A MODO DE OTRO SEGUNDO RESUMEN DE LA UNIDAD 1 PROGRAMACIÓN DE PROCESOS

1. Programa y Proceso

Un programa es un **objeto estático**, normalmente almacenado en un fichero binario en un medio de **almacenamiento secundario**.

Un programa contiene un conjunto de instrucciones que se pueden **ejecutar directament**e en una máquina, esta puede ser una máquina física o, como en el caso dellenguaje Java, una máquina virtual.

Un proceso es una **entidad dinámica**, es una instancia de un programa en ejecución. La ejecución de un programa consiste en la creación y ejecución de un proceso. El programa a ejecutar debe cargarse en memoria.

Durante la ejecución del programa, es decir, durante la vida del proceso, este utiliza diversos **recursos: (recursos asociados al proceso no al programa)**

1. Memoria principal. **Imagen de memoria**. Consta de una secuencia o bloque de celdas de memoria, todas con la misma longitud de bits, y cada celda se indentifica por su posición. El programa en ejecución debe cargarse en un bloque de la memoria principal que se asigna al proceso. Este proceso también puede obtener más memoria dinamicamente durante su ejecución.
2. **Procesador de CPU**. El procesador guarda, en un registro especial, el contador de programa (PC o program counter) la dirección de memoria de la siguiente instrucción a ejecutarse, es decir, la siguiente en ser cargada en la unidade de control para su decodificación. Un proceso no puede operar directamente con los contenidos de la memoria, estos deben antes traerse de ella y cargarse en registros del procesador. Las operaciones se realizan en una unidad aritmético-lógica o UAL. Su resultado se obtiene ebn un registro de la UAL desdeel que se puede transferir a una posición de la memoria.
3. **Dispositivos de entrada/salid**a. Los procesos comparten los dispositivos de E/S. Debe guardarse información acerca de a qué procesos se les ha otorgado acceso a un dispositivo de E/S.

Toda la información asociada a un proceso se guarda en un registro **bloque de control de proceso** (en inglés, **PCB**, o process control block). Si un mismo programa se ejecuta varias veces, se obtiene una instancia del mismo para cada ejecución, es decir, se creará cada vez un nuevo proceso y cada uno tendrá su propio **bloque de control de proceso** y su propio **identificador de proceso** y su bloque o **imagen de memori**a donde reside o existe el proceso.

Un proceso puede crear nuevos procesos durante su ejecución, que pueden, a su vez, crear nuevos procesos, de esta forma a partir de un proceso inicial puede crearse una **jerarquía de procesos**. **Se construye un árbol n-ario de procesos.**

Los procesos que forman parte del mismo árbol de procesos están relacionados en una relación de jerarquía (proceso padre – proceso hijo), todos ellos colaboran entonces en la ejeción de una misma tarea, tenemos una aplicación multiproceso. Habitualmente los procesos coexisten en una misma máquina, se ejecutan concurrentemente, pero no se comunican nin sincronizan entre sí. Sin embargo, en una aplicación multiproceso el proceso padre podrá comunicarse con el/los procesos hijos que lance, esta comunicación entre proceso padre y proceso hijo es una suerte de sincronización entre procesos.

En windows se puede ver la jerarquía de procesos con **Process Explorer**, programa que suple carencias del **administrador de tareas**.

Los sistemas operativos actuales permiten ejecutar múltiples procesos “a la vez” CONCURRENTEMENTE. Aunque exista un solo procesador en el sistema. A intervalos regulares de tiempo la ejecución del proceso en curso se detiene porque toma el control un programa especial

del sistema operativo que detiene la ejecución del proceso actual (lo desaloja de la unidad de control) y pasa a ejecutar otro proceso (lo carga en la unidad de control).

Definimos **MULTITAREA (MULTITASKING) CONCURRENCIA DE PROCESOS** como la ejecución simultánea de más de un proceso en un procesador **a lo largo de un intervalo de tiempo. En un instante dado sólo un proceso está en estado de ejecución.**

En cada momento hay un único proceso en ejecución, pero la rápida alternancia entre ellos hace que todos puedan seguir avanzando a lo largo del tiempo, de manera que, considerando no un instante, sino un intervalo de tiempo se puede decir que múltiples procesos se ejecutan simultáneamente, y así lo percibe un usuario del sistema.

