

**Tecnológico Nacional de México**

**Instituto Tecnológico de Reynosa**

Materia: Sistemas Operativos 2

ACTIVIDAD09: Investigación exhaustiva del Tema04: Tolerancia a fallas con sus 4 subtemas.

Tema 4: Tolerancia a fallas

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: 22/11/2022



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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** |  |  |
| **2** | **19580595** | **Flores** | **Acosta** | **Sheila Lizeth** | **L@19580595@reynosa.tecnm.mx** |  |  |
| **3** | **19580867** | **Morales** | **Calixto** | **Daniel Alexander** | **L@19580867@reynosa.tecnm.mx** | No hay descripción disponible. |  |

Equipo #12

# Tabla de contenidos

[Tabla de contenidos 3](#_Toc117436537)

[Contenido 4](#_Toc117436538)

[3.1 Introducción de modelos de consistencia centrada en los datos 4](#_Toc117436539)

[3.2 Modelos de consistencia centrada en el cliente 7](#_Toc117436540)

[3.3 Administración de replicas 10](#_Toc117436541)

[3.4 Protocolos de consistencia 15](#_Toc117436542)

[Conclusiones 23](#_Toc117436543)

[Herramientas y recursos 24](#_Toc117436544)

[Bibliografía 25](#_Toc117436545)

# Contenido

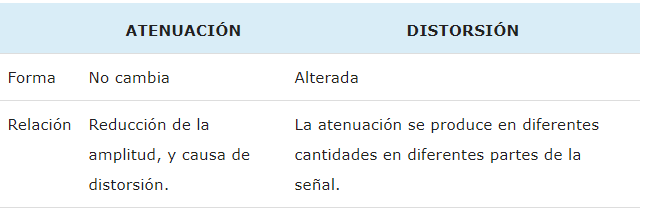
## **3.1 Introducción atenuación de un proceso**

4.1 Introducción: Atenuación de un proceso

Este tema nos demostrara el cómo se logra la tolerancia a fallas en sistemas distribuidos, tratando de tocar temas de cómo protegernos contra las fallas de proceso y una manera es replicando los procesos en grupos. Considerando los temas de diseño generales de grupos de procesos y analizamos el significado de un grupo tolerante a fallas.

Dentro de atenuación de procesos podemos encontrar puntos como:

* Forma clave de afrontar la tolerancia a un proceso defectuoso: organizar varios procesos idénticos en un grupo.
* La propiedad fundamental que tienen todos los grupos es que cuando un mensaje es enviado al grupo, todos los miembros de éste lo reciben.
* De esta manera, si en un grupo falla un proceso, afortunadamente alguno de los demás procesos puede hacerse cargo de él (Guerraoui y Schiper, 1997).



Sus características:

* Los grupos de procesos pueden ser dinámicos
* Se pueden crear grupos nuevos y destruir los viejos
* Un proceso puede unirse a un grupo o abandonarlo durante la operación del sistema.
* Un proceso puede ser miembro de varios grupos a la vez.
* En consecuencia, se requieren mecanismos adecuados para gestionar los grupos y la membresía a un grupo.

**TEMAS DE DISEÑO**

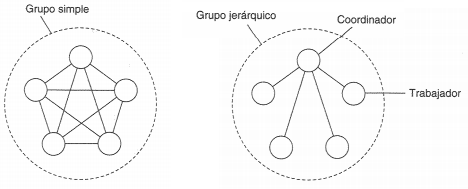
Una de las claves de afrontar la tolerancia a un proceso defectuoso es organizar varios procesos idénticos en un grupo ya que la propiedad fundamental que tiene todos los grupos es que cuando un mensaje se envía al grupo, todos los del grupo perteneciente reciben el mensaje también. De esta forma, si es que llega a fallar un proceso, afortunadamente alguno de los demás procesos se pueden hacer cargo de este.

Estos grupos pueden ser dinámicos, se pueden crear grupos nuevos y que los viejos queden en el olvido o destruirlos en palabras más claras. El proceso se puede unir a un grupo o abandonarlo durante la operación del sistema, pero necesitan mecanismos adecuados para gestionar los grupos y membresía a un grupo.

