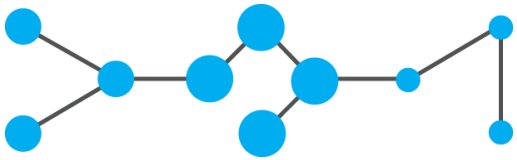
Sistemas de Bases de Datos 2

2024

Ing. Luis Alberto Arias Solórzano

Unidad 5

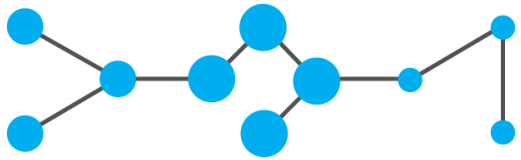
BD Distribuidas



¿Por qué son necesarias las bases de datos distribuidas?

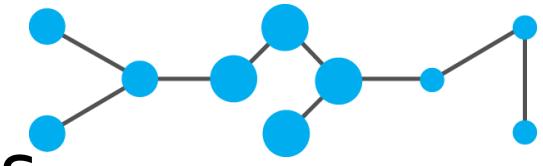
- La respuesta clasica a esta pregunta es que las empresas ya están generalmente distribuidas al menos de manera lógica (en divisiones, departamentos, grupos de trabajo, etcétera) y es muy probable que también lo estén de manera física (en plantas, fábricas, laboratorios, etcétera).
- De esto deducimos que por lo general también los datos ya están distribuidos, ya que cada unidad organizacional dentro de la empresa mantendrá naturalmente los datos que son importantes para su propia operación.
- Por lo tanto, el valor de la información total de la empresa está dividido en lo que a veces llamamos ***islas*** *de información.*
- Y lo que hace un sistema distribuido es proporcionar los *puentes* necesarios para conectar a esas islas entre sí. En otras palabras, permite que la estructura de la base de datos refleje la estructura de la empresa —los datos locales son conservados localmente en el lugar donde pertenecen de manera más lógica— y al mismo tiempo, permite tener acceso a datos remotos cuando sea necesario.
- El arreglo distribuido combina la **eficiencia de procesamiento** (los datos se mantienen cerca del punto en donde se usan más frecuentemente) con una **mayor** accesibilidad (es posible acceder por medio de la red de comunicaciones).
- Probablemente, el mayor beneficio de los sistemas distribuidos es que permiten que la estructura de la base de datos refleje la estructura de la empresa.

BD Distribuidas
Objetivos



El principio fundamental de las BD distribuidas nos conduce a doce reglas complementarias u objetivos:

1. Autonomía local.
2. No dependencia de un sitio central.
3. Operación continua.
4. Independencia de ubicación.
5. Independencia de fragmentación.
6. Independencia de replicación.
7. Procesamiento de consultas distribuidas.
8. Administración de transacciones distribuidas.
9. Independencia de hardware.
10. Independencia de sistema operativo.
11. Independencia de red.
12. Independencia de DBMS.



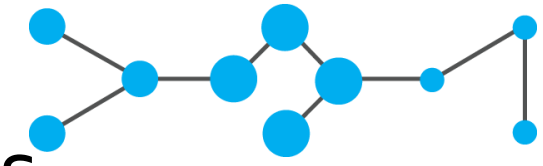
BD Distribuidas
Objetivos

1. **Autonomía local**

- Los sitios en un sistema distribuido deben ser **autónomos.** La autonomía local significa que todas las operaciones en un sitio dado están controladas por ese sitio; ningún sitio *X* debe depender de algún otro sitio F para su operación satisfactoria (ya que de lo contrario, cualquier falla en el sitio y podría significar que el sitio *X* no pueda ejecutar operaciones, aun cuando no haya nada malo en el propio sitio *X*, lo que es obviamente una situación indeseable).

- La autonomía local también implica que los datos locales son poseídos y administrados localmente con contabilidad local: todos los datos pertenecen "realmente" a alguna base de datos local, aun cuando estén accesibles desde otros sitios remotos. Por lo tanto, la seguridad, integridad y representación de almacenamiento de los datos locales permanecen bajo el control y jurisdicción del sitio local.

