

## **UNIDAD 1 - Introducción, Definición y Evolución de los SO**

### **¿Qué es un Sistema Operativo?**

Es un programa que actúa de intermediario entre el usuario de la computadora y el hardware.

Objetivos:

1. Ejecutar programas.
2. Hacer un uso eficiente de los recursos.
3. Proporcionar visión de máquina virtual extendida.

### Ejecución del SO

Una vez finalizado el arranque, el SO solamente se ejecuta en respuesta a **interrupciones**.

Se activa cuando debe responder a:

- Una petición de servicio de un proceso.
- Una interrupción.
- Excepción hardware.

### Servicios

Son programas cargados por el propio SO (corren en segundo plano).

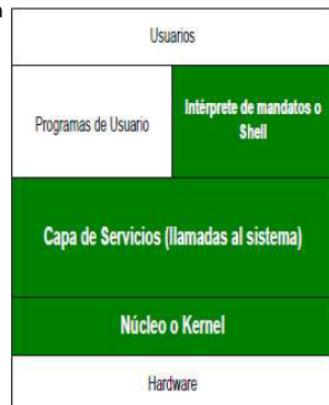
### Servicios del SO

- 1) **Desarrollo de Programas.** El SO proporciona una variedad de utilidades y servicios, tales como editores y depuradores, para asistir al programador en la creación de los programas.
- 2) **Acceso a dispositivos de E/S.** Cada dispositivo de E/S requiere su propio conjunto de instrucciones para cada operación.
- 3) **Acceso controlado a los ficheros.** El SO debe reflejar una comprensión detallada de la estructura de los datos contenidos en los ficheros del sistema de almacenamiento.
- 4) **Acceso al sistema.** El SO controla el acceso al sistema completo y a recursos del sistema específicos. La función de acceso debe proporcionar protección a los recursos y a los datos, evitando el uso no autorizado de los usuarios y resolviendo conflictos en el caso de conflicto de recursos.
- 5) **Detección y respuesta a errores.** Se pueden dar gran variedad de errores durante la ejecución de un sistema de computación. El SO debe proporcionar una respuesta que elimine la condición de error, suponiendo el menor impacto en las aplicaciones que están en ejecución.
- 6) **Contabilidad.** Un buen SO recogerá estadísticas de uso de los diferentes recursos y monitorizará parámetros de rendimiento tales como el tiempo de respuesta.

## Niveles del sistema operativo/Usuarios del Sistema operativo

ASO - UAI

- El SO está formado conceptualmente por 3 capas principales:
  - Núcleo o *Kernel*
  - Servicios o llamadas al sistema
  - Intérprete de mandatos



Estructura conceptual del SO

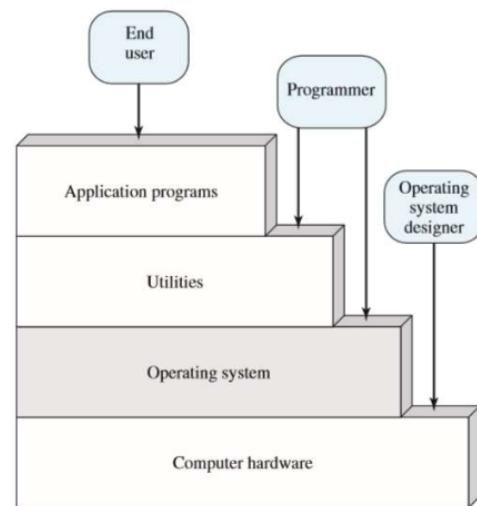
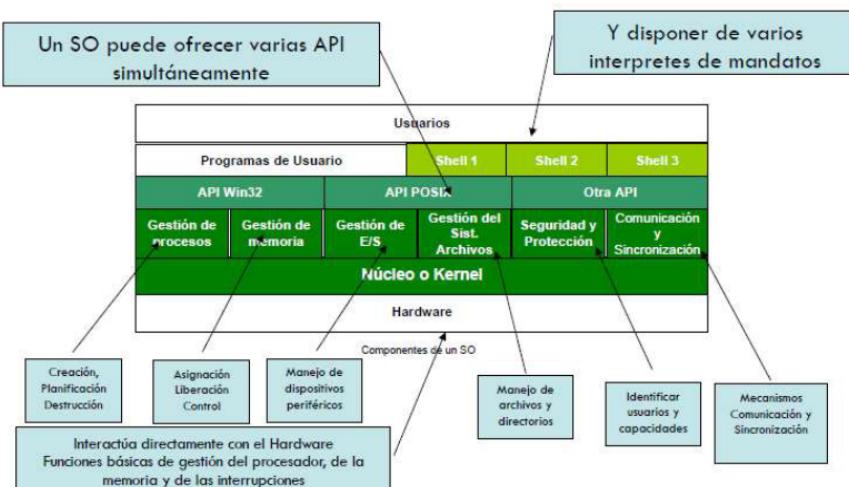


Figure 2.1 Layers and Views of a Computer System

## Componentes del Sistema Operativo

ASO - UAI



## GESTOR DE PROCESOS

Un **PROCESO** es una instancia de un programa en ejecución.

- Necesita recursos tales como CPU, memoria, ficheros, etc. para llevar a cabo su tarea.

Está formado por tres componentes:

- 1) Un programa ejecutable
- 2) Los datos asociados que necesita el programa.
- 3) El contexto de ejecución del programa.

El SO tiene la responsabilidad de gestionar los siguientes aspectos de los procesos:

- 1) Creación y destrucción.
- 2) Suspensión y reanudación.
- 3) Proporcionar mecanismos de sincronización y comunicación.
- 4) Asignación y mantenimiento de los recursos del proceso.

## GESTOR DE MEMORIA

**MEMORIA**: Vector enorme de bytes, cada uno con su propia dirección.

El SO tiene la responsabilidad de gestionar los siguientes aspectos de la memoria:

- 1) Mantener un mapa de las partes de memoria en uso y saber quién las está usando.
- 2) Decidir qué procesos se deben cargar, y dónde, cuándo hay memoria disponible.
- 3) Asignar y liberar espacio de memoria.

## SERVIDOR FICHEROS Y DIRECTORIOS

- **FICHERO**. Conjunto de información lógicamente relacionada y definida por su creador.
- **DIRECTORIO**. Conjunto de identificadores lógicos de ficheros.
- **FICHEROS FRECUENTES**. Programas y datos.

El SO tiene las siguientes responsabilidades:

- 1) Creación y borrado de ficheros y directorios.
- 2) Primitivas para manipular ficheros y directorios.
- 3) Proyectar los ficheros sobre almacenamiento secundario.

- **crear:** Crea un fichero con un nombre y unos atributos.
- **borrar:** Borra un fichero a partir de su nombre.
- **abrir:** Abre un fichero a partir de su nombre para permitir operaciones de acceso.
- **cerrar:** Cierra un fichero abierto.
- **leer:** Lee datos de un fichero abierto a un almacén en memoria.
- **escribir:** Escribe datos a un fichero abierto desde un almacén en memoria.
- **posicionar:** Mueve el apuntador usado para acceder al fichero, afectando a operaciones posteriores.
- **control:** Permite manipular los atributos de un fichero.

### Estándar POSIX

Es una interfaz estándar de SOs de IEEE.

Objetivo: Portabilidad de las aplicaciones entre diferentes plataformas y sistemas operativos.

NO es una implementación. Sólo define una interfaz.

- Nombres de funciones cortos y en letras minúsculas:
  - fork
  - read
  - close
- Las funciones normalmente devuelven 0 en caso de éxito o -1 en caso de error.
  - Variable errno.
- Recursos gestionados por el sistema operativo se referencian mediante descriptores (números enteros)

## UNIDAD 2 - Algoritmos de Planificación y Sincronización de Procesos

### Itinerario 3: Introducción a la Gestión de Procesos

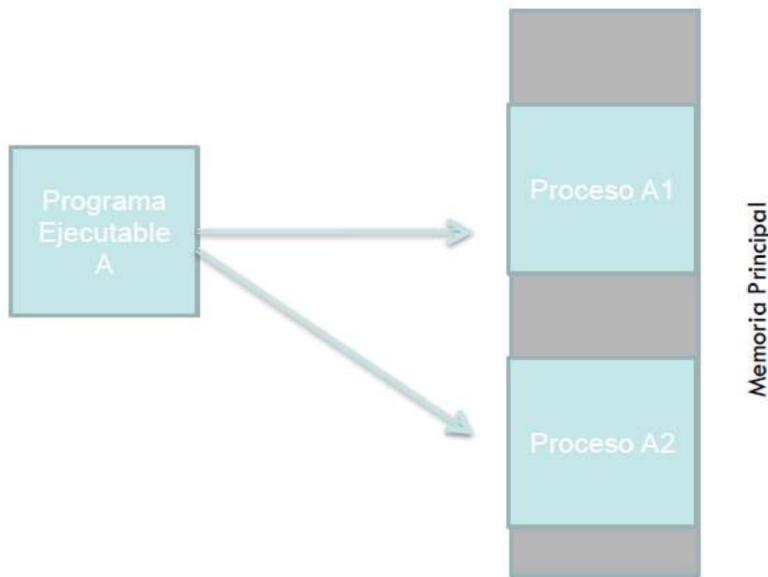
#### PROCESO

Instancia de un programa en ejecución.