Para la explicación de los grupos, digamos que son más o menos como las organizaciones sociales. El propósito de esto es permitir que los procesos se ocupen de los conjuntos como una sola abstracción, por lo tanto, un proceso puede enviar un mensaje a un grupo determinado de servidores sin que deba saber cuáles son o cuantos son o donde están, lo cual pueden cambiar de una invocación a la siguiente.

Por eso, repitiendo de nuevo la mejor forma de enfrenar la tolerancia de un proceso es con la replicación de procesos en grupos.

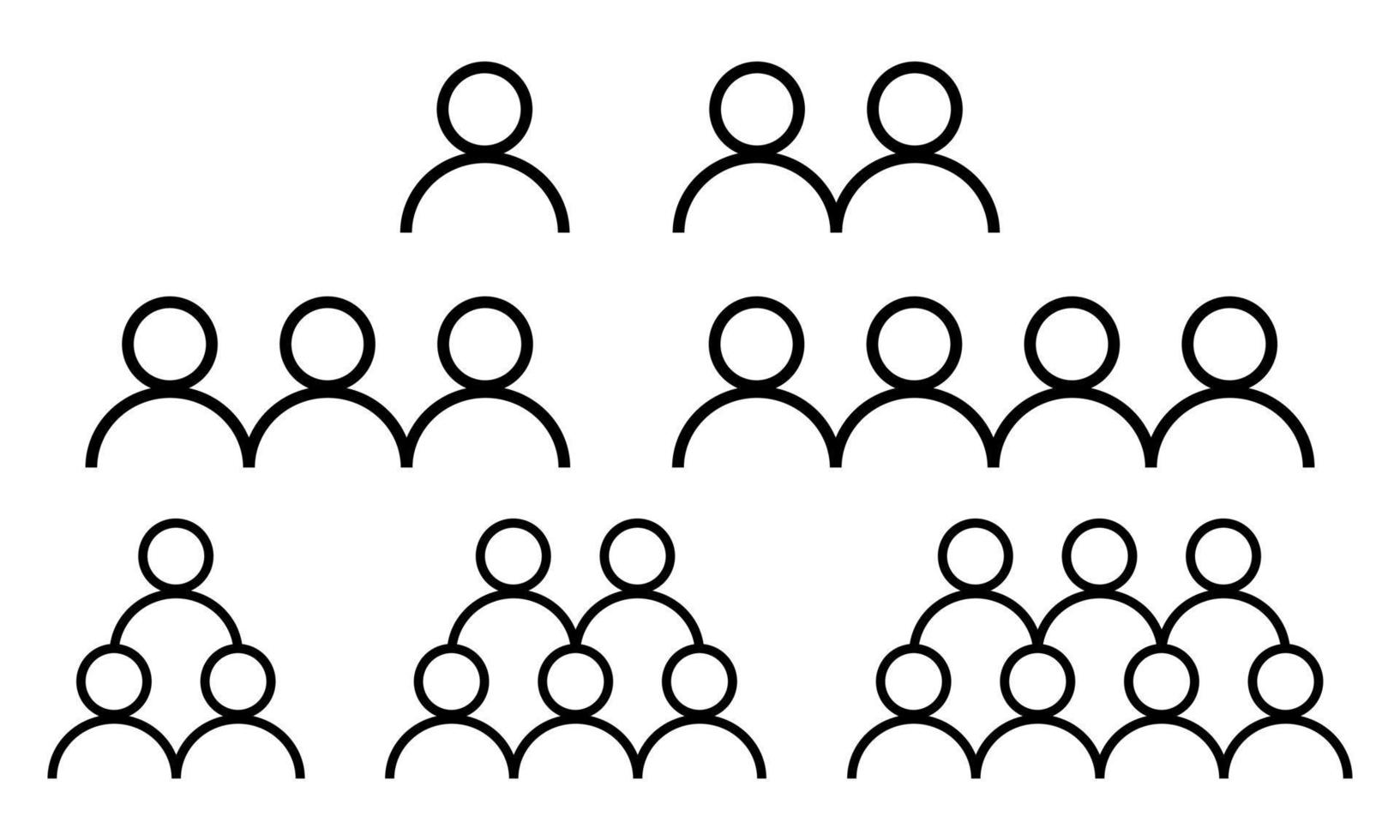
* Grupos dinámicos en su gestión.
* Abrir grupos nuevos y eliminar los viejos.



