MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS

Tutorial sobre Bancos de Dados Geográficos

GeoBrasil 2006

Instrutores:

Gilberto Ribeiro Queiroz
Karine Reis Ferreira

Índice

1 Representação Computacional de Dados Geográficos

Gilberto Câmara

1.1 Introdução

Este capítulo examina os problemas básicos de representação computacional de dados geográficos, e esclarece questões da seguinte natureza: *Como representar os dados geográficos no computador? Como as estruturas de dados geométricas e alfanuméricas se relacionam com os dados do mundo real? Que alternativas de representação computacional existem para dados geográficos?*

Em seu livro "Olhos de Madeira", Carlo Ginzburg nos traz um fascinante ensaio sobre a origem da palavra 'representação'. A origem do termo remonta ao século XIII, chamando-se *représentation* aos manequins de cera exibidos junto ao cadáver dos reis franceses e ingleses durante as cerimônias funerárias (Ginzburg, 2001). Enquanto o soberano era velado, a presença do manequim era um testemunho à transcendência do rei e a sua presença futura no mundo dos mortos. O manequim tinha a função de lembrar aos presentes que o rei havia assumido uma outra forma e que uma nova vida se iniciava para o morto. Nesta nova forma, apesar de morto o rei continuaria presente para seus súditos ("*re* + *présentation*").

Assim, desde a sua origem a palavra 'representação' está associada a uma forma abstrata de descrição do mundo. O uso do manequim como representação do soberano morto é apenas um exemplo do problema mais geral da construção de abstrações que descrevem o mundo. Para explicar como funcionam os bancos de dados geográficos, este capítulo descreve o processo de transformar aos conceitos abstratos de *espaço geográfico* no referindo ao *espaço computacionalmente representado*. Para exemplificar, consideremos alguns problemas:

- Uma cientista social deseja entender e quantificar o fenômeno da *exclusão social* uma grande cidade brasileira, através de mapas de exclusão/inclusão social, gerados a partir de dados censitários (Sposati, 1996).
- Uma ecóloga pretende estudar os remanescentes florestais da Mata Atlântica, através de estudos de *fragmentação* obtidos a partir de interpretação de imagens de satélite (Pardini et al., 2005).
- Uma pedóloga pretende determinar a *distribuição* de propriedades do solo uma área de estudo, a partir de um conjunto de amostras de campo (Bönisch et al., 2004).

O que há de comum nesses casos? A especialista lida com conceitos de sua disciplina (*exclusão social, fragmentos, distribuição de propriedades do solo*) e precisa de representações que traduzam estes conceitos para o computador. Após esta tradução, ela poderá compartilhar os dados de seu estudo, inclusive com pesquisadores de outras disciplinas.

1.2 Descrição geral de sistemas de informação geográfica

O termo *sistemas de informação geográfica* (SIG) é aplicado para sistemas que realizam o tratamento computacional de dados geográficos. A principal diferença de um SIG para um sistema de informação convencional é sua capacidade de armazenar tanto os atributos descritivos como as geometrias dos diferentes tipos de dados geográficos. Assim, para cada lote num cadastro urbano, um SIG guarda, além de informação descritiva como proprietário e valor do IPTU, a informação geométrica com as coordenadas dos limites do lote. A partir destes conceitos, é possível indicar as principais características de SIGs:

- Inserir e integrar, numa única base de dados, informações espaciais provenientes de meio físico-biótico, de dados censitários, de cadastros urbano e rural, e outras fontes de dados como imagens de satélite, e GPS.
- Oferecer mecanismos para combinar as várias informações, através de algoritmos de manipulação e análise, bem como para consultar, recuperar e visualizar o conteúdo da base de dados geográficos.

