ARQUITETURA DE SOFTWARE PARA CONSTRUÇÃO DE BANCOS DE DADOS GEOGRÁFICOS COM SGBD OBJETO-RELACIONAIS

KARINE REIS FERREIRA, GILBERTO RIBEIRO QUEIROZ, JOÃO ARGEMIRO PAIVA , RICARDO CARTAXO MODESTO DE SOUZA, GILBERTO CÂMARA

Instituto Nacional de Pesquisas Espaciais - INPE
Av. dos Astronautas, 1758, São José dos Campos (SP), Brazil 12227-001

{karine, gribeiro, miro, cartaxo, gilberto}@dpi.inpe.br

Abstract. This paper describes a software architecture for development of geographic databases that uses object-relational DBMS, like PostgreSQL and Oracle Spatial, sharing a common programming interface. This work is part of TerraLib, an open source software allowing a collaborative environment and use for the development of multiple GIS tools.

Keywords. Object-relational DBMS, Geographic Databases, Software Architecture, GIS.

ARQUITETURA DE SOFTWARE PARA CONSTRUÇÃO DE BANCOS DE DADOS GEOGRÁFICOS COM SGBD OBJETO-RELACIONAIS

1. Introdução

A área de bancos de dados geográficos (BDG) e o desenvolvimento de tecnologia de sistemas de informação geográfica (GIS) estarão passando por mudanças substanciais nos próximos anos, induzidas por uma nova geração de SGBDs objeto-relacionais, como o ORACLE e POSTGRESQL, que permitem incorporar tipos de dados espaciais. Deste modo, abre-se a perspectiva da construção de GIS onde tanto os atributos como as geometrias de dados espaciais sejam gerenciados pelo SGBD. Esta integração tem o potencial de mudar completamente o desenvolvimento de tecnologia de GIS, permitindo a transição dos atuais sistemas monolíticos (que contém centenas de funções) para uma nova geração de aplicativos geográficos (*"spatial information appliances"),* sistemas dedicados para necessidades específicas (Egenhofer 1999). Deste modo, um desafio importante para a comunidade de BDG é encontrar maneiras de utilizar a nova geração de SGBD com tipos de dados espaciais.

Uma das respostas possíveis para este desafio é o estabelecimento de uma rede de desenvolvimento cooperativo, baseado em tecnologia "open source". De forma similar às soluções ligadas à tecnologia Linux, a disponibilidade de software livre para GIS permitiria a pesquisadores e provedores de soluções o acesso a um conjunto mais amplo de ferramentas do que é atualmente oferecido por companhias comerciais. Com esta motivação, os autores estão desenvolvendo a **TerraLib** (Câmara, Vinhas et al. 2001)**,** uma biblioteca de software livre base para uma nova geração de aplicativos geográficos.

Um dos desafios importantes no desenvolvimento de uma biblioteca como a **TerraLib** é compatibilizar as capacidades oferecidas pelos diferentes SGBD objeto-relacionais numa única interface de programação de aplicações. Para isto, torna-se necessário descrever as operações de consulta e armazenamento de dados espaciais de forma genérica, e realizar o mapeamento para as características específicas de cada SGBD. Neste contexto, este artigo descreve uma arquitetura de software para construção de bancos de dados geográficos que utilizam SGBD objeto-relacionais, mantendo uma mesma interface de programação. O trabalho faz parte do desenvolvimento da TerraLib, cuja primeira versão possui *drivers* para os SGBD ORACLE e POSTGRESQL. Nas seções seguintes, detalharemos a arquitetura de

software utilizada. Na seção 2, descrevemos as extensões espaciais do ORACLE e do POSTGRESQL. Na seção 3, descrevemos em detalhe a arquitetura de software. Na Seção 4, ilustramos as capacidades da TerraLib a partir da descrição do aplicativo TerraView, um ambiente simplificado para visualizar dados espaciais.

2. Extensões Espaciais de SGBD

Atualmente, existem basicamente três extensões comerciais disponíveis no mercado para tratar de dados geográficos no formato vetorial: Oracle Spatial(Ravada and Sharma 1999), IBM DB2 Spatial Extender (Mullins 2000) e Informix (Flannery 2000). No universo do software de código fonte aberto e gratuito existe um projeto para a construção de uma extensão geográfica baseada no PostgreSQL (Geschwinde and Schoenig 2001), chamada de PostGIS (Ramsey 2002). Todas essas extensões baseiam-se nas especificações do OpenGIS (Consortium 1995), porém, apresentam variações relevantes entre os modelos de dados, semântica dos operadores espaciais, mecanismos de indexação e esquema e sintaxe da SQL extendida com tipos espaciais. A seguir, são apresentadas as características e funcionalidades de algumas destas extensões.