- Cada ejecución de un programa da lugar a un proceso.

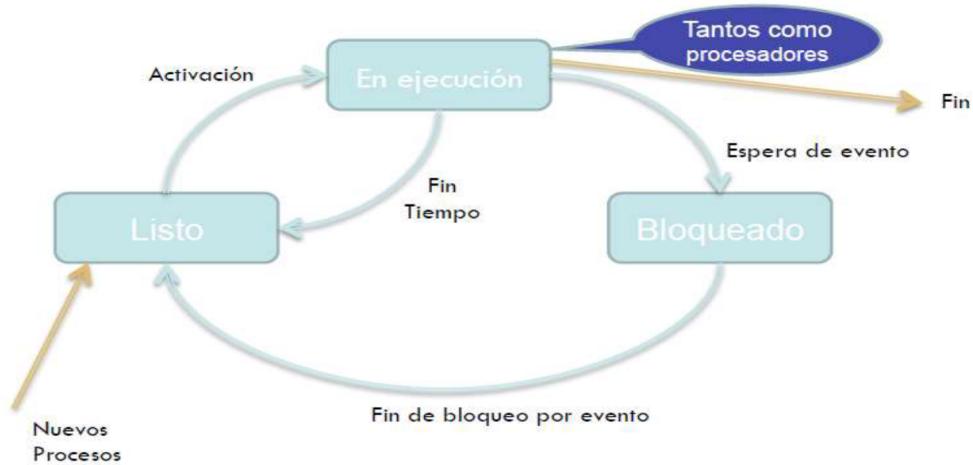
#### Concepto de Proceso

ASO - UAI



## Ciclo de vida básico de un proceso

ASO - UAI



## INFORMACIÓN DEL SO

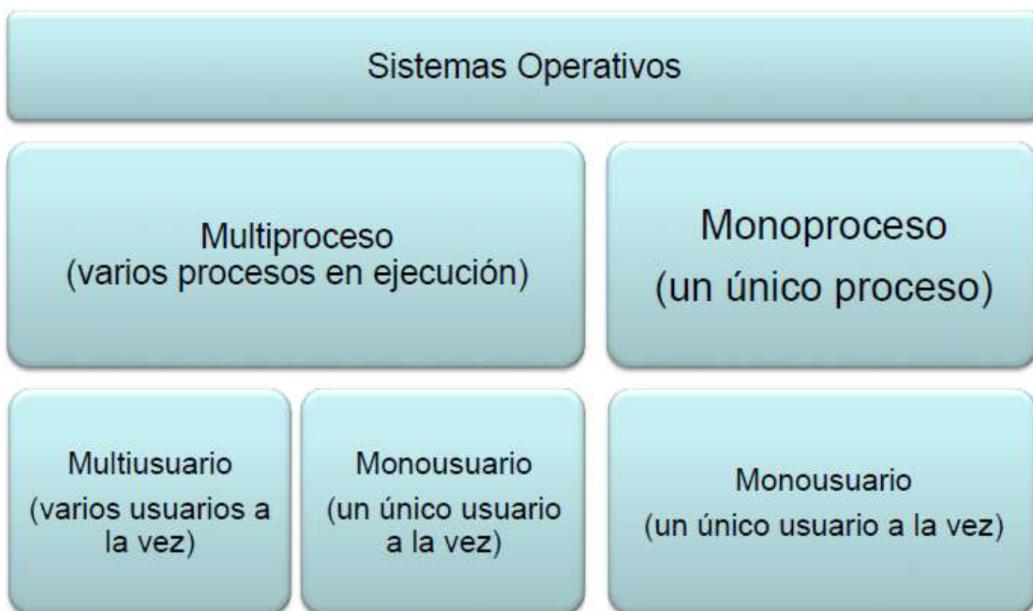
El SO mantiene información adicional sobre los procesos en la **Tabla de Procesos**.

- **Bloque de Control de Procesos (BCP)**. Cada entrada de la tabla que mantiene la información sobre un proceso.

Contiene:

- 1) Información de Identificación.
- 2) Estado del procesador.
- 3) Información de control de proceso.

## TIPOS DE SISTEMAS OPERATIVOS



### Ventajas de la MULTITAREA

- Facilita la programación, dividiendo los programas en procesos.
- Permite el servicio interactivo simultáneo de varios usuarios.
- Aumenta el uso de la CPU.
- Aprovecha los tiempos que los procesos pasan esperando a que se completen sus operaciones de E/S.

### Itinerario 4: Planificación de Procesos

#### CREACIÓN DE PROCESOS

- Los SOs proveen mecanismos para que los procesos puedan crear otros procesos  
-> Llamada al sistema.
- El proceso de creación se puede repetir recursivamente creándose una “estructura familiar”
- Asignación de recursos al nuevo proceso:
  - Los obtiene directamente el SO.

#### TERMINACIÓN DE PROCESOS

- Cuando un proceso termina todos los recursos asignados son liberados:

- 1) Memoria.
  - 2) Ficheros abiertos.
  - 3) Entradas en tablas.
- Un proceso puede terminar de 2 formas:
- 1) Voluntariamente: Llamada a Exit()
  - 2) Involuntariamente: Excepciones o abortados por el usuario.

## NIVELES DE PLANIFICACIÓN

- 1) Planificación a Corto Plazo. Selecciona el siguiente proceso a ejecutar.
- 2) Planificación a Medio Plazo. Selecciona qué procesos se añaden o se retiran de memoria principal.
- 3) Planificación a Largo Plazo. Realiza el control de admisión de procesos a ejecutar.

## TIPOS DE PLANIFICACIÓN

- 1) No Apropiativa. El proceso en ejecución conserva el uso de la CPU mientras lo desee.
- 2) Apropiativa. El SO puede expulsar a un proceso de la CPU.

## MEDIDAS

- Utilización de CPU
  - Porcentaje de tiempo que se usa la CPU.
- Productividad
  - Número de trabajos terminados por unidad de tiempo.
- Tiempo de Retorno (Tq)
  - Tiempo que está un proceso en el sistema. Instante final (Tf) - Instante Inicial (Ti)
- Tiempo de Servicio (Ts)
  - Tiempo dedicado a tareas productivas (cpu, entrada/salida).  $Ts = T_{CPU} + TE/S$
- Tiempo de Espera (Te)
  - Tiempo que un proceso pasa en colas de espera.  $Te = Tq - Ts$
- Tiempo de Retorno Normalizado (Tn)
  - Tiempo entre tiempo de Retorno y tiempo de Servicio.  $Tn = Tq - Ts$

## **LINK EXPLICATIVO PARA LOS ALGORITMOS DE PLANIFICACIÓN**

<https://www.youtube.com/watch?v=xQDi62YZuuw>

### **PLANIFICACIONES**

#### **1) FIRST COME FIRST SERVE.**

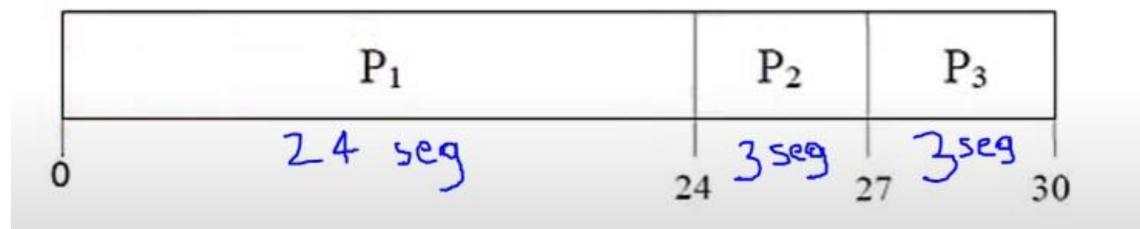
- Primero en llegar, primero en servirse.
- Algoritmo no apropiativo.
- Penaliza a los procesos más cortos.

### **Planificación First-Come, First-Served (FCFS)**

Proceso	Burst time
P <sub>1</sub>	24
P <sub>2</sub>	3
P <sub>3</sub>	3

Lo que está en rojo muestra la orden de llegada de los procesos, 1 siendo primero y 3 tercero

Lo que está en azul es el tiempo que ese proceso va a usar la CPU



- Tiempo de espera para P<sub>1</sub> = 0; P<sub>2</sub> = 24; P<sub>3</sub> = 27
- Tiempo promedio de espera: (0 + 24 + 27)/3 = 17

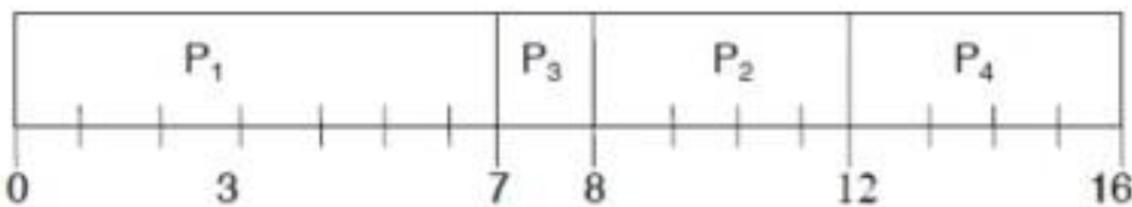
#### **2) SJB (SHORTEST JOB FIRST)**

- Primero el trabajo más corto.
- Algoritmo no apropiativo.
- Selecciona el trabajo más corto.
- Solamente se puede aplicar si se conoce de antemano la duración de cada trabajo.

