Universidad Autónoma de Nuevo León

Facultada de Ingeniería Mecánica y Eléctrica

Sistemas Operativos

Actividad Fundamental #1

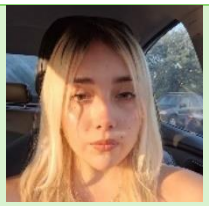
"Multitarea: Control de concurrencia"

Docente: DR. Norma Edith Marín Martínez

Grupo: 010 Hora: M4

Equipo: 2

Foto	Matricula	Nombre	Carrera	Aportación
	1948932	Antonio Enrique Hernández Ramírez	ITS	100
	2005930	Eden Leonardo Candelas Andrade	ITS	100
	2022830	Daniel Alejandro Segura Vázquez	ITS	100
	2045231	Denilson Gustavo Aguilar Puente	IAS	100
	2052523	Jorge Paz Villarreal	IAS	100

	2131973	Uriel Ramiro De La Fuente Del Ángel	ITS	100
	196213	Alexis Yahir Soria Salazar	IAS	100|
	1958098	Alan Jahir Rivas Urbina	ITS	100
	2052193	Sofia Giovanna Espinoza Zapata	IAS	100

Semestre Agosto – Diciembre 2024

San Nicolas De Los Garza, Nuevo León a 01 de septiembre de 2024

Contenido

Introducción

En esta investigación nosotros como equipo lo que haremos será llevar a cabo una investigación en la cual hablaremos básicamente acerca de la concurrencia en los sistemas operativos, aprenderemos acerca de sus conceptos básicos y todo lo que lo rodea. Nosotros como equipo ya sabemos de antemano que la concurrencia es un conjunto de cosas ocurriendo al mismo tiempo, pero ¿cómo podría aplicarse la concurrencia a los sistemas operativos? ¿Cuáles son los tipos de concurrencia que esta trae consigo? ¿Qué es lo que necesitamos saber acerca de la programación concurrente y las características que tiene?

Todas estas incógnitas serán respondidas y llevadas a cabo en esta investigación, en la cual hablaremos de la concurrencia, los tipos de concurrencia que esta tiene, también hablaremos acerca de algunos modelos de la programación concurrente junto con sus características, dichos modelos son el modelo de hilos, el modelo de procesos, el modelo de memoria compartida, el modelo de paso de mensajes, el modelo de programación basada en eventos, el modelo de futuro o promesa y por último el modelo de transacciones en memoria. Junto con estos modelos también llevaremos a cabo un análisis para poder analizar las ventajas y desventajas que traen consigo.

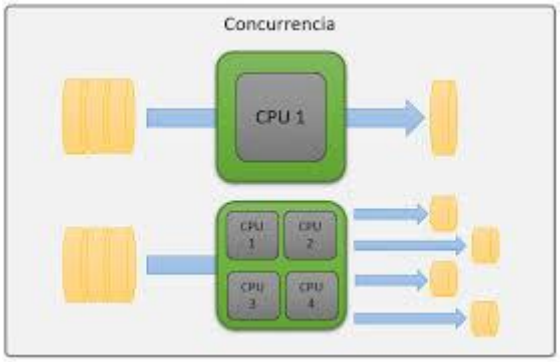
Así mismo hablaremos de los tipos de procesos concurrentes, llevaremos a cabo una definición para que quede muy claro el concepto de procesos concurrentes y así mismo definir estos tipos de procesos, los cuales son los procesos independientes y los procesos cooperantes; así mismo hablar acerca de los tipos de interacciones entre los procesos dentro de la concurrencia. Esto con la finalidad de enfocarnos en aprender y analizar un nuevo punto de vista en el cual podremos desarrollar la autonomía suficiente para en un futuro cercano poder entender el funcionamiento de los sistemas operativos.

Video del tema

Link del video: https://youtu.be/9wgAGaOfZhc

Investigación

Definición de Concurrencia



La concurrencia en los sistemas operativos es fundamental para el funcionamiento óptimo del sistema. Gracias a la concurrencia, el sistema operativo puede ejecutar múltiples tareas o procesos de manera simultánea, como abrir varias aplicaciones al mismo tiempo. Además, la concurrencia permite un mejor aprovechamiento de los recursos del equipo, lo que resulta en una mayor eficiencia en la ejecución de tareas y procesos.

