Programación Concurrente y Paralelo

3er año – Informática

Ing. María del Milagro Paredes
Escuela de Minas

2020

PROCESAMIENTO SECUENCIAL

El procesamiento secuencial, tal como expresa la misma palabra, implica secuencialidad, temporalidad, dando sentido a las partes y no al todo. En su aplicación al campo educativo, el procesamiento secuencial se expresa en actividades como la de memorización de números y de palabras, la asociación entre letras y sus sonidos, el aprendizaje de las reglas gramaticales, la cronología de hechos históricos ,seguir paso a paso el método científico y entender las reglas de muchos juegos.

Se produce un procesamiento secuencial cuando una tarea va después de otra. Es un proceso lento en el que si una tarea se enlentece, el sistema completo debe esperar. La ventaja es que es fácil de entender y de implementar.

Ejemplos de Tareas secuenciales

- Leer una novela (obligatoriamente una página detrás de otra, si no, el resultado varía).
- Consultar el saldo, dar dinero, actualizar el saldo (en un cajero automático)
- Ver a alguien, reconocerle, preguntarle por su familia.

PROCESO CONCURRENTE

•Dos procesos son concurrentes cuando se ejecutan de manera que sus intervalos de ejecución se solapan.

Si hay concurrencia

No hay concurrencia

PROCESAMIENTO CONCURRENTE

Definición de concurrencia

- Dos o más procesos decimos que son concurrentes o que se ejecutan concurrentemente, cuando son procesados al mismo tiempo, es decir, que para ejecutar uno de ellos, no hace falta que se haya ejecutado otro.
- En un sistema multiprocesador, esta ejecución simultánea podría conseguirse completamente, puesto que podremos asignarle, por ejemplo, un proceso A al procesador A y un proceso B al procesador B y cada procesador realizaran la ejecución de su proceso.

PROCESAMIENTO CONCURRENTE

- Cuando tenemos un solo procesador se producirá un intercalado de las instrucciones de ambos procesos, de tal forma que tendremos la sensación de que hay un paralelismo en el sistema (concurrencia, ejecución simultánea de más de un proceso).
- Tenemos que tener en cuenta que mientras un proceso está escribiendo un valor en una variable determinada, puede darse el caso que otro proceso que es concurrente al primero vaya a leer o escribir en esa misma variable. Para estudiar esto, y determinar el tipo de operaciones que se pueden realizar sobre recursos compartidos utilizaremos las condiciones de Bernstein.
- Ejemplos:
  - ✓Acceso a disco por dos procesos que implica una gestión del SO. Dos procesos que modifican los registros de una Base de Datos.
  - ✓Compilador y Ensamblador se comunican y sincronizan para producir código maquina.

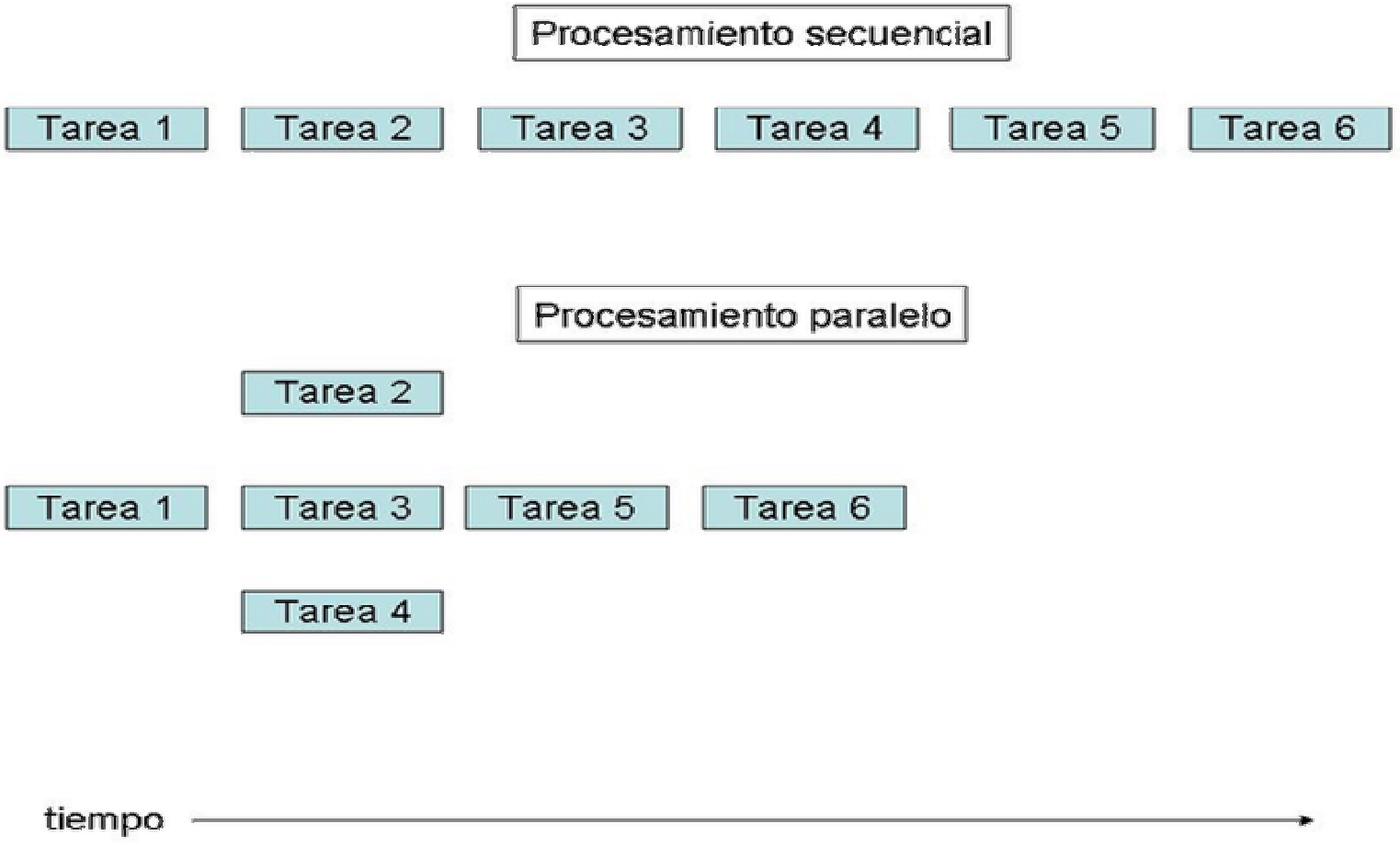
PROCESAMIENTO PARALELO

En el procesamiento paralelo, varios procesos ocurren al mismo tiempo mejorando el rendimiento general del sistema. Su contrapartida es la complejidad. No siempre se puede paralelizar una tarea. Aunque se pueda, no siempre es fácil hacerlo. En principio, cuanto más nodos actúen en paralelo, más rápido ira el sistema. Hasta cierto punto. Lo común es que haya un límite en que añadir más nodos no aumente la eficiencia, sino que incluso la reduzca. Normalmente las tareas en paralelo tienen una fase secuencial al principio en la que se dividen las tareas y otra al final en la que consolidan resultados. Los procesos paralelos operan independientemente unos de otros.

