**Fase de Desarrollo de software**

**By🡪 Creador**

**>> Idea:> Sistema Operativo**

**>> Portada**

**>>Índice**

**>> Planificación**

**-Estudios**

**Sistema operativo**

Un sistema operativo (SO) es el conjunto de programas de un sistema informático que gestiona los recursos de hardware y provee servicios a los programas de aplicación de software. Estos programas se ejecutan en modo privilegiado respecto de los restantes.

Uno de los propósitos del sistema operativo que gestiona el núcleo intermediario consiste en gestionar los recursos de localización y protección de acceso del hardware, hecho que alivia a los programadores de aplicaciones de tener que tratar con estos detalles. La mayoría de los aparatos electrónicos que utilizan microprocesadores para funcionar, llevan incorporado un sistema operativo (teléfonos móviles, reproductores de DVD, computadoras, enrutadores, etc.). En cuyo caso, son manejados mediante una interfaz gráfica de usuario, un gestor de ventanas o un entorno de escritorio, si es un celular, mediante una consola o control remoto si es un DVD y, mediante una línea de comandos o navegador web si es un enrutador.

El sistema operativo de escritorio dominante es Microsoft Windows con una cuota de mercado de alrededor del 82,74%. macOS de Apple Inc. ocupa el segundo lugar (13,23%), y las variedades de GNU/Linux están colectivamente en tercer lugar (1,57%).2​ En el sector móvil (incluidos teléfonos inteligentes y tabletas), la participación de Android es de hasta un 70% en el año 2017.3​ Las distribuciones Linux son dominantes en los sectores de servidores y supercomputación.4​ Existen otras clases especializadas de sistemas operativos, como los sistemas integrados y en tiempo real, para muchas aplicaciones.

**Funciones principales**

Algunas de las funciones principales de un sistema operativo son las siguientes:

* Gestionar la memoria de acceso aleatorio y ejecutar las aplicaciones, designando los recursos necesarios.
* Administrar la CPU gracias a un algoritmo de programación.
* Direccionar las entradas y salidas de datos (a través de drivers) por medio de los periféricos de entrada o salida.
* Administrar la información para el buen funcionamiento de la PC.
* Dirigir las autorizaciones de uso para los usuarios.
* Administrar los archivos.

Los primeros sistemas (1945-1954) eran grandes máquinas operadas desde la consola maestra por los programadores. Durante la década siguiente (1955-1965) se llevaron a cabo avances en el hardware: lectoras de tarjetas, impresoras, cintas magnéticas, etc. Esto a su vez provocó un avance en el software: compiladores, ensambladores, cargadores, manejadores de dispositivos, etc.

A finales de los años 1980, una computadora Commodore Amiga equipada con una aceleradora Video Toaster era capaz de producir efectos comparados a sistemas dedicados que costaban el triple. Un Video Toaster junto a Lightwave ayudó a producir muchos programas de televisión y películas, entre las que se incluyen Babylon 5, SeaQuest DSV y Terminator.

Problemas de explotación y soluciones iniciales

El problema principal de las primeras computadoras era su baja utilización, la primera solución fue poner un operador profesional que lo manejase, con lo que se eliminaron las hojas de reserva, se ahorró tiempo y se aumentó la velocidad.

Para ello, los trabajos se agrupaban de forma manual en lotes mediante lo que se conoce como procesamiento por lotes (batch processing) sin automatizar.

**Monitores residentes**

Fichas en lenguaje de procesamiento por lotes, con programa y datos, para ejecución secuencial.

Según fue avanzando la complejidad de los programas, fue necesario implementar soluciones que automatizaran la organización de tareas sin necesidad de un operador. Debido a ello se crearon los monitores residentes: programas que residían en memoria y que gestionaban la ejecución de una cola de trabajos.

Un monitor residente estaba compuesto por un cargador, un Intérprete de comandos y un controlador (drivers) para el manejo de entrada/salida.

