**3.1. ¿Qué es una traza de instrucciones?**

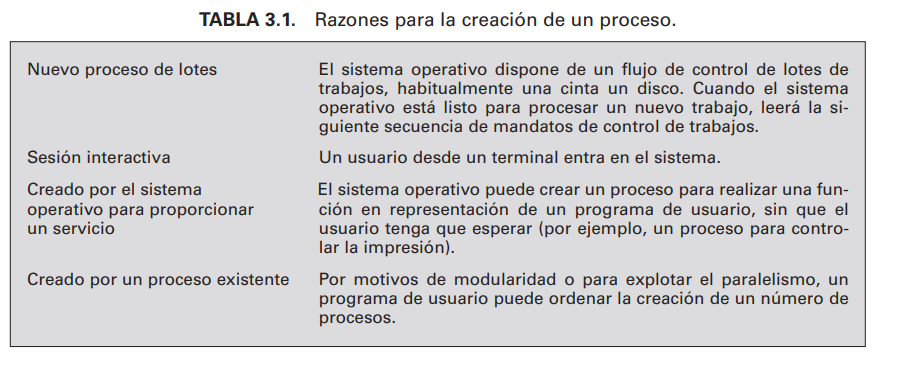
Una traza de instrucciones es una secuencia ordenada de las instrucciones ejecutadas por un proceso durante su ejecución. En el contexto del procesamiento, cada proceso tiene su propia traza de instrucciones, que representa la secuencia de acciones realizadas por el procesador mientras ejecuta ese proceso en particular. Por lo tanto, la traza de instrucciones de un proceso proporciona una visión detallada de las operaciones específicas llevadas a cabo por ese proceso a lo largo del tiempo.

Además, las trazas de instrucciones de varios procesos pueden ser analizadas para comprender cómo se entrelazan las ejecuciones de los diferentes procesos en el procesador. Esto permite caracterizar el comportamiento del procesador en términos de cómo gestiona y ejecuta múltiples procesos concurrentemente.

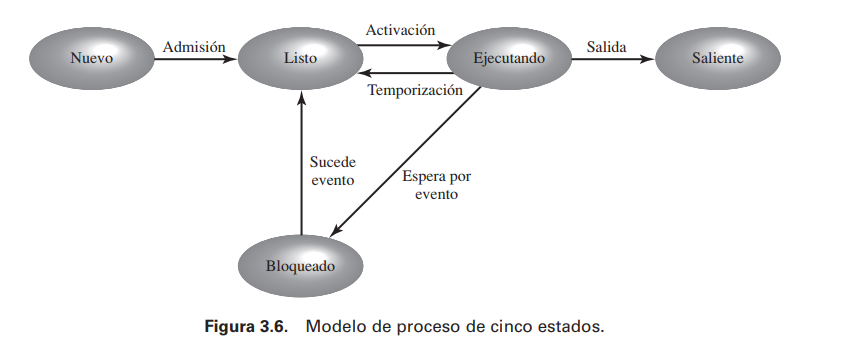
**3.2. ¿Cuáles son los eventos comunes que llevan a la creación de un proceso?**

Existen cuatro eventos comunes que llevan a la creación de un proceso, como se indica en la Tabla 3.1. En un entorno por lotes, un proceso se crea como respuesta a una solicitud de trabajo. En un entorno interactivo, un proceso se crea cuando un nuevo usuario entra en el sistema. En ambos casos el sistema operativo es responsable de la creación de nuevos procesos. Un sistema operativo puede, a

petición de una aplicación, crear procesos. Por ejemplo, si un usuario solicita que se imprima un fichero, el sistema operativo puede crear un proceso que gestione la impresión. El proceso solicitado puede, de esta manera, operar independientemente del tiempo requerido para completar la tarea de impresión.