Ejemplos de Tareas paralelas

- Consultar entra varias personas palabras en la enciclopedia (personas en paralelo)
- Contar los votos en un proceso electoral (mesas en paralelo).
- Buscar los “López” en una base de datos. (se divide la base de datos en trozos y se busca en paralelo).
- Reconocer a una persona. Se analizan los rasgos faciales, el movimiento, los gestos, la voz, la entonación en paralelo.

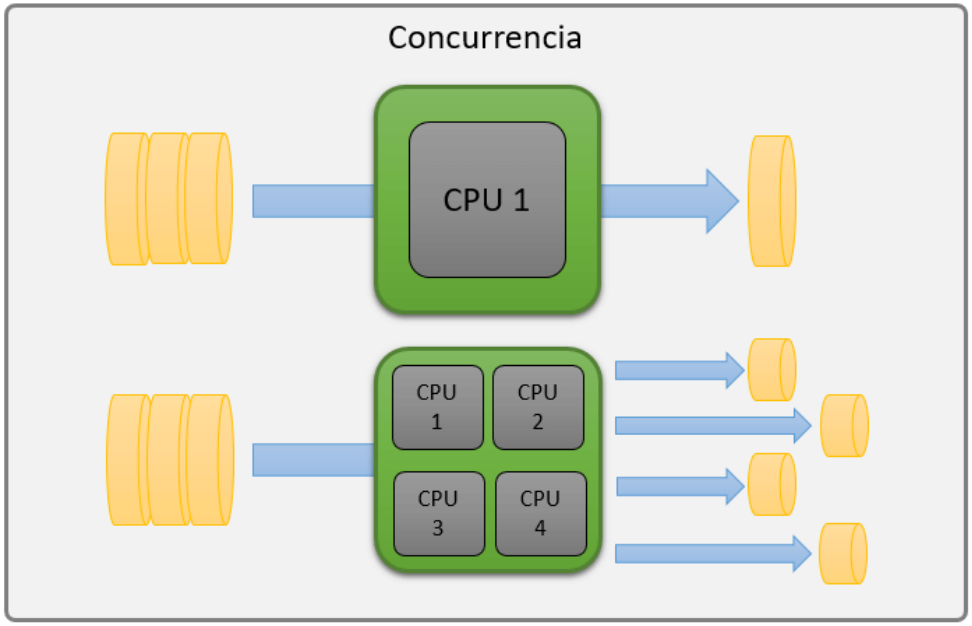
PROCESAMIENTO SECUENCIAL Y PARALELO



CONCURRENCIA

❖**La concurrencia es la capacidad del CPU para procesar más de un proceso al mismo tiempo**.

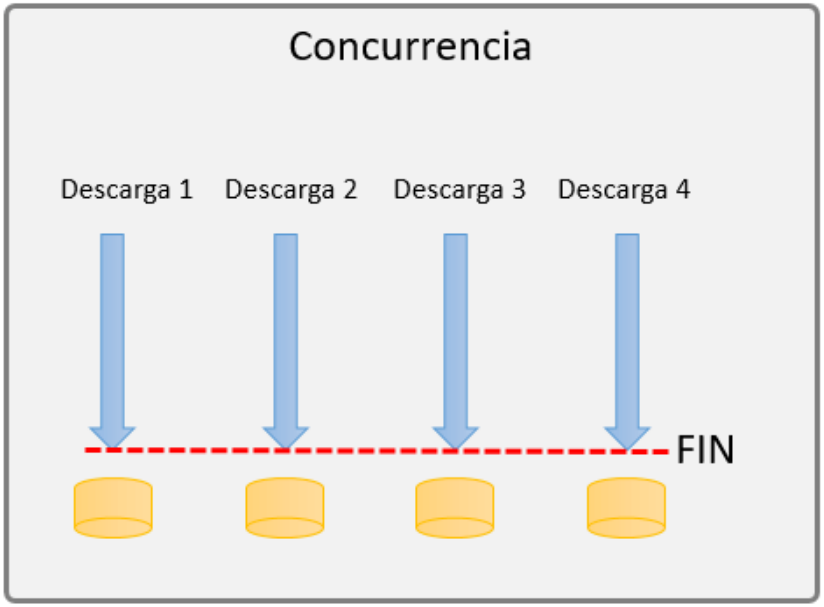
Un procesador puede procesar al mismo tiempo el mismo número de procesos que el número de CORES que tiene, de esta forma, si un procesador tiene un CORE, entonces solo podrá ejecutar un proceso a la vez, por otro parte, si tenemos 8 CORES, entonces podremos ejecutar hasta 8 procesos al mismo tiempo.



En la concurrencia, los procesos en ejecución no tienen por qué estar relacionados, es decir, cualquiera puede iniciar y terminar en el momento que sea, y el resultado de uno no afecta al otro.

EJEMPLO DE CONCURRENCIA

Imagina una aplicación de descarga de música, en la cual puedes descargar un número determinado de canciones al mismo tiempo, cada canción es independiente de la otra, por lo que la velocidad y el tiempo que tarde en descargarse cada una no afectara al resto de canciones. Esto lo podemos ver como un proceso concurrente, ya que cada descarga es un proceso totalmente independiente del resto.

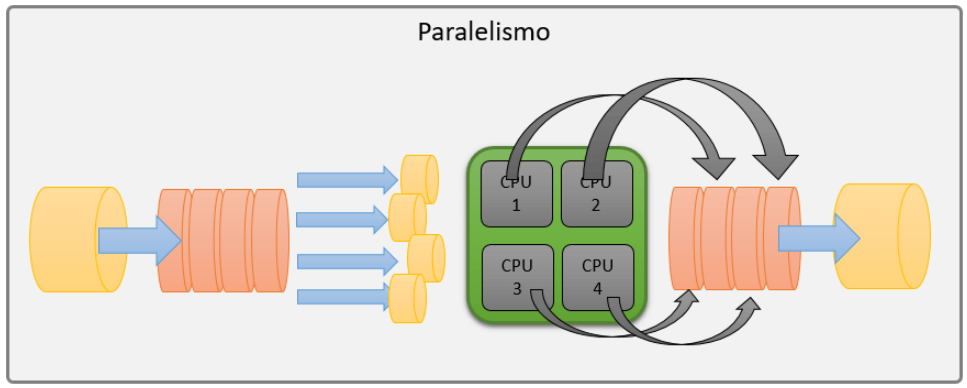


Observa la imagen, cada descarga se procesa de forma separada, y al final a una descarga no le importa el estado de las demás. Ya que cada descarga es una tarea completamente diferente.

PARALELISMO

El paralelismo sigue la filosofía de “divide y vencerás”, ya que consiste en **tomar un único problema, y mediante concurrencia llegar a una solución más rápido**. El paralelismo lo que hace es tomar el problema inicial, dividir el problema en fracciones más pequeñas, y luego cada fracción es procesada de forma concurrente, aprovechando al máximo la capacidad del procesador para resolver el problema. La principal diferencia del paralelismo contra la concurrencia es que, en el paralelismo, todos los procesos concurrentes están íntimamente relacionados a resolver el mismo problema, de tal forma que el resultado de los demás procesos afecta al resultado final.

