Introdución aos Sistemas Operativos

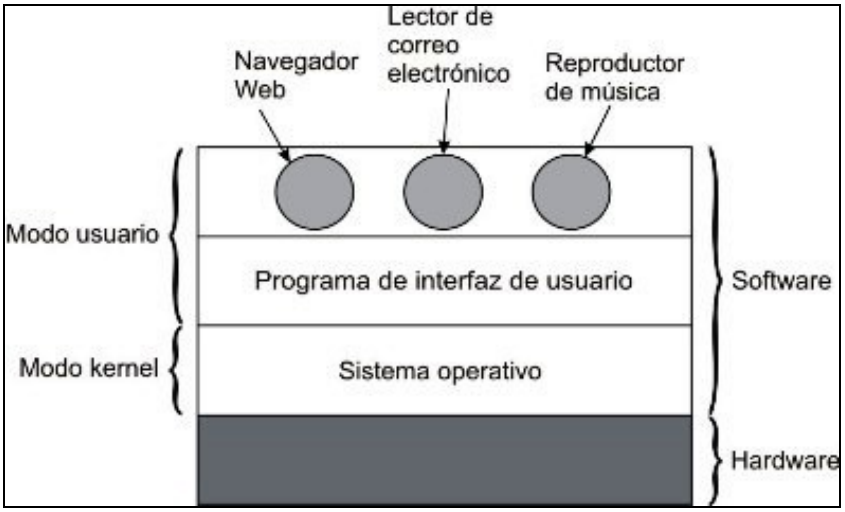
Sumario

Concepto de Sistema Operativo

Unha computadora moderna é un sistema complexo que consta de un ou mais procesadores, memoria principal, discos duros, impresoras, teclado, rato, pantalla, interface de rede e outros dispositivos de entrada/saída. Se os programadores de aplicacións tiveran que comprender o funcionamento de todos estes compoñentes, non escribirían código algún.

> "Por este motivo, as computadoras están equipadas cunha capa de *software* chamada **Sistema Operativo**, que ten como misión proporcionar aos programas de usuario un modelo de computadora mellor, mais simple e limpo, así como encargarse da administración de todos os recursos antes mencionados."

Cando un usuario interactúa cun Sistema Operativo como Windows, Linux, FreeBSD ou Mac OS X, faino cun programa que se denomina **shell**, cando está baseado en texto, e **GUI (*Graphical User Interface*, Interface Gráfica de Usuario)** cando utiliza elementos gráficos.



Na seguinte imaxe aparece un esquema xeral dos compoñentes principais dunha computadora e o seu *software*, nela podemos ver que a ubicación do Sistema Operativo no conxunto fai que este se execute directamente sobre o *hardware* e proporciona a base para as demais aplicacións *software*.

A maioría das computadoras teñen dous modos de operación: **modo kernel** e **modo usuario**.

- O Sistema Operativo é a peza fundamental do *software* e se executa en **modo kernel** (ou **modo supervisor**). Neste modo, o Sistema Operativo ten acceso completo a todo o *hardware* e pode executar calquera instrución que a máquina sexa capaz.
- O resto do *software* se executa en **modo usuario**, no que só un subconxunto de instrucións máquina están permitidas. En particular, as que afectan ao control da máquina ou as que se encargan da E/S están prohibidas para os programas que se executan neste modo.

Os Sistemas Operativos difieren dos programas de usuario en varias cuestións ademais do lugar onde residen. En particular, son "enormes", complexos e teñen unha vida útil moi longa. O código fonte dun SO como Linux ou Windows contén cerca de cinco millóns de liñas de código (só o kernel), polo que non é moi normal que cambien "a miúdo".

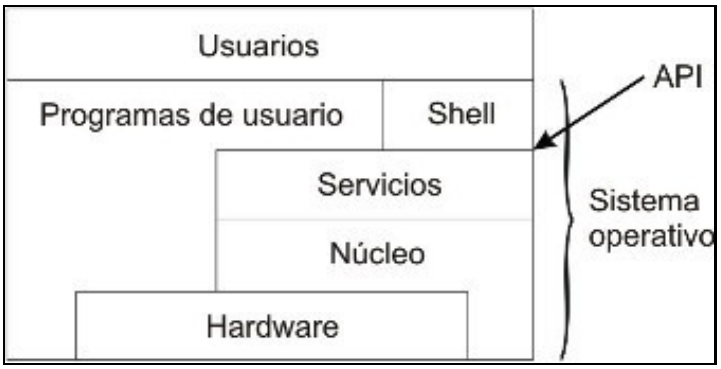
Funcións do Sistema Operativo

Un Sistema Operativo (SO) é un programa que ten encomendadas unha serie de funcións diferentes cuxo obxectivo é simplificar o manexo e a utilización da computadora, facéndoo seguro e eficiente.

Históricamente fóronse completando as misións encomendadas ao SO, polo que os produtos comerciais actuais inclúen unha grande cantidade de funcións, como son as interfaces gráficas, protocolos de comunicación, etc.

As funcións clásicas do sistema operativo pódense agrupar nas tres categorías seguintes:

- Xestión dos recursos da computadora.
- Execución de servizos para os programas - máquina estendida
- Execución dos mandatos dos usuarios - interface de usuario



Tal e como pode verse na Figura, o SO está formado conceptualmente por tres capas principais:

- **A capa mais próxima ao *hardware* denomínase núcleo (*kernel*)** e é a que xestiona os recursos *hardware* do sistema e a que subministra a funcionalidade básica do SO. **Esta capa execútase a nivel de núcleo,** mentres que as outras poden executarse en niveis menos permisivos.
- **A capa de servizos ou chamadas ao sistema** ofrece aos programas uns servizos en forma dunha interface de programación ou **API (*Application Programming Interface*).** Dende o punto de vista dos programas, esta capa estende a funcionalidade da computadora, polo que adóitase dicir que **o SO ofrece unha máquina virtual estendida aos programas.** Desta forma facilítase a elaboración destes, posto que se apoian nas funcións que lle subministra o SO.
- **A capa de intérprete de mandatos ou *shell*** subministra unha interface ao través da cal o usuario pode dialogar de forma interactiva coa computadora. O *shell* recibe os mandatos ou ordes do usuario, os interpreta e, se pode, os executa. Dado que o *shell* adoita executarse no nivel de usuario, algúns autores consideran que non forma parte do SO.

O Sistema Operativo como xestor de recursos

Nunha computadora actual adoitan coexistir varios programas, do mesmo ou de varios usuarios, executándose simultaneamente. Estes programas compiten polos recursos da computadora, sendo o SO o encargado de arbitrar a súa asignación e uso. Como complemento á xestión de recursos, o SO debe garantir a protección duns programas fronte a outros e debe subministrar información sobre o uso que se fai dos recursos.

a) Asignación de recursos.

O SO encárgase de asignar os recursos aos programas en execución. Para isto, mantén unhas estruturas que lle permiten saber que recursos están libres e cales están asignados a cada programa. A asignación de recursos realízase segundo a dispoñibilidade dos mesmos e a prioridade dos programas, debéndose resolver os conflitos que aparecen polas peticións simultáneas.
Especial mención reviste a recuperación dos recursos cando os programas xa non os precisan. Unha mala recuperación de recursos pode facer que o SO considere, por exemplo, que xa non lle queda memoria dispoñible cando, en realidade, si a ten. A recuperación pódese facer porque o programa que ten asignado o recurso lle comunica ao SO que xa non o precisa ou ben porque o programa xa terminara.
Os recursos manexados polo SO son físicos e lóxicos. Entre os físicos atópase o procesador, a memoria principal e os periféricos. Entre os lóxicos se poden citar os arquivos e os portos de comunicación.

b) Protección.

O SO garantirá a protección entre os usuarios do sistema. Asegurará a confidencialidade da información e que uns traballos non interfiran con outros. Para conseguir este obxectivo impedirá que uns programas poidan acceder aos recursos asignados a outros programas.

c) Contabilidade.

A contabilidade permite medir a cantidade de recursos que, ao longo da súa execución, utiliza cada programa. Deste xeito pódese coñecer a carga de utilización que ten cada recurso e pódese imputar a cada usuario os recursos que utilizou. Cando a contabilidade se emprega só para coñecer a carga dos compoñentes do sistema denomínase normalmente **monitorización.**

O Sistema Operativo como unha máquina estendida

O SO ofrece aos programas un conxunto de servizos, ou chamados ao sistema, que poden solicitar cando o precisen, proporcionando aos programas unha de máquina estendida.

O sistema operativo, como máquina estendida, proporciona unha abstracción do hardware, permitindo unha visión sinxela do mesmo e ofrecendo uns servicios para manipulalo sen ter que coñecer os detalles concretos físicos.
O modelo de programación que ofrece o *hardware* compleméntase con estes servicios *software*, que permiten executar de forma cómoda e

protexida certas operacións. A alternativa consistiría en complicar os programas de usuario e en non ter protección fronte a outros usuarios.
Os servicios pódense agrupar nas catro clases seguintes: execución de programas, operacións de E/S, operacións sobre arquivos e detección e tratamento de erros.

a) Execución de programas.

O SO inclúe servizos para lanzar a execución dun programa, así como para parala ou abortala. Tamén existen servizos para coñecer e modificar as condicións de execución dos programas, para comunicar e sincronizar uns programas con outros.
A execución de programas da lugar ao concepto de proceso. **Un proceso pódese definir como un programa en execución.** O proceso é un concepto fundamental nos SOs, posto que o obxectivo último destes é crear, executar e destruír procesos, de acordo ás ordes dos usuarios.
Para que un programa poida converterse nun proceso terá que estar traducido a código máquina e almacenado nun dispositivo de almacenamento como o disco. Baixo a petición dun usuario, o sistema operativo creará un proceso para executar o programa.

b) Operacións de E/S.

Os servizos de E/S ofrecen unha grande comodidade e protección pois dan aos programas operacións de lectura, escritura e modificación do estado dos periféricos. En efecto, a programación das operacións de E/S é moi complexa e dependente do *hardware* específico de cada periférico. Os servizos do SO ofrecen un alto nivel de abstracción de forma que o programador de aplicacións non teña que preocuparse deses detalles.

c) Operacións sobre arquivos.

Os arquivos ofrecen un nivel de abstracción maior que o das ordes de E/S, permitindo operacións tales como creación, borrado, renomeado, apertura, escritura e lectura de arquivos.
Observe que moitos dos servizos son parecidos ás operacións de E/S e terminan concretándose neste tipo de operacións.

d) Detección e tratamento de erros.

Ademais de analizar detalladamente todas as ordes que recibe, para comprobar que se poden realizar, o SO encárgase de tratar todas as condicións de erro que detecte o *hardware*.
Entre as condicións de erro que poden aparecer destacaremos as seguintes: erros nas operacións de E/S, erros de paridade nos accesos a memoria ou nos buses e erros de execución nos programas, como desbordamentos, violacións de memoria, códigos de instrución prohibidos, etc.