Existen dos tipos de grupos:

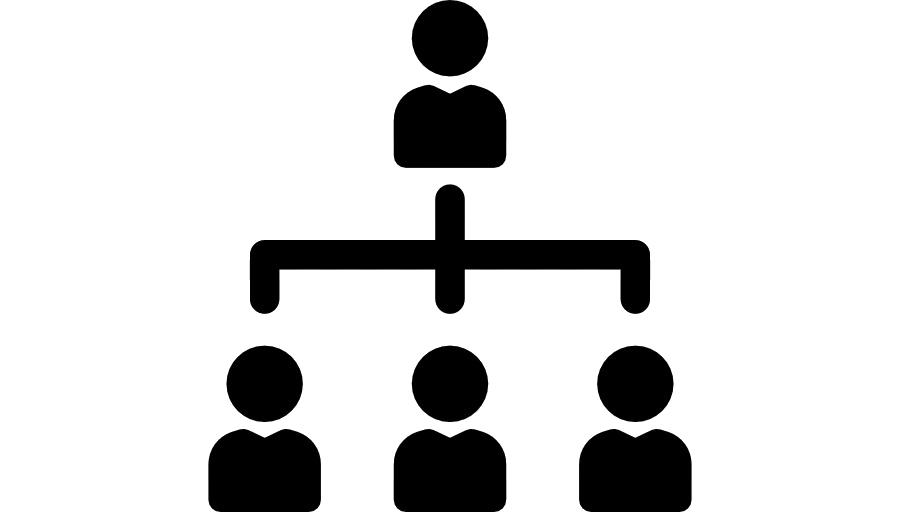
* Grupo simple

Si uno de los nodos se llega a congelar, el grupo se convierte más pequeño pero continua pero la desventaja es que la toma de decisiones es difícil. Por ejemplo, para poder decidir alguna cosa a menudo hay que someterlo a votación, eso implica retrasos y gastos indirectos.



* Grupo jerárquico

Aquí la perdida es del coordinador que hace que todo el grupo se detenga de manera brusca, pero muestra este funciona, puede tomar las decisiones sin molestar a los demás.



Una importante decisión entre los distintos grupos tiene que ver con la estructura interna. En algunos, todos los procesos son los mismos; ninguno es el que manda y todas las decisiones que se llegan a tomar de manera colectiva. En otros grupos, existe la jerarquía, por ejemplo; un proceso es el coordinador y los demás son los trabajadores. Este modelo es cuando un cliente externo o uno de los trabajadores genera una solicitud de trabajo y esta se envía al coordinador. El coordinador decide entonces cual de los trabajadores es el más apto para poder realizar el trabajo y lo remite allí, también son posibles jerarquías más complejas.

**Membresía a un grupo**

Cuando en algún grupo existe comunicación, se requiere de algún método para poder crear y eliminar los grupos, así como poder permitir que los procesos se unan a los grupos o los abandonen.

La comunicación en el grupo requiere un método para:

* Crear y eliminar grupos.
* Unión y separación de procesos a grupos.

Aquí existen dos métodos:

* Tener un servidor de grupos al cual estar enviando todas las solicitudes.

El servidor nos puede ayudar ya que puede mantener una base de datos completa de todos los grupos y la membresía exacta, suele ser un método eficiente y fácil de implementar, pero la desventaja es que el punto de fallo que representa la administración centralizada de los grupos.

* La administración distribuida de las membresías a grupos.

Este es un grupo abierto, donde un proceso extraño puede enviar un mensaje a los integrantes del grupo para poder anunciar su presencia. En un grupo cerrado se hace algo similar, ya que se debe contemplar la admisión de nuevos miembros al grupo cerrado.

Si se salen del grupo, el proceso se le debe de comunicar a los demás del grupo de que se saldrá.

Un aspecto problemático que suele presentarse cuando un miembro suele fallar, al salir, por lo tanto:

* No hay un anuncio apropiado de este hecho.
* Los demás miembros del grupo lo deben descubrir de forma experimental y luego se lo puede eliminar del grupo.