3.3. Para el modelo de procesamiento de la Figura 3.6, defina brevemente cada estado.



* **Ejecutando.** El proceso está actualmente en ejecución. Para este capítulo asumimos que el computador tiene un único procesador, de forma que sólo un proceso puede estar en este estado en un instante determinado.
* **Listo.** Un proceso que se prepara para ejecutar cuando tenga oportunidad.
* **Bloqueado.** Un proceso que no puede ejecutar hasta que se cumpla un evento determinado o se complete una operación E/S.
* **Nuevo.** Un proceso que se acaba de crear y que aún no ha sido admitido en el grupo de procesos ejecutables por el sistema operativo. Típicamente, se trata de un nuevo proceso que no ha sido cargado en memoria principal, aunque su bloque de control de proceso (BCP) si ha sido creado.
* **Saliente.** Un proceso que ha sido liberado del grupo de procesos ejecutables por el sistema operativo, debido a que ha sido detenido o que ha sido abortado por alguna razón.

**3.4. ¿Qué significa la expulsión de un proceso?**

Una de las transiciones que podemos observar en la Figura 3.6 es la de:

**Ejecutando** 🡪 **Listo.** La razón más habitual para esta transición es que el proceso en ejecución haya alcanzado el máximo tiempo posible de ejecución de forma ininterrumpida; prácticamente todos los sistemas operativos multiprogramados imponen este tipo de restricción de tiempo. Existen otras posibles causas alternativas para esta transición, que no están incluidas en todos los sistemas operativos. Es de particular importancia el caso en el cual el sistema operativo asigna diferentes niveles de prioridad a diferentes procesos. Supóngase, por ejemplo, que el proceso A está ejecutando a un determinado nivel de prioridad, y el proceso B, a un nivel de prioridad mayor, y que se encuentra bloqueado. Si el sistema operativo se da cuenta de que se produce un evento al cual el proceso B está esperando, moverá el proceso B al estado de Listo. Esto puede interrumpir al proceso A y poner en ejecución al proceso B. Decimos, en este caso, que el sistema operativo **ha expulsado al proceso** A**(\*)**. Adicionalmente, un proceso puede voluntariamente dejar de utilizar el procesador. Un ejemplo son los procesos que realiza alguna función de auditoría o de mantenimiento de forma periódica.

**(\*)** En general, el término expulsión (*preemption*) se define como la reclamación de un recurso por parte de un proceso antes de que el proceso que lo poseía finalice su uso. En este caso, el recurso es el procesador. El proceso está ejecutando y puede continuar su ejecución pero es expulsado por otro proceso que va a entrar a ejecutar.

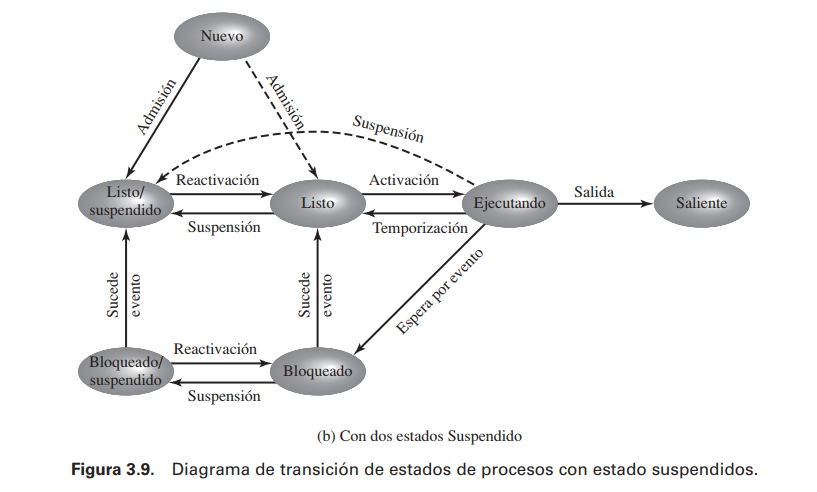
**3.5. ¿Que es el *swapping* y cuál es su objetivo?**

El swapping es una técnica utilizada en los sistemas operativos para gestionar la memoria cuando la capacidad de la memoria principal es insuficiente para alojar todos los procesos activos. Consiste en mover parte o la totalidad de un proceso de la memoria principal al almacenamiento secundario, como el disco duro, liberando así espacio en la memoria principal para otros procesos.