- De hecho, el objetivo de autonomía local no es totalmente alcanzable; existen varias situaciones en las que un sitio *X* dado *debe* transferir un determinado grado de control a algún otro sitio y. Por lo tanto, el objetivo de autonomía queda establecido con mayor precisión como: los sitios deben ser autónomos **en el mayor grado posible.**



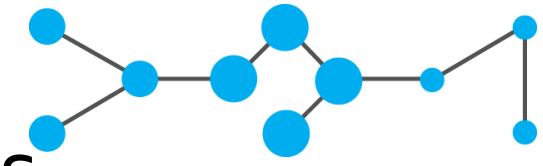
BD Distribuidas
Objetivos

2. No dependencia de un sitio central

- La autonomía local implica que **todos los sitios deben ser tratados como iguales.** Por lo tanto, no debe haber particularmente ninguna dependencia de un sitio "maestro" central para algún servicio central —por ejemplo, procesamiento centralizado de consultas, administración centralizada de transacciones o servicios centralizados de asignación de nombres— tal que todo el sistema dependa de ese sitio central. Por lo tanto, este segundo objetivo es un corolario del primero (si el primero se logra, el segundo también *forzosamente).*

- Pero "la no dependencia de un sitio central" es necesaria por sí misma, aunque no se logre la autonomía local completa. Por lo tanto, vale la pena mencionarlo como un objetivo independiente.

- La dependencia de un sitio central sería indeseable por las siguientes dos razones (como mínimo).
  - Primero, el sitio central puede ser un cuello de botella.
  - Segundo y más importante, el sistema sería *vulnerable;* es decir, si el sitio central falla, también fallará todo el sistema (el problema del "único punto de falla").



BD Distribuidas
Objetivos

3. Operación continua

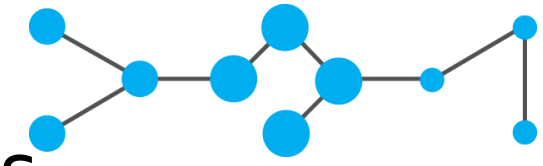
- En general, una ventaja de los sistemas distribuidos es que deben proporcionar mayor *confiabilidad* y mayor *disponibilidad*

■ La **confiabilidad** (es decir, la probabilidad de que el sistema esté listo y funcionando en cualquier momento dado) mejora, ya que los sistemas distribuidos no son una propuesta de todo o nada; pueden continuar operando (en un nivel reducido) cuando hay alguna falla en algún componente independiente, tal como un sitio individual.

■ La **disponibilidad** (es decir, la probabilidad de que el sistema esté listo y funcionando continuamente a lo largo de un período especificado) también mejora, en parte por la misma razón y en parte debido a la posibilidad de replicación de datos.

- Lo anterior se aplica al caso en el que ocurre un **apagado no planeado** (es decir, una falla de algún tipo) en algún punto del sistema. Los apagados no planeados son obviamente indeseables, es difícil evitarlos.

  Por el contrario, los apagados **planeados** *nunca* deberían ser requeridos; es decir, nunca debería ser necesario apagar el sistema para realizar alguna tarea como la incorporación de un nuevo sitio o la actualización del DBMS a una nueva versión en un sitio existente.



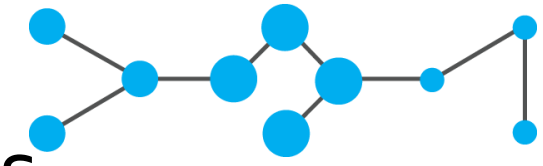
BD Distribuidas
Objetivos

4. Independencia de ubicación

- La idea básica de la **independencia de ubicación** (también conocida como **transparencia** de ubicación) es simple. Los usuarios no tienen que saber dónde están almacenados físicamente los datos, sino que deben ser capaces de comportarse —al menos desde un punto de vista lógico— como si todos los datos estuvieran almacenados en su propio sitio local.

- La independencia de ubicación es necesaria debido a que simplifica los programas de usuario y las actividades terminales. En particular, permite que los datos emigren de un sitio a otro sin invalidar ninguno de estos programas o actividades. Dicha capacidad de emigración es necesaria, ya que permite mover los datos por la red en respuesta a los diferentes requerimientos de desempeño.

La independencia de ubicación es simplemente una extensión al caso distribuido del concepto familiar de la independencia de *datos* (física).