Otro aspecto de suma importancia es que la entrada y la salida al grupo se debe de sincronizar con el envió del mensaje:

* Un proceso que se unió a un grupo debe recibir todos los mensajes que se envíen al mismo.
* Un proceso que ha salido de un grupo: no debe recibir más mensajes del grupo, el grupo no debe recibir más mensajes del proceso y los otros miembros no deben recibir más mensajes del proceso saliente.

**ENMASCARAMIENTO DE FALLAS Y REPLICACIÓN**

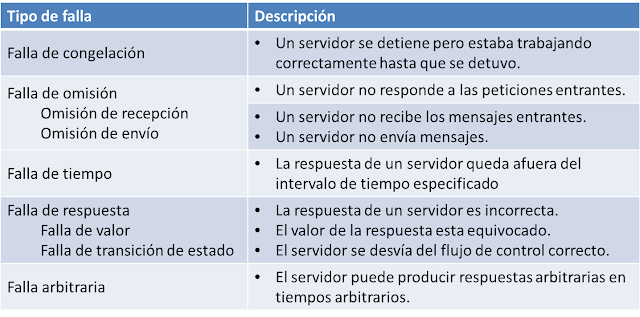
Los grupos de procesos son parte de la solución implementada que construye sistemas tolerantes a las fallas. Al tener un grupo de procesos idénticos permite disfrazar uno o más procesos defectuosos presentes a dichos grupos. En otras palabras, nosotros podemos replicar procesos y organizarlos en un grupo para reemplazar un solo proceso vulnerable con un grupo de procesos que sea tolerante a fallas.

**Replicación casada en un protocolo primario**

En el caso de fallas, generalmente aparece en forma de un protocolo de respaldo primario, así un grupo de procesos se puede organizar de forma jerárquica en la cual un protocolo primario coordina todas las operaciones de escritura.

**Protocolos de escrituras replicados**

Se utiliza en la forma de replicación activa, así como también por medio de protocolos. Estas soluciones corresponden a organizar un conjunto de los procesos idénticos en un grupo simple, una de las ventajas principales es que los grupos no tienen un solo punto de falla a causa de la coordinación distribuida.



**ACUERDO EN SISTEMAS DEFECTUOSOS**

La organización de los procesos replicados en un grupo incrementa la tolerancia a las fallas, es decir que, si un cliente puede basar las decisiones en un mecanismo de votación, incluso se puede tolerar que k de 2k + 1 procesos mienten sobre sus resultados. Es una suposición que se hace, sin embargo, es que los procesos no hacen equipo para producir un resultado equivocado.

