MEDAC
Instituto Oficial de Formación Profesional

PROGRAMACIÓN DE SERVICIOS Y PROCESOS

TÉCNICO EN DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Tipos de programación

01

ÍNDICE

OBJETIVOS

Conocer el concepto de proceso.

Conocer el concepto de programa.

Diferenciar entre proceso y programa.

Conocer el concepto de programación concurrente.

Conocer el concepto de programación paralela.

Conocer el concepto de programación distribuida.

/ 1. Introducción y contextualización práctica

En esta unidad, adelantaremos una pequeña introducción de todo lo que veremos durante las siguientes unidades.

En primer lugar, veremos qué entorno de desarrollo vamos a utilizar y cómo podemos instalarlo.

Seguidamente, estudiaremos una serie de conceptos básicos en el mundo de la programación multiproceso, tales como proceso, programa, etc.

Por último, veremos los tres grandes tipos de programación que existen hoy en día: la programación concurrente, la paralela y la distribuida. Estudiaremos en qué consisten y cuáles son sus ventajas e inconvenientes.

Todos estos conceptos los repasaremos a lo largo de los temas restantes, así que no te preocupes, que tenemos mucho tiempo por delante.

Escucha el siguiente audio, en el que planteamos el caso práctico que iremos resolviendo a lo largo de esta unidad:

Fig. 1. Programando.

Audio Intro. "El Big Data y su complejidad"

https://bit.ly/32IRqO9

/ 2. Entorno de desarrollo: NetBeans 8.2

En esta asignatura, trabajaremos programando en Java, por lo que necesitaremos un entorno de desarrollo adecuado como NetBeans.

Ya conocimos **NetBeans** en otras asignaturas, así que sabemos que para poder programar en el lenguaje de programación Java necesitaremos tener instalada la JDK, que nos habilitará la máquina virtual de Java que nos permitirá programar y ejecutar programas escritos en este lenguaje.

La única diferencia de esta asignatura con respecto a las demás es la versión de NetBeans que vamos a utilizar: la 8.2.

El motivo por el que hemos elegido esta versión en concreto reside en que necesitaremos, en unidades posteriores, una serie de funcionalidades que esta versión tiene integradas (y las demás no), por lo que, para facilitar el trabajo con el entorno de desarrollo, usaremos la versión 8.2.

Para poder **descargar** NetBeans 8.2, lo puedes hacer desde el siguiente enlace:

https://netbeans.org/downloads/old/8.2/

Una vez hayamos accedido a la página de descarga, podremos ver que tenemos varias versiones de NetBeans 8.2.

NetBeans IDE Download Bundles

Supported technologies *	Java SE	Java EE	HTML5/JavaScript	PHP	C/C++	All
NetBeans Platform SDK	•	•				•
Java SE	•	•				•
Java FX	•	•				•
Java EE		•				•
Java ME						•
HTML5/JavaScript		•	•	•		•
PHP			•	•		•
C/C++					•	•
Groovy						•
Java Card™ 3 Connected						•
Bundled servers						
GlassFish Server Open Source Edition 4.1.1		•				•
Apache Tomcat 8.0.27		•				•

Fig. 2. Versiones de NetBeans 8.2.

La versión que nosotros vamos a utilizar es la denominada All, que inetgra todas las funcionalidades disponibles de NetBeans. Esta versión es la más pesada de todas, por lo que puede tardar un poco en descargarse.

Cabe destacar que, antes de instalar NetBeans, deberemos tener instalada la JDK. De lo contrario, el instalador del entorno de desarrollo nos mostrará un error y no lo podremos instalar.

Vídeo 1. "Instalación de NetBeans 8.2"
https://bit.ly/3hlpCmU

/ 3. Conceptos básicos I: Programa, proceso, servicio

Vamos a ver una serie de conceptos básicos que necesitaremos, no solo para esta unidad, sino también para entender mejor lo que veremos en las unidades posteriores.

Fig. 3. Analogía de procesos trabajando.

- **Programa:** Nos podemos referir a un programa como toda la información, tanto código como datos, almacenada en disco de una aplicación y que nos resolverá un problema concreto.

