

**Tecnológico Nacional de México**

**Instituto Tecnológico de Reynosa**

Materia: Sistemas Operativos 2

ACTIVIDAD05: Investigación exhaustiva del Tema02: Procesos y comunicación con sus 4 subtemas

Tema 2: Procesos y comunicación

Alumno: Castillo Jr. Gregorio

Numero de control: 19580589

Correo electrónico: L19580589@reynosa.tecnm.mx

7mo Semestre Matutino Salón 7

Docente: Mario José Santiago Sánchez

Fecha de entrega: 06/10/2022



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Consecutivo** | **Numero de**  **control** | **Apellido Paterno** | **Apellido Materno** | **Nombres(s)** | **Correo electrónico**  **Institucional** | **Firma de que está**  **autorizando que se**  **entregue esta Publicación** | **Fotografía del rostro de cada**  **Integrante del equipo (selfie)** |
| **1** | **19580589** | **Castillo Jr** |  | **Gregorio** | **L@19580589@reynosa.tecnm.mx** | Abrir foto |  |
| **2** | **19580595** | **Flores** | **Acosta** | **Sheila Lizeth** | **L@19580595@reynosa.tecnm.mx** | No hay descripción disponible. |  |
| **3** | **19580867** | **Morales** | **Calixto** | **Daniel Alexander** | **L@19580867@reynosa.tecnm.mx** | No hay descripción disponible. |  |

Equipo #12

# Tabla de contenidos

[Tabla de contenidos 3](#_Toc115678382)

[Contenido 4](#_Toc115678383)

[2.1 Procesos 4](#_Toc115678384)

[2.2 Comunicación 5](#_Toc115678385)

[2.3 Nombres 6](#_Toc115678386)

[2.4 Sincronización 7](#_Toc115678387)

[Conclusiones 8](#_Toc115678388)

[Herramientas y recursos 9](#_Toc115678389)

[Bibliografía 10](#_Toc115678390)

# Contenido

## 2.1 Procesos

***Definición***

Se le conoce como un programa o mandato que se ejecuta de manera real en el sistema. Estos procesos existen dentro de las jerarquías llamadas como “parde-hijo”, ya que un proceso iniciado por un programa es un proceso llamado “padre” y un proceso es “hijo” es el producto del “padre”.

El proceso padre puede tener varios hijos (procesos), pero un hijo (proceso) solo puede tener un padre.