Cuando no hay procesos en la memoria principal que estén listos para ser ejecutados de inmediato, es decir, no hay procesos en estado "Listo", el sistema operativo puede seleccionar uno de los procesos bloqueados y transferirlo al disco, colocándolo en un estado llamado "Suspendido". Luego, puede cargar otro proceso de la cola de procesos suspendidos en la memoria principal para su ejecución.

El objetivo del swapping es optimizar el uso de la memoria principal y mejorar el rendimiento del sistema al garantizar que los procesos activos tengan acceso rápido a la memoria cuando sea necesario. Aunque el swapping implica operaciones de E/S, como la transferencia de datos entre la memoria principal y el disco, generalmente mejora el rendimiento del sistema al permitir una gestión más eficiente de la memoria.

**3.6. ¿Por qué la Figura 3.9 tiene dos estados bloqueados?**



**Bloqueado.** El proceso está en memoria principal y esperando un evento.

**Bloqueado/Suspendido.** El proceso está en almacenamiento secundario y esperando un evento.

Entre estos 2 estados, sus transiciones quedarían:

**Bloqueado 🡪 Bloqueado/Suspendido. Suspensión.** Si no hay procesos listos, entonces al menos uno de los procesos bloqueados se transfiere al disco para hacer espacio para otro proceso que no se encuentra bloqueado. Esta transición puede realizarse incluso si hay procesos listos disponibles, si el sistema operativo determina que el proceso actualmente en ejecución o los procesos listos que desea ejecutar requieren más memoria principal para mantener un rendimiento adecuado.

**Bloqueado/Suspendido** 🡪 **Bloqueado. Reactivación.** La incursión de esta transición puede parecer propia de un diseño más bien pobre. Después de todo, si hay un proceso que no está listo para ejecutar y que no está en memoria principal, ¿qué sentido tiene traerlo? Pero considérese el siguiente escenario: un proceso termina, liberando alguna memoria principal. Hay un proceso en la cola de Bloqueados/Suspendidos con mayor prioridad que todos los procesos en la cola de Listos/Suspendidos y el sistema operativo tiene motivos para creer que el evento que lo bloquea va a ocurrir en breve. Bajo estas circunstancias, sería razonable traer el proceso Bloqueado a memoria por delante de los procesos Listos.

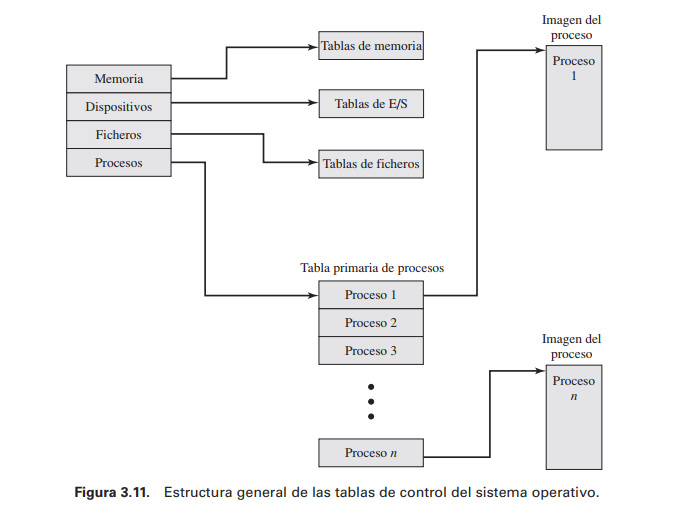
**3.7. Indique cuatro características de un proceso suspendido.**

Podemos analizar el concepto de procesos suspendidos, definiendo un proceso suspendido como el que cumple las siguientes características:

1. El proceso no está inmediatamente disponible para su ejecución.
2. El proceso puede estar o no a la espera de un evento, si es así, la condición de bloqueo es independiente de la condición estar suspendido, y si sucede el evento que lo bloquea, eso no habilita al proceso para su ejecución inmediata.
3. El proceso fue puesto en estado suspendido por un agente: bien el proceso mismo, el proceso padre o el sistema operativo, con el propósito de prevenir su ejecución.
4. El proceso no puede ser recuperado de este estado hasta que el agente explícitamente así lo indique.