- **Proceso:** Un proceso se va a crear cuando ejecutamos un programa, por lo que podremos decir, de una forma muy simplificada, que un programa es, en cierto modo, un proceso. Así pues, podemos definir un proceso como un programa en ejecución. El concepto de proceso no se refiere solo al código y a los datos, sino que incluye todo lo necesario para que este se ejecute. Es muy importante tener en cuenta que un proceso es una entidad independiente de todo lo demás, aunque se ejecute en un mismo programa.

- **Servicios:** Los servicios son procesos que no son interactivos, pero que se están ejecutando continuamente. Son controlados por el sistema, sin ninguna intermediación por parte del usuario. Este tipo de procesos proporciona un servicio básico para el resto de procesos. El sistema operativo tiene sus propios servicios y los arrancará automáticamente en su inicialización. Los servicios van a pasar totalmente desapercibidos al usuario. Sin embargo, aunque se encuentren permanentemente en ejecución, se suele decir que se ejecutan en segundo plano (background). Cada proceso puede tener uno o varios servicios.

Audio 1. "Concepto de aplicación"
https://bit.ly/3jtS0W2

/ 4. Caso práctico 1: "¿Son todos los ejecutables iguales?"

Fig. 4. Ejecutando programas.

Planteamiento: Pilar y José están estudiando los conceptos básicos que van a necesitar para entender esta unidad, así como las posteriores.

Uno de estos conceptos es el de ejecutable.

«*Los ejecutables son los ficheros con los que podemos instalar las aplicaciones*», le comenta Pilar a José. Sin embargo, cuando utilizamos sistemas operativos diferentes, los ejecutables no son iguales: no se puede instalar un programa con un ejecutable de Windows en GNU/Linux, y viceversa.

Nudo: ¿Crees que todos los ejecutables son iguales? ¿Podrá haber ejecutables diferentes?

Desenlace: Todos hemos utilizado un ejecutable, alguna vez, cuando usamos el ordenador. En sistemas operativos del tipo Windows, estos son muy fáciles de reconocer, ya que su extensión es la más conocida de todas: .exe.

Si utilizamos otro tipo de sistemas operativos como GNU/Linux, los ficheros ejecutables son «diferentes», ya que no tienen la extensión .exe que tenían en los sistemas operativos Windows, sino que son ficheros que tienen permisos de ejecución y que no tienen ninguna extensión.

Esto nos puede llevar a pensar que hay diferentes tipos de ejecutables, lo cual es cierto. Dentro de este tipo de ficheros, podemos encontrar los binarios, que son ficheros que contienen código que va a ser ejecutado directamente por el procesador; los interpretados, que son ficheros que se ejecutarán en una máquina virtual; y, por último, las bibliotecas, que son un conjunto de funciones que pueden utilizadas por otros programas.

/ 5. Conceptos básicos II: Hilos, ejecutable, demonio

- **Hilos:** Los hilos, hebras, o **thread** son una tarea que podemos ejecutar en paralelo a otras, es decir: todos los hilos se estarán ejecutando al mismo tiempo, por lo que la tarea que ejecutan se podrá realizar mucho más rápido. Todos y cada uno de los hilos comparten una serie de recursos, como pueden el espacio de memoria, los archivos abiertos, los puertos para la comunicación en red, una base de datos, por ejemplo.

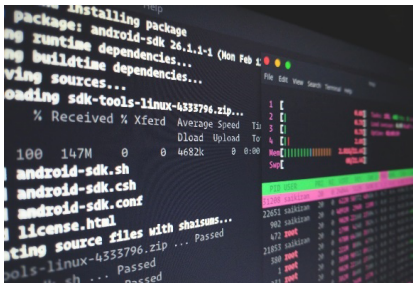

Fig. 5. Procesos en ejecución.

Todos los hilos que comparten los mismos recursos forman parte de un mismo proceso, lo que implica que cualquier hilo puede acceder a estos y modificarlos, pero hay que tener en cuenta que esta no es una tarea banal, ya que pueden producirse infinidad de errores. Hay que tener muy presente que, si un hilo modifica, por ejemplo, un dato en la memoria, los otros hilos posteriores a él ya accederán a ese dato modificado.

