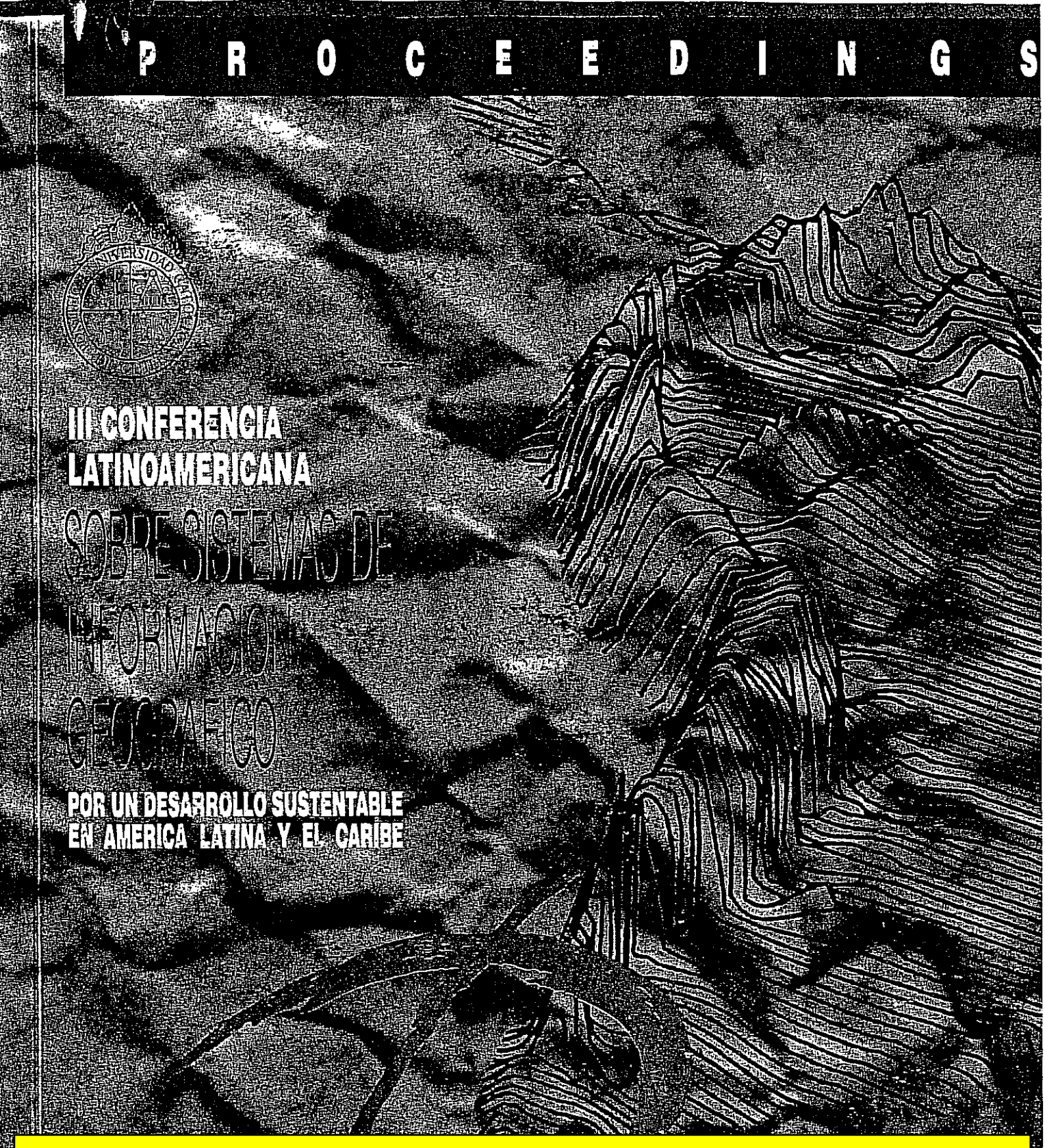
PROCEEDINGS



# III CONFERENCIA LATINOAMERICANA SOBRE SISTEMAS DE INFORMACION GEOGRAFICO

## POR UN DESARROLLO SUSTENTABLE EN AMERICA LATINA Y EL CARIBE



OCTUBRE 21 • 25, 1991
VIÑA DEL MAR - CHILE

# Requisitos de un sistema administrador de bases de datos para un SIG(*)

FRANK, Andrew U.

Departamento de Ingeniería Topográfica
Universidad de Maine
Orono, Maine 04469, E.U.A.

## RESUMEN

Los sistemas de información geográfica (SIG) almacenan grandes cantidades de datos que deben estar a disposición de muchos usuarios. Los sistemas administradores de bases de datos (DBMS = Database Management Systems) fueron diseñados para facilitar el almacenamiento y la recuperación de grandes colecciones de datos. Incluyen elementos para proteger y asegurar los datos, para imponerles consistencia una vez almacenados y para posibilitar que estén a disposición de muchos usuarios al mismo tiempo. Estos servicios son necesarios para los SIG, y por ende los SIG deben construirse mediante la utilización de sistemas administradores de bases de datos. Sin embargo, los SIG requieren una *performance* elevada y plantean requisitos muy especiales en materia de administración de bases de datos. Los DBMS diseñados para uso comercial no se adaptan bien a los SIG porque no pueden alojar datos espaciales y satisfacer la recuperación de la gráfica de los mapas. En este trabajo se presenta una visión general de la arquitectura de un DBMS especialmente adaptado al manejo de datos espaciales. Para cada nivel se indican técnicas específicas que resultan útiles a los DBMS para SIG, por ejemplo, para la administración de *buffers*, *clustering* de los datos y el acceso espacial. Se describen los esfuerzos efectuados para implementar el DBMS PANDA.

## INTRODUCCION

Los sistemas de información geográfica (SIG) deben almacenar grandes cantidades de datos y permitir el acceso a ellos cuando sea requerido. A partir de sus experiencias con las computadoras personales, los usuarios han aprendido a exigir respuestas casi instantáneas aun para pedidos relativamente complejos. Las soluciones tradicionales en las que los datos son almacenados en disco o en cinta magnética y deben ser buscados secuencialmente no pueden responder con bastante rapidez a las consultas de los usuarios y ya no son suficientes para adaptarse a los cambios frecuentes en las necesidades de los usuarios.

Se espera que un SIG moderno sea capaz de integrar datos correspondientes a distintos temas y provenientes de distintas fuentes. Se supone que la integración de múltiples conjuntos de datos, a menudo visualizados como múltiples niveles o estratos [*layers*] de datos, ha de producir un efecto sinérgico y rendirá mejor información para la toma de decisiones. El almacenamiento tradicional, orientado a los archivos, tampoco puede responder fácilmente a ese requerimiento.

Los sistemas de información geográfica consisten en un complejo de varias partes que interactúan. Para poder construir SIG computarizados necesitamos afrontar problemas de organización, de *software* y de *hardware*.

Debe señalarse que organizar la cooperación de grupos distintos para recolectar datos y compartir los resultados es una tarea particularmente difícil, para la que se cuenta con pocos lineamientos y reglas. Muchos proyectos fracasan, no por razones técnicas sino por falta de arreglos organizativos o debido a una escasa comprensión de las implicaciones sociales o económicas.

Los problemas de hardware son más fáciles de resolver, pues diversos fabricantes proveen los componentes para el almacenaje y procesamiento de cantidades muy grandes de datos. Los precios son cada vez más razonables y la tendencia general es hacia "el *hardware* de costo cero" (DANGERMOND y MOREHOUSE, 1987). Por el

---

(*) Traducción al español del artículo "Requirements for a Database Management System for a GIS", publicado en *Photogrammetric Engineering and Remote Sensing*, Vol. 54, Nº 11, November 1988, pp. 1557-1564. [Traductor: Fernando LIDA GARCIA, Buenos Aires, República Argentina.]

contrario, el software para SIG es mucho más difícil de construir que lo que muchos habían creído antes. El sistema de *software* para administrar datos de SIG debe contener un módulo que provea funcionalidad de sistema administrador de base de datos. El presente trabajo se refiere primordialmente a este componente de *software* y a los requisitos que le imponen las aplicaciones de SIG.