La concurrencia no solo mejora la eficiencia al permitir que los recursos del sistema se utilicen de manera más efectiva, sino que también facilita la ejecución de múltiples procesos de manera intercalada, lo que permite una experiencia de usuario más fluida y la realización de diversas operaciones sin que una tarea bloquee a otras.

Defina los tipos de concurrencia

Existen varios tipos de concurrencia, a continuación, se muestran algunos tipos de estos:

Concurrencia simultánea: se refiere a la ejecución al mismo tiempo de múltiples procesos o hilos en distintos procesadores o núcleos. Esta forma de concurrencia maximiza la utilización del hardware disponible, permitiendo que diversas tareas se completen en paralelo.

Concurrencia intercalada: en sistemas con un solo procesador o núcleo, la concurrencia se logra mediante la ejecución alternada de procesos o hilos. El procesador se comparte entre las tareas, ejecutando fragmentos de cada una en sucesión rápida.

Concurrencia paralela: Similar a la concurrencia simultánea, pero con un enfoque en la ejecución conjunta de procesos o hilos que trabajan en la misma tarea. Esto se hace para completar tareas más rápido utilizando múltiples procesadores.

Concurrencia secuencial: En este tipo de concurrencia, las tareas se ejecutan una tras otra sin solapamiento. Esto se aplica en sistemas donde los recursos son limitados o donde la simplicidad en la gestión de procesos es prioritaria.

Concurrencia asíncrona: La concurrencia asíncrona permite que los procesos o hilos se ejecuten de manera independiente, sin una sincronización estricta entre ellos. Esto proporciona mayor flexibilidad y escalabilidad en sistemas que deben manejar múltiples tareas de manera dinámica.

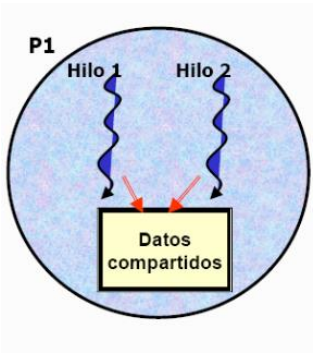
Concurrencia síncrona: Implica una coordinación precisa entre procesos o hilos, asegurando que se ejecuten en un orden específico y con la sincronización necesaria para evitar conflictos.

Modelos de programación concurrente

Modelos de Hilos

Uno de los enfoques más comunes en la programación concurrente es el uso de hilos. Un hilo es un flujo de ejecución dentro de un proceso que puede ejecutarse

de manera independiente, compartiendo los recursos del proceso principal, como la memoria y los archivos. Existen dos tipos principales de modelos de hilos:



- **Hilos a nivel de usuario:** Son gestionados por una biblioteca en el espacio de usuario, lo que los hace ligeros y rápidos. Sin embargo, tienen la desventaja de que, si un hilo se bloquea, todo el proceso se bloquea, ya que el sistema operativo no tiene control directo sobre estos hilos.
- **Hilos a nivel de núcleo:** Estos hilos son gestionados directamente por el sistema operativo, lo que permite que se ejecuten en paralelo en diferentes núcleos de la CPU. Aunque son más pesados en términos de recursos, ofrecen un mejor rendimiento en sistemas con múltiples procesadores.

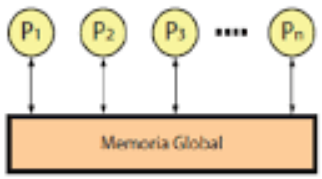
Modelo de Procesos

Otro enfoque importante es el modelo de procesos, donde la concurrencia se logra mediante la ejecución simultánea de múltiples procesos. Cada proceso tiene su propio espacio de memoria, lo que ofrece un mayor aislamiento y seguridad. Sin embargo, también implica un mayor uso de recursos del sistema.

- **Multiprocesamiento simétrico (SMP):** Este modelo permite que múltiples procesos se ejecuten al mismo tiempo en diferentes núcleos de la CPU, compartiendo los recursos del sistema de manera equilibrada.
- **Multiprocesamiento asimétrico:** En este caso, un procesador principal maneja todas las tareas del sistema operativo, mientras que los demás procesadores ejecutan los procesos del usuario. Este enfoque puede ser útil en sistemas donde se quiere optimizar el uso de recursos.