**3.8. ¿Para qué tipo de entidades el sistema operativo mantiene tablas de información por motivos de gestión?**

Si el sistema operativo se encarga de la gestión de procesos y recursos, debe disponer de información sobre el estado actual de cada proceso y cada recurso. El mecanismo universal para proporcionar esta información es el siguiente: el sistema operativo construye y mantiene tablas de información sobre cada entidad que gestiona. Se indica una idea general del alcance de esta tarea en la Figura 3.11, que muestra cuatro diferentes tipos de tablas mantenidas por el sistema operativo: memoria, E/S, ficheros y procesos. A pesar de que los detalles difieren de un sistema operativo a otro, fundamentalmente, todos los sistemas operativos mantienen información de estas cuatro categorías.

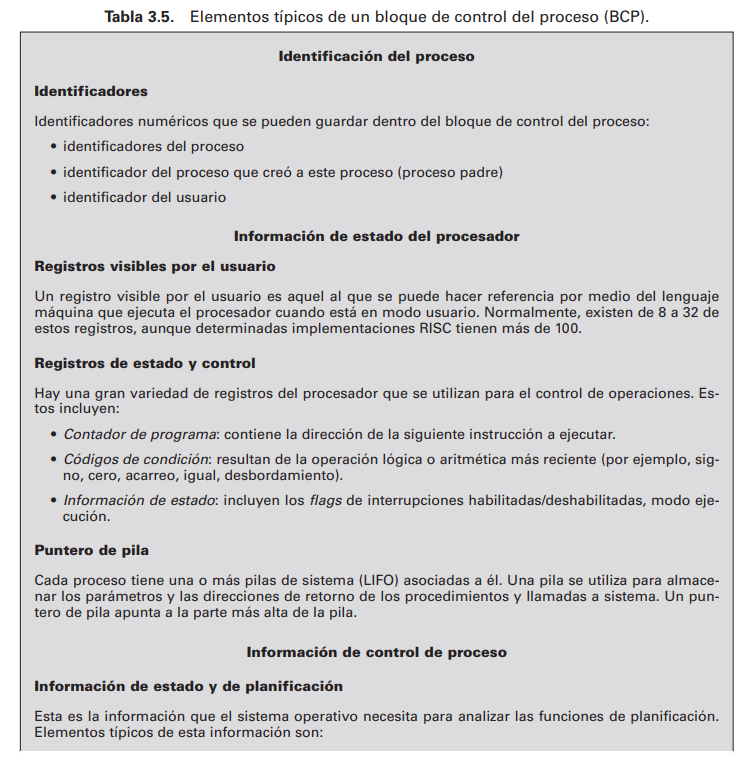
****

**3.9. Indique tres categorías generales de información que hay en el bloque de control de proceso.**

Cada proceso está asociado a un número de atributos que son utilizados por el sistema operativo para controlar el proceso. Normalmente, el conjunto de estos atributos se denomina **bloque de control del proceso (BCP).** Nos podemos referir al conjunto de programa, datos, pila, y atributos, como la **imagen del proceso.**

Podemos agrupar la información del bloque de control del proceso en tres categorías generales:

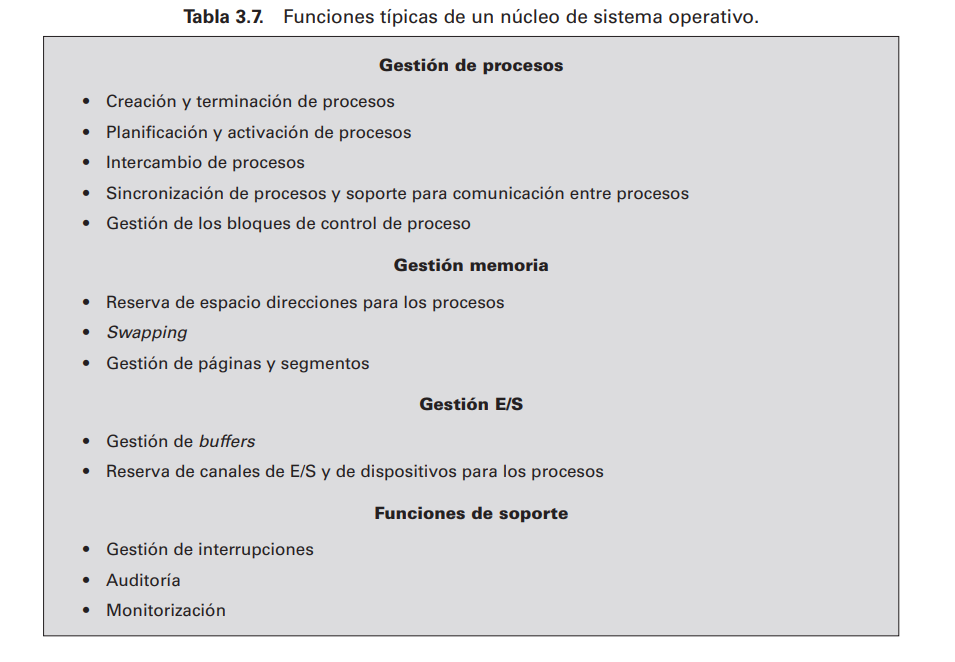
* Identificación del proceso
* Información de estado del procesador
* Información de control del proceso

****

**3.10. ¿Por qué se necesitan dos modos (usuario y núcleo)?**

Antes de continuar con nuestra explicación sobre cómo el sistema operativo gestiona los procesos, necesitamos distinguir entre los modos de ejecución del procesador, normalmente asociados con el sistema operativo y los asociados con los programas de usuario. Muchos procesadores proporcionan al menos dos modos de ejecución. Ciertas instrucciones se pueden ejecutar en modos privilegiados únicamente. Éstas incluirían lectura y modificación de los registros de control, por ejemplo la palabra de estado de programa; instrucciones de E/S primitivas; e instrucciones relacionadas con la gestión de memoria. Adicionalmente, ciertas regiones de memoria sólo se pueden acceder en los modos más privilegiados.

El modo menos privilegiado a menudo se denomina **modo usuario**, porque los programas de usuario típicamente se ejecutan en este modo. El modo más privilegiado se denomina **modo sistema**, **modo control** o **modo núcleo**. Este último término se refiere al núcleo del sistema operativo, que es la parte del sistema operativo que engloba las funciones más importantes del sistema. La Tabla 3.7 lista las funciones que normalmente se encuentran dentro del núcleo del sistema operativo.



El motivo por el cual se usan los otros modos es claro. Se necesita proteger al sistema operativo y a las tablas clave del sistema, por ejemplo los bloques de control de proceso, de la interferencia con programas de usuario. En modo núcleo, el software tiene control completo del procesador y de sus instrucciones, registros, y memoria. Este nivel de control no es necesario y por seguridad no es recomendable para los programas de usuario.

Aparecen dos cuestiones: ¿cómo conoce el procesador en que modo está ejecutando y cómo este modo puede modificarse? En lo referente la primera cuestión, existe típicamente un bit en la palabra de estado de programa (PSW) que indica el modo de ejecución. Este bit se cambia como respuesta a determinados eventos. Habitualmente, cuando un usuario realiza una llamada a un servicio del sistema operativo o cuando una interrupción dispara la ejecución de una rutina del sistema operativo, este modo de ejecución se cambia a modo núcleo y; tras la finalización del servicio, el modo se fija de nuevo a modo usuario.

