

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY  
CAMPUS ESTADO DE MÉXICO



**ADMINISTRADOR DE BASES DE DATOS DISTRIBUIDAS.**

TESIS PARA OPTAR EL GRADO DE MAESTRO EN CIENCIAS  
*96901*

BIBLIOTECA

COMPUTACIONALES

PRESENTA

**GUILLERMO KUNZ ARRACHE**



Asesor : DR. JUAN FRANCISCO CORONA BURGUEÑO.

Comité de tesis : DR. JUAN FRANCISCO CORONA BURGUEÑO.

DR. JESÚS SÁNCHEZ VELÁZQUEZ

MCC. RALF EDER LANGE

Jurado : DR. JUAN FRANCISCO CORONA BURGUEÑO.

DR. JESÚS SÁNCHEZ VELÁZQUEZ

MCC. RALF EDER LANGE

Atizapán de Zaragoza, Estado de México, Diciembre, 1998

## INDICE DE CONTENIDO

<b>CAPÍTULO 1. ESTADO DEL ARTE .....</b>	<b>15</b>
1.1. ANTECEDENTES .....	15
1.1.1. Bases De Datos Relacionales .....	16
1.1.1.1 Redundancia o repetición de información .....	18
1.1.1.2. Pérdida de información .....	18
1.1.1.3. Recuperación .....	18
1.1.1.3.1. Fallas del sistema .....	19
1.1.1.3.2. Falla de medios de almacenamiento .....	20
1.1.1.4. Concurrencia .....	21
1.1.1.4.1. Estampillas de tiempo .....	21
1.1.1.4.2. Candados .....	21
1.1.1.4.3. Problemas debidos a la concurrencia .....	23
1.1.2. Sistema De Base De Datos Distribuido .....	24
1.1.2.1. Autonomía local .....	26
1.1.2.2. No dependencia de un sitio central .....	26
1.1.2.3. Operación continua .....	26
1.1.2.4. Independencia con respecto a la localización .....	26
1.1.2.5. Independencia con respecto a la fragmentación .....	27
1.1.2.6. Independencia de réplica .....	27
1.1.2.7. Procesamiento distribuido de consultas .....	27
1.1.2.8. Manejo distribuido de transacciones .....	27
1.1.2.9. Independencia con respecto al equipo .....	27
1.1.2.10. Independencia con respecto al sistema operativo .....	28
1.1.2.11. Independencia con respecto a la red .....	28

1.1.2.12. Independencia con respecto al DBMS .....	28
1.2. FUNCIONAMIENTO DE BASE DE DATOS DISTRIBUIDAS .....	28
1.3. BENEFICIOS EN EL USO DE BASE DE DATOS DISTRIBUIDAS .....	30
<b>CAPÍTULO 2. PLANTEAMIENTO DEL PROBLEMA. ....</b>	<b>32</b>
2.1. JUSTIFICACIÓN .....	32
2.2. OBJETIVOS .....	33
2.2.1. Continuidad .....	33
2.2.2. Tolerancia a fallas .....	33
2.2.3. Transparencia .....	33
2.2.4. Consistencia .....	33
2.2.5. Recuperación .....	34
<b>CAPÍTULO 3. DISEÑO CONCEPTUAL .....</b>	<b>35</b>
3.1. ELEMENTOS BÁSICOS DE UNA BASE DE DATOS DISTRIBUIDA .....	35
3.1.1. Autonomía De Diseño .....	36
3.1.2. Autonomía De Ejecución .....	36
3.1.3. Autonomía De Comunicación .....	36
3.1.4. Estrategias Comunes .....	37
3.1.4.1. Estrategia 1 .....	37
3.1.4.2. Estrategia 2 .....	38
3.1.4.3. Estrategia 3 .....	40
3.1.4.4. Estrategia 4 .....	41
3.1.5. Estrategias Distribuidas .....	43
3.1.5.1. Estrategia d1 .....	43
3.1.5.2. Estrategia d2 .....	43

3.1.5.3. Estrategia d3.	46
3.1.5.4. Estrategia d4	47
3.1.5.5. Estrategia d5.	49
<b>CAPITULO 4. DISEÑO GLOBAL .....</b>	<b>52</b>
<b>4.1. ESTRUCTURA DEL DISEÑO GLOBAL .....</b>	<b>52</b>
4.1.1. Proceso Del Diseño Global .....	53
<b>4.2. ESQUEMA GLOBAL .....</b>	<b>54</b>
4.2.1. Módulo Controlador De Base De Datos Y De Comunicación .....	54
4.2.1.1. Consulta de las tablas <b>GAR</b> y <b>GEL</b> para localización de archivos ..	56
4.2.1.2. Actualización de la tabla <b>GEG</b> por parte del administrador .....	58
4.2.1.3. Actualización de la tabla <b>GEG</b> por parte de otro esquema global .....	59
4.2.1.4. Actualización de la tabla <b>GEL</b> por parte del administrador .....	60
4.2.1.5. Actualización de la tabla <b>GEL</b> por parte de otro esquema global .....	61
4.2.1.6. Actualización de la tabla <b>GAR</b> por parte del administrador .....	62
4.2.1.7. Actualización de <b>GAR</b> por parte de otro esquema global .....	63
4.2.1.8. Quita bloqueo por parte del administrador .....	64
4.2.1.9. Quita el bloqueo por parte de otro esquema global .....	65
4.2.2. Módulo De Diccionario De Datos .....	66
4.2.3. Módulo De Recuperación .....	68
4.2.4. Módulo De Edición De Esquemas Globales Alternos .....	69
4.2.5. Módulo De Actualización .....	71
<b>4.3. ESQUEMA LOCAL .....</b>	<b>72</b>
4.3.1. Módulo Controlador De Base De Datos Y De Comunicación .....	72
4.3.1.1. La respuesta a una consulta a un esquema global y/o local .....	74

4.3.1.2. Baja de un LEG por parte del administrador .....	74
4.3.1.3. Consulta de las tablas GAR, GEL, LAR y LEL para localización de archivos .....	75
4.3.1.4. Actualización de la tabla LEG por parte de un esquema global .....	76
4.3.1.5. Actualización de la tabla LEL por parte de un esquema global .....	77
4.3.1.6. Actualización de la tabla LAR por parte de un esquema global .....	78
4.3.1.7. Quita bloqueos por parte de un esquema global .....	79
4.3.1.8. Actualización de la tabla LEG por parte del administrador .....	80
4.3.2. Módulo De Diccionario De Datos .....	81
4.3.3. Módulo De Edición Del Esquema Local .....	82
 <b>CAPÍTULO 5. REPLICACIÓN Y FRAGMENTACIÓN .....</b>	 83
5.1. CONCEPTO DE RAID .....	84
5.1.1. Características De RAID .....	84
5.1.2. Descripción De RAID .....	85
5.1.2.1. RAID 0. Estructura en tiras .....	85
5.1.2.2. RAID 1. Estructura en espejo .....	86
5.1.2.3. RAID 2. Acceso paralelo (redundante) .....	87
5.1.2.4. RAID 3. Acceso paralelo (intercalado) .....	88
5.1.2.5. RAID 4. Acceso independiente (intercalado) .....	89
5.1.2.6. RAID 5. Acceso independiente (distribuido) .....	90
 <b>CAPÍTULO 6. ESTUDIO COMPARATIVO .....</b>	 95
6.1. ESTUDIO COMPARATIVO .....	95
6.1.1. Administradores Que Crean Comunicación Por Medio De Compuertas ...	95
6.1.2. Administradores Administrados Por Uno Local .....	96

9.1.3. Tesis De Flamenco Sandoval .....	96
<b>CAPÍTULO 7. CONCLUSIONES .....</b>	<b>100</b>
<b>BIBLIOGRAFÍA.</b> .....	<b>102</b>
<b>ANEXO A</b> .....	<b>104</b>
<b>ANEXO B</b> .....	<b>106</b>

## INDICE DE TABLAS

### **CAPITULO 1. ESTADO DEL ARTE**

TABLA 1: Inquilinos .....	17
TABLA 2: Conflictos entre cандados .....	22

### **CAPITULO 3. DISEÑO CONCEPTUAL**

TABLA 3: Resultados de la estrategia 2 .....	39
TABLA 4: Resultados de la estrategia 3 .....	41
TABLA 5: Resultados de la estrategia 4 .....	42
TABLA 6: Resultados de la estrategia d2 .....	45
TABLA 7: Resultados de la estrategia d3 .....	47
TABLA 8: Resultados de la estrategia d4 .....	48
TABLAS 9,10: Resultados de la estrategia d4 .....	49
TABLA 11: Resultados de la estrategia 5 .....	50

### **CAPÍTULO 4. DISEÑO GLOBAL**

TABLA 12: Formato de mensaje .....	56
TABLA 13: Esquema global (GEG) .....	66
TABLA 14: Esquema local (GEL) .....	67
TABLA 15: Archivos (GAR) .....	67
TABLA 16: Esquema global (LEG) .....	81
TABLA 17: Esquema local (LEL) .....	81
TABLA 18: Archivos (LAR) .....	82

**CAPITULO 6. ESTUDIO COMPARATIVO**

TABLA 19:	VISTAS_GLOBALES .....	97
TABLA 20:	TABLAS_GLOBALES .....	97
TABLA 21:	COLUMNAS_GLOBALES .....	97
TABLA 22:	TABLAS_FRAGMENTADAS .....	97
TABLA 23:	COLUMNAS_FRAGMENTADAS .....	97
TABLA 24:	TABLA COMPARATIVA .....	101

## INDICE DE FIGURAS

### **CAPITULO 1. ESTADO DEL ARTE**

FIGURA 1:	Caída del sistema .....	19
FIGURA 2:	Modificación perdida .....	23
FIGURA 3:	Inconsistencia .....	24
FIGURA 4:	Manejadores con compuertas .....	29
FIGURA 5:	Manejadores administrados por uno local .....	30

### **CAPITULO 3. DISEÑO CONCEPTUAL**

FIGURA 6:	Estrategias .....	37
FIGURA 6.1:	Estrategia 1 .....	38
FIGURA 6.2:	Estrategia 2 .....	39
FIGURA 6.3:	Estrategia 3 .....	40
FIGURA 6.4:	Estrategia 4 .....	42
FIGURA 7:	Estrategias distribuidas .....	43
FIGURA 7.1:	Estrategia d1 .....	44
FIGURA 7.2:	Estrategia d2 .....	45
FIGURA 7.3:	Estrategia d3 .....	46
FIGURA 7.4:	Estrategia d4 .....	48
FIGURA 7.5:	Estrategia d5 .....	50

### **CAPITULO 4 DISEÑO GLOBAL**

FIGURA 8:	Diseño global .....	53
-----------	---------------------	----

**CAPITULO 5 REPLICACIÓN Y FRAGMENTACIÓN**

FIGURA 9:	RAID 0 .....	85
FIGURA 10:	RAID 1 .....	86
FIGURA 11:	RAID 2 .....	87
FIGURA 12:	RAID 3 .....	88
FIGURA 13:	RAID 4 .....	89
FIGURA 14	RAID 5 .....	90
FIGURA 15	Archivos RAID 0 .....	91
FIGURA 16	Archivos RAID 1 .....	91
FIGURA 17	Archivos RAID 2 .....	92
FIGURA 18	Archivos RAID 3 .....	92
FIGURA 19	Archivos RAID 4 .....	93
FIGURA 20	Archivos RAID 5 .....	93

## INDICE DE DIAGRAMAS

### CAPITULO 4. ESQUEMA GLOBAL

DIAGRAMA 1:	Módulo controlador de base de datos y comunicación	55
DIAGRAMA 2:	811 .....	57
DIAGRAMA 3:	812 .....	58
DIAGRAMA 4:	813 .....	59
DIAGRAMA 5:	814 .....	60
DIAGRAMA 6:	815 .....	61
DIAGRAMA 7:	816 .....	62
DIAGRAMA 8:	817 .....	63
DIAGRAMA 9:	818 .....	64
DIAGRAMA 10:	819 .....	65
DIAGRAMA 11:	Módulo de recuperación .....	68
DIAGRAMA 12:	Edición de esquemas globales alternos .....	69
DIAGRAMA 13:	Verificación de mensajes .....	70
DIAGRAMA 14:	Revisa bloqueos RB .....	71
DIAGRAMA 15:	Módulo controlador de base de datos y comunicación	73
DIAGRAMA 16:	912 .....	74
DIAGRAMA 17:	913 .....	75
DIAGRAMA 18:	914 .....	76
DIAGRAMA 19:	915 .....	77
DIAGRAMA 20:	916 .....	78
DIAGRAMA 21:	917 .....	79
DIAGRAMA 22:	918 .....	80
DIAGRAMA 23:	Edición del esquema local .....	82

**INDICE DE ALGORITMOS****ANEXO 2**

ALGORITMO 1:	Módulo controlador de base de datos y comunicación	106
ALGORITMO 2:	Módulo de recuperación .....	113
ALGORITMO 3:	Edición de esquemas globales alternos .....	114
ALGORITMO 4:	Verificación de mensajes .....	115
ALGORITMO 5:	Revisa bloqueos RB .....	116
ALGORITMO 6:	Módulo controlador de base de datos y comunicación	117
ALGORITMO 7:	Edición del esquema local .....	120

**CAPITULO 5 REPLICACIÓN Y FRAGMENTACIÓN**

FIGURA 9:	RAID 0 .....	88
FIGURA 10:	RAID 1 .....	89
FIGURA 11:	RAID 2 .....	90
FIGURA 12:	RAID 3 .....	91
FIGURA 13:	RAID 4 .....	92
FIGURA 14	RAID 5 .....	93
FIGURA 15	Archivos RAID 0 .....	94
FIGURA 16	Archivos RAID 1 .....	94
FIGURA 17	Archivos RAID 2 .....	95
FIGURA 18	Archivos RAID 3 .....	95
FIGURA 19	Archivos RAID 4 .....	96
FIGURA 20	Archivos RAID 5 .....	96

## INDICE DE DIAGRAMAS

### **CAPITULO 4. ESQUEMA GLOBAL**

DIAGRAMA 1:	Módulo controlador de base de datos y comunicación .	58
DIAGRAMA 2:	811 .....	59
DIAGRAMA 3:	812 .....	60
DIAGRAMA 4:	813 .....	61
DIAGRAMA 5:	814 .....	62
DIAGRAMA 6:	815 .....	63
DIAGRAMA 7:	816 .....	64
DIAGRAMA 8:	817 .....	65
DIAGRAMA 9:	818 .....	66
DIAGRAMA 10:	819 .....	67
DIAGRAMA 11:	Módulo de recuperación .....	70
DIAGRAMA 12:	Edición de esquemas globales alternos .....	71
DIAGRAMA 13:	Verificación de mensajes .....	72
DIAGRAMA 14:	Revisa bloqueos RB .....	73
DIAGRAMA 15:	Módulo controlador de base de datos y comunicación .	76
DIAGRAMA 16:	912 .....	77
DIAGRAMA 17:	913 .....	78
DIAGRAMA 18:	914 .....	79
DIAGRAMA 19:	915 .....	80
DIAGRAMA 20:	916 .....	81
DIAGRAMA 21:	917 .....	82
DIAGRAMA 22:	918 .....	83
DIAGRAMA 23:	Edición del esquema local .....	85

**INDICE DE ALGORITMOS****ANEXO 2**

ALGORITMO 1:	Módulo controlador de base de datos y comunicación .	109
ALGORITMO 2:	Módulo de recuperación .....	116
ALGORITMO 3:	Edición de esquemas globales alternos .....	117
ALGORITMO 4:	Verificación de mensajes .....	118
ALGORITMO 5:	Revisa bloqueos RB .....	119
ALGORITMO 6:	Módulo controlador de base de datos y comunicación .	120
ALGORITMO 7:	Edición del esquema local .....	123

## **ADMINISTRADOR DE BASES DE DATOS DISTRIBUIDAS**

# CAPÍTULO 1.

## ESTADO DEL ARTE

### 1.1 ANTECEDENTES.

Podemos decir, a manera de introducción, que los orígenes de las bases de datos están ciertamente en los modelos centralizados, pero el constante crecimiento de la industria, el cambio de la tecnología en las empresas y el desarrollo de las comunicaciones, nos obligan a descentralizar tanto procesos administrativos como de información.

La tecnología actual en el ámbito de la microelectrónica y el desarrollo de la industria del cómputo permiten disponer hoy en día de poder de cómputo a precios reales antes no imaginados, lo que ha permitido a la empresa mediana y pequeña acceder a satisfactores para el tratamiento y difusión de su información en formas previamente posibles solamente a las grandes organizaciones gubernamentales o privadas, incrementando en órdenes de magnitud la demanda de equipos, software y servicios complementarios para tales efectos.

Algunas cifras pueden ayudarnos a identificar la importancia económica de lo anterior, al considerar que en el mercado mexicano en este año de 1998 el valor de mercado del software empaquetado, que incluye el suministro de los manejadores de bases de datos, llegará prácticamente a los 548 millones de dólares y el de los servicios de implementación de sistemas, que incorporan las tareas de asesoría y desarrollo adicional para hacer productivas las inversiones en software, a los 536 millones de dólares.

Por otra parte, la dependencia creciente en la información y la cultura de toma de decisiones más informadas que ha caracterizado a las escuelas de negocios de los últimos años ha originado un

interés especial en hacer disponible más y más información al interior de la empresa y compartirla con otras organizaciones, mediando o no costos por tal disponibilidad.

En forma simultánea, las telecomunicaciones acercaron a los generadores, procesadores y consumidores de información haciendo posible el intercambio de datos a distancia, así como la posibilidad de almacenar la información regionalmente bajo esquemas de descentralización. El explosivo crecimiento de la red Internet, con la multitud de prestadores de servicios de acceso que han surgido en los años muy recientes, ha mostrado ya lo que seguramente será la constante en el futuro: es posible, y redituable, hacer disponible información a escala mundial con inversiones relativamente pequeñas en equipo y sistemas.

Dado lo anterior, la demanda de sistemas apropiados que organicen y pongan a disposición información de valor es creciente y el imperativo de eficiencia en las tareas de organización, recuperación, procesamiento y difusión de la información es definitivo.

Estas son algunas de las razones que nos motivan a la realización del presente trabajo, el que hace una propuesta para descentralizar uno de los procesos de información, como es el de guardar los datos de una o varias empresas para poder hacer uso de éstos en cualquier punto en donde se desee consultar, con la facilidad de tener disponible y actualizada toda la información para su consulta.

El modelo en el que vamos a trabajar es el modelo relacional. Éste, además de ser el principal modelo de datos para aplicaciones comerciales, ayuda al diseño de bases de datos relacionales y al procesamiento eficiente de solicitudes de información a la base de datos por parte de los usuarios.

### 1.1.1. BASES DE DATOS RELACIONALES.

"Una base de datos relacional es aquélla cuyos usuarios la perciben como un conjunto de tablas" (1). A cada tabla se le asigna un nombre único y una fila de cada tabla representa una relación entre un conjunto de valores. Una estructura básica de una base de datos relacional se crea con una serie de **atributos**, donde cada atributo tiene un conjunto de valores permitidos o **dominio**; por ejemplo, podríamos hacer una tabla llamada **INQUILINOS** con cuatro atributos llamados: **contrato, ciudad, nombre y teléfono**, en la que el dominio de cada uno lo determina su propio nombre.

TABLA 1: INQUILINOS			
contrato	ciudad	nombre	teléfono
2	D.F.	Ana	2345675
4	Veracruz	Gabriela	4325675
3	Monterrey	Adolfo	3445642
9	Monterrey	Pablo	3245227

Cada relación que existe entre el **contrato**, **ciudad**, **nombre** y **teléfono** en la tabla **INQUILINOS** se llama **tupla** y se identifica con la letra "t" (contrato, ciudad, nombre, teléfono).

En la tabla del ejemplo contamos con 4 tuplas, que son las siguientes:

$t(2, \text{D.F.}, \text{Ana}, 2345675)$

$t(4, \text{Veracruz}, \text{Gabriela}, 4325675)$

$t(3, \text{Monterrey}, \text{Adolfo}, 3445642)$

$t(9, \text{Monterrey}, \text{Pablo}, 3245227)$

Dentro de las bases de datos, necesitamos diferenciar entre varias tuplas para saber cuál estamos necesitando, para esto tenemos lo que se conoce con el nombre de **llaves**: campos especiales que identifican de manera única las tuplas. En el ejemplo de la tabla propuesta, una superllave puede ser el número de contrato y el nombre del inquilino, ya que un mismo inquilino puede tener varios contratos y si quitamos el elemento nombre del inquilino, entonces el número de contrato se convierte en una llave primaria.

A continuación analizaremos los problemas a los que nos podemos enfrentar en el diseño de una base de datos, mismos que crecerán al diseñar una base de datos distribuida, como son: repetición de información, pérdida de información, recuperación y concurrencia; estos problemas generalmente se derivan del procesamiento de transacciones. Después, para poder explicar cómo podemos solucionar estos problemas, explicaremos los comandos que utilizaremos para crear protocolos de seguridad y de integridad de datos. Éstos son COMMIT y ROLLBACK, que nos sirven para cuando un programa o una consulta modifica la base de datos. Normalmente esta modificación es solo tentativa, de tal manera que si algo sale mal, entonces se puede recuperar la información y hacer permanente la modificación, mientras que ROLLBACK anula la modificación.

#### **1.1.1.1. Redundancia o repetición de información.**

Este problema de diseño ocurre en tablas en las cuales existen dependencias funcionales parciales, transitivas o multivaluadas. En estos casos el costo de una modificación a la tabla sería caro porque necesitaríamos modificar muchas tuplas, además del riesgo de inconsistencias; por otro lado el espacio que se ocupa es mayor. Para evitar estos problemas de redundancia o repetición de información, se debe buscar un grado de normalización adecuado, como la tercera forma normal, que impida la posibilidad de duplicar los datos.

#### **1.1.1.2. Pérdida de información.**

Al descomponer una relación en varias, el resultado de un "join" de las subrelaciones puede producir tuplas adicionales que falsearían la respuesta, cuando exista información común a dos o más de las tuplas procesadas.

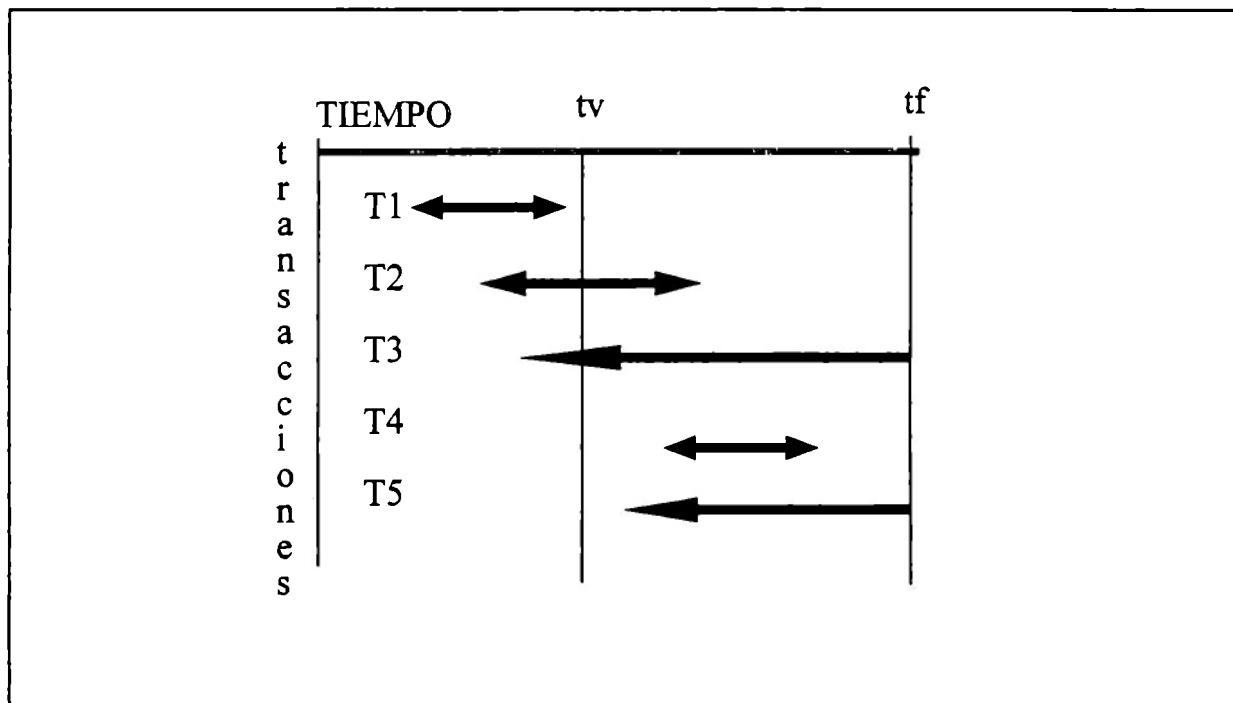
#### **1.1.1.3. Recuperación.**

El sistema debe estar preparado para la recuperación no sólo de fallas locales, sino también de fallas globales. Una forma de prevenir la pérdida de información por alguno de estos dos tipos de fallas es tener puntos de sincronización, es decir puntos donde decisiones globales son tomadas. El sitio en donde están los comandos COMMIT y ROLLBACK dentro de un programa, así como el inicio de la ejecución de un programa, son puntos de sincronización. Estos comandos bloquean el registro en la base de datos poniendo un cursor. En caso de una caída del sistema, el procedimiento de reinicio deberá ver todos los cursores del sistema y por medio de un registro ver cuáles deben grabarse y cuáles anularse. A esto se conoce con el nombre de *protocolo de escritura adelantada*. Los problemas de recuperación se dividen en dos categorías: fallas del sistema y fallas de medios de almacenamiento.

### 1.1.1.3.1. Fallas del sistema.

El problema de las fallas del sistema nos puede ocasionar la pérdida de los **buffers** y la **memoria**, sin los cuales no sabemos en qué punto se quedaron las transacciones. Esto nos lleva a tener un sistema de respaldo de buffers para que al reactivar el sistema podamos saber qué transacciones terminaron con éxito y en qué punto de su ejecución se quedaron las demás transacciones (T). En el tiempo, esto se vería de la siguiente forma (figura 1):

- Se toma un tiempo de verificación (tv) en el cual vamos a crear un archivo que nos va a decir qué transacciones terminaron de hacer las modificaciones o cancelaciones de la base de datos y qué transacciones iniciaron y quedaron en proceso después del último tiempo de verificación (tv).
- Se presenta una falla del sistema en el tiempo de falla (tf).



**Fig 1:CAÍDA DEL SISTEMA**

Según los tiempos de inicio y terminación podemos clasificar a las transacciones en cinco tipos:

- Las transacciones del tipo T1, que se completaron antes de tv.
- Las transacciones del tipo T2, que se iniciaron antes de tv y se completaron después de tv y antes de tf.
- Las transacciones del tipo T3, que se iniciaron antes de tv y no se completaron antes de tf.
- Las transacciones del tipo T4, que se iniciaron después de tv y se completaron antes de tf.
- Las transacciones del tipo T5, que se iniciaron después de tv y no se completaron antes de tf.

Para nosotros es evidente que las transacciones del tipo T3 y T5 deben de ser las transacciones que se cancelen, por no haber hecho una confirmación de los cambios o cancelación de éstos. La transacción T1 no vamos a tocarla por haber terminado antes del último tiempo de verificación, mientras que las transacciones T2 y T4 necesitamos repetirlas, ya que no estamos seguros de que a todas las réplicas les llegó un COMMIT o un ROLLBACK, como se señaló en el punto 1.1.

Una forma de recuperar estas fallas es utilizando dos listas, ANULAR y REPETIR; en ANULAR se ponen todas las transacciones activas al tiempo tv (T2, T3), mientras que REPETIR inicialmente está vacía. En seguida se revisa la bitácora hacia adelante a partir del punto tv; si se encuentra en la bitácora el inicio de una transacción, entonces se agrega a la tabla de ANULAR (T4, T5); si se encuentra en la bitácora el comando COMMIT para una transacción, se mueve de ANULAR a REPETIR (T2, T4). Cuando se llegue al final, las listas ANULAR y REPETIR identificarán las transacciones que deben anularse y las que deben repetirse respectivamente. En nuestro ejemplo las transacciones T3 y T5 de la lista ANULAR deben anularse por no saberse si la transacción después del tiempo tf tuvo éxito o no, ya que no se encontró ningún comando COMMIT. Las transacciones T2 y T4, de la lista REPETIR, deben repetirse para que las modificaciones que se hicieron antes del tiempo tf se tomen en cuenta y las réplicas del sistema queden iguales.

En cuanto a las réplicas, no se puede enviar ninguna instrucción a los sitios en tanto no confirmen cada una su situación. En caso afirmativo (COMMIT) se ordena continuar y repetir en general; en el caso de que una o más de ellas no envie la respuesta correcta (ROLLBACK), deberán cancelarse todas y reponer el procedimiento.

#### **1.1.1.3.2. Falla de medios de almacenamiento.**

Este tipo de fallas se genera cuando hay un aterrizaje de las cabezas del disco o falla de una tarjeta controladora. En estos casos, la recuperación de medios físicos no es posible, por lo que recomienda un sistema de procesamiento en espejo, el cual va a estar trabajando con dos discos que tienen la misma información. En caso de fallar uno, el otro entra en funcionamiento hasta reemplazar el disco defectuoso; otra opción para prevenir la perdida de información por este tipo de fallas es haciendo respaldos de información en cintas o en algún otro medio, con la finalidad de repetir únicamente las transacciones desde el punto en el cual se hizo el respaldo hasta el momento en que falló el disco. En el caso de ser sólo la controladora, se debe cambiar por una en buen estado y seguir con el procedimiento descrito anteriormente de recuperación de caídas de sistema.



#### 1.1.1.4. Concurrencia.

En un sistema de base de datos lo mas común es que haya varios usuarios trabajando al mismo tiempo. Todos estos usuarios van a ejecutar transacciones las cuales estarán modificando, borrando y agregando información a la misma base de datos. Para que todos estos movimientos no se afecten entre ellos es necesario contar con un sistema de *control de concurrencia*. Nosotros vamos a trabajar con un mecanismo de bloqueo, que aunque no es el único, pensamos que es más adecuado que el de estampillas de tiempo, por tener mayor tolerancia en los tiempos.

##### 1.1.1.4.1 Estampillas de tiempo.

En este método, cada operación de transacción es validada antes de llevarse a cabo. Si no puede realizarse, la transacción es abortada inmediatamente.

Cada transacción recibe una estampilla de tiempo al iniciar. Las operaciones realizadas dentro de una transacción llevan la estampilla de ésta.

Reglas claras deben ser definidas para saber si una operación es válida o no.

Ejemplo de reglas cuando no hay versiones repetidas de un dato:

- a) La petición de escribir un dato es válida si el dato fue leído y escrito por transacciones de tiempos anteriores.
- b) La petición de leer un dato es válida sólo si el dato fue escrito por una transacción anterior.

##### 1.1.1.4.2. Candados

Pueden ser utilizados para resolver problemas como el de modificaciones perdidas (figura 2).

Existen tres tipos: lectura, escritura y exclusivos.

Para asegurar una equivalencia serie, es necesario que los candados no sean liberados hasta el final de la transacción.

Ocurre entonces un uso de candados a dos fases:

- a) En la fase creciente, se adquieren los candados.
- b) En la fase reductora, se liberan.

En la fase creciente, los candados pueden ser promovidos a candados de escritura.

<b>Tabla 2: Conflictos entre candados</b>			
Operaciones con diferentes transacciones		Conflicto	Razón
Lectura	Lectura	No	Por que el efecto de que se ejecuten dos lecturas no depende del orden en el que fueron echas
Lectura	Escritura	Si	Por que el efecto de una lectura y una escritura si depende del orden en el que se ejecutaron
Escritura	Escritura	Si	Por que el efecto de dos escrituras si depende del orden en el que se ejecutaron

Hay que limitar el uso de candados exclusivos, pues en ocasiones podrían realizarse lecturas en paralelo.

Las siguientes reglas de operación nos permiten escoger los candados a poner:

- a) Si una transacción T ya ha leído un dato, una transacción U no debe escribir el dato hasta que T termine.
- b) Si una transacción T ya ha escrito un dato, una transacción U no debe leer ni escribir ese dato hasta que T termine.

El problema con los candados es que pueden generar deadlocks, es decir estados en los cuales algunos miembros del grupo de transacciones esperan a que otros miembros quiten un candado y viceversa.

Hay tres maneras generales de tratar los deadlocks:

- a) Prevención. Antes de la ejecución, se revisan los posibles grafos de espera en el programa.
- b) Detección. El manejador de candados puede supervisar la ejecución, si en algún momento detecta un ciclo en el grafo de espera, se rompe el deadlock.
- c) Evitar. Antes de cualquier asignación de candados, el programador o el manejador revisan el grafo de espera para ver si podría haber un deadlock.