Los sistemas administradores de bases de datos (DBMS) son herramientas apropiadas para los SIG. El acceso rápido a los datos espaciales a partir de una gran colección de datos es difícil de lograr. Muchos SIG actuales almacenan datos bajo la forma de una colección de hojas de mapas (o particiones espaciales similares) que son entonces manejadas como unidades. Esto requiere que todos los usuarios comprendan su estructura y dificulta el acceso por direcciones postales u otros conceptos lógicos, por ejemplo. Para lograr la deseada base de datos "inconsútil" [es decir, sin "costuras"] en la que los objetos (es decir, los rasgos cartográficos) no son divididos arbitrariamente por los bordes de los mapas y en la que los usuarios pueden desplazarse libremente o [acercarse/alejarse] en *zoom* sobre el mapa, se requieren métodos y optimizaciones especiales.

El *software* de DBMS brinda los servicios necesarios para integrar y proteger los datos, pero el DBMS convencional no rinde la *performance* y no puede recuperar datos para mapas con suficiente rapidez para permitir el trabajo interactivo. Ni tampoco todos los paquetes de *software* para SIG que se comercializan en la actualidad contienen un DBMS, ni incluyen todos los servicios necesarios para la protección de los datos.

En el presente trabajo damos un detalle de estos servicios necesarios de DBMS y mostramos una perspectiva arquitectónica de cómo interactúan. Utilizamos modernos conceptos de ingeniería de *software* para sistematizar el análisis. Dedicamos especial atención a la integración de los DBMS con otro *software* específicamente escrito para el procesamiento de datos espaciales. Se hace hincapié en las funciones de almacenaje y recuperación de los datos, incluyendo la protección de los datos de un SIG. Por razones de espacio se han excluido, y se tratan en otro trabajo (EGENHOFER y FRANK, 1988a), problemas igualmente importantes atinentes a una adecuada modelación de la realidad y al apoyo al modelado de datos necesarios para los SIG. Por eso se ha limitado deliberadamente la discusión de métodos de acceso y, en particular, de lenguajes de consulta [*query languages*].

Muchas de las ideas aquí informadas se basan en la experiencia con el DBMS PANDA (FRANK, 1982a, 1984b, 1986a; EGENHOFER y FRANK, 1987a). Identificamos aquí los métodos implementados con buen éxito e incluimos una crítica de aquellos que no han funcionado tan bien y que en el futuro serán reemplazados.

## LOS SISTEMAS DE INFORMACION ESPACIAL

La utilización de computadoras para el procesamiento "por lotes" [*"batch" processing*], por el cual se recolectan todos los datos que hay que ingresar, en tanto que la salida con el resultado es proporcionada después, ha sido mayormente reemplazada por sistemas interactivos de información, en los que el sistema mantiene una colección de datos que los usuarios consultan a medida que necesitan la información.

En términos generales, un sistema de información contiene una imagen o modelo de la realidad, del que podemos hacer uso para tomar decisiones sin tener que reinvestigar los hechos cada vez. Esto resulta sumamente importante en todas las situaciones en las que la recolección de datos es costosa, engorrosa o lenta, y constituye una de las principales fuerzas que impulsan a los SIG: se espera lograr ahorros sustanciales (NATIONAL ACADEMY OF SCIENCE, 1980) compartiendo el costo de la recolección de datos, y al mismo tiempo un mejor uso de estos y una mayor calidad en la información de salida.

Los SIG manejan datos relacionados con la localización en el espacio del mundo real, que aquí denominamos **datos espaciales**. Muchas operaciones gubernamentales en todos los niveles, así como la planificación y la investigación, explotan datos que poseen una componente espacial. Tales sistemas suelen designarse con diversos otros nombres, vgr., sistemas de información territorial, sistemas AM/FM, catastros multipropósitos. Nos concentraremos en los aspectos generales de los sistemas que tratan datos espaciales, denominados **sistemas de información espacial**, sin tener en cuenta las diferencias entre los sistemas diseñados para tareas específicas.

Dedicaremos especial atención a los sistemas que almacenan datos con una referencia exacta a la ubicación y que describen la geometría utilizando puntos y vectores. No se pretende con esto excluir los sistemas de otros

tipos, puesto que el uso de operaciones *raster* para ciertas tareas ofrece indudables ventajas, pero parecen plantear requerimientos sustancialmente distintos en materia de almacenamiento de datos y merecen ser analizadas por separado.

Un SIG es un modelo de la realidad y no solo un depósito de datos cartográficos necesarios para dibujar mapas (FRANK, 1984a). En consecuencia, adquieren importancia los métodos para representar aspectos complejos de la realidad en un sistema de computación. Solo si la estructura de la realidad está debidamente modelada en los datos almacenados, podemos aspirar a que la combinación de múltiples fuentes de datos y la extracción de información compleja produzcan resultados significativos. En tales situaciones encontramos que existen relaciones entre los elementos de los datos; p. ej., un edificio se halla al mismo tiempo relacionado con el lote sobre el que está construido, con la calle donde está situado y con las personas que viven en él. Para almacenar y recuperar los datos, hace falta un método que utilice y conserve estas múltiples relaciones.

## LOS SISTEMAS ADMINISTRADORES DE BASES DE DATOS

Los datos reunidos en una base de datos son valiosos porque se requiere mucho esfuerzo para recolectar e ingresar los datos en el sistema y para mantenerlos actualizados. Los datos almacenados deben mantenerse disponibles por un lapso prolongado para justificar los costos de su ingreso. Durante la vida útil de los datos es probable que haya cambios nuevos e imprevistos en las aplicaciones. Los programas orientados a archivos tienen la tendencia a requerir cambios en todos los programas de acceso a un archivo si se hace necesaria una modificación en ese archivo. Los DBMS separan el procesamiento de los datos de su almacenaje y limitan los cambios a los programas directamente afectados.

Hacer que los mismos datos resulten disponibles para aplicaciones diversas e integrar datos provenientes de distintas fuentes es difícil en un sistema orientado a archivos, porque ello crea más dependencias entre los programas y el archivo y por ende hace más costosa la adaptación de los programas a los requerimientos cambiantes. En tales circunstancias, la estructura tradicional de archivo simple, ideada para facilitar un determinado programa de aplicación, deja de ser adecuada.

Un DBMS debe cumplir las siguientes funciones:

Almacenaje y recuperación de los datos; selección de los datos basada en múltiples claves de acceso (p. ej., el nombre de una persona, la dirección de un edificio en una calle).
Acceso estandarizado a los datos y separación de las funciones de almacenaje y recuperación respecto de los programas que utilizan los datos (esto hace que la base de datos sea independiente de los programas de aplicaciones, de suerte que los cambios en un aspecto no requieran necesariamente cambios en el otro).
Interfaz entre la base de datos y los programas de aplicación, basada en una descripción lógica de los datos (los detalles de la estructura física de almacenaje deberían ser transparentes a las aplicaciones).
Hacer que las funciones de acceso en las aplicaciones sean independientes de la estructura física de almacenaje, de modo que las adaptaciones a crecientes necesidades de almacenaje no influyan en los programas de aplicaciones.
Permitir el acceso simultáneo a los datos a varios usuarios.
Asegurar la definición de restricciones de consistencia de los datos que se aplicarán automáticamente. Las restricciones de consistencia son reglas que deben aplicarse a todos los datos almacenados y constituyen una técnica excelente para reducir el número de errores en una gran colección de datos.

El acceso a los datos debería ser posible tanto desde un lenguaje de alto nivel como desde un lenguaje de consulta amigable para con el usuario. El nivel de integración del lenguaje para manipular la base de datos con el lenguaje de programación utilizado influye sobre los esfuerzos necesarios para desarrollar y modificar los programas de aplicaciones. Un lenguaje de consulta autónomo [*free-standing*] es útil para usuarios eventuales que deseen recuperar datos de la base para responder a preguntas *ad hoc* sin ninguna programación formal. Esto hará que la base de datos sea utilizable para preguntas del tipo "una de cada clase", que suelen ser planteadas cuando se tratan situaciones anormales o en aplicaciones de planificación. Un sistema administrador

de bases de datos es entonces un método de encapsular los datos valiosos para hacerlos disponibles a una multitud de usuarios al tiempo que se protegen los datos (Figura 1).

En una etapa temprana de la historia del procesamiento de datos, los programadores se percataron de las necesidades similares de distintas aplicaciones en materia de almacenaje y recuperación de datos. En lugar de volver a escribir procedimientos para estas funciones en cada aplicación, se intentó escribir un programa aplicable en general para suministrar tales servicios. Nació así la idea de un sistema (generalizado) para la administración de bases de datos (CODASYL, 1962, 1971).

