

**Tecnológico Nacional de México**

**Instituto Tecnológico de Reynosa**

Materia: Sistemas Operativos 2

ACTIVIDAD03: Investigación exhaustiva del Tema01: Introducción a los sistemas distribuidos con sus 3 subtemas.

Tema 1: Introducción a los sistemas distribuidos

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: 16/9/2022



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **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](#_Toc114090274)

[Contenido 4](#_Toc114090275)

[1.1 Definición de sistemas distribuidos 4](#_Toc114090276)

[1.2 Objetivos de un sistema distribuido 7](#_Toc114090277)

[1.3 Tipos de sistema distribuidos 9](#_Toc114090278)

[1.4 Arquitectura 16](#_Toc114090279)

[Conclusiones 22](#_Toc114090280)

[Herramientas y recursos 23](#_Toc114090281)

[Bibliografía 24](#_Toc114090282)

# Contenido

## 1.1 Definición de sistemas distribuidos

Un sistema distribuido es aquel en el cual varios equipos de computo trabajan al unisonó para resolver una tarea en concreto permitiendo hacer un trabajo mas pesado en un menor tiempo

Sin embargo, su principal objetivo es el de descentralizar la información que en ella se encuentra, fragmentándola y haciéndola rebotar a través de varios sitios para que al final lleguen al usuario el cual la necesita

Es esto entonces donde podemos contar con diferentes maquinas físicas trabajando en un mismo objetivo es decir un sistema, estos sistemas tienen la característica de contar con:

Recursos compartidos: Estos sistemas cuentas ya sea con hardware, software y/o datos de forma compartida, es decir que si una maquina contiene ciertos datos, la otra maquina también lo hará, o incluso, si una maquina está funcionando al 25% de su capacidad y otra se está esforzando a más, la primera compartirá sus recursos para que la segunda no se sobresature, esto hará que las operaciones sean mucho más rápidas y que los equipos de los cuales estamos obteniendo los datos que necesitamos, no sufran de cuellos de botella o calentamiento por alto consumo de recursos

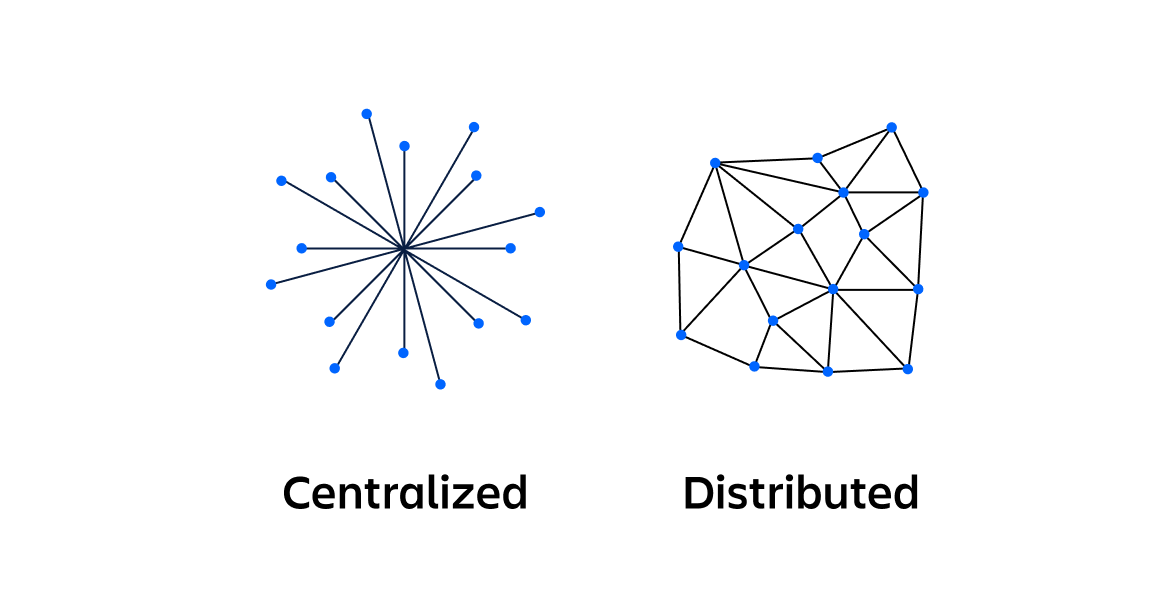
Procesamiento simultaneo: Con esto y volviendo al punto anterior, nos permite que varios equipos se centren en una misma tarea, quitándole peso a un solo equipo y distribuyéndolo entre toda la red de ordenadores con la cual contamos para este trabajo

Escalabilidad: Este punto es uno de los mas importantes puesto que denota una de las peculiaridades mas importantes a la hora de trabajar con un sistema distribuido, el cual es la escalabilidad, esta se refiere a que podemos mejorar el sistema de forma sencilla solamente agregando un equipo mas a la red y que para ello no haya que destruir toda la estructura con la que contamos ya previamente

Detección de errores: Los sistemas distribuidos nos permiten detectar errores de una forma mas sencilla ya que al estar la información distribuida, seria poca la información que habría que reparar tras una corrección o desconexión de un nodo, generando así un puenteo entre otros equipos para el trabajo que estaba generando el equipo dañado

Transparencia: Con la transparencia podemos tener acceso a toda la información que se encuentra dentro del sistema, puesto que no existen jerarquías, o nodos clave, que sean lideres a la hora de trabajar con este tipo de sistemas, por lo tanto, contamos con un sistema donde todos y a la vez ningún equipo es importante, debido a que no existen divisiones, si no mas bien un conjunto de todo

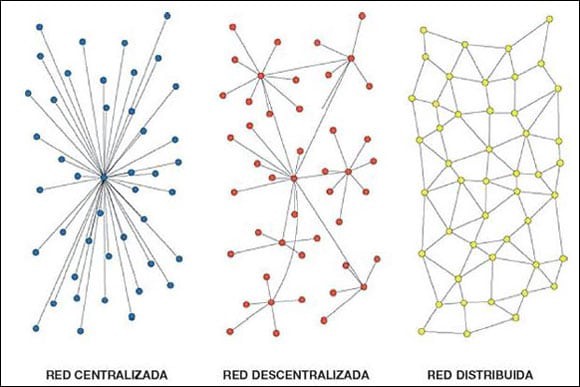
La principal diferencia entre un sistema distribuido y uno centralizado, es que en este ultimo contamos con nodos claves, que, de fallar, todo nuestro sistema queda inutilizado y de perder datos, veríamos nuestro sistema comprometido, entonces, con un sistema distribuido se podría decir de manera muy superficial que contamos con copias de este sistema repartidas a lo largo de varios equipos y que si uno fallara, podríamos sacar la información que necesitamos de un equipo alternativo al que estábamos usando.