Modelo de Memoria Compartida

En el modelo de memoria compartida, varios hilos o procesos pueden acceder a la misma área de memoria, lo que facilita la comunicación rápida y eficiente entre ellos. Sin embargo, esto requiere mecanismos de sincronización para evitar problemas como las condiciones de carrera.



- **Semáforos:** Son una de las herramientas más comunes para sincronizar el acceso a recursos compartidos, asegurando que solo un hilo o proceso pueda acceder a un recurso crítico a la vez.
- **Monitores:** Proporcionan un nivel de abstracción más alto, encapsulando variables compartidas y operaciones críticas, permitiendo que solo un hilo las ejecute a la vez, lo que simplifica la programación concurrente.

Modelo de Paso de Mensajes

Este modelo se basa en la comunicación entre procesos o hilos a través del envío y recepción de mensajes. Es especialmente útil en sistemas distribuidos, donde los procesos pueden estar ejecutándose en diferentes máquinas.

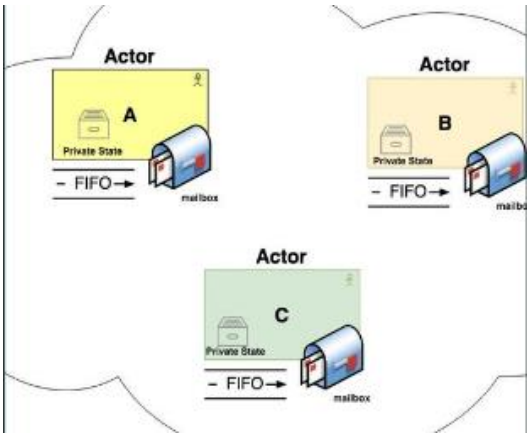
- **MPI (Message Passing Interface):** Es un estándar muy utilizado en la programación paralela para sistemas distribuidos, permitiendo la comunicación eficiente entre procesos.
- **Colas de mensajes:** Permiten la comunicación asincrónica entre procesos, facilitando la coordinación en sistemas donde los procesos no comparten memoria.

Modelo de Programación Basada en Eventos

En este modelo, las tareas se ejecutan en respuesta a eventos específicos, como la llegada de datos o la finalización de una operación de E/S. Es común en

aplicaciones donde la concurrencia se maneja a través de bucles de eventos y callbacks, en lugar de hilos o procesos múltiples.

Modelos de actores: En este paradigma, los actores son unidades de concurrencia que se comunican mediante el envío de mensajes. Cada actor maneja su propio estado, lo que evita la necesidad de sincronización explícita y facilita la programación concurrente.



Modelo de Futuro o Promesa

Este modelo es útil para manejar operaciones asincrónicas, donde una "promesa" representa un valor que estará disponible en el futuro. Mientras se espera que se cumpla la promesa, otros hilos pueden seguir ejecutando tareas, lo que mejora la eficiencia sin necesidad de bloqueo.

Modelo de Transacciones en Memoria (STM - Software Transactional Memory)

STM es un enfoque que permite la concurrencia controlada mediante transacciones, que aseguran que las operaciones sobre la memoria compartida se realicen de manera atómica. Esto significa que si se detecta un conflicto durante la ejecución de una transacción, esta puede ser abortada y reintentada, lo que garantiza la consistencia de los datos.

Ventajas y desventajas de la ejecución de concurrencia

Una de las principales ventajas es que aumenta significativamente la velocidad de ejecución de una aplicación. Los procesos concurrentes también pueden mejorar la robustez de la aplicación, ya que, si un proceso falla, los demás procesos pueden seguir funcionando sin problemas. Además, la programación concurrente puede mejorar la escalabilidad de una aplicación, ya que el programador puede agregar más procesos para mejorar el rendimiento. Otra ventaja

de la programación concurrente es que permite la creación de aplicaciones escalables. Al dividir una tarea en múltiples procesos, se puede distribuir la carga de trabajo en diferentes núcleos de procesamiento o incluso en diferentes servidores. Esto significa que la aplicación puede manejar un mayor volumen de trabajo sin afectar su rendimiento.

Aunque la programación concurrente ofrece muchas ventajas, también hay algunas desventajas. Una de las principales desventajas es la complejidad adicional que se requiere para programar una aplicación concurrente. Debido a que varios procesos deben ejecutarse simultáneamente, el programador debe tener una comprensión profunda de la forma en que los procesos interactúan entre sí para evitar conflictos. Además, debido a que los procesos se ejecutan simultáneamente, el programador debe ser capaz de asegurarse de que cada proceso se ejecuta en el orden correcto. Esto puede ser una tarea complicada, especialmente cuando hay muchos procesos en juego.