O sistema operativo como interface de usuario

O módulo do SO que permite que os usuarios dialoguen de forma interactiva có sistema é o **intérprete de mandatos ou *shell*.** O *shell* compórtase como un un bucle infinito que está repetindo constantemente a seguinte secuencia:

1. Espera unha orde do usuario. No caso de interface textual, ou *shell* está pendente do que escribe o usuario na liña de mandatos. Nas interfaces gráficas (GUI) está pendente dos eventos do apuntador (rato) que manipula o usuario, ademais, dos do teclado.
2. Analiza a orde e, en caso de ser correcta, a executa, empregando os servizos do SO.
3. Concluída a orde volve á espera.

O diálogo mediante interface textual esixe que o usuario memorice a sintaxe dos mandatos, coa agravante de que son distintos para cada SO (por exemplo: para listar o contido dun arquivo en MS-DOS emprégase *type*, pero en UNIX emprégase o mandato *cat*). Por esta razón son mais populares os intérpretes de mandatos con interface gráfica, como o de WINDOWS, aínda que os administradores de sistemas seguen preferindo o intérprete de comandos e a súa automatización con *shell scripts*.

 Arquivos de Mandatos ou *Shell Scripts*

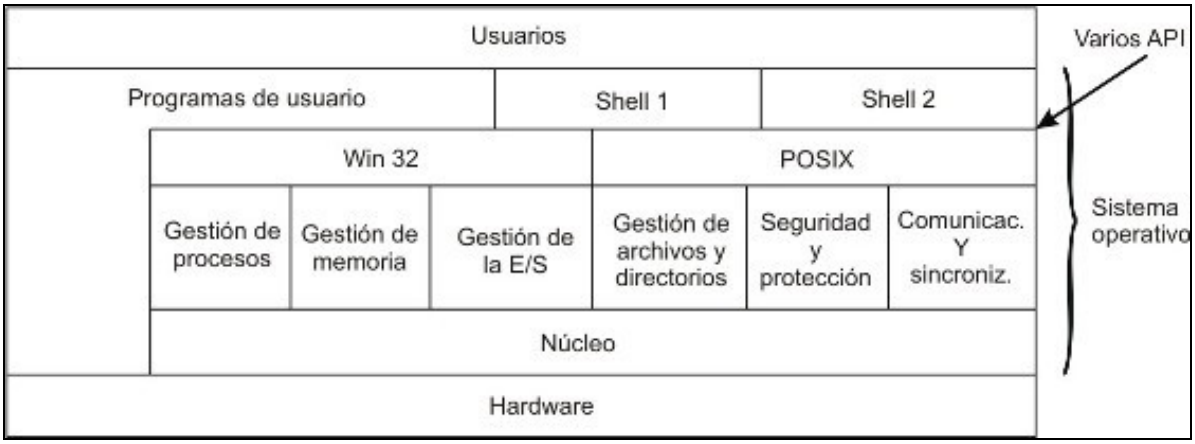
Case todos os intérpretes de mandatos poden executar arquivos de mandatos, estes arquivos inclúen varios mandatos dun xeito secuencial, ademais de funciones mais complexas. Ademais, poden incluír control de fluxo de execución, como poden ser sentencias *goto*, *for* e *if*, así como etiquetas para identificar liñas de mandatos. Para executar un arquivo de mandatos normalmente basta con invocalo de igual xeito que un mandato estándar do intérprete de comandos.

Compoñentes e estrutura do Sistema Operativo

O SO está formado por unha serie de compoñentes especializados en determinadas funcións. Cada SO estrutura estes compoñentes de forma distinta. Nesta sección descríbense, en primeiro lugar, os distintos compoñentes que conforman un SO, para pasar, a continuación, a ver as distintas formas que teñen os SO de estruturar estes compoñentes.

Compoñentes do Sistema Operativo

Como se comentou previamente e pode verse na seguinte figura, normalmente considérase que un SO está formado por tres capas: o núcleo, os servizos e o intérprete de mandatos ou *shell*.



- **O núcleo** é a parte do SO que se comunica directamente có *hardware* da máquina. As funcións do núcleo céntranse na xestión de recursos, como o procesador, tratamento de interrupcións e as funcións básicas de manipulación de memoria.
- **Os servizos** adóitanse agrupar segundo a súa funcionalidade en varios compoñentes, cada un dos cales se ocupa das seguintes funcións:

  - Xestión de procesos. Encargada da creación, planificación e destrución de procesos.
  - Xestión de memoria. Compoñente encargada de saber que partes da memoria están libres e cales ocupadas, así como da asignación e liberación de memoria segundo a necesiten os procesos.
  - Xestión da E/S. Ocúpase de facilitar o manexo dos dispositivos periféricos.
  - Xestión de arquivos e directorios. Encárgase do manexo de arquivos e directorios e da administración do almacenamento secundario.
  - Comunicación e sincronización entre procesos. Encargada de ofrecer mecanismos para que os procesos poidan comunicarse e sincronizarse.
  - Seguridade e protección. Este compoñente debe encargarse de garantir a identidade dos usuarios e de definir o que poden facer cada un deles cos recursos do sistema.

Todos estes compoñentes ofrecen unha serie de servizos a través dunha interface de chamadas ao sistema. Como pode verse na figura anterior, un SO pode incluír mais dunha interface de servizos (na figura anterior consideráronse as interfaces Win32 e POSIX). Neste caso, os programas poderán elixir sobre que interface queren executar, pero non poderán misturar servizos de varias interfaces. Dise, neste caso, que o SO presenta ao usuario varias máquinas virtuais.

De igual forma, o SO pode incluír varios intérpretes de mandatos, uns textuais e outros gráficos, podendo o usuario elixir o que mais lle interese. Con todo, hai que observar que non se poderán misturar mandatos de varios intérpretes.

Estrutura do Sistema Operativo

Un SO é un programa grande e complexo que está composto, tal e como vimos, por unha serie de compoñentes con funcións ben definidas. Cada SO estrutura estes compoñentes de distinta forma. En función desta estrutura pódense agrupar os SOs en dous grandes grupos: **sistemas operativos monolíticos** e **sistemas operativos estruturados**.

- **Sistemas operativos monolíticos.**

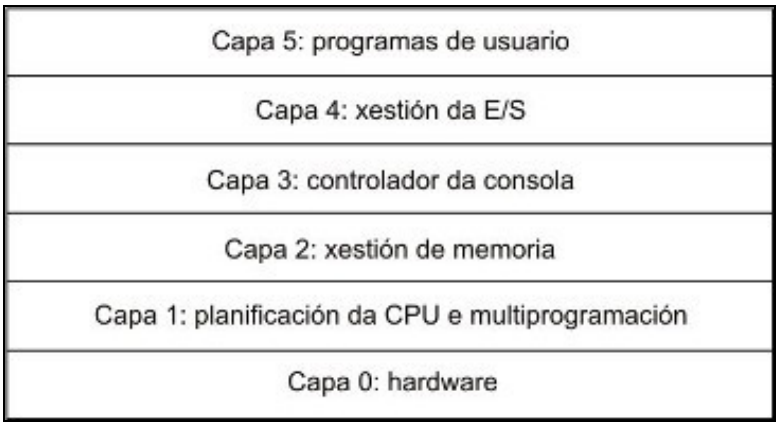
  Un SO deste tipo non ten unha estrutura clara e ben definida. Todos os seus compoñentes atópanse integrados nun único programa (o SO) que se executa nun único espacio de direccións. Neste tipo de Sistemas todas as funcións que ofrece o SO se executan en modo núcleo. Estes SOs xurdiron, normalmente, de SOs sinxelos e pequenos aos que se lles foron engadindo un número maior de funcionalidades. Isto

lles fixo evolucionar e crecer ata convertelos en programas grandes e complexos formados por moitas funcións situadas todas elas nun mesmo nivel. **Exemplos claros deste tipo de sistemas son MS-DOS, UNIX e Linux.** Ambos comezaron sendo pequenos SOs, que foron facéndose cada vez mais grandes debido á grande popularidade que adquiriron.
O problema que expón este tipo de sistemas radica no complicado que é modificar o SO para engadir novas funcionalidades e servizos. En efecto, engadir unha nova característica o SO implica a modificación dun gran programa, composto por miles de liñas de código fonte e funcións, cada unha das cales pode invocar a outras cando así o requira. Ademais, neste tipo de sistemas non se segue o principio de ocultación da información. Para solucionar este problema é necesario dotar de certa estrutura ao SO.

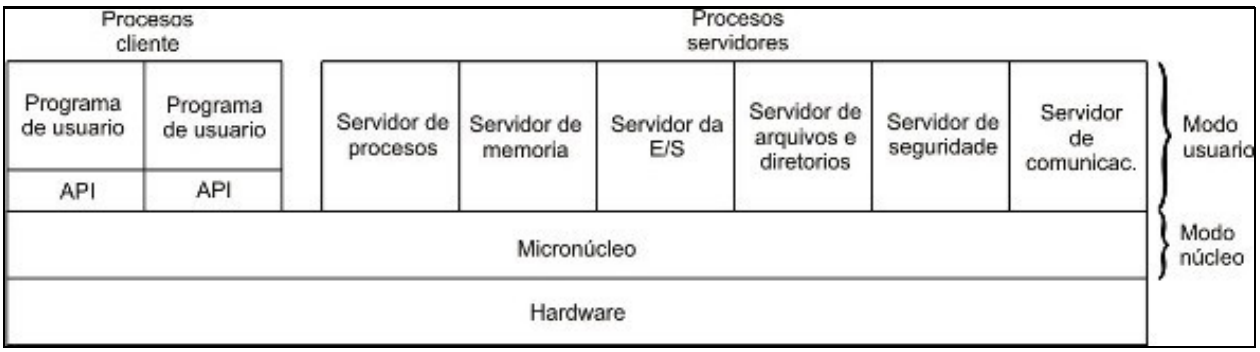
- **Sistemas operativos estruturados.**

Cando se quere dotar de estrutura a un SO, normalmente recórrese a dous tipos de solucións: **Sistemas por capas** e **Sistemas cliente-servidor**.

a) Sistemas por capas.
Nun sistema por capas, o SO organízase como unha xerarquía de capas, onde cada capa ofrece unha interface clara e ben definida á capa superior e só utiliza os servizos que lle ofrece a capa inferior.
A principal vantaxe que ofrece este tipo de estruturas é a modularidade e a ocultación da información. Unha capa non precisa coñecer como se implementou a capa sobre a que se constrúe, unicamente precisa coñecer a interface que ofrece. Isto facilita enormemente a depuración e verificación do sistema, posto que as capas pódense ir construíndo e depurando por separado.
Este enfoque utilizouno por primeira vez o SO **THE** [1968], un SO sinxelo que estaba formado por seis capas, como pode verse na figura seguinte. Outro exemplo de SO deseñado por capas é o **OS/2** [1994], descendente de **MS-DOS**.



b) Modelo cliente-servidor.
Neste tipo de modelo, o enfoque consiste en implementar a maior parte dos servizos e funcións do SO en procesos de usuario, deixando só unha pequena parte do SO executando en modo núcleo. A esta parte denomínase **micronúcleo** e aos procesos que executan o resto de funcións se lles denomina servidores. A figura seguinte presenta a estrutura dun SO con estrutura cliente-servidor. Como pode apreciarse na figura, o SO está formado por diversas partes, cada unha das cales pode desenrolarse por separado.