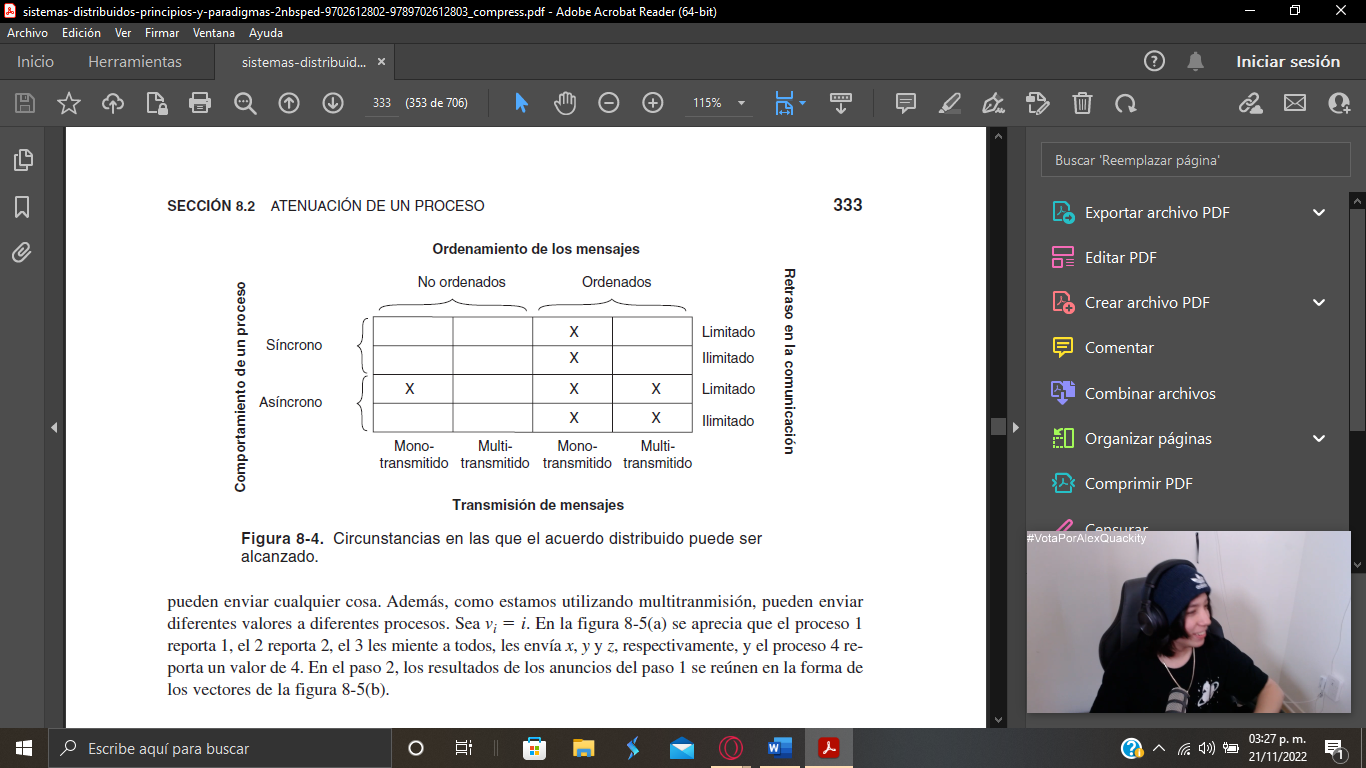
Las cosas se complican cuando se demanda que un grupo de procesos llegue a un acuerdo, lo cual es requerido en muchos casos.

El objetivo general de los algoritmos de acuerdo distribuidos es hacer que todos los procesos no defectuosos alcancen un consenso en un tema, y poder establecer dicho consenso dentro en algún numero finito de pasos. Pero se complica cuando las diferentes suposiciones sobre el sistema subyacente requieren diferentes soluciones.

Sistemas síncronos contra sistemas asíncronos:

1. El retraso de la comunicación está o no limitado
2. La entrega del mensaje se hace o no en orden
3. Los mensajes se transmiten unitransmisión o multitransmisión

Y solo es posible llegar a un acuerdo en la siguiente situación:



**DETECCIÓN DE FALLAS**

Para enmascarar las fallas es necesario detectarlas antes. La detección de fallas es una de las piedras angulares de la tolerancia a fallas en los sistemas distribuidos, a lo que esto se reduce para que en un grupo de procesos los miembros no defectuosos deberán de ser capaces de decidir quien sigue siendo miembro y quien no. De otra forma, debemos ser capaces de detectar cuando un miembro falla.

Cuando se detectan fallas de procesos, en esencia solo existen solo dos mecanismos o los procesos son enviados mensajes de manera activa del tipo ‘¿estas vivo?’ entre sí o esperan pasivamente hasta que lleguen mensajes de los diferentes procesos. Lo segundo tiene sentido solo cuando se garantiza que existe suficiente comunicación entre los procesos. Dentro de la practica generalmente se envía activamente un ping a los procesos.

La detección de fallas también puede ocurrir como un efecto colateral del intercambio regular de la información con los vecinos. Este enfoque también se adopta esencialmente en Obduro: los procesos son informados periódicamente acerca de su disponibilidad de servicio. Con esta información se disemina gradualmente por la red a través de charlas. Durante un tiempo cada proceso se enterará de la existencia de todos los demás procesos, pero lo más importante es que tendrá suficiente información localmente disponible para decidir si un proceso ha fallado o no. Un miembro para el cual la información sobre disponibilidad es obsoleta presumiblemente ha fallado.