Un proceso que esté formado por varios hilos va a seguir en ejecución mientras alguno de sus hilos siga activo. Hasta que todos los hilos finalizan, no finaliza el proceso, y todos los recursos que estuviesen siendo utilizados son liberados. Un hilo siempre se va a ejecutar dentro del contexto de un proceso, que conoce como **programa padre.** Los programas que se ejecutan mediante varios hilos se denominan **flujo múltiple**, mientras que los de **flujo único** son los que tienen un solo hilo. La utilización de hilos nos va a permitir simplificar el diseño de aplicaciones que deben realizar varias funciones de forma simultánea.

- **Ejecutable:** Podemos definir un ejecutable como un fichero que contiene toda la información que será necesaria para crear un proceso partiendo de los datos almacenados de un programa, es decir, vamos a llamar ejecutable a todo aquel fichero que nos permita poner el programa en ejecución como un proceso.

- **Demonio:** Un demonio es un proceso no interactivo que se estará ejecutando continuamente en segundo plano, es decir, es un proceso que será controlado por el sistema operativo, sin ningún tipo de intermediación del usuario. Lo más común es que los demonios proporcionen un servicio básico para el resto de procesos como, por ejemplo, el control del tráfico de red que entra por la tarjeta de red.

/ 6. Programación concurrente

La programación concurrente es un tipo de programación que nos va a permitir tener en ejecución al mismo tiempo varias tareas que pueden ser interactivas. Esto es, este tipo de programación nos va permitir realizar varias tareas al mismo tiempo como, por ejemplo, escuchar música, ver la pantalla del ordenador, imprimir documentos, etc. Es impensable hacer esas tareas una a una y no simultáneamente. El tiempo que perderíamos si todas esas tareas se tuvieran que realizar una tras otra sería desmesurado. Además, proporciona mecanismos de comunicación y sincronización entre procesos.

Las tareas de la programación concurrente se pueden ejecutar en:

- **Multiprogramación (ejecución en un único procesador)**: Cuando hablamos de multiprogramación, aunque para el usuario que está frente al ordenador pueda parecer que se están ejecutando varios procesos al mismo tiempo (música, imprimir, escribir...), como solamente existe un único procesador, solamente se va a poder

estar ejecutando un único proceso en un momento determinado de tiempo. Para poder cambiar entre los diferentes procesos que se deben ejecutar, el sistema operativo se encargará de cambiar el proceso que está actualmente en ejecución cada cierto período de tiempo, en el orden de milisegundos. Todo esto permite que en un segundo se ejecuten múltiples procesos, creando la impresión en el usuario de que muchos programas se están ejecutando al mismo tiempo, cuando no es cierto. Este complejo proceso se denomina programación concurrente, la cual no mejora el tiempo de ejecución global de los programas, ya que todos estos se ejecutan intercambiándose unos por otros en el procesador, aunque sí que va a permitir que varios programas tengan la apariencia de que se ejecutan al mismo tiempo.

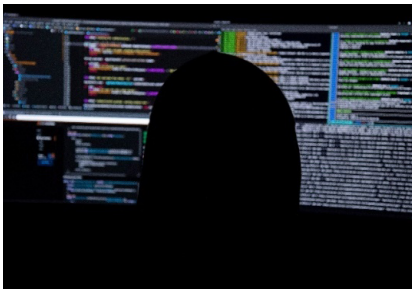

Fig. 6. Ejecutando varias tareas de forma concurrente.

- **Multitarea (varios núcleos en un mismo procesador)**: Cuando hablamos de multitarea, nos referimos a la existencia de varios núcleos en un procesador, apareciendo con esto los Dual Cores, Quad Cores, etc. Cada uno de estos núcleos podría estar ejecutando una instrucción diferente al mismo tiempo.

- **Programación paralela:** Este tipo de programación permite mejorar el rendimiento de un programa si este se ejecuta de forma paralela en diferentes núcleos, ya que permite que se ejecuten varias instrucciones a la vez. Profundizaremos sobre esto a continuación.