**Sistemas con almacenamiento temporal de E/S**

Los avances en el hardware crearon el soporte de interrupciones y posteriormente se llevó a cabo un intento de solución más avanzado: solapar la E/S de un trabajo con sus propios cálculos, por lo que se creó el sistema de búfers con el siguiente funcionamiento:

* Un programa escribe su salida en un área de memoria (búfer 1).
* El monitor residente inicia la salida desde el buffer y el programa de aplicación calcula depositando la salida en el buffer 2.
* La salida desde el buffer 1 termina y el nuevo cálculo también.
* Se inicia la salida desde el buffer 2 y otro nuevo cálculo dirige su salida al buffer 1.

**El proceso se puede repetir de nuevo.**

Los problemas surgen si hay muchas más operaciones de cálculo que de E/S (limitado por la CPU) o si por el contrario hay muchas más operaciones de E/S que de cálculo (limitado por la E/S).

**Spoolers**

Hace aparición el disco magnético con lo que surgen nuevas soluciones a los problemas de rendimiento. Se eliminan las cintas magnéticas para el volcado previo de los datos de dispositivos lentos y se sustituyen por discos (un disco puede simular varias cintas). Debido al solapamiento del cálculo de un trabajo con la E/S de otro trabajo se crean tablas en el disco para diferentes tareas, lo que se conoce como Spool (Simultaneous Peripherial Operation On-Line).

**Sistemas operativos multiprogramados**

Surge un nuevo avance: el hardware con protección de memoria, ofreciendo nuevas soluciones a los problemas de rendimiento:

* Se solapa el cálculo de unos trabajos con la entrada/salida de otros trabajos.
* Se pueden mantener en memoria varios programas.

Se asigna el uso de la CPU a los diferentes programas en memoria.

Con los cambios anteriores el monitor residente debe abordar nuevas tareas, naciendo los Sistemas Operativos multiprogramados con las siguientes funciones:

* Administrar la memoria.
* Gestionar el uso de la CPU (planificación).
* Administrar el uso de los dispositivos de E/S.
* Cuando desempeña esas tareas, el monitor residente se transforma en un sistema operativo multiprogramado.

**Llamadas al sistema**

Cada SO implementa un conjunto propio de llamadas al sistema. Ese conjunto de llamadas es la interfaz del SO frente a las aplicaciones. Constituyen el lenguaje que deben usar las aplicaciones para comunicarse con el SO. Por ello si cambiamos de SO, y abrimos un programa diseñado para trabajar sobre el anterior, en general el programa no funcionará, a no ser que el nuevo SO tenga la misma interfaz. Para ello:

* Las llamadas correspondientes deben tener el mismo formato.
* Cada llamada al nuevo SO tiene que dar los mismos resultados que la correspondiente del anterior.

**Modos de ejecución en un CPU**

Las aplicaciones no deben poder usar todas las instrucciones de la CPU. No obstante, el Sistema Operativo, tiene que poder utilizar todo el conjunto de instrucciones del CPU. Por ello, una CPU debe tener (al menos) dos modos de operación diferentes:

* Modo usuario: el CPU podrá ejecutar sólo las instrucciones del juego restringido de las aplicaciones.
* Modo supervisor: la CPU debe poder ejecutar el juego completo de instrucciones.

**Llamadas al sistema**

Una aplicación, normalmente no sabe dónde está situada la rutina de servicio de la llamada. Por lo que si esta se codifica como una llamada de función, cualquier cambio en el S.O. haría que hubiera que reconstruir la aplicación.

Pero lo más importante es que una llamada de función no cambia el modo de ejecución de la CPU. Con lo que hay que conseguir llamar a la rutina de servicio, sin tener que conocer su ubicación, y hacer que se fuerce un cambio de modo de operación de la CPU en la llamada (y la recuperación del modo anterior en el retorno).

Esto se hace utilizando instrucciones máquina diseñadas específicamente para este cometido, distintas de las que se usan para las llamadas de función.

**Bibliotecas de interfaz de llamadas al sistema**

Las llamadas al sistema no siempre tienen una expresión sencilla en los lenguajes de alto nivel, por ello se crean las bibliotecas de interfaz, que son bibliotecas de funciones que pueden usarse para efectuar llamadas al sistema. Las hay para distintos lenguajes de programación.

La aplicación llama a una función de la biblioteca de interfaz (mediante una llamada normal) y esa función es la que realmente hace la llamada al sistema.