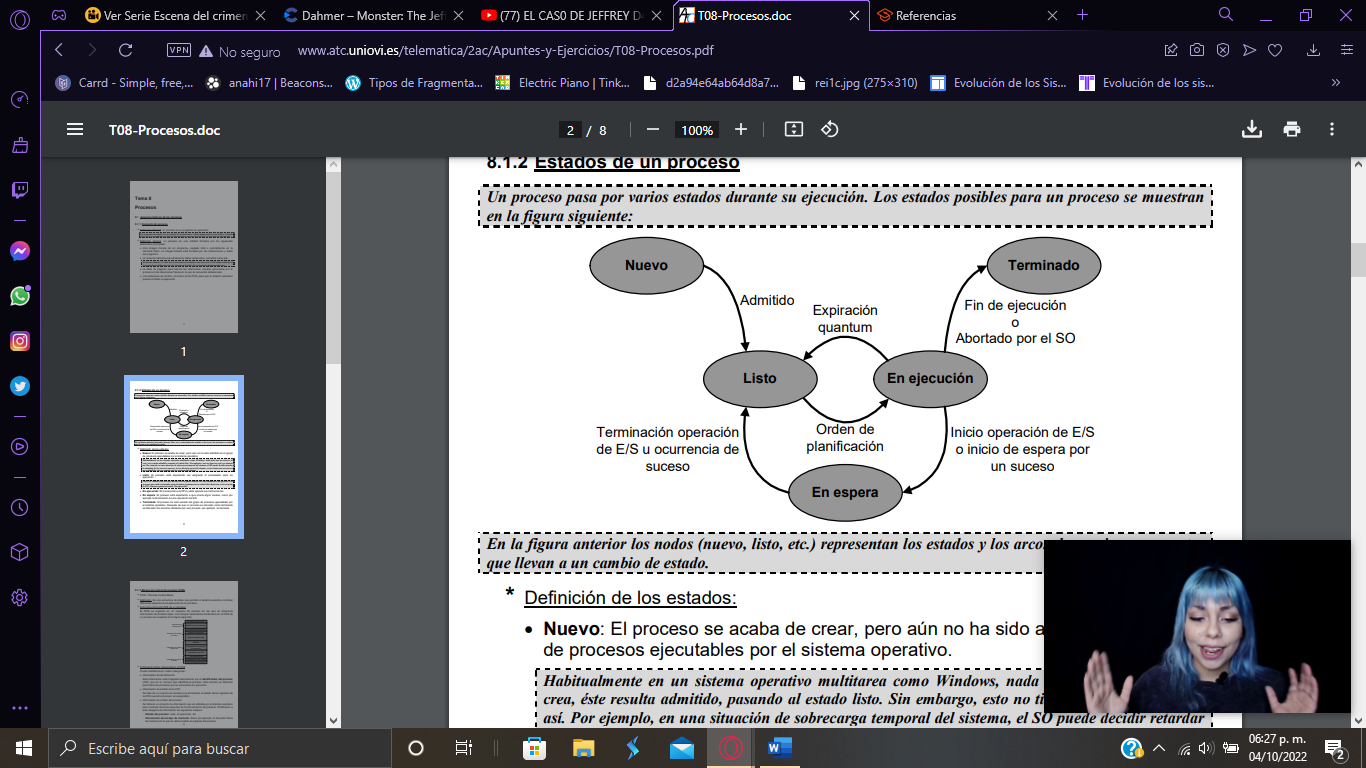
***Dentro de…***

El sistema nos asigna un número de identificación de proceso (PID), cada que el proceso es iniciado. Si el mismo programa inicia varias veces, este tendrá un numero PID distinto cada vez.

Cuando iniciamos un proceso en el sistema, este proceso utiliza una parte de recursos disponibles en el sistema. Cuando se ejecuta más de un proceso, un planificador (que se encuentra incorporado al SO) proporciona a cada proceso su parte del tiempo del sistema, dependiendo de las prioridades establecidas. Las prioridades pueden llegar a cambiarse mediante algunos comandos dependiendo del sistema.

***Estados de un proceso***

Los procesos siempre pasan por varios estados durante la ejecución. A continuación, se mostrará un diagrama de los estados correspondientes por los que pasa el proceso.



**Nuevo:** es cuando el proceso se acaba de crear, pero aun no ha sido admitido en el grupo de los procesos ejecutables por el SO.