Otra desventaja de la programación concurrente es que puede aumentar la complejidad de los problemas de sincronización. Los problemas de sincronización se refieren a la necesidad de asegurar que dos o más procesos no se desempeñen al mismo tiempo, por ejemplo, cuando un proceso intenta leer un archivo mientras otro proceso lo está escribiendo. Los programadores deben tener cuidado de implementar mecanismos de sincronización adecuados para asegurar que los procesos se ejecuten correctamente sin interferir entre sí.

Otro riesgo asociado con la programación concurrente son los problemas de sincronización. Cuando múltiples procesos se ejecutan al mismo tiempo, es necesario garantizar que se sincronicen correctamente para evitar problemas como la corrupción de datos o los bloqueos del sistema. La sincronización de procesos puede ser complicada y requiere un conocimiento profundo de la programación concurrente.

Tipos de procesos concurrentes

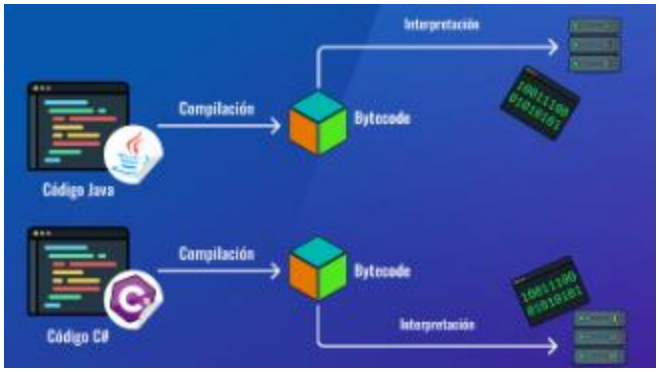
Procesos Independientes

Son aquellos que se ejecutan sin la necesidad de intervención o cooperación de otros procesos.

Un ejemplo de ellos son los shells que se ejecutan en un sistema operativo.

Procesos Cooperantes

Son aquellos que trabajan en conjunto a otros procesos para la realización de una actividad, en donde son capaces de comunicarse e interactuar entre ellos.



Un ejemplo puede verse en los compiladores que se comunican y sincronizan con los ensambladores.

Procesos de Usuarios

Los procesos de usuarios son aquellos iniciados y gestionados por los usuarios del sistema. Estos procesos pueden ser interactivos, requiriendo la intervención constante del usuario, o pueden ejecutarse en segundo plano, sin necesidad de interacción directa. Los procesos interactivos incluyen aplicaciones como los editores de texto, mientras que los procesos en segundo plano pueden incluir tareas como las descargas de archivos o las actualizaciones automáticas.

Procesos del Sistema

Los procesos del sistema son aquellos que forman parte integral del núcleo del sistema operativo y son fundamentales para su funcionamiento. Estos procesos suelen ejecutarse con privilegios elevados y no requieren intervención del usuario. Realizan tareas esenciales como la gestión de memoria, el manejo de interrupciones y la planificación de procesos. Estos procesos son críticos para el rendimiento y la estabilidad del sistema, asegurando que los recursos se asignen y gestionen de manera eficiente.

Procesos en Tiempo Real

Los procesos en tiempo real están diseñados para cumplir con plazos estrictos en su ejecución. Son comunes en sistemas donde el tiempo de respuesta es crucial, como en sistemas embebidos y de control industrial. Estos procesos deben responder a eventos en un tiempo predecible para garantizar el correcto funcionamiento del sistema. Ejemplos de esto incluyen los sistemas de control industrial, donde cualquier retraso puede resultar en fallos críticos del sistema.

Procesos Ligeros (Hilos)

Los hilos, o procesos ligeros, son unidades de ejecución dentro de un proceso que pueden funcionar concurrentemente. A diferencia de los procesos completos, los hilos comparten el mismo espacio de direcciones del proceso principal, lo que les permite ejecutarse simultáneamente y compartir recursos de manera eficiente. Un ejemplo práctico de hilos es un procesador de textos que utiliza

un hilo para interactuar con el usuario mientras otro realiza operaciones de formato en segundo plano. Esta concurrencia dentro de un mismo proceso mejora la capacidad de respuesta y la eficiencia del sistema.