**Interrupciones y excepciones**

El SO ocupa una posición intermedia entre los programas de aplicación y el hardware. No se limita a utilizar el hardware a petición de las aplicaciones ya que hay situaciones en las que es el hardware el que necesita que se ejecute código del SO. En tales situaciones el hardware debe poder llamar al sistema, pudiendo deberse estas llamadas a dos condiciones:

* Algún dispositivo de E/S necesita atención.
* Se ha producido una situación de error al intentar ejecutar una instrucción del programa (normalmente de la aplicación).
* En ambos casos, la acción realizada no está ordenada por el programa de aplicación, es decir, no figura en el programa.

Según los dos casos anteriores tenemos las interrupciones y las excepciones:

Interrupción: señal que envía un dispositivo de E/S a la CPU para indicar que la operación de la que se estaba ocupando, ya ha terminado.

Excepción: una situación de error detectada por la CPU mientras ejecutaba una instrucción, que requiere tratamiento por parte del SO.

**Tratamiento de las interrupciones**

Una interrupción se trata en todo caso, después de terminar la ejecución de la instrucción en curso.

El tratamiento depende de cuál sea el dispositivo de E/S que ha causado la interrupción, ante la cual debe poder identificar el dispositivo que la ha causado.

La ventaja de este procedimiento es que no se tiene que perder tiempo ejecutando continuamente rutinas para consultar el estado del periférico. El inconveniente es que el dispositivo debe tener los circuitos electrónicos necesarios para acceder al sistema de interrupciones del computador.

**Importancia de las interrupciones**

El mecanismo de tratamiento de las interrupciones permite al sistema operativo utilizar la CPU en servicio de una aplicación, mientras otra permanece a la espera de que concluya una operación en un dispositivo de E/S.

El hardware se encarga de avisar al SO cuando el dispositivo de E/S ha terminado y el SO puede intervenir entonces, si es conveniente, para hacer que el programa que estaba esperando por el dispositivo, se continúe ejecutando.

En ciertos intervalos de tiempo puede convenir no aceptar señales de interrupción. Por ello las interrupciones pueden inhibirse por programa (aunque esto ellas no deben poder hacerlo).

Un ejemplo de sincronismo por interrupción es el almacenamiento de caracteres introducidos mediante el teclado. Cuando se introduce un carácter, se codifica en el registro de datos del dispositivo y además se activa un bit del registro de estado quien crea una interrupción en el hardware. El procesador deja temporalmente la tarea que estaba completando y ejecuta la rutina de atención a la interrupción correspondiente. El teclado almacena el carácter en el vector de memoria intermedia (también llamado buffer) asociada al teclado y despierta el proceso que había en el estado de espera de la operación de entrada/salida.

**Excepciones**

Cuando la CPU intenta ejecutar una instrucción incorrectamente construida, la unidad de control lanza una excepción para permitir al SO ejecutar el tratamiento adecuado. Al contrario que en una interrupción, la instrucción en curso es abortada. Las excepciones al igual que las interrupciones deben estar identificadas.

**Clases de excepciones**

Las instrucciones de un programa pueden estar mal construidas por diversas razones:

**El código de operación puede ser incorrecto.**

Se intenta realizar alguna operación no definida, como dividir por cero.

La instrucción puede no estar permitida en el modo de ejecución actual.

La dirección de algún operando puede ser incorrecta o se intenta violar alguno de sus permisos de uso.

**Importancia de las excepciones**

El mecanismo de tratamiento de las excepciones es esencial para impedir, junto a los modos de ejecución de la CPU y los mecanismos de protección de la memoria, que las aplicaciones realicen operaciones que no les están permitidas. En cualquier caso, el tratamiento específico de una excepción lo realiza el SO.

Como en el caso de las interrupciones, el hardware se limita a dejar el control al SO, y este es el que trata la situación como convenga.

Es bastante frecuente que el tratamiento de una excepción no retorne al programa que se estaba ejecutando cuando se produjo la excepción, sino que el SO aborte la ejecución de ese programa. Este factor depende de la pericia del programador para controlar la excepción adecuadamente.

**Componentes del Sistema Operativo.**

Todos los componentes de un sistema operativo existen para hacer que las diferentes partes de una computadora trabajen juntas. Todo el software del usuario debe pasar por el sistema operativo para poder utilizar cualquier hardware, ya sea tan simple como un mouse o un teclado o tan complejo como un componente de Internet.