Non hai unha definición clara das funcións que debe levar a cabo un micronúcleo. A maioría inclúen a xestión de interrupcións, xestión básica de procesos e de memoria e servizos básicos de comunicación entre procesos. Para solicitar un servizo neste tipo de sistemas, como por exemplo crear un proceso, o proceso de usuario (proceso denominado cliente) solicítao ao servidor do SO correspondente, neste caso ao servidor de procesos. Á súa vez, o proceso servidor pode requirir os servizos de outros servidores, como é o caso do servidor de memoria. Neste caso, o servidor de procesos convertese en cliente do servidor de memoria.
A vantaxe deste modelo é a grande flexibilidade que presenta. Cada proceso servidor só se ocupa dunha funcionalidade concreta, o que fai que cada parte poida ser pequena e manexable. Isto, á súa vez, facilita o desenrolo e depuración de cada un dos procesos servidores.
En canto ás desvantaxes, citar que estes sistemas presentan unha maior sobrecarga no tratamento dos servizos que os sistemas monolíticos. Isto débese a que os distintos compoñentes dun SO deste tipo executan espazos de direccións distintos, o que fai que a súa activación requira mais tempo.
Minix [1998], **Mach** [1986] e **Amoeba** [1990] son exemplos de SOs que seguen este modelo. **WINDOWS NT**, **Windows 2000** e **XP** tamén seguen esta filosofía de deseño, aínda que moitos dos servidores (o xestor de procesos, xestor de E/S, xestor de memoria, etc.) execútanse en modo núcleo por razóns de eficiencia.

Procesos e fíos

Os procesos son unha das abstraccións mais antigas e importantes que proporcionan os SOs: proporcionan a capacidade de operar (*pseudo*) concorrentemente, incluso cando hai só unha CPU dispoñible. Converten unha CPU en varias CPUs virtuais. Sen a abstracción dos procesos, a computación moderna non podería existir.

Procesos

O proceso pódese definir como un programa en execución.

> Todo o *software* executable na computadora organízase en varios "procesos secuenciais". Así, un proceso non é mais que unha instancia dun programa en execución, incluíndo os valores actuais do contador de programa, os rexistros e as variables. En teoría, cada proceso ten a súa propia CPU virtual;

na realidade, a CPU real conmuta dun proceso a outro, pero para entender o sistema é moito mais fácil pensar nunha colección de procesos que se executan en *pseudo*-paralelo, en vez de tratar de levar a conta de como a CPU conmuta de programa en programa. Esta conmutación rápida dun proceso a outro coñécese como como "multiprogramación".

> **A diferencia entre un proceso e un programa é sutil pero crucial. A idea clave é que un proceso é unha actividade de certo tipo: ten un programa a seguir (o algoritmo), unha entrada, unha saída e un estado.** Varios procesos poden compartir un só procesador mediante o uso dun algoritmo de planificación para determinar cando se debe deter o traballo nun proceso para dar servizo a outro.
> O sistema operativo ofrece unha serie de servizos que permiten definir a vida dun proceso. Esta vida está constituída polas seguintes fases: **creación, execución e morte do proceso.**

Crear un proceso

Hai catro eventos principais que provocan a creación de procesos:

1. O arranque do sistema.
2. A execución, dende un proceso, dunha chamada ao sistema para creación de procesos.
3. Unha petición de usuario para crear un proceso.
4. O inicio dun traballo por lotes.

> Os procesos que permanecen en segundo plano para manexar certas actividades como correo electrónico, páxinas Web, novas, impresións, etc, coñécense como **demos (*daemons*)**. En UNIX/Linux podemos utilizar o programa *ps* para listar os procesos en execución, en Windows podemos empregar o *Administrador de Tarefas*.
> En todos estes casos, para crear un proceso hai que facer que un proceso existente execute unha chamada ao sistema de creación de procesos. Esta chamada ao sistema indica ao SO que cree un proceso e lle indica, directa ou indirectamente, que programa debe executalo.
> En UNIX/Linux só hai unha chamada ao sistema para crear un proceso: *fork*. Esta chamada crea un clon exacto do proceso que fixo a chamada. Despois de *fork*, os dous procesos (pai e fillo) teñen a mesma imaxe de memoria, as mesmas cadeas de entorno e os mesmos arquivos abertos.
> Polo contrario, en Windows unha soa chamada a unha función de Win32 (*CreateProcess*) manexa a creación de procesos e carga o programa correcto no novo proceso. Esta chamada ten 10 parámetros, que inclúen o programa a executar, os parámetros da liña de comandos para introducir datos a ese programa, varios atributos de seguridade, bits que controlan se os arquivos abertos se herdan,

información de prioridade, unha especificación da fiestra que se vai a crear para o proceso (se se vai crear unha) e un apuntador a unha estrutura onde se devolve ao proceso que fixo a chamada á información acerca do proceso recentemente creado. Ademais de *CreateProcess*, Win32 ten cerca de 100 funcións mais para administrar e sincronizar procesos e temas relacionados.
Tanto en Unix como en Windows, unha vez que se crea un proceso, o pai e o fillo teñen os seus propios espazos de direccións distintos.

Executar un proceso

Os procesos poden executarse de dúas formas: ***batch* e interactiva**.

- Un proceso que se executa en modo *batch*, tamén chamado *background*, deberá tomar os seus datos de entrada dun arquivo e logo depositar os seus resultados noutro arquivo.
- Un proceso que se executa en modo interactivo, recibe a información do usuario por un terminal polo que contesta cós resultados.

Terminación de procesos

Tarde ou cedo, todo proceso terminará. Polo xeral, os procesos terminan por un dos seguintes motivos:

1.- Saída normal (voluntaria).
A maioría dos procesos terminan debido a que concluíron o seu traballo. Cando un compilador terminou de compilar o programa que recibe, executa unha chamada ao sistema para indicar ao SO que terminou. Esta chamada é *exit* en UNIX e *ExitProcess* en Windows.
2.- Saída por error (voluntaria).
3.- Erro (involuntaria).
4.- Eliminado por outro proceso (involuntaria).
Pódese executar unha chamada ao sistema que indique ao SO que elimine outros procesos. En UNIX esta chamada é *kill*. A función correspondente en Win32 é *TerminateProcess*.

Implementación dos procesos

Para implementar o modelo de procesos, o SO mantén unha táboa chamada **táboa de procesos,** con só unha entrada por cada proceso (algúns autores chaman a estas entradas **bloques de control de procesos**). Esta entrada contén información importante acerca do estado do proceso, incluíndo o seu contador de programa, apuntador de pila, asignación de memoria, estado dos seus arquivos abertos, información de contabilidade e planificación, e todo o demais que debe gardarse acerca do proceso cando este cambia de estado "en execución" a "listo" ou "bloqueado", de maneira que se poida reiniciar posteriormente como se nunca se detivera.
A seguinte táboa mostra algúns campos clave nun sistema típico. Os campos na primeira columna relaciónanse coa administración de procesos; os outros dous se relacionan coa administración de memoria e arquivos, respectivamente. Hai que recalcar que os campos contidos na táboa de procesos varía dun sistema a outro, pero nesta táboa dásenos unha idea xeral do tipo de información precisa.

Administración de procesos	Administración de memoria	Administración de arquivos
Rexistros	Apuntador á información do segmento de texto	Directorio raíz
Contador do programa	Apuntador á información do segmento de datos	Directorio de traballo
Palabra de estado do programa	Apuntador á información do segmento de pila	Descritores de arquivos
Apuntador da pila		ID de usuario
Estado do proceso		ID de grupo
Prioridade		
Parámetros de planificación		
ID do proceso		
Proceso pai		
Grupo de procesos		

Administración de procesos	Administración de memoria	Administración de arquivos
Sinais Tempo de inicio do proceso Tempo utilizado da CPU Tempo da CPU utilizado polo fillo Hora da seguinte alarma		

Con cada clase de E/S hai unha localización asociada chamada **vector de interrupción.** Esta localización contén a dirección do procedemento do servizo de interrupcións.
Podemos ver, de xeito resumido, o manexo de interrupcións e a planificación de procesos:

1. O *hardware* mete o contador do programa á pila, etc.
2. O *hardware* carga o novo contador de programa do vector de interrupcións.
3. Procedemento en linguaxe ensamblador garda os rexistros.
4. Procedemento en linguaxe ensamblador establece a nova pila.
5. O servizo de interrupcións de C se executa (polo xeral lee e garda a entrada no búfer).
6. O planificador decide que proceso se vai a executar a continuación.
7. Procedemento en C regresa ao código de ensamblador.
8. Procedemento en linguaxe ensamblador inicia o novo proceso actual.

Cando o proceso remata, o SO mostra un carácter indicador e espera un novo comando. Cando recibe o comando, carga un novo programa en memoria, sobrescribindo o anterior.

Fíos

Nos SOs tradicionais, cada proceso ten un espazo de direccións e un só fío de control. Con todo, con frecuencia hai situacións nas que é conveniente ter varios fíos de control no mesmo espazo de direccións que se executan en *cuasi*-paralelo, como se fosen procesos (case) separados (excepto polo espazo de direccións compartido).

Uso de fíos

A principal razón de ter fíos é que en moitas aplicacións desenvólvense varias actividades á vez. Algunhas desas pódense bloquear de vez en cando. Ao descompoñer unha aplicación en varios fíos secuenciais que se executan en *cuasi*-paralelo, o modelo de programación simplifícase.

- A xustificación de ter procesos é a mesma que a de ter fíos. **Só que agora, cós fíos, agregamos a habilidade das entidades en paralelo de compartir un espazo de direccións e todos os seus datos entre elas.** Esta habilidade é esencial para certas aplicacións, razón pola que non funcionará o ter varios procesos (cós seus espazos de direccións separados).
- Un segundo argumento para ter fíos é que, como **son mais lixeiros que os procesos,** son mais fáciles de crear (mais rápido) e destruír. En moitos sistemas, a creación dun fío é de 10 a 100 veces mais rápida que a dun proceso. Cando o número de fíos necesarios cambia de maneira dinámica e rápida, é útil ter esta propiedade.
- Unha terceira razón de ter fíos é tamén un argumento relacionado có rendemento. Os fíos non producen un aumento no rendemento cando todos eles están ligados á CPU, pero cando hai unha cantidade considerable de cálculos e **operacións de E/S, ao ter fíos estas actividades poden solaparse,** có que se axiliza a velocidade da aplicación.
- A cuarta e última razón é que **os fíos son moi útiles en sistemas con varias CPUs** onde, gracias a eles, é posible conseguir o verdadeiro paralelismo.