## SISTEMAS ADMINISTRADORES DE BASES DE DATOS ESPACIALES

Los DBMS comerciales estándares, como los que se utilizan para llevar datos sobre personal o sobre clientes, etc., están diseñados para pautas de utilización distintas de las que se encuentran en las aplicaciones de ingeniería y científicas. Los usuarios comerciales requieren el número telefónico o la dirección de una persona o bien transfieren cierta suma de dinero de una cuenta a otra. En un sistema de información espacial los usuarios piden que en la pantalla aparezca un croquis tipo mapa que muestre, vgr., un edificio con sus lindes, los edificios linderos y posiblemente las líneas de servicios públicos con los que está conectado. Los entes [*entities*] de un sistema de información espacial suelen estar conectados lógicamente a muchos otros entes más que en un sistema comercial. Además, se hallan vinculados por relaciones espaciales tales como "vecindad" o "cercanía", que no se usan en las aplicaciones comerciales.

La tarea crucial en un sistema de información espacial consiste en la recuperación de un conjunto de entes necesarios para dibujar un mapa pequeño en la pantalla. Después de un recuento de los entes de tales dibujos para diversas aplicaciones, estimamos que deben recuperarse entre 2000 y 5000 entes (puntos, líneas, símbolos, etc.) de las colecciones de datos, para poder producir ese dibujo. Las pantallas con muchos menos datos que esos parecen vacías y no ofrecen información suficiente sobre una zona, en tanto que las pantallas con más datos resultan demasiado atestadas y difíciles de leer. En una operación interactiva la respuesta debe demorar menos de medio minuto, pues si no los operadores se ponen a hacer otras tareas, pierden concentración y la productividad sufre.

Los DBMS orientados a lo comercial no están diseñados para una recuperación rápida de tantos entes vinculados espacialmente. El *software* actual de SIG, o bien está basado en estructuras de archivo específicas, al menos en lo que atañe a los datos espaciales, y carece de muchas de las otras ventajas de los DBMS, o bien reparte los datos en conjuntos de datos más pequeños que son almacenados entonces como bases de datos separadas. Esto resulta aceptable en los sistemas destinados al mantenimiento de mapas, con los que se producen mapas actualizados sobre papel, pero no es útil cuando los usuarios finales preguntan cosas como "¿Cómo está conectado el edificio de la calle Tal número Cual con la red de agua corriente?" A estos usuarios no se los puede preocupar con los límites de las hojas del mapa, pues ello los distraería de su tarea principal, y tienen que poder escoger las zonas mediante criterios lógicos así como por *zoom* o deambulando gráficamente en una base de datos inconsútil.

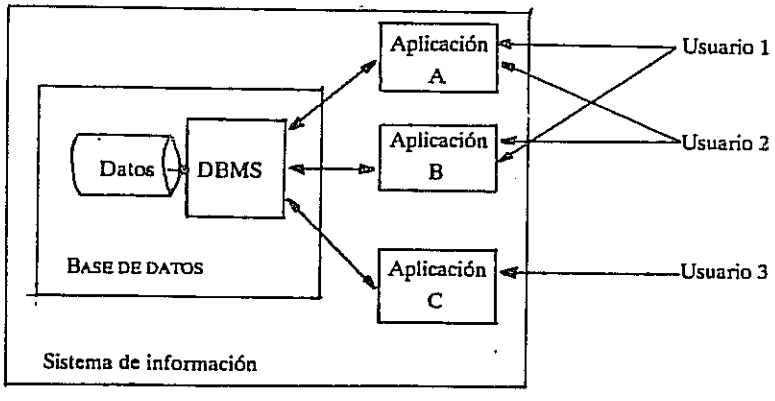



FIG. 1. Sistema administrador de una base de datos

La investigación de bases de datos halló que una cantidad de aplicaciones de ingeniería y científicas (p. ej., CAD [Computer Aided Design = diseño asistido por computadora], procesamiento de datos médicos) necesitan todas ellas las funciones básicas de los DBMS tales como las ofrecen los sistemas comerciales, pero guardan cierta semejanza en requerimientos que hacen inadecuados a los DBMS estándares (PLOUFFE *et al.*, 1984). La investigación en estas bases de datos llamadas "no estándares" comenzó ya en los primeros años del decenio 1981-1990 (FRANK, 1981; HAERDER y REUTER, 1982; SCHEK y LUM, 1983) y prosigue en la actualidad (MANOLA *et al.*, 1987). En las siguientes secciones se da primero un marco teórico para tales DBMS no estándares y luego se mencionan algunas técnicas específicas que han resultado de utilidad.

## DBMS ESPACIALES

Para un DBMS espacial, proponemos una arquitectura de estratos [o niveles: *layered architecture*]. Consistirá en una jerarquía de módulos, cada uno de los cuales proporcionará ciertos tipos de servicios o funciones al nivel inmediato superior. El nivel más bajo(**) está directamente vinculado con los servicios suministrados por el sistema operativo, en tanto que el más alto proporciona servicios al usuario del SIG (Figura 2).

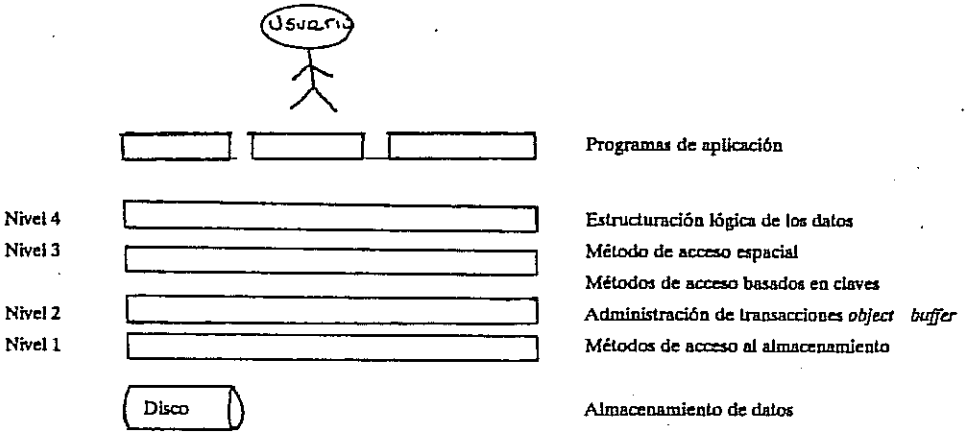



FIG. 2. Los niveles de un sistema administrador de bases de datos espaciales

El nivel 1 **almacena los datos** y utiliza el sistema operativo para tener acceso al sistema de archivos. Este nivel se ocupa sobre todo de mejorar el rendimiento del acceso a los datos. Los servicios ofrecidos son operaciones de *almacenar* y *recuperar* los elementos de datos (registros) mediante el empleo de identificadores internos de registros.

El siguiente nivel ofrece esencialmente las mismas operaciones, pero las hace más seguras: garantiza que las modificaciones de la base de datos no sean afectadas por pérdidas o por la interferencia de otros usuarios.

El tercer nivel aporta distintos tipos de métodos de acceso, p. ej., acceso a los datos basado en un valor (vgr., el número en una calle) o en una ubicación espacial.

El cuarto nivel ofrece **una herramienta de estructuración lógica de los datos** y manipulaciones basadas en ese esquema lógico. Estos servicios son ofrecidos luego como una extensión a un lenguaje de programación de alto nivel o a un lenguaje de consulta independiente.

---

(**) En este contexto, *nivel* denota una colección de módulos de *software* (programas) y no debe confundírselo con la acepción cartográfica del término que designa un estrato en un sistema de mapas superpuestos.

## REQUISITOS DEL NIVEL DE ALMACENAMIENTO DE LOS DATOS

La finalidad principal del nivel 1 es la de establecer una interfaz con el sistema operativo y aumentar la velocidad de almacenaje y recuperación. El rendimiento de un DBMS se percibe sobre todo como tiempo de respuesta a las consultas o en general por el tiempo que lleva recuperar los datos almacenados en el sistema. La experiencia indica que el tiempo de recuperación de los datos está determinado por el número de accesos físicos al disco y que influye poco el tiempo de procesamiento de los datos una vez que son transferidos a la memoria de trabajo.