**Núcleo**

En informática, un núcleo o kernel (de la raíz germánica Kern, núcleo, hueso) es un software que constituye una parte fundamental del sistema operativo, y se define como la parte que se ejecuta en modo privilegiado (conocido también como modo núcleo).6​ Es el principal responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Como hay muchos programas y el acceso al hardware es limitado, también se encarga de decidir qué programa podrá usar un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiprogramación. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporcionar una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso al programador.

En algunos sistemas operativos, no existe un núcleo como tal (algo común en sistemas empotrados), debido a que en ciertas arquitecturas no hay distintos modos de ejecución.

**Controladores de dispositivos**

Esquema de un subsistema de controladores de dispositivos como parte del «núcleo del sistema operativo», actuando como interfaz (controlador) entre las “aplicaciones de usuario” y un dispositivo externo (dispositivo).

Un controlador de dispositivo o manejador de dispositivo (en inglés: device controlador, o simplemente driver) nota 1​ es un programa informático que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz (posiblemente estandarizada) para utilizar el dispositivo.​

Es una pieza esencial del software, y en particular, del núcleo de un sistema operativo, sin la cual el hardware sería inutilizable.

**Gestión de procesos**

Un proceso es simplemente, un programa en ejecución que necesita recursos para realizar su tarea: tiempo de CPU, memoria, archivos y dispositivos de E/S. El SO es el responsable de lo siguiente:

* Crear y destruir procesos.
* Parar y reanudar procesos.

La gestión de procesos podría ser similar al trabajo de oficina. Se puede tener una lista de tareas a realizar y a estas fijarles prioridades: alta, media, baja, por ejemplo. Debemos comenzar haciendo las tareas de prioridad alta primero y cuando se terminen seguir con las de prioridad media y después las de baja. Una vez realizada la tarea se tacha.

Esto puede traer un problema que las tareas de baja prioridad pueden que nunca lleguen a ejecutarse y permanezcan en la lista para siempre. Para solucionar esto, se puede asignar alta prioridad a las tareas más antiguas.

**Gestión de la memoria**

La gestión de memoria o administración de memoria es el acto de gestionar la memoria de un dispositivo informático. El proceso de asignación de memoria a los programas que la solicitan.

Los sistemas de memoria virtual separan las direcciones de memoria utilizadas por un proceso de las direcciones físicas reales, permitiendo la separación de procesos e incrementando la cantidad efectiva de memoria de acceso aleatorio utilizando la paginación. La calidad de la gestión de la memoria es crucial para las prestaciones del sistema.

El recolector de basura es la asignación y liberación automática de los recursos de memoria para un programa. La implementación suele ser a nivel del lenguaje de programación en contraposición a la gestión manual de memoria, que asigna y libera los recursos de memoria de una computadora de forma explícita.

La administración de memoria se refiere a los distintos métodos y operaciones que se encargan de obtener la máxima utilidad de la memoria, organizando los procesos y programas que se ejecutan de manera tal que se aproveche de la mejor manera posible el espacio disponible.

Para poder lograrlo, la operación principal que realiza es trasladar la información que deberá ser ejecutada por la unidad central de procesamiento o procesador, a la memoria principal. Actualmente esta administración se conoce como memoria virtual, porque no es la memoria física del procesador sino una memoria virtual que la representa. Entre algunas ventajas, esta memoria permite que el sistema cuente con una memoria más extensa teniendo la misma memoria real, por lo que esta se puede utilizar de manera más eficiente. Y por supuesto, que los programas que son utilizados no ocupen lugar innecesario.

Las técnicas que existen para la carga de programas en la memoria son: partición fija, que es la división de la memoria libre en varias partes (de igual o distinto tamaño) y la partición dinámica, que son las particiones de la memoria en tamaños que pueden ser variables, según la cantidad de memoria que necesita cada proceso.

Entre las principales operaciones que desarrolla la administración de memoria se encuentran la reubicación, que consiste en trasladar procesos activos dentro y fuera de la memoria principal para maximizar la utilización del procesador; la protección, mecanismos que protegen los procesos que se ejecutan de interferencias de otros procesos; uso compartido de códigos y datos, con lo que el mecanismo de protección permite que ciertos procesos de un mismo programa que comparten una tarea tengan memoria en común.