**3.11. ¿Cuáles son los pasos que realiza el sistema operativo para la creación de un proceso?**

Una vez que el sistema operativo decide, por cualquier motivo, crear un proceso procederá de la siguiente manera:

1. **Asignar un identificador de proceso único al proceso**. En este instante, se añade una nueva entrada a la tabla primaria de procesos, que contiene una entrada por proceso.
2. **Reservar espacio para proceso**. Esto incluye todos los elementos de la imagen del proceso. Para ello, el sistema operativo debe conocer cuánta memoria se requiere para el espacio de direcciones privado (programas y datos) y para la pila de usuario. Estos valores se pueden asignar por defecto basándonos en el tipo de proceso, o pueden fijarse en base a la solicitud de creación del trabajo remitido por el usuario. Si un proceso es creado por otro proceso, el proceso padre puede pasar los parámetros requeridos por el sistema operativo como parte de la solicitud de la creación de proceso. Si existe una parte del espacio direcciones compartido por este nuevo proceso, se fijan los enlaces apropiados. Por último, se debe reservar el espacio para el bloque de control de proceso (BCP).
3. **Inicialización del bloque de control de proceso**. La parte de identificación de proceso del BCP contiene el identificador del proceso así como otros posibles identificadores, tal como el indicador del proceso padre. En la información de estado de proceso del BCP, habitualmente se inicializa con la mayoría de entradas a 0, excepto el contador de programa (fijado en el punto entrada del programa) y los punteros de pila de sistema (fijados para definir los límites de la pila del proceso). La parte de información de control de procesos se inicializa en base a los valores por omisión, considerando también los atributos que han sido solicitados para este proceso. Por ejemplo, el estado del proceso se puede inicializar normalmente a Listo o Listo/Suspendido. La prioridad se puede fijar, por defecto, a la prioridad más baja, a menos que una solicitud explicita la eleve a una prioridad mayor. Inicialmente, el proceso no debe poseer ningún recurso (dispositivos de E/S, ficheros) a menos que exista una indicación explícita de ello o que haya sido heredados del padre.
4. **Establecer los enlaces apropiados**. Por ejemplo, si el sistema operativo mantiene cada cola del planificador como una lista enlazada, el nuevo proceso debe situarse en la cola de Listos o en la cola de Listos/Suspendidos.
5. **Creación o expansión de otras estructuras de datos**. Por ejemplo, el sistema operativo puede mantener un registro de auditoría por cada proceso que se puede utilizar posteriormente a efectos de facturación y/o de análisis de rendimiento del sistema.

**3.12. ¿Cuál es la diferencia entre interrupción y *trap*?**

La principal diferencia entre una interrupción y una trampa (trap) radica en su origen y propósito:

1. **Interrupción:** Ocurren debido a eventos externos al proceso en ejecución, como la finalización de una operación de E/S. Las interrupciones pueden provenir de dispositivos de hardware u otros elementos externos al procesador. Cuando se produce una interrupción, el procesador suspende temporalmente la ejecución del proceso actual, guarda su estado y ejecuta un manejador de interrupciones específico para manejar el evento. Después de procesar la interrupción, el control se devuelve al proceso interrumpido o se asigna a otro proceso, dependiendo del diseño del sistema operativo.