Planificación

Cando un ordenador se multiprograma, con frecuencia ten varios procesos ou fíos que compiten pola CPU ao mesmo tempo. Se só temos unha CPU dispoñible, hai que decidir que proceso se vai a executar a continuación. A parte do SO que realiza esta decisión coñécese como **planificador de procesos** e o algoritmo que utiliza coñécese como **algoritmo de planificación.**

Moitas das cuestións que se aplican á planificación de procesos tamén se aplican á planificación de fíos, aínda que algunhas son distintas. Cando o *kernel* administra fíos, polo xeral a planificación lévase a cabo por fío, e importa moi pouco (ou nada) a que proceso pertence ese fío.

Dividiremos os algoritmos de planificación atendendo a tres contornas distintas:

1. **Procesamento por lotes.** Nestes sistemas non hai usuarios que esperen unha resposta rápida á súa petición. Polo tanto, son aceptables os algoritmos non apropiativos. Este método reduce a conmutación de procesos e polo tanto, mellora o rendemento. Nesta contorna búscase mellorar o rendemento (maximizando o número de traballos por hora), diminuír o tempo de retorno (minimizando o tempo entre a entrega e a terminación) e manter ocupada a CPU todo o tempo.
2. **Interactivo.** Nesta contorna é preciso evitar que un proceso acapare a CPU e negue o servizo ás demais. Búscase que se responda ás peticións coa maior rapidez posible e así cumprir as expectativas dos usuarios.
3. **De tempo real.** Nestes sistemas búscase evitar, sobre todo, a perda de datos e evitar a degradación da calidade nos sistemas multimedia.

Planificación en sistemas de procesamento por lotes

- **Primeiro en entrar, primeiro en ser atendido (FCFS,*First-Come, First-Served*).**

Trátase dun algoritmo "non apropiativo". Con este algoritmo, a CPU asígnase aos procesos na orden na que a solicitan. Ningún proceso se interrompe debido a que se executa durante moito tempo.

- **O traballo mais curto primeiro.**

Tamén é un algoritmo non apropiativo, pero neste caso suponse que os tempo de execución coñécense de antemán. Cando hai varios traballos de igual importancia esperando a seren iniciados na cola de entrada, o planificador selecciona **o traballo mais curto primeiro (SJF, *Shortest Job First*).**

- **O menor tempo restante a continuación.**

Unha versión do algoritmo do tipo "o traballo mais curto primeiro" é **o menor tempo restante a continuación (SRTN, *Shortest Remaining Time Next*).** Con este algoritmo, o planificador sempre selecciona o proceso cuxo tempo restante de execución sexa o mais curto. Se o novo traballo precisa menos tempo para terminar que o proceso actual, este se suspende e o novo traballo se inicia.

Planificación en sistemas interactivos

- **Planificación por quenda circular (*round-robin*).**

A cada proceso se lle asigna un intervalo de tempo, coñecido como ***quántum***, durante o que se lle permite executarse. Se o proceso segue executándose ao final dese tempo, a CPU é apropiada para darlla a outro proceso. Se o proceso se bloquea ou remata antes de que remate o tempo, a conmutación da CPU realízase cando iso ocorre. O tempo do ***quántum*** é unha cuestión interesante: se é demasiado curto, prodúcense demasiadas conmutacións entre procesos e se reduce a eficiencia da CPU, pero si se establece demasiado longo pódese producir unha mala resposta ás peticións interactivas curtas. Normalmente, o ***quántum*** ten unha duración de entre 20 e 50 mseg.

- **Planificación por prioridade.**

A cada proceso se lle asigna unha prioridade e o proceso executable coa prioridade mais alta é o que se pode executar. Para evitar que os procesos con alta prioridade se executen de xeito indefinido, o planificador pode reducir a prioridade do proceso actual en execución en cada pulso de reloxo.

- **O proceso mais curto a continuación.**
- **Planificación garantida.**
- **Planificación por sorteo.**
- **Planificación por partes equitativas.**

Planificación en sistemas de tempo real

Nun sistema de **tempo real**, o tempo desempeña un papel esencial. Polo xeral, un ou mais dispositivos físicos externos á computadora xeran estímulos e esta debe reaccionar de maneira apropiada a eles dentro de certo espazo de tempo.

En xeral, os sistema de tempo real categorízanse como de **tempo real duro**, o que significa que hai tempos límite absolutos que se deben cumprir, e como de **tempo real suave**, o que significa que non é axeitado saírse dese tempo a cumprir pero si é tolerable. En ambos casos, o comportamento en tempo real acádase dividindo o programa en varios procesos, onde o comportamento de cada un destes é predicible e coñécese de antemán e teñen un tempo de vida curto.

Os algoritmos de planificación en tempo real poden ser **estáticos** e **dinámicos**. Os primeiros toman as súas decisión antes de que o sistema comece a executarse. Os segundos o fan durante o tempo de execución.

Administración de memoria

O xestor de memoria é un dos compoñentes principais do sistema operativo. Encárgase de:

- Asignar memoria aos procesos e liberala cando o requiran.
- Permitir que os procesos poidan compartir memoria entre eles, e así comunicarse.
- Xestionar a xerarquía de memoria.

Ademais destas funcións, o xestor de memoria, na categoría de servizos aos programas subministra os seguintes servizos: o de solicitar memoria, o de liberala e o de permitir que os procesos compartan memoria.

Existen varios esquemas de administración de memoria, que varían dende moi simples ata moi sofisticados.

Sen abstracción de memoria

É, sen dúbida, a abstracción mais simple de memoria. As primeiras computadoras non tiñan abstracción de memoria. Cada programa vía simplemente a memoria física, polo que nos sistemas que se organizan deste xeito, polo xeral, só poden executar un proceso de cada vez.

Un xeito de obter certo grao de paralelismo nun sistema sen abstracción de memoria, é programar con múltiples fíos. Como se supón que todos os fíos nun proceso ven a mesma imaxe de memoria, o feito de que se vexan obrigados a facelo non é un problema. Aínda que esta idea funciona, é de uso limitado xa que moitas veces deséxase que programas "non relacionados" se executen ao mesmo tempo, algo que a abstracción sen fíos non proporciona.

Unha abstracción de memoria: espazos de direccións

Un **espazo de direccións (*address space*)** é o conxunto de direccións que pode utilizar un proceso para direccionar a memoria. Cada proceso ten o seu propio espacio de direccións, independentemente dos que pertencen a outros procesos (excepto en certas circunstancias especiais onde os procesos desexan compartir os seus espazos de direccións).

Unha tarefa difícil para os SOs é proporcionar a cada programa o seu propio espacio de direccións. Sobre todo, tendo en conta que a cantidade de memoria necesaria para executar todos os procesos requiridos é, normalmente, maior que a cantidade de memoria existente:

 Intercambio

Esta estratexia consiste en levar cada proceso completo a memoria, executalo durante certo tempo e despois devolvelo a disco. En contraposición coa estratexia denominada **Memoria virtual,** que permite que os programas se executen, incluso, cando só se atopan en forma parcial na memoria.

Memoria virtual

A idea básica deste método é que cada programa ten o seu propio espazo de direccións, que se divide en anacos chamados **páxinas**. Cada páxina é un rango contiguo de direccións. Estas páxinas asócianse á memoria física, pero non todas teñen que estar na memoria física para poder executar o programa. Cando o programa fai referencia a unha parte do seu espazo de direccións que está na memoria física, o *hardware* realiza a asociación necesaria ao instante. Cando o programa fai referencia a unha parte do seu espazo de direccións que non está na memoria física, o SO recibe unha alerta para buscar a parte que falta e volver a executar a instrución que faiou.

- **Paxinación.**

  Cando un programa se executa xera unha serie de direccións coñecidas como **direccións virtuais** e que forman o **espazo de direccións virtuais**. Nos equipos sen memoria virtual, a dirección física colócase directamente no bus de memoria e fai que se lea ou escriba a palabra de memoria física coa mesma dirección. Cando se utiliza memoria virtual, as direccións virtuais non van directamente ao bus de memoria. En vez disto, van a unha **MMU (*Memory Management Unit*, Unidade de administración de memoria)** que asocia as direccións virtuais ás direccións de memoria físicas.
  O espazo de direccións virtuais divídese en unidades de tamaño fixo chamadas páxinas. As unidades correspondentes na memoria física chámanse **marcos de páxina**. As páxinas e os marcos de páxina son, polo xeral, do mesmo tamaño (de 512B a 64KiB). As transferencias entre a RAM e o disco sempre son en páxinas completas.

- **Táboas de páxinas.**

Nunha implementación simple, a asociación de direccións virtuais a direccións físicas pódese resumir do seguinte xeito: a dirección virtual divídese nun número de páxina virtual (bits de maior orden) e nun desprazamento (bis de menor orden). Por exemplo, cunha dirección de 16 bits e un tamaño de páxina de 4KiB, os 4 bits superiores poderían especificar unha das 16 páxinas virtuais e os 12 bits inferiores poderían entón especificar o desprazamento de bytes (0 a 4095) dentro da páxina seleccionada. Non obstante, tamén é posible unha división con 3, 5 ou outro número de bits para a páxina. As distintas divisións implican diferentes tamaños de páxina.

Xestión de Entrada/Saída

Unha das principais funcións dun SO é a xestión dos recursos da computadora e, en concreto, dos dispositivos periféricos. O xestor de E/S debe controlar o funcionamento de todos os dispositivos de E/S para acadar os seguintes obxectivos:

- Facilitar o manexo dos dispositivos periféricos. Para isto debe ofrecer unha interface sinxela, uniforme e fácil de empregar entre os dispositivos, e xestionar os erros que se poden producir no acceso aos mesmos.
- Ofrecer mecanismos de protección que impidan aos usuarios acceder sen control aos dispositivos periféricos.

Principios do *hardware* de E/S

- **Dispositivos de E/S**

  Os dispositivos de E/S pódense dividir basicamente en dúas categorías:

  - Dispositivos de bloque: Os dispositivos de bloque almacenan información en bloques de tamaño fixo, cada un coa súa propia dirección. Os tamaños de bloque comúns varían dende 512 bytes ata 32.768 bytes. Todas as transferencias realízanse en unidades de un ou mais bloques completos (consecutivos). A propiedade esencial dun dispositivo de bloque é que é posible ler ou escribir cada bloque de maneira independente dos demais. Os discos duros, CD-ROMs e memorias USBs son dispositivos de bloque comúns. As cintas tamén se consideran dispositivos de bloque, pero non de acceso aleatorio.
  - Dispositivos de carácter: Un dispositivo de carácter envía ou acepta fluxo de caracteres, sen importar a estrutura do bloque. Non é direccionable e non ten ningunha operación de busca. As impresoras, interfaces de rede, os ratos e a maioría dos demais dispositivos que non son parecidos aos discos pódense considerar como dispositivos de carácter.