**Gestión del almacenamiento secundario**

Un sistema de almacenamiento secundario es necesario, ya que la memoria principal (almacenamiento primario) es volátil y además muy pequeña para almacenar todos los programas y datos. También es necesario mantener los datos que no convenga mantener en la memoria principal. El SO se encarga de:

* Planificar los discos.
* Gestionar el espacio libre.
* Asignar el almacenamiento.
* Verificar que los datos se guarden en orden.

**Sistema de entrada y salida**

Consiste en un sistema de almacenamiento temporal (caché), una interfaz de manejadores de dispositivos y otra para dispositivos concretos. El sistema operativo debe gestionar el almacenamiento temporal de E/S y servir las interrupciones de los dispositivos de E/S.

**Sistema de archivos**

Un sistema de archivos o sistema de ficheros, (file system: fs.), en informática, es un elemento que controla cómo se almacenan y recuperan los datos. Sin un sistema de archivos, los datos colocados en un medio de almacenamiento serían un gran cuerpo de datos sin manera de saber dónde termina un dato y comienza el siguiente. Es el encargado de administrar y facilitar el uso de las memorias periféricas, ya sean secundarias o terciarias.​

Sus principales funciones son la asignación de espacio a los archivos, la administración del espacio libre y del acceso a los datos resguardados. Estructuran la información guardada en un dispositivo de almacenamiento de datos o unidad de almacenamiento (normalmente un disco duro de una computadora), que luego será representada ya sea textual o gráficamente utilizando un gestor de archivos.

La mayoría de los sistemas operativos manejan su propio sistema de archivos.

Lo habitual es utilizar dispositivos de almacenamiento de datos que permiten el acceso a los datos como una cadena de bloques de un mismo tamaño, a veces llamados sectores, usualmente de 512 bytes de longitud (también denominados clústers). El software del sistema de archivos es responsable de la organización de estos sectores en archivos y directorios y mantiene un registro de qué sectores pertenecen a qué archivos y cuáles no han sido utilizados. En la práctica, un sistema de archivos también puede ser utilizado para acceder a datos generados dinámicamente, como los recibidos a través de una conexión de red de computadoras (sin la intervención de un dispositivo de almacenamiento).

Los sistemas de archivos proveen métodos para crear, mover, renombrar y eliminar tanto archivos como directorios, pero carecen de métodos para crear, por ejemplo, enlaces adicionales a un directorio o archivo (enlace duro en Unix) o renombrar enlaces padres (".." en Unix).

El acceso seguro a sistemas de archivos básicos puede estar basado en los esquemas de lista de control de acceso (access control list, ACL) o capacidades. Las ACL hace décadas que demostraron ser inseguras, por lo que los sistemas operativos experimentales utilizan el acceso por capacidades. Los sistemas operativos comerciales todavía funcionan con listas de control de acceso.

Existen diferentes sistemas de archivos, es decir, existen diferentes formas de organizar la información que se almacena en las memorias (normalmente discos) de los ordenadores. Por ejemplo, existen los sistemas de archivos FAT, FAT32, ext3, ext4, NTFS, XFS, etc.

**Sistemas de protección**

Mecanismo que controla el acceso de los programas o los usuarios a los recursos del sistema. El SO se encarga de:

* Distinguir entre uso autorizado y no autorizado.
* Especificar los controles de seguridad a realizar.
* Forzar el uso de estos mecanismos de protección.

**Sistema de comunicaciones**

Para mantener las comunicaciones con otros sistemas es necesario poder controlar el envío y recepción de información a través de las interfaces de red. También hay que crear y mantener puntos de comunicación que sirvan a las aplicaciones para enviar y recibir información, y crear y mantener conexiones virtuales entre aplicaciones que están ejecutándose localmente y otras que lo hacen remotamente.

**Programas de sistema**

Son aplicaciones de utilidad que se suministran con el SO pero no forman parte de él. Ofrecen un entorno útil para el desarrollo y ejecución de programas, siendo algunas de las tareas que realizan:

* Manipulación y modificación de archivos.
* Información del estado del sistema.
* Soporte a lenguajes de programación.