/ 7. Programación paralela

La programación paralela es un tipo de programación concurrente que se diseñó para ejecutarse únicamente en un sistema multiprocesador, es decir, con más de un núcleo en su microprocesador. Este tipo de programación permite la ejecución simultánea de una o varias tareas de un proceso.

La programación paralela nos va a permitir mejorar enormemente el rendimiento de nuestros programas, si estos se ejecutan en diferentes núcleos, puesto que cada una de las ejecuciones en cada núcleo será una tarea del mismo programa. Estas podrán cooperar entre sí, por lo que resuelven el problema muchísimo más rápido. La programación paralela también se conoce como **multitarea multihilo, multihebra o multithreading.**

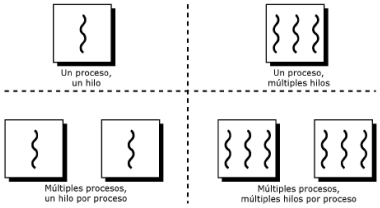

Fig. 7. Diferentes modelos de paralelismo.

La mayor ventaja que nos ofrece la programación paralela es que va a conseguir aumentar el rendimiento de nuestros programas, siempre y cuando sepamos implementar de forma correcta esta técnica, ya que podremos hacer que cada una de las tareas que creemos se ejecute en un procesador diferente.

Al igual que muchas otras técnicas, la programación paralela tiene una serie de inconvenientes. El mayor de ellos es la complejidad del diseño de sus algoritmos, concretamente, en la parte de comunicación de los distintos procesos o hilos. Aunque para resolver este problema tenemos varias opciones, como la utilización de semáforos, interbloqueos, algoritmos de exclusión mutua, etc., su implementación sigue siendo compleja.

Todos estos tipos de mecanismos los veremos en unidades posteriores.

En la actualidad, podemos encontrar las siguientes arquitecturas que utilizan programación paralela:

- **Sistemas multinúcleo:** Aquí podemos considerar a los microprocesadores actuales, ya que todos tienen varios núcleos.

- **Microprocesadores específicos:** Como procesadores gráficos, procesadores para videojuegos, procesadores embebidos, etc.

/ 8. Caso práctico 2: "¿Dónde aplicar los diferentes tipos de programación?"

Planteamiento: Nuestros amigos Pilar y José están estudiando los diferentes tipos de programación que existen. «No tenía ni idea de que hubiese tantos tipos de programación», le comenta José a Pilar, que le responde que ella tampoco tenía ni idea. «La programación no es solamente crear clases», piensa Pilar.

Ahora que Pilar y José conocen la existencia de estos tipos de programación se preguntan dónde se aplicarán...

Fig. 8. Videojuego.

Nudo: ¿Qué piensas al respecto? ¿Crees que los diferentes tipos de programaciones tienen aplicaciones reales en ámbitos, por ejemplo, que no sean científicos?

Desenlace: Es normal que, cuando estudiamos algo nuevo, nos preguntemos si se utiliza realmente o si es algo meramente teórico que forma parte de algún concepto, pero que no tiene aplicación real.

En el caso de los diferentes tipos de programación, pueden parecer conceptos que solo se van a utilizar en un marco teórico, sin aplicación práctica, pero nada más lejos de la realidad: sin este tipo de programaciones, hay tareas que no se podrían realizar. Pensemos en algo que nos gusta a casi todos: los videojuegos. En este tipo de programas, sin la programación paralela, no podríamos tener los resultados a los que estamos acostumbrados hoy en día, ya que, sin un correcto tratamiento de los procesos, no se podrían llevar a cabo los renderizados por parte de las tarjetas gráficas, la reproducción fluida de sonidos, etc.

/ 9. Programación distribuida

La programación distribuida se da en **sistemas distribuidos**, que son un conjunto de ordenadores interconectados que comparten un mismo estado, dando la impresión de que son un único sistema cuyo objetivo es compartir recursos. El sistema distribuido más conocido por todos es la red Internet.