2.1. Oracle Spatial

O Oracle Spatial é uma extensão espacial do SGBD Oracle, que utiliza seu modelo objeto-relacional. Esta extensão contém um conjunto de funcionalidades e procedimentos que pemite armazenar, acessar e analizar dados espaciais em um banco de dados Oracle. Seu modelo de dados consiste em uma estrutura hierárquica de elementos, geometrias e layers; onde layers são compostos por geometrias, que por sua vez são compostas por elementos. Os elementos podem ser do tipo Point, LineString ou Polygon (com ou sem ilhas). Uma geometria pode ser formada por um único elemento ou por um conjunto homogêneo (MultiPoint, MultiLinesString ou MultiPolygon) ou heterogêneo (Collection) de elementos. E, finalmente, um layer é formado por um conjunto de geometrias que possuem os mesmos atributos.

Devido à utilização de um modelo objeto-relacional, cada geometria é armazenada em um objeto chamado SDO_GEOMETRY. Este objeto contém a geometria em si, suas coordenadas, e informações sobre seu tipo e projeção. Em uma tabela espacial, os atributos alfanuméricos da geometria são definidos como colunas de tipos básicos (VARCHAR2, NUMBER, etc) e a geometria, como uma coluna do tipo SDO_GEOMETRY. Sendo assim,

cada tabela espacial armazena um layer, o qual é composto pelo conjunto de todas geometrias desta tabela.

O Oracle Spatial fornece um conjunto de operadores e funções espaciais, que são utilizados juntamente com a linguagem SQL, para suportar consultas espaciais. Para consultar relações topológicas entre duas geometrias é utilizado um operador chamado SDO_RELATE. Este operador implementa o Modelo de 9-Interseções definido por Egenhofer (Egenhofer and Herring 1991). Este modelo considera as interseções, vazia (0) ou não vazia (1), entre os interiores, fronteiras e exteriores de duas geometrias. O SDO_RELATE recebe como parâmetro o tipo de relação topológica que deve ser computada. Os possíveis parâmetros são : `Equal`, `Disjoint`, `Touch`, `Inside`, `OverlapBdyIntersect`, `OverlapBdyDisjoint`, `Anyinteract`, `Contains`, `On`, `Covers` e `Coveredby`.

Quanto à indexação espacial, esta extensão fornece dois tipos de índices R-tree e Quadtree. Cada um desses índices é apropriado para diferentes situações e podem ser usados simultaneamente para indexar uma mesma coluna geométria. Um cenário que ilustra o uso prático desta extensão é o seguinte: “O oficial encarregado do serviço contra incêndios precisa de uma lista de todas as áreas sensíveis dentro de de um raio de 8 km de uma área de lixo tóxico”.

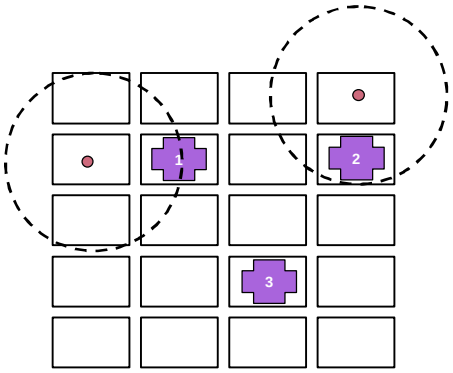

Figura 01: Cenário do problema

Partindo do princípio da existência de duas tabelas com atributos espaciais:

zona_sensível	
Nome Atributo	**Tipo**
nome	VARCHAR2(50)
zona	SDO_GEOMETRY

area_risco	
Nome Atributo	**Tipo**
nome	VARCHAR2(50)
local	SDO_GEOMETRY

A seguinte consulta em SQL poderia ser utilizada para responder o problema:

```
SELECT ass.nome, ar.nome
  FROM AREA_RISCO ar, AREA_SENSIVEL ass,
       USER_SDO_GEOM_METADATA m
 WHERE m.table_name = 'AREA_RISCO' AND
      (SDO_RELATE(ass.zona,
       SDO_GEOM.SDO_BUFFER(ar.local, m.diminfo,
       8000), 'mask=ANYINTERACT querytype=WINDOW') =
      'TRUE');
```