¿Cuáles son los tipos de interacciones entre los procesos dentro de la concurrencia?

Comunicación Directa vs. Indirecta:

Directa: Los procesos se comunican entre sí de forma directa, como mediante el envío de mensajes o señales. Esto es útil cuando es necesario un tiempo de respuesta rápido.

Ejemplo: En un sistema de chat en tiempo real, los mensajes enviados por un usuario (proceso) se transmiten directamente al receptor a través de sockets de red.

Indirecta: La comunicación se realiza a través de un intermediario, como una cola de mensajes o un archivo compartido, permitiendo que los procesos sean más independientes y escalables.

Ejemplo: Un sistema de impresión donde múltiples aplicaciones envían documentos a una cola de impresión centralizada (el intermediario) que los gestiona en el orden de llegada.

Sincronización vs. Asincronización:

Sincronización: Los procesos esperan unos a otros para coordinar su ejecución, utilizando mecanismos como cerrojos o semáforos para evitar conflictos.

Ejemplo: Un sistema bancario que utiliza mutexes (cerrojos) para asegurar que dos transacciones no modifiquen la misma cuenta al mismo tiempo, evitando errores o inconsistencias en los saldos.

Asincronización: Los procesos funcionan de manera independiente, sin esperar unos a otros. Esto puede ser más eficiente, pero requiere un cuidado especial para mantener la coherencia de los datos.

Ejemplo: Una aplicación web donde el cliente envía una solicitud al servidor y continúa funcionando sin esperar la respuesta del servidor, mejorando la eficiencia y la experiencia del usuario.

Competencia vs. Cooperación

Competencia: Los procesos compiten por recursos limitados, lo que puede provocar conflictos si no se gestionan bien.

Ejemplo: En un servidor web, múltiples procesos de servidor compiten por acceder a la base de datos para leer o escribir datos al mismo tiempo.

Cooperación: Los procesos trabajan juntos hacia un objetivo común, compartiendo información y recursos.

Ejemplo: En un sistema de procesamiento de imágenes distribuido, diferentes procesos colaboran para analizar distintas partes de una imagen grande, combinando sus resultados al final.

Sincronización Fuerte vs. Débil:

Fuerte: Los procesos siguen un orden estricto para acceder a recursos compartidos, garantizando la integridad de los datos.

Ejemplo: Un sistema de control de tráfico aéreo donde los procesos que manejan diferentes aviones deben coordinarse estrictamente para evitar colisiones en el espacio aéreo.

Débil: Hay más flexibilidad en el orden de ejecución, lo que puede mejorar la eficiencia, pero aumenta el riesgo de errores.

Ejemplo: Un servicio de streaming de video que ajusta la calidad del video según la velocidad de la red. Los procesos de codificación y transmisión pueden

ejecutarse en diferentes órdenes para mejorar la eficiencia, pero deben mantener una coherencia mínima.

Interacciones en Memoria Compartida vs. Sistemas Distribuidos:

Memoria Compartida: Los procesos acceden a una memoria común, permitiendo una comunicación rápida pero requiriendo medidas para evitar conflictos.

Ejemplo: En un sistema operativo multiprocesador, los procesos de usuario acceden a un área común de memoria para intercambiar datos rápidamente, utilizando semáforos para evitar conflictos.

Sistemas Distribuidos: Los procesos se ejecutan en diferentes máquinas conectadas por una red, enfrentando desafíos como la latencia y la consistencia de los datos.inóptico con la información de tu investigación.

Ejemplo: En un sistema de comercio electrónico, los servidores de diferentes regiones del mundo manejan transacciones de usuarios locales y se comunican a través de mensajes para actualizar el inventario global en tiempo real.

Categorías adicionales sobre las Interacciones de la concurrencia

Interbloqueo (Deadlock)

Ejemplo: Dos procesos (A y B) compiten por dos recursos (X y Y). A bloquea X y necesita Y, mientras B bloquea Y y necesita X. Ambos procesos quedan bloqueados esperando al otro, generando un ciclo de espera mutua.

Contexto de Uso: Común en sistemas operativos donde múltiples procesos compiten por recursos compartidos como impresoras, archivos o memoria.