- **Controladores de dispositivos**

  Polo xeral, as unidades de E/S consisten nun compoñente mecánico e un compoñente electrónico. O compoñente electrónico dun dispositivo chámase **controlador do dispositivo** ou **adaptador**. Nos PCs, comunmente teñen a forma dun chip na tarxeta principal ou unha tarxeta de circuíto integrado.

- **E/S por asignación de memoria**

  Cada controlador ten uns cantos rexistros que se utiliza para comunicarse coa CPU. Ao escribir neles, o SO pode facer que o dispositivo envíe ou acepte datos, se acenda ou se apague, ou realice calquera outra acción.
  Ademais dos rexistros de control, moitos dispositivos teñen un *buffer* de datos que o SO pode ler e escribir.
  De todo isto xorde a cuestión acerca de como se comunica a CPU cós rexistros de control e os *búferes* de datos dos dispositivos. Existen dúas alternativas:

  - O primeiro método, a cada rexistro de control se lle asigna un número de **porto de E/S**, un enteiro de 8 ou 16 bits. O conxunto de todos os portos de E/S forma o **espazo de portos de E/S** e está protexido de maneira que os programas de usuario ordinarios non poidan utilizalos (só o SO).
  - O segundo método é asignar todos os rexistros de control ao espazo de memoria. A cada rexistro de control se lle asigna unha dirección de memoria única á cal non hai memoria asignada. Este sistema coñécese como **E/S con asignación de memoria** (*mapped-memory*). Polo xeral, as direccións asignadas atópanse na parte superior do espazo de direccións.

- **Acceso directo a memoria (DMA)**

  Sen importar que unha CPU teña ou non E/S por asignación de memoria, necesita direccionar os controladores de dispositivos para intercambiar datos con eles. A CPU pode solicitar datos dun controlador de E/S un bit á vez, pero ao facelo desperdíciase o tempo da CPU, polo que a miúdo se utiliza un esquema distinto, coñecido como *DMA (Acceso Directo a Memoria)*. Cando se traballa en modo DMA a transferencia dos datos non é procesada polo micro senón por un procesador específico chamado "controlador DMA". O SO só pode utilizar DMA se o *hardware* ten un controlador DMA.

- **Interrupcións**

Cando un dispositivo de E/S rematou un traballo que se lle asignou, produce unha interrupción. Para iso, impón un sinal nunha liña de bus que se lle asignara. Esta sinal é detectada polo chip controlador de interrupcións na tarxeta principal, que despois decide o que debe facer. O sinal de interrupción fai que a CPU deixe o que está facendo e empece a facer outra cousa. O número nas liñas de dirección utilízase como índice nunha táboa chamada *vector de interrupcións* para obter un novo contador do programa, que apunta ao inicio do procedemento de servizo de interrupcións correspondente.

Fundamentos do *software* de E/S

Dende o punto de vista do SO pódese realizar a E/S das seguintes formas:

- **Obxectivos do *software* de E/S**

Un concepto clave no deseño do *software* de E/S coñécese como **independencia de dispositivos**. O que significa é que debe ser posible escribir programas que poidan acceder a calquera dispositivo de E/S sen ter que especificar o dispositivo por adiantado.
Un obxectivo moi relacionado coa independencia dos dispositivos é a **denominación uniforme**. O nome do arquivo ou dispositivo simplemente debe ser unha cadea ou un enteiro sen depender do dispositivo de ningunha forma.
Outra cuestión importante relacionada có *software* de E/S é o **manexo de erros**.
Outra cuestión clave é a das transferencias **síncronas** (de bloqueo) contra as **asíncronas** (controladas por interrupcións). A maioría das operacións de E/S son asíncronas: a CPU inicia a transferencia e vaise facer algo mais ata que chega a interrupción. Os programas de usuario son moito mais fáciles de escribir se as operacións de E/S son de bloqueo: despois dunha chamada ao sistema *read*, o programa se suspende de maneira automática ata que existan datos dispoñibles no *buffer*. Así que, o SO fai que as operacións que en realidade son controladas por interrupcións parezan de bloqueo para os programas de usuario.
O uso de ***buffer***.
Comparación entre os dispositivos compartidos e os dispositivos dedicados. Os discos duros, en xeral poden ser utilizados por moitos usuarios á vez, en cambio, outros dispositivos como as cintas só poden ser utilizados por un usuario de cada vez.
Hai tres maneiras fundamentalmente distintas nas que se poden levar a cabo a E/S:

- **E/S programada.** Neste caso, todo o traballo o fai a CPU.
- **E/S controlada por interrupcións.** O xeito de permitir que a CPU faga algo mentres espera a que un dispositivo estea listo dende que se lle mandou facer algo, é utilizando interrupcións. Así e todo se, por exemplo, falamos dun dispositivo de caracteres, o paso de cada carácter produce unha interrupción.
- **E/S mediante o uso de DMA.** Un gran problema na E/S controlada por interrupcións é que ocorre unha interrupción en cada carácter. Unha solución é utilizar DMA. Aquí a idea é permitir que o controlador de DMA se encargue de enviar todos os caracteres e logo avise á CPU de que a transferencia foi realizada.

Xestión de arquivos e directorios

Os obxectivos fundamentais do servidor de arquivos son los seguintes:

- Facilitar o manexo dos dispositivos periféricos. Para iso ofrece unha visión lóxica simplificada dos mesmos en forma de arquivos.
- Protexer aos usuarios, poñendo limitacións aos arquivos que é capaz de manipular cada usuario.

Os servizos que se engloban no servidor de arquivos son de dous tipos:

- Os servizos dirixidos ao manexo de datos.
- Os dirixidos ao manexo dos nomes, ou directorios.

O servidor de arquivos ofrece ao usuario unha visión lóxica composta por unha serie de obxectos (arquivos e directorios) identificables por un nome lóxico sobre os que pode realizar unha serie de operacións. A visión física debe incluír os detalles de como están almacenados estes obxectos nos periféricos correspondentes.
Os Sistema de arquivos (FS, *File System*) son o conxunto de arquivos incluídos nunha unidade de disco. O sistema de arquivos está composto polos datos dos arquivos, así como por toda a información auxiliar que se require (*boot*, metainformación...).
Os SOs soportan varios tipos de arquivos:

- Os arquivos regulares son os que conteñen información do usuario.
- Os directorios son arquivos especiais empregados para manter a estrutura do sistema de arquivos.
- Os arquivos especiais de caracteres relaciónanse coa entrada/saída e se utilizan para modelar dispositivos de E/S de caracteres.
- Os arquivos especiais de bloques se utilizan para modelar discos.

Historia dos Sistemas Operativos

O SO o forman un conxunto de programas que axudan aos usuarios na explotación dunha computadora, simplificando, por un lado, o seu uso, e permitindo, por outro, obter un bo rendemento da máquina.

É difícil tratar de dar unha definición precisa de SO, posto que existen moitos tipos, segundo sexa a aplicación desexada, o tamaño da computadora usada e o traballo que se vaia a facer con ela. Por iso imos realizar unha evolución histórica o mais completa posible dos SOs, xa que así quedará plasmada a finalidade que se lles foi dando.

Pódense atopar as seguintes etapas no desenrolo dos SOs, que coinciden coas catro xeracións das computadoras.

Prehistoria

Durante esta etapa, que cobre os anos corenta, construíronse as primeiras computadoras. Como exemplo de computadoras desta época pódese citar o **ENIAC (*Electronic Numerical Integrator Analyzer and Computer*)**, financiado polo Laboratorio de Investigación Balística dos Estados Unidos. O ENIAC era unha máquina enorme cun peso de 30 toneladas, que era capaz de realizar 5.000 sumas por segundo, 457 multiplicacións por segundo e 38 divisións por segundo.

Outra computadora desta época foi o **EDVAC (*Electronic Discrete Variable Automatic Computer*)**. Nesta etapa **non existían SOs**. O usuario debía codificar o seu programa a man e en instrucións máquina, e debía introducilo persoalmente na computadora, mediante conmutadores ou tarxetas perforadas. As saídas se imprimían ou se perforaban en cinta de papel para a súa posterior impresión. En caso de erros na execución dos programas, o usuario tiña que depuralos examinando o contido da memoria e os rexistros da computadora.

Nesta primeira etapa todos os traballos se realizaban en serie. Introducíase un programa na computadora, executábase e imprimíanse os resultados e se repetía este proceso con outros programas. Outro aspecto importante desta época é que se requiría moito tempo para preparar e executar un programa, xa que o programador debía encargarse de codificar todo o programa e introducilo na computadora de forma manual.

Primeira xeración (anos cincuenta)

Coa aparición da primeira xeración de computadoras (anos cincuenta) faise necesario racionalizar a súa explotación, posto que xa comeza a existir unha base maior de usuarios. O tipo de operación seguía sendo serie como no caso anterior, isto é, tratábase un traballo detrás de outro, tendo cada traballo as fases seguintes:

- Instalación de cintas ou fichas perforadas nos dispositivos periféricos. No seu caso, instalación do papel na impresora.
- Lectura mediante un programa cargador do programa a executar e dos seus datos.
- Execución do programa.
- Impresión ou gravación dos resultados.
- Retirada de cintas, fichas e papel.

A realización da primeira fase denominábase "montar o traballo".

O problema básico que abordaban estes SOs primitivos era optimizar o fluxo de traballos, minimizando o tempo empregado en retirar un traballo e montar o seguinte. Tamén empezaron a abordar o problema da E/S, facilitando ao usuario paquetes de rutinas de E/S, para simplificar a programación destas operacións, aparecendo así os primeiros manexadores de dispositivos.

Introduciuse tamén o concepto de **system file name**, que empregaba un nome ou número simbólico para referirse aos periféricos, facendo que a súa manipulación fora moito mais flexible que mediante as direccións físicas.

Para minimizar o tempo de montaxe dos traballos, estes agrupábanse en **lotes (*batch*)** do mesmo tipo (por exemplo: programas Fortran, programas Cobol, etc.), o que evitaba ter que montar e desmontar as cintas dos compiladores e montadores, aumentando o rendemento.

Nas grandes instalacións utilizábanse computadoras auxiliares, ou satélites, para realizar as funcións de montar e retirar os traballos. Así se melloraba o rendemento da computadora principal, posto que se lle subministraban os traballos montados en cinta magnética e este se limitaba a procesalos e a gravar os resultados tamén en cinta magnética. Neste caso dicíase que a E/S facíase **fora de liña (*off-line*)**.