2.2. PostgreSQL

O PostgreSQL (Stonebraker and Rowe 1986) é um sistema gerenciador de banco de dados objeto-relacional, gratuito e de código fonte aberto. Em sua distribuição oficial, é oferecido os seguintes recursos para trabalhar com dados espaciais:

- Tipos geométricos: point, box, path, polygon e circle.
- Indexação espacial: possui uma R-Tree, cuja implementação está limitada a dados com até 8Kbytes, sendo bastante limitada para dados geográficos reais. No entanto, permite a definição de uma R-Tree sobre o mecanismo de indexação conhecido como GiST (Hellerstein, Naughton et al. 1995).
- Operadores espaciais: apresenta apenas alguns poucos operadores e bem limitados. Por exemplo, no operador "contém" (@) uma das geometrias deve ser do tipo ponto e o operador "igual" (~=) só é aplicado a duas geometrias do tipo polígono (polygon).

Como as funcionalidades oferecidas são bastante limitadas para o desenvolvimento de Sistemas de Informação Geográficas, uma nova extensão está em desenvolvimento. O PostGIS, como ela é chamada, é uma extensão geográfica, gratuita e de código fonte aberto, que visa permitir ao SGBD PostgreSQL gerenciar informações geo-espaciais. Sendo sua definição e implementação baseadas nas especificações do OpenGIS. Atualmente, o PostGIS conta com os tipos de dados espaciais contidos na especificação do OGIS e utiliza um mecanismo de indexação R-Tree sobre o esquema do GiST. Os operadores espaciais encontram-se em desenvolvimento.

3. Arquitetura de Software

Um dos objetivos da TerraLib é o desenvolvimento de aplicativos GIS baseados nos avanços tecnológicos dos sistemas de bancos de dados, especialmente os espaciais, realizando a completa integração dos tipos de dados espaciais dentro dos SGBDs (Câmara, Souza et al. 2000). Para realizar tal tarefa, ela fornece uma arquitetura que dá acesso direto aos dados que podem ser armazenados em diversos SGBDs, como Oracle Spatial, PostgreSQL e MySQL, possibilitando a criação e a manipulação de bancos de dados geográficos. Essa arquitetura fornece uma interface comum que possibilita ao desenvolvedor em não ter que se preocupar com os detalhes de cada SGBD. Isso permite que este enfoque nas funcionalidades que um GIS deve ter, como ferramentas para análise espacial, visualização gráfica dos dados geográficos e entrada de dados ao invés de se preocuparem com o gerenciamento dos dados. A arquitetura para construção de bancos de dados geográficos apresentada neste trabalho (Figura 02) encontra-se implementada na TerraLib e é composta pelos seguintes componentes:

- Modelo de Dados: composto por um conjunto específico de tabelas para a representação dos dados geográficos nos SGBDs.
- Kernel: composto pelas classes básicas (estruturas de dados) para representação em memória dos dados geográficos tanto no formato vetorial quanto matricial, por operadores topológicos e direcionais, classes de sistema de projeção e algoritmos.
- Drivers: formado por classes que fornecem uma interface comum e que dão o suporte básico para trabalhar com os dados geográficos nos SGBDs.
- SGBDs: a TerraLib permite a integração tanto com SGBDs Relacionais quanto com Objeto-Relacionais. Atualmente, ela possui interface com o SGBD relacional MySQL e com os objeto-relacionais Oracle Spatial e PostgreSQL. Os drivers são os responsáveis em manter essas interfaces.