Es típico que los sistemas de información espacial requieran grandes bases de datos almacenadas de modo permanente en dispositivos de almacenamiento masivo (discos). Para la ejecución de una operación sencilla, generalmente solo hace falta tener acceso a porciones pequeñas de esas grandes colecciones. Cada acceso a los datos de un disco requiere solo unos 30 milisegundos y resulta casi independiente de la cantidad de datos leídos. La tecnología de discos ha mejorado considerablemente en el último decenio, con lo cual se han conseguido una mucho mayor capacidad de almacenaje y precios más bajos. El tiempo de acceso, empero, se ha mantenido casi constante; se lo debe comparar con el tiempo necesario para el procesamiento de los datos en la memoria central, que es de menos de una diezmillonésima de segundo (0,1 microsegundo), es decir, unas 300 000 veces más veloz.

Un requisito fundamental para este nivel se origina en el plazo máximo que podemos permitir para que se dibuje un mapa en la pantalla. Si cada uno de los 2000 a 5000 registros de datos necesarios para tener "una pantalla llena de datos" se trajera del disco por acceso independiente (que requeriría por lo menos 30 mseg), el usuario tendría que esperar de uno a tres minutos a que estuviera dibujado el mapa, lo cual es absolutamente inaceptable en un ambiente interactivo. Por lo tanto, el nivel 1 debe reducir la cantidad de accesos físicos necesarios para recuperar los registros que el mapa requiere. Se conocen para ello dos técnicas básicas: el *clustering* de datos y la administración de *buffer*. Nosotros utilizamos los dos. Debe observarse que los sistemas operativos a veces emplean métodos parecidos para mejorar la *performance* de las operaciones de disco y que estas técnicas de optimización pueden interferir en los procedimientos que utiliza una base de datos. Puede ser ventajoso, entonces, utilizar operaciones de acceso al disco, de bajo nivel, y dejar de lado los servicios del sistema operativo para el *buffering*.

### *CLUSTERING*

El método primario para reducir la cantidad de accesos al disco consiste en formar *clusters* de objetos vinculados lógica o espacialmente sobre páginas del disco (Figura 3). Un acceso físico al disco aporta una porción mayor de datos, denominada generalmente una página de disco (que puede tener entre 512 y algunos miles de bytes). Si podemos ordenar nuestros registros sobre las páginas del disco de tal suerte que cada página contenga varios registros de datos necesarios para dibujar el mapa, se reducirá mucho la cantidad de accesos al disco para transferir los datos a la memoria de trabajo, que tanto tiempo consumen. En 20 segundos se podrían leer unas 200 páginas; si cada una contuviera 25 de estos registros necesarios (y posiblemente algunos otros), se habrán leído todos los datos requeridos.

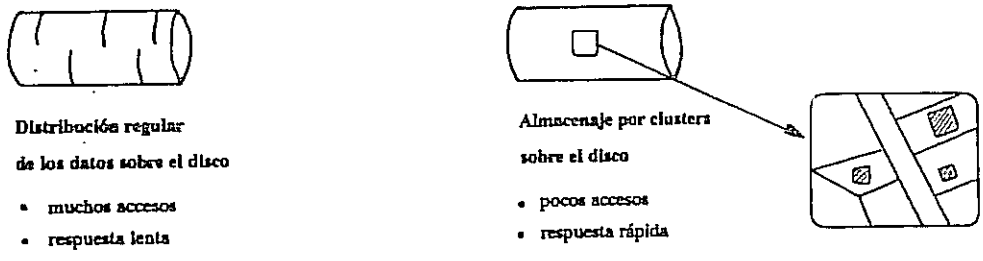



FIG. 3. *Clustering* espacial

Este procedimiento es factible en los casos en que es posible predecir razonablemente qué datos se utilizarán juntos. Estos son entonces almacenados juntos formando un cluster **físico** sobre el disco (SALTON y WONG, 1978). Afortunadamente para la recuperación de datos espaciales para dibujar mapas, tales predicciones son relativamente fáciles de hacer, basándose en la relación de **vecindad**. Los datos que recuperamos para un mapa están situados dentro de una determinada zona; si recuperamos un elemento de dato de un objeto, hay buenas probabilidades de que se necesiten seguidamente los datos correspondientes a otros objetos vecinos. También sería conveniente que los datos de distinta índole (vgr., casas, caminos y ríos) pudieran almacenarse en la misma "página" y que no requiriesen necesariamente accesos físicos distintos. Si tales datos son agrupados en *clusters* y recuperados con un solo acceso físico, podemos lograr el objetivo de recuperar un mapa entero en un breve lapso, lo cual posibilita el trabajo interactivo.

Un sistema de base de datos para datos espaciales debe al menos permitir el *clustering* físico de los datos (lo típico es que el programador de rutinas de bajo nivel para el almacenaje de datos [*low-level data storage routines*] debe poder disponer de una orden como STORE NEAR X [almacenar cerca de x], siendo x un registro ya almacenado). Este requisito excluye muchos de los más simples DBMS relacionales, en los que el almacenaje físico de los datos está dirigido por la clave primaria y no puede ser influido por el usuario. No es necesario ni deseable que el *clustering* físico sea visible en el nivel de la interfaz con el usuario. En cambio, debería ser usado internamente para un acceso espacial rápido y no debería preocupar al usuario.

*BUFFERING*

Muchos programas tienen acceso por localización (es decir, los mismos elementos de datos son utilizados repetidamente en un lapso breve). Si estos elementos de datos pueden ser mantenidos en un *buffer*, se puede reducir la cantidad de accesos físicos a un dispositivo de almacenaje masivo más lento. Para cada elemento de dato, solo el primer acceso utiliza el dispositivo lento, y todos los pedidos siguientes se satisfacen empleando los datos del *buffer*. Esta es la estrategia empleada generalmente en las computadoras (vgr., memoria virtual, *cache*). Un DBMS contiene generalmente un solo juego de *buffers* para las páginas traídas del disco, a fin de aprovechar un *clustering* físico de datos en el dispositivo de almacenaje.

Es típico que los programas para el tratamiento de datos espaciales posibiliten gran cantidad de accesos a los datos por su localización, pero eso supone el empleo de conjuntos de trabajo muy numerosos (vgr., todos los registros necesarios para dibujar un mapa). Los usuarios tienden a trabajar en una misma zona geográfica durante varias interacciones, antes de que decidan pedir otro mapa de base. Muy a menudo piden primero un mapa de conjunto y luego se acercan con el *zoom* a un detalle de interés, para empezar a trabajar sobre este. En nuestro DBMS experimental, el PANDA, decidimos incluir un segundo nivel de *buffers* para registros sencillos. Ahora hemos establecido el tamaño del *buffer* en 5000 registros, lo que nos permite guardar en él el dibujo de un mapa completo. No hacen falta accesos físicos adicionales para redibujar un mapa o hacer un *zoom*, y por lo tanto estas operaciones se efectúan muy velozmente.

## LA PROTECCION DE LOS DATOS

Las grandes colecciones de datos geográficos o administrativos, igual que otras colecciones de datos, son bienes valiosos y deben ser protegidos. Aun cuando la información se guarde en paralelo bajo otras formas tradicionales (registros, mapas), el costo de transferirla a una forma legible por una máquina es considerable. Además, la información deducible de los datos puede resultar de enorme valor económico para alguien que sepa cómo aprovecharla (la información es poder). Aun cuando este aspecto sea menos pronunciado en las bases de datos espaciales que en las de uso comercial, los datos recolectados deben preservarse de un uso no autorizado. Por ejemplo, la mayoría de las oficinas nacionales de estadística recopilan datos que solo deben ser publicados en conjuntos estadísticos que no revelen los datos de personas individualizables.

Se acostumbra a distinguir los siguientes cuatro tipos de amenazas:

las pérdidas debidas a errores operativos o fallas de funcionamiento de los equipos o programas, o a manipulaciones erróneas por los usuarios autorizados;

el acceso de usuarios no autorizados o los daños que puedan causar;

la introducción de datos falsos por personal autorizado mediante el empleo de procedimientos correctos; y

la corrupción de los datos por muchos usuarios al mismo tiempo (actualizaciones concurrentes).