Proceso Tiempo de llegada Tiempo de exp

$P_1$	0.0	7
$P_2$	2.0	4
$P_3$	4.0	1
$P_4$	5.0	4

- SJF (non-preemptive)



- Tiempo de espera promedio =  $(0 + 6 + 3 + 7)/4 = 4$

### 3) ROUND-ROBIN

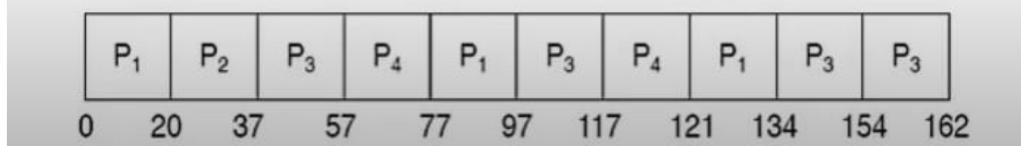
- Mantiene una cola FIFO con los procesos listos para ser ejecutados.
- Un proceso recibe el procesador durante un *quantum de tiempo*.
- Algoritmo Apropiativo.

### Ejemplo de RR con Time Quantum = 20

Proceso    Burst Time

$P_1$	$53 - 20 = 33$	$33 - 20 = 13$	$13 - 13 = \text{Exit}$
$P_2$	$17 - 17 = \text{Exit}$		
$P_3$	$68 - 20 = 48$	$48 - 20 = 28$	$28 - 20 = 8$
$P_4$	$24 - 20 = 4$	$4 - 20 = \text{Exit}$	

- La gráfica de Gantt es:



#### 4) Asignación por Prioridades

- Cada proceso tiene una prioridad asignada.
- Se selecciona primero los procesos más prioritarios. (**ENTERO MÁS PEQUEÑO = PRIORIDAD MÁS ALTA**)

### Itinerario 5: Hilos Y Procesos

#### HILOS

La mayoría de los SO modernos proporcionan procesos con múltiples hilos de control en su interior.

Se considera una **unidad básica de utilización de CPU** y cada uno comprende:

- Identificador de Thread
- Contador de programa
- Conjunto de registros
- Pila

**ES UNA LÍNEA DE EJECUCIÓN EN PROCESO.**

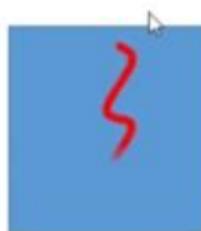
## Procesos e hilos



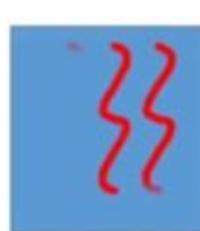
un proceso  
un hilo



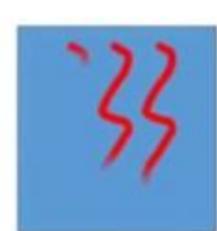
un proceso  
varios hilos



varios procesos  
un hilo por proceso



varios procesos  
varios hilos por proceso



## BENEFICIOS

- Capacidad de respuesta
- Compartición de recursos
- Economía de recursos
- Utilización sobre arquitecturas multiprocesador

## SOPORTE DE HILOS

### 1) **USER LEVEL THREAD (ULT)**

- Implementados en forma de biblioteca de funciones en espacio de usuario.

### 2) **KERNEL LEVEL THREAD (KLT)**

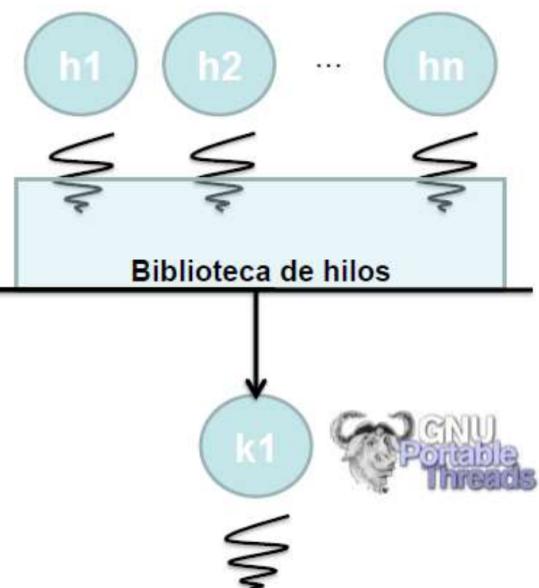
- El kernel se ocupa de crearlos, planificarlos y destruirlos.
- Es un poco más lento ya que involucra al kernel y supone un cambio en el modo de ejecución.

## MODELOS DE MÚLTIPLES HILOS

### Modelos de múltiples hilos: Muchos a uno

ASO - UAI

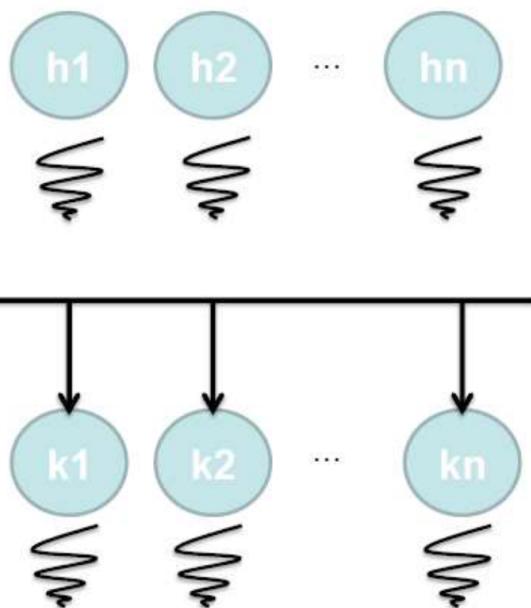
- Hace corresponder múltiples hilos de usuario a un único hilo del núcleo.
- Biblioteca de hilos en espacio de usuario.
- Llamada bloqueante:
  - Se bloquean todos los hilos.
- En multiprocesadores no se pueden ejecutar varios hilos a la vez.



## Modelos de múltiples hilos: uno a uno

ASO - UAI

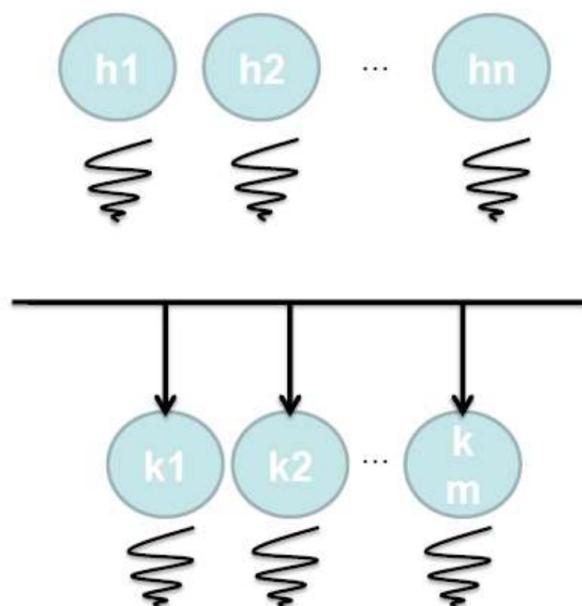
- Hace corresponder un hilo del kernel a cada hilo de usuario.
- La mayoría de las implementaciones restringen el número de hilos que se pueden crear.
- Ejemplos:
  - Linux 2.6.
  - Windows.
  - Solaris 9.

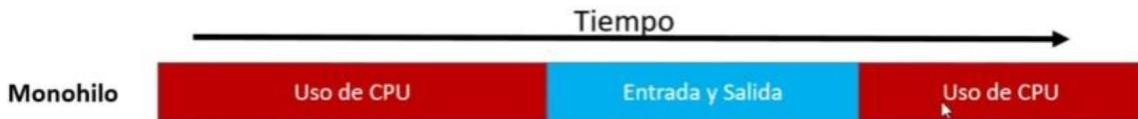


## Modelos de múltiples hilos: Muchos a Muchos

ASO - UAI

- Este modelo multiplexa los threads de usuario en un número determinado de threads en el kernel.
- El núcleo del sistema operativo se complica mucho.
- Ejemplos:
  - Solaris (versiones anteriores a 9).
  - HP-UX.
  - IRIX.