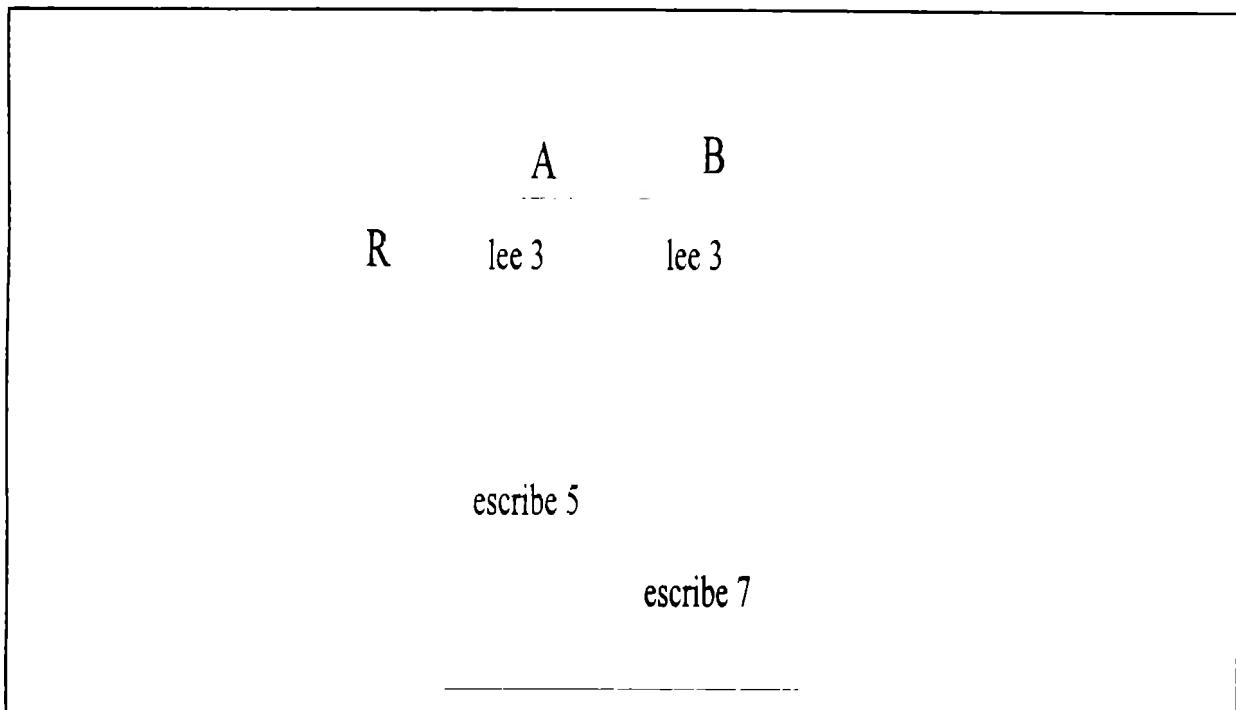
El uso de temporizadores permite salir de condiciones de deadlock: un candado con el tiempo se vuelve vulnerable.

Si nadie pide un candado, funciona como un candado normal.

En caso contrario, el candado vulnerable puede romperse.

#### 1.1.1.4.3. Problemas debidos a la concurrencia.

Los problemas a los que nos enfrentamos debido a la concurrencia son: modificación perdida (figura 2), transacción no comprometida (figura 2) y errores y análisis inconsistentes (figura 3).



**Fig 2: MODIFICACIÓN PERDIDA Y TRANSACCION COMPROMETIDA**

En el caso de la *modificación perdida*, ejemplificado en la figura 2, lo que vamos a tener es el supuesto más común, en que dos transacciones, A y B, van a leer el registro R sin usar un COMMIT o un ROLLBACK; por ejemplo, si el valor de R es 3, entonces tanto A como B van a tener el valor de 3 y después, lo pueden modificar por otro valor, por ejemplo, A lo modifica por 5 y B por 7 respectivamente. De este modo se pierde la modificación que hizo A. Si éste fuera el caso de dos depósitos bancarios donde A va a agregar 2 y B va a agregar 4, el primer depósito se perdería, al no acumulase con el segundo.,

El segundo problema se refiere a la *transacción no comprometida* (figura 2); para evitar que suceda lo mismo que en el problema anterior se pueden marcar o comprometer, las modificaciones. Esto se hace poniendo una bandera en el dato R cuando un valor se ha modificado por la primera operación; de esta manera sabemos si éste es un valor real o un valor probable. Si alguna otra transacción quiere trabajar con R, puede hacerlo con un valor real y no provocar errores y análisis inconsistentes (figura 3). En este supuesto la segunda transacción tiene que esperar a que la primera quede concluida.

		A	B
X	lee 5	X	lee 5
y	lee 3	X	resta 5-2
$x+y$	suma 5+3	X	escribe 3

Fig 3: INCONSISTENCIA

En el último caso, *análisis inconsistentes*, al igual que en el anterior vamos a tener errores en cascada. El ejemplo de la figura 3 muestra un análisis inconsistente; cuando la transacción A requiere la suma de los valores X y Y, en ese momento lee el valor de un registro X que vale 5 y lo suma. Por otro lado la transacción B lee el valor de X y le resta 2; ahora X es igual a 3, B actualiza y compromete, pero A lee el valor de Y que es 3 y lo suma a su total, que ahora es de 8. Este valor está equivocado ya que el valor que debería tener es el de 6, puesto que X es igual a 3 y Y es también igual a 3.

### 1.1.2. Sistema de base de Datos Distribuido.

Un sistema de base de datos distribuido es un conjunto de **localidades** conectadas entre sí; cada localidad es una base de datos en sí misma pero trabaja con las otras localidades para que un usuario pueda tener información de cualquier localidad; es decir, como si toda la base de datos estuviera en su propia localidad, sin notar ninguna diferencia.

La principal diferencia entre una base de datos centralizada y una distribuida radica en que la segunda se encuentra esparcida entre todas las localidades.

Entre las ventajas que podemos encontrar en una base de datos distribuida tenemos, que los usuarios; (empresas, instituciones, autoridades, etc.) al estar distribuidos físicamente, no requieran de toda la información para poder trabajar, por lo que pueden separar la información más utilizada

en un servidor local y al momento de requerir información de otro lado, poder usar la red para la consulta. Esto debe ser transparente para el usuario y como la utilización de la red nos consume tiempo, tenemos que hacer bien la distribución de la misma y de los datos. Por otro lado, si este acceso fuera de una forma centralizada, los mensajes en la red serían muy grandes y harían las consultas más lentas. Desde el punto de vista del usuario, un sistema distribuido debe mostrarse idéntico a uno centralizado. Esto causa problemas al diseñador de la base de datos distribuida. Los primeros van a ser de carácter físico y económico, esta clase de problemas los podemos ver en:

- a) El tipo de instalación de la red donde el costo de comunicación, que es el de enviar los mensajes de un punto a otro, va a variar dependiendo de la topología existente.
- b) La frecuencia con la que falla una línea de comunicación o una localidad, haciendo más lento el flujo de mensajes o aislando partes de la red y
- c) La disponibilidad, que es la posibilidad de acceder a la información a pesar de los fallos en la comunicación, teniendo distintas rutas para los nodos.

Los problemas de desarrollo que se generan por todas estas variaciones son:

- a) El costo del desarrollo del software, que es más alto debido a la dificultad de estructurar un sistema de base de datos distribuido.
- b) La mayor posibilidad de errores, dado que las localidades del sistema operan en paralelo lo que dificulta el desarrollo de algoritmos confiables y tolerantes a fallas y
- c) Un mayor tiempo de procesamiento, por el envío de mensajes.

Las medidas de seguridad que afectan el diseño de una base de datos distribuida y que hay que tomar en cuenta son tres principalmente:

- a) Las réplicas, que es tener al menos una copia igual de la misma relación, debiéndose almacenar cada copia en localidades distintas, para disminuir el riesgo de su pérdida.
- b) La fragmentación, que consiste en dividir la relación en varias partes; esta fragmentación puede ser vertical, horizontal o mixta y cada parte debe almacenarse en localidades distintas y
- c) La fragmentación con réplica, que es una mezcla de las dos anteriores y que debe seguir los mismos criterios.

Las ventajas que vamos a encontrar en estas tres medidas son:

- a) La disponibilidad, que nos permite mantener el sistema funcionando, al disponerse de réplicas para el caso en que tenga fallas o daños alguno de los archivos o de los nodos.
- b) Mayor paralelismo, que permite que cada proceso puede hacer su consulta sobre réplicas diferentes y trabajar en paralelo, en el caso de que varios de ellos hagan una consulta sobre una misma relación.

Sin embargo no todo es tan simple como parece, tenemos que ver que todas las réplicas sean consistentes, lo que implica que cada vez que se hace una actualización de la base de datos se

necesite hacer una actualización de todas las réplicas. Es por ello que es necesario hacer un buen cálculo del número de réplicas que se van a necesitar.

En un artículo publicado en 1990, C.J.Date (2), propone las siguientes doce reglas que deben contemplar los sistemas de bases de datos distribuidos:

#### **1.1.2.1. Autonomía local.**

Cada localidad debe ser autónoma, la autonomía local indica que una localidad no debe depender de otra para su perfecto funcionamiento. Implica también que la administración y control de datos son de responsabilidad local. Lo anterior, como lo aclara el autor en su artículo, es imposible de lograr en un cien por ciento y hay que ceder en cierto grado el control a otra localidad.

#### **1.1.2.2. No dependencia de un sitio central.**

Este punto se convierte en un corolario de la autonomía local, ya que si se es autónomo no se depende de otra localidad. Es bueno requerirlo por la siguiente razón: si hubiera una dependencia de proceso de una localidad central, esta misma dependencia formaría un cuello de botella, porque al fallar la localidad central fallaría el resto del sistema.

#### **1.1.2.3. Operación continua.**

Al igual que en un sistema centralizado, lo ideal sería no apagar el equipo para agregar nuevas localidades al sistema o para eliminarlas.

#### **1.1.2.4. Independencia con respecto a la localización.**

No debe ser necesario que los usuarios del sistema conozcan dónde se encuentran físicamente los datos, únicamente deben de saber cuál es el nombre de las relaciones y sus características.

#### **1.1.2.5. Independencia con respecto a la fragmentación.**

La fragmentación debe ser ideal para no perder tiempo con mensajes en la red. Por ejemplo si nosotros tenemos una aplicación en donde se trabaja con una nómina en México y otra en Canadá, pero debemos de seguir teniendo toda la información de la nómina para cálculos globales. La fragmentación permite otorgar los registros más utilizados por una localidad a ella.

#### **1.1.2.6. Independencia de réplica.**

La autonomía de las réplicas simplifica los programas de los usuarios y sus actividades en terminal. En particular permite la creación y eliminación dinámica de réplicas.

#### **1.1.2.7. Procesamiento distribuido de consultas.**

Se requiere saber cuál es la mejor estrategia para mandar un resultado a la localidad que lo esté solicitando, dependiendo de la velocidad y el número de mensajes que van a viajar por la red.

#### **1.1.2.8. Manejo distribuido de transacciones.**

Es necesario pensar en qué forma queremos trabajar, si mandamos pedir una tabla y en nuestra localidad vamos a hacer el proceso o si vamos a solicitar que se procesen los trabajos en las localidades en donde están las bases de datos. En el segundo caso el proceso se crea en cada localidad y hay que esperar a que termine para que nos mande por la red únicamente el resultado.

#### **1.1.2.9. Independencia con respecto al equipo.**

Se requiere poder trabajar con un mismo DBMS sobre distintas plataformas, sin que esto implique un retraso en el proceso.

#### **1.1.2.10 Independencia con respecto al sistema operativo.**

Al igual que en el punto anterior, resulta obvia la conveniencia de poder trabajar con un mismo DBMS en distintos sistemas operativos.

#### **1.1.2.11. Independencia con respecto a la red.**

No importa qué tipo de red se trate o la topología de ésta, para el usuario no debe haber diferencia.

#### **1.1.2.12. Independencia con respecto al DBMS.**

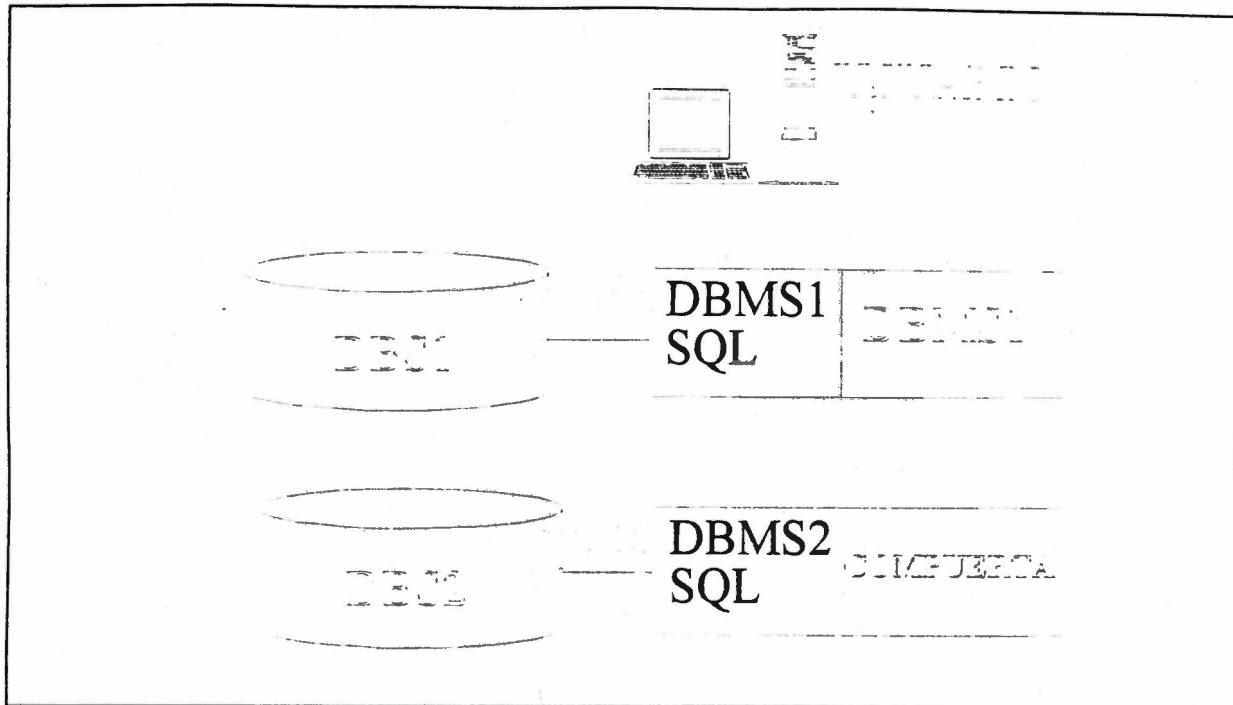
El ideal de una base de datos distribuida sería el que se manejara un cierto lenguaje oficial como SQL para que fuera en cierta forma una base homogénea, pero desgraciadamente en la realidad esto no funciona ya que se tienen equipos diferentes, sistemas operativos diferentes y DBMS también diferentes, haciendo muy difícil la interacción de todos estos factores.

### **1.2. FUNCIONAMIENTO DE BASE DE DATOS DISTRIBUIDAS.**

Analizaremos previamente como funcionan los manejadores de bases de datos distribuidas más conocidos.

Hay dos tipos de manejadores de bases de datos comerciales que son:

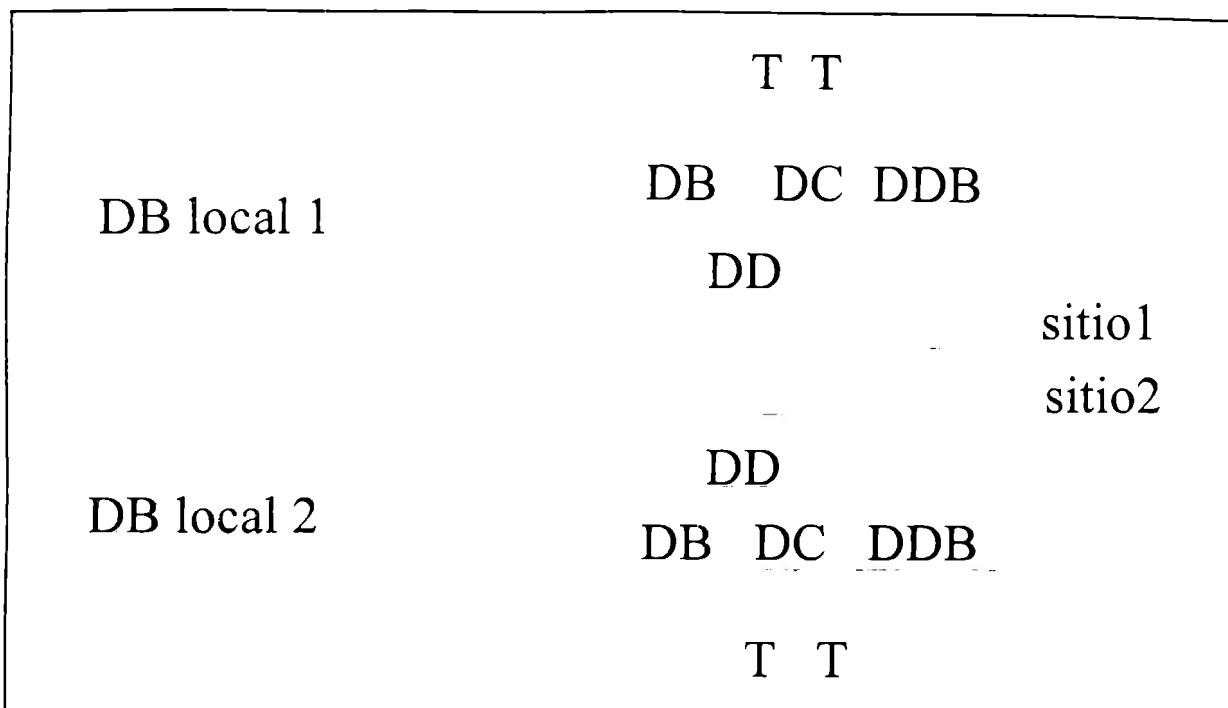
- a) Los que crean comunicación entre manejadores de datos por medio de compuertas.
- b) Los formados por un manejador de base de datos, un diccionario de datos, un componente de comunicación de datos y un manejador distribuido de bases de datos.



**Fig 4: MANEJADORES CON COMPUERTAS**

En el primer caso, se podría pensar que los manejadores de bases de datos trabajan con una misma interfaz, para que las comunicaciones entre un manejador y otro manejador sean completamente transparentes para el usuario de la base de datos. En la figura 4 vemos dos bases de datos que pueden estar en un mismo sitio, con un usuario del manejador de bases de datos, DBMS1, que quiere hacer una consulta global. El objetivo de la compuerta es manejar los protocolos para el intercambio de información entre los manejadores de bases de datos. Realizan una función de despachador relacional, consistente en la capacidad de ejecutar proposiciones de SQL no planeadas, hacer corresponder los tipos de datos entre las dos bases de datos, hacer corresponder el dialecto entre los dos manejadores, hacer corresponder el diccionario de las bases de datos, funcionar como participante en el protocolo a dos fases y asegurar que los datos cuyo bloqueo sea requerido, realmente sean bloqueados.

El segundo caso de manejadores (figura 5) son los formados por los componentes siguientes: un manejador de base de datos, que va a controlar la base de datos local; un diccionario de datos, el cual va a contener la información general de dónde se encuentran los datos; un componente de comunicación de datos, que va a ser la interfaz con las terminales y un manejador distribuido de base de datos, que va a controlar el intercambio de información entre las distintas bases de datos. Este modelo contempla dos tipos de acceso a bases de datos remotas, que son por medio de funciones primitivas y por medio de un programa auxiliar.



**Fig 5:MANEJADORES ADMINISTRADOS POR UN LOCAL**

### 1.3. BENEFICIOS EN EL USO DE BASES DE DATOS DISTRIBUIDAS.

Las razones que tienen las empresas, instituciones o usuarios en general para utilizar una base de datos distribuida son de diferente naturaleza. A continuación analizamos las seis que consideramos de mayor importancia:

- Organizacional. Las empresas, al crecer física y económicamente, van descentralizando su lugar de trabajo; lo que hace más complejo esto, es que a veces se encuentran en ciudades o países distintos los diferentes departamentos o secciones que la integran. En estos casos, no se puede tener toda la información en un solo lugar o en el otro extremo una réplica en todos esos lugares, ya que tendríamos muy serios problemas con la integridad de los datos de la base general.
- Económica. Las razones económicas vienen de la mano con la organizacional, ya que si tenemos que hacer una réplica de la base de datos en cada sitio donde vamos a trabajar, el espacio del disco local debe ser más grande. Por otro lado, el costo de llevar una base de datos completa por medio de la red es alto y el proceso muy lento. Por otro lado, si no hiciéramos réplicas, dependeríamos de la conexión con el servidor de datos centralizado y en el caso en que éste falle podría representarnos grandes pérdidas, al retrasar el trabajo de toda la organización.
- La interconectividad de las bases de datos. Es la solución natural para poder trabajar con bases de datos que fueron desarrolladas con anterioridad y se requiere formar con todas esas bases una

aplicación global. Un ejemplo que permite ver este caso más claro es el de la unión de dos empresas, donde cada una de ellas tiene desarrollada su propia base de datos, con un manejador diferente; en lugar de pasar toda la información de una base de datos a otra, la solución ideal sería crear una relación entre ellas para formar una base de datos global.

d) Reducción en el sobreflujo de mensajes dentro de la red. Si se tuviera una base de datos centralizada, al requerir una consulta todas las localidades deberían hacer contacto con el servidor de datos, sin importar donde esté y por cuantos nodos tenga que pasar. Si la localidad que va a hacer la consulta está cerca del servidor de datos puede tener su información rápidamente. Por el contrario la consulta de un nodo más alejado puede tardar un tiempo considerablemente más alto por cruzar por una cantidad mayor de nodos. Por otra parte, si tenemos una base de datos distribuida geográficamente, la cantidad de nodos por la que va a viajar nuestra consulta va a ser la menor posible, mejorando esto el desempeño de la base de datos y de la red en general.

e) Las consideraciones de desempeño. El hecho de que la base de datos se encuentre distribuida en varios sitios y cada sitio tenga su propio procesador, hace más eficiente el proceso, al hacerlo en forma simultánea.

f) Confiabilidad y disponibilidad. La utilización de réplicas permite tener una mayor confiabilidad y disponibilidad, porque si falla una línea y perdemos comunicación con una réplica, podremos tomar una ruta distinta o utilizar una réplica diferente y poder así tener una mayor accesibilidad a los datos. Esto genera una base de datos más segura y garantiza una máxima confiabilidad en la operación.

Los puntos anteriores nos muestran la conveniencia de utilizar una base de datos distribuida.

Nota (1): Date, C.J., "Introducción a los sistemas de bases de datos", Volumen 1, quinta edición, Addison-Wesley Iberoamericana, USA, 1993, página 113.

Nota: El mismo autor, en la página 261 de su obra, ofrece una definición más formal y la define de la siguiente forma "Una base de datos relacional es una base de datos percibida por el usuario como una colección de relaciones normalizadas de diversos grados que varía con el tiempo".

Nota (2): Date C.J., "What is a Distributed Database System", en C.J.Date Relational Database Writings 1985-1989. Addison-Wesley, Reading, Mass. 1990.

## CAPÍTULO 2.

### PLANTEAMIENTO DEL PROBLEMA

#### 2.1. JUSTIFICACIÓN

Hemos analizado en el primer capítulo el estado actual de las bases de datos relacionales y de los sistemas de base de datos distribuidos, enfocandolos en los problemas que puede haber en las primeras y en las características con que deben contar los segundos. Igualmente hemos estudiado el funcionamiento de las bases de datos distribuidas y los beneficios que se obtienen con su utilización.

Ahora que conocemos las ventajas de las bases de datos distribuidas y el funcionamiento de sus manejadores, en el presente trabajo proponemos crear un **administrador**, que funcione por encima de estos, destinado a mejorar los manejadores de bases de datos existentes en el mercado.

El administrador de bases de datos que se propone tiene como finalidad resolver el problema de la transparencia de acceso en los archivos, mediante la creación de un esquema global y un esquema local que explicaremos más adelante.

El desarrollo del trabajo cubre los siguientes puntos: la elaboración de un diseño conceptual, la elaboración de un diseño global, el análisis de los conceptos de replicación y fragmentación, y la propuesta de un administrador, como base de las conclusiones.

## **2.2. OBJETIVOS.**

Los objetivos que se buscan en el administrador propuesto, mismos que implican la resolución de diversos problemas en su diseño y funcionamiento, son los siguientes:

### **2.2.1. Continuidad.**

El administrador debe ser capaz de proponer rutas alternativas para la localización de archivos o consulta en las réplicas.

### **2.2.2. Tolerancia a Fallas.**

No importando que algún nodo no se encuentre funcionando, si existe otro nodo conectado, el administrador sera capaz de encontrar otras réplicas para su consulta.

### **2.2.3. Transparencia.**

Nuestro administrador debe crear un esquema global de archivos con información de la ubicación de estos y el lugar en donde se encuentran las réplicas y un esquema local, el cual va a servir para tener la ubicación de los archivos más utilizados por el sitio, agregando al nombre del archivo la dirección de la red en la que se encuentren.

### **2.2.4. Consistencia.**

Nuestro administrador intentará obtener constancia mediante la aplicación de un algoritmo que nos permita la actualización completa de las réplicas y las transacciones.

### **2.2.5. Recuperación.**

Mediante un algoritmo que nos permita actualizar los archivos del sitio cada vez que es dado de alta o que entra a la distribución.

## CAPÍTULO 3: DISEÑO CONCEPTUAL

### 3.1. ELEMENTOS BÁSICOS DE UNA BASE DE DATOS DISTRIBUIDA.

En un sistema multibase de datos que se encuentra repartido en distintos nodos de una red, ya sea **LAN** o **WAN**, podemos encontrar que no todas las bases están controladas por un mismo manejador de bases de datos o que tampoco están bajo un mismo sistema operativo. En este caso estamos ante un sistema de base de datos heterogéneo.

En este capítulo vamos a trasladar los requerimientos de un sistema manejador de base de datos distribuido a un modelo formal, integrado e independiente de dicho manejador de base de datos. Definiremos los requerimientos operacionales, en donde vamos a incluir el modo y la forma en la que se van a presentar los datos en cada una de las partes del sistema para facilitar el manejo de éstos; los requerimientos de transacciones, incluyendo el tipo de protocolo de comunicación empleado y los mecanismos para aceptar o cancelar transacciones; y los requerimientos sobre restricciones, en donde vamos a ver las posibles restricciones sobre los datos, las operaciones y los eventos relacionados con el control de la base de datos.

Para poder lograr crear un administrador que trabaje según lo descrito en el capítulo anterior, debemos contar con los siguientes elementos:

### **3.1.1. Autonomía De Diseño.**

No podemos modificar los sistemas manejadores de base de datos locales existentes para acoplarlos a nuestro sistema global. Lo anterior se debe a que hacer cambios al software existente es costoso, puede resultar en una degradación de la eficiencia y puede dejar inoperantes a las aplicaciones existentes.

### **3.1.2. Autonomía De Ejecución.**

Cada sistema manejador de base de datos local conserva el control completo sobre la ejecución de las transacciones en ese sitio. Una implicación de esta restricción es que el sistema manejador de base de datos local puede abortar una transacción que se ejecute en ese sitio en cualquier instante durante su ejecución, incluyendo el tiempo de cuando una transacción global esté en proceso de compromiso por el sistema global. Debemos encontrar la forma de detectar estos eventos para poder dar aviso al resto del sistema.

### **3.1.3. Autonomía De Comunicación.**

Los sistemas manejadores de bases de datos locales, integrados por el sistema global, no son capaces de coordinar las acciones de transacción global que se ejecutan en varios sitios. Esta restricción implica que el sistema manejador de base de datos local no comparte su información de control con los otros sistemas manejadores de bases de datos o con el sistema global. Es el sistema global quien debe coordinar el flujo de información y de comandos.

Uno de los principales problemas de una consulta global en un sistema de bases de datos distribuidos, es la gran cantidad de datos que vamos a transmitir. A continuación vamos a mencionar un estudio de las cuatro estrategias, elaborado por Victor J. Sosa y Sosa et al (3), mismo que se ilustra con la figura 6.

### 3.1.4. Estrategias Comunes.

Supongamos tres localidades, Alfa, Beta y Gama y dos archivos S y SP, los cuales están en Beta y Gama respectivamente. El tamaño del registro en S en Beta es de 28 bytes y el tamaño del registro en SP es de 12 bytes en Gama, además que sólo existen 5 registros en S y 3 en SP.

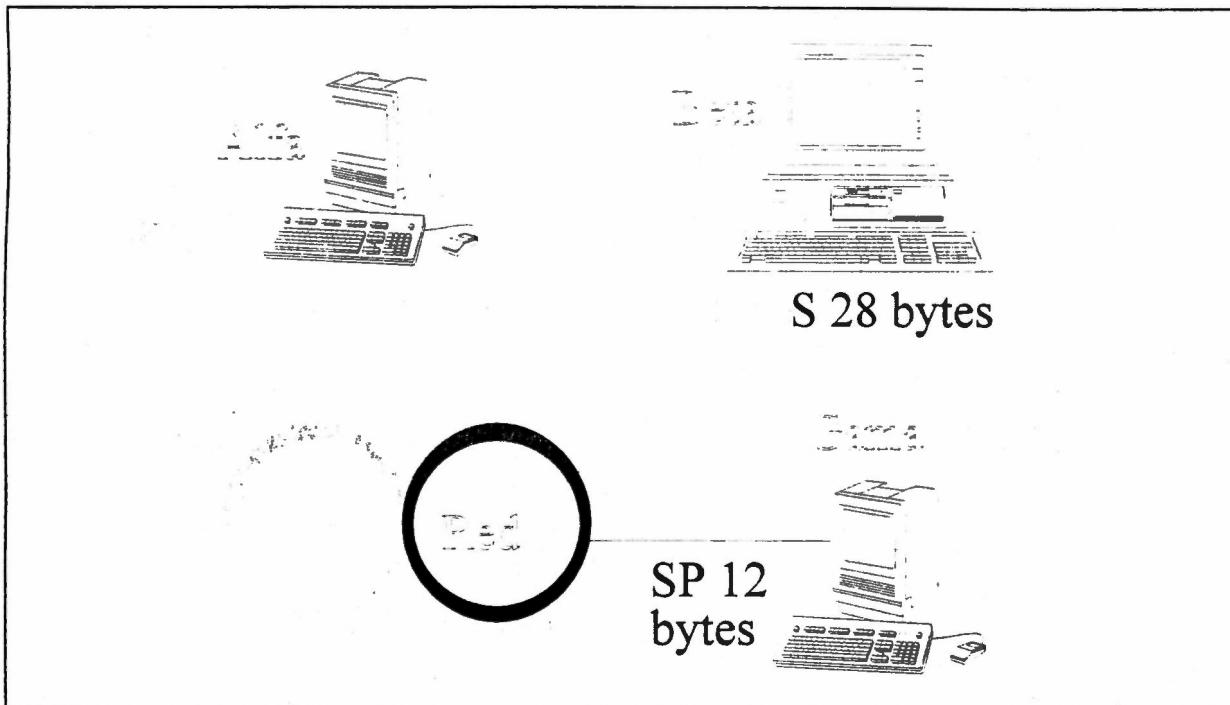


Fig 6: ESTRATEGIAS

Si la consulta produce a X que son los registros de S que cumplen la condición y a Y que son los registros de SP que cumplen la condición, tenemos las siguientes estrategias:

#### 3.1.4.1. Estrategia 1.

Enviar las tablas S y SP a la máquina Alfa y procesar ahí la consulta. La cantidad de datos transmitidos por parte de Beta es

$$28 \times 5 = 140 \text{ bytes}$$

y por parte de Gama

$$12 \times 3 = 36 \text{ bytes}$$

dando un total de bytes transmitidos: 176.

