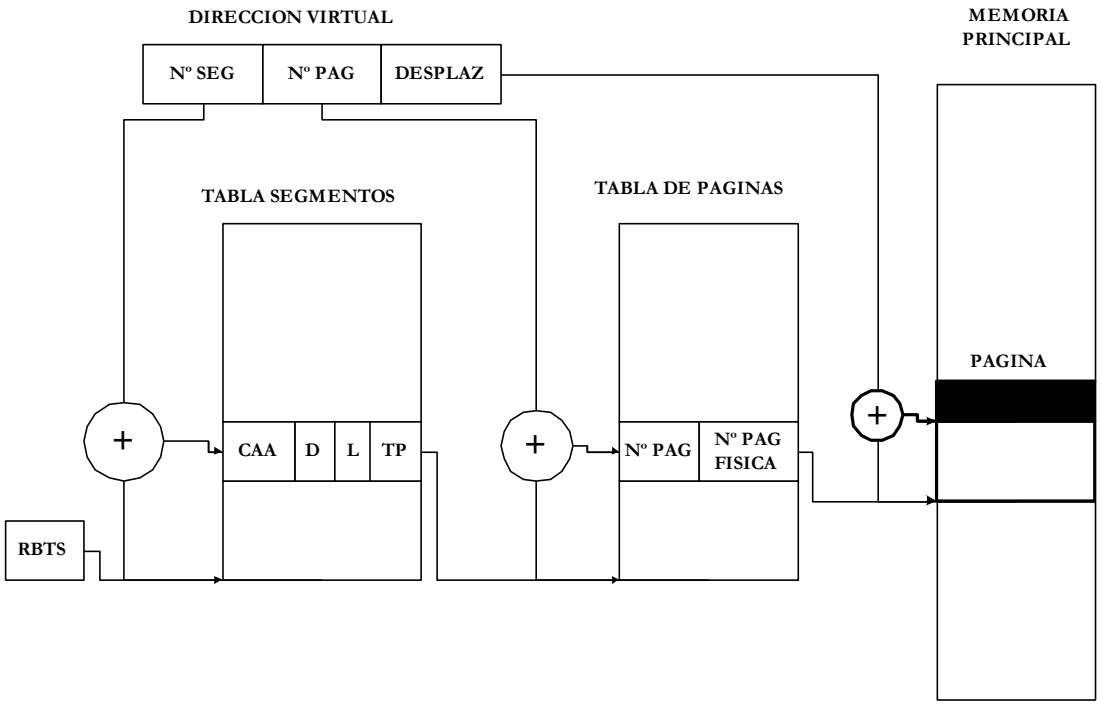

Figura 4.18. Representación del funcionamiento de búsqueda de una página con una Tabla de Segmentos que contiene una Tabla de Páginas.

4.6- MULTITAREA

La multitarea es una característica que poseen los sistemas operativos avnzados. Permite que varios procesos sean ejecutados al mismo tiempo compartiendo uno o más recursos y/o procesadores.

4.6.1- TIPOS DE MULTITAREA

- **Nula**: El sistema operativo carece de multitarea. Aún así puede lograrse a veces algo parecido a una multitarea implementándola en espacio de usuario, ó usando trucos como los TSR de MS-DOS. Un ejemplo sería Windows, hasta la versión 3.11

- **Cooperativa**: Los procesos de usuario son quienes ceden la CPU al sistema operativo a intervalos regulares. Resulta muy problemática, puesto que si el proceso se cuelga y no cede la CPU al sistema operativo, todo el sistema estará colgado. Da lugar también a latencias muy irregulares, y la imposibilidad de tener en cuenta este esquema en sistemas operativos de tiempo real. Un ejemplo sería Windows hasta la versión 3.11.

- **Peemtiva**: El sistema operativo es el encargado de administrar el procesador(es). Repartiendo el tiempo de uso de éste entre los procesos que estén esperando para utilizarlo. Cada proceso utiliza el procesador durante cortos periodos de tiempo, pero el resultado final es prácticamente igual que si estuviese ejecutándose al mismo tiempo. Ejemplo de sistemas de este tipo serían Unix y clones(FreeBSD, Linux,..), VMS y derivados, AmigaOS, etc,....