Os SOs das grandes instalacións tiñan as seguintes características:

- Procesaban un único fluxo de traballos en lotes.
- Dispoñían dun conxunto de rutinas de E/S.
- Usaban mecanismos rápidos para pasar dun traballo ao seguinte.

- Permitían a recuperación do sistema se un traballo acababa en erro.
- Tiñan unha linguaxe de control de traballos que permitía especificar os recursos a utilizar e as operacións a realizar por cada traballo.

Como exemplos de SOs desta época pódense citar o **FMS (*Fortran Monitor System*)** e **IBYSS**, o SO da IBM 7094.

Segunda xeración (anos sesenta)

Coa aparición da Segunda Xeración de computadoras (principios dos sesenta) fíxose mais necesario, dada a maior competencia entre os fabricantes, mellorar a explotación destas máquinas de altos prezos. A **multiprogramación** impúxose en sistemas de lotes como unha forma de aproveitar o tempo empregado nas operacións de E/S. A base destes sistemas reside na gran diferenza que existe entre as velocidades dos periféricos e da UCP, polo que esta última, nas operacións de E/S, pásase moito tempo esperando aos periféricos. Unha forma de aproveitar ese tempo consiste en manter varios traballos simultaneamente en memoria principal (técnica chamada de multiprogramación), e en realizar as operacións de E/S por acceso directo a memoria (DMA). Cando un traballo necesita unha operación de E/S a solicita ao SO que se encarga de:

- Conxelar o traballo solicitante.
- Iniciar a mencionada operación de E/S por DMA.
- Pasar a realizar outro traballo residente en memoria.

Estas operacións as realiza o SO multiprogramado de forma transparente ao usuario.

Tamén nesta época aparecen outros modos de funcionamento moi importantes:

- Constrúense os primeiros **multiprocesadores**, nos que varios procesadores forman unha soa máquina de maiores prestacións.
- Introdúcese o concepto de **independencia de dispositivo**. O usuario xa non ten que referirse nos seus programas a unha unidade de cinta magnética ou a unha impresora en concreto. Limítase a especificar que quere gravar un arquivo determinado ou imprimir uns resultados. O SO encárgase de asignarlle, de forma dinámica, unha unidade dispoñible, e de indicar ao operador do sistema a unidade seleccionada, para que este monte a cinta ou o papel correspondente.
- Comezan os Sistemas de **tempo compartido ou *timesharing***. Estes sistemas, aos que estamos moi acostumados na actualidade, permiten que varios usuarios traballen de forma **interactiva ou conversacional** coa computadora desde terminais, que en aqueles días eran teletipos electromecánicos. O SO encárgase de repartir o tempo da UCP entre os distintos usuarios, asignando de forma rotativa pequenos intervalos de tempo de UCP denominadas "rodaxas" (*time slice*). En sistemas ben dimensionados, cada usuario ten a impresión de que a computadora lle atende exclusivamente a el, respondendo rapidamente as súas ordes. Aparecen así os primeiros planificadores.
- Aparecen, nesta época, os primeiros sistemas de **tempo real**. Trátase de aplicacións militares, en concreto para detección de ataques aéreos. Neste caso, a computadora está conectada a un sistema externo e debe responder velozmente ás necesidades dese sistema externo. Neste tipo de sistemas as respostas deben producirse en períodos de tempo previamente especificados, que na maioría dos casos son pequenos. Os primeiros sistemas deste tipo construíanse en ensamblador e se executaban sobre máquina espida, o que facía destas aplicacións sistemas moi complexos.

Durante esta época desenrolábanse, entre outros, os seguintes SOs: o **CTSS** [1962], desenrolado no **MIT** e que foi o primeiro sistema de tempo compartido. Este SO utilizouse nun IBM 7090 e chegou a manexar ata 32 usuarios interactivos. O **OS/360** [1966], SO utilizado nas máquinas da liña 360 de IBM; e o sistema **MULTICS** [1972], desenrolado no MIT con participación dos laboratorios Bell e que evolucionou posteriormente para converterse no SO UNIX. MULTICS foi deseñado para dar soporte a centos de usuarios; con todo, aínda que unha versión primitiva deste SO executouse en 1969 unha computadora GE 645, non proporcionou os servizos para os que foi deseñada e os laboratorios Bell finalizaron a súa participación no proxecto.

Terceira xeración (anos setenta)

A terceira xeración é a época dos sistemas de propósito xeral e se caracteriza polos SOs multimodo de operación, isto é, capaces de operar en lotes, en multiprogramación, en tempo real, en tempo compartido e en modo multiprocesador. Estes SOs foron moi custosos de realizar e interpuxeron entre o usuario e o *hardware* unha grosa capa *software*, de forma que este só vía esta capa, sen terse que preocupar dos detalles da circuitería.

Un dos inconvenientes destes sistemas operativos era a súa complexa linguaxe de control, que debían aprenderse os usuarios para preparar os seus traballos, posto que era necesario especificar multitude de detalles e opcións. Outro dos inconvenientes era o gran consumo de recursos que ocasionaban, isto é, os grandes espazos de memoria principal e secundaria ocupados, así como o tempo de UCP consumido, que nalgúns casos superaba o 50 por 100 do tempo total.

Esta década foi importante pola aparición de dous sistemas que tiveron unha grande difusión, UNIX [1986] e MVS [1990] de IBM. De especial importancia foi UNIX, desenvolvido nos laboratorios Bell para unha PDP-7 en 1970. Pronto se transportou a unha PDP-11, para o que se reescribiu utilizando a linguaxe de programación C. Isto foi algo moi importante na historia dos SOs, xa que ata a data ningún se escribira utilizando unha linguaxe de alto nivel, recorrendo para iso ás linguaxes ensambladoras propios de cada arquitectura. Só unha pequena parte de UNIX, aquela que accedía de

forma directa ao *hardware*, seguiu escribíndose en ensamblador. A **programación dun SO utilizando unha linguaxe de alto nivel** como é C fai que un SO sexa facilmente transportable a unha ampla gama de computadoras. Na actualidade, practicamente todos os SOs escríbense en linguaxes de alto nivel, fundamentalmente en C.

A primeira versión amplamente dispoñible de UNIX foi a versión 6 dos laboratorios Bell, que apareceu en 1976. A esta seguiulle a versión 7 distribuída en 1978, antecesora de practicamente todas as versións modernas de UNIX. En 1982 aparece unha versión de UNIX desenvolvida pola Universidade de California en Berkeley, a cal se distribuíu como a versión BSD (*Berkeley Software Distribution*) de UNIX. Esta versión de UNIX introduciu melloras importantes como a inclusión de memoria virtual e a interface de *sockets* para a programación de aplicacións sobre protocolos TCP/IP.

Máis tarde, AT&T (propietario dos laboratorios Bell) distribuíu a versión de UNIX coñecida como **System V** ou **RVS4**. Desde entón moitos foron os fabricantes de computadoras que adoptaron a UNIX como sistema operativo das súas máquinas. Exemplos destas versións son: Solaris de SUN, HP-UX de HP, IRIX de SGI e AIX de IBM.

Cuarta xeración (anos oitenta ata a actualidade)

A cuarta xeración caracterizase por unha evolución dos SOs de propósito xeral da terceira xeración, tendente á súa especialización, á súa simplificación e a dar máis importancia á produtividade do usuario que ao rendemento da máquina.

Adquire cada vez máis importancia o tema das redes de computadoras, tanto redes de longo alcance como locais. En concreto, a diminución do custo do *hardware* fai que se difunda o **proceso distribuído**, en contra da tendencia centralizadora anterior. O proceso distribuído consiste en dispor de varias computadoras, cada unha situada en o lugar de traballo das persoas que a empregan, en lugar de unha única central. Estas computadoras adoitan estar unidas mediante unha rede, de forma que poidan compartir información e periféricos.

Difúndese o concepto de **máquina virtual**, consistente en que unha computadora X, incluíndo o seu SO, sexa simulada por outra computadora Y. A súa vantaxe é que permite executar, na computadora Y, programas preparados para a computadora X, o que posibilita o emprego de *software* elaborado para a computadora X, sen necesidade de dispor de devandita computadora.

Durante esta época, os sistemas de **bases de datos** substitúen aos arquivos en multitude de aplicacións. Estes sistemas diferéncianse dun conxunto de arquivos en que os seus datos están estruturados de tal forma que permiten acceder á información de diversos xeitos, evitar datos redundantes e manter a súa integridade e coherencia.

A difusión das computadoras persoais trouxo unha humanización nos sistemas informáticos. Aparecen os sistemas "amigables" ou ergonómicos, nos que se evita que o usuario teña que aprenderse complexas linguaxes de control, substituíndose estes por os sistemas dirixidos por menú, nos que a selección pode ata facerse mediante un manexador de cursor. Nestes sistemas, de orientación monousuario, o obxectivo primario do sistema operativo xa non é aumentar o rendemento do sistema, senón a produtividade do usuario. Neste sentido, a tendencia actual consiste en utilizar sistemas operativos multiprogramados sobre os que se engade un xestor de fiestras, o que permite que o usuario teña activas, en cada momento, tantas tarefas como desexe e que as distribúa ao seu antollo sobre a superficie do terminal.

Os sistemas operativos que dominaron o campo das computadoras persoais foron UNIX, **MS-DOS** e os sucesores de MICROSOFT para este sistema: **WINDOWS 95**/**98**, **WINDOWS NT**, **WINDOWS 2000** e **WINDOWS XP**. A primeira versión de **WINDOWS NT** (versión 3.1) apareceu en 1993 e incluía a mesma interfaz de usuario que WINDOWS 3.1. En 1996 aparece a versión 4.0, que se caracterizou pola inclusión dentro do executivo de WINDOWS NT de diversos compoñentes gráficos que executaban anteriormente en modo usuario. Durante o ano 2000, MICROSOFT distribúe a versión denominada WINDOWS 2000, en o ano 2001 comeza a distribuír **Windows XP** e no 2007 lanza **Windows Vista** substituído xa no ano 2009 por **Windows 7**.

Tamén tivo importancia durante esta época o desenvolvemento de LINUX. LINUX é un SO similar a UNIX, desenvolvido de forma desinteresada durante a década dos noventa por miles de voluntarios conectados a INTERNET. LINUX está crecendo fortemente debido sobre todo a o seu baixo custo e a a súa gran estabilidade, comparable a calquera outro sistema UNIX. Unha das principais características de LINUX é que o seu código fonte está dispoñible, o que lle fai especialmente atractivo para o estudo de a estrutura interna de un sistema operativo. A súa aparición tivo tamén moita importancia en o mercado do software xa que fixo que se difunda o concepto de software libre.

Durante esta etapa desenvólvense tamén os **sistemas operativos de tempo real,** encargados de ofrecer servizos especializados para o desenvolvemento de aplicacións de tempo real. Algúns exemplos son: **QNX** [1997], **RTEMS** e **VRTX** [1986].