En esta imagen podemos notar de forma sencilla como es que funciona un sistema distribuido y uno centralizado, dándonos cuenta que el distribuido conecta con varios puntos al mismo tiempo, mientras que en el centralizado solamente con el centro.

Es en este momento cuando empezamos a ver las ventajas de este tipo de sistemas, donde trabajos que se pueden hacer desde diversos puntos ya ni siquiera físico, incluso virtuales, podemos entonces generar una red que abarque a estos elementos para un flujo de información muchísimo mayor y de mejor calidad, ya que podemos comparar en tiempo real los beneficios tanto en procesamiento como en recursos y estabilidad, aun así uno de los problemas mas habituales de estos sistemas es que no siempre tienden a ser los mas seguros en cuanto a robo de información y en caso de tener a alguien malicioso dentro de nuestro equipo, es donde tendremos problemas, no solo para detectar el error si no incluso para detectar a quien o quienes lo están generando, ya que al no exigir una jerarquía de niveles en estos sistema y todos contar casi con la misma todo el tiempo, es difícil descartar a quienes no tienen acceso a ese tipo de información delicada, trayéndonos problemas de salida de información no deseada y pudiendo revelar al publico contenido que no debería ser revelado

Si bien esto puede ser una desventaja, depende mucho de como sea observado, puesto que, de estar en el caso contrario, seria una ventaja el que la demás gente no sepa quienes somos, ni que hacemos exactamente, en nuestro equipo o en el resto de equipos que manipulamos.

Además de lo ya planteado y que la idea en general suene muy bien, también tenemos como inconveniente el hecho de que estas redes cuanto más crezcan, mayor es su complejidad y el poder mantener el sistema trabajando por tiempos muy prolongados o que en todo caso se de una falla total, nos encontraríamos en un verdadero problema que habría que resolver en el mayor tiempo posible.

## 1.2 Objetivos de un sistema distribuido

El objetivo de estos sistemas en esencia principal es simple, que, de fallar, no falle todo y que, de trabajar, todos lo hagan, con ello en mente los objetivos con los cuales podemos relacionar a este tipo de sistemas son unos muy claros

Empecemos por buscar la estabilidad del sistema, con este objetivo es el principal por el cual fueron creadas estos sistemas distribuidos, cuando nosotros tenemos 2 equipos, y uno de ellos nos da un rendimiento muchísimo mayor al otro con el que contamos, nos gustaría poder distribuir el rendimiento que nos sobra, para poder así evitar que un equipo se dañe o caiga, esto debido a problemas de software o hardware, aun así lo que queremos evitar es que los sistemas se estropeen, y con ello nuestra información

Por otro lado, tenemos la administración, si bien con un sistema centralizado podemos tener un mayor orden a la hora de asignar roles, esto limita mucho las acciones con las que los usuarios cuentan, de modo que para grandes equipos de trabajo el estar llamando a un rango superior para autorizar cierto progreso en el sistema o dato, esto se resuelve haciendo que la información sea generalizada para lograr una mejor distribución de este trabajo

En cuanto hablamos de seguridad, podemos decir que tiene ciertos inconvenientes, si bien la red nos permite trabajar con varios equipos al mismo tiempo y distribuir tanto la información como los recursos, tenemos que asegurarnos a quien o quienes se los vamos a entregar, porque si bien en caso de detectar al intruso podemos expulsarlo del sistema y que este siga en pie sin mayor problema, la dificultad de esto radica en encontrarlo ya que puede ser un solo nodo el que está siendo atacado por un solo usuario, o incluso múltiples usuarios atacando múltiples nodos, en este tipo de casos encontrar al culpable cuanto antes, radica o en que no pase nada con el sistema o en poder perderlos todo

Por el lado de la escalabilidad nos encontramos con que si bien el sistema es bastante fácil de llevar a cabo esta práctica, esta tiene que llevarse con cautela para no dejar vulnerabilidades en el sistema que puedan generar fuga de información o perdida de datos, esto se logra teniendo el sistema con una escalabilidad casi desde el principio de este, es decir que el sistema este bien fundamentado desde el inicio y funcione correctamente, ubicando las partes esenciales de este y siendo mejorable de una forma, rápida, sencilla y efectiva

Si bien podemos trabajar sobre estas bases, no siempre podemos fiarnos de ellas, ya que representan un sistema ideal, el cual sabemos de antemano que es casi imposible de que exista, ya ni siquiera por cuestión de fallas humanas, si no por cuestiones externas a uno de nosotros que no podemos controlar, y si bien podemos intentar predecirlas, estas predicciones no siempre serán del todo ciertas, teniendo un margen de error con el cual si o si tenemos que trabajar y arriesgarnos a que en cualquier momento nos topemos con la realidad de un sistema optimo, especifico, potente y que aun así falle

## 1.3 Tipos de sistema distribuidos

Ha varias formas o taxonomías propuestas para clasificar a los sistemas distribuidos. En nuestro caso, diferenciaremos los sistemas distribuidos en tres clases: *sistemas computacionales distribuidos, sistemas de información distribuidos, y sistemas embebidos distribuidos*.

***Sistemas computacionales distribuidos***

Una de las clases de sistemas distribuidos mas importantes y populares en nuestros días son los que son usados para procesar las tareas que requieren de un gran desempeño computacional, estos sistemas son conocidos como sistemas computacionales distribuidos. Los sistemas computacionales distribuidos pueden ser clasificados en dos grupos: *sistemas computacionales distribuidos tipo clúster y sistemas computacionales tipo grid*.

***Sistemas computacionales distribuidos tipo CLUSTER***

En la computación cluster, el hardware subyacente consiste en un grupo de estaciones de trabajos o PCs similares (alto grado de homogeneidad), localizadas cercanas unas de otras e interconectadas a través de una red de área local (LAN) de alta velocidad.

Los sistemas de computación tipo cluster se popularizaron cuando la relación precio/desempeño de las computadoras personales y estaciones de trabajo mejoró considerablemente. En un momento dado, resultó técnica y económicamente atractivo construir sistemas de *supercómputo* mediante la interconexión de computadoras simples a través de una red de computadoras de alta velocidad. Virtualmente en todos los casos, la *computación cluster* es utilizada para la programación paralela, en la que un programa sencillo de alta demanda computacional es ejecutado en *paralelo* o *concurrentemente* en múltiples computadoras conectadas a la red local.