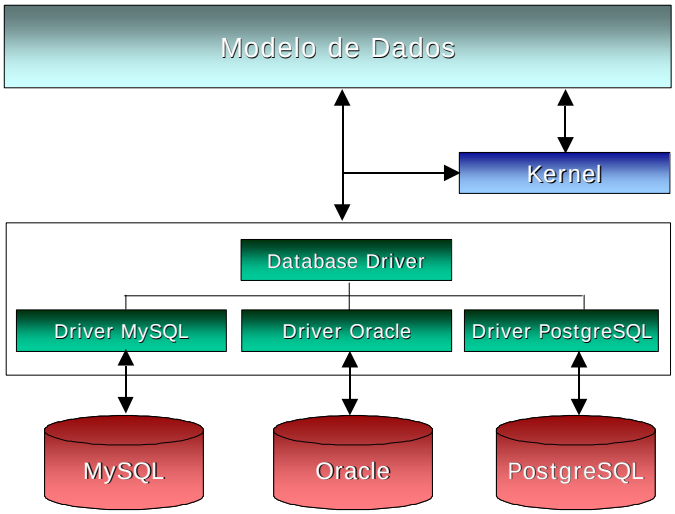

Figura 02: Arquitetura da TerraLib para construção de bancos geográficos

3.1. Modelo de Dados

O modelo de dados dessa arquitetura define um conjunto específico de tabelas com o objetivo de facilitar a representação dos dados geográficos nos SGBDs. Neste esquema, as informações geográficas sobre um determinado objeto ou fenômeno geográfico são representadas em um plano de informação (*layer*). Um plano de informação (PI) pode ser composto por mais de um tipo de representação (vetorial ou matricial), estando ou não associado a uma tabela de atributos não espaciais.

Na representação vetorial, um dado pode ser mapeado para três tipos de tabelas com colunas capazes de representar polígonos, linhas ou pontos. Estas tabelas possuem um campo para fazer a ligação com a tabela de atributos não espaciais correspondente. Na representação matricial (*raster*) é feito um particionamento do *raster* por linhas, onde cada linha é armazenada em um registro da tabela em colunas do tipo BLOB. A Figura 03, abaixo, ilustra de maneira simplificada o modelo de dados.

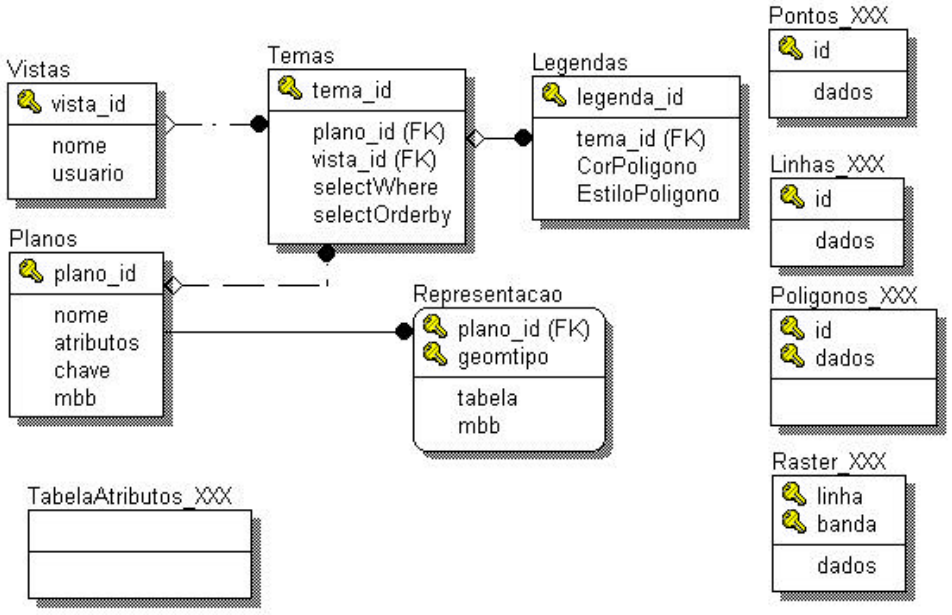

Figura 03 : Modelo de Dados simplificado

Conforme pode ser observado no modelo apresentado na Figura 03, a tabela “planos” contém as informações de cada PI, seu nome, retângulo envolvente (mbb) e o nome da tabela de atributos não espaciais associada ele. A tabela “representacao” possui um campo que contém o índice do PI, outro chamado “geomtipo” que indica o tipo de representação (ponto, linha, polígono ou matricial) e outro chamado “tabela” que contém o nome da tabela onde os dados espaciais daquela representação estão armazenados.

Neste modelo, um PI pode estar associado a diferentes temas, cada um com características específicas. Por exemplo, podemos ter um PI que contenha todos os estados do Brasil com seus atributos, área, população, taxa de analfabetismo, dentre outros. Uma consulta sobre esse PI, como por exemplo, “selecione todos os estados onde a população seja maior que 5 milhões”, resultaria em um novo tema associado a este PI, e não a um novo PI. Este tema é armazenado na tabela “temas” contendo as cláusulas da consulta em SQL que o gerou, e é associado ao PI correspondente através do relacionamento entre as tabelas “planos” e “temas”. Além disso, cada tema é associado a uma legenda, a qual é armazenada na tabela “legendas”, que contém características de visualização, como cores, formatos e tamanhos.

