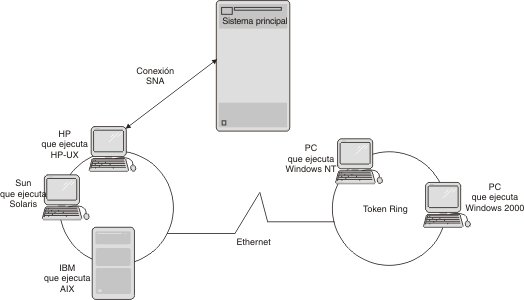
<http://www.ibm.com/support/knowledgecenter/es/SSAL2T_7.1.0/com.ibm.cics.tx.doc/reference/r_eg_open_distd_sys.html>

1. Ejemplo de sistema distribuido abierto

Este sistema contiene dos redes de área local (LAN) conectadas entre sí. Una LAN está formada por estaciones de trabajo UNIX® de diversos fabricantes; la otra LAN consta principalmente de varios PC que ejecutan diversos sistemas operativos de PC. Una de las LAN también está conectada a un sistema principal mediante una conexión SNA.

*Ejemplo de sistema distribuido abierto*



<http://www.monografias.com/trabajos16/sistemas-distribuidos/sistemas-distribuidos.shtml#MIDDLE>

1. Cuál es el papel del middleware en un sistema distribuido.

Capa de software intermedio entre el cliente y el servidor. Es la capa de software que nos permiten gestionar los mecanismos de comunicaciones. Ejemplo si se hace la petición de una página web desde un browser en el cliente, el middleware determina la ubicación y envía una petición para dicha página. El servidor Web, interpreta la petición y envía la página al software intermedio, quien la dirige al navegador de la máquina cliente que la solicitó.

Existen dos tipos:

 **Software intermedio general**. Servicios generales que requieren todos los clientes y servidores, por ejemplo: software para las comunicaciones usando el TCP/IP, software parte del sistema operativo que, por ejemplo, almacena los archivos distribuidos, software de autenticación, el software intermedio de mensajes de clientes a servidores y viceversa.

 **Software intermedio de servicios**. Software asociado a un servicio en particular, por ejemplo: software que permite a dos BD conectarse a una red cliente/servidor (ODBC: Conectividad abierta de BD), software de objetos distribuidos, por ejemplo la tecnología CORBA permite que objetos distribuidos creados en distintos lenguajes coexistan en una misma red (intercambien mensajes), software intermedio para software de grupo, software intermedio asociado a productos de seguridad específicas (Conexiones Seguras: Sockets), etc.

**Características:**

 Independiza el servicio de su implantación, del sistema operativo y de los protocolos de comunicaciones.

 Permite la convivencia de distintos servicios en un mismo sistema.

 Permite la transparencia en el sistema.

 Modelo tradicional: Monitor de teleproceso o CICS, Tuxedo, Encina.

 Modelo OO: CORBA.

<http://www.unap.edu.pe/cidiomas/licing/pdf/sd.pdf>

1. Muchos sistemas en red están organizados en términos de back office y el front office. ¿Cómo es que las empresas coinciden con el punto de vista coherente que demandamos para un sistema distribuido?

Las empresas coinciden con un sistema distribuido por los objetivos y características que contienen a la hora de implementarlo.

Objetivos:

* Transparencia
* Fiabilidad
* Rendimiento
* Escalabilidad
* Flexibilidad

Características:

* Heterogeneidad
* Extensibilidad y apertura
* Seguridad
* Escalabilidad
* Tratamiento de fallos
* Concurrencia
* Transparencia

<https://www.codifica.me/transparencia-sistemas-distribuidos/>

1. un sistema distribuido es proporcionar al usuario y a las aplicaciones una visión de los recursos del sistema como gestionados por una sola máquina virtual. La distribución física de los recursos es transparente.

Proporcione ejemplos de distintos tipos de transparencia:

Existen **ocho formas de transparencia**, las que más consideración tienen en un sistema distribuido son la transparencia a nivel de acceso y la transparencia a nivel de localización.