Diagrama

Descripción generada automáticamente

Los *clusters* *Beowulf* basados en el sistema operativo *Linux*. Cada cluster está compuesto por un conjunto de *nodos de cómputo* que son controlados y accesos por medio de un *nodo maestro* único. El *nodo maestro* típicamente controla la asignación de los nodos de cómputo a un programa paralelo en particular (determina a que nodos de cómputo se le asigna el procesamiento de una de las partes del programa); administra la fila de solicitudes de trabajo o tareas en espera (partes del programa en espera de ser procesados por uno de los nodos de cómputo); y provee una interfaz para los usuarios del sistema. En sí, el nodo maestro corre el *middleware* requerido para la ejecución de los programas y la administración del cluster, mientras que los nodos de cómputo no requieren más que un sistema operativo estándar.

Una parte importante del *middleware* está constituido por las librerías que hacen posible la ejecución de programas en paralelo. Muchas de estas librerías proveen las facilidades requeridas para establecer una comunicación avanzada basada en el intercambio de mensajes; sin embargo, no proveen un mecanismo para el manejo de falla de procesos, seguridad, etc.

Como alternativa a esta organización jerárquica, se tiene el modelo simétrico del sistema *MOSIX*. Este sistema provee la apariencia o imagen de un sistema cluster como si este fuera una sola computadora, lo cual transparenta la distribución de las partes del programa (procesos) a ejecutar en paralelo. Los procesos migran dinámica y preventivamente a los nodos del cluster. Este proceso de migración permite que el usuario inicia una aplicación en cualquier nodo (conocido como nodo *hogar* o *home*), después de lo cual puede migrar transparentemente a otros nodos.

***Sistemas computacionales distribuidos tipo GRID***

Un sistema computacional distribuido tipo *grid* consiste en sistemas distribuidos que comúnmente se construyen como una federación de sistemas de cómputo, en el que cada sistema suele estar bajo un dominio administrativo distinto, y en los que los componentes de hardware, software y tecnología de red pueden ser muy diferentes. Los sistemas *grid* son heterogéneos, por lo que no se hacen suposiciones de uniformidad en cuanto al hardware, sistemas operativos, redes, dominios administrativos, políticas de seguridad, etc.

El punto crucial de un *sistema computacional grid* estriba en que los recursos de distintas organizaciones son puestos a la disposición de un grupo de personas o instituciones para que colaboren entre sí. Esta colaboración se implementa como una organización virtual, en la que sus miembros tienen derecho a usar los recursos que provee la organización virtual. Típicamente, los recursos del *grid* consiste en servidores (incluyendo supercomputadoras, posiblemente clusters), medios de almacenamiento, bases de datos, etc. Además, también se pueden compartir otro tipo de recursos que pueden ser conectados a las redes, tales como telescopios, microscopios, sensores, etc.

Diagrama

Descripción generada automáticamente

Como se puede observar en la figura anterior, la arquitectura de Foster consiste en cuatro capas, las cuales pueden ser descritas como sigue:

* *Capa de fabrica* – Provee interfaces para los recursos locales en un sitio en particular. Estas interfaces son diseñadas para compartir estos recursos dentro de la organización virtual. Típicamente realizan funciones para solicitar el estatus y capacidades de los recursos, además de administrar los recursos (cerrar los recursos en uso).
* *Capa de conectividad* – Consiste en los protocolos de comunicación para dar soporte a las transacciones que abarcan recursos múltiples en el sistema *grid*. Por ejemplo, los protocolos que se requieren para transferir datos entre recursos, o para simplemente acceder a un recuso de un lugar remoto. Además, esta capa contiene protocolos de seguridad para autentificar usuarios y recursos. En muchos casos, más que el usuario, son las aplicaciones las que deben ser autentificadas, por lo que resulta indispensable delegar los derechos del usuario a la aplicación.
* *Capa de recurso* – Se responsabiliza de administrar un recurso individual. Usa las funciones provistas por la capa de conectividad y llama directamente a las interfaces que la capa de fábrica pone a disposición. Por ejemplo, esta capa ofrecerá funciones para obtener información sobre la configuración de un recurso en específico, o, en lo general, para desempeñar operaciones específicas tales como crear un proceso o leer datos. Entonces, la capa de recurso es considerada responsable del control de acceso y, por lo tanto, dependerá de la información de identificación y autorización de la capa de conectividad.
* *Capa colectiva* – Controla el acceso a recursos múltiples y típicamente consiste en servicios para el descubrimiento de recursos, asignación y programación en tiempo de las tareas en recursos múltiples, replicación de datos, etc. A diferencia de las capas de conectividad y recurso, las cuales consisten en una pequeña colección estándar de protocolos, la capa de colectividad puede consistir en muchos protocolos diferentes para muchos propósitos diferentes, lo cual refleja el amplio espectro de servicios que se pueden ofrecer a una organización virtual.
* *Capa de aplicación* – Consiste en las aplicaciones que operan dentro de la organización virtual y que usan el ambiente de computación grid.
* Las capas colectivas, de conectividad y recurso forman el corazón de lo que puede llamarse *capa de middleware del grid*. Estas capas, en conjunto, proveen acceso y administración de recursos que potencialmente estás dispersos en varios *sitios* (*unidades administrativas*).

***Sistemas de información distribuidos***

Otra clase importante de sistemas distribuidos se encuentra en organizaciones que poseen una gran cantidad de aplicaciones de red (distribuidas), pero cuya interoperabilidad se dificulta mucho. Muchas de las soluciones middleware son el resultado de trabajar con infraestructuras en la que es más fácil integrar aplicaciones en un sistema de información global en la empresa.

Al tornarse las aplicaciones más sofisticadas y al ser separadas en varios componentes independientes, se volvió evidente que la integración de varias de estas aplicaciones debería efectuarse permitiendo que las aplicaciones se comunicaran directamente unas con otras.

La necesidad de implementar la comunicación entre aplicaciones llevo a desarrollar muchos tipos diferentes de modelos de comunicación. La idea fundamental es que las aplicaciones existentes puedan intercambiar información directamente.

Diagrama

Descripción generada automáticamente

Existen varios tipos de middleware de comunicación. Con el uso de RPC’s (del inglés Remote Procedure Call, o Llamada a Procedimiento Remoto en español), un componente de una aplicación puede enviar una solicitud (cliente) a otro componente de aplicación, lo que resulta en una solicitud empaquetada en forma de un mensaje el cual es enviado a la aplicación destino (servidor). El resultado a la vez será regresado al cliente en forma de RPC.