Con un solo hilo, una aplicación se **ejecuta secuencialmente** aunque utilice diferentes recursos sin dependencias, por ej: puede ejecutar una sección de código sin depender que la anterior haya sido terminada.

### **Multihilo**



Con varios hilos un **hilo puede estar haciendo procesamiento** (uso del CPU) mientras otro **hilo efectúa operaciones de E/S**.

## Itinerario 6: Mecanismos de comunicación y sincronización

### **PROCESO CONCURRENTE**

Dos procesos son **concurrentes** cuando se ejecutan de manera que sus intervalos se **solapan**.



### **TIPOS DE CONCURRENCIA**

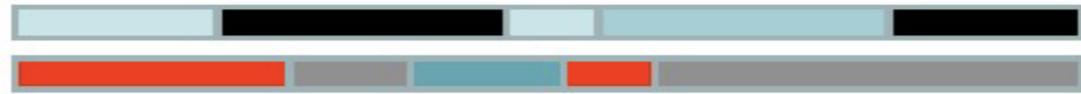
1) **CONCURRENCIA APARENTE**. Hay más procesos que procesadores.

- Los procesos se multiplexan en el tiempo
- Pseudoparalelismo

## 1 CPU



## 2 CPUs



- 2) **CONCURRENCIA REAL.** Cada proceso se ejecuta en un procesador.

- Se produce una ejecución en paralelo.
- Paralelismo real.

## 4 CPUs



## VENTAJAS

- Facilita la programación ya que diversas tareas se pueden estructurar en procesos separados.
- Acelera la ejecución de cálculos.
- Mejora la interactividad de las aplicaciones.
- Mejora el aprovechamiento de la CPU.

## TIPOS DE PROCESOS CONCURRENTES

- 1) **INDEPENDIENTES.** Procesos que se ejecutan concurrentemente pero sin ninguna relación.
- No necesitan comunicarse y sincronizarse.
- 2) **COOPERANTES.** Procesos que se ejecutan concurrentemente pero con alguna interacción entre ellos.
- Pueden comunicarse y sincronizarse entre sí.

## INTERACCIONES ENTRE PROCESOS

## 1) **ACCESO A RECURSOS COMPARTIDOS.**

- Procesos que comparten un recurso.

## 2) **COMUNICACIÓN.**

- Procesos que intercambian información.

## 3) **SINCRONIZACIÓN.**

- Un proceso debe esperar a un evento en otro proceso.

## MECANISMOS DE COMUNICACIÓN

Los mecanismos de comunicación permiten la transferencia de información entre dos procesos.

- Archivos
- Tuberías (Pipes, FIFOs)
- Variables en memoria compartida
- Paso de mensajes

## MECANISMOS DE SINCRONIZACIÓN

Los mecanismos de sincronización permiten forzar a un proceso a detener su ejecución hasta que ocurra un evento en otro proceso.

Servicios del SO:

- Señales (asincronismo)
- Tuberías (Pipes, FIFOs)
- Semáforos
- *Mutex\** y *Variables Condicionales* (asociadas a un Mutex)
- Paso de mensajes

\*Mutex = es un mecanismo de sincronización indicado para procesos ligeros. Dos operaciones: -Lock y -Unlock

## **LINK EXPLICATIVO PARA LOS ALGORITMOS DEL BANQUERO (DIJKSTRA)**

<https://www.youtube.com/watch?v=oIXj9FfSyxY>

## **UNIDAD 5 - Virtualización y Computación en la Nube**

### **Itinerario 7: Virtualización y Computación en la Nube**

#### **VIRTUALIZACIÓN**

Se refiere a la abstracción de los recursos de una computadora, llamada **VMM** (Virtual Machine Monitor) que crea una capa de abstracción entre el hardware de la máquina física y el sistema operativo de la máquina virtual.

#### **TIPOS DE VIRTUALIZACIÓN**

##### **1) VIRTUALIZACIÓN ASISTIDA POR SOFTWARE**

Es necesario un sistema operativo para poder virtualizar (**VirtualBox**, **VMware**).

##### **2) PARAVIRTUALIZACIÓN**

No necesitan un sistema operativo, son un sistema operativo y utilizan acceso a hardware específico

#### **HYPERVERISOR**

Un **HIPERVISOR** es una plataforma que permite aplicar diversas técnicas de control de virtualización para utilizar, al mismo tiempo, diferentes sistemas operativos en una misma computadora.

#### **MÁQUINAS VIRTUALES Y CONTENEDORES**

- 1) **CONTENEDOR**. Es una máquina virtual ligera, que comparte funciones con el kernel del SO, que está realizando las funciones de hipervisor. (Ejemplo: Docker, **Parallels**).
- 2) **MÁQUINA VIRTUAL**. Es una máquina a todos los efectos, y para ello el sistema realiza una reserva de los recursos físicos según las características especificadas para la máquina.

#### **CLOUD COMPUTING**

Es la utilización de recursos (servidores, aplicaciones, etc) que se encuentran en Internet y la gestión de la infraestructura (servidores, almacenamiento, red) se encarga un tercero para obtener una optimización total de la infraestructura mediante la automatización para conseguir flexibilidad y adaptabilidad de los recursos informáticos.

¿Qué se quiere resolver con la utilización de Cloud Computing?

- Costos de la infraestructura
- Escalabilidad
- Flexibilidad

## CONCEPTOS DE NUBE

- 1) **PÚBLICA**. Son recursos informáticos accesibles desde Internet.
- 2) **PRIVADA**. Son recursos informáticos para nosotros. El acceso es exclusivo para nosotros (VPN).
- 3) **HÍBRIDA**. Es una mezcla de las Nubes públicas y privadas.

## **UNIDAD 6 - Compiladores**

### Itinerario 8: Compiladores

#### INTÉRPRETE

Es un programa informático que procesa el código fuente de un programa mientras se está ejecutando, y actúa como una *interfaz* entre ese programa y el procesador.

Un intérprete siempre procesa el código línea por línea, de modo que lee, analiza y prepara cada secuencia de forma consecutiva para el procesador.

## Lenguaje interpretado



#### COMPILADOR

Es un programa informático que traduce el código fuente de un programa a código máquina antes de ejecutarlo.

- En muchos casos, durante el proceso de compilación tiene lugar un paso intermedio: Antes de generar la traducción final en código máquina, suelen convertir el código fuente en un **CÓDIGO INTERMEDIO** (compatible con otras plataformas y puede ser utilizado por un intérprete).
- El compilador determina **qué instrucciones** van a enviarse al procesador y en **qué orden**.

## TIPOS DE COMPILADORES

- Compiladores Cruzados.
- Compiladores Optimizadores.
- Compiladores de una sola pasada.
- Compiladores de varias pasadas.
- Compiladores.

## FASES DE UN COMPILADOR

Un compilador está formado por dos procesos: **análisis** y **síntesis**

- 1) **ANÁLISIS**. Se trata de la escritura correcta del código fuente.
- 2) **SÍNTESIS**. Se procede a generar grupos de los componentes que conforman el programa, para generar una salida.

## RESUMEN

### Compilador e intérprete

ASO - UAI

	Intérprete	Compilador
Momento en que se traduce el código fuente	Durante el tiempo de ejecución del software	Antes de ejecutar el software
Procedimiento de traducción	Línea por línea	Siempre todo el código
Presentación de errores de código	Después de cada línea	En conjunto, después de toda la compilación
Velocidad de traducción	Alta	Baja
Eficiencia de traducción	Baja	Alta
Coste de desarrollo	Bajo	Alto
Lenguajes típicos	PHP, Perl, Python, Ruby, BASIC	C, C++, Pascal