- **Real**: Sólo se da en sistemas multiprocesador. Es aquella en la que varios procesos se ejecutan realmente al mismo tiempo en distintos microprocesadores. Suele ser también preemptiva. Ejemplo de sistemas operativos con esa capacidad: Linux.

4.7-MECANISMOS DE PROTECCIÓN

En un sistema multitarea el sistema operativo divide a la memoria en "trozos"; tantos como tareas haya en el sistema más una: la global. Esto implicará que la memoria se divide en dos grupos de áreas:

1.Global: Área compartida por todas las tareas.
2.Local: Área propia y sólo propia de cada tarea.(indica **privacidad**)

Para evitar accesos indeseados en un espacio de memoria protegida deben estar especificados las áreas locales de cada tarea como el área global, con sus propios objetos.

Dentro de la zona de memoria existen distintos niveles de privilegio.(Indicando si el objeto debe estar muy **protegido** o no)

El esquema de protección de Intel dispone de cuatro niveles de privilegio, lo que implica una división de la memoria en cuatro porciones diferenciadas entre sí por. Siendo el valor de prioridad de cada nivel:

	Grado prioridad/seguridad	Programas de cada nivel
Nivel 0	El mayor/máximo	Funciones del núcleo
Nivel 1	Buena	Extensión del núcleo o E/S
Nivel 2	Aceptable	Programas comerciales
Nivel 3	Mínima	Programas del usuario

4.8- REGLAS DE ACCESO

Los microprocesadores del Pentium por hardware controlan un conjunto de reglas de acceso a objetos situados en diversas zonas de la memoria, y dentro de ellas, el acceso según los niveles de privilegio, que haya asignado el programador de sistemas.

Esto implica que Intel tiene mayor control de seguridad y mayor precisión y a su vez evita interferencias en el sistema multitarea, por lo que los accesos de un objeto a otro están más controlados.

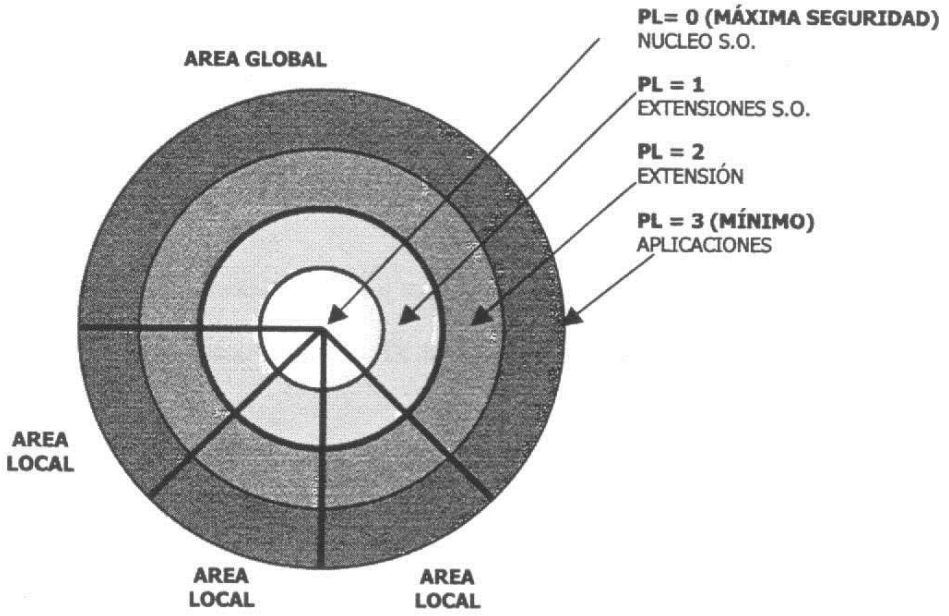

Figura 4.19. Espacio de memoria dividido en 4 niveles de privilegio. Utilizado por INTEL en los Pentium

1. PROTECCIÓN ENTRE TAREAS

La posibilidad que desde una Tarea se acceda a otra Tarea es NULA. Por el contrario está permitido el acceso de una Tarea al espacio global.