Antes de desalojar a un proceso de la unidad de control, detener su ejecución, **se guarda el estado** de ejecución del proceso, para poder más adelante poder retomar su ejecución en el mismo punto en que se interrumpió. En particular se guardan los contenidos del registro contador de programa y de los registros del procesador, de la Unidad de Control. Esta operación se conoce como **cambio de contexto**. Se cargan enlos registros del procesador el estado del proceso que pasa a ejecución.

Si disponemos de un **sistema** con **más de un procesador en ese caso podremos ejecutar en paralelo varios procesos, uno cada vez en cada procesador.**

Pero normalmente siempre habrá muchos más procesos que procesadores, de modo que se hará **multitarea o ejecución concurrente de procesos en cada procesador.**

Con la multitarea o concurrencia de procesos el tiempo de ejecución total de un proceso se ve incrementado porque al tiempo de ejecución del proceso hay que sumar el tiempo necesario para los cambios de contexto además del tiempo que necesita el sistema operativo para planificar los distintos procesos.

La ejecución concurrente de varios procesos en un **sistema monoprocesador** se conoce con el nombre de **multiporgramación**. **La técnica del cambio del contexto** permite una multitarea efectiva en un intervalo de tiempo y un **mejor aprovechamiento del procesador** que asigna a otro proceso tiempo de procesador durante el tiempo en que un proceso está inactivo o a la espera de que termine de realizar una operación de E/S.

Los actuales procesadores multinúcleo (multicore, dual-core con dos núcleos, quad-core con cuatro núcleos, hexa-core con seis núcleos y octa-core con ocho núcleos) contienen varios núcleos o unidades de proceso integradas. Se pueden considerar un tipo de multiprocesadores aunque con importantes diferencias en relación con sistemas de varios procesadores independientes. En estos casos los procesos se pueden ejecutar con **paralelismo real**, los hilos o hebras de un mismo proceso en ejecución se ejecutan paralelamente en cada núcleo o core.

Sistemas multiprocesadores

Los sistemas multiprocesadores disponen de más de un procesador. Se pueden clasificar dependiendo de su arquitectura en:

- sistemas multiprocesadores fuertemente acoplados
- sistemas multiprocesadores débilmente acoplados

Sistemas multiprocesadores fuertemente acoplados, en estos sistemas existe UNA memoria compartida para todos los procesadores a la que todos ellos acceden a través de un bus de conexión y también acceden al istema de E/S a través del mismo bus.

Se pueden distinguir dos tipos de sistemas multiprocesador fuertemente acoplados:

  - Sistemas multiprocesadores simétricos, en los que los procesadores del sistema funcionan todos de igual manera y no hay ninguno diferenciado del resto. es

    - Sistemas multiprocesadores asimétricos en los que uno de los procesadores del sistema, el maestro o master, controla el resto de procesadores(slave), este tipo de sistemas también se llaman **master-slave.**

- **Sistemas multiprocesadores débilmente acoplados**, en estos sistemas no hay memoria compartida, sino que cada procesador tiene su propia memoria y su propio sistema de E/S. **Existe una red de comunicaciones entre los procesadores.**
    - Una posibilidad es que los ordenadores de alto rendimiento estén físicamente cercanos y comunicados por una red a medida de alta velocidad y estructura regular, este tipo de sistemas se utilizan en **supercomputación** y para aplicaciones intensivas de cálculo.
    - Otra alternativa a este tipo de sistemas son los **clusters,** en lugar de procesadores de alto rendimiento interconectados mediante una red especializada y diseñada a medida, están formados por ordenadores convencionales conectados mediante una red de área local normal.
    - Otra posibilidad son los **sistemas distribuídos,** los procesadores están en ordenadores independientes conectados mediante una red de área extensa o a través de internet, se caracterizan por la heterogeneidad del hardware.

La programación distribuída consiste en la ejecución de varios procesos concurrentes en un sistema distribuido. En un sistema distribuido no hay memoria compartida ni red de conexión específica entre los distintos ordenadores. La comunicación entre los procesos se realiza mediante mensajes a través de la red de comunicaciones, para lo que se suelen utilizar los protocolos de red estándares TCP o UDP.

Kernel núcleo del sistema operativo

La parte central del sistema operativo se denomina kernel o núcleo. Es una parte pequeña y muy optimizada del sistema operativo que da respuesta a múltiples eventos mediante un mecanismo de **gestión de interrupciones.** Las interrupciones pueden producirse por múltiples motivos.