Para uma melhor organização dos dados no SGBD, este modelo implementa o conceito de vistas que são associadas aos usuários. Assim, cada usuário cria suas próprias vistas e associam a cada uma os temas relacionados. Pode-se dizer que o conceito de vistas é semelhante a idéia de diretórios, onde cada usuário possui seus diretórios que contém informações relacionadas sobre um mesmo assunto.

3.2. Kernel

As classes do Kernel para manipulação dos dados em memória estão mostradas na Figura 04. As instâncias dessas classes podem ser armazenadas nos SGBDs seguindo o esquema de representação mostrado na seção anterior, através do driver de banco de dados.

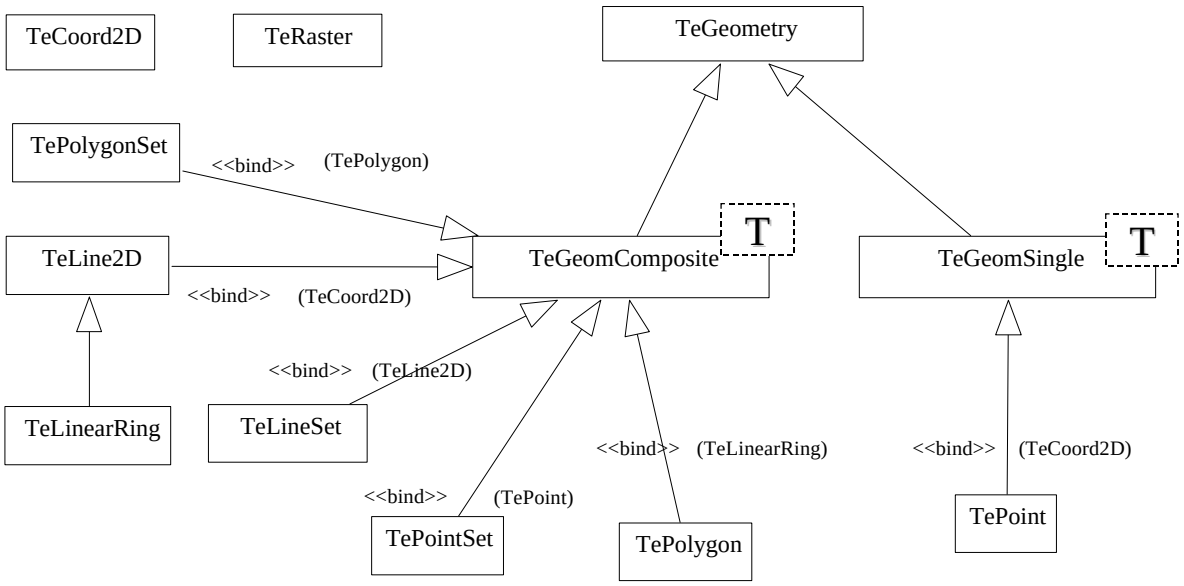

Figura 04: Classes do Kernel da TerraLib

Uma instância de um TePolygonSet pode ser armazenado no banco como um polígono por tupla ou não, dependendo do driver e do suporte oferecido pelo banco, sendo transparente para o desenvolvedor.

3.3. Database Driver

O *driver* de banco de dados é uma interface entre os diversos sistemas de bancos de dados e o Kernel da TerraLib. Ele permite o armazenamento, recuperação e manipulação dos dados geográficos - a saber, os seus atributos espaciais e não espaciais - diretamente nos SGBDs, sem a necessidade de utilização de estruturas proprietárias. Para cada SGBD é criado um driver específico que implementa um conjunto comum de interfaces para manipulação e definição do esquema da base geográfica. Essa arquitetura é muito semelhante à da ODBC.

Cada driver dá o suporte necessário para trabalhar com o esquema em um SGBD específico, fornecendo métodos que criam este esquema e que fazem o mapeamento entre os tipos de dados deste SGBD para os tipos de dados presentes no kernel da biblioteca. Além disso, ele traduz as consultas espaciais para o dialeto SQL de cada SGBD, delegando a este sua execução. Esses drivers devem explorar ao máximo os recursos oferecidos pelos SGBDs,

principalmente, os que apresentam extensões espaciais capazes de fornecer mecanismos de indexação espacial, tipos de dados e operadores espaciais. A interface comum desses drivers é definida por duas classes abstratas, TeDatabase e TeDatabasePortal.