Al incrementarse la popularidad de tecnología objeto, se desarrollaron técnicas similares a los RPC’s, pero que en lugar de ser orientado a aplicaciones son orientados a objetos remotos. Estas técnicas se llaman RMI’s (Remote Method Invocations, o, en español, Invocaciones a Métodos Remotos).

***Sistemas Penetrantes distribuidos***

Los sistemas distribuidos vistos hasta ahora se caracterizan mayormente por su estabilidad: los nodos son fijos y tienen una conexión a la red más o menos permanente y de alta calidad. Sin embargo, muchas cosas han cambiado con la introducción de dispositivos de cómputo móviles o embebidos. Ahora es común tener que trabajar con sistemas distribuidos en la que la inestabilidad es una constante. Los dispositivos en estos sistemas que llamamos *Sistemas Penetrantes Distribuidos* son frecuentemente pequeños, operados con baterías, móviles y con una sola conexión inalámbrica (no necesariamente aplican todas estas características a todos los dispositivos).

Un sistema penetrante distribuido es parte de nuestro entorno, y, por consecuencia, inherentemente distribuido. Una situación importante es la falta de control administrativo humano. En el mejor de los casos, los dispositivos pueden ser configurados por sus propietarios, pero ellos deben descubrir su ambiente e integrarse lo mejor posible a los demás dispositivos. Las aplicaciones penetrantes deben cumplir los siguientes requerimientos:

1. Deben aceptar cambios contextuales (deben estar conscientes de que el ambiente cambia continuamente).
2. Deben promover una composición adaptable (los dispositivos pueden ser utilizados en diferentes formas por diferentes usuarios).
3. Por omisión, deben compartir recursos. Al ser agregados al sistema generalmente deben acceder y proveer información.

Ejemplos comunes de sistemas penetrantes distribuidos son los smartphones (teléfonos inteligentes), PDA’s, video videojuegos, etc.

## 1.4 Arquitectura

Se denomina arquitectura a descripciones de altos niveles que se le hace a un sistema, aquí se define por el objetivo del diseño, la descomposición en subsistemas, plataformas de software y hardware, control de acceso, políticas de control y entre otras estrategias que son consideradas por el equipo de desarrollo para el funcionamiento del sistema.

***¿Qué es un sistema distribuido?***

Es todo lo que nos permite el procesamiento de la información este ubicado en distintas computadoras de forma en que parezca en que esta información este en una sola computadora. Esto ayuda al desarrollador pensar en los recursos que se necesitaran (como software y hardware) al estar en un sistema distribuido, todo esto se podrá compartir.

***¿Qué es una arquitectura en los sistemas distribuidos?***

Es considerada como un estilo de tecnología de información, la arquitectura nos permite integrar el negocio como si fuese un conjunto de servicios interrelacionados. Cada tipo de arquitectura tiene una distribución dependiendo de su trabajo, donde cada diseño es para que cumpla con su respectiva función y donde debe de ser eficaz o eficiente a la hora de aplicarla en el momento preciso. Dentro de esta representación de sistema nos encontraremos correlaciones de funciones entre el software y hardware, y la interacción humana con estos componentes.

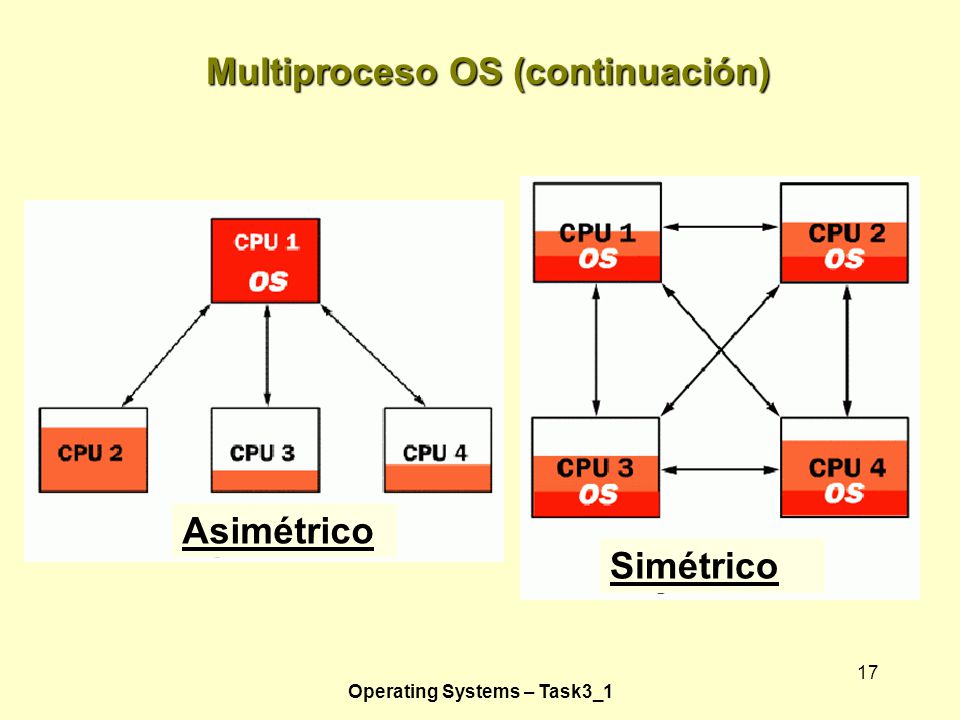
***Tipos de arquitectura de sistemas distribuidos***

Existen distintos tipos de arquitectura en los sistemas operativos y entre estos se mencionarán algunos de los que son más conocidos para los desarrolladores de sistemas distribuidos.

***Arquitectura multiprocesador***

Es considerado el modelo mas sencillo de todos los sistemas distribuidos, este sistema ejecuta muchos procesos en distintos procesadores.

Esta arquitectura se define en dos tipos. La clasificación está basada en que si cuenta con características específicas o por el contrario no tiene bien definidas sus características que no pueda diferenciar el usuario. Cualquiera de estas dos que se usen, garantiza el proceso y control tanto del sistema operativo como el del proceso maestro que es el encargado de dirigir todo el proceso: Multiprocesamiento simétrico SMP y multiprocesamiento asimétrico.