**Listo:** es cuando el proceso esta esperando ser asignado al procesador para su ejecución.

**En ejecución:** pasa cuando el proceso tiene la CPU y esta ejecuta las instrucciones.

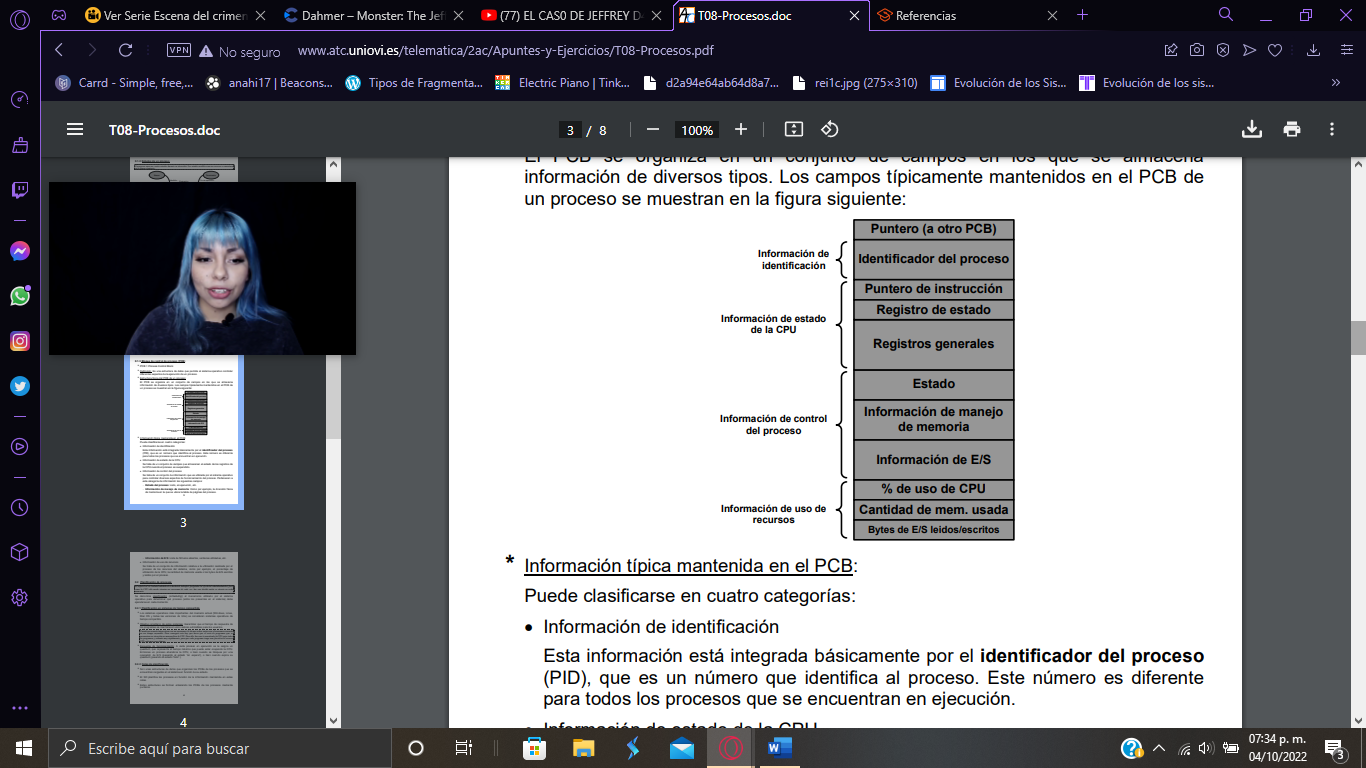
**En espera:** el proceso espera a que ocurra algún suceso, como por ejemplo la terminación de una operación de entrada/salida.

**Terminado:** Es cuando el proceso se saca del grupo de procesos ejecutables por el sistema operativo. Después de que el proceso sea marcado como finalizado, todos los recursos utilizados por ese proceso serán liberados, como, por ejemplo: la memoria.

***Bloque de control de proceso (PCB)***

Es un tipo de estructura de datos que permite al SO controlar distintos aspectos durante la ejecución de nuestro proceso.

La estructura típica del bloque de control de proceso es organizada en un conjunto de campos en los que se almacena la información de distintos tipos. Los típicos campos mantenidos del PCB de un proceso son:



***Planificación de procesos***

Se denomina planificación al mecanismo que es utilizado por el SO para determinar que proceso debe ejecutarse en cada momento.

***Planificación en sistemas de tiempo compartido***

* Tenemos los sistemas más conocidos del mercado actual: Windows, Linux, Mac OS y todas las versiones Unix, estas son consideradas sistemas operativos de tiempo compartido.
* Objetivo prioritario de estos sistemas: nos garantizan que el tiempo de respuesta de los programas se mantienen en unos valores admisibles para los usuarios.
* Esquema de función: cada proceso que este en ejecución se le asigna un quantum, que representa el máximo que puede tener ocupado la CPU, o cuando se bloquea por una operación E/S, en pocas palabras cuando expira su quantum.