Os componentes de um SIG estão mostrados na Figura 1.1. No nível mais próximo ao usuário, a *interface homem-máquina* define como o sistema é operado e controlado. Esta interface pode ser tanto baseada na metáfora da "mesa de trabalho" (Kuhn and Frank, 1991) (Richards and Egenhofer, 1995) (Câmara, 1999), como adaptada ao ambiente de navegação da Internet (Kraak and Brown, 2001), quanto baseada em linguagens de comando como Spatial SQL (Egenhofer, 1994) e LEGAL (Câmara et al., 1995). No nível intermediário, um SIG deve ter mecanismos de processamento de dados espaciais. A *entrada* de dados inclui os mecanismos de conversão de dados (Hohl, 1998). Os algoritmos de *consulta e análise espacial* incluem as operações topológicas (Egenhofer and Franzosa, 1991), álgebra de mapas (Tomlin, 1990), estatística espacial (Druck et al., 2004), modelagem numérica de terreno (Li et al., 2004) e processamento de imagens (Mather, 2004). Os mecanismos de visualização e plotagem devem oferecer suporte adequado para a apreensão cognitiva dos aspectos relevantes dos dados pesquisado (MacEachren, 2004) (Tufte, 1983) (Monmonier, 1993). No nível mais interno do sistema, um *sistema de gerência de bancos de dados geográficos* oferece armazenamento e recuperação dos dados espaciais e seus atributos. Cada sistema, em função de seus objetivos e necessidades, implementa estes componentes de forma distinta, mas todos os subsistemas citados devem estar presentes num SIG.

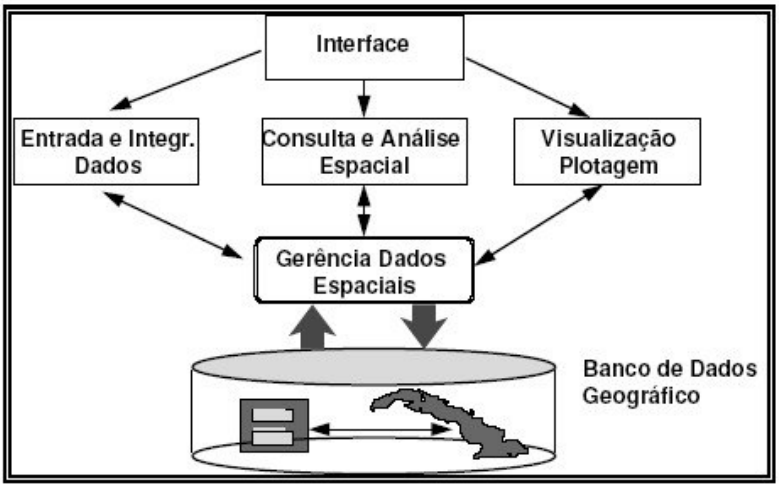


Figura 1.1 – Arquitetura de sistemas de informação geográfica

Do ponto de vista da aplicação, o uso de sistemas de informação geográfica (SIG) implica em escolher as representações computacionais mais adequadas para capturar a semântica de seu domínio de aplicação. Do ponto de vista da tecnologia, desenvolver um SIG significa oferecer o conjunto mais amplo possível de estruturas de dados e algoritmos capazes de representar a grande diversidade de concepções do espaço.

1.3 Traduzindo a informação geográfica para o computador

Para abordar o problema fundamental da Geoinformação, que é a *produção de representações computacionais do espaço geográfico*, usamos o *paradigma dos quatro universos*, proposto inicialmente por Gomes e Velho (Gomes and Velho) e adaptado para a geoinformação por Câmara (Câmara). Este paradigma distingue quatro passos entre o mundo real e sua realização computacional (ver Figura 1.2).

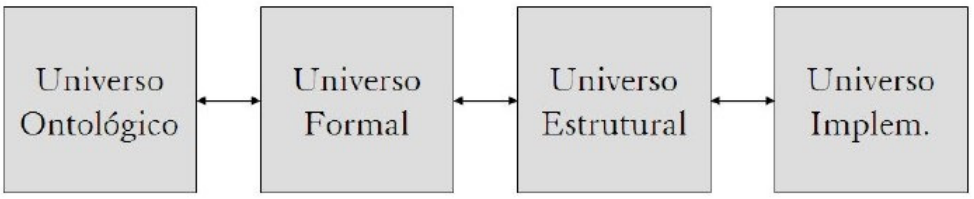


Figura 1.2 – Paradigma dos quatro universos

No primeiro passo, nossas percepções do mundo real são materializadas em conceitos que descrevem a realidade e respondem a questões como: *Que classes de entidades são necessárias para descrever o problema que estamos estudando?* (Smith, 2003). Criamos assim o *universo ontológico*, onde incluímos os conceitos da realidade a serem representados no computador, como os tipos de solo, elementos de cadastro urbano, e caracterização das formas do terreno.