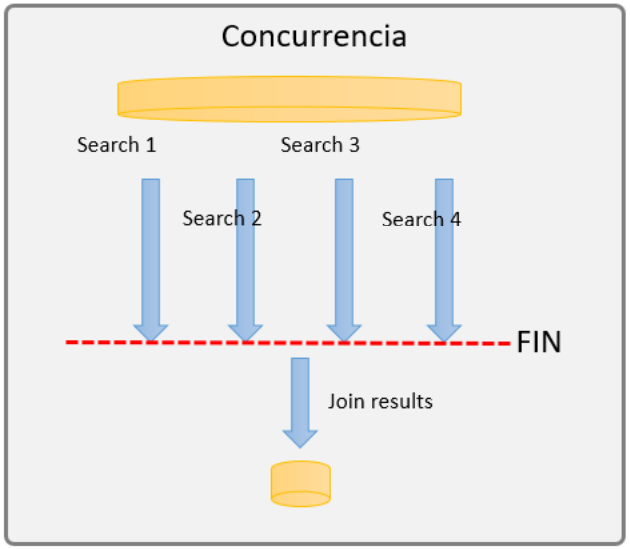
NOTA: En el paralelismo debe de haber un paso final que se encargue de unir los resultados de todos los procesos para poder arrojar un resultado final.



Primero que nada, llega un proceso grande (Amarillo), el cual es divididos en cuatro partes(Naranja), las cuales coinciden con el número de CORES del procesador(Verde). Ahora bien, por cada CORE disponible se lanza un proceso concurrente para resolver una fracción del problema. Cuando el proceso de un CORE termina, deja su resultado como una parte del resultado final (Naranja), así al terminar todos los procesos concurrentes, tenemos un resultado final (Amarillo).

EJEMPLO DE PARALELISMO

Imagina la clásica página de viajes, donde nos ayudan a buscar el vuelo más barato, para hacer esto, la página debe de buscar al momento en cada aerolínea el vuelo más barato, con menos conexiones, etc. Para esto puedo hacerlo de dos formas, buscar secuencialmente en cada aerolínea las mejores promociones (muy tardado) o utilizar el paralelismo para buscar al mismo tiempo las mejores promociones en todas las aerolíneas.



Observemos en la imagen como el proceso parte de una entrada inicial (inputs) los cuales definen las características del vuelo a buscar, luego se utiliza la concurrencia para buscar en las cuatro aerolíneas al mismo tiempo. Veamos que en este proceso es indispensable que las 4 búsquedas terminen para poder arrojar un resultado. Podemos ver claramente la relación entre los 4 procesos, ya que el resultado de uno puede afectar al proceso final. Observemos que una vez que los cuatro procesos terminan, hay un subproceso adicional encargado de unir los resultados y arrojar un resultado final.

CONCLUSIONES

Podrás observar que si bien, la concurrencia y el paralelismo está muy relacionados, no son la misma cosa, por lo tanto, la concurrencia ayuda a procesar varias tareas al mismo tiempo, mientras que el paralelismo resuelve una única tarea de una forma mucho más eficiente.

- Hay concurrencia entre varios procesos cuando existen al mismo tiempo. PROCESOS CONCURRENTES
- Hay paralelismo entre varios procesos cuando se ejecutan al mismo tiempo. PROCESOS PARALELOS

Hay diferencias entre **tareas/procesos concurrentes** y **tareas/procesos paralelos**. Cuál es la diferencia entre ambos?, es muy simple, hacer algo de forma concurrente, implica realizar un poco de cada tarea de forma tal que parezca que se realizan a la vez, este principio básicamente es el que usan los SO que sólo cuentan con un único procesador, tratan de administrar las tareas de forma tal que parezca que se ejecutan a la vez. En cambio, cuando hay más de un procesador presente, los SO modernos, se dan cuenta de esto, y en algunos casos, en donde es posible, ejecutan tareas en paralelo, es decir, realmente de forma simultánea.

CONCEPTO DE PROGRAMACIÓN CONCURRENTE

La **concurrencia** es "un suceso o concurso de varios procesos en un mismo tiempo".

1.2.1. Programa y proceso

-Un **programa** es un conjunto de instrucciones. Consiste en una secuencia de líneas de código que indican qué hacer con un conjunto de datos de entrada para producir algún tipo de salida. Se trata de algo **estático**. Para que un programa pueda hacer algo de verdad hay que ponerlo en ejecución.

-Un **proceso** es algo más que las líneas de código de un programa. Es una entidad **dinámica**. Está representado por el valor del contador del programa, contenido de los registros del procesador, una pila, y una sección de datos que contiene variables globales.

Puede haber múltiples procesos que corresponden al mismo programa.

En la figura 1.1, puede observarse un programa almacenado en disco y 3 instancias de ese programa ejecutándose. Son 3 procesos, cada uno con su propia información. Se trata de un primer nivel de concurrencia, donde existen 3 procesos independientes que se ejecutan al mismo tiempo sobre el SO.

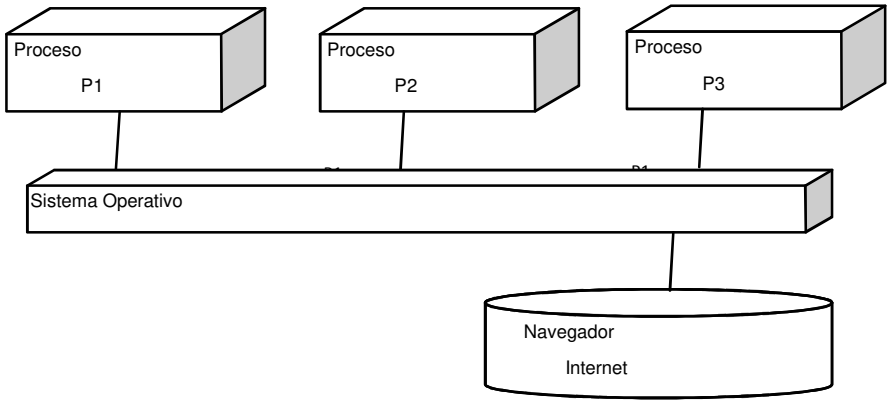


Figura 1.1 Programa y procesos

CONCEPTO DE PROGRAMACIÓN CONCURRENTE

1.2.2. Concurrencia

Dos **procesos** serán **concurrentes** cuando la primera instrucción de uno de ellos se ejecuta después de la primera instrucción del otro y antes de la última. Es decir, existe un solapamiento en la ejecución de sus instrucciones. No tienen por qué ejecutarse exactamente al mismo tiempo, es suficiente que exista un intercalado entre la ejecución de sus instrucciones. Si se ejecutan al mismo tiempo los dos procesos, tenemos una situación de **programación paralela**. La programación concurrente es un paralelismo potencial. Dependerá del hardware subyacente.

No necesariamente un proceso tiene por qué ser todo el programa en ejecución, sino que puede ser solo parte de él. Dicho de otra forma, un programa, al ponerse en ejecución, puede dar lugar a más de un proceso, cada uno de ellos ejecutando un parte del programa.

Con respecto al ejemplo, el programa de navegador de internet puede dar lugar a más de un proceso: uno que controle las acciones del usuario con la interfaz, otro que hace las peticiones al servidor, etc. De esta forma la figura 1.1 se convertiría en la figura 1.2 suponiendo que se crean 2 procesos cada vez que se ejecuta el programa. Todos estos procesos también pueden ejecutarse concurrentemente.