• **Transparencia de Acceso.**  
Hace referencia al acceso a las entidades del sistema, tanto si son locales como si son remotas deben seguir ocultando que son entidades diferentes, formando una única entidad, así pues el acceso al sistema por el usuario debe ser único para toda la entidad.

• **Transparencia de localización.**  
Engloba la transparencia de nombre y la transparencia de movilidad, la primera exige que cualquier referencia a la entidad no deba exponer ninguna indicación de la localización de la entidad. La segunda hace referencia a que el sistema sea igual de consistente independientemente desde dónde el usuario haga referencia al sistema.

<http://www.unap.edu.pe/cidiomas/licing/pdf/sd.pdf>

1. Porque a veces es tan difícil ocultar la ocurrencia y la recuperación de fallas en un sistema distribuido?

Recuperación Frente a Fallos: Una vez que fue detectado un fallo y que se ha decidido arreglarlo, hay que encontrar la mejor manera de hacerlo, y además, de recuperar el estado del sistema antes de que ocurriera el fallo; esto requiere del software adecuado para poder reconstruir o bien retractar los cambios que no fueron completados al momento en que fue interrumpido el sistema, un ejemplo de esto lo podemos ver en los sistemas manejadores de bases de datos, que se sirven de una bitácora de las transacciones que se realizan y de acuerdo a esta bitácora se decide reconstruir o retractar las transacciones hechas sobre la base de datos antes de que se interrumpiera el funcionamiento de la misma.

<http://fa-mas-dbms.blogspot.com.co/>

Una ocurrencia de un registro almacenado está formada por un grupo de ocurrencias de campos almacenados entre sí (una ocurrencia para cada tipo distinto de parte).

Un **archivo almacenado** es el conjunto (con nombre) de todas las ocurrencias de un tipo de registro almacenado.

<https://www.u-cursos.cl/ingenieria/2013/2/CC5303/1/material_docente>

1. Porque no siempre es buena idea implementar el grado más alto de trasparencia?

La transparencia tiene límites y grados.

No siempre es posible esconder todo Ejemplo: desfase temporal de comunicación entre puntos muy distantes (delay). Siempre hay un trade-off entre transparencia y rendimiento Ejemplo: Bases de datos replicadas world-wide. Un caso particular son los servidores DNS que pueden tomar días en actualizarse, lo cual no se oculta al usuario.

<http://augcyl.org/?page_id=231>

1. Que es un sistema distribuido abierto, y que beneficios proporciona la apertura?

Un sistema informático es abierto si el sistema puede ser extendido de diversas maneras. Un sistema puede ser abierto o cerrado con respecto a extensiones hardware (añadir periféricos, memoria o interfaces de comunicación, etc…) o con respecto a las extensiones software (añadir características al sistema operativo, protocolos de comunicación y servicios de compartición de recursos, etc…). La apertura de los sistemas distribuidos se determina primariamente por el grado hacia el que nuevos servicios de compartición de recursos se pueden añadir sin perjudicar ni duplicar a los ya existentes.

<http://sistemas-distribuidos-unerg.blogspot.com.co/2008/10/caractersticas-principales-de-los.html>

**Apertura (opennesss)**  
Un sistema informático es abierta si el sistema puede ser extendido de diversas maneras. Un sistema puede ser abierto o cerrado con respecto a extensiones hardware (añadir periféricos, memoria o interfaces de comunicación, etc...) o con respecto a las extensiones software (añadir características al sistema operativo, protocolos de comunicación y servicios de compartición de recursos, etc...). La apertura de los sistemas distribuidos se determina primariamente por el grado hacia el que nuevos servicios de compartición de recursos se pueden añadir sin perjudicar ni duplicar a los ya existentes.