O segundo universo (o *universo formal*) inclui modelos lógicos ou construções matemáticas que generalizam os conceitos do universo ontológico e dão resposta à pergunta: *Quais são as abstrações formais necessárias para representar os conceitos de nosso universo ontológico?* Estas abstrações incluem modelos de dados e álgebras computacionais. Exemplos: o modelo entidade-relacionamento (Chen, 1976) e o modelo OMT (Rumbaugh et al., 1991).

O terceiro universo é o *universo estrutural*, onde as diversas entidades dos modelos formais são mapeadas para estruturas de dados geométricas e alfanuméricas, e algoritmos que realizam operações. Neste universo, respondemos a questões como: *Quais são os tipos de dados e algoritmos necessários para representar os modelos e as álgebras do universo formal?* As estruturas de dados são os elementos básicos de construção dos sistemas computacionais.

O universo de *implementação* completa o processo de representação computacional. Neste universo, realizamos a implementação dos sistemas, fazendo escolhas como arquiteturas, linguagens e paradigmas de programação.

O paradigma dos quatro universos é uma forma de compreendermos que a transposição da realidade para o computador requer uma série complexa de mediações. Primeiro, precisamos dar nomes às entidades da realidade. Depois, geramos modelos formais que as descrevem de forma precisa. A seguir, escolhemos

as estruturas de dados e algoritmos que melhor se adaptam a estes modelos formais. Finalmente, fazemos a implementação num suporte computacional apropriado. Nas próximas seções, examinaremos em detalhe cada um destes universos.

1.4 O universo ontológico

Ontologia é o campo da filosofia cujo objetivo é descrever os tipos e estruturas de entidades, eventos, processos e relações que existem no mundo real (Smith, 2003). Sua gênese remonta a Aristóteles, mas o interesse recente por ontologias em sistemas de informação decorre principalmente da necessidade de compartilhar informação de forma eficiente para um público cada vez mais interdisciplinar.

Um sistema de informação pode ser concebido como um mecanismo de comunicação entre duas partes: o produtor e o usuário. Para que funcione, é necessário que haja uma concordância entre os conceitos das partes. Numa perspectiva mais geral, seu sucesso depende da existência de uma comunidade que compartilhe as definições utilizadas para construí-lo. Por exemplo, considere o caso de um estudo sobre *segregação* em áreas urbanas. Existem diferentes conceitos de segregação na literatura sociológica (Caldeira, 2000) (Massey and Denton, 1993) (Torres, 2004) (White, 1983). Para construir um sistema de informação que permita o estudo da segregação urbana, é preciso que o produtor de informação defina qual dos diferentes conceitos estará sendo representado, como esta representação será construída, e como o usuário pode compreender as características e limitações desta representação.

Deste modo, o problema fundamental de um sistema de informação é definir o conjunto de conceitos a ser representado. Se quisermos que estes conceitos sejam compartilhados por uma comunidade interdisciplinar, é fundamental que os conceitos utilizados sejam devidamente explicitados. Assim, surge a pergunta: "*Qual o papel dos conceitos na representação do mundo?*" A melhor forma de responder é baseando-se na perspectiva realista (Searle, 1998):

1. A realidade existe independentemente das representações humanas.
2. Nós temos acesso ao mundo através de nossos sentidos e de nossos instrumentos de medida.
3. As palavras em nossa linguagem podem ser usadas para referir-se a objetos do mundo real.
4. Nossas afirmações são verdadeiras ou falsas dependendo de sua correspondência aos fatos do mundo.
5. Algumas afirmações em nossa linguagem dizem respeito a uma realidade externa e independente ("*há neve no topo do Monte Evereste*"). Outras afirmações dizem respeito a convenções socialmente construídas ("*este papel é uma certidão de nascimento*").