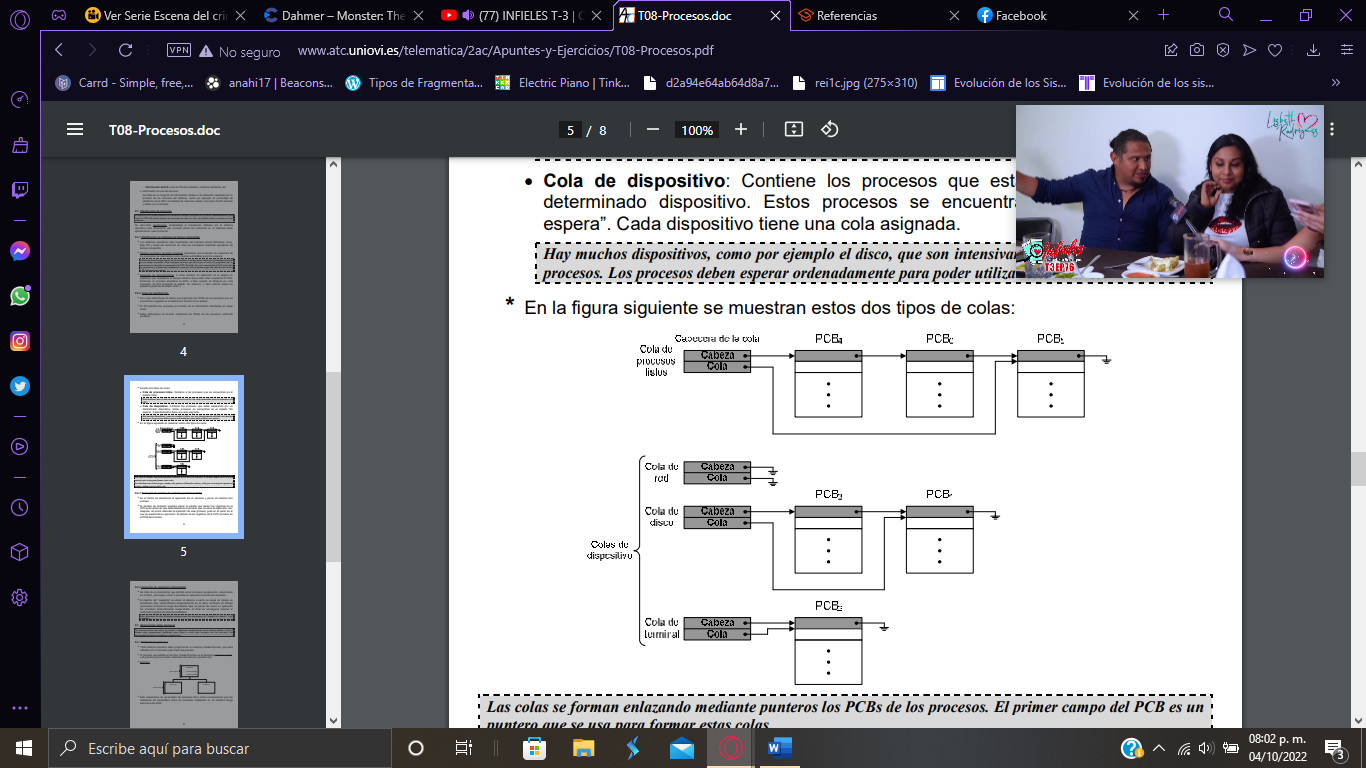
***Colas de planificación***

Las colas son estructuras de datos que organizan PCB de los procesos que están cargados en el sistema en función de su estado. Las estructuras se forman enlazando los PCB de los procesos mediante punteros.

El sistema operativo planifica los procesos en funciones de la información mantenida en estas colas.