Los datos numéricos son muy complicados de proteger, y es típico que los organismos posean escasa experiencia en la protección de bases de datos. Los seres humanos no pueden percibir directamente la presencia (ni la ausencia) de datos: hacen falta equipo de computación y programas para evaluar una colección de datos. Tampoco puede evaluarse fácilmente la calidad de los datos, ya que la sola presencia de estos no garantiza que sean útiles, correctos, actualizados o completos.

Si bien el nivel 2 no agrega funcionalidad sustancialmente nueva, sí incrementará la seguridad de las operaciones. Sin embargo, la mayoría de las funciones de este nivel ocasionan una disminución del rendimiento. Tal es el precio que pagamos por la seguridad lograda.

Para poder determinar las medidas de seguridad apropiadas, debemos evaluar los posibles perjuicios resultantes de la pérdida de datos (vgr., el costo de reingresar los datos, el costo asociado a la interrupción de las operaciones) y ponderarlos con los costos de las operaciones tendientes a impedir tales pérdidas. La mayoría de las operaciones comerciales asignan gran importancia a la continuidad operativa y a la confidencialidad; por consiguiente, se han ideado programas muy complicados pero seguros. En las aplicaciones de SIG puede resultar aceptable que los datos no estén disponibles durante algunas horas o inclusive por un día, siempre que tales interrupciones no sean frecuentes. Por ende, pueden resultar aceptables niveles más bajos de seguridad.

## PROTECCION CONTRA PERDIDAS

La protección de los datos contra pérdidas por fallas de funcionamiento de los equipos o programas es absolutamente necesaria. En todas las situaciones en las que se efectúan modificaciones no del todo infrecuentes, el mecanismo de protección debería incluir las actualizaciones de la base de datos. No deben perderse una modificación introducida por el usuario ni la confirmación recibida de la base de datos. Se suele distinguir dos tipos de problemas:

Interrupción del programa administrador de la base de datos, debida a errores operativos, a problemas en el sistema operativo, o a falla del equipo. Tales interrupciones suelen ocurrir con bastante frecuencia en la mayoría de las instalaciones (de una por día a una por semana). Durante esos episodios se pierden todos los contenidos de la memoria principal, y por lo tanto resulta necesario escribir los datos modificados en el almacenamiento masivo permanente antes de confirmar una actualización.

Pérdida de los medios de almacenamiento, debida nuevamente a errores operativos o fallas del *hardware* (los llamados "aterrizajes de las cabezas" [*head crashes*]). Estos problemas suelen ser raros (una vez por año) y resultan aceptables procedimientos de recuperación más lentos.

Los servicios ofrecidos en los DBMS comerciales son por lo general suficientes. Empero, muchos de los sistemas utilizados en las microcomputadoras y las PC, así como los sistemas especializados en datos geométricos y geográficos, muy a menudo no protegen debidamente los datos.

## PRESERVAR LOS DATOS DE UN USO NO AUTORIZADO

Algunas bases de datos contienen información confidencial, que es necesario proteger de un uso no autorizado. Debe negarse el acceso a las personas que tienen prohibido el uso de sus contenidos. Es preciso prever medidas para excluir a ciertos usuarios autorizados de que tengan acceso a ciertas clases de datos. Por ejemplo, el personal de las compañías telefónicas no tiene por qué conocer el monto de las hipotecas que gravan un inmueble. Algunos usuarios serán autorizados solo a leer datos, pero no a modificarlos. Se requiere una reglamentación precisa de la responsabilidad por la calidad de los datos, y solo a las personas con responsabilidad específica se les permitirá completar modificaciones. Un problema propio de las bases de datos

estadísticos (y muchas bases de datos espaciales van a ser utilizadas como ellas) consiste en limitar el acceso de los usuarios autorizados solo a la información general e impedirles el acceso directo a datos individuales. La mayoría de los sistemas conocidos han demostrado ser protección insuficiente contra un ataque decidido, y las medidas protectoras propuestas resultan sumamente complicadas (DENNING y SCHLORER, 1980).

En general, podemos suponer que el sistema operativo contiene procedimientos para excluir a los usuarios no autorizados, así como para identificar a los autorizados. Sin embargo, solo el DBMS puede limitar el acceso a determinadas partes de la base de datos accesibles a ciertos usuarios.

## LA ADMINISTRACION DE TRANSACCIONES

Esto se refiere a las salvaguardas que aseguran los datos contra pérdidas accidentales causadas por usuarios autorizados. Tenemos que evitar el ingreso erróneo de datos por los usuarios, pero también considerar toda clase de desperfectos del *hardware*. Hemos de habérnoslas con cortes de energía, errores de los programas, y multiplicidad de usuarios simultáneos. El objetivo es impedir la pérdida de datos por usuarios autorizados y la introducción de datos nuevos falsos. El control de los usuarios autorizados se centra primordialmente en los cambios que se introducen en la base de datos, suponiendo que los accesos para lectura solamente no modifican la base de datos y por ende no pueden introducir errores.

Se acostumbra a agrupar varias modificaciones en la base de datos vinculadas lógicamente, de modo que constituyan una transacción. La administración de transacciones en una DBMS se ocupa de:

la atomización de las transacciones,
la concurrencia de transacciones por múltiples usuarios,
la integridad de la base de datos después de la transacción, y
la durabilidad de las transacciones;

estos cuatro conceptos forman la sigla mnemónica [en inglés] ACID [ácido] (HAERDER, 1985).

### LA ATOMIZACION DE LAS TRANSACCIONES

Desde el punto de vista de la base de datos, todas las modificaciones incluidas en una transacción se hallan vinculadas lógicamente (esa es la idea expresada al "empaquetarlas" todas juntas en una transacción) y, por ende, o bien se las ejecuta todas juntas o no se ejecuta ninguna. Una base de datos no debería nunca contener resultados parciales de una transacción, aun si esta fuera interrumpida por un repentino desperfecto del *hardware* (o por un corte de energía). La atomización posibilita que los programas se reinicien y reanuden su labor después de una interrupción súbita, puesto que la base de datos se halla siempre en un estado definido (esto es: al final de la última transacción ejecutada), y el programa no necesita nunca ocuparse de la ejecución parcial de la transacción.

### LA CONCURRENCIA

Varios usuarios simultáneamente pueden tener acceso a la misma porción de datos y modificarlos. Estas acciones concurrentes pueden contraponerse y deben ser coordinadas. En un DBMS se requiere por lo general que los usuarios concurrentes vean solo los efectos de las transacciones completadas y no puedan observar ningún resultado parcial de las transacciones que realizan los otros usuarios. Además, el DBMS debe verificar que las acciones de un usuario no invaliden los cambios introducidos por otro usuario. Es típico que los sistemas actualmente comercializados para la administración de datos espaciales no contengan la prevención automática de conflictos entre los usuarios, sino que tienden a confiar en mecanismos organizativos para asegurar que un solo usuario por vez trabaje con un archivo.

### LA INTEGRIDAD

Una colección de datos puede destruirse fácilmente mediante la incorporación de datos "equivocados". Los usuarios no aceptarán una base de datos de la que obtienen frecuentemente información errónea o

contradictoria. Lamentablemente, no existen procedimientos fáciles para que un programa verifique que los datos (o los cambios) introducidos son correctos (esto es, que describen correctamente la realidad). Lo más que podemos lograr es la inclusión en el DBMS de reglas que comprueben que los datos nuevos (o modificados) no contradigan otros datos ya almacenados en la base. Decimos que una base de datos es consistente cuando está libre de tales contradicciones.

LA DURABILIDAD

Una transacción que ha sido confirmada una vez no debería perderse nunca por desperfectos del *hardware* o por otros inconvenientes. Un DBMS debería contar con mecanismos para llevar un diario [o registro de novedades: *journal*] de todas las modificaciones anunciadas, de modo que una copia del archivo, producida regularmente, pueda mantenerse actualizada hasta la última versión, en caso de que se perdieran los datos corrientes. Esto es parecido, pero más complejo, que el acostumbrado método de "dos generaciones" que se utiliza para asegurar los archivos en un sistema de procesamiento por lotes.

Los métodos de la administración de transacciones son bien conocidos, pero lamentablemente reducen el rendimiento, ya que alargan el tiempo de procesamiento de las transacciones y pueden llegar a reducir casi a la mitad el rendimiento durante las actualizaciones. Por este motivo no suelen ser incluidos en los sistemas de información espacial disponibles en el mercado. Al comparar sistemas (en un *benchmark*, por ejemplo), debe especificarse siempre qué tipo de administración de transacciones se utiliza y qué niveles de protección se ofrecen para evitar pérdidas e inconsistencias entre los datos.