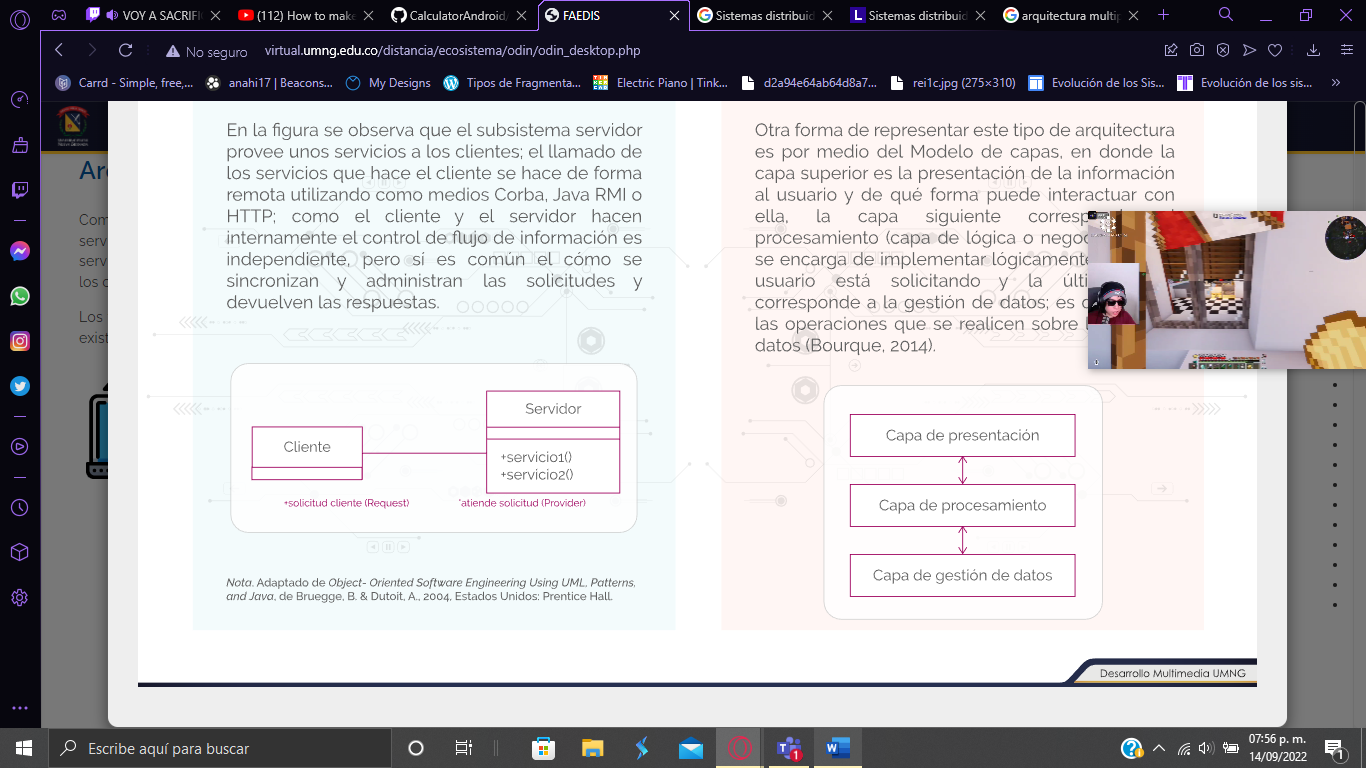
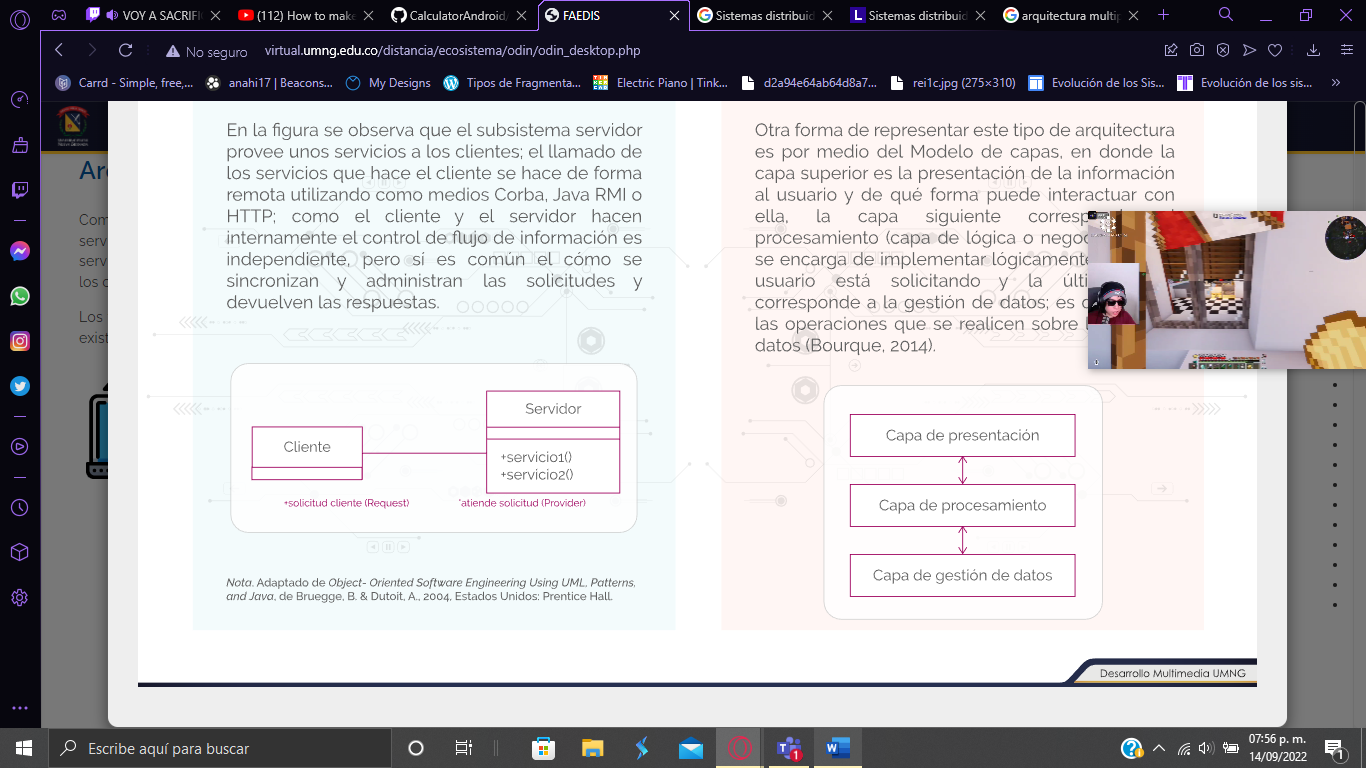
***Arquitectura cliente-servidor***

Esta arquitectura esta compuesta por un servidor y algunos clientes que acceden a los servicios dados por el servidor. El servidor no conoce nada sobre los clientes que utilizan estos servicios: los clientes solo necesitan saber que servidores están disponibles. Es decir, un dispositivo en particular llamada “servidor” puede realizar algunas actividades, denominadas servicios.