2. ACCESO DESDE OBJETOS DE CÓDIGO A OTROS OBJETOS DE LA MEMORIA

Desde un **objeto de código** se puede acceder de forma **JERÁRQUICA**:

Tipo de objeto	**Instrucción(es) utilizada(s)**	**Niveles de acceso permitidos**	**Permiso de acceso entre Tareas**
De código-código	JMP/CALL	Sólo igual	Sólo la misma tarea en curso y Área Global
De código-datos	MOV	Igual ó inferior	Sólo la misma tarea en curso y Área Global
De código-pila (**asociado al objeto de código**)	PUSH/POP	Sólo igual	Sólo la misma tarea en curso

3. EJECUCIÓN DE INSTRUCCIONES ESPECIALES

Dentro del repertorio de instrucciones de los procesadores con esquema de protección. Existen instrucciones **privilegiadas** que sólo pueden ser ejecutadas por el nivel máximo de privilegio (**jerárquicamente hablando***)*

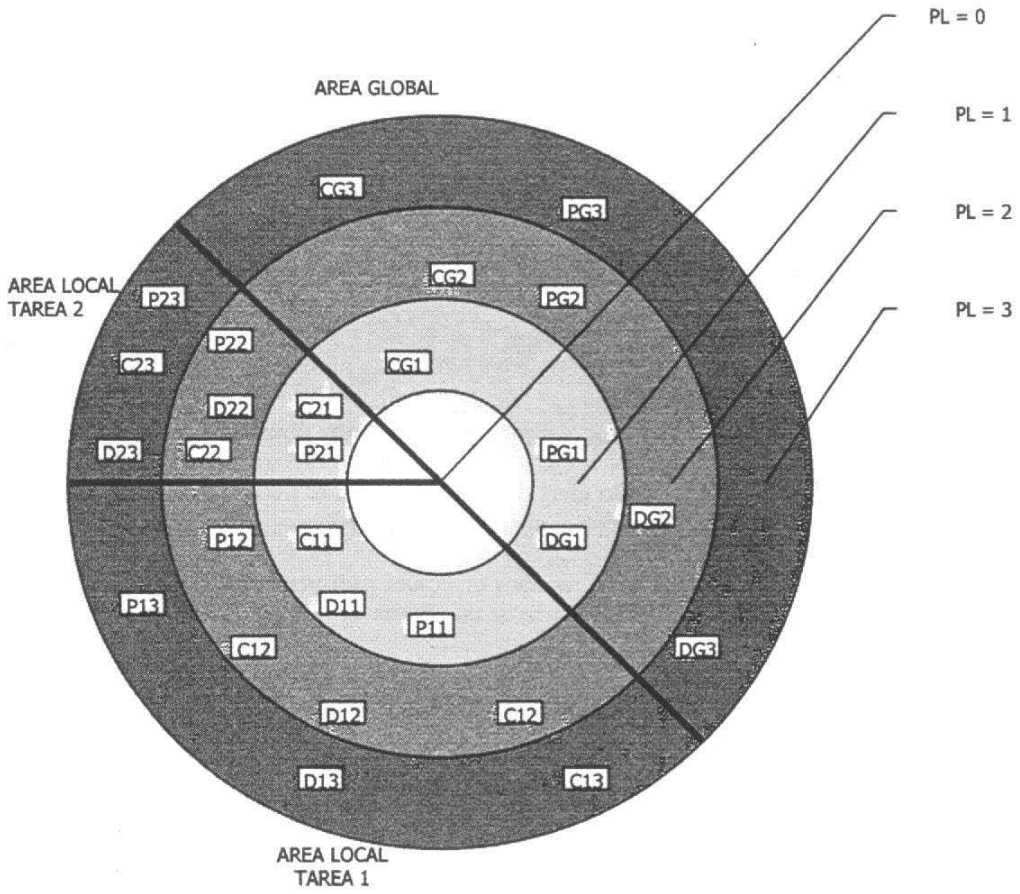

Figura 4.20. Reglas de acceso entre tareas.

Viendo la división de la memoria según la figura 4.12 aquí tenemos algunos ejemplos:

- **C11** podrá acceder a los objetos de datos D11, D12, D13 de su propia tarea y también podría acceder a cualquier objeto de datos del área global con PL =< al suyo; pero en este caso no hay objetos de datos en el área global.