En el DBMS experimental PANDA, hemos incluido un gran *buffer* objetal, que puede contener todos los objetos de datos a los que tiene acceso un usuario durante una transacción. Con este *buffer*, podemos preparar dentro de él todas las modificaciones de la base de datos. Los archivos de datos no son afectados antes de que se haya completado la transacción. Luego todos los objetos modificados se escriben en el disco al mismo tiempo. Este es un esquema relativamente sencillo que puede ampliarse para incluir el registro de transacciones [*transaction logging*], a modo de libro de novedades [o diario] con todos los cambios que pueden (que deben) escribirse antes de que se consume alguna actualización de los archivos de la base de datos.

El uso de la administración de transacciones para lograr diversos objetivos que compiten entre sí limita la flexibilidad de lo que puede definirse como transacción. Si las transacciones sirven como unidades de durabilidad, deben ser breves porque no es aceptable perder mucho trabajo. Análogamente, las transacciones usadas para controlar la concurrencia deben ser rápidas, porque no se puede tolerar que otros usuarios estén impedidos por lapsos prolongados de modificar los datos. Por otra parte, ciertas operaciones de un SIG son demasiado complicadas para poderse cumplir en una transacción breve. En algunas aplicaciones, una transacción en el nivel de usuario puede ser muy complicada y requerir largo tiempo para completarse (vgr., la subdivisión de un predio). Tiene que ser posible registrar componentes y resultados parciales, aun cuando la consistencia general solo pueda verificarse al terminar. Se propone componer tales "transacciones largas" a partir de transacciones cortas (comunes) y hacer que la base de datos contenga reglas especiales que controlen la terminación de una transacción prolongada. Este es actualmente un campo de investigación en materia de bases de datos y aún no existen implementaciones disponibles.

## METODOS DE ACCESO BASADOS EN LA LOCALIZACION ESPACIAL

Los niveles considerados hasta ahora prevén el almacenaje y la recuperación de los datos mediante un identificador interno (clave de la base de datos [*database key*]) que es asignado a un registro de datos cuando se lo almacena por primera vez. Se considera que el registro en sí consiste en un número fijo de *bytes* que se almacenan y son luego reproducidos. Para las aplicaciones no usuales, tiene especial importancia que el sistema de almacenamiento no dependa de la interpretación de los datos almacenados, porque esto limitaría la flexibilidad del ordenamiento de la información en un registro y causaría dependencias indeseables entre los distintos niveles del sistema y la definición de los tipos de registro propios de la aplicación. El subsistema de la base de datos de una aplicación más grande debe poder manejar registros de datos arbitrariamente estructurados sin dependencia respecto de su estructura interna. Las estructuras de los registros de datos construidos a partir

de los tipos de datos básicos (esto es, números enteros, números reales y cadenas [*strings*]) no son suficientes siquiera para las aplicaciones comerciales, según lo ejemplifican diversas extensiones que se ofrecen (vgr., tipos de datos para dinero, fecha, etcétera).

Los DBMS estándares incluyen métodos para tener acceso a los elementos de datos basados en valores claves u otras construcciones del modelo de datos:

> A menudo los usuarios necesitan tener acceso a registros de datos basados en el valor de uno o más campos de datos (campos claves). En el caso más simple, esto requiere que los valores almacenados correspondientes a esos campos sean únicos, de suerte que los valores dados identifiquen inequívocamente el registro deseado (esto será una restricción de la consistencia de la base de datos). En general resulta necesario poder definir más de una clave para un registro, de modo que este pueda ser hallado usando cualquiera de ambos valores (por ejemplo, el registro de una persona debe poder encontrarse a partir de su número de documento o de su nombre).
>
> El modelo de datos posibilita la agrupación de objetos en unidades de orden superior. Tales combinaciones crean caminos de acceso, pues los usuarios pueden ir de un objeto a los grupos en los que está contenido o a los objetos de que consta. Por ejemplo, todos los edificios de una calle deben ser accesibles una vez que esta ha sido hallada.

Gran cantidad de los datos de un sistema de información espacial están relacionados con objetos en el espacio, objetos cuya ubicación y extensión son conocidas. Muchas aplicaciones requieren acceso a los datos basado en la localización (BURTON, 1978). Esto es obvio para todas las recuperaciones asociadas con la producción de mapas (todos los objetos dentro del marco del mapa deben ser recuperados y luego vertidos en forma gráfica). Se necesitan accesos similares para las operaciones internas destinadas a verificar los datos geométricos entrantes y a las manipulaciones geométricas de los datos almacenados. El acceso espacial a los datos está basado en una consulta de la forma: *Retrieve all <object-type> within <area>* [recuperar todos los tipos de objeto dentro de la zona], siendo *<area>* la descripción de la zona de interés.

Para conseguir una función general eficiente, parece apropiado reducir *<area>* a una caja rectangular paralela al sistema de coordenadas (Figura 4). Análogamente, la ubicación y extensión de cada objeto se registran como una caja rectangular (el mínimo rectángulo circunscripto: Figura 5). Esta parece ser una solución general y sencilla desde el punto de vista computacional.

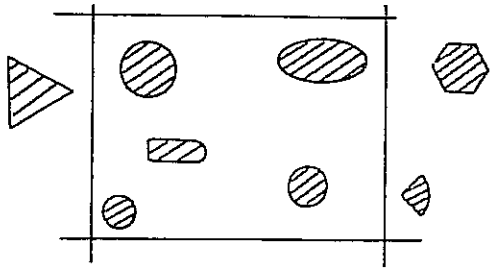


FIG. 4. Ventana de consulta

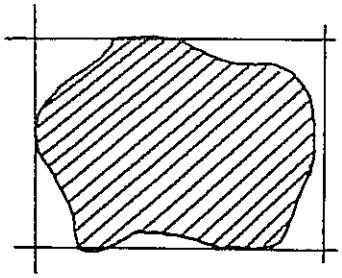


FIG. 5. Objeto con caja
(rectángulo circunscripto mínimo

Los pedidos más específicos se tratan luego en dos pasos. En el primero, todos los datos dentro de la envoltura rectangular de la zona son recuperados (a partir de una comparación del rectángulo de consulta con la caja del objeto). El proceso exacto de selección, más costoso, se efectúa en un segundo paso, pero solo sobre los datos que han superado el primer filtro. Por ejemplo, para recuperar todos los edificios de una ciudad, recuperamos primero todos los edificios dentro del rectángulo circunscripto mínimo de la ciudad y luego, en el segundo paso, cotejamos solo estos con el límite exacto de la ciudad.

El método de acceso ha de dividirse en (1) un método de *clustering* físico para lograr una cantidad mínima de accesos físicos al disco, y (2) una estructura de datos lógica que permita el acceso espacial y garantice que sea correcto, preferiblemente aun cuando haya algunas deficiencias en el *clustering* físico. Este servirá para acelerar la recuperación de la salida de mapas y favorece a muchas otras formas de procesamiento de datos geométricos, tales como el cálculo de superficies. Así podemos aprovechar la localización geométrica específica del acceso que se observa en la mayoría de los algoritmos de la geometría computacional (DUTTON, 1978).

El enfoque tradicional que se encuentra en muchos de los SIG que se comercializan consiste en dividir el espacio del universo en hojas de mapas (a menudo denominadas *"facettes"* ["carillas"]) y almacenar los datos correspondientes a estas hojas de mapas como archivos separados. La cantidad de datos dentro de tales archivos es entonces lo bastante pequeña como para permitir el empleo de métodos lineales de búsqueda. A nuestro entender, ello es insuficiente por las siguientes razones:

> La granularidad del acceso es fija y proporciona acceso rápido solo si la zona de consulta es comparable con el tamaño de la hoja. Para consultas sobre zonas mucho más grandes, la respuesta es lenta porque deben abrirse y leerse muchos archivos distintos.
>
> El acceso a más de una hoja requiere la recombinación de objetos sobre los límites de la hoja. Esta es una operación complicada que consume tiempo y que solo es posible con una comprensión de la estructura interna de la representación de los objetos. Así, los tipos de los objetos geométricos tratados se definen en el nivel del almacenaje, y le resulta difícil a un usuario agregar nuevas definiciones de objetos geométricos.
>
> La mayoría de los sistemas no le ocultan las hojas al usuario, ni ofrecen un lenguaje de consultas que opere automáticamente a través de los límites de las hojas. Parece aceptable pedir a los dibujantes que conozcan qué hoja de mapa están actualizando, pero evidentemente es claramente impropio que se le requiera ese conocimiento a un usuario accidental que necesita un mapa de las cañerías de agua por una emergencia.