Como nos ensina Searle (Searle), esta perspectiva tem conseqüências importantes sobre nossa concepção do mundo:

"*Apesar de termos representações mentais e lingüísticas do mundo sob a forma de crenças, experiências, afirmações, teorias, etc., há um mundo, 'lá fora', totalmente independente destas representações. A órbita elíptica dos planetas relativamente ao Sol e a estrutura do átomo de hidrogênio são inteiramente independentes das representações que os seres humanos têm de tais fenômenos. Já coisas como o*

dinheiro, a propriedade, o casamento e os governos são criados e sustentados pelo comportamento cooperativo humano."

"*Na sua maior parte, o mundo existe independentemente da linguagem (princípio 1) e uma das funções da linguagem é representar como são as coisas no mundo (princípio 3). Um aspecto crucial no qual a realidade e a linguagem entram em contato é marcado pela noção de verdade. Em geral, as afirmações são verdadeiras na medida em que representam com precisão uma característica da realidade que existe independentemente da afirmação (princípio 4).*".

O projeto de um sistema de informação requer, como passo inicial, a escolha das entidades a ser representados e, se possível, a descrição organizada destas entidades por meio de conceitos. Esta descrição forma uma *ontologia de aplicação*, definida como "*um conjunto de conceitos compartilhados por uma comunidade*" (Gruber, 1995). Para os dados geográficos, uma geo-ontologia tem dois tipos básicos de conceitos: (a) conceitos que correspondem a fenômenos físicos do mundo real; (b) conceitos que criamos para representar entidades sociais e institucionais (Smith and Mark, 1998) (Fonseca et al., 2003). Chamamos o primeiro tipo de *conceitos físicos* e o segundo de *conceitos sociais* (Tabela 1.1). Embora todos os conceitos resultem do uso compartilhado da linguagem, há uma diferença entre conceitos que se referem ao mundo físico ("*A Amazônia possui uma floresta tropical*") e aqueles que resultam de convenções humanas ("*Esta é uma reserva indígena*").

Nossa geo-ontologia diferencia entre conceitos associados a entidades que pode ser individualizadas e identificadas nominalmente (caso de *lagos e lotes*) e aquelas que variam de forma contínua no espaço (caso de *poluição*).

Tabela 1-1 – Tipos de conceitos associados a entidades geográficas

	Realidade física	*Realidade social*
Entidades individualizáveis	*indivíduos bona fide* (e.g., montanha)	*indivíduos fiat* (e.g., lote)
Entidades com variação contínua	*topografias físicas* (e.g., poluição)	*topografias sociais* (e.g., segregação)

Os conceitos físicos podem ser subdivididos em:

- Conceitos associados a entidades individualizáveis, que possuem uma fronteira bem definida a partir de diferenciações qualitativas ou descontinuidades na natureza. Designados como *indivíduos bona fide* (do latim "boa fé"), sua existência decorre de nossa necessidade de dar nomes aos elementos do mundo natural. Por exemplo, embora a superfície da Terra apresente uma variação contínua no espaço, nossa percepção do espaço depende da associação de nomes especiais a variações bem definidas no terreno. Daí nascem conceitos como *montanha, vale e desfiladeiro*.
- Conceitos associados a entidades que tem variação contínua no espaço, associadas aos fenômenos do mundo natural, não estando a princípio limitadas por fronteiras. Chamamos estes conceitos de topografias físicas, onde o termo "topografia" está associado a qualquer grandeza que varia

continuamente. Exemplos incluem *temperatura, altimetria, declividade e poluição*.

- Os conceitos sociais podem ser subdivididos em:
- Conceitos que descrevem entidades individuais criadas por leis e por ações humanas. Estas entidades possuem uma fronteira que as distingue do seu entorno e tem uma identidade única. Sua existência depende usualmente de um registro legal. Designadas como *indivíduos fiat* (do latim "fazer"), incluem conceitos como *lotes, municípios e países*.
- Conceitos descrevendo entidades que têm variação contínua no espaço, associadas a convenções sociais. Tome-se o caso de *pobreza*, conceito socialmente definido que ocorre no espaço de forma ininterrupta ("*em cada lugar há algum tipo diferente de pobreza*"). Chamamos estes conceitos de topografias sociais. Exemplos incluem: *exclusão social, segregação urbana, desenvolvimento humano*.