<http://www.unap.edu.pe/cidiomas/licing/pdf/sd.pdf>

1. Describa precisamente lo que significa un sistema escalable.

Escalabilidad: La escalabilidad es una de las características más importantes para los desarrolladores de un sistema distribuido. Se dice que un sistema es escalable si logra conservar su efectividad cuando hay el número de recursos y el número de usuarios incrementa significativamente. La escalabilidad de un sistema pude medirse en tres aspectos diferentes: Con respecto a su tamaño: lo que significa que se pueden agregar más usuarios y más recursos al sistema de una manera muy fácil. Con respecto a su localización o área de implementación: lo que significa que tanto los usuarios como los recursos pueden estar en locaciones remotas y separadas el uno del otro. Con respecto a su administración: lo que significa que puede ser fácil de administrar a pesar de que se utiliza en diferentes organizaciones independientes que cuentan con diferentes políticas de seguridad y que hacen un uso particular del sistema. Desafortunadamente, un sistema que es escalable en uno o más de estos aspectos por lo general afecta el rendimiento del sistema conforme al crecimiento del mismo.

<http://www.unap.edu.pe/cidiomas/licing/pdf/sd.pdf>

1. La escalabilidad puede lograrse aplicando distintas técnicas, cuáles son esas técnicas?

Técnicas de Escalabilidad Una vez que mencionamos ya los problemas de la escalabilidad, analizaremos algunas maneras de solucionar dichos problemas. Como los problemas de escalabilidad de los sistemas distribuidos se manifiestan como problemas de rendimiento causados por la capacidad limitada de servidores y de las redes de comunicaciones, existen solamente tres técnicas de escalabilidad: eliminar la latencia de las comunicaciones, distribución y replicación.

Técnicas:

1. Eliminar la latencia de las comunicaciones es útil en el caso de querer lograr la escalabilidad geográfica de un sistema, la idea básica es simple: tratar de evitar la espera de respuestas a las peticiones que se hagan a servicios remotos lo más que se pueda. Esencialmente, esto significa que se tiene que construir la aplicación que realiza las peticiones de tal manera que use solamente métodos de comunicación asíncronos, es decir, que el cliente envía la petición al servidor, mientras espera la respuesta el cliente aprovecha ese tiempo para realizar tareas locales más importantes y cuando recibe la respuesta del servidor, el proceso que se estaba realizando se interrumpe y se atiende la respuesta recibida. Esta solución parece fácil, sin embargo, hay muchas aplicaciones que no pueden hacer un uso efectivo de la comunicación asíncrona, por ejemplo, aplicaciones interactivas en las que el usuario no tiene nada mejor que hacer más que esperar la respuesta del servidor, por que esta se tiene que dar lo más rápido que sea posible. En dichos casos, es mucho mejor solucionar el problema de la latencia reduciendo el tráfico generado entre el cliente y el servidor cuando se comunican entre sí; esto lo podemos lograr moviendo parte de la computación que normalmente se hace del lado del servidor al cliente, para que así sea el mismo proceso que hace la solicitud del servicio quien tenga que procesarlo.
2. Eliminar la latencia de las comunicaciones es útil en el caso de querer lograr la escalabilidad geográfica de un sistema, la idea básica es simple: tratar de evitar la espera de respuestas a las peticiones que se hagan a servicios remotos lo más que se pueda. Esencialmente, esto significa que se tiene que construir la aplicación que realiza las peticiones de tal manera que use solamente métodos de comunicación asíncronos, es decir, que el cliente envía la petición al servidor, mientras espera la respuesta el cliente aprovecha ese tiempo para realizar tareas locales más importantes y cuando recibe la respuesta del servidor, el proceso que se estaba realizando se interrumpe y se atiende la respuesta recibida. Esta solución parece fácil, sin embargo, hay muchas aplicaciones que no pueden hacer un uso efectivo de la comunicación asíncrona, por ejemplo, aplicaciones interactivas en las que el usuario no tiene nada mejor que hacer más que esperar la respuesta del servidor, por que esta se tiene que dar lo más rápido que sea posible. En dichos casos, es mucho mejor solucionar el problema de la latencia reduciendo el tráfico generado entre el cliente y el servidor cuando se comunican entre sí; esto lo podemos lograr moviendo parte de la computación que normalmente se hace del lado del servidor al cliente, para que así sea el mismo proceso que hace la solicitud del servicio quien tenga que procesarlo.