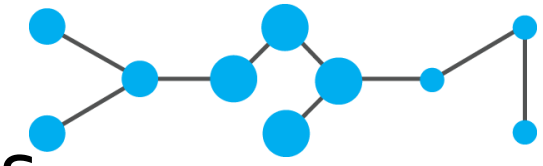


BD Distribuidas
Objetivos

5. Independencia de fragmentación

- Un sistema soporta la **fragmentación de datos** cuando una varrel dada puede ser dividida en partes *o fragmentos*, para efectos de almacenamiento físico.

- La fragmentación es necesaria por razones de rendimiento: los datos pueden estar almacenados en la ubicación donde son usados más frecuentemente para que la mayoría de las operaciones sean locales y se reduzca el tráfico en la red.



BD Distribuidas
Objetivos

6. Independencia de replicación

- Un sistema soporta la **replicación de datos** cuando una varrel almacenada dada —o más generalmente, un *fragmento* dado de una varrel guardada— puede ser representada por muchas copias distintas, o *réplicas*, guardadas en muchos sitios distintos.

- Por ejemplo:

  REPLICATE N_EMP AS

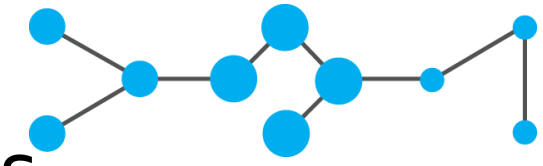
  LN_EMP AT SITE 'Londres' ;

  REPLICATE L_EMP AS

  NL_EMP AT SITE 'Nueva York' ;

- Las réplicas son necesarias por dos razones (como mínimo). Primero, puede significar un mejor rendimiento (las aplicaciones pueden operar sobre copias locales en lugar de tener que comunicarse con sitios remotos). Segundo, también puede significar una mejor disponibilidad (un objeto replicado dado permanece disponible para su procesamiento —al menos para su recuperación— mientras esté disponible al menos una copia).

- Por supuesto, la principal desventaja de la replicación es que al actualizar un objeto replicado dado es necesario actualizar *todas las copias* de ese objeto: el problema de la **propagación de la actualización.**



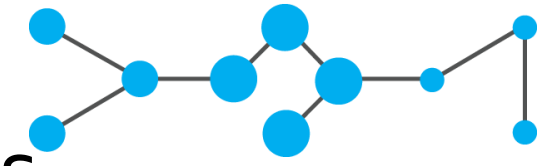
BD Distribuidas
Objetivos

7. **Procesamiento de consultas distribuidas**

- Hay dos temas amplios a tratar bajo este encabezado.

- Primero, considere la consulta "obtener los proveedores de partes rojas de Londres". Suponga que el usuario está en el sitio de Nueva York y los datos están en el sitio de Londres. Suponga también que hay *n* proveedores que satisfacen la solicitud. Si el sistema es relacional, la consulta involucrará básicamente dos mensajes: uno para enviar la solicitud desde Nueva York a Londres y otro para regresar el conjunto resultante de *n* tupias desde Londres a Nueva York. Por otro lado, si el sistema no es relacional, sino que es un sistema de un registro a la vez, la consulta involucrará básicamente 2« mensajes: n desde Nueva York hacia Londres solicitando al "siguiente" proveedor y n desde Londres hacia Nueva York para regresar ese "siguiente" proveedor. El ejemplo ilustra entonces el hecho de que es probable que un sistema relacional supere a uno que no lo es por varios órdenes de magnitud posibles.

- Segundo, la **optimización** es todavía más importante en un sistema distribuido que en uno centralizado. El punto básico es que en una consulta como la anterior —que involucra a varios sitios— habrá muchas formas posibles de mover los datos en el sistema para satisfacer la solicitud, y es crucialmente importante que se encuentre una estrategia eficiente.
- Por ejemplo, una solicitud de (por decir algo) la unión de una relación *Rx* almacenada en el sitio *X* y una relación *Ry* almacenada en el sitio *Y*, podría ser realizada moviendo *Rx* hacia *Y* o *Ry* hacia *X*, o moviendo ambas hacia un tercer sitio Z (etcétera).
- La optimización es claramente crucial, y este hecho puede ser considerado a su vez como otra razón por la que los sistemas distribuidos siempre son relaciónales (donde el punto importante es que las peticiones relaciónales son optimizables, mientras que las no relaciónales no lo son)



BD Distribuidas
Objetivos

8. Administración de transacciones distribuidas

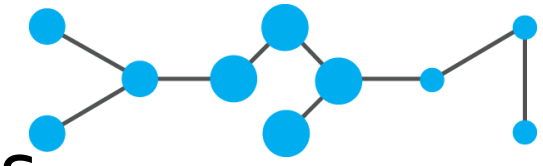
- Como usted sabe, hay dos aspectos principales en la administración de transacciones: el control de la recuperación y el control de la concurrencia. Ambos requieren un tratamiento amplio en el ambiente distribuido.