Inanición (Starvation)

Ejemplo: En un sistema de planificación de CPU basado en prioridades, un proceso de baja prioridad puede ser continuamente postergado si siguen llegando procesos de alta prioridad, nunca obteniendo tiempo de CPU.

Contexto de Uso: Sucede en sistemas de tiempo compartido o en colas de prioridad mal gestionadas.

Transacciones y Consistencia

Ejemplo: En un sistema de banca en línea, se usa el protocolo de Two-Phase Commit (2PC) para asegurar que todas las bases de datos acuerden los cambios en una transacción (como transferir dinero) para mantener la consistencia. Si una parte falla, la transacción se aborta.

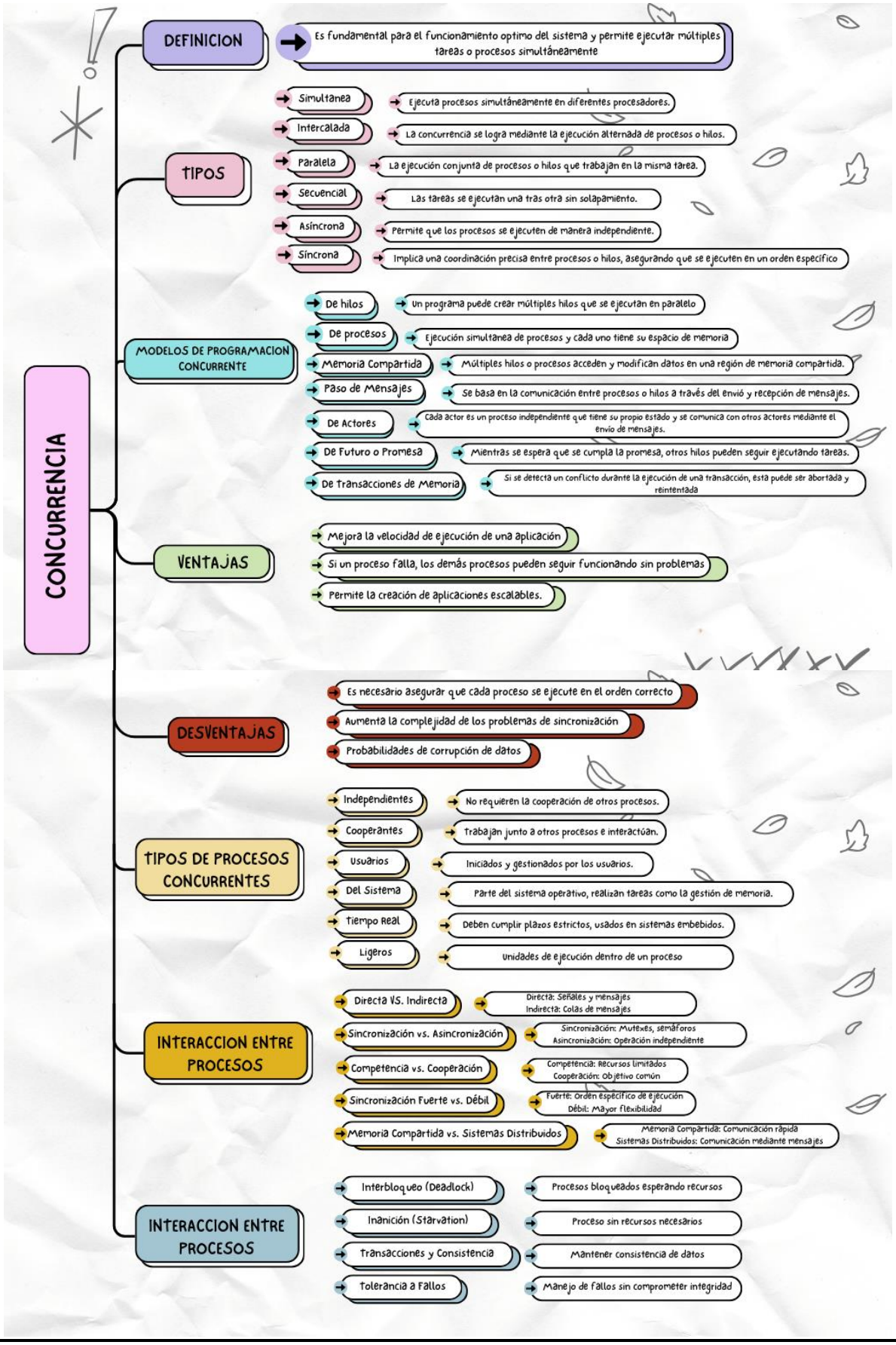
Contexto de Uso: Fundamental en bases de datos distribuidas y sistemas financieros donde la consistencia de los datos es crítica.