A classe TeDatabase fornece a interface para:

- Estabelecimento de conexão com o servidor de banco de dados
- Execução de comandos SQL DDL e DML
- Criação de novas tabelas
- Criação do modelo de dados da TerraLib
- Definição dos índices espaciais
- Criação de integridade referencial e índice secundários

A classe TeDatabasePortal apresenta em sua interface métodos para:

- Execução de consultas SQL que retornem um conjunto de resultados
- Manipular o conjunto de resultados da execução de uma consulta
- Inserção, atualização e recuperação das geometrias segundo o modelo de dados apresentado anteriormente.
- Execução de consultas espaciais
- Armazenamento e recuperação de dados matriciais

O driver de cada banco será responsável pelos detalhes de implementação desta interface. Abaixo, são apresentados os drivers do Oracle Spatial e do PostgreSQL mostrando a particularidades de cada um.

3.3.1. Driver Oracle

O driver de interface com o SGBD Oracle, utiliza os recursos oferecidos pela sua extensão espacial Oracle Spatial, como tipos de dados, operadores e métodos de indexação espaciais. Neste driver, os tipos vetoriais da TerraLib, TePoint, TeLine2D e TePolygon, são mapeados para objetos do tipo SDO_GEOMETRY. Este objeto armazena o tipo da geometria (como por exemplo, Point, Line String, Arc Line String, Polygon, Polygon with holes), sua projeção e suas coordenadas. Cada geometria é armazenada em um registro, por exemplo, um polígono que contém ilhas, é armazenado completamente em um SDO_GEOMETRY do tipo Polygon with holes.

Quanto à indexação espacial, ele utiliza os métodos fornecidos pela extensão, R-tree e Quadtree. Neste caso, o desenvolvedor escolhe o tipo de indexação mais apropriada para seus dados geográficos. Além disso, este driver utiliza funções disponíveis nesta extensão para avaliar a performance dos índices criados (ANALYZE_RTREE) e para reconstruir um antigo

índice, depois de inserir novas geometrias (ALTER INDEX REBUILD). As consultas espaciais utilizam os operadores e funções espaciais disponíveis. Por exemplo, SDO_RELATE para consultar relações topológicas entre geometrias, SDO_BUFFER para retornar uma nova geometria a partir de outra e SDO_DISTANCE para retornar a distância entre duas geometrias. Neste caso, o driver monta a consulta em SQL utilizando os operadores e funções correspondentes, e o passa para o SGBD computar.

3.3.2. Driver PostgreSQL

O driver para banco de dados PostgreSQL utiliza os tipos POINT, PATH, POLYGON e BLOB para armazenar em uma tupla os tipos de dados TePoint, TeLine2D, TeLinearRing e TeRaster respectivamente. Cada anel (TeLinearRing) de uma instância de um TePolygon é armazenado em uma tupla, pois o tipo POLYGON não permite o armazenamento de polígonos com ilhas. Quanto à indexação, este driver não utiliza a R-Tree padrão do PostgreSQL devido às suas limitações, empregando a R-Tree construída sobre o GiST.

Os métodos de consultas espaciais deste driver utilizam uma estratégia de duas etapas para o seu procesamento. Primeiramente, é feito um filtro pelo retângulo envolvente da geometria através de operadores que utilizam o índice espacial do SGBD. Na segunda etapa, o resultado é avaliado com os operadores espaciais do kernel, que computam a geometria exata. Isso é feito porque o PostgreSQL não apresenta suporte para todos os operadores topológicos. Futuramente, este driver será construído sobre o PostGIS, de forma a delegar a esta extensão tanto os recursos de indexação e tipos de dados quanto os operadores espaciais.

4. Terraview

O TerraView, desenvolvido pela DPI/INPE, é um aplicativo "*open source*" construído sobre a TerraLib, utilizando a arquitetura apresentada neste trabalho. As Figuras 05 e 07 ilustram sua interface gráfica. Este software contém as seguintes características:

- Interface com diferentes SGBDs: possibilita que o usuário selecione o SGBD a ser utilizado, instanciando o driver correspondente e fazendo a conexão com o servidor de dados.
- Suporte a tipos de dados espaciais: o software suporta diferentes tipos de dados espaciais, incluindo polígonos, linhas, redes, células, pontos, imagens e superfícies.