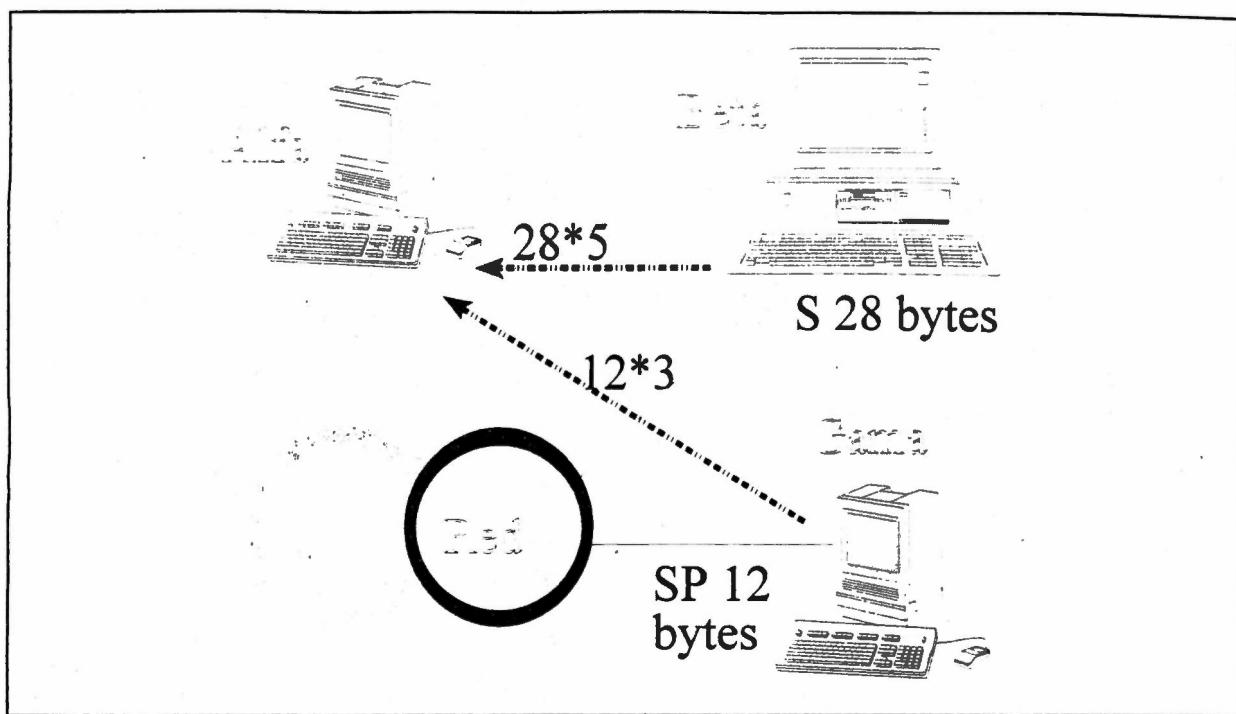


Fig 6.1ESTRATEGIA 1

### 3.1.4.2 Estrategia 2.

Enviar la tabla SP a Beta y ahí realizar la consulta para posteriormente enviar los resultados a Alfa. La cantidad de datos transmitidos por Gama es de 36 bytes. El número de bytes transmitidos por Beta hacia Alfa (sólo los que cumplen la condición) es de

$$28Y + 12X$$

entonces tenemos

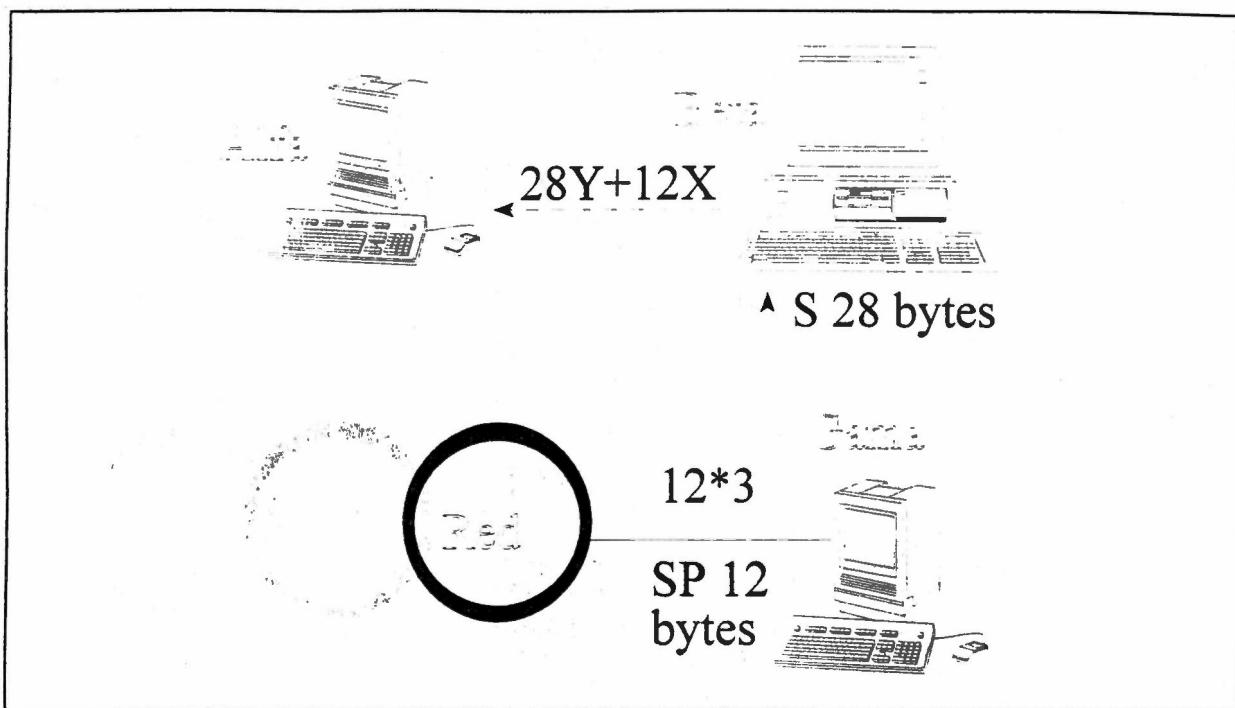


Fig 6.2: ESTRATEGIA 2

TABLA 3: RESULTADOS DE LA ESTRATEGIA 2

Y\X	0	1	2	3
0	36	48	60	72
1	64	76	88	100
2	92	104	116	128
3	120	132	144	156
4	148	160	172	184
5	176	188	200	212

El número total de bytes transmitidos en la red es

$$36 + 28Y + 12X = 36 + 28Y + 12X$$

dando un promedio de 124.

### 3.1.4.3. Estrategia 3.

Enviar la tabla S a Gama y ahí procesar la consulta para posteriormente enviar los resultados a Alfa. La cantidad de datos transmitidos por Beta es de 140 bytes; mientras que el número de bytes transmitidos por Gama a Alfa es

$$12X + 28Y$$

El número total de bytes transmitidos en la red es:

$$140 + 12X + 28Y$$

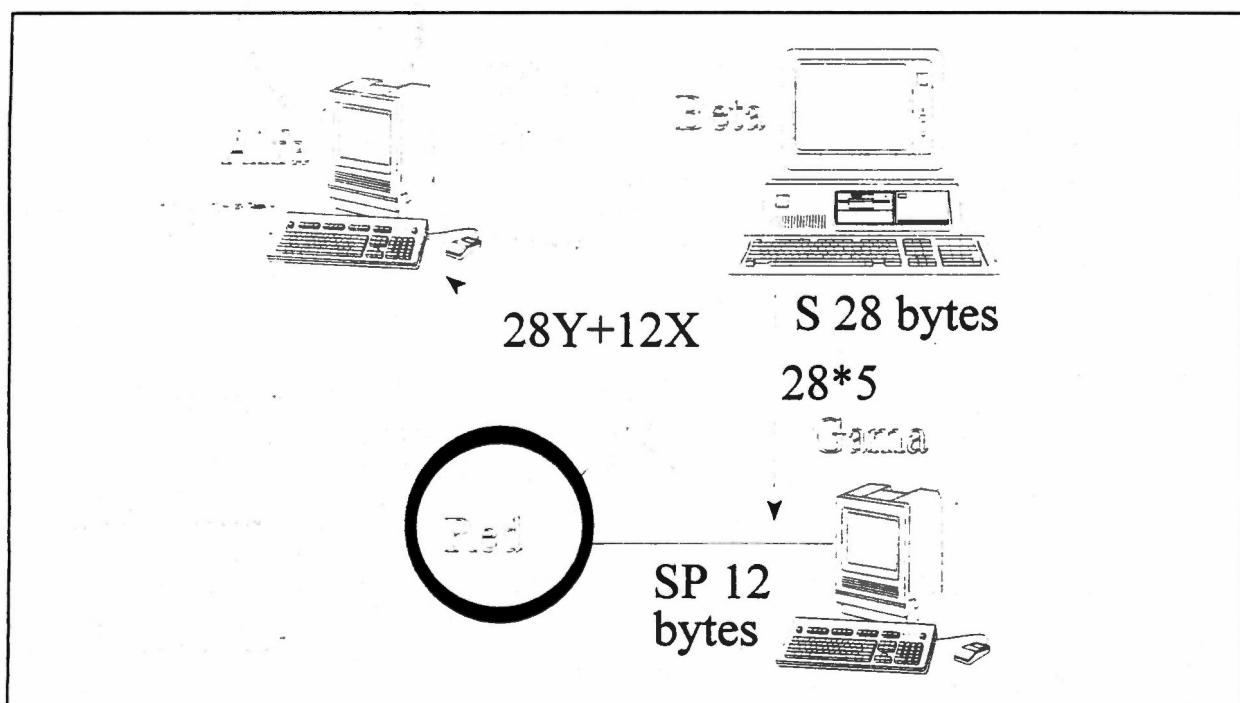


Fig 6.3: ESTRATEGIA 3

**TABLA 4: Resultados de la estrategia 3**

<b>Y\X</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>
<b>0</b>	140	152	164	176
<b>1</b>	168	180	192	204
<b>2</b>	196	208	220	232
<b>3</b>	224	236	248	260
<b>4</b>	252	264	276	288
<b>5</b>	270	292	304	316

Si tomamos un promedio de los bytes transmitidos sería de **228**

#### **3.1.4.4. Estrategia 4.**

Enviar subconsultas a las máquinas Beta y Gama. El número de bytes transmitidos de Beta a Alfa es de **28 Y**, y el número de bytes transmitidos de Gama a Alfa es de **12 X**. El total de bytes transmitido en la red es

$$\mathbf{28 Y + 12 X}$$

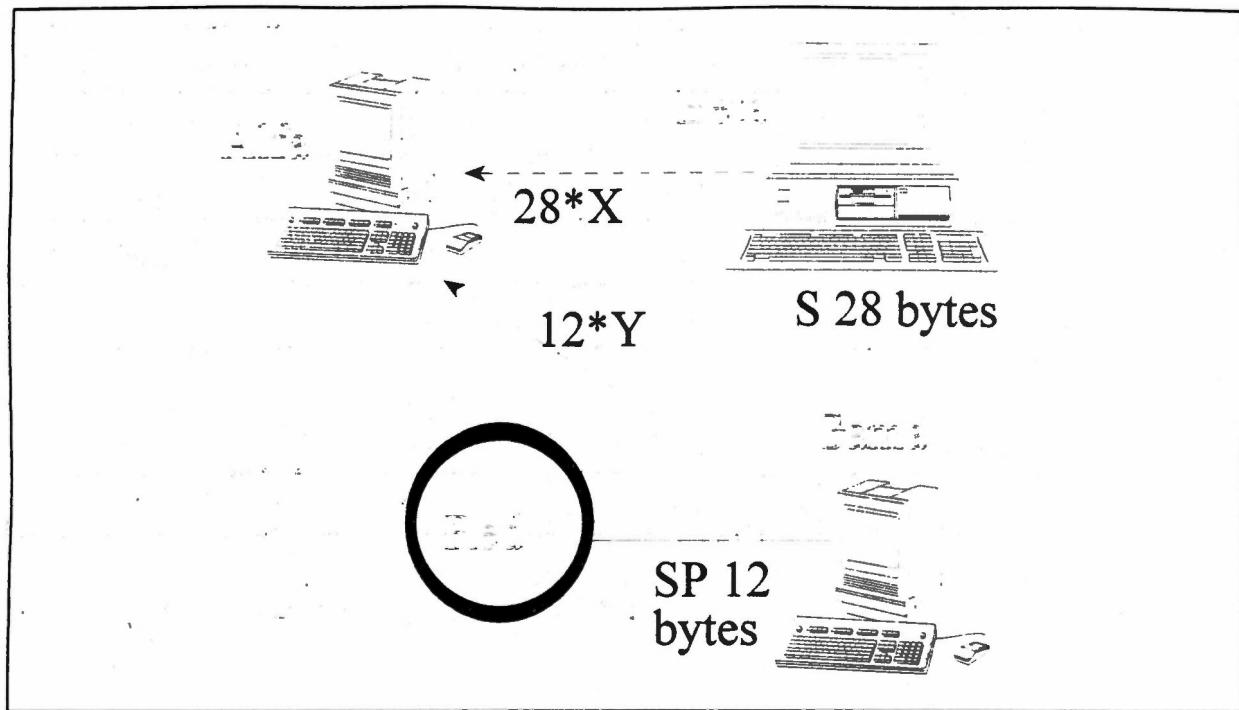


Fig 6.4: ESTRATEGIA 4

TABLA 5: Resultados de la estrategia 4

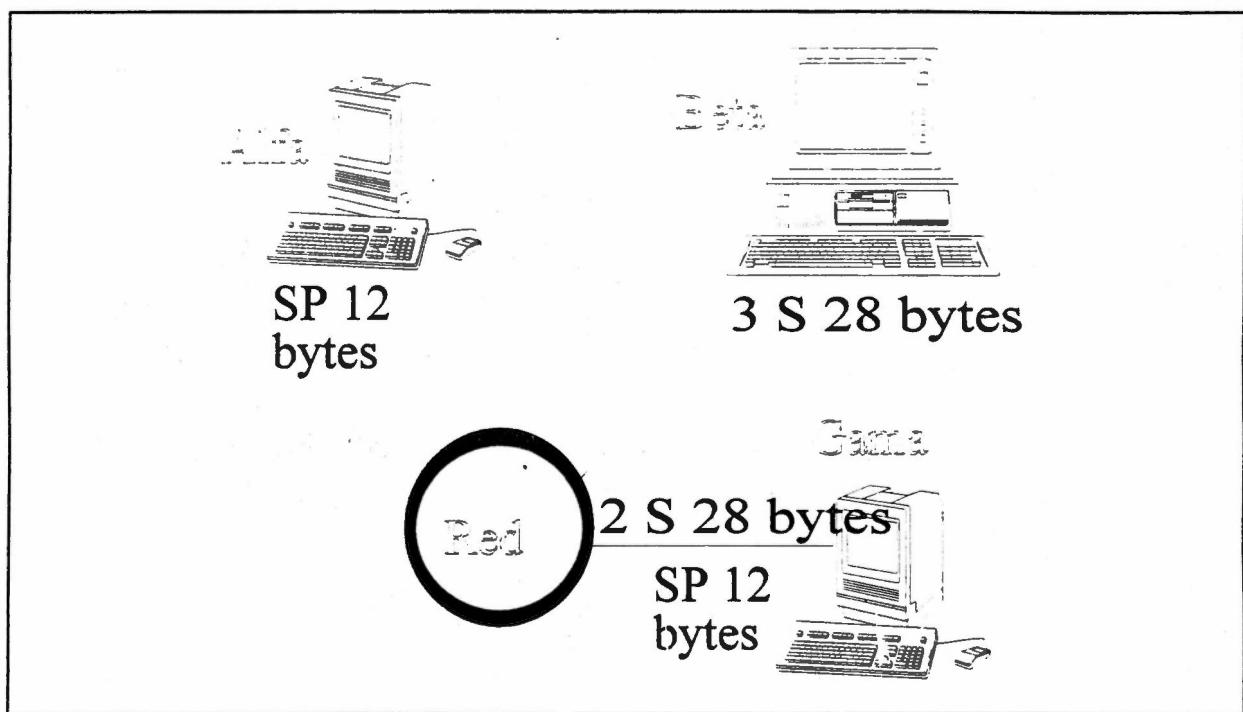
$Y \setminus X$	0	1	2	3
0	0	12	24	36
1	28	40	52	64
2	56	68	80	92
3	84	86	98	110
4	112	124	136	148
5	140	152	164	176

Si tomamos el promedio sería de 88

Como podemos observar, para esta consulta la mejor opción que se presenta es la estrategia 4, la cual consiste en mandar las subconsultas a Beta y a Gama. Tenemos que tener presente que en este ejemplo la cantidad de datos es muy pequeña; sin embargo en sistemas reales cada transmisión podría involucrar miles de bytes.

### 3.1.5. Estrategias Distribuidas.

A continuación para un administrador distribuido, se proponen las siguientes estrategias, tomando en cuenta que tenemos fragmentación y replicación, utilizando el mismo ejemplo de la figura 6. Supongamos que hay una consulta de Alfa, que el tamaño del registro en S es de 28 bytes, que el tamaño del registro en SP es de 12 bytes y además que solo existen 5 registros en S y 3 en SP.



**Figura 7:ESTRATEGIAS DISTRIBUIDAS**

Hagamos una segunda suposición de que para SP existe una réplica en Alfa y de que S lo fragmentamos en dos archivos, en Beta y en Gama, con tres y dos registros respectivamente, podemos hacer los mismos planteamientos de las estrategias normales:

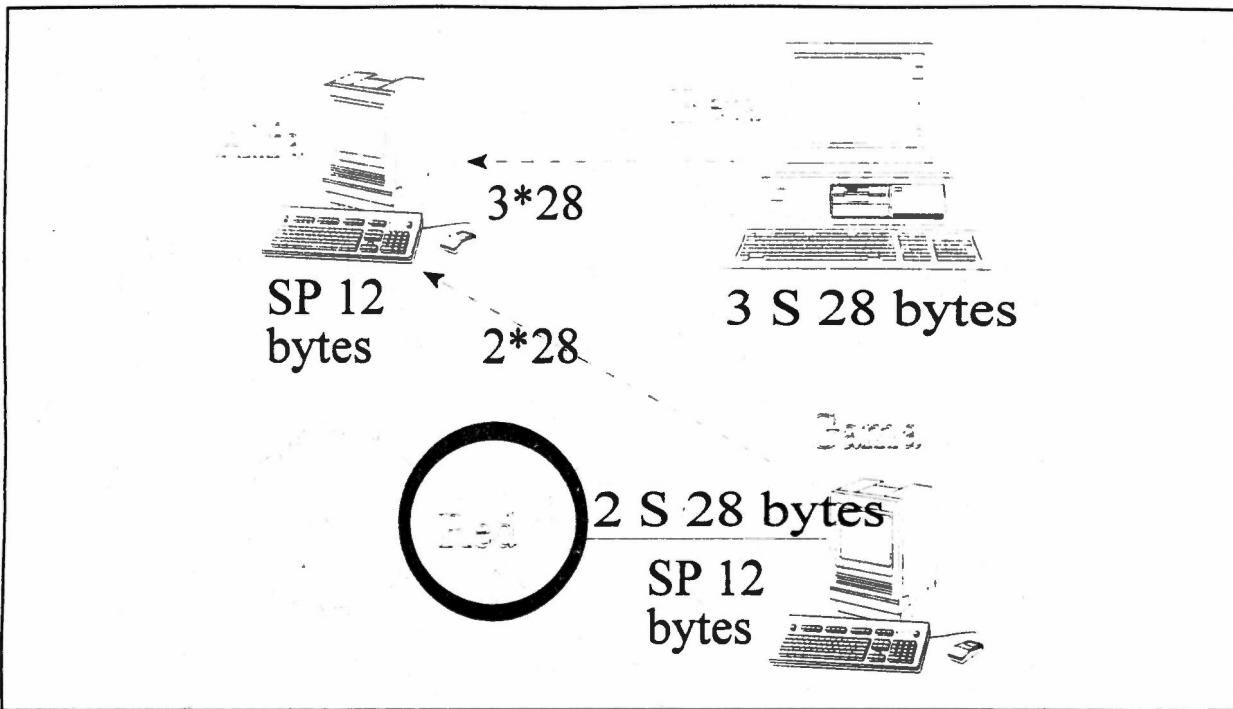
#### 3.1.5.1. Estrategia d1.

Enviar las tablas S y SP a la máquina Alfa y procesar ahí la consulta. La cantidad de datos transmitidos por parte de Beta es

$$28 \times 3 = 84 \text{ bytes}$$

y por parte de Gama

$$28 \times 2 = 56 \text{ bytes.}$$



**Fig 7.1: ESTRATEGIA D1**

Dando un total de bytes transmitidos de 140.

#### 6.5.2. Estrategia d2.

Enviar la tablas S y SP a Beta y ahí realizar la consulta para posteriormente enviar los resultados a Alfa. La cantidad de datos transmitidos por Gama es de

$$12 \times 3 = 36 + 28 \times 2 = 56 = 92 \text{ bytes}$$

El número de bytes transmitidos por Beta hacia Alfa (sólo los que cumplen la condición) es

$$28 Y + 12 X.$$

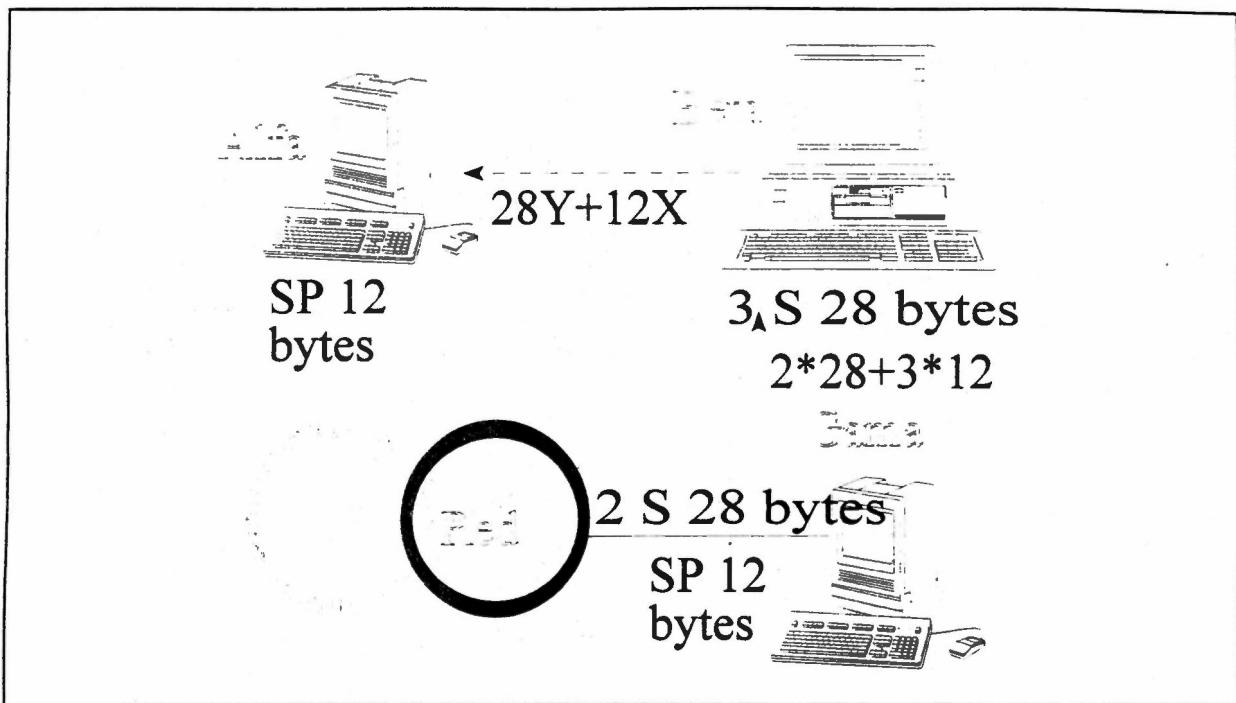


Fig 7.2: ESTRATEGIA D2

El número total de bytes transmitidos en la red es

$$92 + 28 Y + 12 X$$

TABLA 6: Resultados de la estrategia d2				
Y\X	0	1	2	3
0	92	104	116	128
1	110	122	134	146
2	138	150	162	174
3	166	178	190	202
4	194	206	218	230
5	222	234	246	258

Si tomamos el promedio es de 175.

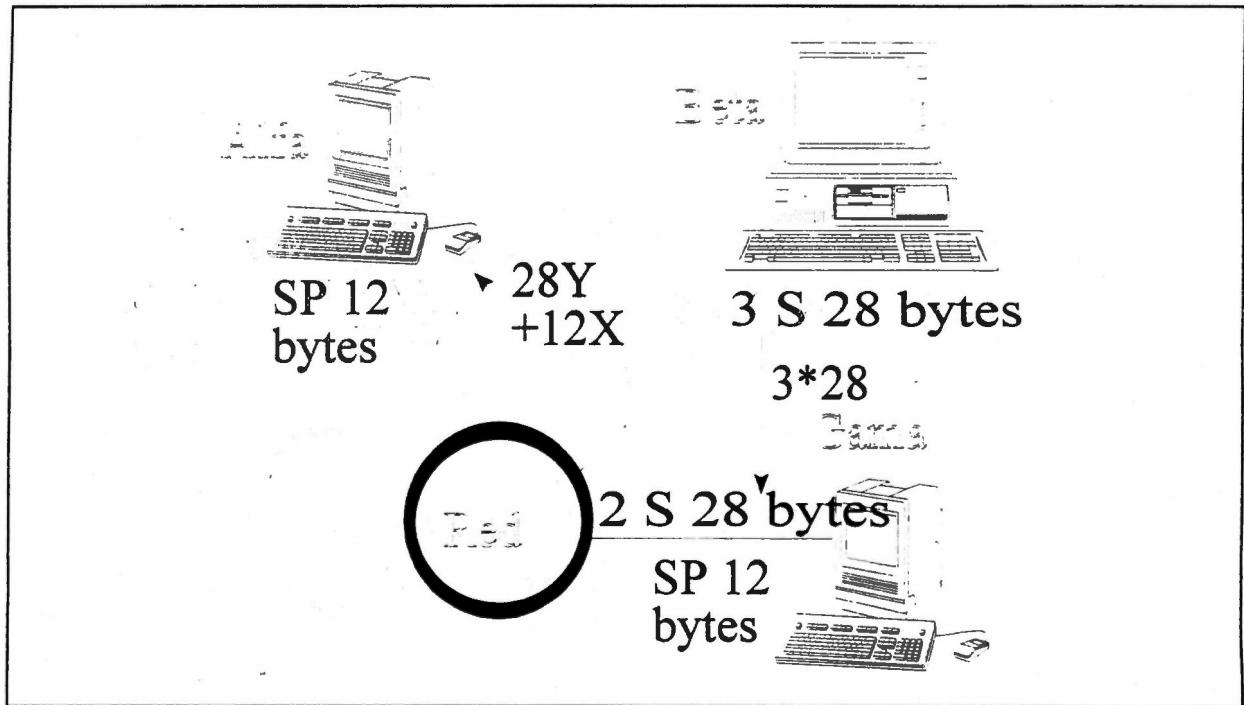
### 3.1.5.3. Estrategia d3.

Enviar la tabla S a Gama y ahí procesar la consulta para posteriormente enviar los resultados a Alfa.  
La cantidad de datos transmitidos por Beta es

$$28 \times 3 = 84 \text{ bytes}$$

mientras que el número de bytes transmitidos por Gama a Alfa es

$$12 X + 28 Y$$



**Fig 7.3ESTRATEGIA D3**

El número total de bytes transmitidos en la red es:

$$84 + 12 X + 28 Y$$

<b>TABLA 7: Resultados de la estrategia d3</b>				
<b>Y\X</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>
<b>0</b>	84	92	104	116
<b>1</b>	112	124	136	148
<b>2</b>	140	152	164	176
<b>3</b>	168	180	192	204
<b>4</b>	196	208	220	232
<b>5</b>	224	236	248	260

Si tomamos el promedio es de 172.

### 3.1.5.4. Estrategia d4.

Enviar subconsultas a las máquinas Beta y Gama. El número de bytes transmitidos de Beta a Alfa es  $28Y_1$ , donde  $Y_1$  es el total de registros que cumplen con la consulta en el fragmento de S en la localidad Beta y el número de bytes transmitidos de Gama a Alfa es

$$12X + 28Y_2$$

donde  $Y_2$  es el total de registros que cumplen con la consulta en el fragmento de S en la localidad Gama.

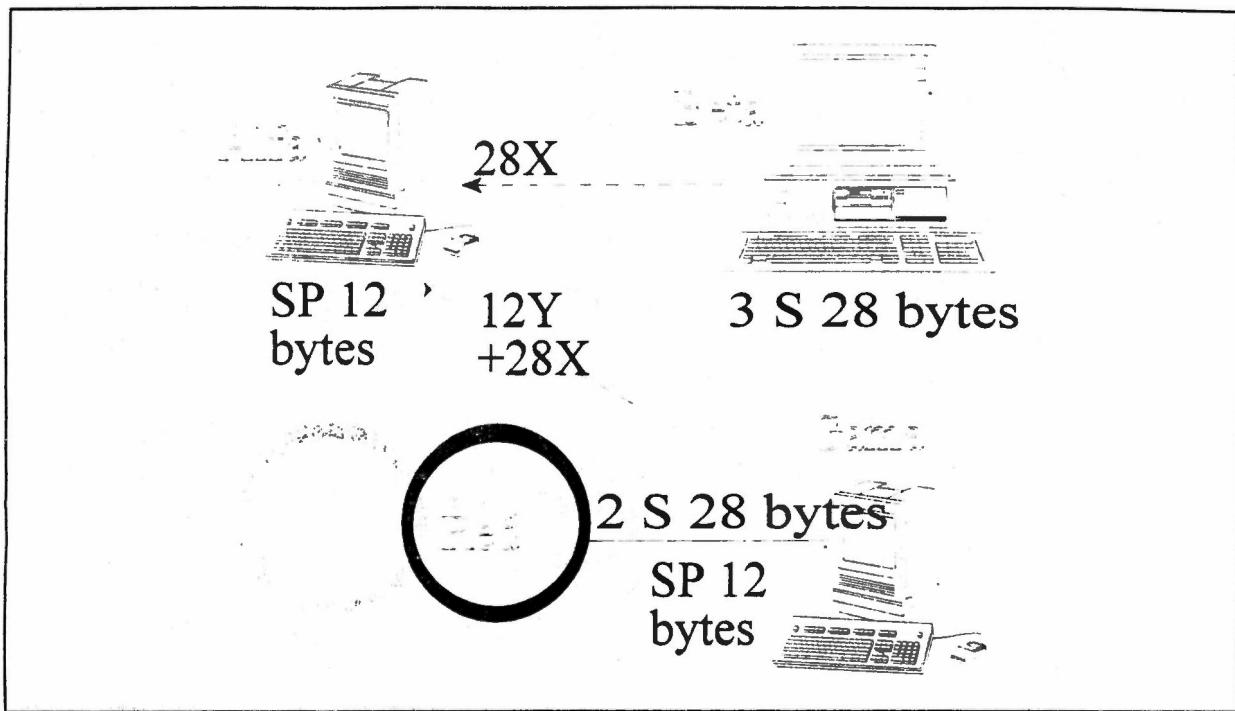


Fig 7.4:ESTRATEGIA D4

El total de bytes transmitido en la red es

$$28(Y_1+Y_2)+12X.$$

Para  $Y_1 = 0$

TABLA 8: Resultados de la estrategia d4

$Y_2 \setminus X$	0	1	2	3
0	0	12	24	36
1	28	40	52	64
2	56	68	80	92
3	84	96	108	120

Para  $Y_1 = 1$

<b>TABLA 9: Resultados de la estrategia d4</b>				
$Y_2 \setminus X$	0	1	2	3
0	12	24	36	48
1	40	52	64	76
2	68	80	92	104
3	96	108	120	132

Para  $Y_1 = 2$ .

<b>TABLA 10: Resultados de la estrategia d4</b>				
$Y_2 \setminus X$	0	1	2	3
0	24	36	48	60
1	52	64	76	88
2	80	92	104	116
3	108	120	132	132

Sacando un promedio de los posibles mensajes que viajarían dentro del sistema, serían 72.

### 3.1.5.5. Estrategia d5.

Hacer consulta en la réplica de SP de Alfa y enviar subconsultas a Beta y Gama en las fragmentos de S.

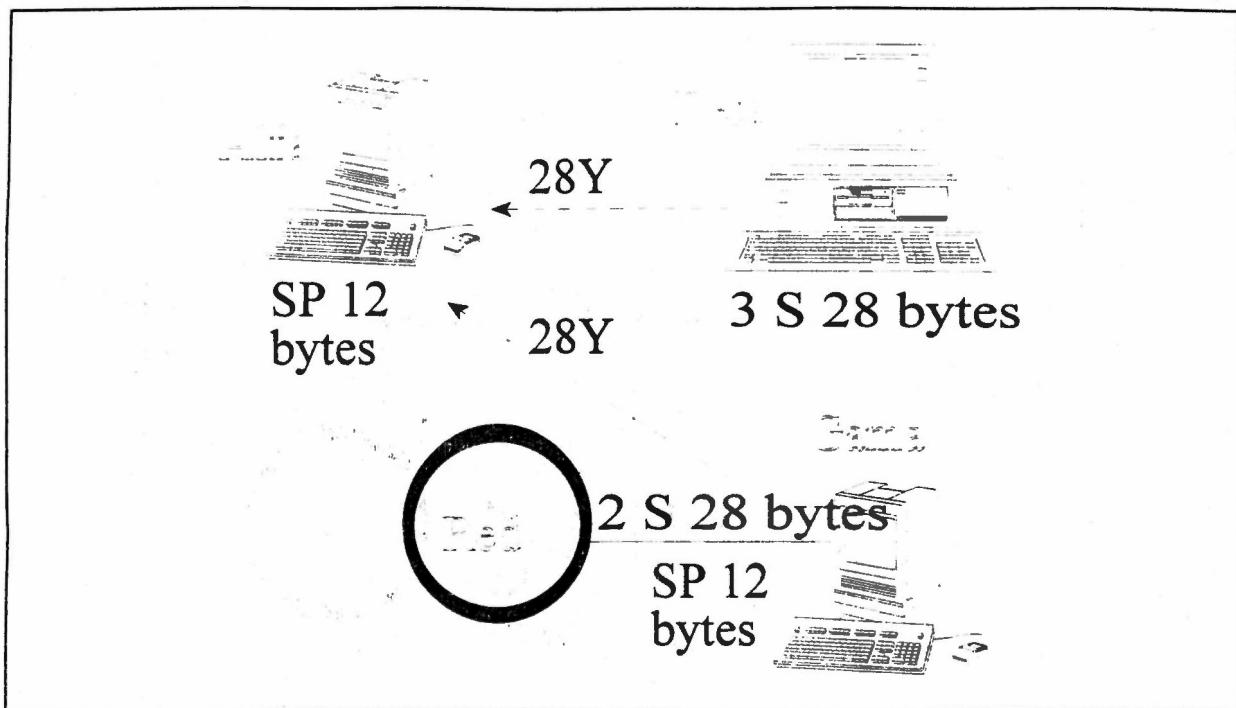


Fig 7.5: ESTRATEGIA D5

El número de bytes transmitidos de Beta o Gama a Alfa seria de

$$28Y_1 + 28Y_2.$$

TABLA 11: Resultados de la estrategia 5

$Y_1 \setminus Y_2$	0	1	2	3
0	0	28	56	84
1	28	56	84	76
2	56	84	76	104

Si tomamos el promedio seria de 52.