La programación distribuida siempre ha estado enfocada a desarrollar software para sistemas distribuidos, abiertos, escalables, transparentes y muy tolerantes a fallos. Este tipo de programación utiliza la arquitectura hardware **cliente-servidor.**

En la actualidad, encontraremos sistemas donde se utiliza arquitectura y programación distribuida en los siguientes casos:

- **Redes:** En las redes de ámbito local, se conectan varios microprocesadores a través de una red de conexión de alta velocidad, lo cual va a formar lo que conocemos como clúster, que son varios ordenadores que funcionan como un único ordenador.

- **Supercomputadores:** Estos computadores son sistemas computacionales muy potentes y se utilizan para tareas que necesitan una enorme capacidad de cálculo, como militares, meteorología, simulación de moléculas, etc.

- **Grid Computing:** En este tipo de computación distribuida, van a poder usarse ordenadores muy potentes conectados en red entre sí.

- **Cloud Computing:** El cloud computing o cómputo en la red son sistemas donde podremos tener varios recursos (uno de los más conocidos es el espacio en disco). Las máquinas que ofrecen ese servicio pueden estar en otra parte del mundo y están interconectadas.

Vídeo 2. "Tipos de paralelismo"
https://bit.ly/39jSQQo

/ 10. Resumen y resolución del caso práctico de la unidad

A lo largo de esta unidad, hemos visto qué entorno de desarrollo utilizaremos: NetBeans en su versión 8 completa. Además, hemos aprendido cómo podemos instalarlo correctamente.

Lo siguiente que hemos expuesto ha sido una serie de conceptos básicos que necesitaremos durante esta y las futuras unidades, tales como proceso, programa, hilo, ejecutable, etc. Todos estos conceptos son muy importantes, ya que los utilizaremos a lo largo de todas las unidades.

A continuación, hemos estudiado 3 tipos de programaciones, definiendo en qué consisten y cuáles son sus ventajas e inconvenientes:

- La programación concurrente.
- La programación paralela.
- La programación distribuida.

Resolución del Caso práctico de la unidad

El Big Data es, sin duda, uno de los mayores retos que existe hoy en día. Estas enormes cantidades de datos deben ser procesados con mucho cuidado, ya que, si no tenemos precaución con los algoritmos que utilizamos para manejarlos, podemos no tener una solución en un tiempo razonable.

Fig. 9. Big data

Es aquí donde entran en juego las técnicas de programación paralela, distribuida y concurrente, que estudiamos en esta unidad. Es tal la diferencia cualitativa y de eficiencia que ofrecen estas técnicas, que existen problemas a los que no podrían solucionarse sin utilizarlas.

/ 11. Bibliografía

Concurrencia (informática), en Wikipedia, la enciclopedia libre. Recuperado de: https://es.wikipedia.org/wiki/Concurrencia_(inform%C3%A1tica)

Computación concurrente, en Wikipedia, la enciclopedia libre. Recuperado de: https://es.wikipedia.org/wiki/Computaci%C3%B3n_concurrente

Computación en la nube, en Wikipedia, la enciclopedia libre. Recuperado de: https://es.wikipedia.org/wiki/Computaci%C3%B3n_en_la_nube

Computación paralela, en Wikipedia, la enciclopedia libre. Recuperado de: https://es.wikipedia.org/wiki/Computaci%C3%B3n_paralela

Gómez, O. (2016). Programación de Servicios y Procesos Versión 2.1. Recuperado de: https://github.com/OscarMaestre/ServiciosProcesos/blob/master/build/latex/Servicios Procesos.pdf

Proceso (informática), en Wikipedia, la enciclopedia libre. Recuperado de: https://es.wikipedia.org/wiki/Proceso_(inform%C3%A1tica)

Sánchez, J. M. y Campos, A. S. (2014). Programación de servicios y procesos. Madrid: Alianza Editorial.

VV. AA. (s. f.). Programación Paralela. Recuperado el 7 de julio de 2020 de: http://ferestrepoca.github.io/paradigmas-de-programacion/paralela/paralela_teoria/index.html