Un claro ejemplo es que ofrece archivos a través de la red, tiene una capacidad de ejecutar ciertos comandos o enrutar datos a una impresora y el cliente es la computadora que requiere el servicio. La computadora conocida principalmente por el servidor que brinda, puede denominarse servidor de impresión, servidor de archivos, etc.

Otro ejemplo sería el internet, donde un cliente accede fácilmente a millones de datos de múltiples servidores

Recordemos que esta arquitectura se define con base al numero de capas: existen 1, 2, 3, 4 y N capas.

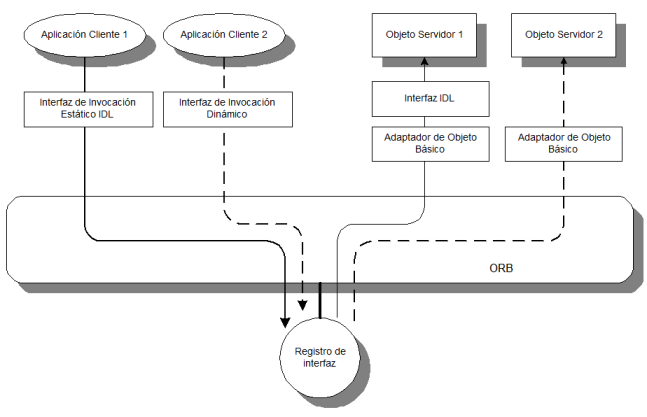
 

***Arquitectura de sistemas distribuidos***

Su objetivo no es etiquetar un cliente o un servidor, sino que el sistema se vea como un conjunto de objetos. La característica que tienen es que una parte de los objetos proporcionan una interfaz a un conjunto de servicios.

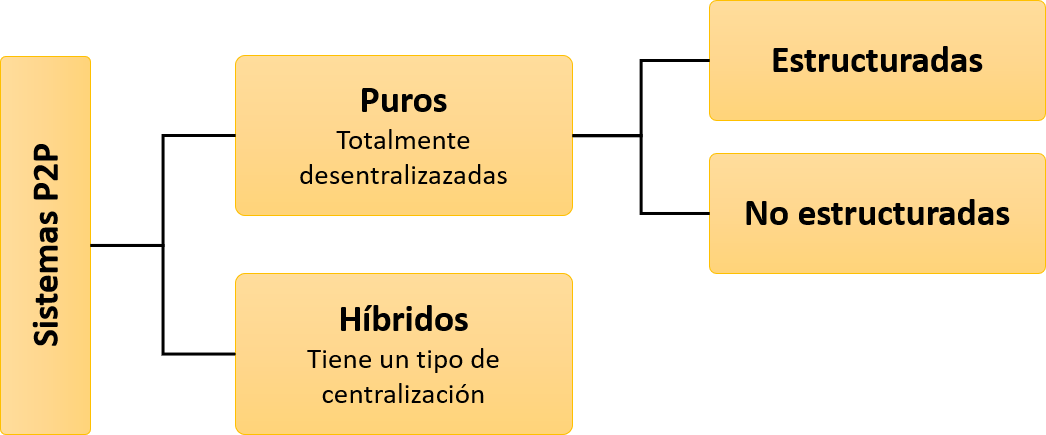
***CORBA***

Common Object Request Broker Architecture, es un middleware que realiza tareas de intermedio en las peticiones realizan los objetos. La importancia de este es que radica en objetos que pueden estar, por un lado, implementados en distintos lenguajes de programación, son ejecutados en diferentes plataformas y sus nombres no son conocidos por otros objetos.