Si hacemos una comparación burda entre las estrategias, tomando en cuenta que en la segunda serie se hizo una réplica y una fragmentación al azar, podemos ver que en el mejor de los casos la distribución disminuye el trafico de mensajes en la red.

Podríamos decir que la mejor de las estrategias sería la d5 de la segunda serie, pero debemos tomar en cuenta que éste es un caso en el que una de las réplicas o fragmentaciones quedó en el lugar de donde partió la consulta.

Ya que tomamos estas consideraciones podemos empezar a hacer nuestro diseño.

Nota (3): Sosa Sosa Victor J., Joaquin Pérez O. y Rodolfo Pazos R., "Algoritmo para Descomposición de Consultas SQL en Subcuentas Bajo la Arquitectura Cliente/Servidor", Memorias del segundo Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.

## **CAPITULO 4:**

## **DISEÑO GLOBAL**

### **4.1. ESTRUCTURA DEL DISEÑO GLOBAL**

El diseño global de la arquitectura del administrador de base de datos involucra la solución integral del problema. El administrador que se propone se divide en dos esquemas, el Esquema Global, que es el encargado de indicar en qué parte de la red se encuentran los archivos, las réplicas y o los fragmentos de las réplicas y si todos éstos se encuentran bloqueados por no estar actualizados y el Esquema Local, que se encarga de tomar los archivos de las consultas de los manejadores de bases de datos y, en caso necesario, de hacer una consulta al Esquema Global, para saber la localización de los archivos y obtener un resultado. Si el archivo se encuentra distribuido propone una ruta óptima a la réplica más accesible, siguiendo los pasos representados en la figura 8.

#### 4.1.1. Proceso del diseño global

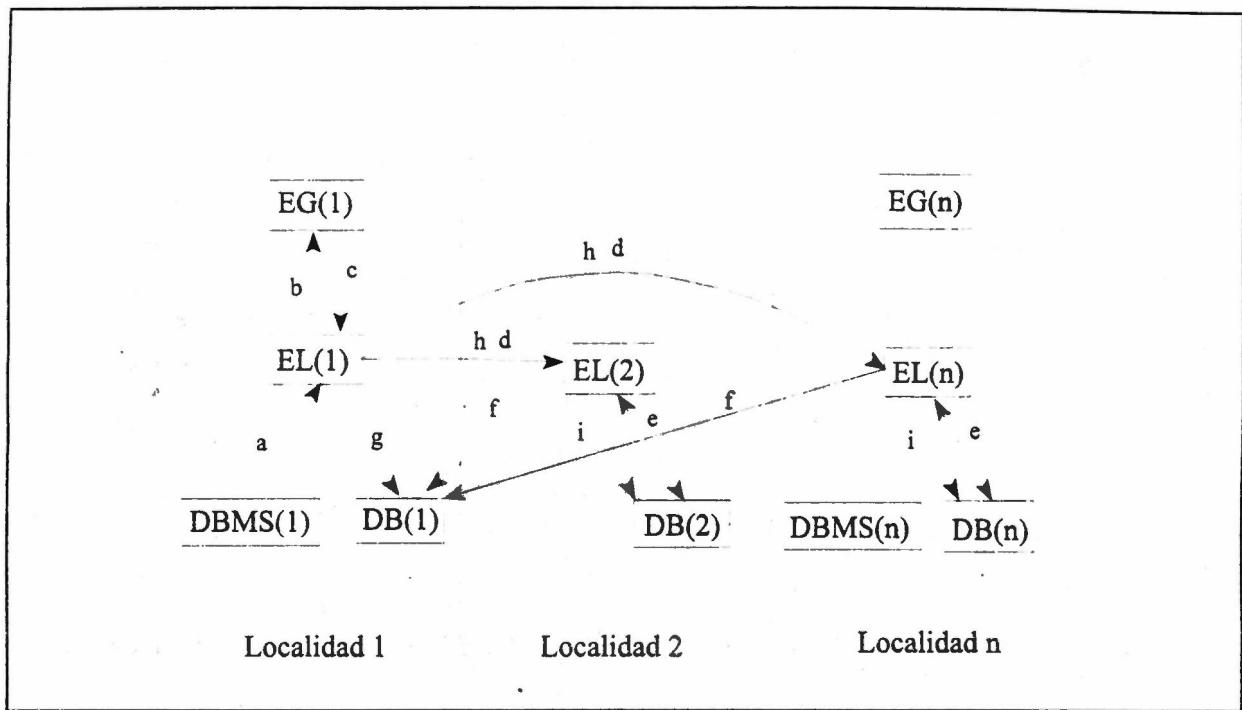


Fig 8: DISEÑO GLOBAL

- El DBMS<sub>1</sub> envía una consulta que es interceptada por el Esquema Local<sub>1</sub>, éste revisa los archivos involucrados, para ver si son distribuidos. Si la consulta es totalmente local, pasa al punto g). Nota: el número que aparece después de cada nombre se refiere a la localidad de su ubicación.
- El Esquema Local<sub>1</sub> hace una consulta al Esquema Global<sub>x</sub> que no necesariamente existe en cada localidad aunque, en nuestro caso si hay uno en la localidad. Esta consulta es con el fin de saber en qué localidad se encuentran los archivos, réplicas y/o fragmentos.
- El Esquema Global contesta al Esquema Local<sub>1</sub> con una lista de los archivos, ubicación en la red y si están fragmentados.
- El Esquema Local<sub>1</sub> se conecta con otros Esquemas Locales<sub>x</sub> que tengan algún archivo, fragmento o réplica y manda nombres temporales para su envío y el tipo de manejador que está pidiendo la consulta.
- Los Esquemas Locales<sub>x</sub> toman el DB<sub>x</sub> de su localidad el archivo; si no es compatible con el DBMS<sub>1</sub> que lo está solicitando, lo traduce y los manda al Esquema Local<sub>1</sub> y si sí es compatible, lo mandan directamente al Esquema Local<sub>1</sub>.
- El Esquema Local<sub>1</sub> recibe los archivos y escribe los primeros en llegar en uno temporal de DB<sub>1</sub>.
- El Esquema Local<sub>1</sub> hace la consulta del DBMS<sub>1</sub> con los archivos del DB<sub>1</sub> y revisa si hay cambios en los archivos temporales que generó.

h) Si existe modificación en algún archivo, el Esquema Local, manda únicamente los cambios a los Esquemas Locales, que tengan los archivos modificados y al terminar de mandar los cambios, el Esquema Local, manda borrar los archivos temporales generados por la consulta.

i) Los Esquemas Locales, si es necesario, traducen la modificación y la efectúan en los archivos de su DBx.

Como quedó dicho anteriormente el administrador se divide en dos módulos, el global y el local, y cada uno de éstos se subdivide en varios módulos menores, para tratar cada una de las responsabilidades.

El sistema operativo sobre el cual se implante el sistema distribuido puede proporcionarle algunas funciones, por lo que nos ayudaremos de éste en esas circunstancias, para trabajar en el diseño del sistema.

El sistema distribuido propuesto consta de dos partes, el Esquema Global y el Esquema Local.

## **4.2. ESQUEMA GLOBAL.**

Éste es el módulo que tiene el esquema general de la base de datos distribuida, conteniendo en su información: las direcciones lógicas, el status de cada archivo, las ubicaciones de las réplicas y/o de sus fragmentos y los Esquemas Locales que los controlan; este Esquema Global puede o no estar en cada localidad y como máximo habrá uno por localidad.

Está formado por los siguientes módulos de proceso:

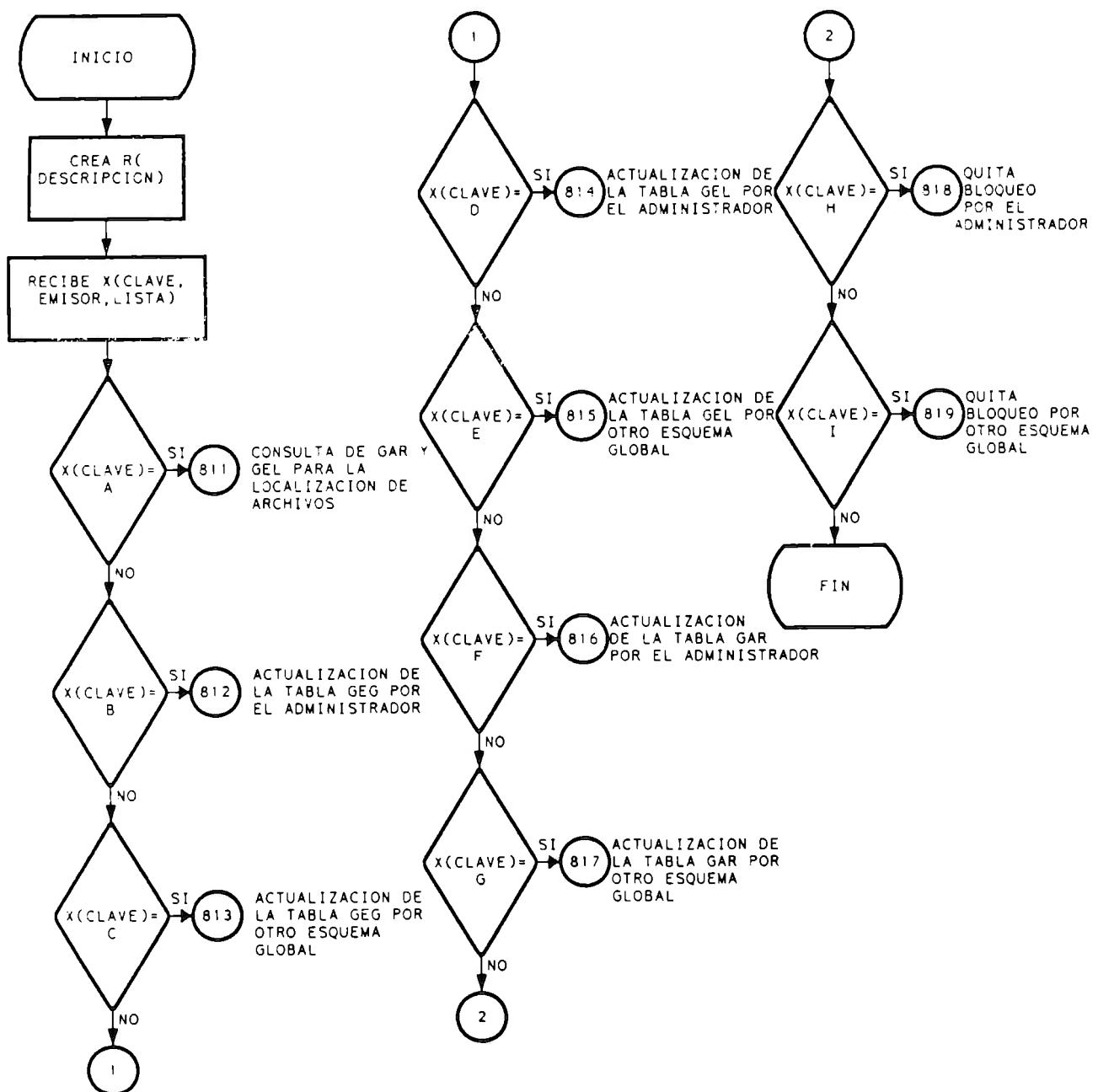
- a) Módulo controlador de la base de datos y de comunicación.
- b) Módulo de diccionario de datos.
- c) Módulo de recuperación.
- y los siguientes módulos de administración.
- d) Módulo de edición de esquemas globales alternos.
- e) Módulo de actualización.

### **4.2.1. Módulo Controlador De Base De Datos Y De Comunicación.**

Este módulo es el encargado de hacer las consultas dentro de la base de datos del Esquema Global, el cual se integra por las tablas **GEG** (Global Esquema Global), **GEL** (Global Esquema Local) y **GAR** (Global Archivo), que son las que contienen las direcciones de todos los esquemas y archivos en la red, y envía el resultado de dichas consultas al Esquema o Administrador que lo solicite. El

funcionamiento de este módulo es el siguiente: al recibir el **mensaje** toma el primer carácter para saber el tipo de mensaje que es y, dependiendo del tipo de mensaje, va a hacer la consulta o actualización en los archivos. En el caso de ser el administrador el que haga modificaciones, mandará éstas a los demás módulos.

DIAGRAMA 1: Módulo controlador de base de datos y comunicación.



Cabe aclarar que al principio del mensaje deberá haber un carácter que dirá qué tipo de mensaje se está recibiendo.

El formato del mensaje aparece en la tabla 10.

TABLA 12: FORMATO DE MENSAJE		
CARACTER INICIAL	CARACTER FINAL	DESCRIPCIÓN
0	0	Clave de tipo de mensaje
1	8	Clave del administrador, Esquema Local o Global
9	EOF	Lista de archivos, Esquema Local o Global

Donde las variables van a tener el siguiente significado:

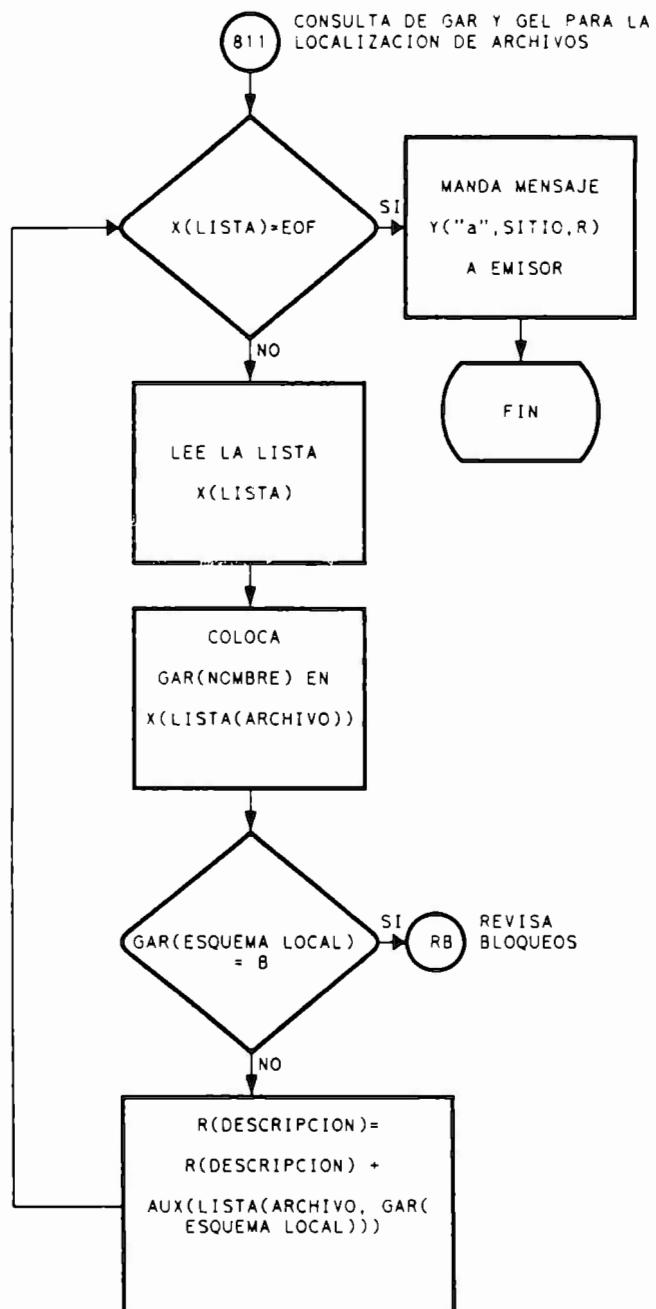
- a) Clave de tipo de mensaje: este carácter indica el tipo de mensaje que se está enviando y servirá para el control de las estrategias a seguir por parte del administrador. Este mensaje va a estar definido por una letra, que significará un procedimiento a seguir, dependiendo del esquema que reciba el mensaje. Estos procedimientos se describirán a detalle después de la explicación de las variables.
- b) Clave del administrador, Esquema Local o Global: este campo va a indicar quién es el que está mandando el mensaje, para saber quién está administrando la consulta.
- c) Lista de Archivos, Esquema Local o Global: este campo es una relación en la cual se tiene el archivo, Esquema Local o Esquema Global en el cual se desea trabajar.

El módulo controlador de base de datos y de comunicación es el encargado de las funciones que serán descritas a detalle a continuación.

#### 4.2.1.1. Consulta de las tablas GAR y GEL para localización de archivos.

El esquema global recibe una lista de los archivos de los cuales desea conocer su localización y la revisa con la tabla **GAR**. Se toman los datos de la tabla para saber cuál o cuáles Esquemas Locales contienen los archivos. De la tabla **GEL** toma los datos de direcciones y estos se mandan como respuesta al Esquema Local que los requiere.

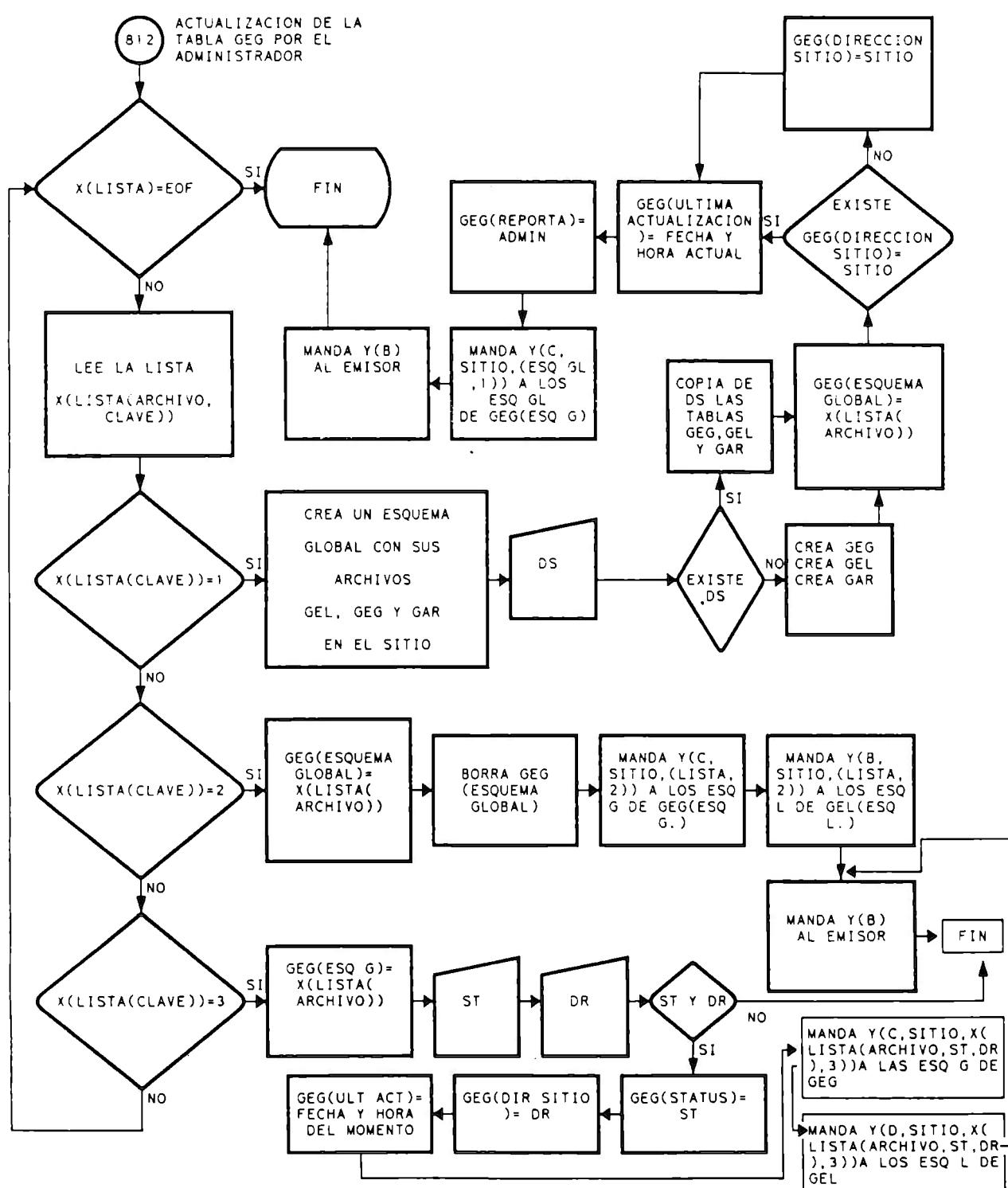
DIAGRAMA 2: 811 Consulta de GAR y GEL para la localización de archivos.



#### 4.2.1.2. Actualización de la tabla GEG por parte del administrador.

Si el administrador agrega, modifica o elimina un Esquema Global, se manda la actualización a la tabla **GEG** y a todos los Esquema Global y Esquema Local de la red.

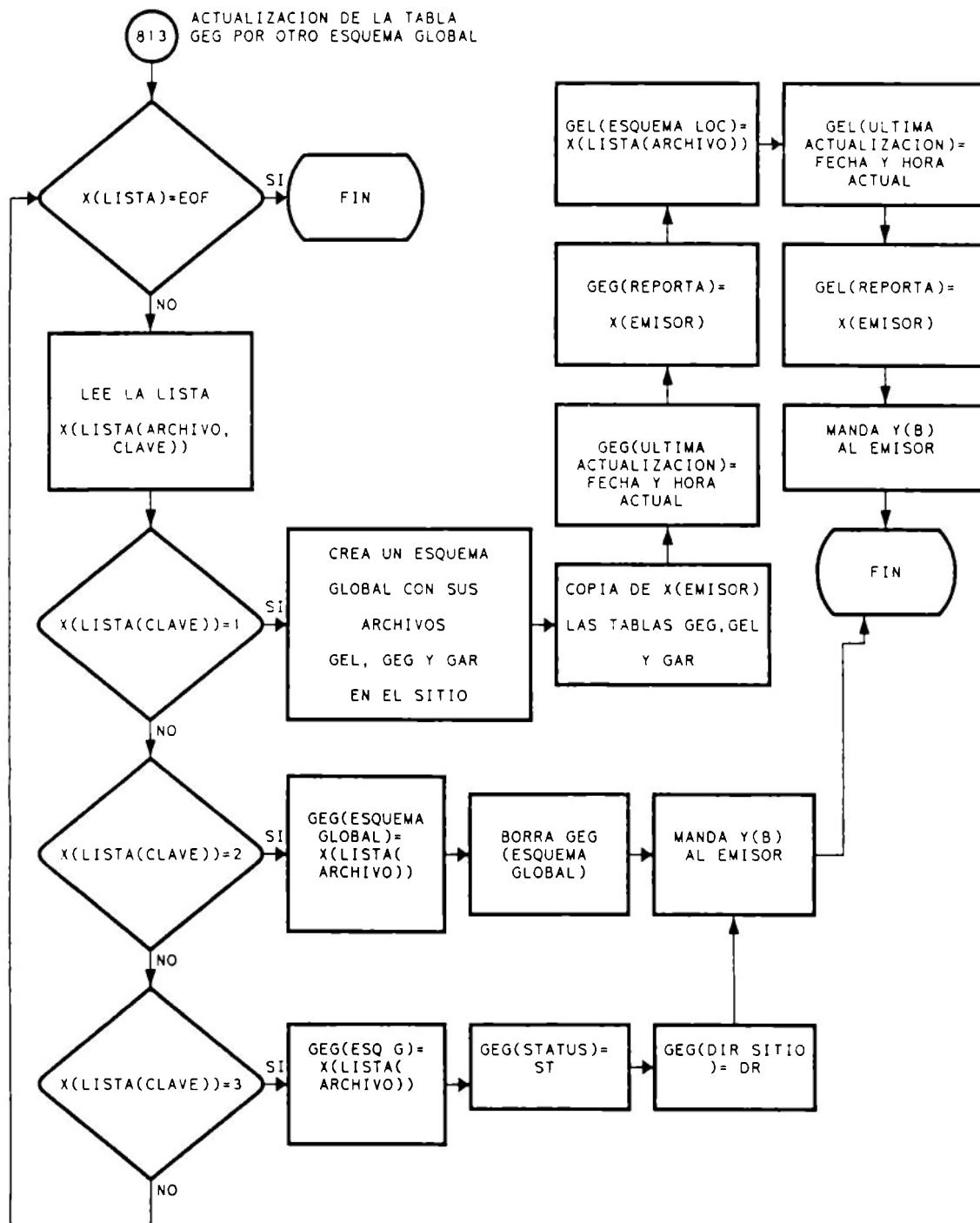
DIAGRAMA 3: 812 Cactualización de la tabla GEG por el administrador



#### 4.2.1.3. Actualización de la tabla GEG por parte de otro Esquema Global.

Para agregar, modificar o eliminar un Esquema Global, la actualización se manda siempre a la tabla GEG.

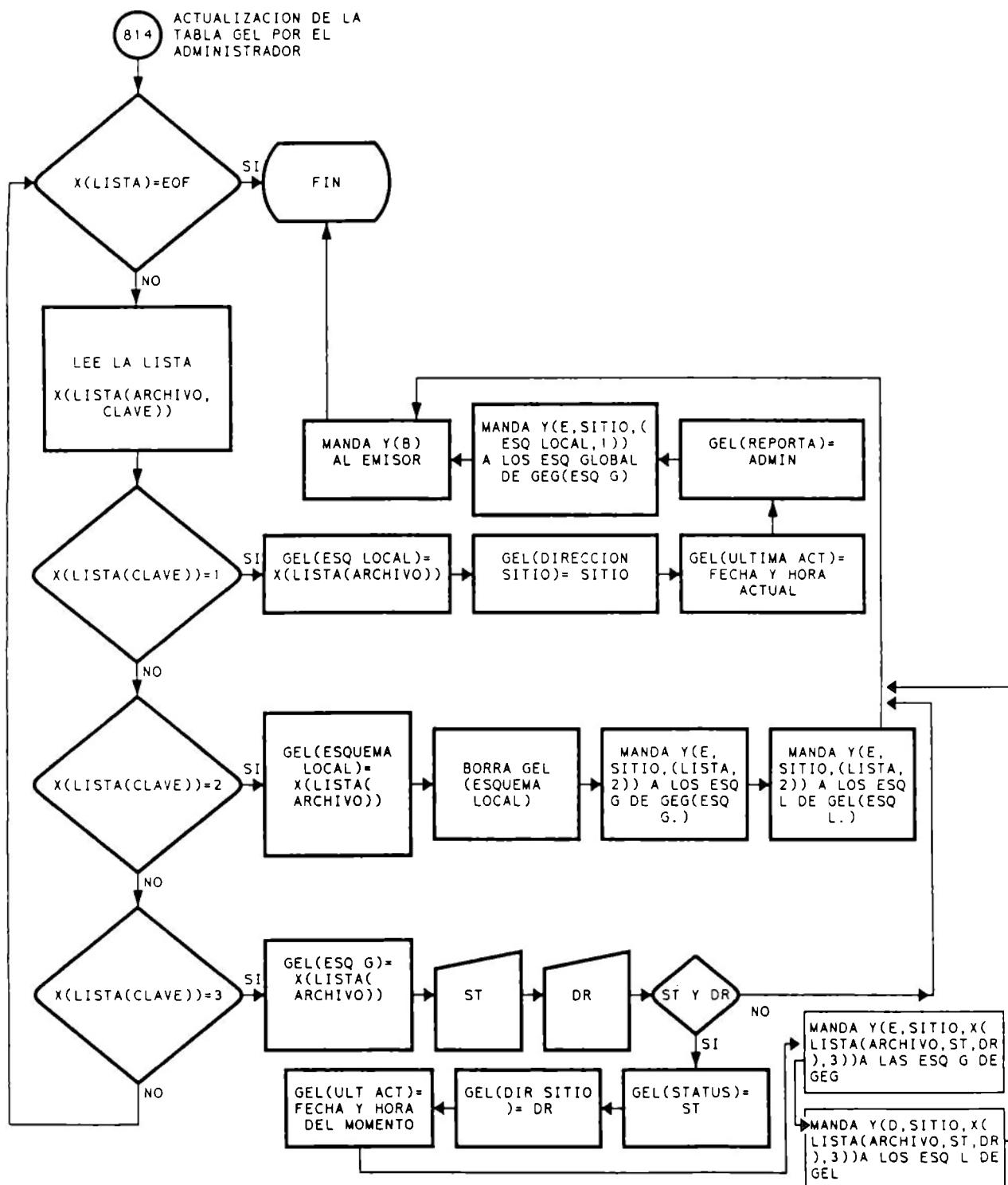
DIAGRAMA 4: 813 Actualización de la tabla GEG por otro esquema Global



#### 4.2.1.4. Actualización de la tabla GEL por parte del administrador.

Si el administrador agrega, modifica o elimina un Esquema Local, la actualización se manda a la tabla GEL, a los Esquemas Locales que utilicen el Esquema Local y a todos los Esquema Global de la red.

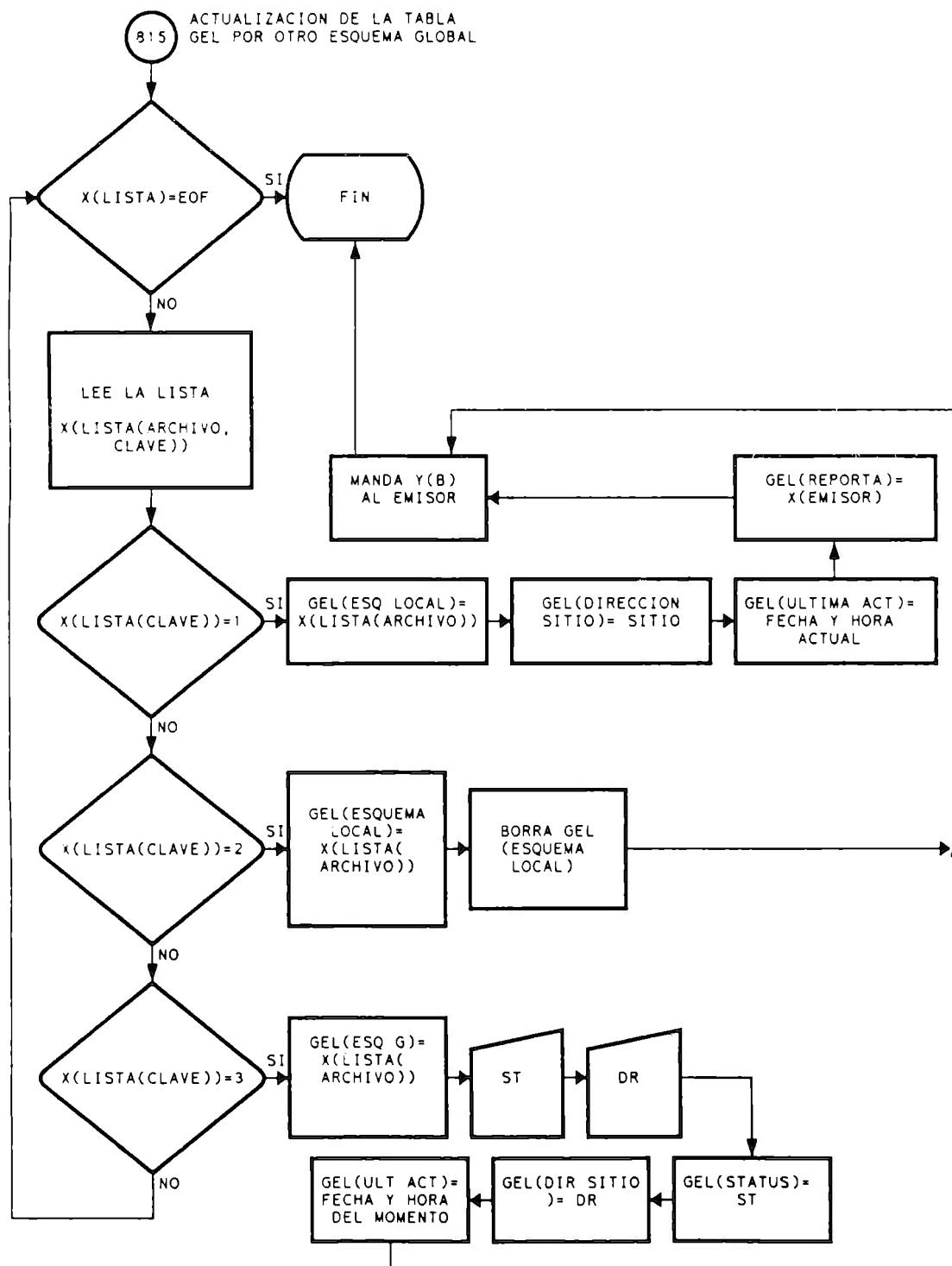
DIAGRAMA 5: 814 Actualización de la tabla GEL por el administrador



#### 4.2.1.5. Actualización de la tabla GEL por parte de otro Esquema Global.

Para agregar, modificar o eliminar un Esquema Local, la actualización se manda siempre a la tabla GEL.