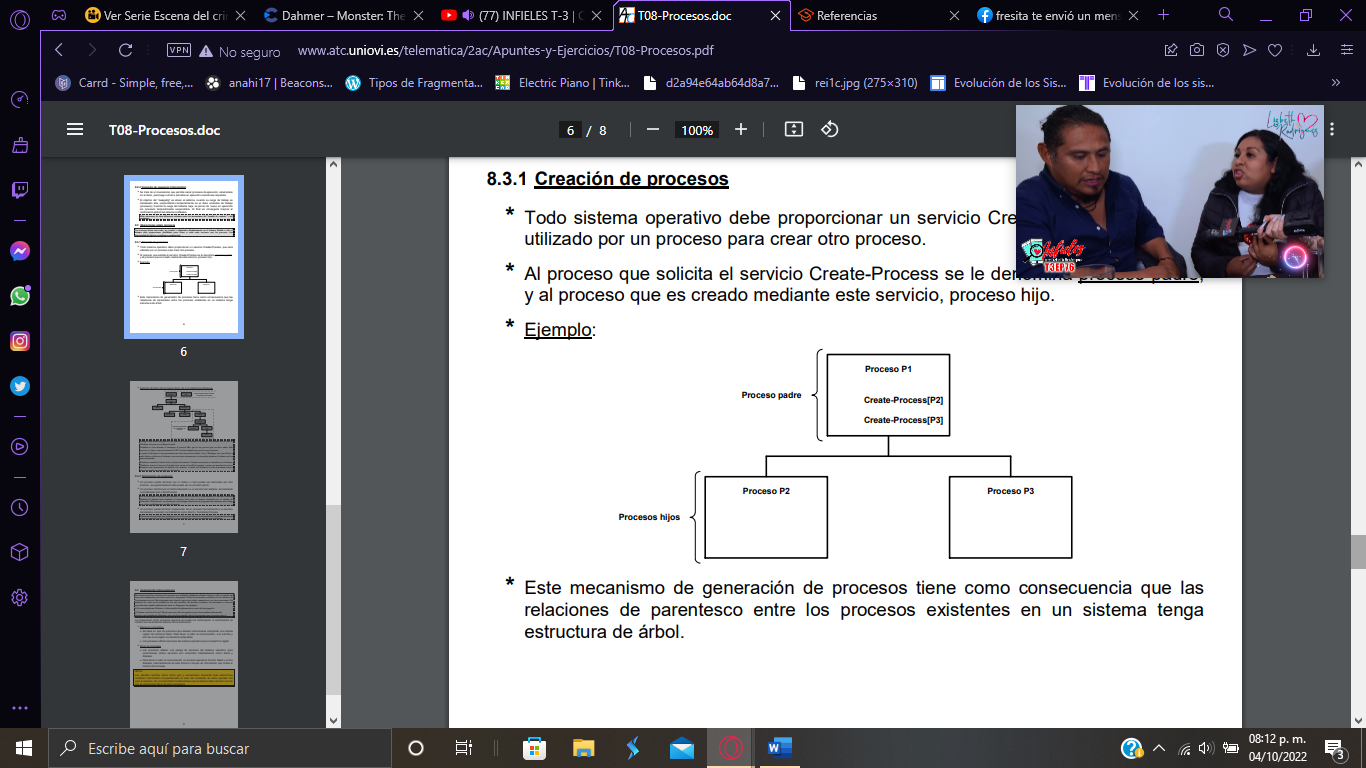
Aquí existen dos tipos de colas:

* Cola de procesos listos: contiene procesos que se encuentran en el estado “listo”.
* Cola de dispositivo: contiene los procesos que están esperando por un determinado dispositivo. Los procesos son encontrados en el estado de “espera”. Cada dispositivo tiene una cola asignada.