En la mayoría de los casos, los problemas de escalabilidad los vemos reflejados en el rendimiento del sistema, por lo que generalmente la replicación de los componentes del sistema distribuido puede resultar ser una buena idea. La replicación aumenta la disponibilidad de los componentes del sistema y además ayuda a balancear la carga entre los componentes que se replican, con lo que se logra una mejora del rendimiento del sistema. Si consideramos un sistema que se encuentra distribuido en una red muy extensa, el hecho de tener una copia de algún componente más cerca, también mejora el rendimiento del sistema puesto que soluciona los problemas de latencia de las comunicaciones que ya mencionamos anteriormente.

1. Una forma especial de replicación es el Cacheo, el cual consiste en guardar una copia de algún recurso (por lo general, de datos) de forma temporal en un lugar cercano al cliente, para que éste lo pueda acceder más fácilmente. En contraste con la replicación, el cachear un recurso es una decisión que es tomada por el cliente, y no por los diseñadores del sistema. Un problema muy serio que puede traer el cacheo, es que las copias que se hacen del recurso (o de datos) pueden no actualizarse a su debido tiempo, por lo que al haber varias copias diferentes del mismo recurso, lo que provoca problemas de consistencia dentro del sistema; el nivel de inconsistencia que pueda ser tolerado por un sistema depende del uso que se le da a al recurso que se está replicando, en el caso de páginas Web estáticas, por ejemplo, pudiese ser bastante tolerable, sin embargo, en el caso de páginas dinámicas o de sistemas en tiempo real, la replicación puede traer consigo muchos problemas.

<https://carolromero.wordpress.com/2013/08/17/la-organizacion-virtual/>

1. Explique lo que quiere decir organización virtual, y proporcione un indicio sobre cómo es que tales organizaciones podrían implementarse.

La Organización Virtual o también llamada la Organización en Red, se basa en la contratación de empresas independientes para realizar aquellas actividades en las cuales son mejores asociándose en una red, que actúa como una sola empresa.

Las organizaciones virtuales tienen como objetivo principal la flexibilidad, y son muy parecidas a las organizaciones en trébol y en red. Son organizaciones orientadas al mercado, que se configuran como un conjunto de cadenas de valor relacionadas entre proveedores, clientes, competidores, otras organizaciones y la propia empresa.

De esta manera las organizaciones podrían implementarse, ejemplo:

Entre los primeros modelos de organizaciones virtuales podemos mencionar a mercados electrónicos como eBay (subastas directas), PayPal (pagos electrónicos entre empresas o individuos), que siguen un modelo de cooperación directa entre individuos o pequeñas organizaciones, supermercados online como Amazon (libros, electrónicos, general), PeaPod (comestibles), NetFlix (películas), Skype, Vonage y Net2Phone (comunicaciones) banca online y toda clase de servicios profesionales y asistenciales a distancia.

<http://sdequipo2.blogspot.com.co/2009/07/unidad-3-transacciones.html>

1. Cuando se aborta una transacción, hemos dicho que el mundo se restaura a su estado anterior, como si la transacción nunca hubiera ocurrido. Mentimos. Proporcione un ejemplo donde reiniciar el mundo resulte imposible.

Motivos del uso de transacciones: Los sistemas distribuidos son potencialmente muy fiables debido a la posibilidad de proveer redundancia y autonomía de recursos en diferentes nodos, esto permite detectar y localizar fallas, sin embargo comúnmente tenemos varios aspectos que representan problemas para la integridad de los recursos y que a su vez motivan el uso de transacciones:

1. Dificultad para mantener consistencia en los datos.

2. Una misma vía de comunicación no siempre puede ser utilizada para proveer interacción entre 2 procesos.

3. Requerimientos de procesamiento en paralelo.

4. Manejo interactivo de uno o más usuarios

Un ejemplo sería un sistema distribuido donde no presenten fallas y aspectos que representen problemas para la integridad de los recursos.