Tolerancia a Fallos

Ejemplo: En sistemas como Google Spanner, se utiliza un algoritmo de consenso (Paxos o Raft) para mantener la consistencia de datos incluso si algunos nodos fallan, redirigiendo solicitudes a réplicas sin interrumpir el servicio.

Contexto de Uso: Es crucial en sistemas distribuidos y servicios de alta disponibilidad, como la nube y bases de datos distribuidas.

Cuadro sinóptico



Conclusiones individuales

Antonio Enrique Hernández Ramírez 1948932

En esta actividad aprendí acerca de lo que es la concurrencia en los sistemas operativos. Así mismo, considero que esta es importante a la hora de crear aplicaciones eficientes, también que es clave en los sistemas operativos para que nuestros dispositivos realicen múltiples tareas al mismo tiempo. Así mismo aprendí acerca de los procesos en el sistema operativo LINUX y los estados de estos, junto con como ver estos procesos.

Daniel Alejandro Segura Vázquez 2022830

Durante la realización de esta actividad he podido comprender un poco más acerca de la concurrencia en general y en sistemas operativos, además de adentrarme en el tema de los tipos de concurrencia, sus procesos, ventajas y desventajas,etc. Por lo que puedo decir que está actividad me ayudó a comprender mejor cómo se manejan los procesos dentro de un sistema operativo.

Denilson Gustavo Aguilar Puente 2045231

Para concluir, debo aceptar que no conocía este término, la concurrencia, un término bastante importante hoy en día en los nuevos dispositivos electrónicos ya que es un requisito indispensable para que un electrónico haga múltiples tareas con las misma calidad y eficiencia que espera el usuario. Por otro lado, el ver como los sistemas operativos son un tema bastante complejo en comparación a otras cosas me resulta realmente complicado entender sus orígenes, esto debido a que la tecnología antigua es ampliamente extensa por las diferentes aportaciones de diferentes empresas que de poco a poco fueron moldeando estos sistemas.

Eden Leonardo Candelas Andrade 2005930

Me pareció muy interesante aprender sobre los procesos e hilos en los sistemas operativos ya que es la base mediante la cual las computadoras trabajan. Tomando en cuenta lo que vimos en la clase de arquitectura de computadoras sobre los procesadores y el hardware de una computadora, los sistemas operativos han tenido que encontrar maneras de aprovechar esos recursos lo más posible para darle a sus usuarios una experiencia lo más satisfactoria posible. Mediante la concurrencia y conociendo acerca de sus bases que son los procesos, hilos y las interacciones de los procesos que compiten por los recursos del hardware nos podemos dar cuenta de cómo funciona un sistema operativo.

Uriel Ramiro De La Fuente Del Ángel 2131973

Esta actividad me ayudó mucho sobre la concurrencia y sus tipos, qué es un proceso, que es un hilo, etc, tambien me ayudo a saber sobre el sistema linux y como sucede cada tema de la presentación en este sistema, fue de gran ayuda porque me enseñó cosas importantes de linux una de las que se me hizo interesante fue como abrían la línea de comandos y ponían una instrucción y le mostraba los diferentes procesos que se estaban ejecutando en el sistema.

Jorge Paz Villarreal 2052523

Cuando realizamos esta actividad me di cuenta que la concurrencia es un concepto fundamental en la computación moderna, que permite a múltiples procesos o hilos de ejecución operar de manera paralela o concurrente. Su relevancia radica en la optimización de recursos, la mejora en la velocidad de procesamiento y la capacidad de manejar múltiples tareas simultáneamente, lo cual es crucial en sistemas operativos, aplicaciones de servidor, y dispositivos móviles, entre otras cosas.

Alan Jahir Rivas Urbina 1958098

En esta investigación, hemos explorado la concurrencia en los sistemas operativos, destacando su importancia para la ejecución simultánea de tareas y la optimización de recursos. La concurrencia permite a los sistemas manejar múltiples procesos a la vez, mejorando la eficiencia y la experiencia del usuario. Hemos abordado los diferentes tipos de concurrencia, como la ejecución simultánea y la intercalada, y los modelos de programación concurrente, incluyendo hilos, procesos y memoria compartida. Cada modelo tiene sus propias ventajas y desafíos, como el riesgo de interbloqueo o inanición, que requieren técnicas específicas de gestión para garantizar la estabilidad y la consistencia del sistema.