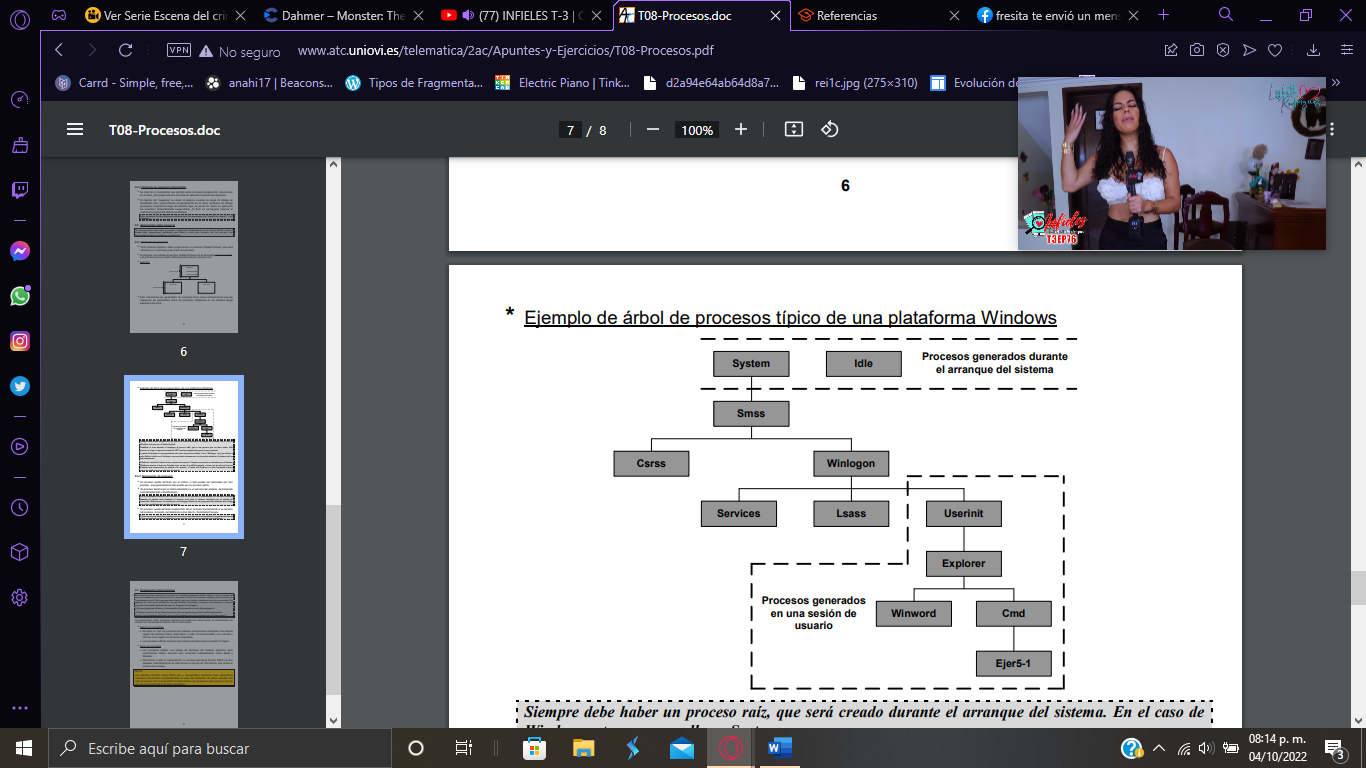


***Creación de procesos***

El SO debe proporcionar un servicio create-process, que se utilizara por un proceso para poder crear otro proceso. El proceso que necesita el servicio create-process es denominado como el padre y el proceso que es creado mediante este servicio es el hijo.



Este tipo de mecanismo de generación de procesos tiene como consecuencia que las relaciones de parentesco entre distintos procesos existentes en un sistema tengan la estructura de árbol.



**Hilos**

Los hilos son una unidad básica de utilización en el CPU, esta contiene un id de hilo, el propio programa counter, conjunto de registros, y una pila. Se representa a nivel del SO con una estructura llamada thread control block. Los hilos suelen compartir con otros hilos que pertenezcan al mismo proceso la sección de código, la selección de datos, etc.