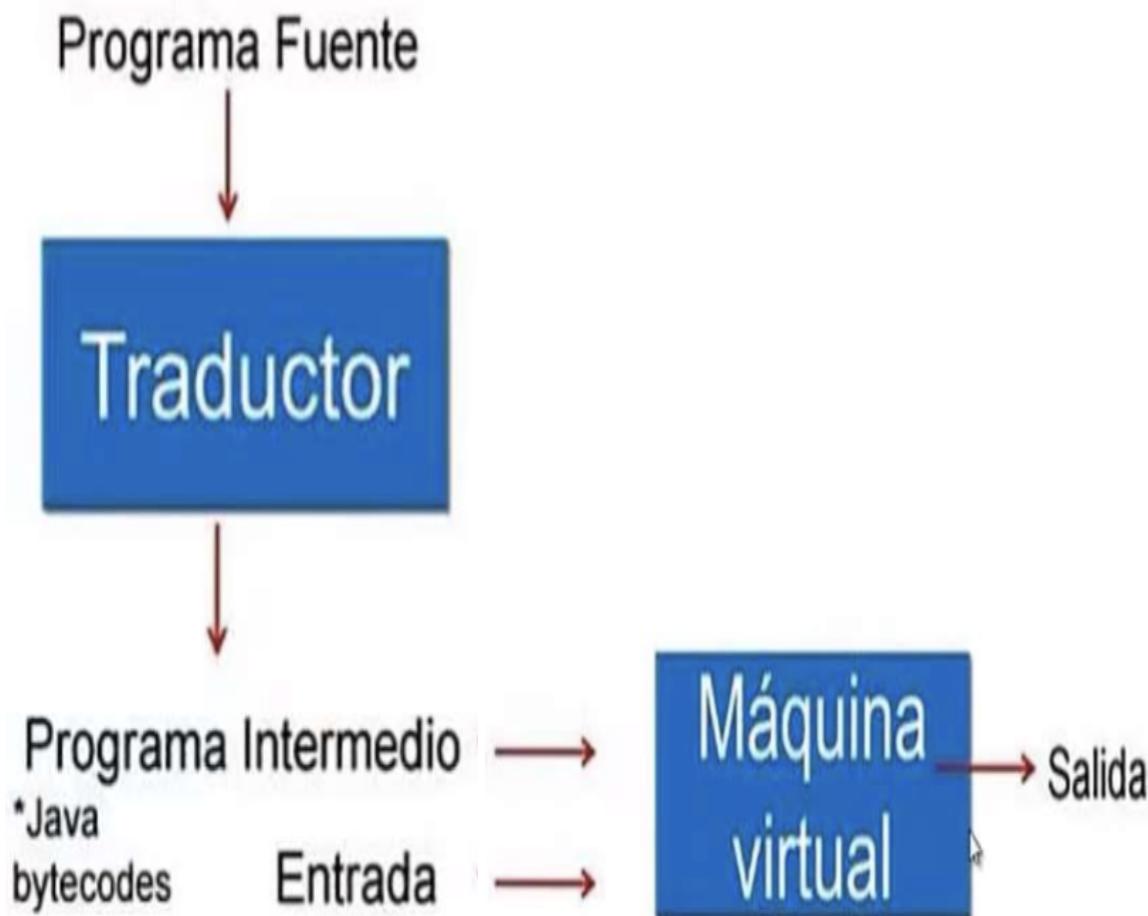
	Ventaja	Inconveniente
Intérprete	Proceso de desarrollo sencillo (sobre todo en términos de depuración)	Proceso de traducción poco eficiente y velocidad de ejecución lenta
Compilador	Proporciona al procesador el código máquina completo y listo para ejecutar	Cualquier modificación del código (resolución de errores, desarrollo del software, etc.) requiere volverlo a traducir

## JUST-IN-TIME COMPILER

Este tipo de compilador traduce el código del programa durante el tiempo de ejecución, al igual que el intérprete.

De esta forma, la **alta velocidad** de ejecución típica de los compiladores **se complementa con la simplificación del proceso de desarrollo**.

**Java** es uno de los ejemplos más conocidos de lenguaje basado en compilación en tiempo de ejecución.



## **UNIDAD 3 - Mecanismos de Gestión de Memoria**

### **Itinerario 10: Manejo de Memoria**

#### **GESTOR DE MEMORIA**

##### **OBJETIVOS**

- Ofrecer a cada proceso un espacio lógico propio.
- Proporcionar protección entre procesos.
- Permitir que los procesos comparten memoria.
- Maximizar el grado de la multiprogramación.

#### **MEMORIA PRINCIPAL**

Memoria volátil (datos no persistentes), los datos son accedidos por el procesador.

#### **MEMORIA SECUNDARIA**

Memoria no volátil (datos persistentes). Está organizada en bloques de datos, pero para simplificar se crean estructuras de archivos y directorios.

#### **REUBICACIÓN**

La **REUBICACIÓN** es la traducción de direcciones lógicas en direcciones físicas.

- **DIRECCIONES LÓGICAS.** Direcciones de memorias generadas por el programa.
- **DIRECCIONES FÍSICAS.** Direcciones de memoria principal asignadas al proceso.

Función de Traducción: Traducción(IdProd, dir\_lógica) -> dir\_física

##### **1) REUBICACIÓN HARDWARE.**

- Hardware (MMU) encargado de traducción.
- SO se encarga de:
  - Almacena por cada proceso su función de traducción.
  - Especifica al hardware qué función aplicar para cada proceso.
- Programa se carga en memoria sin modificar.

##### **2) REUBICACIÓN SOFTWARE.**

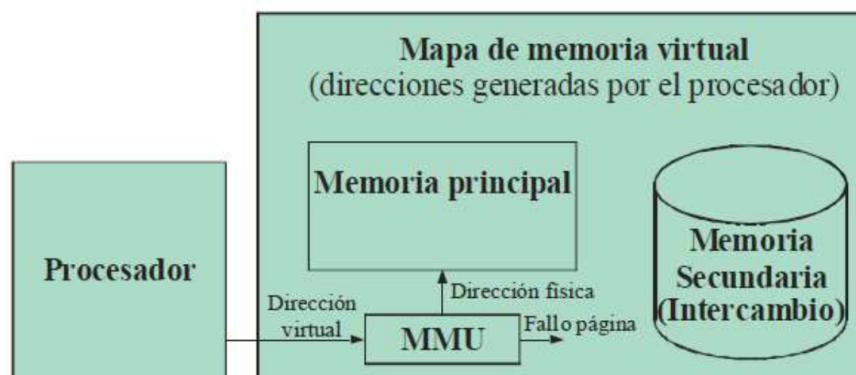
- Traducción de direcciones durante carga del programa.
- Programa en memoria distinto del ejecutable

## Itinerario 11: Gestión de Memoria Virtual

### Gestión de Memoria Virtual

ASO - UAI

- Gestión automática de la parte de la Jerarquía de Memoria formada por los niveles de memoria principal y de disco.



### BENEFICIOS

- Protección y reubicación de procesos.
- Aumento grado de multiprogramación.

### SWAPPING

Es un mecanismo para mover programas entre memoria principal y secundaria, normalmente disco (dispositivo de swap).

Con swapping, los programas pueden salir/entrar de la memoria durante su ejecución.

La función del SO que gestiona el intercambio entre disco y memoria se denomina 'swapper'.

### PAGINACIÓN Y SEGMENTACIÓN

Permiten la ubicación no contigua de programas para combatir la fragmentación y la degradación de la memoria. Al poder ubicarse de forma no contigua, un programa ya no necesita un hueco de su tamaño, sino que la cantidad total de memoria libre sea mayor o igual.

La ubicación no contigua requiere dividir los programas en trozos. En paginación, que es un caso particular del particionado fijo, el programa se divide en páginas del mismo tamaño.

La segmentación, que es un caso particular de particionado variable, divide el programa en sus unidades lógicas (código, pila, datos, ...), denominadas segmentos.

## SISTEMAS COMBINADOS

Es un sistema que combina la paginación y la segmentación. Se denominan sistemas de paginación segmentada ya que los segmentos se dividen en páginas ó que la tabla de páginas se segmenta en varias.

La dirección lógica se divide en tres partes:

- 1) Número de Segmento.
- 2) Número de Página.
- 3) Número de Desplazamiento.

# **UNIDAD 4 - Gestión de Almacenamiento de Información**

## **Itinerario 12: Manejo de Directorios y Ficheros**

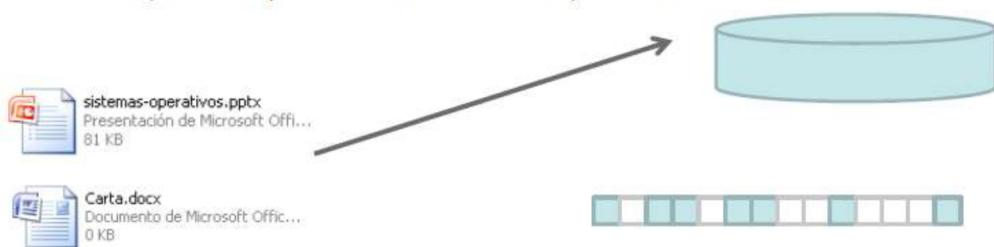
### SISTEMA DE FICHEROS

- Ofrece al usuario una visión lógica simplificada del manejo de los dispositivos periféricos en forma de ficheros.
- Constituye la parte del SO que gestiona los ficheros.
- ES LA CAPA DE SOFTWARE ENTRE DISPOSITIVOS Y USUARIOS

### FUNCIONES

1. Organización
2. Almacenamiento
3. Recuperación
4. Gestión de Nombres
5. Protección

- Visión lógica:
  - Ficheros
  - Directorios
  - Sistemas de Ficheros y particiones
- Visión física:
  - Bloques o bytes ubicados en dispositivos



### ATRIBUTOS DE UN FICHERO/ARCHIVO

- **Nombre:** Identificador en formato legible por una persona.
- **Identificador:** Etiqueta única del archivo – Suele ser numérico.
- **Tipo de fichero:** necesario en sistemas que proporcionan distintos formatos de Ficheros. Como mínimo se suele diferenciar el atributo de ejecutable.
- **Ubicación:** Identificación del dispositivo de almacenamiento y la posición dentro del dispositivo.
- **Tamaño del fichero:** número de bytes en el fichero, máximo tamaño posible, etc.
- **Protección:** control de accesos y de las operaciones sobre el fichero.
- **Información temporal:** de creación, de acceso, de modificación, etc.