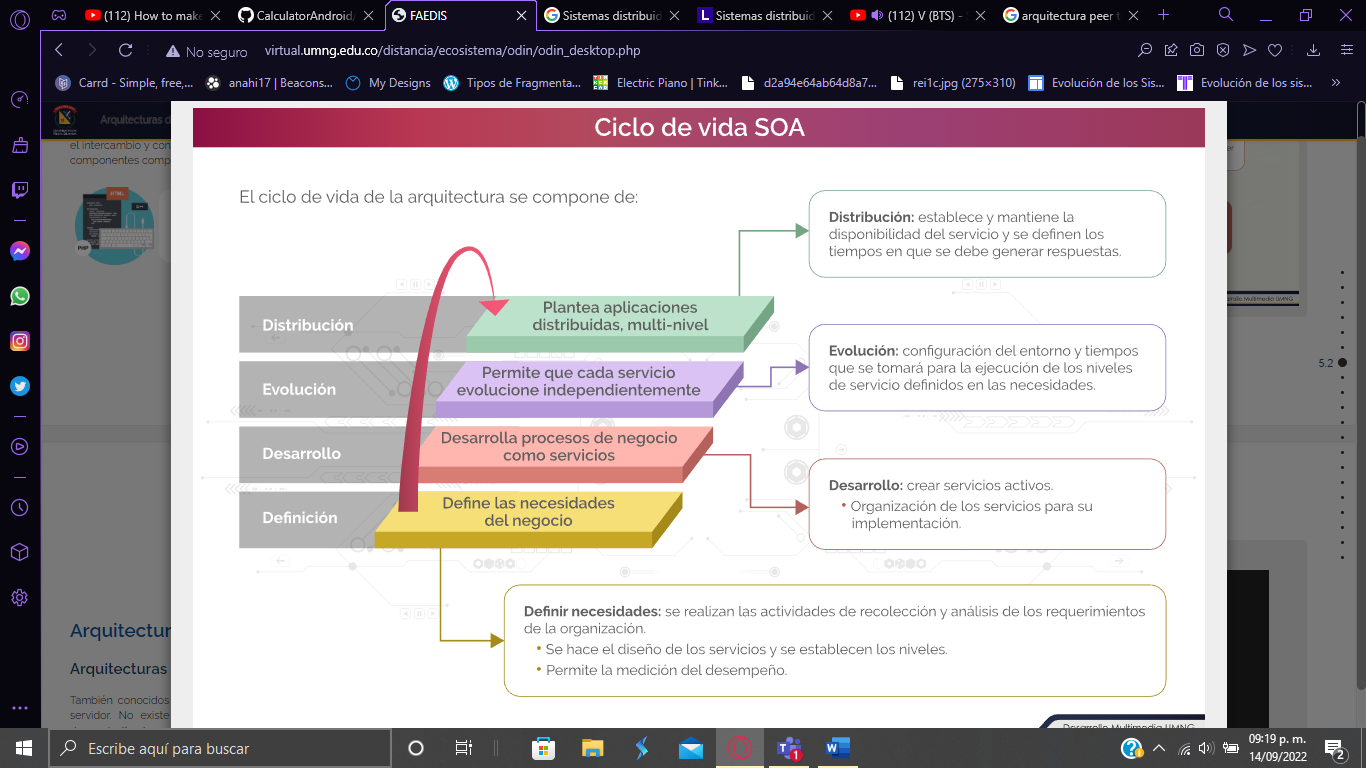
***Arquitectura peer to peer***

En P2P cualquier nodo tiene la capacidad de tomar el control y vocería del servidor. Aquí no existe una distinción entre quien es el cliente y quien es el servidor, este tipo de arquitectura puede ser descentralizada o semicentralizada. Asume que cada computadora tenga capacidades similares y ninguna maquina se quede sirviendo a las demás.



***Arquitecturas de sistemas orientadas a servicios***

Se le considera como un estilo de arquitectura donde permite hacer servicios interrelacionados (tareas repetitivas). Esto surgió del problema que se presentaba antes por el acceso solamente a través de internet y acceso a los datos que tenían compartidas las organizaciones. Se busco una solución y de eso nació lo que hoy se le conoce como web services que nos permite publicar todos los servicios que consideren necesarios.



***Arquitectura de red***

El diseño de una red de computadoras siempre ha sido un problema para nosotros. La misma red de naturaleza implica la interconexión entre las computadoras, cuanto mayor sea esta, mayor será el número de computadoras. Existen varias soluciones que son adoptadas dependiendo de aspectos de alcances, velocidad, topología, tiempo de respuesta, etc.

Esto da a que se ajuste el diseño de la red a las necesidades específicas a cada situación dando lugar a una heterogeneidad mayor en la interconexión de dichas redes.

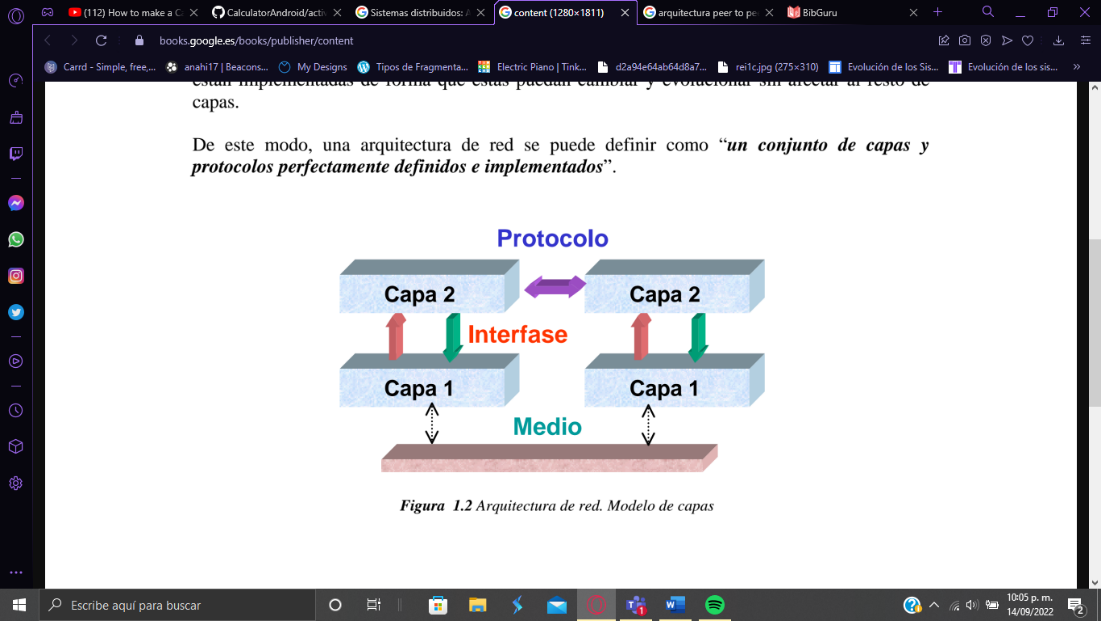
La complejidad y necesidad de la interconexión de estos tipos de sistemas heterogéneos ha dado un lugar al desarrollo de arquitecturas de red abiertas. Un sistema abierto es capaz de interconectarse con otros de acuerdo a las normas establecidas. Por lo tanto, el objetivo de la arquitectura abierta es un conjunto de normas que permitan a los sistemas comunicarse entre sí.

El diseño de la red de computadoras es complejo como para que necesariamente utilicemos una metodología estructurada en una solución adaptada. En otros aspectos de la complejidad, la técnica que se emplea aquí es la división de capas o niveles. Estas capaz son de manera jerarquizadas y dividen el problema en partes mas fáciles. Cada capa está proporcionando un conjunto de servicios perfectamente normalizados a la capa superior y utiliza a la vez los servicios proporcionados por la capa inferior. Los detalles de implementación quedan ocultos en cada capa.