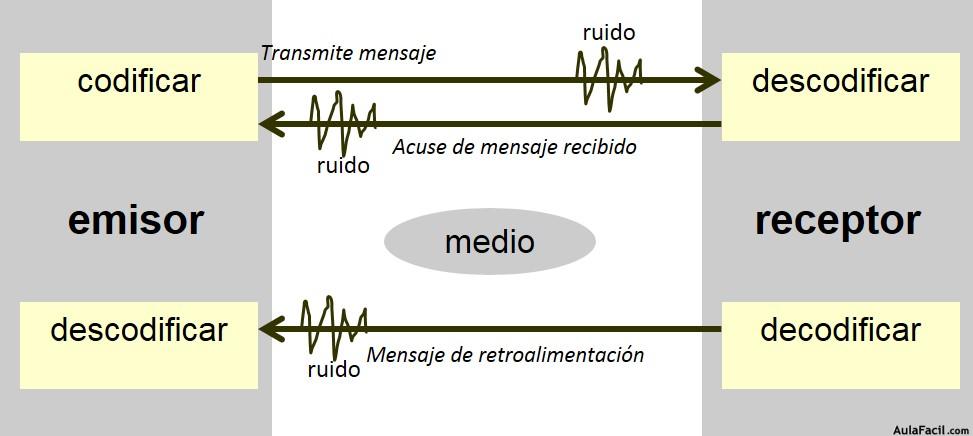
Esto conduce a otro tema: cuando se detecta la falla de un miembro ¿Cómo se debe infórmalo a otros procesos que no son defectuosos? Un método simple es que se sigue FUSE ya que todos los procesos pueden unirse en un grupo que corresponda a una amplia red. Los miembros crean un árbol expansivo para poder monitorear los miembros que fallan. Cuando un vecino no corresponde, el nodo que envió el ping cambia de forma rápida a un estado en el cual tampoco responderá a los pings enviados por otros nodos.

## **3.2 Comunicación confiable cliente-servidor**

Cuando hablamos de la comunicación cliente servidor esta no tiene mas que en como es que conectamos un usuario con otro, es decir, todas las herramientas y procesos que utilizamos para poder trabajar lo que a una conexión remota se refiere, desde lo mas simple como lo serian los cables para realizar las conexiones, hasta lo mas complejo como los protocolos de comunicación previamente establecidos, es con ello que a lo largo de los años se han ido desarrollando nuevas tecnologías que nos permiten elaborar conexiones mucho mas fiables y sobre todo mas seguras, como sabemos, quienes se conectan a este tipo de software, pueden tener acceso no solo a lo que ellos necesitan si no a la información de otras entidades, personas o empresas, generando asi una brecha de seguridad que dependiendo la situación del afectado o la información recabada, puede ser mayor o menos el problema, de tal forma es que se han creado los modelos de comunicación donde encontramos distintas formas de mover, almacenar y utilizar la información

Modelos de comunicación

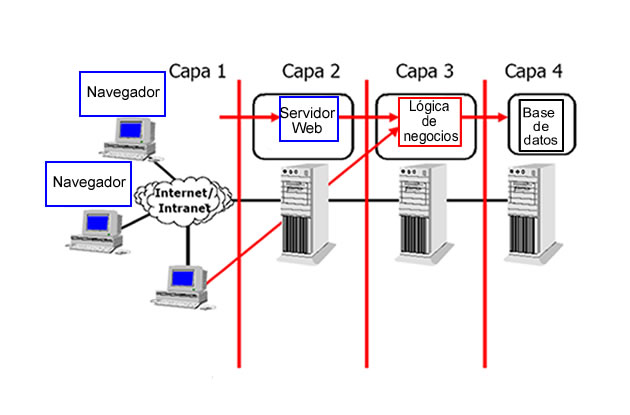
Existen muchos y muy diversos modelos de comunicación, a lo largo del tiempo estos se han ido modificando de acuerdo a la necesidad del usuario y del sistema en el que este se encuentra, dicho esto, también existen modelos que se han quedado obsoletos y hoy en día ya no se utilizan, esto porque han sido modificados, o se han creado versiones mejoradas de este.