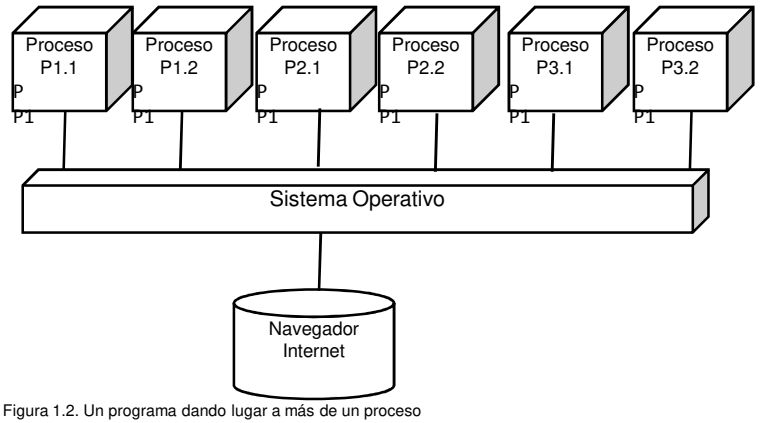


Figura 1.2. Un programa dando lugar a más de un proceso

Un poco de historia de la concurrencia

Evolución en respuesta a los cambios tecnológicos...

- De enfoques ad-hoc iniciales a técnicas generales de programación
- -60's.Evolución de los SO.

  -Controladores de dispositivos (canales) operando independientemente de un procesador permitiendo E/S .

  Multiprogramación. Problema de la sección crítica.
- -70's.Formalización de la concurrencia en los lenguajes
- -80's.Redes, procesamiento distribuido
- -90's.MPP, Internet, C/S, Web computing...

TIPOS DE CONCURRENCIA

•CONCURRENCIA APARENTE: Hay más procesos que procesadores.

- ✓Los procesos se multiplexan en el tiempo.
- ✓Pseudoparalelismo.

TIPOS DE CONCURRENCIA

•CONCURRENCIA REAL: Cada proceso se ejecuta en un procesador.

- ✓Se produce una ejecución en paralelo.
- ✓Paralelismo real.

CONCURRENCIA REAL Y APARENTE

- La concurrencia comprende un gran número de cuestiones de diseño, incluyendo la comunicación entre procesos y competencia por los recursos, sincronización de la ejecución de varios procesos y asignación del tiempo de procesador a los procesos y es fundamental para que existan diseños como Multiprogramación, Multiproceso y Proceso distribuido
  Los procesos son concurrentes si existen simultáneamente. Cuando dos o más procesos llegan al mismo tiempo a ejecutarse, se dice que se ha presentado una concurrencia de procesos. Es importante mencionar que para que dos o más procesos sean concurrentes, es necesario que tengan alguna relación entre ellos.

- En general, **la concurrencia será aparente** siempre que el número de procesos sea mayor que el de procesadores disponibles, es decir, cuando haya más de un proceso por procesador. **La concurrencia será real** cuando haya un proceso por procesador. Aunque puede parecer que la intercalación y la superposición de la ejecución de procesos presentan formas de ejecución distintas, se verá que ambas pueden contemplase como ejemplos de procesos concurrentes

MODELOS DE PROGRAMACIÓN CONCURRENTE

❑MULTIPROGRAMACIÓN CON UN ÚNICO PROCESADOR

-El sistema operativo se encarga de ir repartiendo el tiempo del procesador entre los distintos procesos, intercalando la ejecución de los mismos para dar así una apariencia de ejecución simultánea. *Ejecución de múltiples procesos en un solo procesador.*

❑MULTIPROCESADOR

-Es una maquina formada por un conjunto de procesadores que comparten memoria principal. En este tipo de arquitecturas, los procesos concurrentes no sólo pueden intercalar su ejecución sino también superponerla. Se combinan paralelismo real y pseudoparalelismo. Normalmente más procesos que CPU's. *Ejecución de múltiples procesos en un sistema multiprocesador donde hay acceso a memoria compartida,* es decir que cada proceso corre en su propio procesador pero con memoria compartida.

❑SISTEMA DISTRIBUIDO

-Varios computadores conectados por red. En este tipo de arquitecturas también es posible la ejecución simultánea de los procesos sobre los diferentes procesadores. *Ejecución de múltiples procesos en varios procesadores los cuales no comparten memoria.* Es decir que cada proceso corre en su propio procesador conectado a los otros a través de una red.

TIPOS DE PROCESOS CONCURRENTES

❑Independientes

– Procesos que se ejecutan concurrentemente pero sin ninguna relación.
  - No necesitan comunicarse.
  - No necesitan sincronizarse.
  - Ejemplo: Dos intérpretes de mandatos de dos usuarios ejecutados en distintos terminales.

❑Cooperantes

– Procesos que se ejecutan concurrentemente con alguna interacción entre ellos.
  - Pueden comunicarse entre si.
  - Pueden sincronizarse.
  - Ejemplo: Servidor de transacciones organizado en proceso receptor y procesos de tratamiento de peticiones.

INTERACCIONES ENTRE PROCESOS

- **Acceso a recursos compartidos**
  - Procesos que comparten un recurso.
  - Procesos que compiten por un recurso.
  - Ejemplo: Servidor de peticiones en la que distintos procesos escriben en un registro de actividad.
- **Comunicación**
  - Procesos que intercambian información.
  - Ejemplo: Receptor de peticiones debe pasar información a proceso de tratamiento de petición.
- **Sincronización**
  - Un proceso debe esperar a un evento en otro proceso.
  - Ejemplo: Un proceso de presentación debe esperar a que todos los procesos de cálculo terminen.

La concurrencia aparece cuando dos o más procesos son contemporáneos. La concurrencia es la capacidad de ejecutar múltiples actividades en paralelo o simultáneamente.

Los elementos que constituyen un programa concurrente son módulos independientes denominados tareas o procesos. La programación concurrente se encarga fundamentalmente de temas vinculados a la comunicación y sincronización de procesos.

Los procesos pueden **"competir"** o **colaborar** entre sí por los recursos del sistema.

Cuando varios procesos se ejecutan concurrentemente puede haber procesos que colaboren para un determinado fin y otros que compitan por los recursos del sistema. Incluso aquellos procesos que colaboran deberán competir a la hora de obtener tiempo de procesador.

Para realizar las tareas de colaboración y competencia por los recursos se hace necesaria la introducción de mecanismos de comunicación y sincronización entre procesos.

La comunicación permite que los procesos cooperen entre sí en la ejecución de un objetivo global, en tanto que la **sincronización** permite que un proceso continúe su ejecución después de que un determinado evento ha ocurrido.

El paralelismo implica que existen varios procesadores en el sistema. La programación paralela implica dividir la ejecución de un programa en distintos módulos los cuales se ejecutaran en distintos procesadores.

La comunicación entre tareas o procesos se puede realizar básicamente de dos maneras:

a) comunicación a través de un área común de memoria, o
b) comunicación mediante el intercambio de mensajes.

En el caso a) es necesario contar con mecanismos de sincronización para garantizar la consistencia de los datos almacenados como por ejemplo semáforos, monitores, etc.