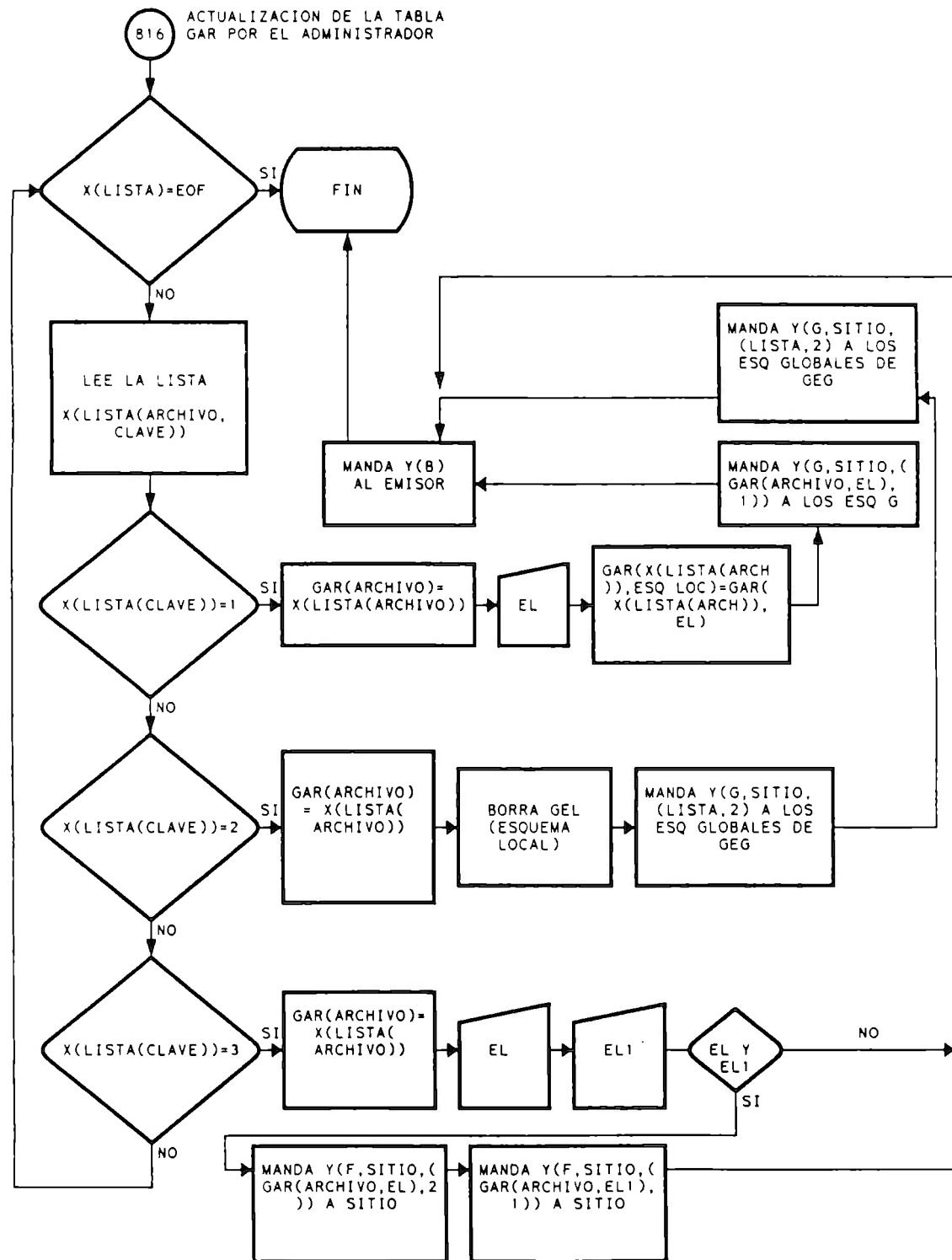
DIAGRAMA 6: 815 Actualización de la tabla GEL por otro esquema global



#### 4.2.1.6. Actualización de la tabla GAR por parte del administrador.

Si el administrador agrega, modifica o elimina un archivo para la distribución, esta modificación va a la tabla **GAR** y se manda la corrección a los Esquema Local que usen el archivo y a todos los Esquema Global de la red.

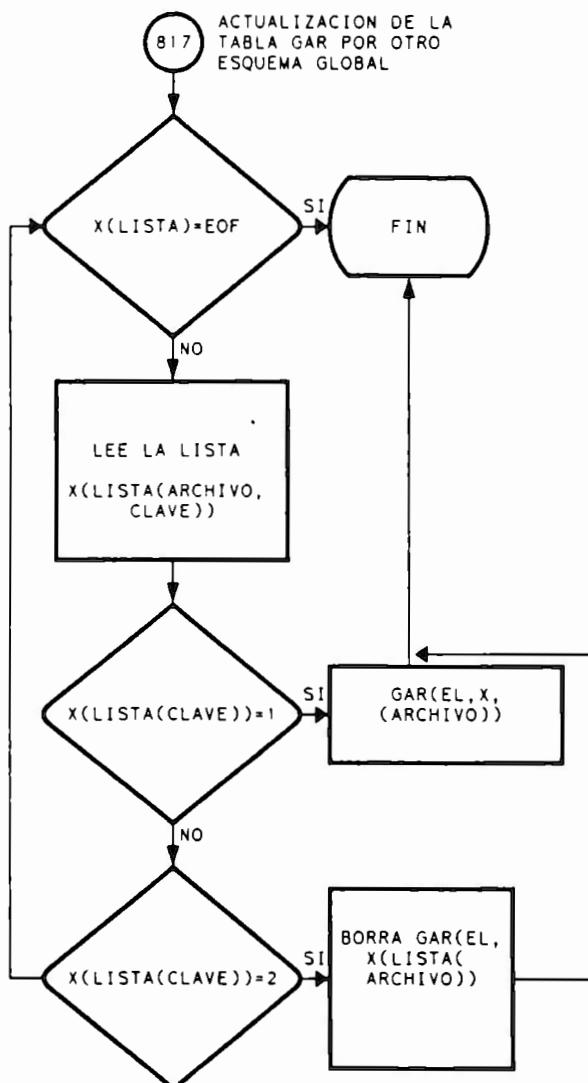
DIAGRAMA 7: 816 Actualización de la tabla GAR por el administrador



#### 4.2.1.7. Actualización de GAR por parte de otro Esquema Global.

La modificación va a la tabla GAR.

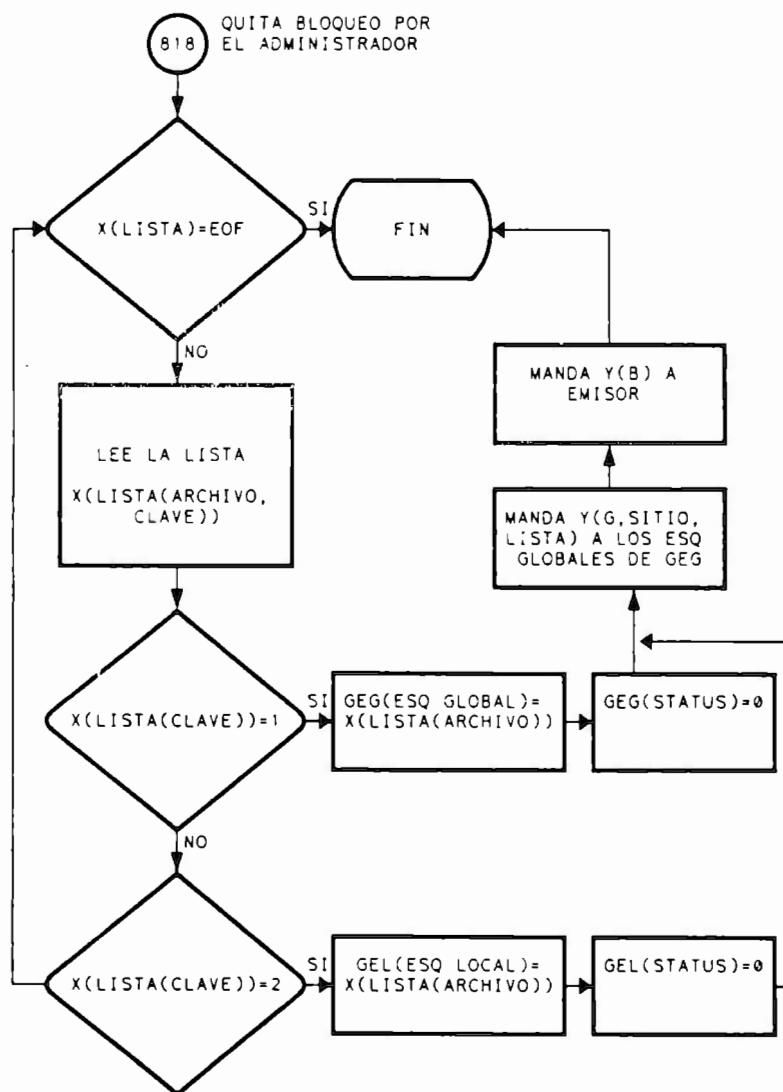
DIAGRAMA 8: 817 Actualización de la tabla GAR por otro esquema global



#### 4.2.1.8. Quita bloqueo por parte del Administrador.

Al quitar el bloqueo por parte del administrador se modifica la variable Status de la tabla GEG o GEL del Esquema Global y se manda la modificación a todos los Esquema Global de la tabla GEG y a los Esquema Local de la tabla GEL.

DIAGRAMA 9: 818 quita bloqueo por el administrador

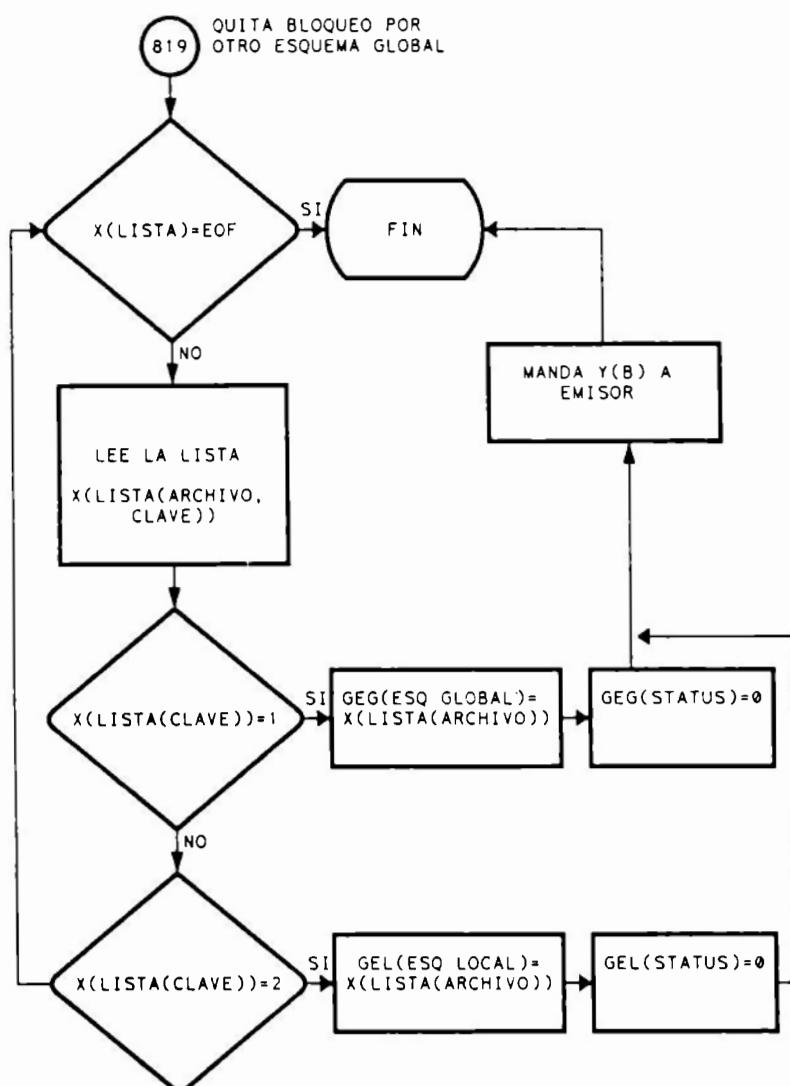


#### 4.2.1.9. Quita el bloqueo por parte de otro Esquema Global.

Al quitar el bloqueo por parte de otro Esquema Global, se modifica la variable Status de la tabla GEG o GEL del Esquema Global.

El algoritmo de éste módulo es el Algoritmo 1:Módulo controlador de base de datos y comunicación, del anexo 2.

DIAGRAMA 10: 819 Quita bloqueo por otro esquema global



#### 4.2.2. Módulo De Diccionario De Datos.

Este módulo contiene las características que tienen los archivos dentro del sistema, como son: *dirección lógica*, que es el lugar donde están ubicados dentro de la red; *rélicas*, que son las copias que tiene un archivo dentro de la red; *fragmentos*, que son los lugares donde están las partes de un archivo, así como qué parte es la que representa cada uno.

El diccionario de datos se divide en un **registro** y tres archivos que son GEG, GEL y GAR.

El registro propuesto contiene los datos generales del Esquema Global, como son el nombre y la dirección.

Los archivos tienen las siguientes características:

a) **GEG** (Global Esquema Global), es una tabla donde se almacenan las direcciones de los Esquemas Globales que se encuentran dentro del sistema. Gráficamente la representamos mediante la siguiente tabla:

TABLA 13: ESQUEMA GLOBAL (GEG)				
Esquema Global	Dirección Sitio	Status	Última actualización	Reporta
llave 1				

Se utiliza el nombre *Esquema Global* tanto para denominar al archivo como a su primera variable, con objeto de tener un mayor control; el nombre que se utilice en este registro puede ser el mismo del servidor que lo controla y es la llave para acceder a los Esquemas Globales. El formato para este campo es alfanumérico de 8 caracteres.

*Dirección Sitio*: este campo es para la dirección dentro de la red que va a tener el Esquema Global y su formato es alfanumérico de 45 caracteres.

El campo *Status* indica el estado en el que se encuentra el Esquema Global. Por medio de la clave "A", indica que está activo y "B" que está bloqueado o hay una caída de esta localidad. El formato de este campo será alfabetico de un caracter.

*Última Actualización* es un campo que indica cuándo fue la última vez que se modificó; esto sirve para saber si el campo Status está en "B", en cuyo caso se revisa la bitácora de Esquema Global, por si hay que hacerle modificaciones al Esquema Global, para poder trabajar con él y avisar al resto de los Esquemas Globales cuando haya un cambio de status a "A". El formato que va a tener es AA:MM:DD:HH:mm:SS; que significa: AA para los dos últimos dígitos del año, MM para el mes, DD para el día, HH para la hora, mm para los minutos y por último SS para los segundos.

*Reporta:* este último campo nos indica qué Esquema Global es el que reporta bloqueado, que será el mismo que quite el bloqueo. Tendrá un formato alfabético de 8 caracteres.

b) **GEL (Global Esquema Local)**, es la tabla donde se almacenan las direcciones de los Esquemas Locales que se encuentran dentro del sistema.

TABLA 14: ESQUEMA LOCAL (GEL)				
Esquema Local	Dirección sitio	Status	Última Actualización	Reporta
llave 1				

Se utiliza el nombre *Esquema Local* tanto para denominar al archivo como a su primera variable, con objeto de tener un mayor control; el nombre que se utilice en este registro puede ser el mismo del servidor que lo controla y es la llave para acceder a los Esquemas Locales. El formato para este campo es alfanumérico de 8 caracteres.

El nombre que se utilice para identificar el Esquema Local puede ser el mismo que se utiliza para hacerlo con el Esquema Global.

*Dirección Sitio:* en este campo está la dirección dentro de la red que va a tener el Esquema Local, es la primera parte de la llave de este archivo para controlar los Esquemas Locales y su formato será alfanumérico de 45 caracteres.

Las definiciones de *Status*, *Última Actualización* y *Reporta* ya fueron enunciadas en el punto anterior.

c) **GAR (Global Archivo)**, es la tabla en donde se almacena todos los archivos que están en la red y qué Esquema Local los administra.

TABLA 15: ARCHIVOS (GAR)		
Nombre	Esquema Local	Fragmento
llave 1	llave 2	

En el campo *Nombre* va el nombre del archivo en disco, el formato para este campo es alfabético de 8 caracteres y es la primera parte de la llave del archivo.

*Esquema Local* es el campo con el nombre lógico del Esquema Local con que se relaciona, es la segunda parte de la llave de este archivo. Su formato es alfanumérico de 8 caracteres.

*Fragmento:* en este campo tenemos la fórmula que va a diferenciar qué parte del archivo es el que se encuentra contenido en esa fracción; si no existe fórmula, significa que contiene el archivo completo.

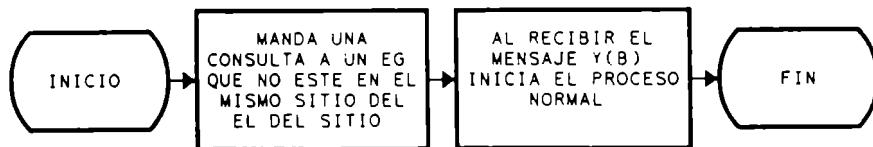
#### 4.2.3. Módulo De Recuperación.

Es el método que va a seguir el sistema para la recuperación de un nodo o varios nodos en caso de caída.

Al iniciar, el sistema manda un mensaje de consulta de algún archivo que esté en el Sitio a un Esquema Global que no se encuentre en el mismo Sitio, en el caso de estar bloqueado, el controlador de base de datos hará que se ejecute el programa RB, el cual quitará el bloqueo y actualizará los archivos GAR, GEL, GEG, LAR (Local Archivo), LEL (Local Esquema Local) y LEG (Local Esquema Global), y al recibir un mensaje de respuesta, inicia el proceso normal.

El algoritmo de éste módulo es el Algoritmo 2:Módulo de recuperación, del anexo 2.

Diagrama 11: Módulo de recuperación



#### 4.2.4. Módulo De Edición De Esquemas Globales Alternos.

Este módulo permite crear Esquemas Globales en otros sitios diferentes al actual, haciendo una copia de los distintos módulos y agregando a la base de datos el módulo que lo crea y la dirección de éste, para que en el caso de cualquier actualización el sitio nuevo pueda modificar el Esquema Global que lo creó; también, en el caso contrario, se puede dar de baja un Esquema Global del sistema, mandando a los demás esquemas globales y a los esquemas locales un aviso para borrarlo de la base de datos de Esquemas Globales.

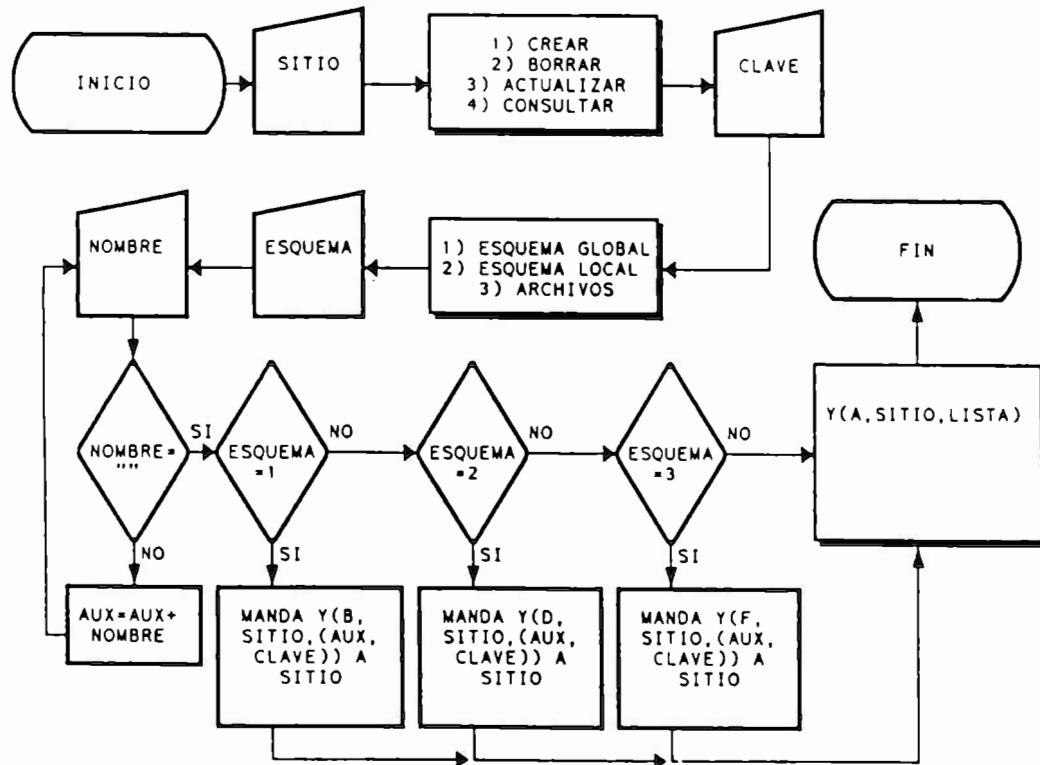
Los algoritmos con los que trabaja este módulo son:

- a) Un algoritmo de edición
- b) Un algoritmo de verificación

El Algoritmo de Edición, se encarga de formar y enviar el mensaje del administrador a los Esquemas Globales o Locales o Archivos del sistema.

El algoritmo de éste módulo es el Algoritmo 3:Edición de esquemas globales alternos del anexo 2.

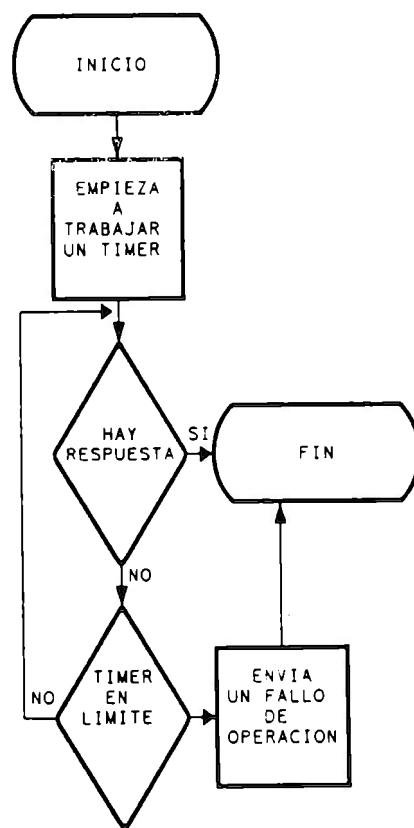
DIAGRAMA 12: Edición de esquemas globales alternos



El Algoritmo de Verificación de Mensajes (VM), trabaja con un protocolo a dos fases, con el fin de revisar que los mensajes estén completos; va a funcionar en el Esquema Global y en el Esquema Local.

El algoritmo VM es el Algoritmo 4: Verificación de mensajes del anexo 2.

DIAGRAMA 13: Verificación de mensajes

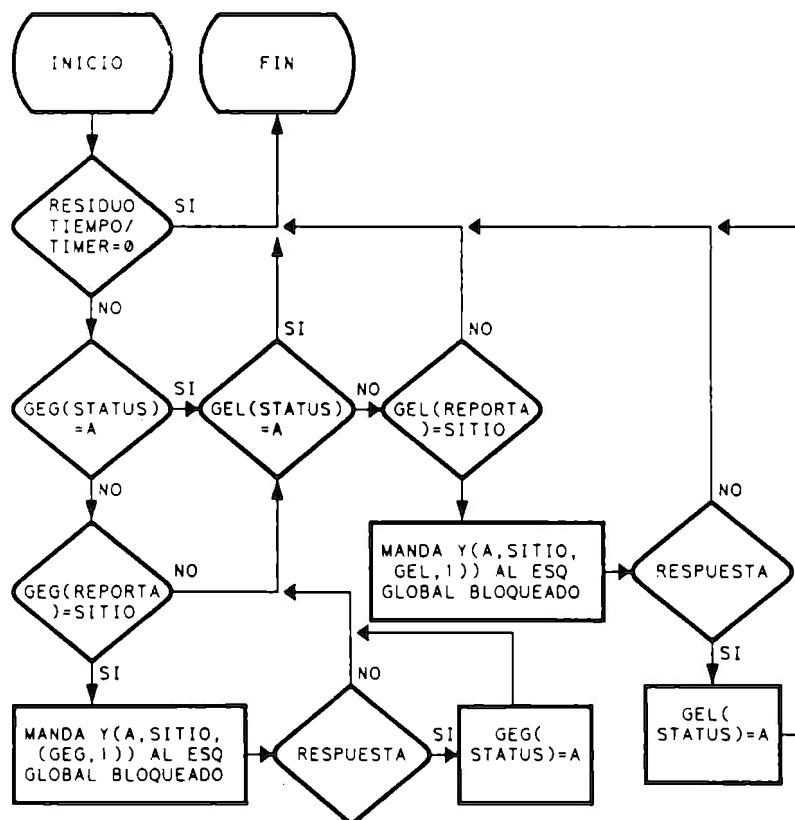


#### **4.2.5. Módulo De Actualización.**

Este módulo está encargado de revisar periódicamente la tabla de bloqueos, para ver si algún Esquema Global o Esquema Local alterno está bloqueado y, en ese caso, mandarle una copia de las actualizaciones que se hayan hecho después del momento en el que fue bloqueado. La periodicidad de este procedimiento depende de los criterios del administrador, el cual define una variable en el sistema, llamada **Timer**, que ejecutará el procedimiento de acuerdo a ésta.

El algoritmo RB es el Algoritmo 5: Revisa Bolqueos del anexo 2.

#### DIAGRAMA 14: Revisa bloqueos



### **4.3. ESQUEMA LOCAL**

Es el módulo que tiene un Esquema de los archivos más utilizados por la localidad, contenido las direcciones lógicas, el status de cada archivo, así como las réplicas y o fragmentos y los Esquemas Locales que los controlan.

Está formado de los siguientes módulos:

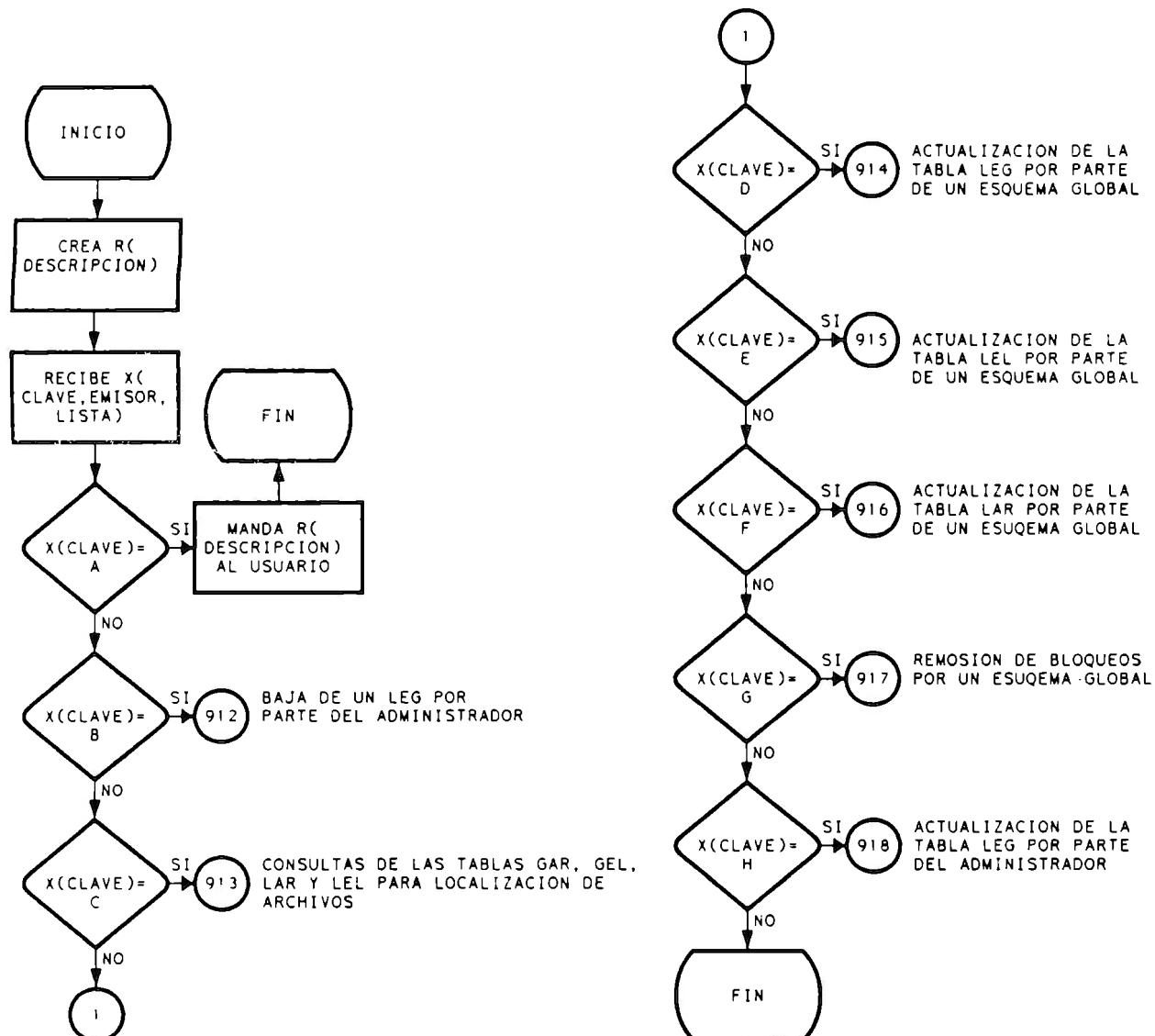
- a) módulo controlador de base de datos y de comunicación.
- b) módulo de diccionario de datos.
- c) módulo de edición del Esquema Local

#### **4.3.1. Módulo Controlador De Base De Datos Y De Comunicación.**

Al igual que en el Esquema Global, este módulo es el encargado de hacer las consultas dentro de la base de datos del Esquema Local, integrado por las tablas **LEG** (Local Esquema Global), **LEL** (Local Esquema Local) y **LAR** (Local Archivo), que son las que contienen las direcciones de todos los esquemas y archivos que han sido utilizados por el Esquema Local; en caso de no encontrarse la información en estos archivos, manda la consulta a un Esquema Global disponible, actualiza sus archivos con el resultado de la consulta y envía el resultado de las consultas al Esquema o Administrador que lo solicite.

Este módulo es el encargado de las funciones que serán descritas a detalle a continuación:

DIAGRAMA 15: Módulo controlador de base de datos y comunicación.



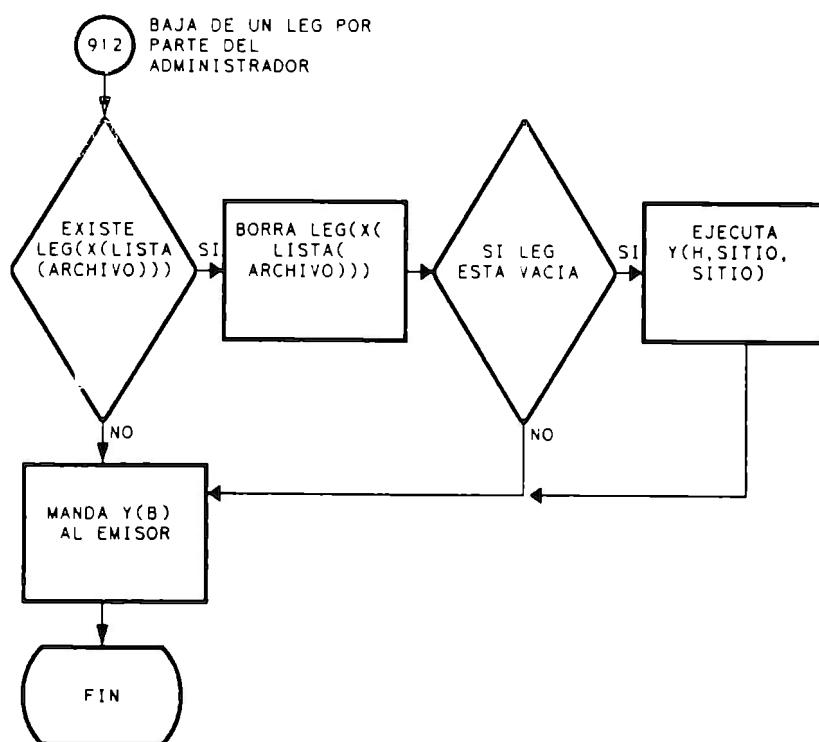
#### 4.3.1.1. La respuesta a una consulta a un Esquema Global y o Local.

Mediante una lista de los archivos y Esquemas Locales en donde se encuentran, los cuales agregará a las tablas del Esquema Local y de Archivos correspondientes.

#### 4.3.1.2. La baja de un LEG por parte del Administrador.

Solicitando, en el caso de ser el único esquema en la tabla LEG, un Esquema Global Substituto.