A mediados dos oitenta aparecen os **sistemas operativos distribuídos.** Un sistema operativo distribuído é un sistema operativo común utilizado nunha serie de computadoras conectadas por unha rede. Este tipo de sistemas aparece ao usuario como un único sistema operativo centralizado, facendo xa que logo máis fácil o uso dunha rede de computadoras. Un sistema operativo deste tipo segue tendo as mesmas características que un sistema operativo convencional pero aplicadas a un sistema distribuído Como exemplo de sistemas operativos distribuídos pódese citar: **Mach** [1986], **Chorus** [1988] e **Amoeba** [1990].

Os sistemas operativos distribuídos deixaron de ter importancia e evolucionaron durante a década dos noventa ao que se coñece como ***middleware***. Un *middleware* é unha capa de *software* que se executa sobre un SO xa existente e que se encarga de xestionar un sistema distribuído. Neste sentido, presenta as mesmas funciones que un SO distribuído. A diferenza é que executa sobre SOs xa existentes que poden ser ademais distintos, o que fai máis atractiva a súa utilización. Dous dos *middleware* máis importantes desta década foron **DCE** [1992] e **CORBA** [1996]. MICROSOFT tamén ofrece o seu propio *middleware* coñecido como **DCOM** [1999].

En canto ás **interfaces de programación,** durante esta etapa ten importancia o desenvolvemento do estándar POSIX. Este estándar persegue que as distintas aplicacións que fagan uso dos servizos dun sistema operativo sexan portables sen ningunha dificultade a distintas plataformas con SOs diferentes. Cada vez é maior o número de sistemas operativos que ofrecen esta interfaz.

Outra das interfaces de programación máis utilizada é a interfaz Win32, interface dos SOs WINDOWS. No futuro próximo, a evolución dos sistemas operativos vaise a orientar cara ás plataformas distribuídas e a computación móbil. Gran importancia terá a construción de sistemas operativos e contornas que permitan utilizar estacións de traballo heteroxéneas (computadoras de diferentes fabricantes con SOs distintos) conectadas por redes de interconexión, como unha gran máquina centralizada, o que permitirá dispor dunha maior capacidade de cómputo e facilitará o traballo cooperativo.

Características básicas dos Sistemas Operativos máis coñecidos

UNIX

Todos os sistemas UNIX teñen o seu reloxo interno que conta os segundos desde o 31 de Decembro de 1970 ás 24:00 horas, que oficialmente se considera o nacemento de UNIX. Foi deseñado nos laboratorios de AT&T BELL, realmente para xogar a un xogo espacial. Orixinalmente corría en un PDP-11, que apenas tiña 16 KB de memoria. Hoxe en día podemos atopar versións de UNIX-Linux correndo en modestos 286 e en grades ordenadores como o "Finisterrae" do CESGA.

UNIX, como Windows 2000, utiliza a **Multitarefa preemptiva,** e é amplamente usado en dispositivos RT (*Real Time*), é dicir, ten que traballar en tempo real.

No referente ao **Multifío,** en UNIX, por exemplo, un proceso pode ter varios segmentos de código executándose en paralelo.

UNIX tamén traballa sobre un equipo con tecnoloxía de **Multiproceso simétrico (SMP),** e ata, do mesmo xeito que en Windows 2000, pode asignar un procesador concreto a unha aplicación.

A estrutura interna de UNIX é distinta dependendo da distribución, hainos de *kernel* monolítico ou micro-*kernel* (como MacOS X).

UNIX pode traballar en modo *cluster* de 1000 e 2000 máquinas, sen ningún problema.

Os programas de usuario pódense executar a calquera prioridade, e existe o usuario "root" que pode botar abaixo o sistema en calquera momento. Non hai limitacións para ese usuario, suponse que sabe o que fai. Tamén pode asignar prioridades altas aos procesos de usuario. En resumo, UNIX confía totalmente no seu "administrador".

As implementacións máis importantes de UNIX son:

- ◊ **Solaris de Sun Microsystems.** Un dos sistemas operativos UNIX máis difundido na contorna empresarial e coñecido pola súa gran estabilidade. Parte do código fonte de Solaris liberouse con licenza de fontes abertas (OpenSolaris).
- ◊ **AIX de IBM.** O UNIX "propietario" de IBM cumpriu 20 anos de vida no 2006 e continúa en pleno desenvolvemento, cunha perceptible herdanza do *mainframe* en campos como a virtualización ou a NIVEL dos servidores, herdada dos seus "irmáns maiores".
- ◊ **HP-UX de Hewlett-Packard.** Este SO tamén naceu ligado ás computadoras departamentais deste fabricante. Tamén é un SO estable que continua en desenvolvemento.
- ◊ **Mac OS X.** Curiosamente os seus propios usuarios adoitan descoñecer que se trata dun UNIX completo, aprobado por The Open Group. A súa diferenza marcada é que posúe unha interface gráfica propietaria chamada **Aqua.**

Linux

Linux é un termo xenérico para referirse a sistemas operativos similares a Unix baseados no núcleo de Linux. O seu desenvolvemento é un dos exemplos máis prominentes de *software* libre; normalmente todo o código fonte pode ser utilizado, modificado e redistribuído libremente por calquera baixo os termos da **Licenza Pública Xeneral de GNU (GNU GPL)** e outras licenzas libres.

As variantes destes sistemas denomínanse distribucións de Linux e o seu obxectivo é ofrecer unha edición que cumpra con as necesidades de determinado grupo de usuarios. Existen edicións especialmente deseñadas para o seu uso en servidores e supercomputadores, como **CentOS** e

Debian, e outras para computadores de escritorio como **Fedora** e **Ubuntu**.

Windows

Windows 2000

Evolución de Windows NT 4.0, que aproveita a experiencia de Microsoft nas gamas de servidores.

Vexamos as súas principais características:

- **Sistema Operativo con Multitarefa preemptiva:** cada aplicación dispón do procesador durante un lapso de tempo predeterminado ou ata que outra aplicación teña unha prioridade superior á aplicación en curso. A planificación (*scheduling*), a atribución do tempo de procesador para a aplicación en curso realízaa o SO sen consultar ás aplicacións executadas. En consecuencia, se unha aplicación se bloquea, perde a atribución do procesador inicialmente prevista e é deixada de lado, sen bloquear nin o sistema, nin as demais aplicacións. A repartición do procesador e dos recursos (porto de impresora, teclado...) entre as aplicacións realízao o SO.

Multitarefa cooperativa

Neste caso cada aplicación que se executa sobre o SO dispón do procesador (e dos recursos asociados) e a continuación libérao, permitindo deste xeito á aplicación que segue na cola de espera de execución, dispor á súa vez do procesador. Se unha tarefa se bloquea todas as aplicacións quedan en espera ata que se desbloquea o SO.

- **Multifío:** Significa que no interior dunha mesma aplicación, varias tarefas poden executarse en paralelo. Por exemplo: cando se teclea un texto en Word, este mostra os caracteres cun deseño de páxina desexado e corrixe á vez os erros ortográficos...
- **Multiproceso simétrico (SMP):** o SO, e as aplicacións que corren sobre el, ven as súas unidades de execución sobre os distintos procesadores. O sistema dispón sempre dunha porcentaxe de tempo do procesador, sexa o que sexa. Pódese, tamén, vincular un proceso concreto a un procesador en particular utilizando o administrador de tarefas.
- Windows 2000 é un composto de sistemas operativos en capas e de sistemas cliente/servidor fundado en micro-kernel. A combinación destas dúas tecnoloxías permite distinguir dúas partes en Windows 2000 chamadas modo de núcleo e modo de usuario:

a. Modo de núcleo.

O **Executive** ou executor agrupa o conxunto de compoñentes do sistema que se executan en modo de núcleo. Estes compoñentes son prioritarios na utilización do procesador. O núcleo ten un lugar predominante xa que se encarga de proporcionar memoria ás aplicacións, de elixir os procesos que se executarán en un instante preciso e de comunicar con os periféricos. As aplicacións dependen do núcleo para todas as súas necesidades, o que evita que entren en contacto directo cos periféricos e provoquen un fallo do sistema.
O núcleo de Windows 2000 provén do núcleo de Windows NT 4.0 con varias melloras, como a utilización de varias sesións de usuario na mesma máquina, a incorporación de cotas de procesador para as necesidades de IIS 5.
Ao contrario que o núcleo de Windows NT 4.0, o de W2000 incorpora a arquitectura **WDM (*Windows Drivers Model*).** Trátase dun novo controlador que permite usar os mesmos controladores en W98 e en W2000.
W2000 incorpora a arquitectura **EMA (*Entreprise Memory Architecture*)** que permite ás aplicacións utilizar ata 32 GB de memoria.
As funcionalidades de W2000 permítenlle funcionar en *cluster* (dúas máquinas poden relevarse en caso de problema).
Para terminar sobre o modo de núcleo, subliñemos que foi deseñado para soportar calquera aplicación sexa cal for a lingua utilizada. Para iso W2000 utiliza UNICODE.

b. Modo de usuario.

En oposición, o modo de usuario agrupa os subsistemas protexidos sobre os que se apoian as aplicacións do usuario. Os procesos en modo usuario non teñen acceso directo ao hardware; están limitados a unha zona de memoria asignada e execútanse con un nivel de prioridade baixo. Unha das grandes evolucións do modo usuario en Windows 2000 é a presenza no subsistema da seguridade de ***Active Directory***.

Windows XP

Windows XP (cuxo nomee en clave inicial foi **Whistler**) é unha liña de sistemas operativos desenvolvido por Microsoft sacados a finais de 2001. Considérase que están no mercado mais 400 millóns de copias funcionando. As letras "XP" proveñen da palabra **eXPeriencia**.
É o sucesor de Windows 2000 e Windows ME e antecesor de Windows Vista; é o primeiro SO de Microsoft orientado ao consumidor que se constrúe con un núcleo e arquitectura de Windows NT e que se atopa dispoñible en versións para PC de 32 e 64 bits.

Windows XP está construído no código de Windows 2000 cunha nova interface gráfica que inclúe características lixeiramente redeseñadas, algunhas das cales aseméllanse á contorna de escritorio presente en Mac VOS X.
Cada certo tempo, Microsoft distribúe uns paquetes denominados ***Service Packs*** **(Paquetes de servizo),** no que están todas as actualizacións á data, ademais dalgúns novas aplicacións cos que aseguran un SO seguro. Para Windows XP lanzáronse os seguintes:

- ◊ ***Service Pack*** **1:**

  O SP1 para Windows XP foi lanzado o 9 de novembro de 2002. A novidade máis visible foi a incorporación de a utilidade Configurar acceso e programas predeterminados, para poder elixir de forma máis sinxela que programas deséxase utilizar para as tarefas máis comúns. Outras novidades que introduciu foron o soporte para USB 2.0 e de LBA de 48 bits, polo que Windows XP podería soportar discos duros de máis de 137 GB.
- ◊ ***Service Pack*** **2:**

  O 6 de agosto de 2004, Microsoft lanzou o SP2, que incluía todas as correccións atopadas no SP1, ademais de varias novidades, centradas sobre todo en dar maior seguridade ao sistema operativo.
- ◊ ***Service Pack*** **3:**

  Windows XP Service Pack 3 (SP3) foi lanzado para fabricantes o 21 de abril de 2008, e ao público en xeral, a través do Centro de descargas de Microsoft e Windows Update, o 6 de maio de 2008. SP3 contén novas características: actualizacións independentes de Windows XP e características tomadas de Windows Vista.
  O soporte para o SP3 finalizará en abril de 2014.