En el caso b) las variables son locales. Un mensaje al llegar a su destino sólo es entregado a su tarea cuando ésta lo requiere. Si lo requiere y aún no está disponible la tarea debe suspenderse y esperar la llegada del mensaje.

Lógicamente los lenguajes necesarios para lograr estos objetivos deben:
- permitir multitareas,
- manejar sentencias del tipo SIGNAL, WAIT, DELAY, etc (es deseable),
- ser de tipo modular,

Dónde encontramos concurrencia?

- Los sistemas biológicos suelen ser masivamente concurrentes: comprenden un gran número de células, evolucionando simultáneamente y realizando (independientemente) sus procesos. En el mundo biológico los sistemas secuenciales rara vez se encuentran.
- Todas las entidades, sean plantas, animales, máquinas, etc, ejecutan en paralelo unas con otras, cualquier sistema más o menos "inteligente" exhibe concurrencia…
- Navegador Web accediendo una página mientras atiende al user, Varios navegadores accediendo a la misma página
- Acceso de varias aplicaciones a disco para guardar
- El teléfono avisa recepción de llamada mientras se habla
- Varios usuarios conectados al mismo sistema (x ej, haciendo una reserva)
- En algunos casos se tiende a pensar en sistemas secuenciales en lugar de concurrentes para simplificar el proceso de diseño, pero la necesidad de sistemas de cómputo cada vez más poderosos y flexibles atenta contra la simplificación de asunciones de secuencialidad
- "Concurrency is everywhere"….concurrencia está en todas partes

Beneficios de la Programación Concurrente

- Facilita la programación de aplicaciones al permitir que éstas se estructuren como un conjunto de procesos que cooperan entre sí para alcanzar un objetivo común.
- *Acelera los cálculos*. Si se quiere que una tarea se ejecute con mayor rapidez, lo que se puede hacer es dividirla en procesos, cada uno de los cuales se ejecuta en paralelo con los demás.
- Posibilita el uso interactivo a múltiples usuarios que trabajan de forma simultánea.
- Permite un mejor aprovechamiento de los recursos, en especial de la CPU, ya que pueden aprovechar las fases de entrada-salida de unos procesos para realizar las fases de procesamiento de otros.
- Mejor utilización de la CPU de cada procesador.
- **La velocidad de ejecución**. Cuando la puesta en ejecución de un programa conlleva la creación de varios procesos y el sistema consta de más de un procesador, existe la posibilidad de asignar un proceso a cada procesador, de tal forma que el programa se ejecuta más rápido. Los programas de cálculo son los más beneficiados de este hecho.
- **Solución a problemas** de naturaleza concurrente.

Beneficios de la Programación Concurrente

➢ Solución a problemas de naturaleza concurrente:

a) Sistemas de control:

Son aquellos en los que hay una captura de datos, generalmente con sensores, un análisis de esos datos y una posible actuación posterior en función del resultado del análisis. La recolección puede hacerse en diferentes entidades físicas. Tanto la captura de datos desde cada entidad, su tratamiento y actuación son candidatos a ser procesos distintos de naturaleza concurrente. Lo que garantiza que el sistema de control pueda responder a alarmas que se produzcan.

b) Tecnologías web:

La mayoría de los programas relacionados con la web son concurrentes. Los servidores web son capaces de atender concurrentemente múltiples conexiones de usuarios, programas de chat (que permitan conversación de varios usuarios); los servidores de correo (permiten que múltiples usuarios puedan enviar y recibir mensajes al mismo tiempo), navegadores que permiten que usuarios puedan descargar un archivo mientras navegan, ejecutan un applet de java, etc.

c) Aplicaciones basadas en interfaces de usuarios:

La concurrencia va a permitir que el usuario pueda interactuar con la aplicación aunque esté realizando alguna tarea que consume mucho tiempo de procesador. Un proceso controla la interfaz, mientras que otro hace las tareas que requieren un uso intensivo de CPU. Facilitando que tareas largas puedan se abortadas con facilidad.

d) Simulación:

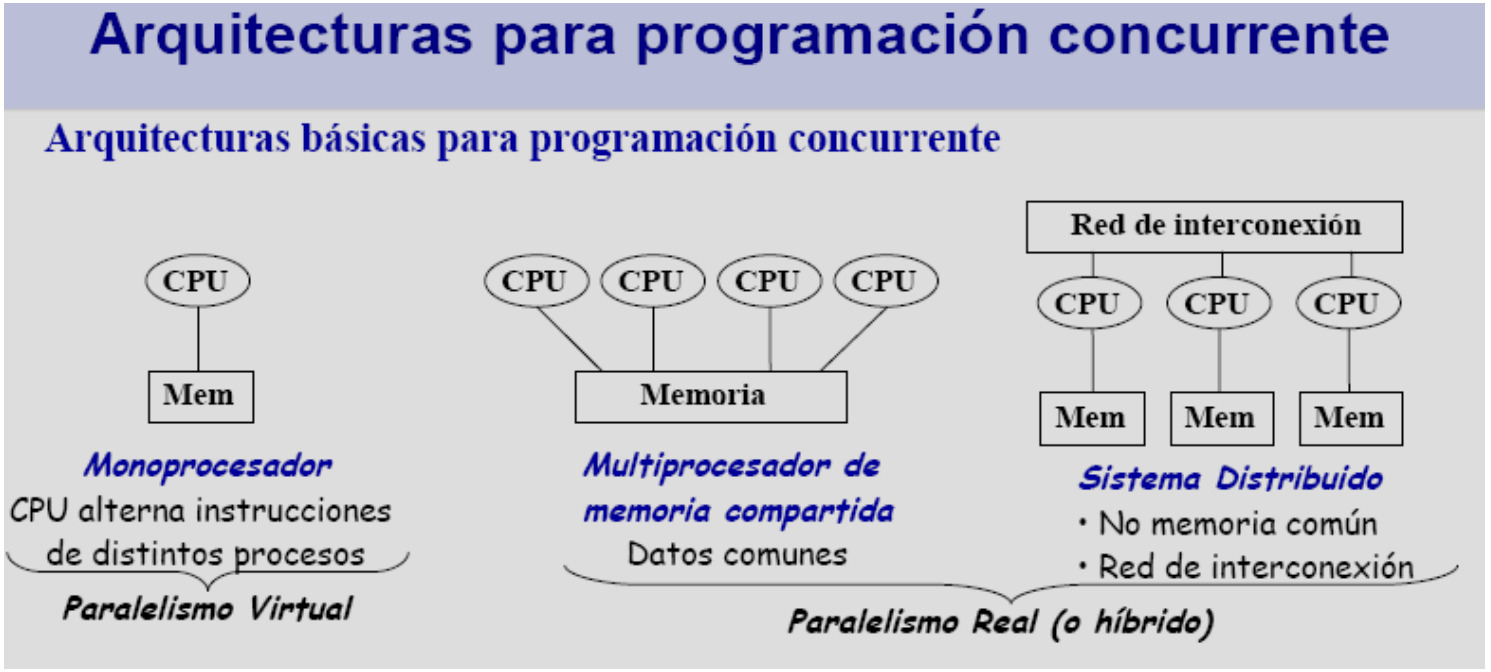
La concurrencia va a permitir modelar objetos físicos que tienen un comportamiento autónomo independiente y ponerlos en ejecución de forma independiente y concurrente sincronizados apropiadamente.

e) Sistema de Gestión de Base de Datos (SGBD):

En estos sistemas la concurrencia va a ser muy importante cuando se va a permitir a varios usuarios interactuar con el sistema. Cada usuario puede ser visto como un proceso. Se debe evitar que varios usuarios modifiquen el mismo registro en el mismo tiempo, y permitir el acceso concurrente para que varios usuarios puedan consultar un mismo instante un registro sin modificarlo.