- Para explicar ese tratamiento amplio, primero es necesario introducir un nuevo término, el *agente.* En los sistemas distribuidos, una sola transacción puede involucrar la ejecución de código en muchos sitios; en particular, puede involucrar actualizaciones en muchos sitios. Por lo tanto, decimos que cada transacción consiste en varios agentes, donde un agente es el proceso realizado en nombre de una transacción dada en un sitio dado. Y el sistema necesita saber cuándo dos agentes son parte de la misma transacción; por ejemplo, no se debe permitir que dos agentes que son parte de la misma transacción caigan en bloqueo mortal entre ellos.

- Continuemos específicamente con el control de la recuperación. Para asegurarse de que una transacción dada sea atómica (todo o nada) en el ambiente distribuido, el sistema debe por lo tanto asegurarse de que el conjunto de agentes para esa transacción sea confirmado o deshecho al unísono.

- Este efecto puede lograrse por medio del protocolo de **confirmación de dos** fases para un sistema distribuido.

- En lo que se refiere al control de concurrencia, en la mayoría de los sistemas distribuidos —al igual que los no distribuidos— el control de concurrencia está basado generalmente en el bloqueo. En la práctica el bloqueo convencional parece seguir siendo la técnica a escoger para la mayoría de los sistemas.



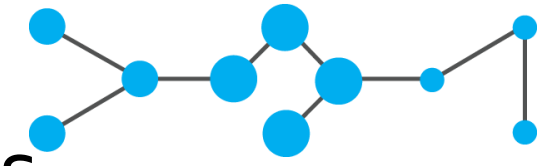
BD Distribuidas
Objetivos

9. Independencia de hardware

- De hecho, no hay mucho que decir sobre este tema, ya que el encabezado lo dice todo. Por lo general, las instalaciones de computadoras involucran en la realidad una gran diversidad de máquinas diferentes — IBM, ICL, HP, PC y estaciones de trabajo de diversos tipos, etcétera— y existe una necesidad real de poder integrar los datos en todos estos sistemas y presentar al usuario una "imagen de sistema único". Por lo tanto, es necesario tener la posibilidad de ejecutar el mismo DBMS en diferentes plataformas de hardware y, además, hacer que esas máquinas diferentes participen como socios igualitarios en un sistema distribuido.

10. Independencia de sistema operativo

- En parte, este objetivo es un corolario del anterior y tampoco requiere de mucha explicación aquí. Obviamente es necesario no sólo tener la posibilidad de ejecutar el mismo DBMS en diferentes plataformas de hardware, sino también ejecutarlo en diferentes plataformas de sistema operativo — incluyendo diferentes sistemas operativos en el mismo hardware— y tener (por ejemplo) una versión MVS, una versión UNIX y una versión NT participando en el mismo sistema distribuido.



BD Distribuidas
Objetivos

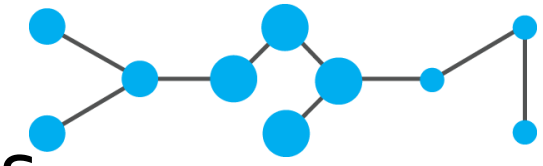
11. Independencia de red

- De nuevo no hay mucho que decir sobre este punto; si el sistema va a tener la posibilidad de soportar muchos sitios distintos —con hardware distinto y sistemas operativos distintos— es obviamente necesario tener la posibilidad de soportar también una variedad de redes de comunicación distintas.

12. Independencia de DBMS

- Bajo este encabezado consideramos lo que está involucrado al flexibilizar la suposición de homogeneidad estricta. Esta suposición es —discutiblemente— un poco más fuerte; todo lo que en realidad necesitamos es que **todos** los ejemplares del DBMS en sitios diferentes **soporten la misma interfaz,** aunque no tienen que ser necesariamente copias del mismo software DBMS.