- Ferramentas de visualização: permite a visualização do dado geográfico, de sua componente espacial através do um canvas e de seus atributos afanuméricos através de uma tabela. Além disso, possuem algumas ferramentas como "*zoom in*" e "*zoom out*".
- Integração de dados espaciais: possibilita a integração de layers de diferentes tipos (superfícies, pontos, etc.) combinados em um mesmo frame.
- Importação de dados: permite a importação de dados a partir de arquivos em formatos proprietário (como por exemplo, shape file e mif) para o esquema definido no SGBD.
- Consultas espaciais: disponobiliza alguns tipos de consultas que podem ser aplicadas aos dados de um único layer ou de dois layers distintos.

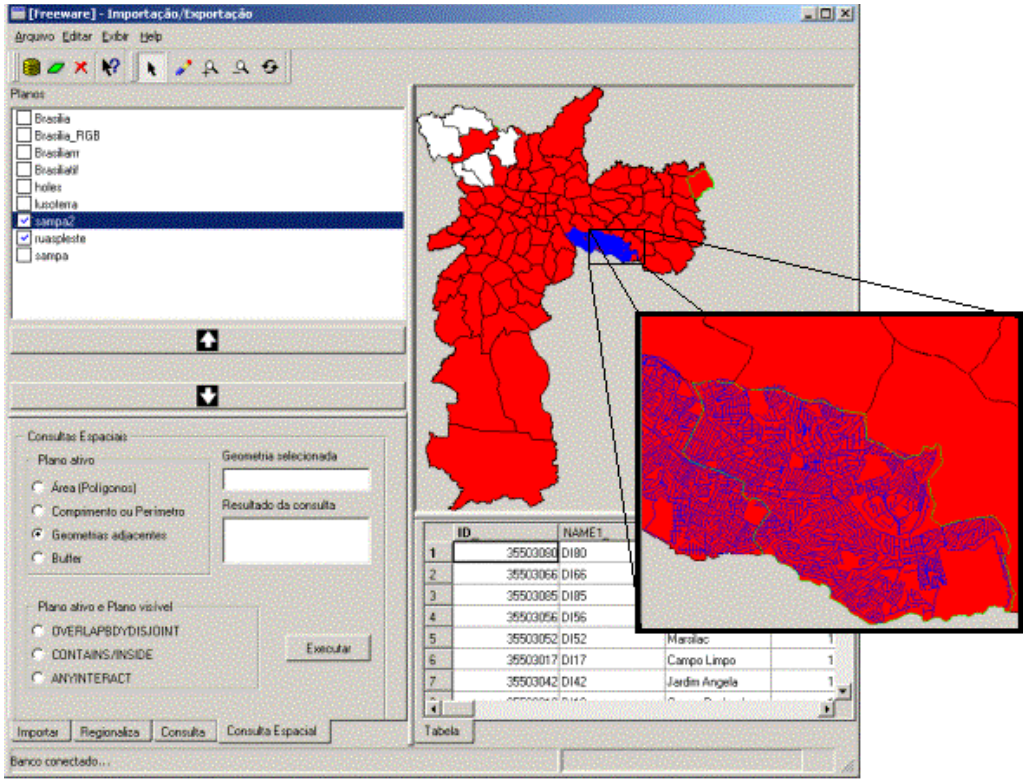

Figura 05: Interface Gráfica do Aplicativo TerraView

Na Figura 05, é mostrado a visualização de dois PIs, setores e ruas, ambos formados por dados de representação vetorial, um por polígonos e o outro por linhas, respectivamente. Um tipo de consulta espacial feita sobre o plano setores é mostrado na parte superior do PI, onde todos os setores adjacentes a um específico, selecionado pelo usuário diretamente no canvas, foram retornados e preenchidos por branco. Há ainda opções de fazer consultas entre os dados de dois PIs distintos, como por exemplo, retornar todas as ruas que cruzam as fronteiras de um setor específico. Esta consulta é ilustrada na Figura 06, onde as ruas

retornadas são pintadas de branco. A Figura 07 mostra a visualização de uma imagem no TerraView, utilizando os recursos dos drivers para armazenar e recuperar dados matriciais.