### OPERACIONES SOBRE FICHEROS

- **Creación:** Asignación de espacio inicial y metadatos.
- **Borrado:** Liberación de recursos asociados.
- **Escritura:** Almacena información en el archivo.
- **Lectura:** Recupera información del archivo.

## MÉTODOS DE ACCESO

## 1) ACCESO SECUENCIAL

- Basado en el modelo de acceso a datos en una cinta magnética.
  - Utilizable en dispositivos de acceso secuencial.

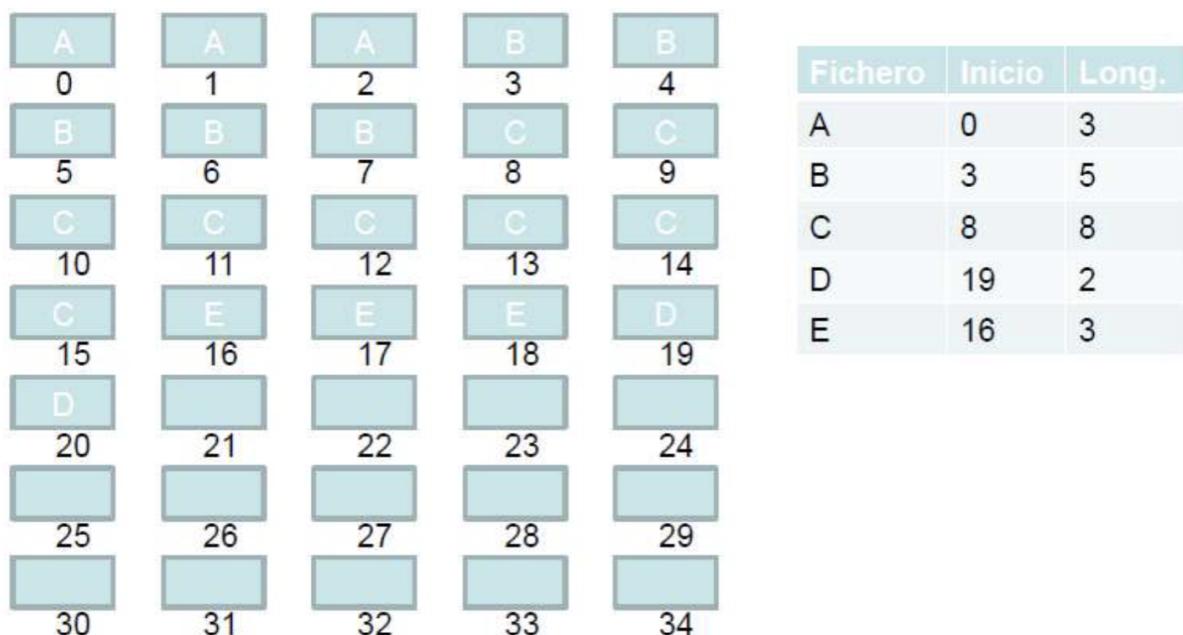
## 2) ACCESO DIRECTO

- Basado en el modelo de acceso a dispositivo de disco.
  - Fichero dividido en registros de longitud fija.

## MÉTODOS DE ASIGNACIÓN DE FICHEROS

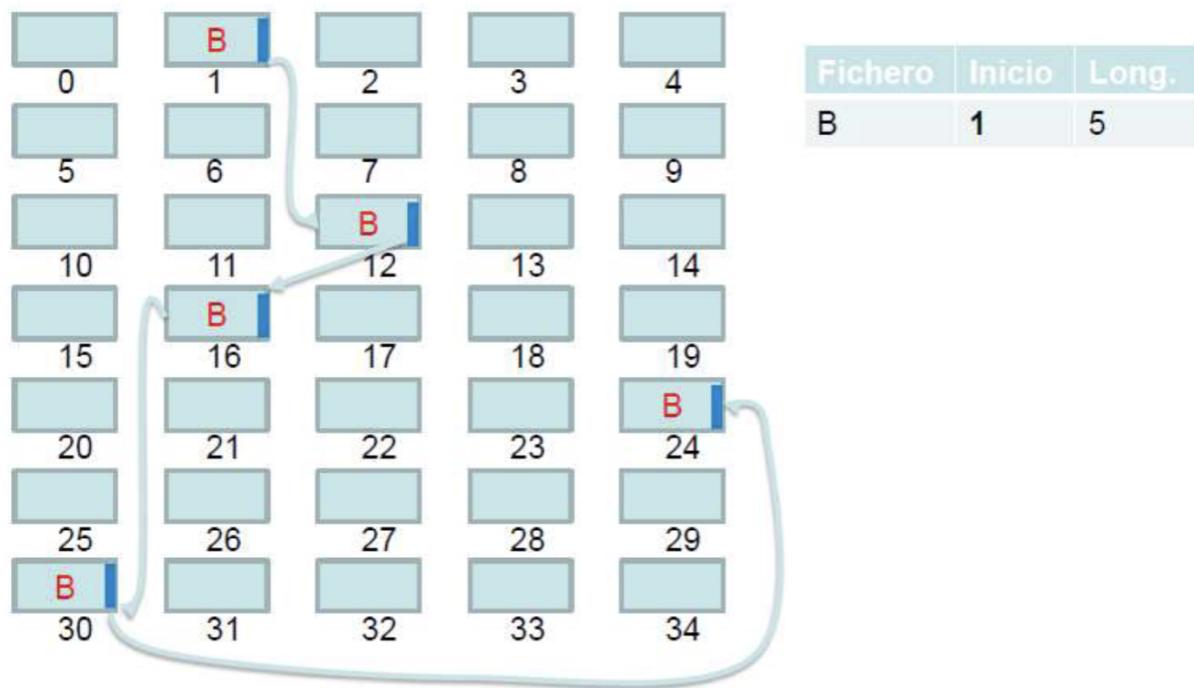
## 1) ASIGNACIÓN CONTIGUA

Se asigna un único conjunto contigo de bloques en tiempo de creación de los ficheros. Por lo tanto, hay una estrategia de *preasignación* que utiliza porciones de tamaño variable.



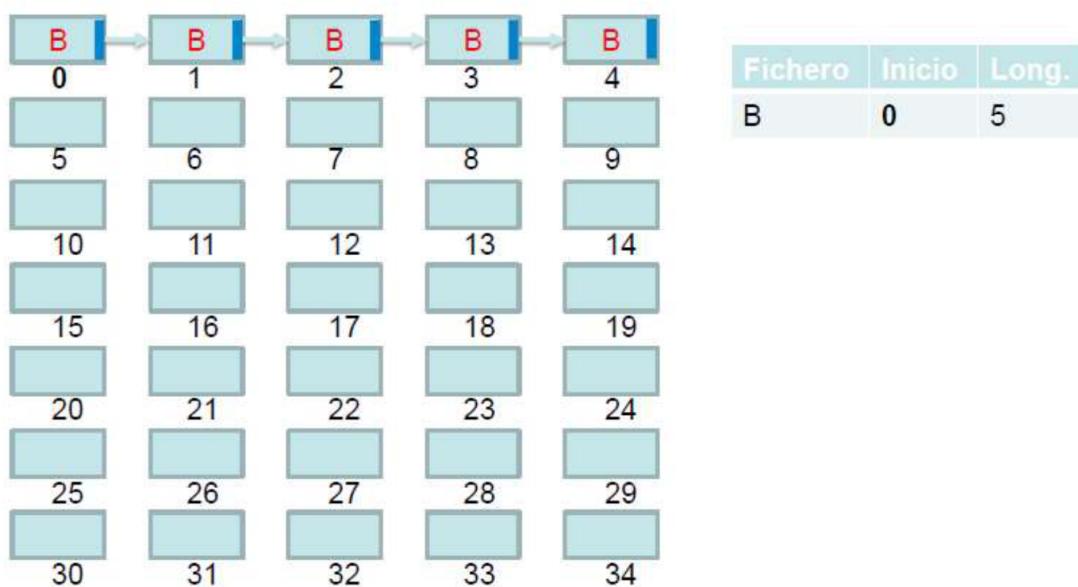
## 2) ASIGNACIÓN ENCADENADA

- Cada bloque contiene un puntero al bloque siguiente.
- Bloques distribuidos por todo el disco.
- Asignación de bloques uno en uno.