- Sería posible que el sistema distribuido fuera *heterogéneo,* al menos en cierto grado. El soporte para la heterogeneidad es definitivamente necesario. El hecho es que por lo general las instalaciones de computación en la realidad emplean no sólo muchas máquinas diferentes y muchos sistemas operativos diferentes, sino que muy frecuentemente, también DBMSs diferentes; y sería muy bueno si esos DBMS diferentes pudieran participar de alguna forma en un sistema distribuido. En otras palabras, el sistema distribuido ideal debe proporcionar **independencia de** DBMSs.



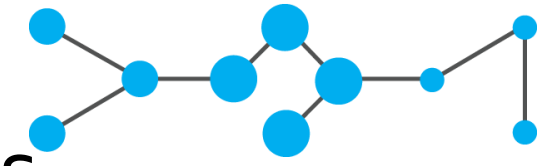
BD Distribuidas Problemas

- El problema principal es que las redes de comunicación —al menos las de "larga distancia" o redes de área amplia (WANs) — son *lentas o limitadas.*

- Por consecuencia, un objetivo principal en los sistemas distribuidos (al menos en el caso de las WAN y hasta cierto punto también en el caso de las LAN) es **minimizar la utilización de la red;** es decir, minimizar la cantidad y el volumen de los mensajes.

- Este objetivo hace a su vez que se presenten problemas en varias áreas complementarias, entre ellas las siguientes (esta lista no pretende ser completa):

  - Procesamiento de consultas.
  - Administración del catálogo.
  - Propagación de la actualización.
  - Control de la recuperación.
  - Control de la concurrencia.

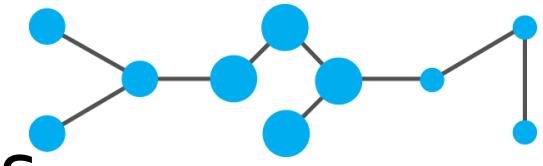


BD Distribuidas
Problemas

Procesamiento de consultas

- El objetivo de minimizar la utilización de la red implica que el propio proceso de optimización de consultas debe ser distribuido, al igual que el proceso de ejecución de la consulta. En otras palabras, el proceso de optimización general consistirá típicamente en un paso de **optimización global** seguido por pasos de **optimización local** en cada sitio afectado.

- Por ejemplo, supongamos que una consulta *Q* es enviada al sitio *X*, y supongamos que *Q* involucra la unión de una relación *Ry* de cien tupias en el sitio *Y* con una relación *Rz* de un millón de tupias en el sitio Z. El optimizador que está en el sitio *X* seleccionará la estrategia global para la ejecución de *Q;* y es claramente necesario que decida mover *Ry* hacia Z y no *Rz* hacia *Y* (por supuesto, tampoco *Ry* y *Rz* hacia *X).* Entonces, una vez que ha decidido mover *Ry* hacia Z, la estrategia para ejecutar la unión real en el sitio Z será decidida por el optimizador local que está en Z.



BD Distribuidas Problemas

Administración del catálogo

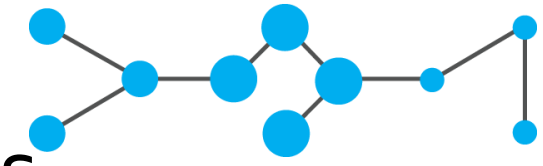
- En un sistema distribuido, el catálogo del sistema incluirá no solamente los datos usuales del catálogo con relación a las varrels base, vistas, autorizaciones, etcétera, sino también toda la información de control necesaria para permitir que el sistema proporcione la independencia de ubicación, fragmentación y replicación necesaria. Surge la siguiente cuestión: ¿dónde y cómo debe ser almacenado el propio catálogo? Estas son algunas posibilidades:

1. ***Centralizado****:* el catálogo total es almacenado exactamente una vez en un sitio central.

2. ***Completamente replicado****:* el catálogo total es almacenado por completo en cada uno de los sitios.

3. ***Dividido****:* cada sitio mantiene su propio catálogo de los objetos que están almacenados en ese sitio. El catálogo total es la unión de todos los catálogos locales disjuntos.

4. ***Combinación de 1 y 3****:* cada sitio mantiene su propio catálogo local, como en el punto 3; además, un único sitio central mantiene una copia unificada de todos esos catálogos locales, como en el punto 1.