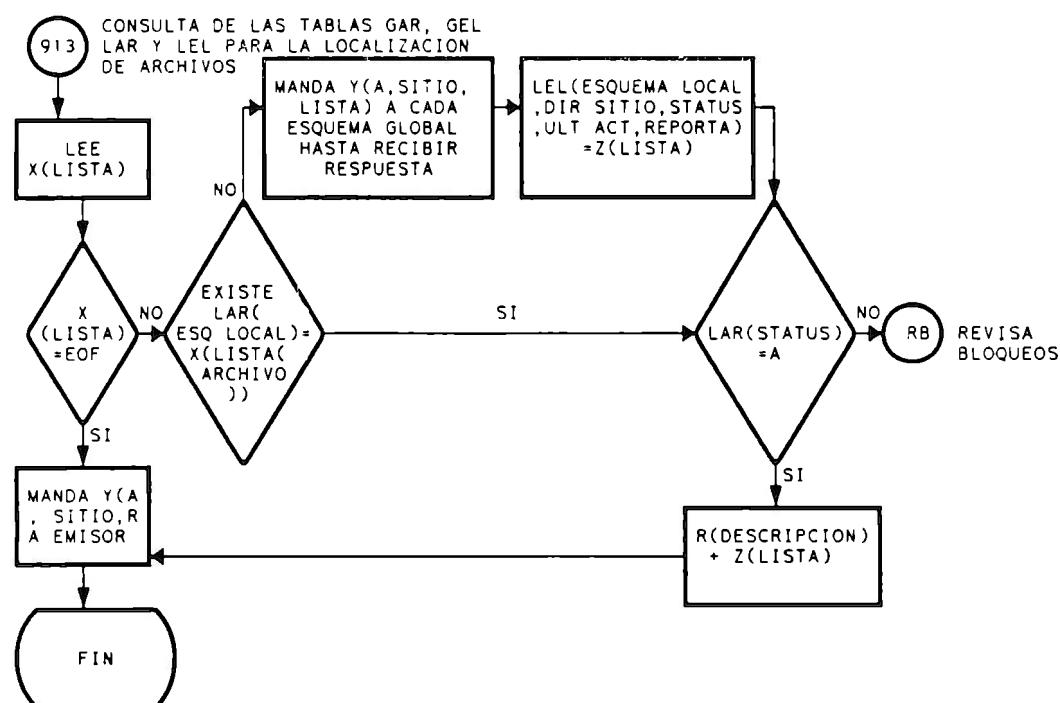
DIAGRAMA 16: 912 Baja de un LEG por parte del administrador



#### 4.3.1.3. La consulta de las tablas GAR, GEL, LAR y LEL para localización de archivos.

Mediante la recepción de una lista con los archivos que se desea localizar. Esta lista se revisará con la tabla de archivos, LAR, y en caso de no encontrarse, hace la consulta a un Esquema Global con el resultado (archivos no localizados); posteriormente se toman los datos de la tabla Archivos para saber cuál o cuáles Esquemas Locales contienen los archivos. De la tabla Esquema Local se toman los datos de las direcciones.

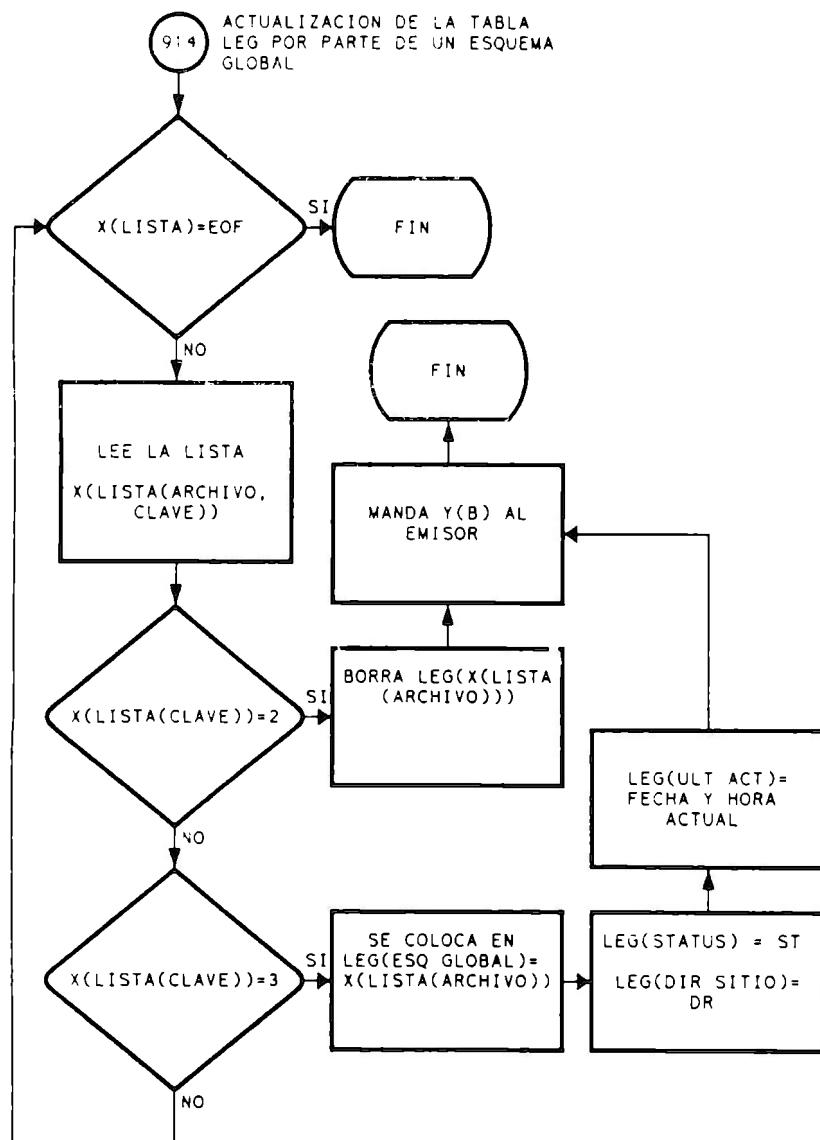
DIAGRAMA 17: 913 Consulta de las tablas GAR, GEL, LAR y LEL para la localización de archivos



#### 4.3.1.4. La actualización de la tabla LEG por parte de un Esquema Global.

Ya sea para modificar, agregar o eliminar un Esquema Global; la actualización se manda a la tabla LEG.

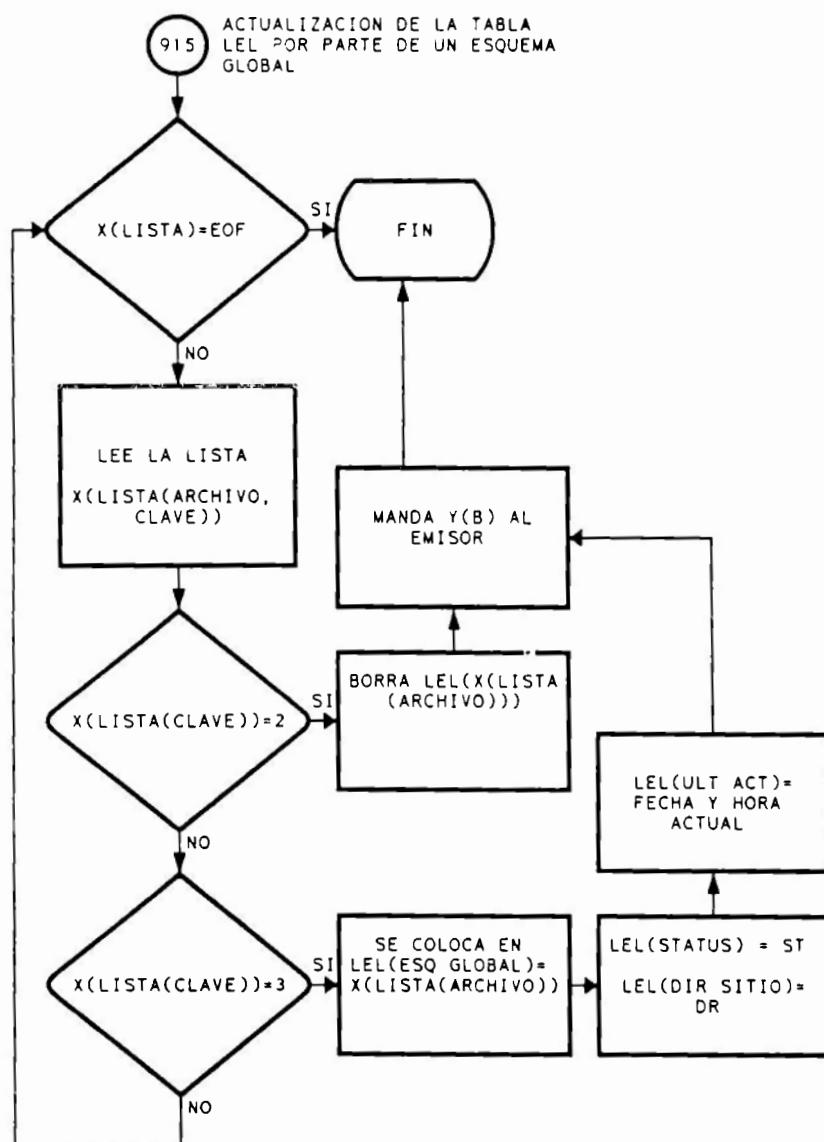
DIAGRAMA 18: Actualización de la tabla LEG por parte de un esquema global



#### 4.3.1.5. La actualización de la tabla LEL por parte de un Esquema Global.

Ya sea para modificar, agregar, o eliminar un Esquema Local; la actualización se manda a la tabla LEL.

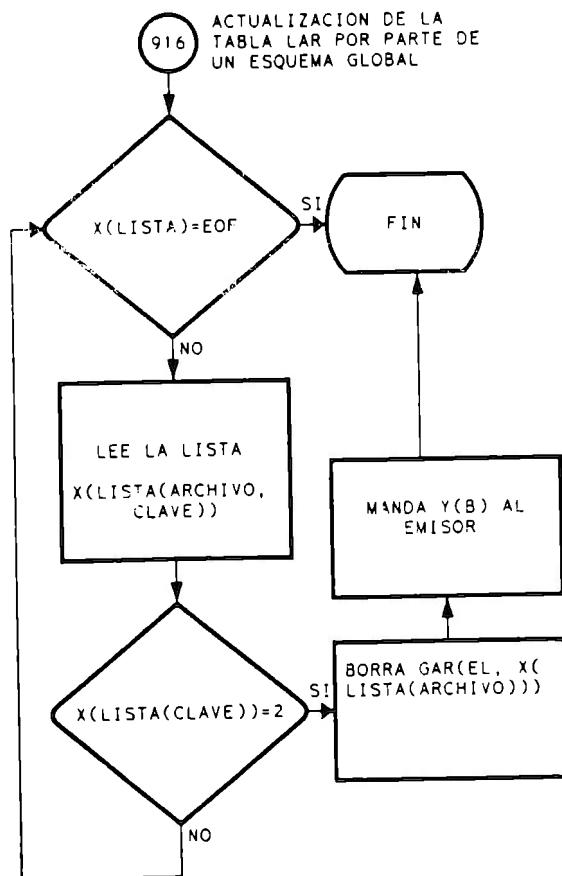
DIAGRAMA 19: 915 Actualización de la tabla LEL por parte de un esquema global



#### 4.3.1.6. La actualización de la tabla LAR por parte de un Esquema Global.

Después de que el administrador hace una corrección en algún Esquema Global, el administrador manda la actualización de ese Esquema Global a los Esquemas Locales, para tener los datos completos en todos.

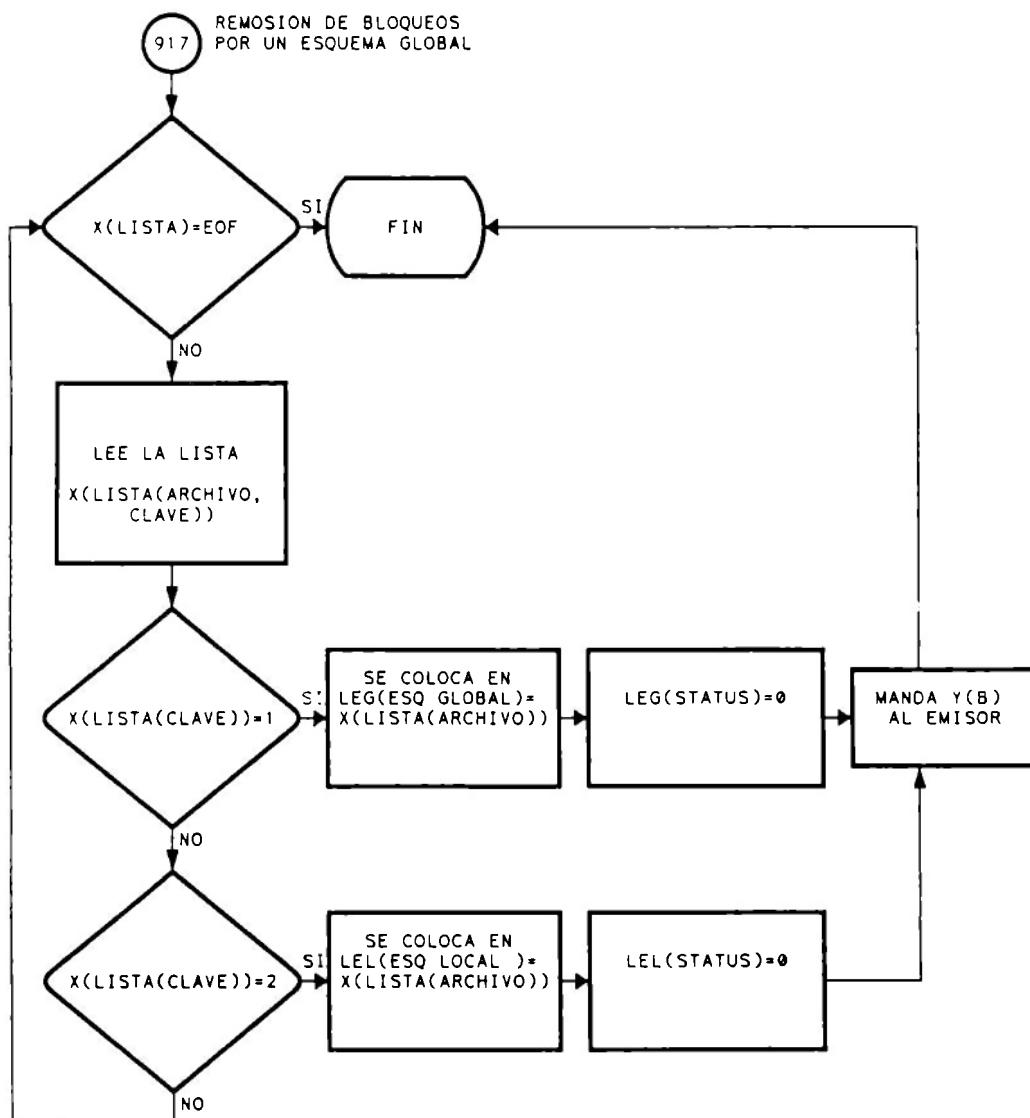
DIAGRAMA 20: 916 Actualización de la tabla LAR por parte de un esquema global



#### 4.3.1.7. La remoción de bloqueos por parte de un Esquema Global.

Busca en las tablas LEG y LEL los Esquemas Globales y Locales a los cuales se les quita el bloqueo y modifica la variable Status por A como se explica en la tabla GEG del inciso 4.2.2.

DIAGRAMA 21: 917 Remoción de bloqueos por un esquema global

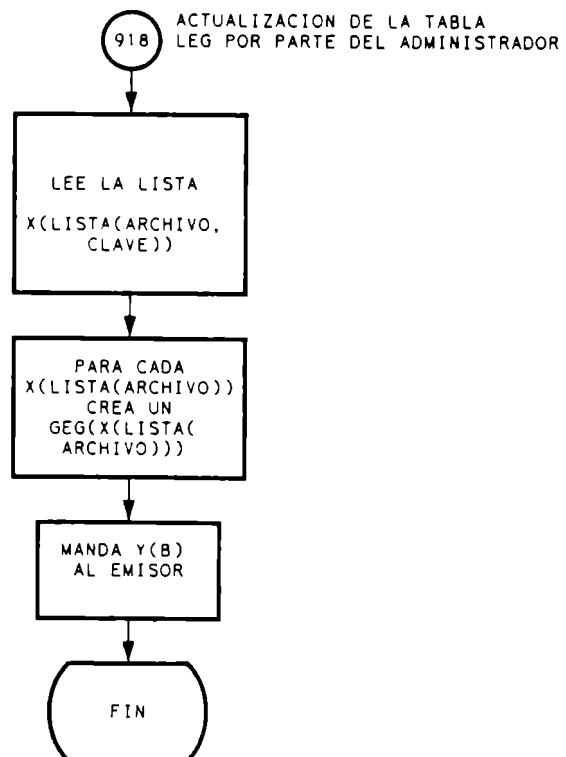


#### 4.3.1.8. La actualización de LEG por parte del Administrador.

Dando de alta un Esquema Global en la tabla LEG, para que en el caso que no pueda hacer una consulta, el Esquema Local pueda buscar en varios Esquemas Globales.

El algoritmo 6 utilizado en este módulo: Módulo controlador de bases de datos y comunicación se encuentra en el anexo 2.

DIAGRAMA 22: 918 Actualización de la tabla LEG por parte del administrador



#### **4.3.2. Módulo De Diccionario De Datos.**

Este módulo va a contener información de las características que van a tener los archivos que han sido utilizados por el Esquema Local dentro del sistema, como son: a) dirección lógica, que es el lugar en donde están ubicados dentro de la red; b) réplicas, que son las copias que tiene un archivo dentro de la red; c) fragmentos, que son los lugares donde están las partes de un archivo, así como la parte que representa cada uno. Este módulo va a estar dividido en un registro con los datos generales del Esquema Local, (como son el nombre y dirección) y tres archivos que son:

- a) **LEG (Local Esquema Global)**, que contiene la dirección de los Esquemas Globales utilizados por el Esquema Local.

<b>TABLA 16: ESQUEMA GLOBAL (LEG)</b>				
<b>Esquema Global</b>	<b>Dirección Sitio</b>	<b>Status</b>	<b>Última actualización</b>	<b>Reporta</b>
llave 1				

Los conceptos de Esquema Global, Dirección Sitio, Status, Última Actualización y Reporta quedaron definidos en la tabla **GEG**, ver inciso 4.2.2.

- b) **LEL (Local Esquema Local)**, que contiene la dirección de los Esquemas Locales utilizados por el Esquema Local.

<b>TABLA 17: ESQUEMA LOCAL (LEL)</b>				
<b>Esquema Local</b>	<b>Dirección sitio</b>	<b>Status</b>	<b>Última Actualización</b>	<b>Reporta</b>
llave 1				

Los conceptos de Esquema Local, Dirección Sitio, Status, Última Actualización y Reporta quedaron definidos en la tabla **GEL**, ver inciso 4.2.2.

- c) **LAR (Local Archivo)**, que contiene la dirección de los Archivos utilizados por el Esquema Local.

TABLA 18: ARCHIVOS (LAR)

Nombre	Esquema Local	Fragmento
llave 1	llave 2	

Los conceptos Nombre, Esquema Local y Fragmento quedaron definidos en la tabla GAR, ver inciso 4.2.2.

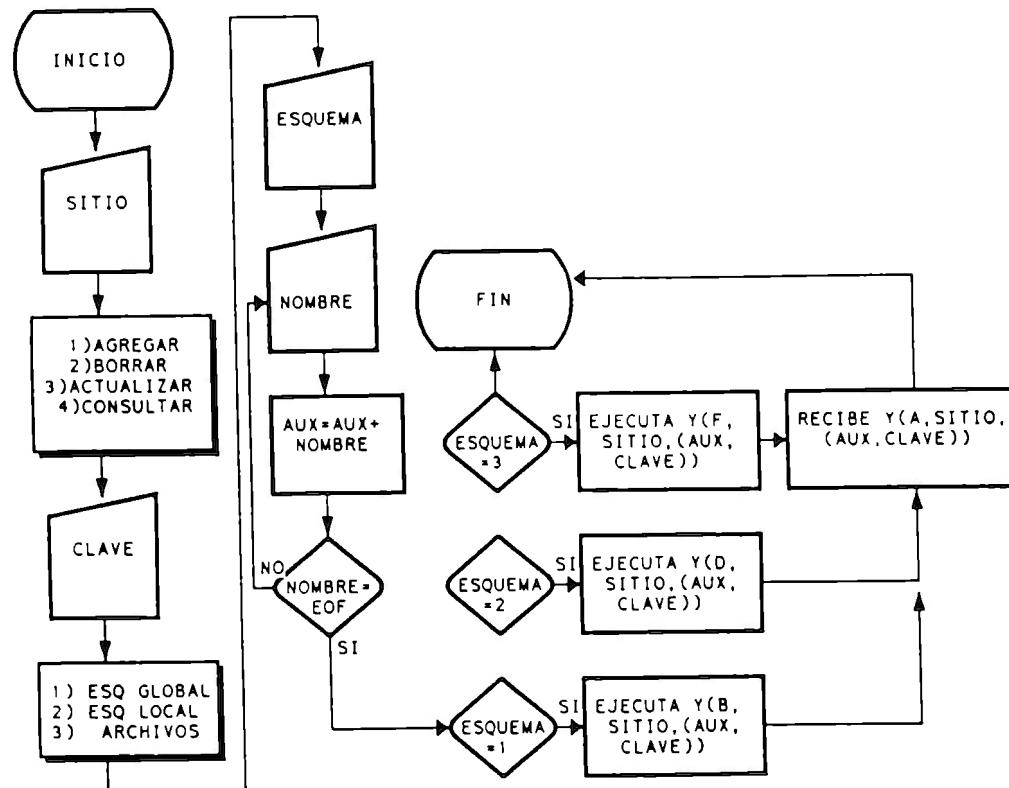
#### 4.3.3. Módulo De Edición Del Esquema Local.

Con este módulo se pueden agregar Esquemas Globales de otras localidades diferentes a la actual, permitiendo que el Esquema Local tenga otras opciones en caso de no poder hacer contacto con el Esquema Global original.

El algoritmo 7, con el que va a trabajar este módulo es Edicion del esquema local del anexo 2.

Hasta aquí se plantea la estructura del Diseño Global, mediante el desarrollo del Esquema Global y del Esquema Local, del administrador de bases de datos propuesto.

DIAGRAMA 23: Edición del esquema local



## CAPÍTULO 5: REPLICACIÓN Y FRAGMENTACIÓN

Dado que el trabajo propuesto en esta tesis está orientado a bases de datos distribuidas, nos hace considerar posibilidades de complementación aprovechando las tecnologías existentes de optimización de las unidades de almacenamiento de información, puesto que a medida que se han hecho disponibles en el mercado sistemas con unidades centrales de procesamiento más rápidas y mayor capacidad de memoria, el acceso a las unidades de almacenamiento se ha constituido en el cuello de botella en el rendimiento total del sistema, ya que la lectura de un solo bloque de información en disco puede tomar tanto tiempo como muchos miles de operaciones de la unidad central de proceso y por otra parte, las fallas en estas unidades implican no solamente una degradación del rendimiento total sino muy posiblemente la pérdida de la información deseada.

Una de las técnicas para evitar la pérdida de información es trabajar con réplicas, y para ello en este trabajo se propone como una alternativa, aplicar el algoritmo utilizado para los discos conocidos como RAID (Redundant Array of Independent Disk), en los niveles 1 y 5 que serán descritos más adelante; este algoritmo tiene la particularidad de hacer una réplica de otro disco o ser un disco de respaldo, en el nivel 1, y la recuperación y fragmentación de archivos de una localidad, para hacer una consulta más eficiente en el nivel 5. Sin embargo, aunque emplear esta tecnología añadirá prestaciones muy valiosas a una implementación específica prolongando los beneficios y versatilidad encontrados, es conveniente aclarar que la propuesta contenida en esta tesis no demanda por necesidad emplear discos RAID en su implementación.

## 5.1. CONCEPTO DE RAID.

RAID es un sistema para el almacenamiento de datos en medios magnéticos como los discos duros, que previene la pérdida de datos en eventos como la falla en ese tipo de discos. Los beneficios de RAID incluyen la seguridad, integridad, tolerancia y, en algunos sistemas, crecimiento escalable, buen desempeño y una gran facilidad para hacer cambios. RAID es necesario si los datos necesitan estar protegidos de accidentes que involucren a los discos duros y, en el caso de este trabajo, a los nodos de procesamiento. Con RAID, un disco duro o una localidad falla, el resto del sistema RAID protege los datos y los deja disponibles para mandarlos a un nuevo sistema.

Stallings (4) nos dice al respecto: "Los diseñadores de memoria de disco reconocen que si uno de los componentes sólo se puede llevar a un determinado límite de desempeño, se puede conseguir una ganancia de prestaciones adicional, usando varios de esos componentes en paralelo. En el caso de memoria en disco, esto conduce al desarrollo de conjuntos de discos que operen independientemente y en paralelo. Con varios discos, las peticiones separadas de E/S se pueden gestionar en paralelo, siempre que los datos requeridos residan en discos separados.". Como se ve, esta técnica no sólo permite mejorar la seguridad, sino también la velocidad en los procesos.

Los discos RAID funcionan mediante dos componentes fundamentales: el software que efectúa el enlace entre el sistema operativo y los dispositivos de control de las unidades de disco y los controladores en circuitos asociados a las unidades de almacenamiento. Las implementaciones RAID en equipos específicos ( y ) tienden a lograr que los controladores supervisen en tal forma que en caso de falla el empleo de las unidades redundantes es automático y transparente al servidor.

### 5.1.1. Características De RAID.

El esquema RAID consta de seis niveles universales, que van del 0 al 5. Estos niveles tienen tres características que son:

- a) es un conjunto de unidades físicas, vistas prácticamente como una.
- b) los datos se distribuyen a través de un conjunto de unidades físicas.
- c) la capacidad de los discos redundantes se usa para almacenar información de paridad, que garantice la recuperación de los datos en caso de falla.

En implementaciones específicas ( x ) pueden combinarse estos niveles para obtener denominaciones como nivel 10 o nivel 53, implicando la combinación de los niveles 1 y 0 en el primer caso y de los niveles 5 y 3 en el segundo.

### 5.1.2. Descripción De RAID.

#### 5.1.2.1. RAID 0. Estructura en tiras.

En este primer nivel no incluye **redundancia** para mejorar los tiempos de respuesta. El usuario y los datos del sistema están distribuidos a lo largo de todos los discos del conjunto, ver figura 9. Los datos se organizan en tiras (fragmentos) a través de los discos disponibles. Un conjunto de fragmentos lógicamente consecutivos que proyectan exactamente una tira en cada miembro del conjunto, se denomina franja. Si hay dos peticiones pendientes para dos fragmentos diferentes y éstos se encuentran en dos localidades distintas, las peticiones se pueden hacer en paralelo.

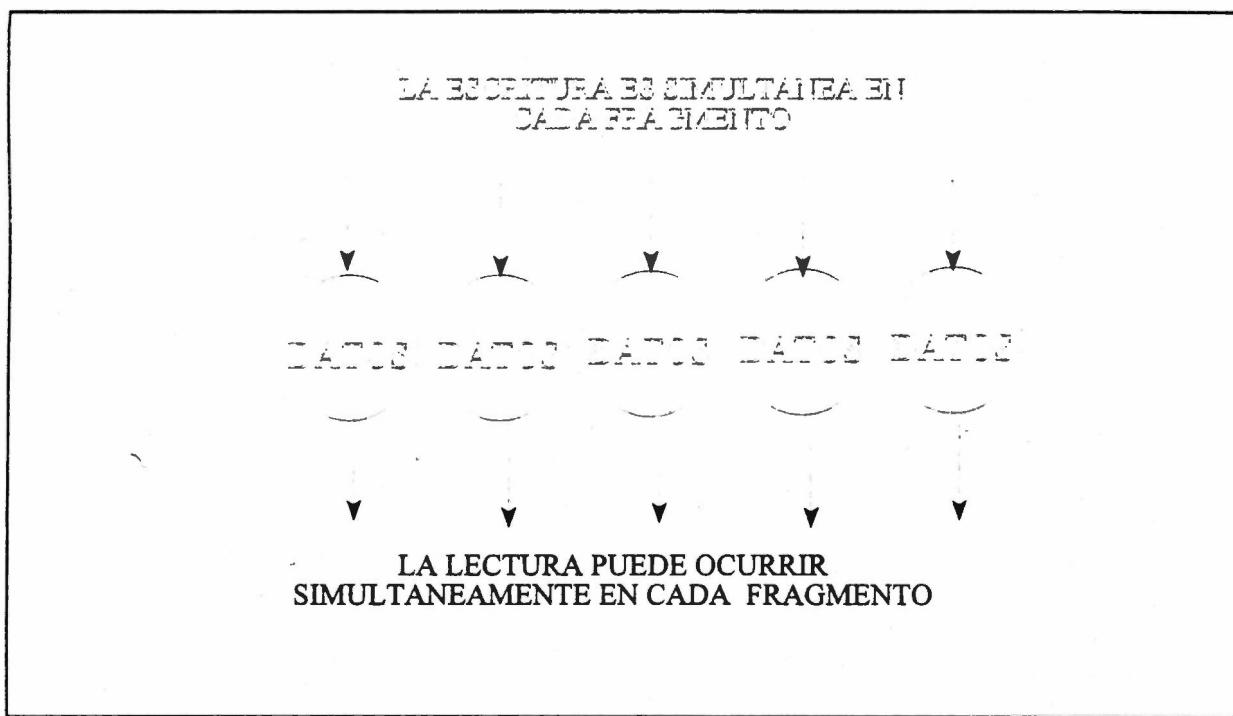


Fig 9: RAID 0

- a) Alta capacidad de transferencia de datos. Para esto se deben reunir dos requisitos. Uno, debe existir una capacidad de transferencia alta entre la memoria del anfitrión y las unidades de disco individuales. Dos, la aplicación debe hacer peticiones de E/S que controlen el conjunto de discos eficientemente. Ésto se satisface si la petición es de una gran cantidad de datos contiguos, comparados con el tamaño de la franja.
- b) Altas velocidades de petición de E/S. Un conjunto de discos puede proporcionar velocidades altas de ejecución de E/S, balanceando la carga de E/S a través de los distintos discos. RAID 0 es utilizado en aplicaciones que requieren respuesta rápida con datos no críticos, también es ideal para

los sistemas que requieren de muchas consultas, pero no sirve para crear réplicas. Puede apreciarse entonces que las mejoras se obtienen en términos de rendimiento y capacidad, pero no de confiabilidad.

### 5.1.2.2. RAID 1. Estructura en espejo.

RAID 1, consigue la redundancia duplicando todos los datos, lo que lo diferencia de los niveles 2 y 5 que utilizan procedimientos diferentes para conseguirla. Hace una distribución de datos (fragmentación), como el RAID 0, pero cada franja lógica se proyecta en dos discos físicos separados, ver figura 10.

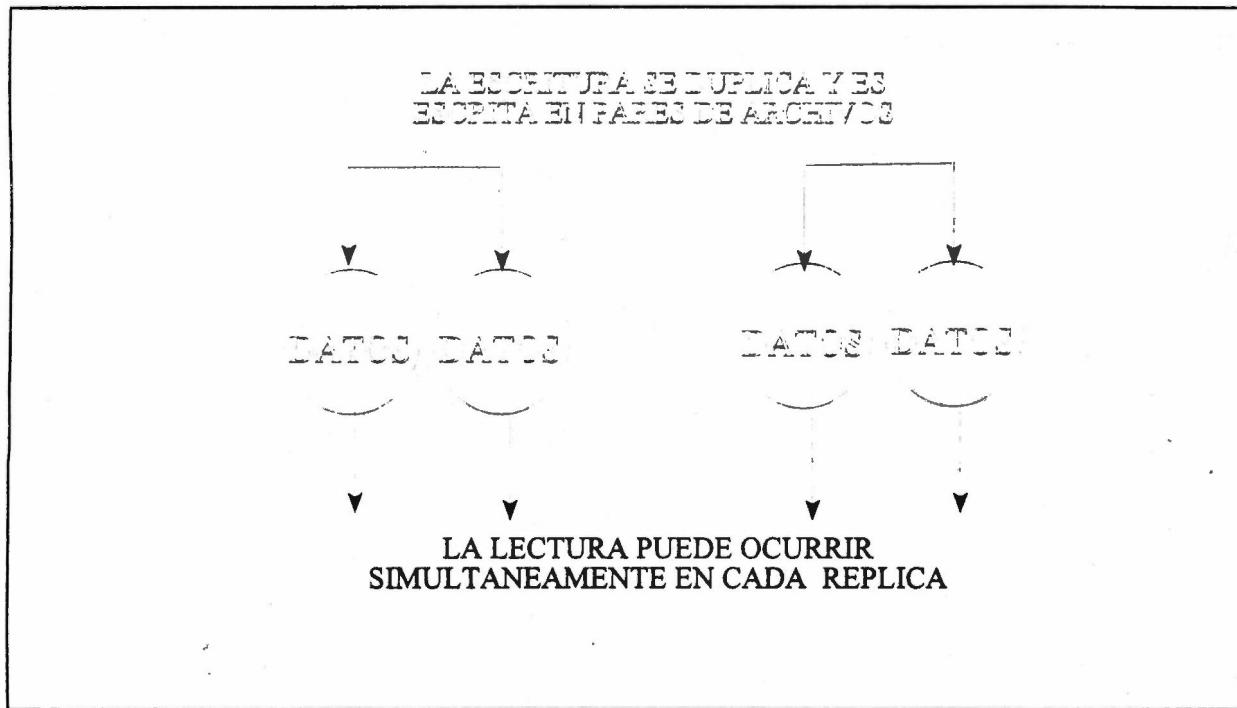


Fig 10: RAID 1

Una petición de lectura puede ser servida por cualquiera de los discos que contienen los datos pedidos, cualquiera de ellos implica un tiempo de búsqueda mínimo más la latencia rotacional. Una petición de escritura requiere que las dos tiras correspondientes se actualicen; ésto se puede hacer en paralelo y el tiempo será el mayor tiempo de búsqueda más la latencia rotacional. La recuperación tras un fallo es sencilla; cuando una unidad falla se puede acceder a los datos desde la segunda unidad. En el nivel 1 de RAID, el algoritmo crea una réplica de cada archivo en dos discos separados, pero tanto el disco primario como el secundario están en línea simultáneamente; la modificación que debe hacerse en este caso es el redireccionamiento de uno de estos dos archivos.

a una localidad diferente dentro de la red, ayudando así a resolver la replicación. El problema que tiene este nivel es que requiere de muchos discos para trabajar, pero de aquí tomamos el concepto de los discos en espejo.