No referente á **Seguridade,** Windows XP foi criticado por a súa susceptibilidad a malware, como virus, troyanos ou vermes. As opcións de seguridade por defecto crean unha conta do administrador que proporciona o acceso sen restrición a todo o sistema, incluíndo os puntos vulnerables.
Windows, cunha cota de mercado grande, foi tradicionalmente un branco para os creadores de virus. Os buracos de a seguridade son a miúdo invisibles ata que se explotan, facendo a súa prevención un feito difícil.

Windows Vista

Windows Vista é unha liña de SOs desenvolvida por Microsoft para ser usada en computadoras de escritorio, portátiles, Tablet PC e centros multimedia. Antes de ser anunciado oficialmente o 22 de xullo de 2003 o seu nome en código foi **Longhorn**.
O 30 de xaneiro de 2007 foi lanzado mundialmente e foi posto a disposición para ser comprado e descargado desde o sitio web de Microsoft. A aparición de Windows Vista vén máis de 5 anos logo de a introdución do seu predecesor, Windows XP, é dicir o tempo máis longo entre dúas versións consecutivas de Microsoft Windows.
Algunhas das melloras de Windows Vista son:

- ◊ **Windows Aero:** A nova interface gráfica incluída en Windows Vista que substitúe á Interfaz gráfica utilizada en Windows XP.
- ◊ **Internet Explorer 7:** Ven con Windows Vista (tamén se pode descargar unha versión para Windows XP SP2) a cal incorpora varias melloras como a navegación con pestanas e a vista Quick Tabs que mostras vistas en miniatura das páxinas abertas. Tamén inclúe algunhas melloras de seguridade como as advertencias antiphishing e o modo protexido (só en Vista) que evita que os sitios web executen código sen permiso do usuario.
- ◊ **Windows Sidebar:** (Barra lateral de Windows) é unha nova ferramenta a cal sitúase en o costado dereito de a pantalla e en a cal hai pequenos programas ou *Gadgets* os cales permiten ter acceso a pequenas ferramentas sen necesidade de abrir unha fiestra
- ◊ **Windows Media Player 11**
- ◊ Windows Vista é o primeiro SO de Microsoft concibido para garantir unha compatibilidade total con **EFI (*Extensible Firmware Interface*)**, a tecnoloxía chamada a substituír ás arcaicas BIOS e, polo tanto, non empregará **MBR (*Master Boot Record*)**, senón **GPT (*GUID Partition Table*)** nos discos duros.
- ◊ **Fiestras debuxadas con gráficos vectoriais usando** XAML e DirectX. Para iso, utilizaríase unha **nova API, chamada *Windows Presentation Foundation*,** cuxo nomee en código é **Avalon**, que require un cartón gráfico con aceleración 3D compatible con DirectX.
- ◊ **WinFX,** unha API orientada a substituír a API actual chamada Win32.
- ◊ **Unha interface de liña de comando denominada Windows *PowerShell*,** que finalmente se ofreceu como unha descarga independente para Windows Vista e Windows XP SP2.
- ◊ **Un sistema antispyware** denominado Windows *Defender*.
- ◊ **Engade ao *firewall* de sistema a capacidade de bloquear conexións** que saen do sistema sen previa autorización.
- ◊ **Windows Mail,** é un cliente de correo electrónico, substituíndo a Outlook Express.

Actualizacións de Windows Vista:

- ◊ ***Service Pack*** **1.** O *Service Pack* 1 (SP1) foi unha actualización xeral que recibiu Windows Vista en o 4 de febreiro do 2008, e ocúpase dos problemas que tivo na versión inicial. SP1 contén os cambios específicos centrados en abordar as cuestións de rendemento, fiabilidade e seguridade, o apoio a novos tipos de *hardware*, mellor administración da memoria, resolver o problema co consumo de enerxía en as baterías, ademais de agregar soporte para varios estándares emerxentes de *hardware* e *software*, entre eles destacando o soporte para sistema de arquivos exFAT, redes sen fíos 802.11n, IPv6 en conexións VPN, e o protocolo SSTP (*Secure Socket Tuneling*). O núcleo do sistema SP1 é acorde coa versión de lanzamento de Windows Server 2008.

◊ ***Service Pack* 2.** Esta actualización actualmente esta en desenvolvemento e seguramente será lanzado ao mercado antes que Windows 7.

Windows Server 2003

Windows Server 2003 é un SO da familia Windows de Microsoft para servidores, saíu ao mercado no ano 2003. Está baseado en tecnoloxía NT e a súa versión do núcleo NT é a mesma que a do SO Windows XP

usado en Workstations.

As súas características máis importantes son:

◊ Sistema de arquivos NTFS:
1.- Cotas de disco
2.- Cifrado (Cipher.exe) e compresión de arquivos e cartafois (Compress.exe).
3.- permite montar dispositivos de almacenamento sobre sistemas de arquivos doutros dispositivos.
◊ Xestión de almacenamento, backups...
◊ *Windows Driver Model*: Implementación básica dous dispositivos mais utilizados.
◊ *ActiveDirectory*: Directorio de organización baseado en LDAP, permite xestionar de forma centralizada a seguridade dunha rede corporativa a nivel local.
◊ Autenticación Kerberos5.
◊ Políticas de seguridade.

Os servidores que manexa Windows 2003 son:

- Servidor de arquivos
- Servidor de impresión
- Servidor de aplicacións
- Servidor de correo (SMTP/POP)
- Servidor de terminal
- Servidor de Redes privadas virtuais (VPN)
- Controlador de Dominios (mediante Active Directory)
- Servidor DNS
- Servidor DHCP
- Servidor de Streaming de Vídeo

Versións de Windows 2003: Actualmente existen catro versións de Windows 2003, aínda que todas elas

contan con versións de 32 e 64 bits (excepto Web Edition). As versións son:

- **Web Edition:** Deseñado para os servizos e a hospedaxe Web.
- **Standard Edition:** O máis versátil de todos, ofrece un gran número de servizos útiles para empresas de calquera tamaño.
- **Enterprise Edition:** Para empresas de maior tamaño que a Standard Edition.
- **Datacenter Edition:** Para empresas que requiran bases de datos máis escalables e un procesamiento de transaccións de gran volume.

Novas versións e actualizacións:

- **Service Pack 1 (SP1).** O 30 de marzo de 2005, Microsoft lanza (Service Pack 1), para todas as versións de Windows 2003. Con o dotan ao Sistema operativo das melloras incluídas en o SP2 de Windows XP, talles como unha nova interface para o *firewall*. O Soporte de Windows Server 2003 Service Pack 1 finalizará ou 14 de Abril de 2009.
- **Windows 2003 R2.** Sae no ano 2006 e trátase dun conxunto de tecnoloxías adicionais que se instalan sobre Windows Server 2003 con Service Pack 1 (SP1).
  Non hai cambios de núcleo respecto de Windows Server 2003 SP1.
  Comparte con Windows Server 2003 SP1 as subseguintes actualizacións de seguridade e Service Packs. Tamén comparte a compatibilidade con aplicacións e ten o mesmo ciclo de vida.
- **Service Pack 2 (SP2).** O 12 de marzo de 2007 lanzouse o Service Pack 2 de Windows Server 2003. Este SP2 está pensado como unha actualización para Windows Server 2003 R2, a a súa úa vez unha actualización do Server 2003 orixinal que Microsoft lanzou en decembro do 2005. No entanto, este Service Pack instálase tanto sobre versións R2 do sistema como sobre a versión orixinal.

Windows Server 2008

Windows Server 2008 foi é o nome do sistema operativo para servidores de Microsoft. É o sucesor de Windows Server 2003 distribuído ao público o 27 de febreiro de 2008. Do mesmo xeito que Windows Vista, Windows Server 2008 baséase no núcleo Windows NT 6.0.
Características: Hai algunhas diferenzas (algunhas sutís e outras non tanto) con respecto a a arquitectura do novo Windows Server 2008, que poden cambiar dramaticamente o xeito en que se usa este sistema operativo. Estes cambios afectan ao xeito en que se xestiona o sistema ata o punto de que se pode chegar a controlar o hardware de forma máis efectiva, pódese controlar moito mellor de forma remota e cambiar de forma radical a política de seguridade.
Entre as melloras que se inclúen, están:

- Novo proceso de reparación de sistemas NTFS: proceso en segundo plano que repara os arquivos danados.
- Creación de sesións de usuario en paralelo: reduce tempos de espera nos Terminal Services e na creación de sesións de usuario a gran escala.
- Peche limpo de Servizos.
- Sistema de arquivos SMB2: de 30 a 40 veces máis rápido o acceso aos servidores multimedia.
- Address Space Loade Randomization (ASLR): protección contra *malware* na carga de controladores en memoria.
- Windows Hardware Erro Architecture (WHEA): protocolo mellorado e estandarizado de reporte de erros.
- Virtualización de Windows Server: melloras en o rendemento de a virtualización.
- PowerShell: inclusión de unha consola mellorada con soporte GUI para administración.
- Server Core: o núcleo do sistema renovouse con moitas e novas melloras.

Windows 7

Windows 7 (anteriormente con nombre código ***Blackcomb***, e logo ***Vienna***) é a versión máis recente de Microsoft Windows, un sistema operativo producido por Microsoft para uso en PCs, incluíndo equipos de escritorio en fogares e oficinas, equipos portátiles, "tablet PC", "netbooks" e equipos "media center".
O desenvolvemento de Windows 7 completouse o 22 de xullo de 2009, sendo entón confirmada a súa data de venda oficial para o 22 de outubro de 2009 xunto ao seu equivalente para servidores Windows Server 2008 R2.
A diferenza do gran salto arquitectónico e de características que sufriu o seu antecesor Windows Vista con respecto a Windows XP, Windows 7 foi concibido como unha actualización incremental e focalizada de Vista e o seu núcleo NT 6.0, o que permitiu o manter certo grao de compatibilidade con aplicacións e hardware nos que este xa era compatible.
Con todo, entre as metas de desenvolvemento para Windows 7 deuse importancia en mellorar a súa interface para volvela máis accesible ao usuario e incluír novas características que permitisen facer tarefas dun xeito máis fácil e rápida, ao mesmo tempo en que se realizarían esforzos para lograr un sistema máis lixeiro, estable e rápido.