BD Distribuidas
Problemas

Propagación de la actualización

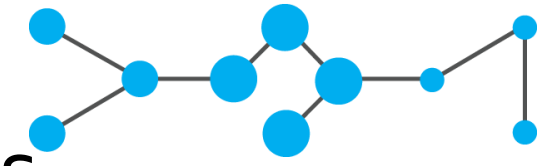
- El problema básico que existe con la replicación de datos es que una actualización a cualquier objeto lógico dado debe ser propagada a todas las copias almacenadas de ese objeto. Una dificultad inmediata es que un sitio que mantiene una copia del objeto podría no estar disponible (debido a una falla del sitio o de la red) en el momento de la actualización. Por lo tanto, la estrategia obvia para propagar inmediatamente las actualizaciones hacia todas las copias, es probablemente inaceptable, ya que implica que la actualización—y por ende la transacción— fallará si alguna de esas copias no está disponible actualmente. De hecho, en cierto sentido los datos están *menos* disponibles bajo esta estrategia que como lo estarían en el caso no replicado, y por lo tanto, debilita a una de las ventajas proclamadas para la replicación en la sección anterior.

- Un esquema común para atacar este problema (aunque no es el único posible) es el llamado esquema de **copia primaria,** que funciona de la siguiente forma:

  ■ A una copia de cada objeto replicado se le designa como copia *primaria.* Todas las demás son copias secundarias.

  ■ Las copias primarias de diferentes objetos están en diferentes sitios (por lo tanto, éste es nuevamente un esquema distribuido).

  ■ Decimos que las operaciones de actualización quedaron lógicamente terminadas tan pronto como se actualizó la copia primaria. Entonces, el sitio que mantiene esa copia es responsable de la propagación de la actualización hacia las copias secundarias en algún tiempo subsecuente. Ese "tiempo subsecuente" debe ser previo al COMMIT, si es que se van a conservar las propiedades ACID de la transacción.



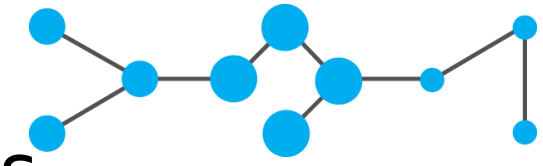
BD Distribuidas Problemas

Control de la recuperación

El control de la recuperación en sistemas distribuidos está basado típicamente en el protocolo de **confirmación de dos fases** (o en alguna variante). La confirmación de dos fases es necesaria en *cualquier* ambiente en el cual una transacción única puede interactuar con varios administradores de recursos autónomos, aunque es particularmente importante en un sistema distribuido debido a que los administradores de los recursos en cuestión —es decir, los DBMSs locales— están operando en sitios distintos y por lo tanto, son *muy* autónomos.

Surgen los siguientes puntos:

- 1. El objetivo de la "no dependencia de un sitio central" indica que la función coordinadora no debe ser asignada a un sitio distinguido en la red, sino que en su lugar, debe ser realizada por sitios diferentes para transacciones diferentes. Por lo general es manejada por el sitio j en el cual se inicia la transacción en cuestión; por lo tanto (en general), cada sitio debe ser I capaz de actuar como sitio coordinador para algunas transacciones y como sitio participante para otras.
- 2. El proceso de confirmación de dos fases requiere que el coordinador se comunique con cada uno de los sitios participantes, lo cual significa más mensajes y más sobrecarga.
- 3. Si el sitio *Y* actúa como participante en un proceso de confirmación de dos fases coordinado por el sitio *X,* entonces el sitio *Y debe* hacer lo que le indica el sitio *X* (confirmar o deshacer,] según se aplique) y esto es una pérdida (tal vez menor) de autonomía local.



BD Distribuidas
Problemas

Control de la concurrencia

- El control de la concurrencia está basado en el bloqueo. Sin embargo, en un sistema distribuido las solicitudes para probar, poner y liberar bloqueos se convierten en *mensajes* (suponiendo que el objeto en consideración está en un sitio remoto) y los mensajes significan una sobrecarga.

- Por ejemplo, considere una transacción *T* que necesita actualizar un objeto para el cual existen réplicas en *n* sitios remotos. Si cada sitio es responsable de los bloqueos sobre los objetos que están almacenados en ese sitio (como será bajo la suposición de autonomía local), entonces una implementación directa requerirá al menos n mensajes:

  - *n* solicitudes de bloqueo,
  - *n* otorgamientos de bloqueo,
  - *n* mensajes de actualización,
  - *n* notificaciones,
  - *n* solicitudes de desbloqueo

Gracias