- Eventos generados por el **hardware**, movimientos del ratón, pulsaciones de teclado, final de una operación de lectura o escritura en disco..
- Llamadas al sistema. El kernel proporciona acceso a los procesos a determinados servicios o funcionalidades del sistema operativo mediante **llamadas al sistemas→ interrupciones**
- I**nterrupciones periódicas** que invoca el **planificador de procesos a corto plazo** del sistema operativo.

Cuando sucede una interrupción, el procesador deja de ejecutar el proceso en curso (desalojo) y pasa a ejecutar una rutina de tratamiento de la interrupción. Cuando esta finaliza, se reanudará la ejecución del proceso en el mismo lugar en que estaba. (cambio de contexto de un proceso)

El procesador tiene dos modos de funcionamiento: modo kernel o supervisor y modo usuario. Las rutinas de tratamiento de interrupciones se ejecutan en modo kernel.

Estados de un proceso

- **Nuevo**
- **Terminado**
    - Gestionados por el p**lanificador de procesos a largo plazo**. Un proceso debe crearse y asignarle memoria interna para su ejecución, el planificador de procesos a largo plazo decide qué procesos son admitidos para su ejecución, se cargan en la memoria

principal y pasan a estado listo. El planificador a largo plazo es importante en sistemas donde hay muchos procesos no interactivos y es el propio sistema operativo el que gestiona su lanzamiento.

- **Listo**
- **En ejecución**
- **Bloqueado**

    - La planificación de procesos a corto plazo, se realiza con todos los procesos vivos cargados en memoria principal, su objetivo es repartir el tiempo de procesador entre todos los procesos, de manera que se consiga un **máximo aprovechamiento del procesador.**

Para la gestión de los procesos el sistema operativo utiliza un **bloque de control de proceso**, uno para cada proceso, un **identificador de proceso**, uno para cada proceso. El sistema operativo mantiene las colas de procesos para cada uno de los estados, también mantiene colas para cada dispositivo de E/S con información de los procesos que tienen operaciones pendientes en dicho dispositivo.

La gestión o manteniemiento de estas colas de procesos permite obtener de forma muy eficiente **la planificación de los todos los procesos**, es decir, la ejecución concurrente de todos los procesos.

Hilos vs Procesos

Crear un proceso consume una considerable cantidad de tiempo y recursos del sistema, necesitamos reservas espacio de memoria interna para él. Además la comunicación entre procesos requerirá el empleo de una serie de recursos o mecanismos del sistema.

Los hilos o hebras (threads) llamados también procesos ligeros, un proceso es un árbol n_ário de hilos, inicialmente como mínimo un hilo, el hilo primario. Pero se podrán crear más de un hilo o secuencia de código en ejecución dentro del proceso, el proceso finaliza cuando finalizan la ejecución de todos sus hilosy a la inversa si elproceso finaliza finalizan también todos sus hilos.

La creación de un nuevo hilo para un proceso ya existente no requiere inicializar ni reservar espacio en la memoria interna. Todos los recursos de un proceso son compartidos por todos los hilos del mismo, los hilos de un proceso se ejecutan en el mismo espacio de direcciones, en el del proceso al que pertenecen. La comunicación entre hilos es muy sencilla, no se necesitarán, como en el caso de los procesos, mecanismo de comunicación especiales. Sin embargo, se necesitarán **mecanismos de sincronización entre hilos** para evitar problemas que puedan darse si dos o más hilos modifican sin control el mismo objeto en memoria.

Servicios

Los servicios son un tipo particular de procesos que proporcionan servicios a otros procesos, se ejecutan en segundo plano y no son utilizados directamente por los usuarios, no interactúan.

Los servicios se ejecutan continuamente, normalmente son iniciados por el sistema operativo durante su arranque. Cada sistema operativo tiene sus propios mecanismos para gestionar los servicios, normalmente se pueden configurar para que arranquen automáticamente alarrancar el sistema o arrancarlos de forma manual.

Concurrencia en Java. La clase ProcessBuilder y la clase Process

La clase Process es abstracta y por tanto, no se pueden crear objetos de ella, se pueden crear objetos de la clase Process con métodos de otras clases, en particular con objetos de la clase ProcessBuilder, con el método ProcessBuilder.start()

Métodos ya conocidos y probados de la clase Process

void destroy()

int exitvalue()

boolean isAlive() Comprueba si el proceso está vivo

long pid() Devuelve el PID o identificador del proceso

int waitfor()

InputStream getInputStream()

OutputStream getOutputStream()

InputStream getErrorStream()