<http://sdequipo2.blogspot.com.co/2009/07/unidad-3-transacciones.html>

1. Ejecutar transacciones anidadas requiere cierta coordinación. Explique en realidad lo que debe hacer un coordinador.

Consiste en una serie de modificaciones (transacciones) a un determinado recurso del sistema (por ejemplo una base de datos) y en donde se define un punto de inicio (Begin Tran) y un punto de terminación que define un bloque entre el conjunto de operaciones que son realizadas.  
  
Dentro de este proceso en bloque los demás usuarios no pueden modificar nada hasta que no se presente un estado estable de los datos, esto ocasiona inconsistencia temporal y conflictos. Para evitar lo anterior se implementan dos maneras diferentes:  
  
\* Ejecutar transacciones serializadas. Es un sistema que permite el procesamiento de transacciones en forma secuencial o serializado y consiste en asignarle una secuencia a cada transacción, este proceso reduce el rendimiento del sistema.  
  
\* Ejecutar transacciones calendarizadas. Es un sistema que permite el proceso de transacciones asignándole tiempos de procesamiento el cual permite incrementar el rendimiento del sistema ya que se ejecuta un máximo de proceso en forma concurrente y no a través de una serie.

<http://www.unap.edu.pe/cidiomas/licing/pdf/sd.pdf>

1. Argumentamos que la trasparencia de distribución puede no ocurrir en sistemas masivos. Esta aseveración no es verdadera para todos los tipos de transparencia. Proporcione un ejemplo.

No es transparente un sistema donde el acceso a los archivos remotos se realice mediante:

El establecimiento explícito de una conexión en la red con un servidor remoto. El envío posterior de mensajes, donde el acceso a los servicios remotos será distinto al acceso a los servicios locales.

Con todo esto en mente es posible diseñar un sistema que cuente con las características necesarias para lograr la transparencia en tantos aspectos como sea posible. Los dos más importantes son la transparencia de acceso y la transparencia de ubicación, la primera se relaciona con la forma en que representamos los datos en un sistema distribuido, es importante presentar al usuario o a los programadores el acceso indistinto a recursos locales o remotos, sin que este se de cuenta de la ubicación de los mismos, lo que al mismo tiempo nos conduce a tener transparencia de ubicación dentro del sistema.

Libro sistemas distribuidos por: Andrew S. Tanenbaum

1. Nosotros ya le dimos algunos ejemplos de sistema masivos distribuidos: sistemas caseros, sistemas electrónicos para el cuidado de la salud, y redes de monitoreo. Amplié esa lista con más ejemplos.

Existen diferentes tipos de sistemas distribuidos, los cuales pueden clasificarse como orientados hacia el soporte de cálculos, procesamiento de información, y masivos. Los sistemas de cómputo distribuidos generalmente se utilizan para aplicaciones de alto rendimiento que se originan de modo habitual en el campo del cómputo en paralelo. Una amplia clase de sistemas distribuidos puede encontrarse en los ambientes tradicionales de oficina, donde vemos a las bases de datos representando un papel importante. Por lo general, los sistemas de procesamiento de transacciones se utilizan en estos ambientes. Por último, una clase de sistemas distribuidos de reciente aparición es aquella donde los componentes son pequeños y el sistema se forma a la medida, pero que sobre todo no es manejado por un administrador de sistemas. Esta última clase generalmente es representada por ambientes de cómputo ubicuos.

<https://www.google.com.co/search?q=que+es+un+equipo+analogo&espv=2&source=lnms&tbm=isch&sa=X&ved=0ahUKEwilzeSdvIjSAhXHMSYKHXbvCYQQ_AUICCgB&biw=1366&bih=613#imgrc=z-1bJuoai5r0SM>:

1. Sistema casero con un servidor que permita la conexión de un cliente inalámbrico. Este último se conecta a un equipo analógico de audio y video y trasforma los flujos de medios digitales en una salida analógica. El servidor se ejecuta en una maquina por separado, posiblemente conectada a internet.