Figura 06: Consulta espacial

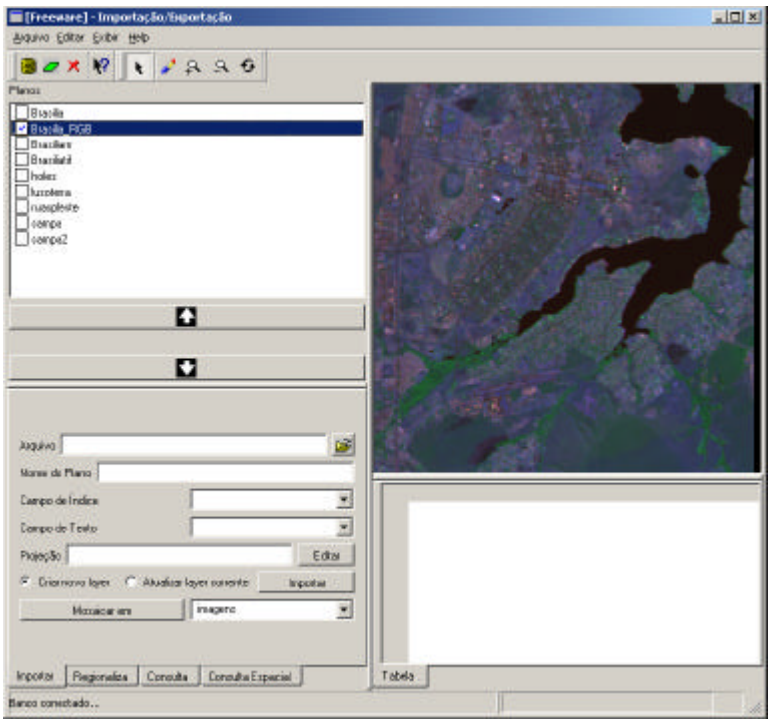

Figura 07: Representação matricial no TerraView

5. Conclusões

Este trabalho mostra como os SGBDs podem desempenhar um papel fundamental no desenvolvimento de novos aplicativos GIS, assim como facilitar o seu desenvolvimento. Outro ponto importante a ser ressaltado é que o armazenamento de dados dentro de um SGBD elimina o problema de uso de estruturas proprietárias, que dificultam a interoperabilidade.

Além disso, este artigo apresenta na prática, a utilização da nova geração de SGBD-OR e o seu potencial para gerenciamento de dados geográficos. Apesar das limitações apresentadas para a manipulação de dados matriciais, os avanços tecnológicos destes SGBDs os tornam promissores para a integração com GIS. Embora o PostgreSQL não forneça um conjunto de funcionalidades satisfatórias para o trabalho com dados geográficos, há a possibilidade de extendê-lo, sendo necessário certos investimentos, que serão realizados pela própria equipe deste trabalho, em parceria com o PostGIS.

Como trabalhos futuros, podemos citar a implementação de operadores para suportar consultas a dados matriciais e realização de análise de desempenho dos drivers existentes.

Referências Bibliográficas

Câmara, G., R. Souza, et al. (2000). TerraLib: Technology in Support of GIS Innovation. II Workshop Brasileiro de Geoinformática, GeoInfo2000, São Paulo.

Câmara, G., L. Vinhas, et al. (2001). Design Patterns in GIS Development: The Terralib Experience. III Workshop Brasileiro de Geoinformática, Rio de Janeiro, SBC.

Consortium, O. G. (1995). OpenGIS Simple Features Specification For SQL Revision 1.1.

Egenhofer, M. (1999). Spatial Information Appliances: A Next Generation of Geographic Information Systems. First Brazilian Workshop on GeoInformatics, Campinas, Brazil.

Egenhofer, M. and J. Herring (1991). Categorizing Binary Topological Relationships Between Regions, Lines, and Points in Geographic Databases. Orono, ME, Department of Surveying Engineering, University of Maine.

Flannery, R. M. (2000). Informix Handbook, Informix Press.

Geschwinde, E. and H.-J. Schoenig (2001). Postgresql : Developer's Handbook. New York, SAMS Publisher.

Hellerstein, J. M., J. F. Naughton, et al. (1995). Generalized Search Trees for Database Systems. Proc. 21st Int'l Conf. on Very Large Data Bases, Zürich.

Mullins, C. S. (2000). DB2 Developer's Guide. New York, SAMS Publisher.

Ramsey, P. (2002). PostGIS Manual. **2002**. <http://postgis.refractions.net>

Ravada, S. and J. Sharma (1999). Oracle8i Spatial: Experiences with Extensible Databases. SSD'99. R. H. Guting, D. Papadias and F. Lochovsky. Berlin, Springer-Verlag**:** 355-359.

Stonebraker, M. and L. A. Rowe (1986). The Design of POSTGRES. ACM-SIGMOD International Conference on the Management of Data. Washington, D.C.**:** 340-355.