CONCURRENCIA Y ARQUITECTURA HARDWARE

-Cuando hablamos del hardware en el que se van a ejecutar los procesos concurrentes, nos referimos a el número de procesadores en el sistema. Se puede entonces hacer una primera distinción: Sistema o Arquitectura Monoprocesador y Sistema o Arquitectura Multiprocesador, en ambos es posible tener concurrencia.



ARQUITECTURA MONOPROCESADOR

Los sistemas operativos mono programados son aquellos que solo son capaces de ejecutar una instrucción a la vez. Son máquinas paralelas a nivel de instrucción, que se caracterizan por tener un único flujo de instrucciones sobre un único flujo de datos. En cualquier momento solo se ejecuta una única instrucción.

En un sistema Monoprocesador podemos tener concurrencia, gestionando el tiempo de procesador para cada proceso.

ARQUITECTURA MONOPROCESADOR

Evidentemente no todos los procesos pueden ejecutarse al mismo tiempo en el procesador, pero el SO va alternando el tiempo de procesador entre los procesos, dando a los usuarios sensación de ejecutarse simultáneamente. De esta forma, cuando un proceso que está ocupando el procesador necesita hacer una operación de entrada salida, puede abandonar el procesador para que otro pueda seguir su ejecución.

Existen otros posibles beneficios del uso de concurrencia en sistemas monoprocesador:

- La posibilidad de proporcionar un servicio interactivo a múltiples usuarios. Ejem: sistema operativo multiusuario ejecutándose sobre una maquina monoprocesador.
- La posibilidad de dar solución adecuada a problemas que son de naturaleza concurrente.

En un sistema monoprocesador los procesos comparten la misma memoria.
Los procesos se sincronizan y comunican mediante variables compartidas.

Como se ve en la figura, cuando un proceso A quiere comunicarle algo al proceso B, lo deja en una variable compartida y conocida por ambos para q B lo pueda leer

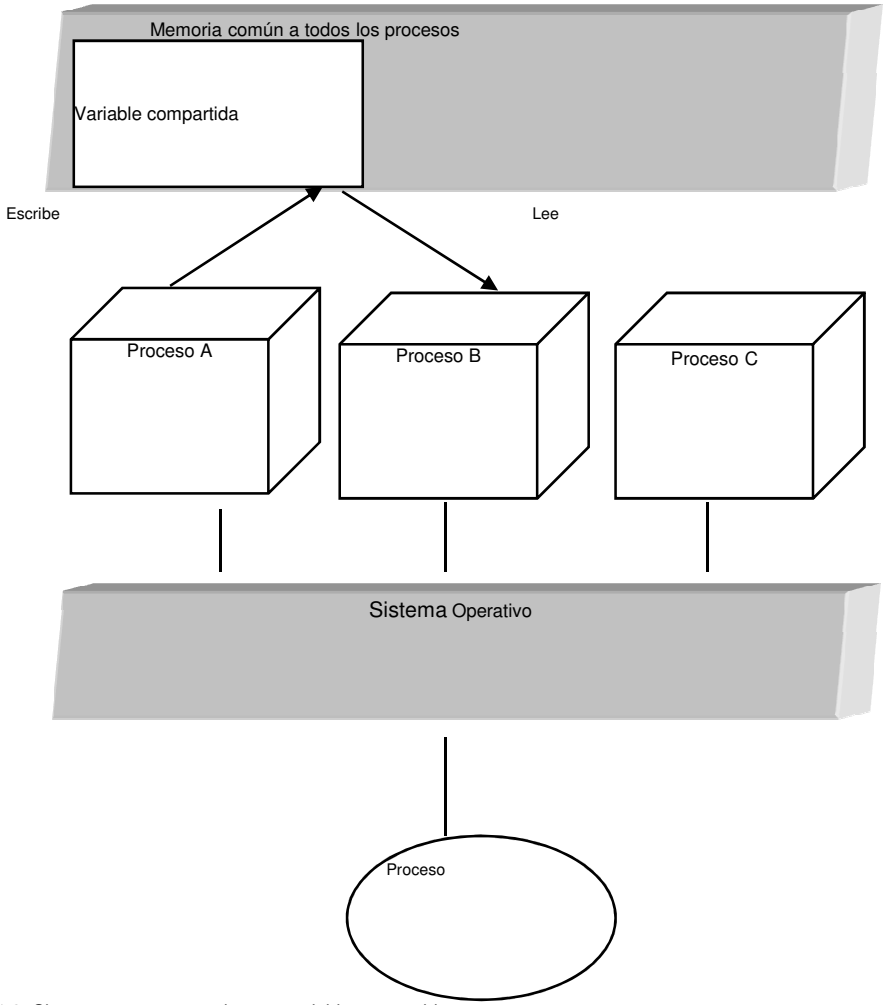


Figura 1.3, Sistema monoprocesador con variable compartida

ARQUITECTURA MULTIPROCESADOR

El concepto de trabajo en equipo, el dividir una tarea entre varias unidades de ejecución para que pueda completarse más rápidamente, fue naturalmente llevado al mundo de las computadoras prácticamente desde sus inicios.

Un sistema multiproceso o multitarea es aquel que permite ejecutar varios procesos de forma concurrente. En un Sistema multiprocesador consiste en un proceso en cada procesador. Éstos pueden ser de memoria compartida (fuertemente acoplados) o con memoria local a cada procesador (débilmente acoplados)

ARQUITECTURA MULTIPROCESADOR

Existen más de un procesador, permite paralelismo real entre procesos. Como en un sistema concurrente hay más procesos que procesadores, es necesario algún esquema de multiprogramación en uno o más procesadores.

Los sistemas multiprocesador se clasifican en:

- **Sistemas fuertemente acoplados:** los procesadores y dispositivos (incluida la memoria) están conectados a un bus. Permitiendo que todos los procesadores puedan compartir la misma memoria. Puede que cada procesador tenga su propia memoria local, pero la sincronización y comunicación se realiza mediante variables situadas en memoria compartida (figura 1.4)

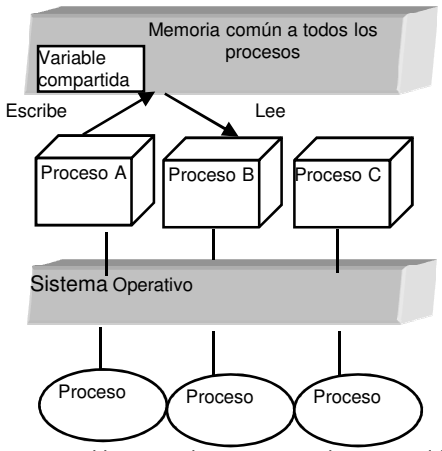


Figura 1.4. Sistema multiprocesador con memoria compartida

ARQUITECTURA MULTIPROCESADOR

- **Sistemas débilmente acoplados:**cada procesador tiene su memoria local y está conectado con los otros procesadores mediante algún enlace de comunicación. Ejem: sistemas distribuidos. Los nodos están distribuidos geográficamente, conectados de alguna forma. Cada nodo puede ser mono o multiprocesador (internet).