**Gestor de recursos**

Como gestor de recursos, el sistema operativo administra:

* La unidad central de procesamiento (donde está alojado el microprocesador).
* Los dispositivos de entrada y salida.
* La memoria principal (o de acceso directo).
* Los discos (o memoria secundaria).
* Los procesos (o programas en ejecución).
* En general todos los recursos del sistema.

**Interfaz de usuario**

Esta sección es un extracto de la evolución de las interfaces de usuario:

* CLI (Command line interface): Interfaz de línea de comandos
* GUI (Graphical user interface): Interfaz gráfica de usuario
* NUI (Natural user interface): Interfaz natural de usuario

La interfaz de usuario es el medio con que el usuario puede comunicarse con una máquina, equipo, computadora o dispositivo, y comprende todos los puntos de contacto entre el usuario y el equipo.

Normalmente suelen ser fáciles de entender y fáciles de accionar, aunque en el ámbito de la informática es preferible referirse a que suelen ser "usables", "amigables e intuitivos" porque es complejo.

**Administración de tareas**

* Monotarea: Solamente permite ejecutar un proceso (aparte de los procesos del propio SO) en un momento dado. Una vez que empieza a ejecutar un proceso, continuará haciéndolo hasta su finalización y/o interrupción.
* Multitarea: Es capaz de ejecutar varios procesos al mismo tiempo. Este tipo de SO normalmente asigna los recursos disponibles (CPU, memoria, periféricos) de forma alternada a los procesos que los solicitan, de manera que el usuario percibe que todos funcionan a la vez, de forma concurrente.

**Administración de usuarios**

* Monousuario: Solo permite ejecutar los programas de un usuario al mismo tiempo.
* Multiusuario: Permite que varios usuarios ejecuten simultáneamente sus programas, accediendo a la vez a los recursos de la computadora. Normalmente estos sistemas operativos utilizan métodos de protección de datos, de manera que un programa no pueda usar o cambiar los datos de otro usuario.

**Administración de sesiones**

* Monosesión: Sistemas operativos que son capaces de ejecutar una sola sesión de usuario. Los sistemas operativos de hace tiempo como Windows 98 o Windows 95.
* Multisesión: Sistemas operativos que son capaces de ejecutar varias sesiones de usuario a la vez. La funcionalidad de multisesión se puede instalar mediante aplicaciones, pero es el sistema operativo el que permite dicho funcionamiento. Los servidores de la familia Windows Server, la mayoría de las versiones de Linux.

**Manejo de recursos**

* Centralizado: Permite usar los recursos de una sola computadora.
* Distribuido: Permite utilizar los recursos (memoria, CPU, disco, periféricos...) de más de una computadora al mismo tiempo.

**-Riegos**

**-Alcance**

**-Tareas**

**>> Análisis**

**-Que tiene que hacer el software**

**-Requerimiento (Herramientas)**

**-Característica (Tipo de proyecto)**

**-Algoritmo (Describe paso a paso el proyecto)**

**-Diagramas (Gantt, Caso de Uso, Diagrama de Flujos, Perl)**

**>>Diseño arquitectura de software**

**-Estructura**

**-Patrones**

**-Catálogos**

**-Primera propuesta de diseño**

**>>Implementación o programación**

**-Entorno de desarrollo**

**-Lenguaje**

**-Duración**

**-Programación**

**-Algoritmos**

**-Diagramas**

**-Documentar código**

**-Reglas de códigos de formatos**

**-Pruebas Modulares**

**>>Prueba Del Sistema**

**-Compatibilidad**

**-Detectar inconformidad de las entidades**

**-Pruebas de los módulos**

**-Calidad del sistema**

**>>Publicación**

**-Instaladores**

**-Plataformas**

**>> Uso Mantenimiento / Cuidado**

**-Preventivo**

**-Asistencia técnica**

**-Soporte**

**-Correctivo**

**-Perfectivo**

**-Propuesta de mejoras**

**-Actualizaciones**

**-Documentación (Manual de Usuario O Tutoriales)**

**>>Diseño de Usabilidad**

**-Dirigida a los usuarios**

**-Experiencia de usuario**