Alexis Yahir Soria Salazar 1962135

Para concluir, creo que al realizar esta investigación no solo buscamos explicar y analizar los diferentes aspectos de la concurrencia en los sistemas operativos, sino también proporcionar una comprensión profunda de cómo estos principios son fundamentales para el funcionamiento eficiente y seguro de los sistemas modernos. Al abordar tanto los conceptos teóricos como los modelos prácticos de programación concurrente, yo como persona he adquirido una perspectiva integral que me permite apreciar las complejidades y beneficios de la concurrencia en el desarrollo de software. Con esta base, ya estoy mucho mejor preparado para enfrentar los desafíos que surgen en un entorno computacional cada vez más paralelo y distribuido.

Sofia Giovanna Espinoza Zapata 2052193

En este proyecto pude comprender que las interacciones entre procesos en sistemas concurrentes son fundamentales para su eficiencia y funcionamiento, ya que pueden clasificarse según diferentes criterios como la comunicación, que puede ser directa, donde los procesos se conectan rápidamente entre sí, o indirecta, utilizando intermediarios como colas de mensajes para lograr una mayor independencia. En sí, aprendí que comprender estos tipos de interacciones es

crucial para diseñar sistemas concurrentes que sean eficientes, seguros y adaptados a las necesidades específicas de cada aplicación.

Conclusión general

La concurrencia en los sistemas operativos es clave para que nuestros dispositivos puedan realizar múltiples tareas al mismo tiempo, haciendo un uso más eficiente de los recursos disponibles. En esta fundamental, hemos visto cómo se gestiona la concurrencia y los distintos tipos de procesos concurrentes que existen, además de cómo se implementan en sistemas operativos como Linux.

En cuanto a Linux, este sistema operativo maneja la concurrencia de manera muy eficiente. Utiliza técnicas avanzadas como los mutexes y semáforos para gestionar la sincronización entre procesos. También facilita la comunicación entre procesos (IPC) y maneja bien la memoria compartida. Linux es capaz de evitar problemas como la inanición y los interbloqueos, asegurando un entorno operativo estable y seguro.

En resumen, entender y manejar la concurrencia es esencial para el buen funcionamiento de los sistemas operativos modernos. Gracias a técnicas avanzadas de planificación y sincronización, los sistemas operativos pueden gestionar múltiples procesos de manera eficiente, mejorando tanto el rendimiento como la experiencia del usuario.

Referencias bibliográficas

- Edri Villagran. (2022, 17 mayo). Procesos e hilos [Vídeo]. YouTube. https://www.youtube.com/watch?v=B2I5nH0R9iU
- Hernández, P. (2015). Modelos de programación concurrente: Teoría y práctica. Universidad Nacional Autónoma de México (UNAM).
- Jammy. (2022, 4 noviembre). ¿Qué Son Los Hilos En El Sistema Operativo? » Adcod.com. Adcod.com. https://adcod.com/es/que-son-los-hilos-en-el-sistema-operativo/
- Mentores Tech: Desbloqueando tu potencial en tecnología. Conéctate con mentores expertos y alcanza nuevas alturas en tu carrera tecnológica. (n.d.). https://www.mentorestech.com/resource-guide/arquitectura-de-software/question/que-tipos-de-concurrencia-existen
- Pérez, J. C., Carballeira, F. G., de Miguel Anasagasti, P., & Costoya, F. P. (2001). Sistemas operativos. McGraw-Hill Interamericana.
- Stallings, W. (2014). Operating Systems: Internals and Design Principles (8th ed.). Pearson Education.
- Tema 4.2 Concurrencia - Módulo 4 - Sistemas en tiempo real. - Instituto Consorcio Clavijero. (s. f.). https://cursos.clavijero.edu.mx/cursos/202_str/modulo4/contenidos/tema4.2.html
- Webmaster, & Webmaster. (2024, 3 agosto). Concepto de Concurrencia ▶¿Que es? Definición y Significado. SignificadosWeb.com. https://significadosweb.com/concepto-de-concurrencia-que-es-definicion/#