Trabajar en espejo es ideal para aplicaciones de bases de datos en donde la disponibilidad y la capacidad transaccional son más importantes que la eficiencia de las unidades de almacenamiento. Adicionalmente, los espejos pueden configurarse a través de diferentes canales (buses) de datos.

#### 5.1.2.3. RAID 2. Acceso paralelo. (Redundante)

Los niveles 2 y 3 usan una técnica de acceso paralelo. Todas las unidades de disco participan simultáneamente en cada petición y todos los giros de los discos están sincronizados, ver figura 11. Como en los casos anteriores los datos se descomponen en tiras. En el caso de RAID 2 y 3 las tiras son muy pequeñas; en RAID 2 los métodos de corrección de errores se operan en cada disco. Normalmente utiliza el código Hamming, que permite corregir y detectar errores. Para cada petición todos los discos trabajan al mismo tiempo en

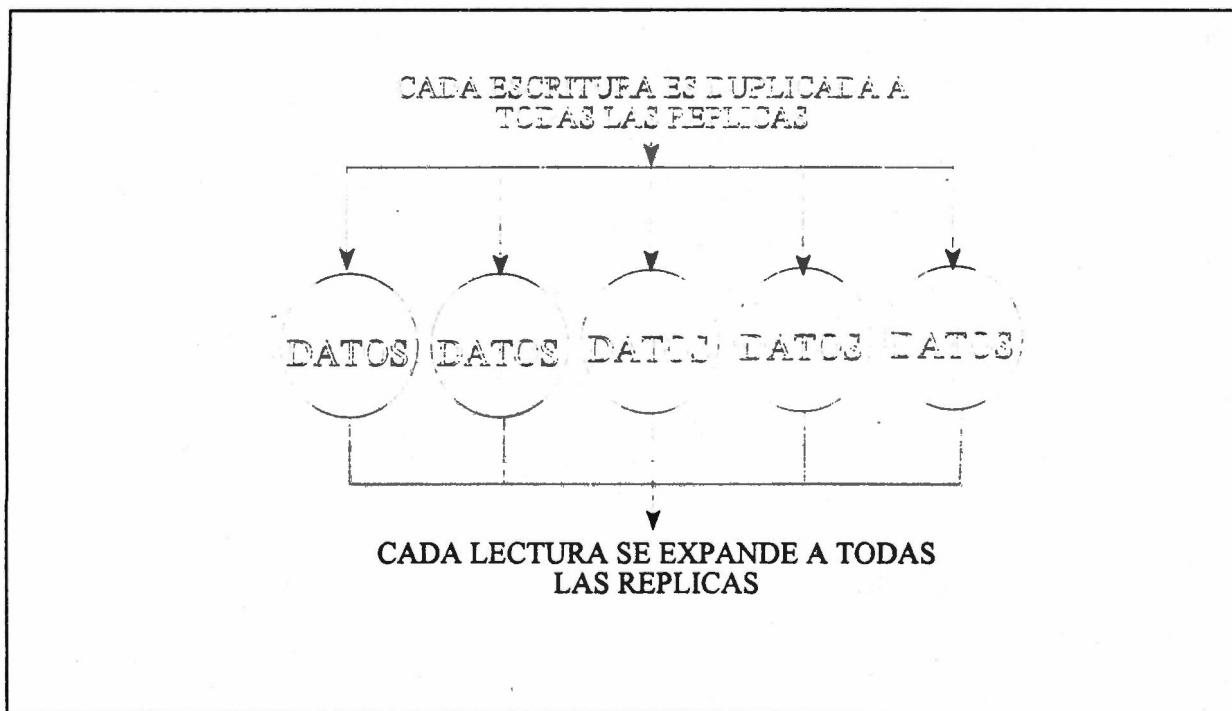


Fig 11: RAID 2

paralelo, corrigiendo los posibles errores de escritura simultáneamente. Este nivel de RAID 2 supera al nivel anterior en la corrección y detección de posibles errores, sin embargo no se recomienda su utilización, ya que trabaja con muchos discos para grabar los archivos, incrementando los costos.

#### 5.1.2.4. RAID 3 Acceso Paralelo. (Intercalado)

RAID 3 a diferencia de RAID 2, requiere solamente de un disco redundante y en lugar de utilizar un código para la detección y corrección de errores, calcula un **bit de paridad** para el conjunto de bits individuales en la misma posición en todos los discos de datos (ver figura 12).

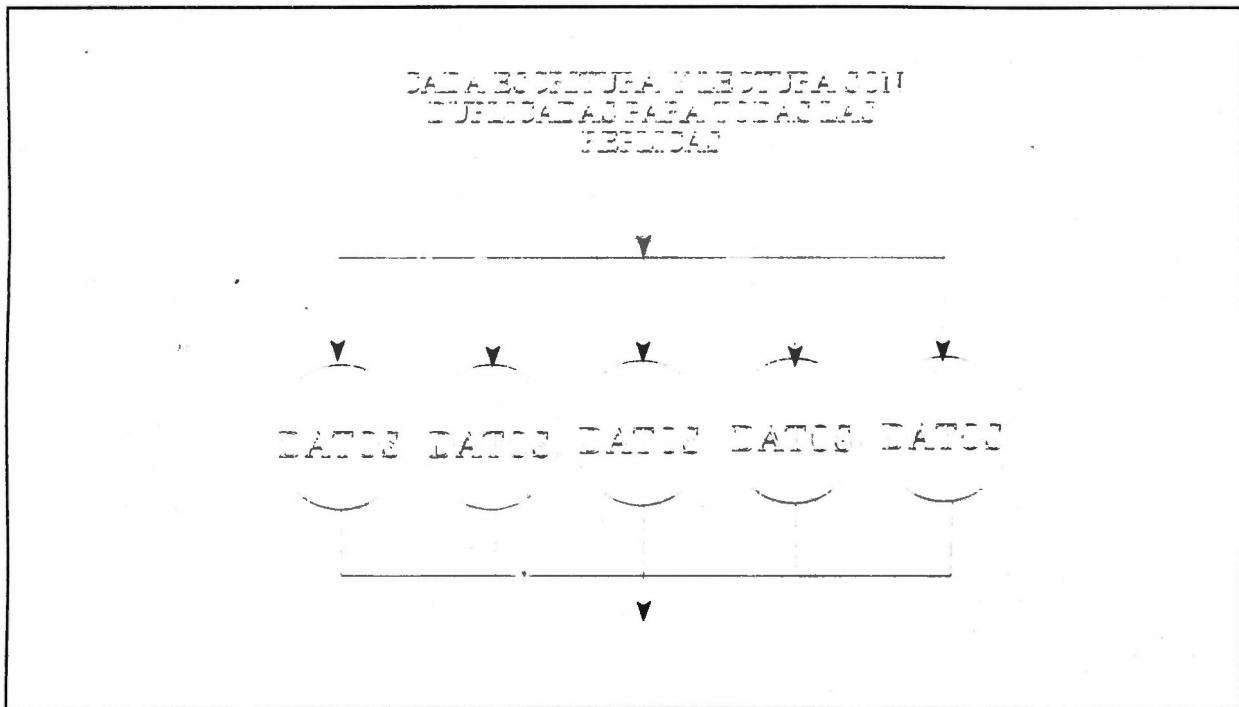


Fig 12: RAID 3

**Redundancia.** En caso de fallo, se accede a la unidad de paridad y se reconstruyen los datos desde el resto de los dispositivos. La reconstrucción de los datos es de la siguiente forma: si tenemos discos desde  $X_1$  hasta  $X_n$ , en donde  $X_n$  es el disco de paridad, la paridad para el bit  $a$  se calcula con la fórmula

$$X_n(a) = X_{n-1}(a) \oplus X_{n-2}(a) \oplus \dots \oplus X_2(a) \oplus X_1(a) \oplus X_0(a).$$

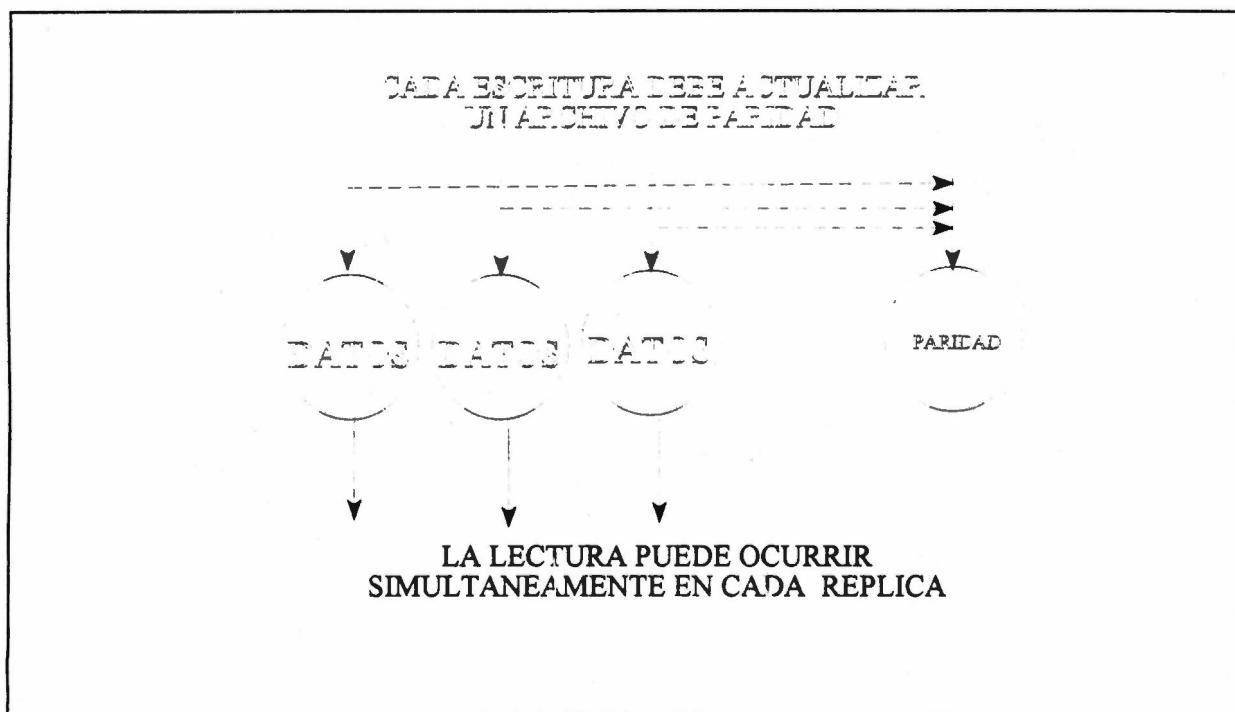
Si suponemos que la unidad  $X_1$  fallo, solamente sumamos  $X_n(a) \oplus X_1(a)$  en ambas partes de la fórmula, nos queda que

$$X_1(a) = X_{n-1}(a) \oplus X_{n-2}(a) \oplus \dots \oplus X_2(a) \oplus X_n(a) \oplus X_0(a)$$

reconstruyendo cualquier tira de datos a partir del resto de los discos. Este principio es válido en los niveles 3, 4 y 5 de RAID. En el caso de fallo de disco, el resto de la información sigue disponible aunque de una forma reducida. Se puede recuperar la información en el momento con la operación O\_exclusiva, teniendo que cambiar y regenerar los discos que fallen por discos nuevos. El problema que tiene este nivel de RAID es que sólo se puede ejecutar a la vez una petición, por lo tanto, en un nivel de transacciones elevado se vuelve ineficiente.

### 5.1.2.5. RAID 4. Acceso independiente. (Bloque de paridad intercalado)

Los niveles 4 y 5 de RAID usan una técnica de acceso independiente, de forma que las transacciones separadas se atienden en paralelo, por lo que son más adecuados para aplicaciones que requieren velocidades altas. Se usan tiras de datos relativamente grandes y en el caso de RAID 4, se calcula una tira de paridad a partir de las correspondientes tiras en cada disco de datos. Los datos se almacenan en la tira correspondiente del disco de paridad (ver figura 13).



Fif 13: RAID 4

El problema con este nivel de RAID se encuentra en la actualización de la tira de paridad, ya que cada escritura en la base de datos representa dos escrituras y dos lecturas. Si suponemos la misma formula

$$X_n(a) = X_{n-1}(a) \oplus X_{n-2}(a) \oplus \dots \oplus X_2(a) \oplus X_1(a) \oplus X_0(a)$$

donde  $X_n$  es la tira de paridad para cada modificación en un bit  $a$  en  $X_1$  y agregamos una prima en todos los bits que han sido alterados

$$X'_n(a) = X_{n-1}(a) \oplus X_{n-2}(a) \oplus \dots \oplus X_2(a) \oplus X'_1(a) \oplus X_0(a)$$

nos queda que

$$X'_n(a) = X_{n-1}(a) \oplus X_{n-2}(a) \oplus \dots \oplus X_2(a) \oplus X'_1(a) \oplus X_0(a) \oplus X_1(a) \oplus X_1(a)$$

entonces

$$X'_n(a) = X_n(a) \oplus X'_{-1}(a) \oplus X_1(a)$$

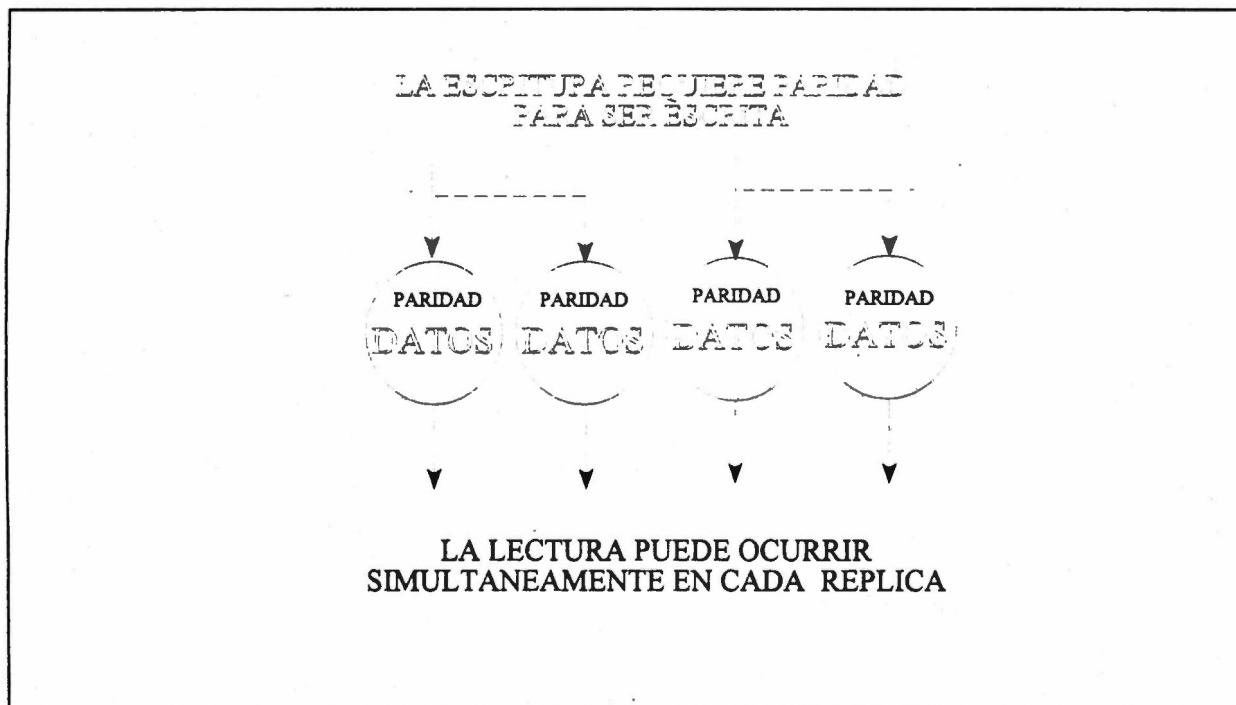
lo que quiere decir que para actualizar la tira de paridad y la tira de datos es necesario leer las antiguas tiras de paridad y de datos y despues calcular las nuevas tiras.

El efecto es más evidente cuando se escriben bloques relativamente pequeños de datos, ya que nuevos datos requieren nueva paridad, haciendo muy ineficiente el proceso y derivando la recomendación de diseño de definir el tamaño de los bloques teniendo en cuenta este comportamiento (z).

#### 5.1.2.6.RAID 5. Acceso independiente. (Paridad distribuida en bloques)

A diferencia de RAID 4 las tiras de paridad se distribuyen a lo largo de todos los discos, evitando así posibles cuellos de botella, al no tener un disco de paridad en el cual se va a detener el proceso (ver figura 14).

Este nivel de RAID está enfocado a un grado de petición alto, con lectura intensiva y para consulta de datos. Es el adecuado, con una mezcla del nivel 1, para hacer discos en espejo y de esta manera, manejar una réplica de los archivos, con un buen control de éstos.



**Fig 14: RAID 5**

En las siguientes tablas se muestra como quedan los archivos distribuidos en los discos, según el nivel RAID:

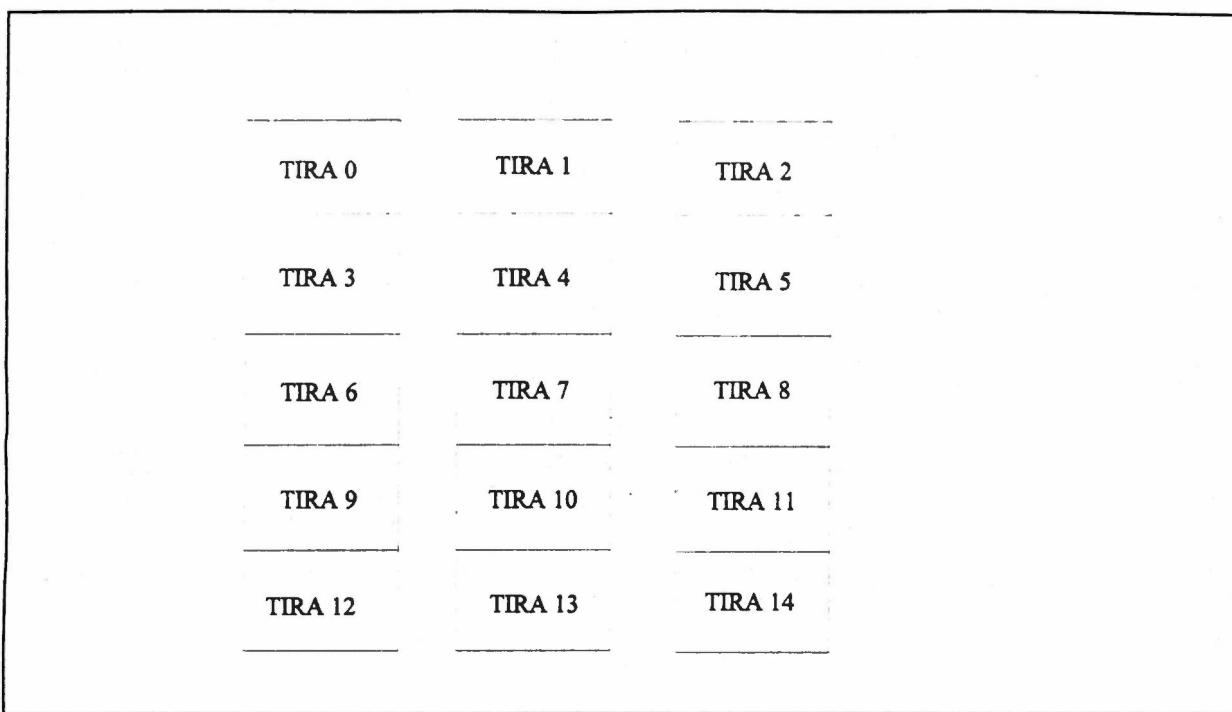


Fig 15: ARCHIVOS RAID 0

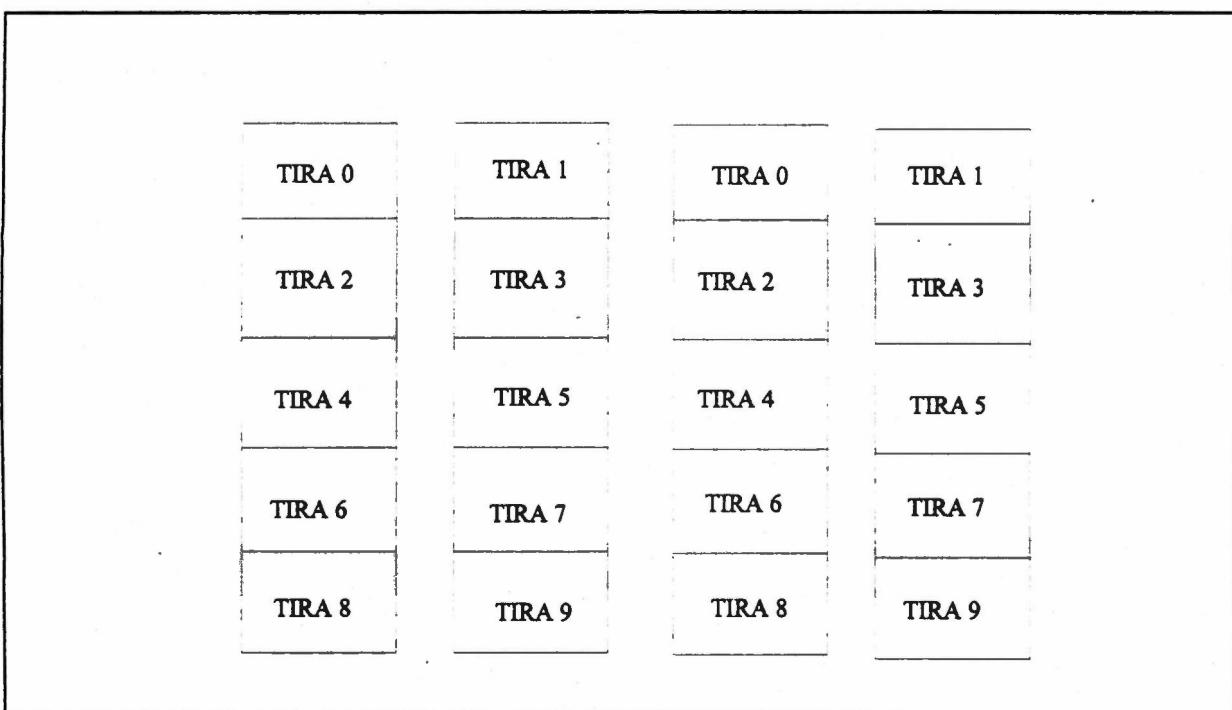


Fig 16: ARCHIVOS RAID 1

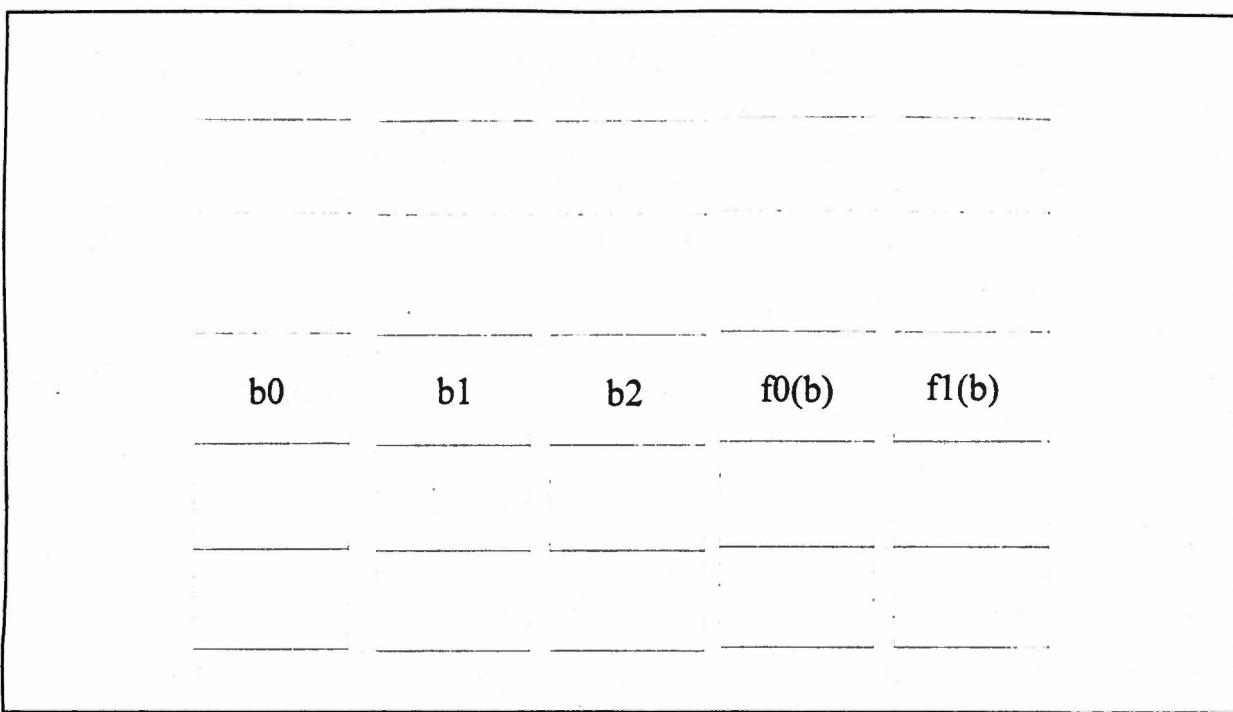


Figura 17: Archivos RAID 2

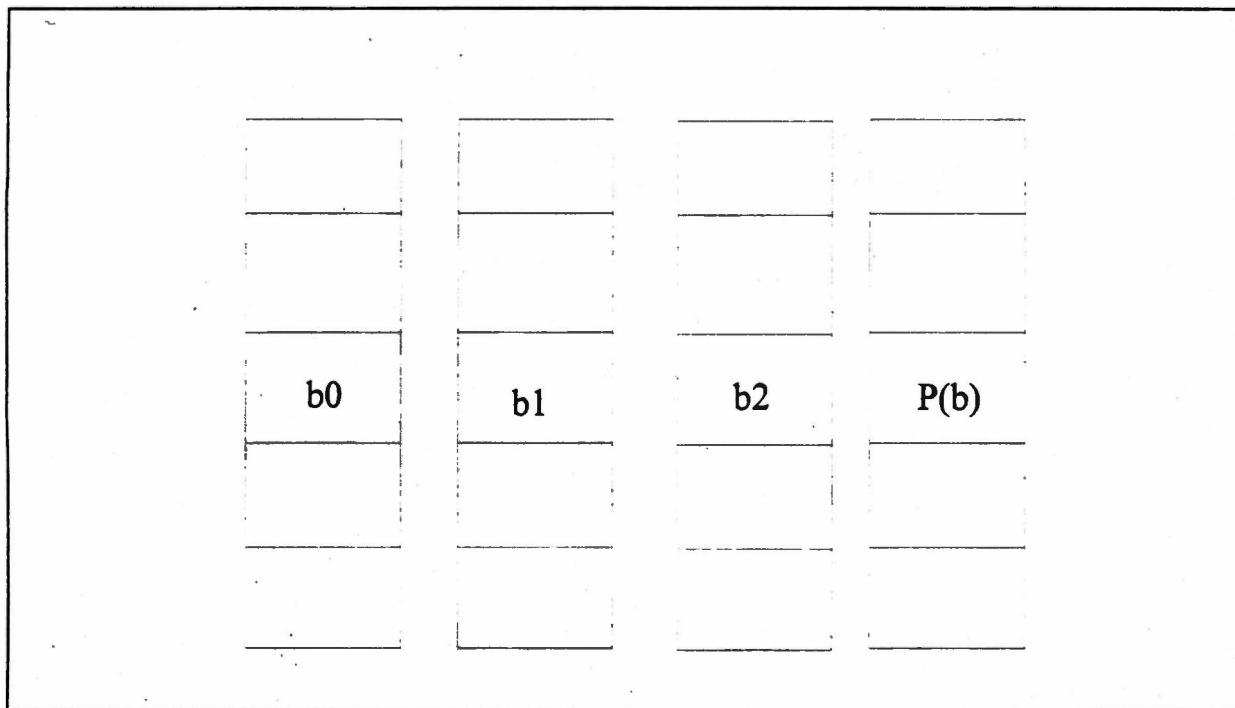


Fig 18: ARCHIVOS RAID 3

bloque 0	bloque 1	bloque 2	bloque 3	P(0-3)
bloque 4	bloque 5	bloque 6	bloque 7	P(4-7)
bloque 8	bloque 9	bloque 10	bloque 11	P(8-11)
bloque 12	bloque 13	bloque 14	bloque 15	P(12-15)

Fig 19: ARCHIVOS RAID 4

bloque 0	bloque 1	bloque 2	bloque 3	P(0-3)
bloque 4	bloque 5	bloque 6	P(4-7)	bloque 7
bloque 8	bloque 9	P(8-11)	bloque 10	bloque 11
bloque 12	P(12-15)	bloque 13	bloque 14	bloque 15

Fig 20: ARCHIVOS RAID 5

En el presente trabajo se propone, para tener una mayor seguridad en los datos de la base, el utilizar RAID de la siguiente forma; del RAID 1 tomar el concepto de los discos en espejo, para poder mandar copias a localidades diferentes, del RAID 5 utilizarlo en las distintas localidades para la

recuperación automática y en caso de fallo de disco, poder manejar réplicas en los archivos y no a nivel de disco. Los manejadores de bases de datos actuales, como ORACLE, SYBASE y DB2, ya contienen los archivos en espejo, mandando las modificaciones de los registros a dos localidades en la red. Igualmente se propone utilizar el nivel 5 de manera que los archivos se distribuyan en varios discos, ya que si la base es distribuida, su número de usuarios sería mayor; de esta forma lograr que las consultas se distribuyan entre varios discos y acelerar así el tiempo de respuesta de los discos, al trabajar en paralelo.

También se propone tomar del RAID 1 el concepto de los archivos en espejo, aunque como ya mencionamos, las últimas versiones de manejadores de bases de datos ya lo contienen. Esta propuesta permite:

- a) Hacer la redundancia de los archivos duplicando la información en dos localidades diferentes, siendo cada una independiente de la otra.
- b) Lograr que las peticiones de lectura se puedan efectuar en cualquiera de las localidades sin alterar a las otras, lo que implica que mientras más busquedas se hagan al mismo tiempo de un archivo, el tiempo de latencia total va a ser menor, ya que se distribuyen las consultas en varias localidades.
- c) Que la recuperación de las localidades sea muy sencilla, al detectar el administrador de la base de datos que una de ellas no está actualizada, crea una copia exacta tomando los datos de otra localidad.
- d) Que las peticiones de escritura sean más rápidas, ya que se hacen las actualizaciones en paralelo.

Se propone tomar del RAID 5 la recuperación de archivos en cada localidad, a través de la detección de errores utilizando una tira de paridad, lo cual permitirá:

- a) La corrección de problemas en alguna localidad, para evitar resolverlos mediante mensajes en la red, utilizando RAID 5 en todas las localidades en que sea posible.
- b) Corregir los archivos defectuosos en cada localidad, en el momento de grabarlos.

No debemos soslayar el problema que dicha aplicación presenta en la escritura que, como se mencionó anteriormente, se vuelve más lenta.

Nota (4): Stallings, William. "Organización y Arquitectura de Computadores", Cuarta edición, Prentice Hall, México, 1994, página 161.

Nota (x): The Santa Cruz Operation. "SCO Virtual Disk Manager". 1997.

Nota (y): Tandem CR1000 RAID Storage Subsystem. Continously available data storage for clustered systems, Compaq Computers.

Nota (z): R. Mattson. "Advances in Disk Array Technology", 1992 IBM Colloquium in Computer Science and Technology Series.

## CAPÍTULO 6:

### ESTUDIO COMPARATIVO

#### 6.1. ESTUDIO COMPARATIVO

Para obtener las conclusiones del presente trabajo, se comparó su propuesta con los métodos encontrados en los dos tipos de manejadores más conocidos, mismos que fueron mencionados en el inciso 1.2., del primer capítulo; igualmente se comparó con la tesis de maestría en ciencias computacionales elaborada por el Ingeniero Francisco Javier Flamenco Sandoval, la que fué presentada en diciembre de 1996.