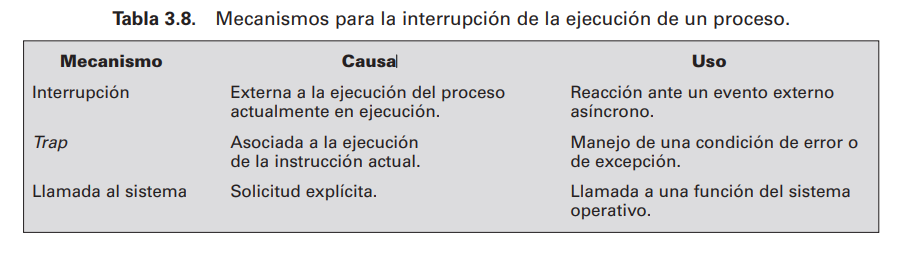
2. **Trampa (Trap):** Se generan debido a condiciones específicas dentro del proceso en ejecución, como errores o excepciones. Por ejemplo, un intento de acceso no permitido a un archivo puede provocar una trampa. Las trampas se utilizan para capturar eventos internos del programa que requieren atención del sistema operativo. Cuando se activa una trampa, el procesador detiene la ejecución normal del programa, guarda su estado y ejecuta un manejador de trampas para manejar la situación. Posteriormente, el control puede volver al proceso original o tomar acciones específicas según el tipo de trampa.

Por lo tanto, las interrupciones son eventos externos que requieren atención del sistema operativo, mientras que las trampas son eventos internos del proceso que necesitan ser manejados por el sistema operativo. Ambos mecanismos permiten que el sistema operativo tome el control del procesador para realizar acciones específicas en respuesta a eventos importantes.

**3.13. Dé tres ejemplos de interrupción.**

Dentro de una **interrupción** ordinaria, el control se transfiere inicialmente al manejador de interrupción, que realiza determinadas tareas internas y que posteriormente salta a una rutina del sistema operativo, encargada de cada uno de los tipos de interrupciones en particular. Algunos ejemplos son:

* **Interrupción de reloj.** El sistema operativo determinar si el proceso en ejecución ha excedido o no la unidad máxima de tiempo de ejecución, denominada **rodaja de tiempo** (*time slice*). Esto es, una rodaja de tiempo es la máxima cantidad de tiempo que un proceso puede ejecutar antes de ser interrumpido. En dicho caso, este proceso se puede pasar al estado de Listo y se puede activar otro proceso.
* **Interrupción de E/S.** El sistema operativo determina qué acción de E/S ha ocurrido. Si la acción de E/S constituye un evento por el cual están esperando uno o más procesos, el sistema operativo mueve todos los procesos correspondientes al estado de Listos (y los procesos en estado Bloqueado/Suspendido al estado Listo/Suspendido). El sistema operativo puede decidir si reanuda la ejecución del proceso actualmente en estado Ejecutando o si lo expulsa para proceder con la ejecución de un proceso Listo de mayor prioridad.
* **Fallo de memoria.** El procesador se encuentra con una referencia a una dirección de memoria virtual, a una palabra que no se encuentra en memoria principal. El sistema operativo debe traer el bloque (página o segmento) que contiene la referencia desde memoria secundaria a memoria principal. Después de que se solicita la operación de E/S para traer el bloque a memoria, el proceso que causó el fallo de memoria se pasa al estado de Bloqueado; el sistema operativo realiza un cambio de proceso y pone a ejecutar a otro proceso. Después de que el bloque de memoria solicitado se haya traído, el proceso pasará al estado Listo.



**3.14. ¿Cuál es la diferencia entre cambio de modo y cambio de proceso? ~~🡪 No estoy entendiendo bien concepto Cambio de modo~~ Es lo de modo kernel modo usuario**

**Cambio de modo**. En la fase de interrupción, el procesador comprueba que no exista ninguna interrupción pendiente, indicada por la presencia de una señal de interrupción. Si no hay interrupciones pendientes, el procesador pasa a la fase de búsqueda de instrucción, siguiendo con el programa del proceso actual. Si hay una interrupción pendiente, el proceso actúa de la siguiente manera:

1. Coloca el contador de programa en la dirección de comienzo de la rutina del programa manejador de la interrupción.
2. Cambia de modo usuario a modo núcleo de forma que el código de tratamiento de la interrupción pueda incluir instrucciones privilegiadas.