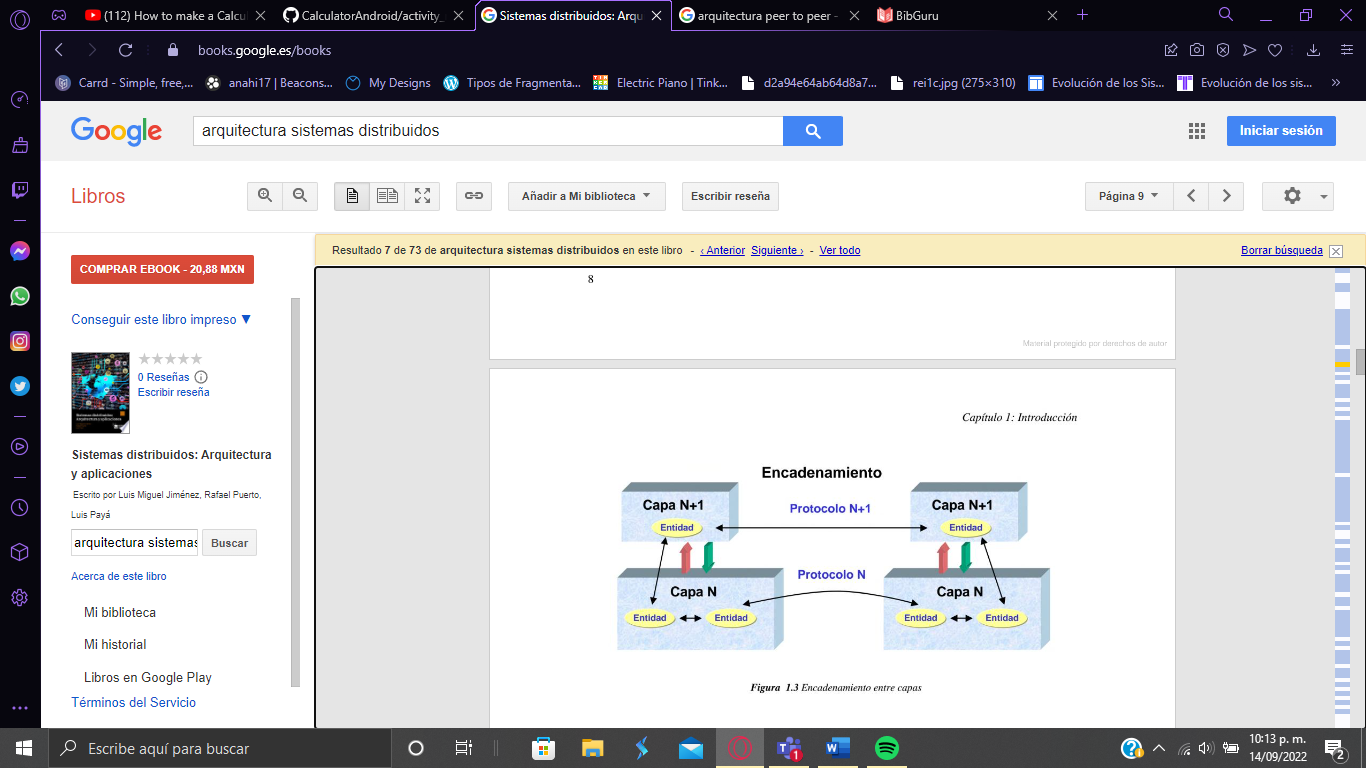
Cada una de las capas contiene entidades que implementan la funcionalidad a cuya capa. Estas entidades podrían ser procesos de software o dispositivos físicos como hardware o una mezcla de ambos.

La comunicación que hay entre las capas se denomina como interfase y esta constituye el camino real que es seguido por la información en una transmisión. Existe un camino lógico que establece entre las entidades que se están comunicando en equipos distintos y pertenecen a la misma capa. Esto se le llama entre pares, viene fijada por el protocolo.

Para la comunicación lógica las entidades deben de utilizar servicios proporcionados por las entidades de la capa inferior. Cabe destacar que no todas las funciones que realiza una capa deben de ser accesibles como servicios por la capa superior. Tampoco se especifica como se implementan de forma que estas pueden sufrir de cambios y evolucionar sin afectar al resto de capas.



Para definir configuraciones más complejas en las cuales existían múltiples interconexiones en una transmisión de datos, se incorporan también el encadenamiento de capas lo cual nos permite que existan varias entidades dentro de la misma capa de una misma estación, usando sus propios protocolos, cada una de estas capas se comunican con estaciones diferentes.



# Conclusiones

Los sistemas distribuidos son una maravilla en la actualidad, pero también representan un riesgo, si bien contamos con ellos en innumerables sitios, como lo puede ser sencillamente, en internet con Google, estos sistemas también tienen que ser observados con cautela y poner una atención muy especifica para poder reparar los problemas que ellos conllevan cuanto antes nos sea posible, claramente estos sistemas cada vez representan una complejidad mayor y lo seguirán haciendo ya que nuestra tecnología también avanza con ello, dejando atrás varios sistemas y consiguiendo unos mucho mejores que sustituyen a los antiguos por resolver problemas que talvez parezcan insignificantes, pero viéndolos desde otra perspectiva, nos han permitido seguir avanzando tanto en tecnologías como en las conexiones que realizamos, beneficiándonos no solo nosotros, si no todas las personas del mundo, debido a una atención mucho más profesional, rápida y eficiente.

# Herramientas y recursos

Para realizar el documento presente se utilizó:

Internet para la búsqueda de información

Word para la elaboración del documento

Scribbr para la elaboración de formatos apa

Equipo de cómputo para generar el archivo de forma digital

# Bibliografía

Cerinza, N. G. (n.d.). FAEDIS. Edu.co. Retrieved September 15, 2022, from <http://virtual.umng.edu.co/distancia/ecosistema/odin/odin_desktop.php?path=Li4vb3Zhcy9pbmdlbmllcmlhX2luZm9ybWF0aWNhL2luZ2VuaWVyaWFfZGVfc29mdHdhcmVfaWkvdW5pZGFkXzIv>

Corvo, H. S. (2020, March 1). Sistemas distribuidos: características, arquitectura, tipos, objetivos, aplicaciones. Lifeder. <https://www.lifeder.com/sistemas-distribuidos/>

Jiménez, L. M., Puerto, R., & Payá, L. (2017). Sistemas distribuidos: Arquitectura y aplicaciones. Universidad Miguel Hernández.

Losada, S. (2020, 8 julio). *Qué es un sistema distribuido y qué ventajas aporta su funcionamiento*. OpenWebinars.net. Recuperado 14 de septiembre de 2022, de https://openwebinars.net/blog/que-es-un-sistema-distribuido/

*¿Qué es un sistema distribuido?* (s. f.). Atlassian. Recuperado 14 de septiembre de 2022, de https://www.atlassian.com/es/microservices/microservices-architecture/distributed-architecture#:%7E:text=Un%20sistema%20distribuido%20es%20un,error%20centrales%20de%20un%20sistema.

11, 12 y 13 Tipos de Sistemas Distribuidos e Investigación 01 - PDF Descargar libre. (649, noviembre 12). Recuperado 14 de septiembre de 2022, de https://docplayer.es/34654649-11-12-y-13-tipos-de-sistemas-distribuidos-e-investigacion-01.html

Sanchez, R. C. C. (s. f.). Tipos de Sistemas Distribuidos. prezi.com. Recuperado 14 de septiembre de 2022, de https://prezi.com/4\_xp3db5e5se/tipos-de-sistemas-distribuidos/

Universidad de Guadalajara. (s. f.). Tipos de sistemas distribuidos. Recuperado 14 de septiembre de 2022, de https://es.slideshare.net/AleksNet/tipos-de-sistemas-distribuidos