- **C22** podrá acceder a los objetos de código C23 en su propia tarea, y además en el área global podrá acceder a los objetos de código CG2 y CG3.

C23 podrá acceder al objeto de pila P23, que es el segmento de pila asociado intrínsicamente al objeto de código siendo único y exclusivo en cada momento para cada objeto de código.

4.9- SISTEMAS OPERATIVOS ACTUALES

4.9.1- WINDOWS XP

Descripción:

- Sistema **multitarea**.
- Sistema **monousuario**.
- Sistema **multiprocesador**.
- *Windows XP Home Edition* : actualización de Windows 2000 Professional/Me.
- *Windows XP Professional* : actualización de Windows 2000 Server/Me.
- Basado en tecnología NT.
- Presenta notables mejoras: interfaz de usuario más sencilla y agradable, saca mayor partido al hardware de nuestro equipo, etc.

Requisitos mínimos y recomendados:

- *Windows XP Home Edition*

  Es la más compatible y poderosa que ninguna otra estación de trabajo anterior ya que consigue que el ordenador sea más fácil de manejar.

  - **Pentium II a 233 MHz mínimo / 300 MHz ó más recomendado**.
  - **64 Mb de RAM mínimo / 128 Mb recomendado** (4 Gb como máximo).
  - Disco duro de 3 Gb con un mínimo de **1'5 Gb de espacio libre**.
  - Unidad de **CD-ROM** o **DVD-ROM** (32x-12x o más rápido recomendado)
  - **Teclado** y **Ratón** Microsoft o compatible.

- *Windows XP Professional*

  Es un sistema operativo de red para negocios de todos los tamaños.

  - **Pentium II a 350 MHz mínimo / 500 MHz o más recomendado**.
  - **64 Mb de RAM mínimo / 256 Mb recomendado** (4 Gb como máximo).
  - Disco duro de 3 Gb con un mínimo de **1'5 Gb de espacio libre**.
  - Unidad de **CD-ROM** o **DVD-ROM** (32x-12x o más rápido recomendado)
  - **Teclado** y **Ratón** Microsoft o compatible.

Novedades:

- Se reducen significativamente el número de reinicializaciones, con lo cual, las instalaciones pueden realizarse en menor tiempo.

- Incluye también el llamado **Instalador de Microsoft** (MSI); Es la tecnología de Windows 2000 que permite la auto reparación de las aplicaciones (problemas causados por instalaciones / desinstalaciones mal realizadas, etc).

Para que Windows XP funcione debe de ser activado. Durante los 30 primeros días de uso tendremos la opción de hacerlo, y tras finalizar el periodo será obligatorio, ya que de lo contrario el sistema no permitirá iniciar sesión. **Windows Product Activation** se encarga de asociar la clave de instalación del sistema con el hardware el que se ha instalado, de forma que si realizamos más de 4 cambios en el hardware del equipo, tendremos que reactivar el producto para que éste siga funcionando.

4.9.2- WINDOWS 2000

Descripción:

- Sistema **multitarea**.
- Sistema **monousuario**.
- Sistema **multiprocesador**.

- *Windows 2000 Professional* : actualización de Windows NT WorkStation.
- *Windows 2000 Server* : actualización de Windows NT Server.
- Basado en tecnología NT.
- Presenta notables mejoras: al contrario que NT es **Plug & Play**, incorpora directorio activo que centraliza la información sobre recursos y usuarios, incluye mejoras en los servicios de impresión, etc.

Requisitos mínimos y recomendados:

- *Windows 2000 Professional*

Es la más compatible y poderosa que ninguna otra estación de trabajo anterior ya que consigue que el ordenador sea más fácil de manejar.

- **Pentium II**.
- **32 Mb de RAM mínimo / 64 Mb recomendado** (4 Gb como máximo).
- Disco duro de 2 Gb con un mínimo de **1 Gb de espacio libre**.
- Unidad de **CD-ROM**.
- **Ratón** Microsoft o compatible.