Multiproceso es la gestión de varios procesos dentro de un sistema multiprocesador, donde cada procesador puede acceder a una memoria común, y procesamiento distribuido es la gestión de varios procesos en procesadores separados cada uno con su memoria local. Figura 1.5

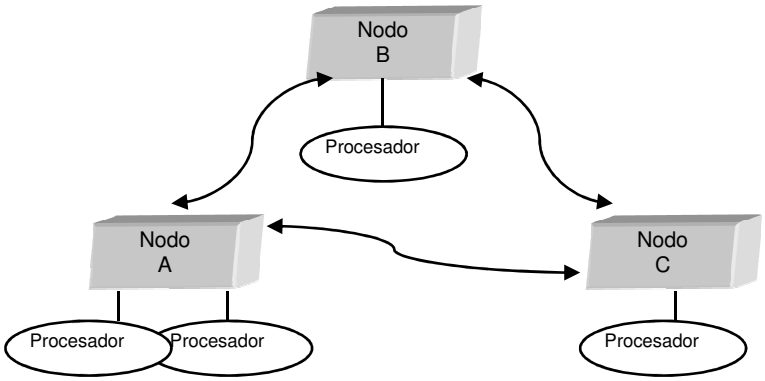
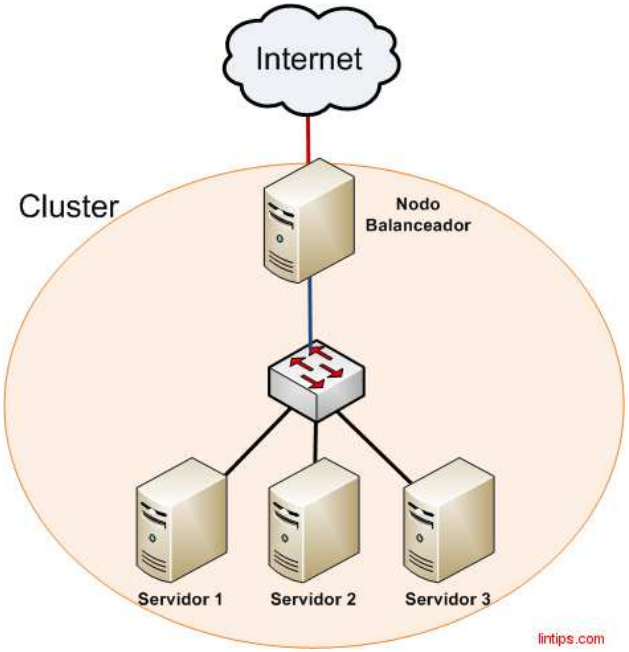


Figura 1.5. Sistema Distribuido

- Hay procesamiento masivo de datos distribuidos, SDTR, computación móvil, Cluster y multicluster computing, sistemas colaborativos, computación ubicua: es entendida como la integración de la informática en el entorno de la persona, de forma que los ordenadores no se perciban como objetos diferenciados, La computación ubicua es un término que estaremos escuchando cada vez más y se nos presentará en un panorama en el que millones de dispositivos inteligentes controlarán nuestro entorno. Como ejemplos tenemos el asistir a una junta de trabajo sin una laptop para hacer una presentación, basta con un smartphone: o ya no hacer fila en la caja del supermercado, porque con tu teléfono celular pagarás en línea,

ARQUITECTURA MULTIPROCESADOR DISTRIBUIDA CLUSTERS

- **Un clúster** es un grupo de múltiples ordenadores unidos mediante una red de alta velocidad, de tal forma que el conjunto es visto como un único ordenador, más potente que los comunes de escritorio.
- Consta de un conjunto de computadoras independientes, interconectadas entre sí, de tal manera que funcionan como un solo recurso computacional. A cada uno de los elementos del clúster se le conoce como nodo.
- Los clústeres son usualmente empleados para mejorar el rendimiento y/o la disponibilidad por encima de la que es provista por un solo computador típicamente siendo más económico que computadores individuales de rapidez y disponibilidad comparables.

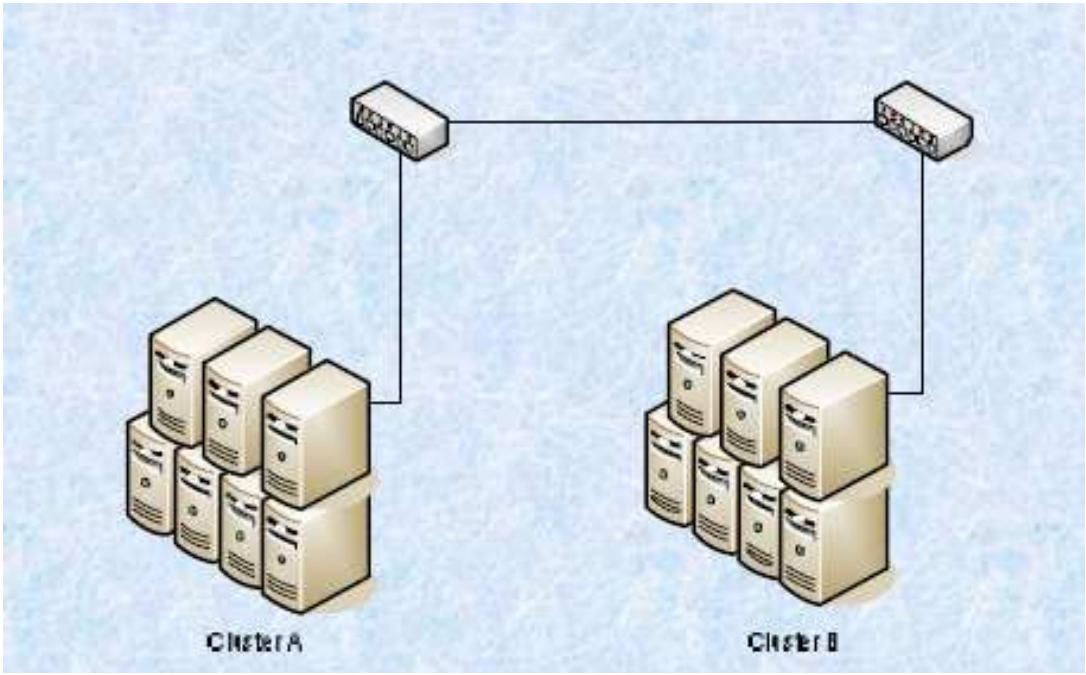


ARQUITECTURA MULTIPROCESADOR DISTRIBUIDA CLUSTERS

- Un **clúster** es un sistema distribuido compuesto por una colección de sistemas autónomos interconectados mediante redes privadas y fuertemente enlazados, que es empleado como un recurso computacional unificado.
- El clúster debe cumplir con una serie de requerimientos para conformarse como tal: Alta disponibilidad, escalable, administrable, que proporcione calidad de servicio, que pueda balancear cargas y que sea seguro.
- El diseño e implementación de un clúster de alta disponibilidad es una labor compleja, que requiere tanto de **software específico** como de una **arquitectura hardware singular**. La configuración del software del clúster no es fácil, ya que entran en juego muchos factores.
- El servidor principal es considerado como **nodo maestro** y los clientes o **nodos esclavos** deben tener los mismos SO e idénticas aplicaciones instaladas. Además tienen que poder acceder a los mismos ficheros de datos, que deben ser actualizados en tiempo real para que no se produzcan errores de concordancia, y asegurar la conmutación automática entre ambos nodos en caso de que se produzca un fallo en el hardware de alguno de ellos.
- Los sistemas basados en clúster se implementan con el objetivo de mejorar la disponibilidad de los servicios que se ofrecen.