El proposito de estos mensajes, como ya se sabe es, permitir la comunicación de información a lo largo de una vasta extensión territorial, y generando mensajes prácticamente instantáneos, del mismo modo para preservar la confidencialidad de estos mensajes se hacen uso de diversos protocolos como lo son el TCP/IP

Comunicación cliente-servidor

Esta comunicación es llevada a cabo por los modelos ya anteriormente mencionados, sin embargo cuando hacemos uso de estos modelos no basta con solo tenerlos, puesto que estos solo moderan el como es que se mandara la información que deseamos llegue al destinatario, sin embargo no contamos con las “conexión o parámetros” necesarios para hacer uso de estos modelos, es ahí donde entra en juego lo que se le conoce como codificación backend o dicho en otras palabras, la programación que hay detrás de la conexiones y nos permite enviar y recibir información, para ello hacemos uso de tecnologías como lo son las bases de datos, ayudándonos con SQL, realizar las conexiones con PHP, modelar interfaces como lo seria con HTML y generar utilidad con Java, esto radica en que no solo una tecnología sea involucrada a la hora de realizar una tarea tan compleja como lo es mandar un mensaje, también para ello es que tenemos que tener en cuenta a lo que el encadenamiento se refiere, que no es nada mas que el modelo de N capas, las cuales nos dicen en número de saltos de un servidor a otro para que se pueda realizar el envió de un mensaje, si bien existe la posibilidad de que este numero de capas aumente, siempre se buscara hacer la conexión por el numero de capas menor, esto puesto que supone una carga menos excesiva de recursos así como de tiempo, ya que de mandarse el mensaje por la capa más grande este puede perderse o llegar corrompido, debido a los múltiples saltos que dio el mensaje, perdiendo pequeñas fracciones de información en el camino



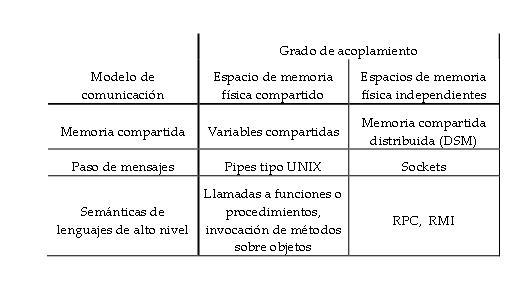
De este modo es que podemos ver como es que funcionan las capas, sirven para poder mandar el mensaje a travez de servidores, es ahi donde se hace uso de de los diferentes protocolos, de tal modo que nuestro sistema ya tiene la utilizacion de software por parte del cliente refiriendonos a php, html, etc. y tambien el uso del lado del servidor como lo son los protocolos TCP/IP



Es en esta tabla donde podemos ver de forma mas clara a lo que el utilizamiento de metodos hablamos, en esta vemos como actividades como el cloud computing que son ejecutadas por parte del servidor y del mismo modo son almacenados en esta, aquí podemos entender como es que funcionan las diversas tecnologias actuales y como de importantes son, debido a lo que esta actividad significa

Comunicacion por RPC (Remote call procedure)

claramente si hablamos de protocolos, uno de los mas importantes es el RPC o llamada remota, la cual nos permite tener acceso a un servidor, computadora o cualquier entidad de carácter informatica, permitiendonos interactuar con dicho dispositivo ya sea leyendo, modificando o agregando datos, dependiendo de los permisos con los que contemos, el proposito escencial de esta tecnologia no es mas alla de simplificar el metodo de administracion de datos en tecnologias remotas, evitando asi redireccionamientos innecesarios por parte del servidor local, y es aquí donde entra en juego lo que conocemos como memoraria compartida, la cual nos permite hacer modificaciones de carácter remoto en otra entidad

Tolerancia a fallas

Es el objetivo principal de un sistema distribuido y la razon por la cual fueron creados, cuando trabajamos con sistema unicos podemos ver reflejados multiples errores a la hora de trabajar con ello, esto debido a que los sistemas o funcionan o no lo hacen pues bien, los sistemas distribuidos permiten ejercer un llamado al rescate ya que al trabajar con nodos, existen muchas formas de llegar a un mismo resultado, ya que de encontrar fallas en una ruta, exiten n mas que nos permiten usarlar para resolver nuestro problema, sin embargo esto no quita que podamos tener fallos a la hora de trabajar con ellas, entre estos problemas encontramos 2 escenciales:

Asincrono: no hay cordinacion en el tiempo.