**Asignación Encadenada (Consolidación)**

ASO - UAI

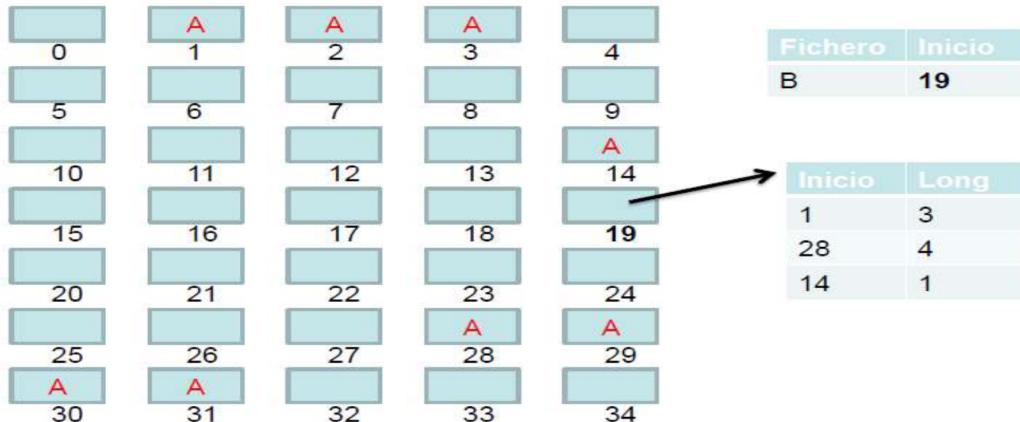
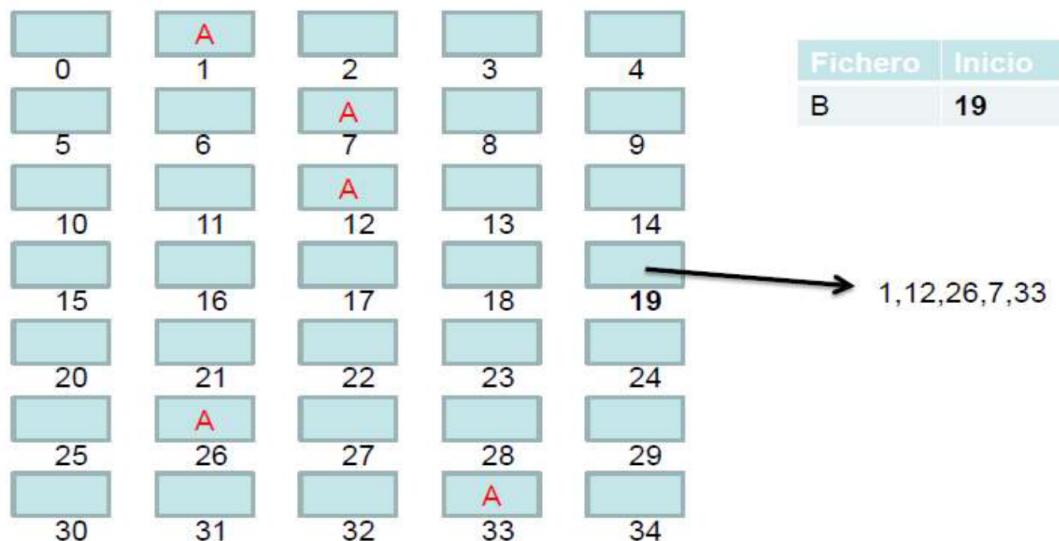


### 3) ASIGNACIÓN INDEXADA

Se mantiene una tabla con los identificadores de las unidades de asignación que forman el fichero.

Puede ser:

- Asignación por Bloque
- Asignación por Porciones



## **DIRECTORIOS**

Directorio = Carpeta

Es un objeto que relaciona de forma única un nombre de usuario de fichero con su descriptor interno.

Organiza y proporciona información sobre la estructuración de los sistemas de archivos.

- Un directorio posee una entrada por cada archivo que alberga y guarda información del descriptor interno del archivo y algunos atributos del archivo.

## **ALTERNATIVA DE ESTRUCTURA**

- Directorio de un único nivel.
- Directorio de dos niveles.
- Directorio con estructura de árbol.
- Directorio con estructura de grafo acíclico.
- Directorio con forma de grafo generalizado.

## **GESTIÓN DEL ESPACIO LIBRE**

El SO debe saber que bloques están libres.

¿Cómo?

- 1) **TABLA DE BITS**. Vector con un bit por bloque.
- 2) **LISTA ENCADENADA DE PORCIONES LIBRES**
- 3) **INDEXACIÓN**. Tabla índice de porciones libres.

## **Itinerario 13: Sistema de Ficheros**

### **SISTEMA DE FICHEROS Y PARTICIONES**

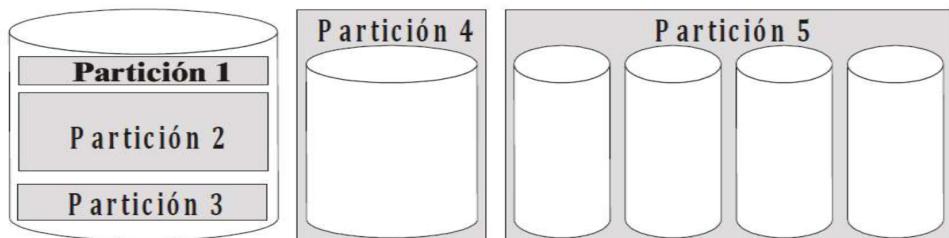
El **sistema de ficheros** permite organizar la información dentro de los dispositivos de almacenamiento secundario en un formato inteligible para el SO.

Previamente a la instalación del SF es necesario dividir físicamente, o lógicamente, los discos **en particiones**.

Una **partición** es una porción de disco a la que se la dota de una identidad propia y que puede ser manipulada por el SO como una entidad lógica independiente.

### Tipos de Particiones

ASO - UAI



### BLOQUES Y AGRUPACIONES

- **BLOQUE.** Agrupación lógica de sectores de disco
- **AGRUPACIÓN.** Conjunto de bloques que se gestionan como una unidad lógica de gestión de almacenamiento.

### SISTEMA DE FICHEROS WINDOWS

#### 1) **FAT (File Allocation Table)**

- Utilizado en algunas versiones de Windows.
- Se sigue utilizando para dispositivos de almacenamiento portátil.

El espacio está dividido en:

1. Bloque de Carga: Arranque del SO.
2. FAT: Tabla de asignación de ficheros.
3. Copia de FAT: Copia de seguridad.
4. Directorio Raíz: Directorio principal del volúmen.
5. Zona de directorios y ficheros.

## Sistema de Ficheros: FAT12

ASO - UAI

- Tabla con una posición por bloque de disco.
- Tabla con direcciones de 12 bits.
- Número máximo de bloques: 4096.
- Tamaño de bloque: 512 bytes a 8 KB.
- Tamaño máximo: 32 MB.
- Utilizado en disquetes.

## Sistema de Ficheros: FAT16

ASO - UAI

- Tabla con una posición por bloque de disco.
- Direcciones de 16 bits.
- Número máximo de bloques → 65.535.
- Tamaño de bloque: 512 bytes hasta 64KB.
- Utilizado en discos antiguos.

Tamaño de bloque	Tamaño máximo	Tamaño de bloque	Tamaño máximo
512 B	32 MB	8 KB	512 MB
1 KB	64 MB	16 KB	1 GB
2 KB	128 MB	32 KB	2 GB
4 KB	256 MB	64 KB	4 GB

**Tamaño máximo de la tabla FAT: 128 KB**

- Tabla con una posición por bloque de disco.
- Direcciones de 32 bits (solamente se usan 28).
- Número máximo de bloques → 256 Mbloques.
- Tamaño de bloque: 4 KB hasta 32 KB.
- Windows lo limita a dispositivos de hasta 32 GB.
- Usado en dispositivos de almacenamiento portátil.
- La tabla FAT puede ocupar cantidad considerable de espacio.
  - No se puede tener permanentemente en memoria y hay que consultarla en disco.
- Tamaño máximo de fichero: 4GB

## 2) NTFS

- Permite la recuperación de datos y fallos del disco.
- Soporta grandes volúmenes.
- Permite múltiples volúmenes de datos.
- Utiliza el modelo de objetos de Windows para forzar la seguridad.

Está formado por:

1. **SECTOR**. Unidad más pequeña. (512kB)
2. **CLÚSTER**. Uno o más sectores contiguos.
3. **VOLÚMEN**. Partición lógica del disco.

## Itinerario 14: Introducción a la Seguridad

### SEGURIDAD INFORMÁTICA

La seguridad de los sistemas informáticos y de la red va dirigida a cuatro requisitos básicos:

1. **CONFIDENCIALIDAD**. Requiere que la información sólo se encuentre accesible para lectura para aquellas partes que estén autorizadas a este tipo de acceso.

2. **INTEGRIDAD.** Requiere que los contenidos sólo podrán modificarse por las partes que se encuentren autorizadas.
3. **DISPONIBILIDAD.** Requiere que los componentes estén disponibles para todas las partes autorizadas.
4. **AUTENTICACIÓN.** Requiere que el sistema informático sea capaz de verificar la identidad de los usuarios.