**Virtualización**

Esta ejecuta los procesos en una maquina física de otras máquinas virtuales, en pocas palabras, se consigue que una computadora pueda trabajar como varias computadoras juntas de manera que el software o hipervisor separa el sistema operativo de las aplicaciones hardware.

**Clientes**

Los clientes son los que realizan peticiones a otro programa, al servidor, quien da la respuesta. También se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque la manera mas ventajosa es un SO multiusuario distribuido a través de una red de computadoras.

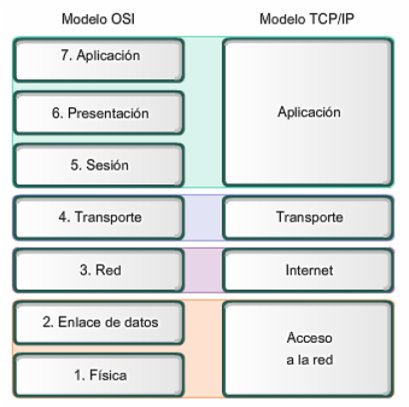
**Migración de código**

Es un movimiento de programación de un sistema a otro. Existen tres niveles distintos de la migración de código con complejidad, un costo y riesgo creciente. La migración simple implica el paso a un idioma de versión nueva. Un segundo nivel de migración mas complicado seria pasar a un lenguaje de programación diferente. La migración a una plataforma u SO completamente nuevo es el tipo de migración más complejo.

## **2.2 Comunicación**

Cuando hablamos de la comunicación tenemos que tener muy en cuenta cual de estos tipos tenemos, esto debido a que no es la misma forma de comunicar un sistema distribuido y uno unificado, esto se debe de forma principal en como es que se ejecutan y realizan los diferentes procesos dentro dichos sistemas, mientras que por un lado tenemos los sistemas distribuidos, donde pequeñas partes son recibidas por múltiples ordenadores para al final juntarlas y tener un resultado final, por el otro lado tenemos a un solo ordenador trabajando en la elaboración, de dicho proceso, con esto no significa que uno sea mejor que otro, si no que depende mucho en cuanto a las circunstancias y las necesidades con las que contemos y que es lo que queramos realizar’

Dicho esto podemos encontrar diversos métodos para establecer conexiones entre nuestros equipos, uno de ellos es el protocolo con capas, este se utiliza ya que en los sistemas distribuidos existe la falta de memoria compartida, ya que al ser diferentes equipos, estos no pueden estar conectados a una sola memoria, tanto por la limitación de conexiones, como la localización, ya que si estamos usando un sistema distribuido, podemos tener un equipo A en un país y un equipo B en otro, imposibilitando el uso de una memoria compartida, es con ello cuando se hace uso de la transferencia de mensajes

Cuando se empieza el proceso de envió de mensajes de una computadora A hacia una computadora B, este empieza a construir un mensaje en su propio espacio de direcciones, tras esto el sistema ejecuta una llamada para que nuestro sistema operativo busque el mensaje y lo envié a través de la red hacia el equipo B, este proceso puede ser multidireccional, es decir, no es necesario hacer un envió uno a uno, si no que nos permite enviar uno a muchos, siempre y cuando así lo queramos

Por últimos y esto es más como un paréntesis, tenemos que saber que, para evitar conflictos entre estas redes, tanto A como B deben coincidir en el significado de los bits que se envían

De igual forma podemos diferenciar distintos modelos, como lo son el OSI o el TCP/IP

Empezando por el modelo OSI, este esta diseñado para permitir la comunicación de los sistemas abiertos, esto se refiere a que cualquier equipo puede hacer conexión, siempre y cuando cuente con las reglas estándares para comunicarse, además encontramos que establece el formato, contenido y significado de los mensajes recibidos y enviados, estos son los que constituyeron los protocolos especializados para la comunicación y como es que esta debe desarrollarse, del mismo modo es que este mismo protocolo distingue entre dos tipos generales, con ello, antes de intercambiar los datos el emisor y el receptor deben: Establecer en forma explícita una conexión, esclarecer el protocolo a utilizar, Al finalizar, deben terminar la conexión esto para evitar vulnerabilidades del sistema,