- *Windows 2000 Server*

Es un sistema operativo de red para negocios de todos los tamaños. La versión más actual de este sistema operativo permite:

1. Compartir ficheros e impresiones con seguridad.
2. Elegir entre cientos de aplicaciones de negocios compatibles ejecutables.
3. Construir aplicaciones de red y conexiones a Internet.

- **Pentium II**.
- **128 Mb de RAM mínimo / 256 Mb recomendado** (4 Gb como máximo).
- Disco duro de 2 Gb con un mínimo de **1 Gb de espacio libre**.
- Unidad de **CD-ROM**.
- **Ratón** Microsoft o compatible.

Novedades:

- Incluye una protección llamada Protección Windows de Archivos, que no permite que se borren ficheros del sistema.
- Incluye también el llamado Instalador de Microsoft (MSI); Es la tecnología de Windows 2000 que permite la auto reparación de las aplicaciones (problemas causados por instalaciones / desinstalaciones mal realizadas, etc).
- Se reducen significativamente el número de reinicializaciones, con lo cual, las instalaciones pueden realizarse en menor tiempo**.**

4.9.3- LINUX

Descripción:

- Sistema operativo libre (desarrollado por voluntarios). "*FREE OPEN CODE*"
- Sistema **multitarea real**.
- Sistema **multiusuario**.
- Sistema **multiprocesador**.
- Basado en **UNIX**.
- Puede coexistir con otros sistemas operativos (en un mismo procesador).

Requisitos mínimos y recomendados:

- **Pentium**.
- **32 Mb de RAM mínimo / 64 Mb recomendado** (4 Gb como máximo).
- Disco duro con un mínimo de **500 Mb de espacio libre**.
- Unidad de **CD-ROM** o **adaptador de red**.
- **Teclado** y **Ratón**.

Características principales:

- Sistema operativo compatible con Unix.
- El sistema lo forman el núcleo del sistema (*kernel*) y un gran número de librerías
- Se distribuye bajo la GNU Public License, por lo que el código fuente es accesible libremente.
- Carga de ejecutables por demanda: Linux sólo lee del disco aquellas partes de un programa que están siendo usadas actualmente.
- Protección de la memoria entre procesos, de manera que uno de ellos no pueda colgar el sistema.

4.9.4- LINDOWS

Descripción:

- Sistema **multitarea real**.
- Sistema **multiusuario**.
- Sistema **multiprocesador**.
- Basado en **Xandros OS**.
- Puede coexistir con otros sistemas operativos (en un mismo procesador).

Requisitos mínimos:

- **Pentium/AMD** a 90MHz.
- **64 Mb de RAM** (4 Gb como máximo).
- Disco duro con un mínimo de **1 GB de espacio libre** o superior.
- Unidad de **CD-ROM** o **adaptador de red**.
- **Teclado** y **Ratón**.

Características:

Lindows es un sistema operativo construido para aprovechar al máximo, en un futuro próximo, el ancho de banda donde la conectitividad con Internet será más frecuente y barata. Precisamente cada día los equipos requieren soluciones más beneficiosas, fáciles de utilizar y altamente personalizables, y en esos aspectos es donde Lindows destaca, pudiendo personalizar los puestos individualmente. Lindows se aprovecha de un interface similar al de Windows, de apariencia más gráfica e intuitiva y conocido por un 80% de los usuarios por la seguridad del sistema Linux. Con esto pretende extenderse salvando los obstáculos causados por la escasez de aplicaciones creadas para Linux. Precisamente es en este aspecto en donde Lindows quiere mejorar respecto a Linux para poder extenderse. **Basado en Xandros OS** y con **Wine,** un emulador de Windows para Linux, Lindows hace posible ejecutar muchas de las aplicaciones diseñadas para Windows.

El nuevo Lindows se puede instalar sobre **Windows 98/NT/XP**, siguiendo el proceso de actualización tradicional de sistema operativo y será compatible con las aplicaciones Windows, principal problema de los usuarios a la hora de funcionar con sistemas puros Linux. Windows está tan extendido a pesar de sus fallos, que las aplicaciones que todos usamos a diario están diseñadas para funcionar con este sistema, por lo que pasarnos a Linux resulta un tanto complicado. Sin embargo, este traumático cambio se ve completamente erradicado gracias al Lindows.