## **TIPOS DE PELIGROS**

- **INTERRUPCIÓN.** Se destruye un componente del sistema o se encuentra no disponible. Es un ataque centrado en la **disponibilidad**. Ejemplo: Destrucción de un disco duro.
- **INTERCEPCIÓN.** Una parte no autorizada consigue acceso a un componente. Es un ataque dirigido a la **confidencialidad**. Ejemplo: La escucha de un canal de comunicación para la captura de datos.
- **MODIFICACIÓN.** Un elemento no autorizado no sólo tiene acceso a un componente sino que también es capaz de modificarlo. Éste es un ataque dirigido a la **integridad**. Ejemplo: Cambiar valores de un fichero.
- **FABRICACIÓN.** Un elemento no autorizado inserta objetos extraños en el sistema. Ataques contra la **autenticación**. Ejemplo: Inserción de mensajes externos.

## **COMPONENTES DE UN SISTEMA INFORMÁTICO**

**Tabla 16.1.** Peligros de seguridad y componentes.

	Disponibilidad	Privacidad	Integridad/Autenticación
<b>Hardware</b>	Equipamiento robado o deshabitado, por lo tanto denegación de servicio.		
<b>Software</b>	Borrado de programas, denegación de acceso a los usuarios.	Copia no autorizada de software.	Modificación de un programa, bien para hacer que falle durante la ejecución o para que realice una tarea diferente.
<b>Datos</b>	Borrar ficheros, denegación de acceso a los usuarios.	Lectura no autorizada de datos. Un análisis estadístico de los datos que revele la información subyacente.	Modificación de los ficheros existentes o creación de nuevos ficheros.
<b>Líneas de comunicación</b>	Borrado o destrucción de mensajes. Las líneas de comunicación o redes no se encuentran disponibles.	Lectura de mensajes. Observación de los patrones de tráfico de mensajes.	Modificación, borrado, reordenación o duplicación de mensajes. Fabricación de mensajes falsos.

## LÍNEAS DE COMUNICACIÓN

Un mecanismo útil para la clasificación de los ataques de seguridad a las redes es en base a los términos de **ataques pasivos** y **ataques activos**.

- 1) **ATAQUES PASIVOS.** Es la monitorización de las transmisiones. Objetivo: Obtener información de qué se está transmitiendo.

Son muy difíciles de detectar debido a que no implican ninguna alteración de los datos.

Ejemplos:

- Lectura de los contenidos de los mensajes
- Análisis de tráfico.

- 2) **ATAQUES ACTIVOS.** Implican algunas modificaciones en el flujo de datos.

Ejemplos:

- **Enmascaramiento:** Ocurre cuando el elemento intenta hacerse pasar por otro diferente.
- **Reenvío:** Implica la captura pasiva de una unidad de datos y su posterior retransmisión para producir un efecto no autorizado.
- **Modificación de mensajes:** Significa que una parte de un mensaje válido se ha alterado, o que los mensajes se han borrado para producir un efecto no autorizado.
- **Denegación de servicio:** Imposibilita el uso normal o la gestión de las instalaciones de comunicaciones.

## PROTECCIÓN

El SO debe ofrecer niveles de protección a lo largo del siguiente rango:

- 1) **SIN PROTECCIÓN ALGUNA.** Apropiado por los procedimientos que son sensibles de ejecutar en instantes diferentes.
- 2) **AISLAMIENTO.** Implica que cada proceso opera de forma separada con otros procesos, sin comunicación alguna.
- 3) **COMPARTICIÓN COMPLETA O SIN COMPARTICIÓN.** El propietario del objeto declara si va a ser público (cualquier proceso puede acceder al objeto) o privado (sólo los procesos del propietario pueden acceder a dicho objeto).
- 4) **COMPARTICIÓN VÍA LIMITACIONES DE USO.** El SO verifica la permisibilidad de cada acceso por parte de cada usuario específico sobre cada objeto.

- 5) **ACCESO VÍA CAPACIDADES DINÁMICAS.** Permite la creación dinámica de derechos de acceso a los objetos.
- 6) **USO LIMITADO DE UN OBJETO.** Limita no sólo el acceso a un objeto sino también el uso que se puede realizar sobre dicho objeto.

## INTRUSOS

- **ENMASCARADO.** Un individuo que no está autorizado a utilizar un ordenador y que penetra en los controles de acceso del sistema para aprovecharse de una cuenta de usuario legítimo.
- **TRANSGRESOR.** Un usuario legítimo que accede a datos para los cuales dicho acceso no está autorizado, o estando autorizado para dicho acceso utilizar sus privilegios de forma maliciosa.
- **USUARIO CLANDESTINO.** Un usuario sobrepasa el control de supervisión del sistema y usa dicho control para evadir la auditoría y el control de acceso.

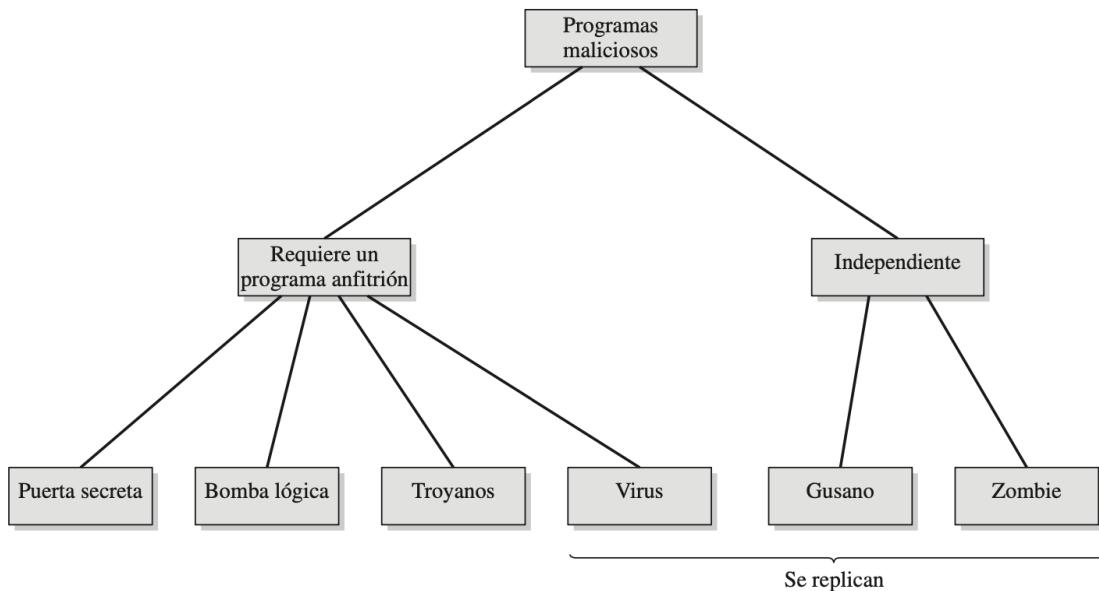
El intruso enmascarado se trata habitualmente de un usuario externo; el trasgresor generalmente es interno; el usuario clandestino puede ser interno o externo.

## TÉCNICAS DE DEFENSA

- **CIFRADO UNIDIRECCIONAL.** El sistema almacena únicamente una forma cifrada de la contraseña de usuario.
- **CONTROL ACCESO.** El acceso al fichero que contiene las contraseñas se encuentra limitado a una o muy pocas cuentas.

## SOFTWARE MALICIOSO

El **malware** es software diseñado para causar daño o utilizar recursos de un ordenador. Frecuentemente se encuentra escondido dentro de un programa enmascarado como software legítimo.



- 1) **BACKDOOR**. Es un punto de entrada secreta dentro de un programa que permite a alguien que conoce la existencia de dicha puerta secreta tener el acceso sin utilizar los procedimientos de acceso de seguridad estándar.
- 2) **BOMBA LÓGICA**. Es un código insertado dentro de un programa legítimo que explotará bajo ciertas condiciones. Una vez activada, la bomba puede alterar o borrar datos o ficheros completos, causando que la máquina se detenga, o que se produzca algún daño.
- 3) **TROYANO**. Es un programa útil que contiene un código oculto que, al invocarse, realiza una función no deseada o dañina. Se utilizan para realizar tareas de forma indirecta que el usuario no autorizado no podría realizar directamente.
- 4) **VIRUS**. Es un programa que puede infectar otros programas modificándolos; las modificaciones incluyen la copia del programa virus, que puede a continuación infectar a otros programas.

Pueden ser:

- Virus Parásito
- Virus residente en memoria.
- Virus en el sector de arranque.
- Virus oculto.
- Virus polimórfico.

- 5) **GUSANO**. Utilizan las conexiones de red para expandirse de un sistema a otro. Una vez se encuentran activos dentro de un sistema, un gusano de red se puede

comportar como un virus informático, puede implantar troyanos o realizar cualquier otro tipo de acciones destructivas.

- 6) **ZOMBIE**. Un programa zombie toma el control de otro ordenador conectado a internet y posteriormente utiliza el mismo para lanzar ataques que son difíciles de trazar como provenientes del creador del zombie. Se utilizan habitualmente para ataques de denegación de servicio.