El modelo cliente- servidor socket

Este modelo es incluso más “sencillo” ya que permite establecer una conexión sin la necesidad de estar conectados a una red, esto debido a que estamos haciendo una conexión directa entre las computadoras, lo bueno de este modelo es que no representa una complicación como lo haría OSI o el TCP/IP, de este modo quien ejecuta los requerimientos necesarios para establecer a conexión, es el servidor, con ello es que nos brinda ciertos beneficios, los cuales están entre ellos, que si este protocolo falla, se nos regresan datos de la ejecución, además de no establecer una conexión hasta que se utilice dicha conexión y por si fuera poco la pila de protocolo es aun mas corta que en otros protocolos y por lo tanto es mucho más eficiente

Direccionamiento en C

En este protocolo es esencial el conocer la dirección a la cual se esta mandando el mensaje, esto nos facilita el hecho de que elimina la ambigüedad acerca de quien es el receptor, pudiendo identificarlo en todo momento, sin embargo, nos trae un gran problema, el cual consiste en la identificación de problemas

Para solucionar esto, podemos encontrar distintas formas para arreglarlo, entre ellas, esta el hecho de no necesitar de coordenadas globales, del mismo modo pueden repetirse los nombres de los procesos en distintas maquinas, del mismo modo podemos utilizar una variable local de forma machine.local en vez de machine.process, este último método lo que nos permite es mandar un numero entero aleatorio de 16 o 32 bit, tras ello el sistema inicia una llamada para indicar al núcleo que se desea escuchar al local-id, y tras ello el núcleo sabe a cual proceso debe darle el mensaje

Comunicación RPC

Por su nomenclatura (Remote procedure call) y traducido como llamada procedural remota, este mecanismo nos permite conectarnos de una forma un tanto distinta, ya que esta lo que realiza es una ejecución, pero no en una maquina propia, si no en una maquina remota, la cual nos permite introducir comandos y ejecuciones desde remoto, permitiéndonos optimizar tiempo y esfuerzo

Para ello tenemos que tener en cuenta que es lo que realmente queremos realizar y desde que tipo de red lo estamos realizando, ya que estas pueden ser redes PAN, LAN, WAN y MAN, Cada una de las conexiones remotas resulta mucho mas complicada que la anterior, por lo que la mas sencilla es PAN y la mas complicada es WAN, esto debido a que tenemos que identificar de forma muy clara, cual es la computadora con la cual nos estamos comunicando, ya que en una red PAN es mucho mas sencillo debido a que existe un menor numero de equipos cosa que no es asi con las WAN donde estamos prácticamente conectadas a internet

## **2.3 Nombres**

## **2.4 Sincronización**

# **Conclusiones**

# **Herramientas y recursos**

# **Bibliografía**

Procesos. (s. f.). Recuperado 4 de octubre de 2022, de <https://www.ibm.com/docs/es/aix/7.2?topic=processes->

Procesos. (2017, 11 agosto). pdf. Recuperado 4 de octubre de 2022, de <http://www.atc.uniovi.es/telematica/2ac/Apuntes-y-Ejercicios/T08-Procesos.pdf>

Hilos. (s. f.). pdf. Recuperado 4 de octubre de 2022, de <https://www.fing.edu.uy/tecnoinf/maldonado/cursos/so/material/teo/so05-hilos.pdf>

Profile, V. M. C. (s. f.). *Unidad II*. Recuperado 4 de octubre de 2022, de http://sistemasoperativos2beresainz.blogspot.com/p/unidad-2-comunicacion-en-los-sistemas.html

*Unidad 2: Procesos y comunicación*. (2016, 24 septiembre). Sistemas operativos ii. Recuperado 4 de octubre de 2022, de https://soperativos2.wordpress.com/2016/09/22/unidad-2-comunicacion-en-los-sistemas-operativos-distribuidos/