Se sabe que solo unas pocas estructuras lógicas de datos permiten una respuesta rápida a este tipo de acceso espacial, denominado técnicamente consulta de rango bidimensional. Se basan todas ellas en una partición autorregulada del espacio y el *clustering* de datos de una partición en una página del disco, de modo que el acceso a una página traiga muchos entes útiles para la confección del mapa pedido (NIEVERGELT *et al.*, 1984; TAMMINEN, 1982; GUTTMAN, 1984; SAMET, 1984). El método descripto por Nievergelt y sus coautores es una adaptación de una estructura *hash based* más general, para el almacenamiento de objetos multidimensionales. Resulta ser muy semejante al método FIELD TREE [árbol de campos] desarrollado por nosotros en el PANDA (FRANK, 1981, 1983).

En el método FIELD TREE, el *clustering* no solo se guía por la ubicación y la extensión de los objetos, sino que incluye también un componente del nivel de importancia del objeto. Ello acelera las respuestas a los pedidos de "visión general" [*"overview" requests*] para los que solo deben recuperarse los objetos "importantes" de una zona grande (vgr., todas las ciudades de un estado), o de mapas de detalle en los que hay que recuperar todos los objetos de una zona pequeña (vgr., un edificio con todas sus conexiones de servicios públicos). El tiempo de respuesta a las consultas es linealmente dependiente de la cantidad de objetos recuperados; así, una pantalla llena de datos cartográficos siempre demora aproximadamente lo mismo. Otros factores que influyen son el tamaño de la zona y la cantidad de objetos almacenados que le corresponden. En cambio, la cantidad total de objetos almacenados en el sistema no influye virtualmente sobre el tiempo de respuesta.

## EL LENGUAJE DE CONSULTA

Los lenguajes de consulta son muy importantes para que la recuperación de datos satisfaga ciertas necesidades inmediatas que no fueron planeadas y para las cuales no se dispone de procedimientos de acceso programados. En los DBMS comerciales suelen incluirse lenguajes interactivos para consultas *ad hoc*. Al parecer, el lenguaje SQL, desarrollado originariamente por IBM (ASTRAHAN *et al.*, 1976), se está convirtiendo actualmente en el patrón aceptado, y la propuesta correspondiente está siendo tratada en el American National Standards Institute [Instituto Norteamericano de Patrones Nacionales]. Varios grupos están trabajando actualmente en extensiones del SQL para que resulte útil a las bases de datos espaciales (EGENHOFER y FRANK, 1987b). Las aplicaciones espaciales requieren servicios que permitan recuperar los datos basados en relaciones espaciales, y el lenguaje de consulta debe ser extendido para abarcarlas. Por ejemplo, deben incluirse en la consulta las especificaciones de la presentación gráfica de los datos (FRANK, 1982b; EGENHOFER y FRANK, 1988b).

## UNA EXPERIENCIA Y EL TRABAJO FUTURO

En los últimos siete años hemos construido un DBMS experimental apto para usarse en un SIG. Se basa en una extensión del modelo de datos en red y utiliza la organización FIELD TREE para lograr un acceso espacial rápido. El PANDA consta de unas 20 000 líneas de código Pascal sumamente modular y ha sido instalado en computadoras de IBM (bajo VM/CMS) y de Digital Equipment Corp. (bajo VMS y TOPS-10). Una empresa comercial está revisando actualmente el PANDA para construir un SIG completo. Hemos utilizado el PANDA para implementar métodos de representar datos espaciales usando complejos de celdas (FRANK y KUHN, 1986). Para comprobar este método se trajeron a la pantalla del tubo de rayos catódicos algunos datos cartográficos simplificados con los que se dibujaron mapas en ella. Encontramos que un lenguaje de programación poderoso, orientado a los objetos, nos permitía escribir programas de aplicación general que pueden integrarse para formar luego aplicaciones específicas. Esto simplificaba la construcción de aplicaciones. Ahora estamos modificando la interfaz de programas del PANDA para que esté totalmente orientado a los objetos. Hemos comprobado que este era un paso necesario para permitir la utilización de métodos de ingeniería de *software* orientados a los objetos en la construcción de SIG a partir de componentes básicos. También estamos trabajando en un lenguaje de consultas para SIG. No está claro si las extensiones del SQL son suficientes o si un diseño más centrado en los objetos daría por resultado un lenguaje más fácil de usar en las aplicaciones de los SIG.

Hace falta investigar la forma de tratar las "transacciones prolongadas". Pensamos agregarle apoyo al PANDA para que pueda mantener muchas versiones del mismo objeto (p. ej. un terreno o una línea de servicios públicos) al mismo tiempo que una transacción larga se vincula con una versión específica. Será preciso construir herramientas para simplificar la fusión [*merging*] de distintas versiones cuando están confirmadas. Esto se halla estrechamente relacionado con la necesidad de tratar acontecimientos y otros datos asociados con el tiempo [cronológico] en muchos tipos de aplicaciones a los SIG. Por ejemplo, a los fines del planeamiento importa no solo el uso actual de la tierra sino también los usos previos, y a menudo existe también considerable interés por la velocidad de los cambios. Los estudios preliminares nos hacen pensar que la incorporación de datos temporales no constituirá un cambio fácil pues afectará por lo menos los métodos de almacenaje y recuperación, la administración de transacciones y la interfaz del usuario. Y también el lenguaje de consulta necesitará extensiones temporales.

## CONCLUSIONES

Como conclusión hacemos algunas recomendaciones:

Las colecciones de datos espaciales, inclusive los SIG, deberían construirse utilizando el concepto de sistema administrador de base de datos [DBMS], que resulta crucial para un sistema de información interactivo que pueda servir a muchos usuarios y a múltiples requerimientos.

Las colecciones de datos espaciales plantean algunos requisitos particulares, que hacen inadecuados los DBMS disponibles en el mercado. Estos carecen generalmente del recurso especial para lograr el *clustering* físico necesario para un rápido acceso a los datos espaciales, y están optimizados para datos

orientado a los objetos, nos permitía escribir programas de aplicación general que pueden integrarse para formar luego aplicaciones específicas. Esto simplificaba la construcción de aplicaciones. Ahora estamos modificando la interfaz de programas del PANDA para que esté totalmente orientado a los objetos. Hemos comprobado que este era un paso necesario para permitir la utilización de métodos de ingeniería de *software* orientados a los objetos en la construcción de SIG a partir de componentes básicos. También estamos trabajando en un lenguaje de consultas para SIG. No está claro si las extensiones del SQL son suficientes o si un diseño más centrado en los objetos daría por resultado un lenguaje más fácil de usar en las aplicaciones de los SIG.

Hace falta investigar la forma de tratar las "transacciones prolongadas". Pensamos agregarle apoyo al PANDA para que pueda mantener muchas versiones del mismo objeto (p. ej. un terreno o una línea de servicios públicos) al mismo tiempo que una transacción larga se vincula con una versión específica. Será preciso construir herramientas para simplificar la fusión [*merging*] de distintas versiones cuando están confirmadas. Esto se halla estrechamente relacionado con la necesidad de tratar acontecimientos y otros datos asociados con el tiempo [cronológico] en muchos tipos de aplicaciones a los SIG. Por ejemplo, a los fines del planeamiento importa no solo el uso actual de la tierra sino también los usos previos, y a menudo existe también considerable interés por la velocidad de los cambios. Los estudios preliminares nos hacen pensar que la incorporación de datos temporales no constituirá un cambio fácil pues afectará por lo menos los métodos de almacenaje y recuperación, la administración de transacciones y la interfaz del usuario. Y también el lenguaje de consulta necesitará extensiones temporales.

## CONCLUSIONES

Como conclusión hacemos algunas recomendaciones:

Las colecciones de datos espaciales, inclusive los SIG, deberían construirse utilizando el concepto de sistema administrador de base de datos [DBMS], que resulta crucial para un sistema de información interactivo que pueda servir a muchos usuarios y a múltiples requerimientos.