##### 6.1.1. Administradores Que Crean Comunicación Por Medio De Compuertas.

A continuación se mencionan las ventajas y desventajas de la propuesta en relación a los administradores que crean comunicación por medio de compuertas:

Las ventajas más notorias son:

- a) La ubicación de los archivos es transparente para los usuarios y programadores de la base de datos.
- b) La utilización de un Esquema RAID permite la recuperación de archivos, en el caso de dañarse éstos.
- c) Se mejora el tiempo de respuesta de los discos, al repartir la consulta entre varios de ellos.

Las desventajas más significativas son las siguientes:

- a) El tiempo para direccionar la consulta es mayor.
- b) La actualización del administrador crea más mensajes en la red, que en el sistema comparado.

- c) Se crean más archivos para la administración del sistema en la red, que en el sistema comparado.

#### **6.1.2. Administradores Administrados Por Uno Local.**

En cuanto a los administradores formados por un manejador de base de datos, un diccionario de datos, un componente de comunicación de datos y un manejador distribuido de base de datos, la comparación nos genera las siguientes ventajas:

- a) Las tres primeras son las mismas que en el caso anterior.
- b) El diccionario de datos es más pequeño, por que en un sistema como éste, la tabla LAR (Local Archivos del Esquema Local) va a contener exclusivamente los archivos utilizados por la localidad y no todos los que se encuentran en la red, provocando que el tiempo de consulta a un archivo sea más rápido.

En cuanto a las desventajas de la propuesta en relación con este tipo de administradores, encontramos que:

- a) El tiempo para direccionar la consulta es más lento, si el archivo que se va a consultar no se encuentra en la tabla LAR.
- b) La actualización del administrador crea más mensajes en la red.
- c) Se crean más archivos para la administración del sistema en la red.

#### **6.1.3. Tesis de Flamenco Sandoval.**

Consideramos interesante hacer la comparación de nuestra propuesta con la tesis de maestría del Ingeniero Francisco Javier Flamenco Sandoval. Para realizar lo anterior, se hace necesaria una explicación del funcionamiento de su propuesta.

El autor propone un monitor global de distribución, que puede estar ubicado en una o en varias localidades, funcionando como un servidor de archivos, que va a contener toda la información de los archivos de la red en dos niveles:

- a) Un Esquema Global, que contiene la información de todas las tablas y está integrado de la siguiente forma:

TABLA 19: VISTAS\_GLOBALES

NVISTA	Nombre de la vista externa del usuario.
SQLG_SELECT	Sentencia SQL empleada para construir la tabla

TABLA 20: TABLAS\_GLOBALES

NTABLA	Nombre de la relación global
NODO	Nombre del nodo donde la tabla está distribuida
TIPO	Tipo: Vista o Tabla

TABLA 21: COLUMNAS\_GLOBALES

NCOLUMNA	Nombre de la columna
NTABLA	Tabla a la que pertenece la columna
TIPO	Tipo de datos que contiene la columna
LON	Longitud en bites de la columna

b) Un Esquema de fragmentos, que contiene toda la información de los mismos, en el caso de encontrarse fragmentada la tabla:

TABLA 22: TABLAS\_FRAGMENTADAS

SITE	Nombre del nodo que contiene la tabla
NTABLA	Nombre de la tabla de la cual se deriva el fragmento
NOMFRAG	Nombre de este fragmento de la tabla
NRENGLON	Número de renglones contenidos en la tabla fragmentada

TABLA 23: COLUMNAS\_FRAGMENTADAS

NOMCOL	Nombre de la columna
NOMFRAG	Nombre del fragmento de la tabla al que pertenece la columna
TIPO	Tipo de dato de la columna
LON	Longitud de la columna

El sistema propone un servidor de archivos, que va a funcionar desde una localidad específica; este servidor contiene la información de todas las tablas de la red que estén distribuidas y la información necesaria para su fragmentación, todo desde el Monitor Global de Distribución.

En el siguiente análisis comparativo de las ventajas y desventajas entre las dos propuestas, se explica el funcionamiento de ésta, por lo que evitaremos ser redundantes, esta comparación la realizamos utilizando el siguiente cuadro, con objeto de lograr una mayor claridad.

TABLA 24: TABLA COMPARATIVA		
No	Tesis Francisco Flamenco	Tesis Guillermo Kunz
SEMEJANZAS		
1	Genera un administrador de bases de datos distribuidas	Genera un administrador de bases de datos distribuidas
2	Genera un Esquema con la información de todos los archivos en la red	Genera un Esquema con la información de todos los archivos en la red
DIFERENCIAS		
1	Contempla desde el Esquema Global un Esquema de Fragmentos que tiene la información a detalle de cada fragmento en la red.	Indica únicamente las localidades en las que se encuentran los fragmentos.
2	La información de toda la red se encuentra en un sitio servidor.	La información de toda la red se encuentra en un sitio servidor y la información utilizada más frecuentemente por cada localidad se encuentra en el mismo, en un Esquema Local
3	Genera un Monitor Global de Distribución único.	Implementa funciones para la administración de la base de datos de archivos
4	Genera rutinas para el intercambio de archivos	Implementa un sistema de recuperación, para el caso de falla en algún nodo.

Una vez realizadas las comparaciones anteriores, que permiten identificar la individualidad del presente trabajo, consideramos que las ventajas del mismo son las siguientes:

- a) La utilización de un Esquema RAID, que permite la recuperación de los archivos, en caso de darse un daño en éstos.
  - b) La mejora en el tiempo de respuesta de los discos, al repartir la consulta entre varios de ellos.
  - c) La aceleración del tiempo de respuesta, al contar con un Esquema Local en cada localidad, con una lista de los archivos más utilizados por éstas.
  - d) El no crear cuellos de botella en los Esquemas Globales o en el Monitor Global de Distribución.
- Sin embargo, no podemos dejar de mencionar las desventajas de nuestra propuesta, las cuales son necesariamente una consecuencia lógica de la misma:
- a) Se crean más archivos para la administración del sistema en la red.
  - b) Se crean más mensajes en la red en una actualización del administrador.

## CAPÍTULO 7: CONCLUSIONES

El empleo de bases de datos para el tratamiento de la información en las organizaciones es creciente y las inversiones efectuadas en la infraestructura de cómputo para su cabal aprovechamiento es significativa. Los negocios han hecho un aprovechamiento singular de sus posibilidades así como las organizaciones educativas y de investigación y las tendencias internacionales en el uso de Internet, siempre creciente, permite suponer que unos y otras continuarán haciéndolo en el futuro. A manera de conclusión podemos afirmar que la utilización de nuestra propuesta, en los administradores de bases de datos, mejora sensiblemente la seguridad de los mismos y en algunos casos sin sacrificar la velocidad, con los siguientes beneficios para sus posibles usuarios:

a) La propuesta significa una alternativa para la transparencia en el manejo de archivos distribuidos.

Esto significa que al nivel de las aplicaciones no es necesario ocuparse de la ubicación de los archivos necesarios para el tratamiento de la información, liberando al diseñador de aplicaciones de tales tareas y al usuario de referencias innecesarias y de ineficiencias involuntarias al tratar de diseñar él mismo un esquema de ubicación de archivos. Esto permitiría contemplar aplicaciones en extremo eficientes en el proceso sustantivo de la información, más cercanas a los usuarios de la misma y por consecuencia potencialmente desarrollables no necesariamente por expertos y con menores requerimientos para el equipo necesario.

Lo anterior apoyaría por ejemplo el empleo del concepto moderno de "clientes ligeros" en un esquema cliente - servidor, o bien apoyándose en la Internet, de aplicaciones de transacciones hacia los clientes, auto - ayuda y comercio electrónico por citar dos de ellas, más allá de las simples páginas Web.

Conviene recordar que las dos razones principales de los grandes negocios para invertir en aplicaciones empacadas o desarrollar su propio software son en primer término mejorar el acceso

del usuario final a la información y la segunda apoyar nuevos productos y servicios. Bajo tales imperativos, un esquema que hace más eficiente el acceso permite, por ejemplo, desarrollar eficientemente la minería de datos que muchas organizaciones podrían efectuar para maximizar el beneficio de sus colecciones, algunas probablemente de varios años de antigüedad y cuya explotación beneficiaría seguramente al negocio.

b) Mejora el tiempo de respuesta de los discos, al distribuir los archivos en ellos y haciendo que las consultas de un mismo archivo se distribuyan.

Aunque la tecnología de la microelectrónica asegura por muchos años la optimización de los dispositivos de almacenamiento de datos, las demandas sobre ellos se incrementan también sensiblemente. Un esquema que tienda a optimizar el funcionamiento de las unidades de disco, sujetas a cargas de trabajo crecientes, colabora a hacer más efectivo el acceso a los datos de interés, que en algunas circunstancias podría ser demandado por miles de usuarios.

c) Resuelve el problema de la continuidad y la tolerancia a fallas, ya que al tener réplicas y rutas alternativas, siempre hay comunicación con los archivos.

Bajo el mismo tenor de la conclusión anterior, la disponibilidad de los datos en todo momento es un imperativo de la moderna informática distribuida para poder asegurar a los usuarios de la información los beneficios que estos esperan de su conocimiento y aplicación. Un esquema de servicios de información, como el requerido por ejemplo por el sistema financiero internacional, sujeto a fallas en los equipos servidores de información es impensable en la época actual. El aprovechamiento de las redes internacionales de telecomunicaciones implica que la información debe hacerse fluir por rutas alternas a la regular sin que los usuarios tengan que intervenir para ello, bajo un esquema como el propuesto que nos permita, a costos razonables, acceder a réplicas, ser tolerantes a fallas y mantener una continuidad en los servicios de acceso a la información.

d) Resuelve también los problemas de consistencia y recuperación, ya que actualiza los archivos y los recupera en caso de caídas de nodos en el sistema

A pesar de los indudables avances en la confiabilidad de los sistemas de almacenamiento de información las fallas pueden ocurrir y comprometer el acceso a la información en un esquema distribuido. El esquema propuesto asegura la consistencia en los contenidos de los archivos y permite su recuperación en caso de caídas de alguno de los nodos del sistema. En algunos casos esto permitiría recuperar información inclusive en nodos no atendidos temporalmente, sin que disminuyera la eficacia de consultas de los usuarios.

La discusión que antecede y las conclusiones presentadas en los párrafos anteriores pretenden contribuir al perfeccionamiento de los esquemas de acceso a la información organizada en computadores teniendo en cuenta la penetración de estas tecnologías en la sociedad, los negocios, la educación e inclusive en el hogar, aportando criterios que permitirían optimizar una de las tecnologías, como la de bases de datos distribuidas, que han aportado avances sistemáticos y soluciones concretas a problemas del tratamiento y difusión de la información.

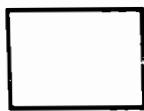
## BIBLIOGRAFÍA.

- [BELL92] Bell David y Jane Grimson, "Distributed Database Systems", Addison-Wesley, 1992.
- [CERI85] Ceri Stefano y Giuseppe Pelagatti, "Distributed Databases Principles & Systems", Mc. Graw Hill, 1985.
- [CHU79] Chu Wesley W., "Optimal File Allocation in a Multiple Computer System", The 1st. International Conference on Distributed Computing Systems, 1979.
- [DATE90] Date C.J., "What is a Distributed Database System" en "Relational Database Writings 1985-1989", Reading, Mass, Addison-Wesley, 1990.
- [DATE93] Date C.J., "Introducción a los Sistemas de Bases de Datos", Volumen 1, quinta edición, Addison-Wesley Iberoamericana, EUA, 1993.
- [GUILLEN95] Guillén Mario, Roberto Valdivia B. Rodolfo Pazos R. y Guillermo Rodriguez O., "Manejo de Transacciones en Multibase de Datos", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.
- [HERNANDEZ95] Hernández L. Carlos, Hector Ruiz B., Raúl Acosta B. y Jaime Cerdá J., "Especificación Formal de un Sistema de Archivos Distribuido", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.
- [KORTH93] Korth Henry F. y Abraham Silberschatz, "Fundamentos de Bases de Datos", Mc. Graw Hill, 1993.
- [MCFADDEN93] McFadden Fred R. y Jeffrey A. Hoffer, "Modern Database Management", The Benjamin/Cummings Publishing Company. Inc., 1993.
- [MONTES95] Montes R. Raúl Jacinto, "Un Protocolo Distribuido de Validación Automática", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.
- [PAZOS95] Pazos R. Rodolfo A., Joaquín Pérez O. y Leopoldo Zepeda S., "Extensiones al Lenguaje SQL para el Logro de Transparencia de Fragmentación de Bases de Datos Distribuidas", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.
- [PEREZ95] Pérez O. Joaquín, Rodolfo A. Pazos R., Guillermo Rodríguez O. y Roberto Valdivia B., "Arquitectura de una Herramienta Case para el Diseño de Bases de Datos Distribuidas", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.
- [RODRIGUEZ95] Rodríguez Martínez Andrés Florencio, Guillermo Rodríguez Ortiz y Sunil Vadera. "A Conceptual Model for Case-Bases", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Zacatepec, México, 1995.

- [SOSA95] Sosa Sosa Víctor J., Joaquín Pérez O. y Rodolfo Pazos R., "Algoritmo para Descomposición de Consultas SQL en Subconsultas Bajo la Arquitectura Cliente/Servidor", 2<sup>a</sup> Congreso Internacional de Investigación en Ciencias Computacionales, Instituto Tecnológico de Záratepec, México, 1995.
- [STALLING94] Stallings William, "Organización y Arquitectura de Computadores", Prentice Hall, cuarta edición, México, 1994.
- [TABORGA82] Taborga Huáscar, "Como Hacer una Tesis", Editorial Grijalbo, México, 1982.
- [THOMAS79] Thomas Robert H., "A Solution to the Concurrency Control Problem for Multiple Copy Data Bases", The 1st. International Conference on Distributed Computing Systems, 1979.
- [SCO97] The Santa Cruz Operation. "SCO Virtual Disk Manager". 1997.
- [TANDEM] Tandem CR1000 RAID Storage Subsystem. Continuously available data storage for clustered systems, Compaq Computers.
- [MATTSON92] R. Mattson. "Advances in Disk Array Technology", 1992 IBM Colloquium in Computer Science and Technology Series.



INICIO Y FIN DE PROCEDIMIENTO



OPERACIONES, ASIGNACION DE  
VALORES Y ENVIO DE MENSAJES



DECISIONES



CONECTORES



PIDE UN VALOR POR TECLADO



DESPLIEGA EN PANTALLA UN VALOR



## ANEXO 2

**ALGORITMO 1:** Módulo controlador de base de datos y comunicación

crea una lista R ( Descripción)

recibe un mensaje X ( Clave, Emisor, Lista)

si X (Clave)

a{ mientras no exista un EOF

[ lee de la lista X (lista)

busca GAR (Nombre) = X (Lista (Archivo)) para encontrar GAR (Esquema Local)

revisa que el Esquema Local GAR (Esquema Local) no tenga algún bloqueo en GAR (Status).

si tiene bloqueo, ejecuta el programa RB (Revisa Bloqueos del módulo de actualización)

si no tiene bloqueo, toma el valor de AUX (Lista (Archivo, GAR (Esquema Local))) y la agrega al final de la lista R (Descripción).

]

manda el mensaje Y ("a", Sitio, R) a Emisor

}

b{ mientras no exista un EOF

[ lee de la lista X (Lista (Archivo, Clave))

si X (Lista (Clave))

1{ crea un formato de ESQUEMA GLOBAL con sus archivos GEG, GEL y GAR en la dirección en la que se encuentre,

solicita y lee un valor para DS que es la dirección de otro Esquema Global trabajando,

si no es dado un valor para DS significa que es un sistema nuevo y se crearan GEG, GEL y GAR únicamente con los datos de la localidad en que se encuentran,

si es dado un valor en DS se hará una copia idéntica del GEG, GEL y GAR del Esquema Global que se encuentra en DS en la localidad actual,

se coloca en GEG (Esquema Global) = X (Lista (Archivo))

revisa en GEG (Dirección Sitio) si existe una equivalente a la localidad en donde se encuentra y si no la agrega.

modifica la variable GEG (Última Actualización) por fecha y hora actual.

modifica la variable GEG (Reporta) por "ADMIN"

manda el mensaje Y ("c", Sitio, (Esquema Global,"1")) a los Esquemas Globales de GEG (Esquema Global).

manda el mensaje Y ("b") a Emisor

}

2{ se coloca en GEG (Esquema Global) = X (Lista (Archivo)) y borra GEG (Esquema Global) de la lista.

manda el mensaje Y ("c", Sitio, (Lista,"2")) a los Esquemas Globales de GEG  
 (Esquema Global)  
 manda el mensaje Y("b", Sitio, (Lista,"2")) a los Esquemas Locales de GEL  
 (Esquema Local)  
 manda el mensaje Y ("b") a Emisor  
 }  
 3{ se coloca en GEG (Esquema Global) = X (Lista (Archivo)) solicita y lee un  
 valor para ST que es el estado en el que se encuentra el GEG  
 Global) (Esquema  
 Solicita y lee un valor para DR que es la nueva dirección del esquema GEG  
 (Esquema Global)  
 si no tiene valor ST o DR cancela la modificación y termina.  
 si tiene valor ST, GEG (Status) toma el valor de ST y GEG (Dirección Sitio)  
 toma el valor de DR  
 GEG (Última Actualización) toma la fecha y la hora del momento en que se  
 hace el cambio  
 manda el mensaje Y ("c", Sitio, X (Lista (Archivo, ST, DR),"3")) a los  
 Esquemas Globales de GEG (Esquema Global)  
 manda el mensaje Y ("d", Sitio, X (Lista (Archivo, ST, DR),"3")) a los  
 Esquemas Locales de GEL (Esquema Local)  
 manda el mensaje Y ("b") a emisor  
 }  
 fin del si condicional  
 ]  
 }  
 c{ mientras no exista un EOF  
 [ lee de la lista X (Lista (Archivo, Clave))  
 si X (Lista (Clave))  
 1{ crea un formato de Esquema Global con sus archivos GEG, GEL y GAR en  
 la dirección en la que se encuentre,  
 se crea copia idéntica del GEG, GEL y GAR del Esquema Global que se  
 encuentra en X (Emisor) en la localidad actual,  
 modifica la variable GEG (Última Actualización) por fecha y hora actual.  
 modifica la variable GEG (Reporta) por X (Emisor)  
 se coloca en GEL (Esquema Local) = X (Lista (Archivo))  
 modifica la variable GEL (Última Actualización) por la fecha y hora actual.  
 modifica la variable GEL (Reporta) por Emisor  
 manda el mensaje Y ("b") a Emisor  
 }

```

2{    se coloca en GEG (Esquema Global) = X (Lista (Archivo)) y borra GEG
      (Esquema Global) de la lista.
      manda el mensaje Y ("b") a Emisor
}
3{    se coloca en GEG (Esquema Global) = X (Lista (Archivo))
      GEG (Status) toma el valor de ST y GEG (Dirección Sitio) toma el valor de
      DR
      GEG (Última Actualización) toma la fecha y la hora del momento en que se
      hace el cambio
      manda el mensaje Y ("b") a Emisor
}
fin del si condicional
}

d{ mientras no exista un EOF
[    lee de la lista X (Lista (Archivo, Clave))
    si X (Lista (Clave))
        1{    busca los archivos GEL y GAR en la dirección en la que se encuentre,
              se coloca en GEL (Esquema Local) = X (Lista (Archivo))
              revisa en GEL (Dirección Sitio) si existe una equivalente al Sitio en donde
              se encuentra y si no la agrega.
              modifica la variable GEL (Última Actualización) por fecha y hora actual.
              modifica la variable GEL (Reporta) por "ADMIN"
              manda el mensaje Y ("e", Sitio, (Esquema Local, "1")) a los Esquemas
              Globales de GEG (Esquema Global)
              manda el mensaje Y ("b") a Emisor
        }
        2{    se coloca en GEL (Esquema Local) = X (Lista (Archivo)) y borra GEL
              (Esquema Local) de la lista.
              manda el mensaje Y ("e", Sitio, (Lista,"2")) a los Esquemas Globales de GEG
              (Esquema Global)
              manda el mensaje Y ("e", Sitio, (Lista, "2")) a los Esquemas Locales de GEL
              (Esquema Local)
              manda el mensaje Y ("b") a Emisor
        }
        3{    se coloca en GEL (Esquema Local) = X (Lista (Archivo))
              solicita y lee un valor para ST que es el estado en el que se encuentra el
              GEL (Esquema Local)
              solicita y lee un valor para DR que es la nueva dirección del esquema GEL
              (Esquema Local)
              si no tiene valor ST o DR cancela la modificación y termina.
        }
}

```

```

    si tiene valor ST GEL (Status) toma el valor de ST y GEL (Dirección Sitio)
    toma el valor de DR
        GEL (Última Actualización) toma la fecha y la hora del momento en que se
        hace el cambio
            manda el mensaje Y ("e", Sitio, X (Lista (Archivo, ST, DR),"3")) a los
            Esquemas Globales de GEG (Esquema Global)
            manda el mensaje Y ("e", Sitio, X (Lista (Archivo, ST, DR)"3")) a los
            Esquemas Locales de GEL (Esquema Local)
            manda el mensaje Y ("b") a Emisor
        }
        fin del si condicional
    ]
}

e{ mientras no exista un EOF
[    lee de la lista X (Lista (Archivo, Clave))
    si X (Lista (Clave))
        1{ busca los archivos GEL y GAR en la dirección en la que se encuentre,
           se coloca en GEL (Esquema Local) = X (Lista (Archivo))
           revisa en GEL (Dirección Sitio) si existe una equivalente al Sitio en donde
           se encuentra y si no la agrega.
           modifica la variable GEL (Última Actualización) por fecha y hora actual.
           modifica la variable GEL (Reporta) por X (Emisor)
           manda el mensaje Y ("b") a Emisor
        }
        2{ busca GEL (Esquema Local) = X (Lista (Archivo)) y lo borra
           manda el mensaje Y ("b") a Emisor
        }
        3{ se coloca en GEL (Esquema Local) = X (Lista (Archivo))
           solicita y lee un valor para ST que es el estado en el que se encuentra el
           GEL (Esquema Local)
           solicita y lee un valor para DR que es la nueva dirección del esquema GEL
           (Esquema Local)
           GEL (Status) toma el valor de ST y GEL (Dirección Sitio) toma el valor de
           DR
           GEL (Última Actualización) toma la fecha y la hora del momento en que se
           hace el cambio
           manda el mensaje Y ("b") a Emisor
        }
        fin del si condicional
]
}

```

```

f { mientras no exista un EOF
[    lee de la lista X (Lista (Archivo, Clave))
    si X (Lista (Clave))
        1{    busca el archivo GAR en la dirección en la que se encuentre,
            se coloca en GAR (Archivo) = X (Lista (Archivo))
            solicita y lee un valor para EL
            revisa si existe GAR (X (Lista (Archivo)), Esquema Local) = GAR (X (Lista
                (Archivo)), EL) y si no la agrega.
            manda el mensaje Y ("g", Sitio, (GAR (Archivo, EL),"1")) a los Esquemas
            Globales de GEG (Esquema Global)
            manda el mensaje Y ("b") a Emisor
        }
        2{    se coloca en GAR (Archivo) = X (Lista (Archivo)) y borra GAR (Archivo,
            Esquema Local) de la lista.
            manda el mensaje Y ("g", Sitio, (Lista,"2")) a los Esquemas Globales de GEG
            (Esquema Global)
            manda el mensaje Y ("f", Sitio, (Lista,"2")) a los Esquemas Locales de GEL
            (Esquema Local)
            manda el mensaje Y ("b") a Emisor
        }
        3{    se coloca en GAR (Archivo) = X (Lista (Archivo))
            solicita y lee un valor para EL que es el lugar donde se encuentra el GAR
            (Esquema Local)
            solicita y lee un valor para EL1 que es el lugar donde se va a encontrar el
            GAR (Esquema Local)
            si no tiene valor EL o EL1 cancela la modificación y termina.
            manda un mensaje Y ("f", Sitio, (GAR (Archivo, EL),"2")) a Sitio
            manda un mensaje Y("f", Sitio, (GAR (Archivo, EL1),"1")) a Sitio
            manda un mensaje Y ("b") a Emisor
        }
    fin del si condicional
]
g { mientras no exista un EOF
[    lee de la lista X (Lista (Archivo, Clave))
    si X (Lista (Clave))
        1 {    crea GAR (EL, X (Lista (Archivo))).
        }
        2 {    borra GAR (EL, X (Lista (Archivo))).
        }

```

```

    fin del si condicional
}

h { lee de la lista X (Lista (Archivo, Clave))
    si X (Lista (Clave))
        1 { busca GEG (Esquema Global) = X (Lista (Archivo)) y modifica GEG (Status)
            por 0
        }
        2{ busca GEL (Esquema Local) = X (Lista (Archivo)) y modifica GEL (Status)
            por 0
        }
    manda mensaje Y ("g", Sitio, Lista)
    manda mensaje Y ("b") a Emisor
}

i{ lee de la lista X (Lista (Archivo, Clave))
    si X (Lista (Clave))
        1 { busca GEG (Esquema Global) = X (Lista (Archivo)) y modifica GEG (Status)
            por 0}
        2 { busca GEL (Esquema Local) = X (Lista (Archivo)) y modifica GEL (Status)
            por 0}
    manda mensaje Y ("b") a Emisor
fin del si condicional

```

Nota: para cada uno de los mensajes enviados, se ejecuta el procedimiento VM (verifica mensajes del módulo de edición de esquemas globales alternos), para verificar que el mensaje haya llegado a su destino; para en el caso de no haber llegado, bloquear el archivo no actualizado, esperando que el programa RB lo actualice después.

**ALGORITMO 2:** Módulo de recuperación

Al iniciar, el sistema manda un mensaje de consulta de algún archivo que esté en el Sitio a un Esquema Global que no se encuentre en el mismo Sitio, en el caso de estar bloqueado, el controlador de base de datos hará que se ejecute el programa RB, el cual quitará el bloqueo y actualizará los archivos GAR, GEL, GEG, LAR (Local Archivo), LEL (Local Esquema Local) y LEG (Local Esquema Global), y al recibir un mensaje de respuesta, inicia el proceso normal.

**ALGORITMO 3:** Edición de esquemas globales alternos.

Inicio

[

Pregunta el sitio del Esquema Global a trabajar y lo guarda en Sitio  
 Despliega el siguiente menú:

- 1) Crear
- 2) Borrar
- 3) Actualizar
- 4) Consultar

y guarda la respuesta en la variable Clave,

Despliega el siguiente menú:

- 1) Esquema Global
- 2) Esquema Local
- 3) Archivos

y guarda la respuesta en la variable Esquema,

repite hasta leer un EOF o vacío

{

Pide un nombre de un archivo, Esquema Local o Esquema Global  
 AUX =  
 AUX + El valor del nombre que acaba de leer

}

si el valor de la variable Esquema

- 1{ manda un mensaje Y ("b", Sitio, (AUX, Clave)) a Sitio}
- 2{ manda un mensaje Y ("d", Sitio, (AUX, Clave)) a Sitio}
- 3{ manda un mensaje Y ("f", Sitio, (AUX, Clave)) a Sitio}

fin del si condicional

si recibe el mensaje Y ("a", Sitio, Lista) lo despliega en la pantalla.

]

**ALGORITMO 4:** Verificación de mensajes

Al salir un mensaje empieza a correr un Timer, que será de la duración suficiente para que el mensaje llegue a su destino y regrese.

Si no hay respuesta del mensaje éste será enviado una vez más.

El tiempo del Timer será definido de acuerdo al tamaño de la red.

**ALGORITMO 5:** Revisa bloqueos RB

Inicio

```
[    si el tiempo dividido entre Timer no da residuo
        {  revisa en la tabla GEG (Status) si existe algún bloqueo
            si existe bloqueo revisa si Sitio = GEG (Reporta)
                si es igual manda un mensaje Y ("a", Sitio, (GEG, "1")) al Esquema Global
                que esté bloqueado,
                    si recibe respuesta cambia GEG (Status) a sin bloqueo
                    revisa en la tabla GEL (Status) si existe algún bloqueo
                    si existe bloqueo revisa si Sitio = GEL (Reporta)
                        si es igual manda un mensaje Y ("d", Sitio, (GEL, "2")) al Esquema Local
                        que esté bloqueado,
                            si recibe respuesta cambia GEL (Status) a sin bloqueo
        }
    ]
```

### **ALGORITMO 6:** Módulo controlador de base de datos y comunicación

Crea una lista R (Descripción)

Recibe un mensaje X (Clave, Emisor, Lista)

Si X (Clave)

a{}

b{ si existe LEG (X (Lista (Archivo))) lo borra

si la tabla LEG queda vacía, ejecuta el mensaje Y ("h", Sitio, Sitio).

manda mensaje Y ("b") a Emisor}

c{ mientras no exista un EOF

[ lee de la lista X (lista)

busca LAR (Nombre) = X (Lista (Archivo)) para encontrar

LAR(Esquema Local)

en el caso de no encontrar el archivo manda un mensaje Y ("a", Sitio,

Lista) al primer Esquema Global que no tenga bloqueo de LEG,

en caso de no recibir respuesta busca en el siguiente.

por cada consulta al Esquema Global el resultado Z ("a", SEG, Lista)

se agregará a la tabla LEL (Esquema Local, Dirección Sitio,

Status, Última actualización, Reporta) = Z (Lista), modificando

Última actualización por la fecha y hora actual y Reporta por

Esquema Local.

revisa que el Esquema Local LAR (Esquema Local) no tenga algún  
bloqueo en LAR (Status).

si tiene bloqueo ejecuta el programa RB

si no tiene bloqueo, toma el valor de AUX (Lista (Archivo, LAR

(Esquema Local)) y la agrega al final de la lista R (Descripción).]

manda el mensaje Y ("a", Sitio, R) a Emisor}

d{ mientras no exista un EOF

[ lee de la lista X (Lista (Archivo, Clave))

si X (Lista (Clave))

2{ se coloca en LEG (Esquema Global) = X (Lista (Archivo))

borra LEG (Esquema Global) de la lista.

manda el mensaje Y ("b") a Emisor}

3{ se coloca en LEG (Esquema Global) = X (Lista (Archivo))

LEG (Status) toma el valor de ST y LEG (Dirección Sitio) toma

el valor de DR

LEG (Última Actualización) toma la fecha y la hora del  
momento en que se hace el cambio

manda el mensaje Y ("b") a Emisor}

fin del si condicional]}

```

e{ mientras no exista un EOF
    [ lee de la lista X (Lista (Archivo, Clave))
        si X (Lista (Clave))
            2{ buscá LEL (Esquema Local) = X (Lista (Archivo)) y lo borra
                manda el mensaje Y ("b") a Emisor}
            3{ si no existe LEL (Esquema Local) = X (Lista (Archivo))
termina
                si existe LEL (Esquema Local) = X (Lista (Archivo)), LEL
                (Status) toma el valor de ST y LEL (Dirección Sitio) toma
                el valor de DR, LEL (Última Actualización) toma la fecha
                y la hora del momento en que se hace el cambio
                    manda el mensaje Y ("b") a Emisor}
                fin del si condicional]}
f{ mientras no exista un EOF
    [ lee de la lista X (Lista (Archivo, Clave))
        si X (Lista (Clave))
            2{ borra GAR (EL, X (Lista (Archivo))).}
        fin del si condicional]}
g{ lee de la lista X (Lista (Archivo, Clave))
    si X( Lista( Clave))
        1{ buscá LEG (Esquema Global) = X (Lista (Archivo)) y modifica
            LEG (Status) por 0}
        2{ buscá LEL (Esquema Local) = X (Lista (Archivo)) y modifica
            LEL (Status) por 0}
            manda mensaje Y ("b") a Emisor}
h{ lee de la lista X (Lista (Archivo, Clave))
    para cada X (Lista (Archivo)) crea LEG (X (Lista (Archivo))) = GEG (X (Lista
        (Archivo)))
        manda mensaje Y ("b") a Emisor}
fin del si condicional

```

Nota: al igual que en el esquema global, para cada uno de los mensajes que se manda, se ejecuta el procedimiento *vm* (verifica mensajes).

**ALGORITMO 7:** Edición del esquema local

Inicio

{

Pregunta la localidad del Esquema Global a trabajar y la guarda en SITIO

Despliega el siguiente menú:

- 1) Agregar
- 2) Borrar
- 3) Actualizar
- 4) Consultar

y guarda la respuesta en la variable CLAVE,

Despliega el siguiente menú:

- 1) Esquema Global
- 2) Esquema Local
- 3) Archivos

y guarda la respuesta en la variable ESQUEMA,

repite hasta leer un EOF o vacío

{

Pide el nombre de un archivo, Esquema Local o Esquema Global

AUX = AUX + El valor del nombre que acaba de leer

}

si el valor de la variable ESQUEMA

- 1 { manda un mensaje Y ("b", SITIO, (AUX, CLAVE))}
- 2 { manda un mensaje Y ("d", SITIO, (AUX, CLAVE))}
- 3 { manda un mensaje Y ("f", SITIO, (AUX, CLAVE))}

fin del si condicional,

recibe el mensaje Y ("a", SITIO, (AUX, CLAVE)) con el resultado de lo que se halla pedido

]