CONCEPTOS CLUSTER CONCEPTOS CLUSTER - MULTICLUSTER

- Un **clúster** consiste de un conjunto de computadoras conectadas en red que se comportan como si fuesen una única computadora.
- Varios clúster conectados forman una arquitectura **multicluster**.



ARQUITECTURA MULTIPROCESADOR DISTRIBUIDA GRID

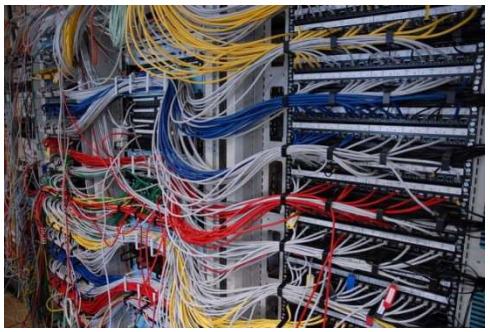
- La **arquitectura grid** es conceptualmente similar a la cluster, en determinados tipos de infraestructura y su utilización. Está basado en la utilización de conjuntos de ordenadores y estaciones conectadas a la red, aprovechando los tiempos muertos de procesamiento y los elementos ociosos, que se pueden re-aprovechar, uniendo el poder de proceso de muchos participantes.
- Se pueden procesar grandes operaciones de cálculo aprovechando la parte libre de cada elemento implicado en el grid, que mediante un software distribuido en cada nodo, participa en el proceso general.
- Se puede definir como un sistema de **procesamiento distribuido** que permite compartir recursos no unidos geográficamente para resolver problemas de gran escala. Los recursos compartidos pueden ser todos aquellos que puedan aportar los elementos necesarios (procesador, memoria, almacenamiento) para el fin general.
- Una infraestructura de este tipo implica muchos beneficios, como recursos balanceables, alta disponibilidad casi asegurada y reducción de costes sobre grandes proyectos.
- El **grid** puede ser de diferentes tamaños, desde una pequeño grupo de equipos a varias grandes redes interconectadas o miles de nodos trabajando en paralelo. Esta tecnología se aplica a muchas aplicaciones diferentes, como grandes cálculos matemáticos y científicos: análisis estructural, grandes servicios Web o investigación de marketing. Existen varias compañías que diseñan, planifican, lanzan e implementan **entornos y proyectos grid** a nivel mundial, respaldados por la mayor red que existe (internet) y basados en el protocolo IP. Estas pueden ser desde grandes corporaciones y universidades hasta grandes grupos de trabajo.

La **computación grid** es una tecnología innovadora que permite utilizar de forma coordinada todo tipo de recursos (cómputo, almacenamiento y aplicaciones específicas) que no están sujetos a un control centralizado. En este sentido es una nueva forma de computación distribuida, en la cual los recursos pueden ser heterogéneos (diferentes arquitecturas, supercomputadores, clusters) y se encuentran conectados mediante redes de área extensa (por ejemplo Internet).

Llamamos grid al sistema de computación distribuido que permite compartir recursos no centrados geográficamente para resolver problemas de gran escala. Los recursos compartidos pueden ser ordenadores (PC, estaciones de trabajo, supercomputadoras, PDA, portátiles, móviles, etc), software, datos e información, instrumentos especiales (radio,telescopios, etc.)
o personas/colaboradores.

El término *grid* se refiere a una infraestructura que permite la integración y el uso colectivo de ordenadores de alto rendimiento, redes y BD que son propiedad y están administrados por diferentes instituciones. Puesto que la colaboración entre instituciones envuelve un intercambio de datos, o de tiempo de computación, el propósito del *grid* es facilitar la integración de recursos computacionales. Universidades, laboratorios de investigación o empresas se asocian para formar *grid* para lo cual utilizan algún tipo de software que implemente este concepto.

En la computación *grid*, las redes pueden ser vistas como una forma de computación distribuida donde un "supercomputador virtual" está compuesto por una serie de computadores agrupados para realizar grandes tareas.



El *cloud computing* consiste en la posibilidad de ofrecer servicios a través de Internet.

La computación en nube es una tecnología nueva que busca tener todos nuestros archivos e información en Internet y sin depender de poseer la capacidad suficiente para almacenar información.

El cloud computing explica las nuevas posibilidades de forma de negocio actual, ofreciendo servicios a través de Internet, conocidos como e-business (negocios por Internet). Toda la información, procesos, datos, etc. se localizan dentro de la red de internet, como en una nube, así todo el mundo puede acceder a la información completa, sin poseer una gran infraestructura.

DIFERENCIAS EN TRE CLOUD, GRID Y CLUSTER

- Clúster difiere de Cloud and Grid en que un clúster es un grupo de computadoras conectadas por una red de área local (LAN), mientras que la nube (Cloud) y la cuadricula (Grid) son de mayor escala y pueden distribuirse geográficamente.
- Un clúster está estrechamente acoplado, mientras que un Grid o un Cloud están ligeramente acoplados.
- Además, los clústeres se componen de máquinas con hardware similar, mientras que las nubes y las cuadrículas se componen de máquinas con configuraciones de hardware posiblemente muy diferentes.

La diferencia entre una nube y una cuadricula se puede expresar de la siguiente manera:

-Distribución de recursos: Cloud computing es un modelo centralizado mientras que la computación grid es un modelo descentralizado en el que el cálculo podría ocurrir sobre muchos dominios administrativos.

-Propiedad: Una cuadrícula es una colección de computadoras que es propiedad de varias partes en múltiples ubicaciones y conectadas entre sí para que los usuarios puedan compartir la potencia combinada de los recursos. Mientras que una nube es una colección de computadoras generalmente propiedad de una sola parte.

Resumen de conceptos

- La Concurrencia es un concepto de software.
- La Programación Paralela se asocia con la ejecución concurrente en múltiples procesadores que pueden tener memoria compartida, y generalmente con un objetivo de incrementar performance.
- La Programación Distribuida es un "caso" de concurrencia con múltiples procesadores y sin memoria compartida.
- En Programación Concurrente la organización de procesos y procesadores constituyen la arquitectura del sistema concurrente.
- **"Especificar la concurrencia es esencialmente especificar los procesos concurrentes, su comunicación y sincronización."**

- **Objetivos de los sistemas concurrentes:**

- Ajustar el modelo de arquitectura de hardware y software al problema del mundo real a resolver. El mundo real ES CONCURRENTE.

- Incrementar la performance, mejorando los tiempos de respuesta de los sistemas de procesamiento de datos, a través de un enfoque diferente de la arquitectura física y lógica de las soluciones.

FIN TEORIA 1