Las colecciones de datos espaciales plantean algunos requisitos particulares, que hacen inadecuados los DBMS disponibles en el mercado. Estos carecen generalmente del recurso especial para lograr el *clustering* físico necesario para un rápido acceso a los datos espaciales, y están optimizados para datos de características muy distintas de las de aquellos. Se observa que en general su rendimiento es insuficiente para las aplicaciones de los SIG.

Los DBMS espaciales deben incluir muchas de las características estándares de los sistemas comerciales, particularmente la protección de datos y la administración de transacciones, para evitar las pérdidas de datos debidas a desperfectos de los equipos y para permitir la concurrencia simultánea de los usuarios.

Recomendamos una arquitectura de niveles para el DBMS espacial, con un núcleo [*kernel*] de base de datos que proporcione los servicios requeridos generalmente, y módulos adicionales para adaptar el DBMS a las necesidades propias de las aplicaciones espaciales (HAERDER, 1986).

## REFERENCIAS BIBLIOGRAFICAS


Astrahan, M. M., D. D. Chamberlin, *et al.*, 1976. Sequel 2: A Unified Approach to Data Definitions, Manipulations and Control, *IBM Journal Research and Development*, Vol. 20, p. 560.

Burton, W., 1978. Efficient Retrieval of Geographical Information on the Basis of Location. *First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems* (G. Dutton (Ed)), Harvard Papers on Geographic Information Systems, Harvard University, Cambridge, Massachusetts.

CODASYL, 1962. An Information Algebra, Phase I Report, *Commun. ACM*, Vol. 5, p. 190–204.

———, 1971. *Data Base Task Group (DBTG) Report*, 1971.

Codd, E. F., 1982. Relational Database: A Practical Foundation for Productivity, *Commun. ACM*, Vol. 25, p. 109.

Dangermond, J., and S. Morehouse, 1987. Trend in Hardware for Geographic Information Systems, *Proceedings Eighth International Symposium on Computer-Assisted Cartography* (N. Chrisman (Ed.)), Baltimore.

Denning, D. E., and U. Schloerer, 1980. A Fast Procedure for Finding A Tracker in a Statistical Database, *ACM Transactions on Database Systems*, Vol. 5.

Dutton, G., 1978. Navigating ODYSSEY, *First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems* (G. Dutton (Ed)), Harvard Papers on Geographic Information Systems, Harvard University, Cambridge, Massachusetts.

Egenhofer, M., and A. Frank, 1987a. *PANDA: An Object-Oriented Database Based on User-Defined Abstract Data Types*, Report No. 62, Surveying Engineering, University of Maine, Orono, Maine.

———, 1987b. An Extended SQL Syntax to Treat Spatial Objects. *Proceedings of the Second International Seminar on Trends and Concerns of Spatial Sciences*, Fredericton, New Brunswick.

———, 1988a. "Project Oriented Modeling a Powerful Tool for GIS", submitted for publication.

———, 1988b. Towards a Spatial Query Language: User Interface Considerations. *14th International Conference on Very Large Data Bases*, Los Angeles, California.

Frank, A., 1981. Applications of DBMS to Land Information Systems, *Proceedings VII International Conference on Very Large Data Bases*, Cannes (France), p. 448.

———, 1982a. *PANDA: Pascal Network Database Management System* (in German), Report 62, Institute for Geodesy and Photogrammetry, Swiss Federal Institute of Technology, Zurich, Switzerland.

———, 1982b. MAPQUERY: Data Base Query Language for Retrieval of Geometric Data and Their Graphical Representation, *SIGGRAPH '82 Conference Proceedings, Computer Graphics*, Vol. 16, p. 199.

———, 1983. "Storage Methods for Space Related Data: The FIELD TREE", *Spatial Algorithms for Processing Land Data with a Microcomputer* (M. Barr (Ed.)), Boston, Lincoln Institute of Land Policy.

———, 1984a. Computer Assisted Cartography: Graphics or Geometry? *Journal of Surveying Engineering*, Vol. 110, No. 2, pp. 159–168.

———, 1984b. Extending a Network Database with Prolog, *Expert Database Systems, Proceedings of the First International Workshop on Expert Database Systems*, Kiawah Island, South Carolina.

———, 1986a. *PANDA: An Object Oriented Pascal Network Database Management System*, Report 57, Surveying Engineering, University of Maine, Orono, Maine.

Frank, A., and W. Kuhn, 1986. Cell Graphs: A Provable Correct Method for the Storage of Geometry, *Proceedings Second International Symposium on Spatial Data Handling*, Seattle, Washington.

Frank, A., and M. Tamminen, 1982. Management of Spatially Referenced Data, *Proceedings International Symposium Land Information at the Local Level*, (A. Leick (Ed.)), University of Maine, Orono, Maine, p. 330.

Guttman, A., 1984. *New Features for a Relational Database System to Support Computer Aided Design*, Memorandum No. UCB/ERL M84/52, Electronic Research Laboratory, College of Engineering, University of California, Berkeley.

Haerder, Th., 1986. New Approaches to Object Processing in Engineering Database, *Proceedings 1986 International Workshop on Object-Oriented Database Systems*, Pacific Grove, California.

Haerder, Th., and A. Reuter, 1982. *Database Systems for Non-Standard Applications*, Report 54/82, Fachbereich Informatik, Universitaet Kaiserslautern, (FRG).

———, 1985. Architecture of Database Systems for Non-Standard Applications (in German). *Database Systems in Office, Engineering and Scientific Environment, (A. Blaser and P. Pistor, eds.), Springer Verlag, New York.*

Manola, F., J. Orenstein, and U. Dayal, 1987. Geographic Information Processing in the Probe Database System, *Proceedings Eighth International Symposium on Computer-Assisted Cartography*, Baltimore.

National Academy of Science, 1980. *Need for a multipurpose Cadastre* by

T - 089

# REQUISITOS DE UN SISTEMA ADMINISTRADOR DE BASES DE DATOS PARA UN SIG(*)

Frank U., Andrew

Departamento de Ingeniería Topográfica. Universidad de Maine
Orono, Maine 04469, E.U.A.

Los sistemas de información geográfica (SIG) almacenan grandes cantidades de datos que deben estar a disposición de muchos usuarios. Los sistemas administradores de bases de datos (DBMS = Database Management Systems) fueron diseñados para facilitar el almacenamiento y la recuperación de grandes colecciones de datos. Incluyen elementos para proteger y asegurar los datos, para imponerles consistencia una vez almacenados y para posibilitar que estén a disposición de muchos usuarios al mismo tiempo. Estos servicios son necesarios para los SIG, y por ende los SIG deben construirse mediante la utilización de sistemas administradores de bases de datos. Sin embargo, los SIG requieren una performance elevada y plantean requisitos muy especiales en materia de administración de bases de datos. Los DBMS diseñados para uso comercial no se adaptan bien a los SIG porque no pueden alojar datos espaciales y satisfacer la recuperación de la gráfica de los mapas. En este trabajo se presenta una visión general de la arquitectura de un DBMS especialmente adaptado al manejo de datos espaciales. Para cada nivel se indican técnicas específicas que resultan útiles a los DBMS para SIG, por ejemplo, para la administración de buffers, clustering de los datos y el acceso espacial. Se describen los esfuerzos efectuados para implementar el DBMS PANDA.

(*) Traducción al español de Photogrammetric Engineering and Remote Sensing, Vol. 54, N°11, November 1988, pp. 1557-1564.
(Traductor: Fernando Lida García).

Jueves 24

SALA 2

5:20 - 15:45 T-089
"Requisitos de un Sistema Administrador de Bases de Datos para un SIG"
Frank, A.
U.S.A.

5:45 - 16:10 T-090
"Aportes para la Construcción de una Teoría General de los SIG"
Olivares, O.R.
Argentina

6:10 - 16:25 Café.

6:25 - 16:50 T-091
"Activities of GRID - The Global Resource Information Database of the United Nations Environment Programme"
Fernández, N.; Croze, H.
Kenya

6:50 - 17:15 T-092
"Resolución de Problemas Geográficos en Inglés y en Español: Implicaciones para el Diseño de SIG."
Gould, M.; Mark, D.; Gavidea G., C.
U.S.A.

7:15 - 17:40 T-093
"El Sistema GPS y su relación con los SIG"
Lange H.; Morales Z., R.
Chile