Sincrono: Se suponen limites maximos para el retraso del mensaje

para empezar con el Asincrono, este se refiere cuando existe un desface entre servidor A y servidor B, esto puede ser dado por mucahs situaciones, ingreso de actualizaciones, zona horaria, cambio de reloj, etc. esto traeria consigo el que los datos que ingresemos desde A hacia B no se vean reflejados en tiempo y forma ya que no se encuentra concordancia en las actualizaciones, y por cuestiones de seguridad el mensaje se pierda

El otro problema que se puede presentar es aque donde nos vemos limitados por l tiempo de conexión, de forma que si nos pasamos de este umbral, no nos sea permitido ni el acceso, ni la modificacion de ningun dato, ya que da a entender que quien esta ingresando puede ser alguien muy remoto y podria comprometer los datos que en dicho servidor se encuentra

Cabe recalcar en que estos errores se pueden dar por muchos factores y no solo por amenazas de seguridad, ya que un simple cable roto puede suponer un retraso en el acceso de la informacion, ya que se asume que los multiples protocolos implementados mantiene segura la red en la que estamos, y los errores presentados en su mayoria deben ser de indole fisica



## **3.3 Comunicación confiable en grupo**

## **3.4 Recuperación**

# **Conclusiones**

A lo largo de la unidad, entendimos en que consisten los procesos de escritura y lectura de un sistema distribuido, entendimos que la complejidad de estos radica esencialmente en cuanto existen diversos usuarios, leyendo y modificando un solo archivos, esto quiere decir que tenemos que desarrollar una forma de darle un orden y secuencia a lo que queramos presentar, ya que de no hacerlo podemos desorganizar todo, o peor aun perderlo todo, no hablamos solamente de archivos, sino que también de bases de datos completas, ya que contamos con los procesos y desarrollos necesarios para esto, aunque siempre se pueden encontrar cosas en las cuales mejorar, estos modelos presentados en el documento son los estándares mínimos con los cuales tiene que cumplir nuestro sistema, esto para generar confianza a la hora de usarlos y no encontremos errores críticos a la hora de trabajar con ello, es importante destacar que la efectividad de estos procesos radica en diversos aspectos y no solo de software, desde el hardware que usamos, hasta las personas con las que conectamos, es fundamental entender que mientras mas complejidad exista en estos sistemas no necesariamente significa que sean mejores, puesto que el objetivo siempre a sido reducir el tiempo de perdida de los usuarios y maximizar el tiempo de producción

# **Herramientas y recursos**

* Imágenes de Google
* Power Point para crear la exposición
* Youtube
* El documento de nuestra investigación del tema
* Scribbr para el APA
* Equipo de cómputo.

# **Bibliografía**

Sistemas Operativos II - Gestión de Curso Edgard. (s. f.). <https://sites.google.com/site/gestiondecursoedgard/agosto---diciembre-2018/sistemas-operativos-ii>

Sistemas Distribuidos - Tolerancia a fallos. (2018, 8 enero). <https://programacion-js.blogspot.com/2018/01/sistemas-distribuidos-tolerancia-fallos.html>

*2.1 Comunicación: comunicación con cliente – servidor, comunicación con llamada a procedimiento remoto, comunicación en grupo, tolerancia a fallos. - mrTripas*. (s. f.). https://sites.google.com/site/mrtripus/home/sistemas-operativos-2/2-1-comunicacion-comunicacion-con-cliente-servidor-comunicacion-con-llamada-a-procedimiento-remoto-comunicacion-en-grupo-tolerancia-a-fallos

*404 Not Found*. (s. f.). http://cs.uns.edu.ar/%7Esd/data/apuntes/SD-2019-mod+03.pdf

*Modelo de Comunicación de Redes Informaticas*. (2022, 13 junio). MindMeister. https://www.mindmeister.com/es/839898047/modelo-de-comunicaci-n-de-redes-informaticas