El procesador, acto seguido, pasa a la fase de búsqueda de instrucción y busca la primera instrucción del programa de manejo de interrupción, que dará servicio a la misma. En este punto, habitualmente, el contexto del proceso que se ha interrumpido se salvaguarda en el bloque de control de proceso del programa interrumpido.

Una pregunta que se puede plantear es, ¿qué constituye el contexto que se debe salvaguardar? La respuesta es que se trata de toda la información que se puede ver alterada por la ejecución de la rutina de interrupción y que se necesitará para la continuación del proceso que ha sido interrumpido. De esta forma, se debe guardar la parte del bloque de control del proceso que hace referencia a la información de estado del procesador. Esto incluye el contador de programa, otros registros del procesador, y la información de la pila.

¿Se necesita hacer algo más? Eso depende de qué ocurra luego. El manejador de interrupción es habitualmente un pequeño programa que realiza unas pocas tareas básicas relativas a la interrupción. Por ejemplo, borra el *flag* o indicador que señala la presencia de interrupciones. Puede enviar una confirmación a la entidad que lanzó dicha interrupción, como por ejemplo el módulo de E/S. Y puede realizar algunas tareas internas variadas relativas a los efectos del evento que causó la interrupción. Por ejemplo, si la interrupción se refiere a un evento de E/S, el manejador de interrupción comprobará la existencia o no de una condición de error. Si ha ocurrido un error, el manejador mandará una señal al proceso que solicitó dicha operación de E/S. Si la interrupción proviene del reloj, el manejador la va a pasar el control al activador, el cual decidirá pasar a otro proceso debido a que la rodaja de tiempo asignada a ese proceso ha expirado.

¿Qué pasa con el resto de información del bloque de control de proceso? Si a esta interrupción le sigue un cambio de proceso a otro proceso, se necesita hacer algunas cosas más. Sin embargo, en muchos sistemas operativos, la existencia de una interrupción no implica necesariamente un cambio de proceso. Es posible, por tanto, que después de la ejecución de la rutina de interrupción, la ejecución se reanude con el mismo proceso. En esos casos sólo se necesita salvaguardar la información del estado del procesador cuando se produce la interrupción y restablecerlo cuando se reanude la ejecución del proceso. Habitualmente, las operaciones de salvaguarda y recuperación se realizan por hardware.

**Cambio del estado del proceso**. El cambio de modo es un concepto diferente del cambio de proceso. Un cambio de modo puede ocurrir sin que se cambie el estado del proceso actualmente en estado Ejecutando. En dicho caso, la salvaguarda del estado y su posterior restauración comportan sólo una ligera sobrecarga. Sin embargo, si el proceso actualmente en estado Ejecutando, se va a mover a cualquier otro estado (Listo, Bloqueado, etc.), entonces el sistema operativo debe realizar cambios sustanciales en su entorno. Los pasos que se realizan para un cambio de proceso completo son:

1. Salvar el estado del procesador, incluyendo el contador de programa y otros registros.
2. Actualizar el bloque de control del proceso que está actualmente en el estado Ejecutando. Esto incluye cambiar el estado del proceso a uno de los otros estados (Listo, Bloqueado, Listo/Suspendido, o Saliente). También se tienen que actualizar otros campos importantes, incluyendo la razón por la cual el proceso ha dejado el estado de Ejecutando y demás información de auditoría.
3. Mover el bloque de control de proceso a la cola apropiada (Listo, Bloqueado en el evento *i*, Listo/Suspendido).
4. Selección de un nuevo proceso a ejecutar; esta cuestión se analiza con más detalle en la Parte Cuatro.
5. Actualizar el bloque de control del proceso elegido. Esto incluye pasarlo al estado Ejecutando.
6. Actualizar las estructuras de datos de gestión de memoria. Esto se puede necesitar, dependiendo de cómo se haga la traducción de direcciones; estos aspectos se cubrirán en la Parte Tres.
7. Restaurar el estado del procesador al que tenía en el momento en el que el proceso seleccionado salió del estado Ejecutando por última vez, leyendo los valores anteriores de contador de programa y registros.