Uma geo-ontologia é um conjunto de conceitos e um conjunto de relações semânticas e espaciais entre estes termos. Cada conceito tem um nome, uma definição e um conjunto de atributos. O conjunto das relações semânticas inclui as relações de sinonímia, similaridade, e hiponímia (também dito especialização: "*hospital é um tipo de prédio*"). Por exemplo:

- rio: "Curso de água natural, de extensão mais ou menos considerável, que se desloca de um nível mais elevado para outro mais baixo, aumentando progressivamente seu volume até desaguar no mar, num lago, ou noutro rio".
- riacho: "rio pequeno, mais volumoso que o regato e menos que a ribeira".
- relação semântica: um riacho é *um* rio. (hiponímia).

O conjunto de relações espaciais inclui as relações topológicas como pertinência e adjacência, relações direcionais como "ao norte de", e relações informais como "no coração de" ou "perto de". Por exemplo:

- afluente: "curso de água que deságua em outro curso de água, considerado principal".
- relação espacial: um afluente *está conectado a* um rio.

Na maior parte dos sistemas de informação atuais, as ontologias de aplicação não estão explicitadas, o que reduz o potencial de compartilhamento da informação. Com o advento da Internet, que permite a disseminação de dados forma ampla e para um público heterogêneo, a necessidade de explicitar as ontologias utilizadas tornou-se ainda mais premente. A explicitação das ontologias de aplicação está na base das propostas recentes da "Web Semântica" (Berners-Lee et al., 2001) e de propostas de padrões como OWL. Como resultado de pesquisas recentes, já temos vários sistemas disponíveis na Internet para criação e gestão de ontologias, como o Protegé (Noy et al., 2001). Para dados geográficos, o consórcio OGC ("Open Geospatial Consortium") propôs o formato GML como mecanismo de descrição de ontologias geográficas.

1.5 O universo formal

O universo formal representa um componente intermediário entre os conceitos do universo ontológico e as estruturas de dados e algoritmos computacionais. Como os

computadores trabalham com estruturas matemáticas, a passagem direta de conceitos informais da ontologia de aplicação para estruturas de dados poderia gerar decisões inconsistentes. No universo formal, buscamos estabelecer um conjunto de entidades lógicas que agrupem os diferentes conceitos da ontologia de aplicação da forma mais abrangente possível. Adicionalmente, neste universo definimos ainda como serão associados valores aos diferentes conceitos; ou seja, como podemos medir o mundo real. Deste modo, o universo formal tem duas partes: (a) como medir o mundo real (teoria da medida); (b) como generalizar os conceitos da ontologia em entidades formais abrangentes. Estas duas partes serão discutidas a seguir.

1.5.1 Atributos de dados geográficos: teoria da medida

Para representar dados geográficos no computador, temos de descrever sua variação no espaço e no tempo. Em outras palavras, precisamos poder a perguntas como: "*qual é o valor deste dado aqui e agora?*". Isto requer uma compreensão dos processos de mensuração da realidade, de forma consistente com os dois primeiros princípios de Searle (Searle): "*a realidade existe independentemente das representações humanas*" e "*nós temos acesso ao mundo através de nossos sentidos e de nossos instrumentos de medida*". O processo de medida consiste em associar números ou símbolos a diferentes ocorrências de um mesmo atributo, para que a relação dos números ou símbolos reflita as relações entre as ocorrências mensuradas. Por exemplo, podemos medir a poluição numa cidade através de sensores localizados em diferentes locais. Cada um destes sensores nos dará uma medida diferente. Esta atribuição é denominada escala de medida. A referência geral mais importante sobre escalas de medidas é o trabalho de Stevens (Stevens), que propõe quatro escalas de mensuração: *nominal, ordinal, intervalo e razão*.

Os níveis nominal e ordinal são temáticos, pois a cada medida é atribuído um número ou nome associando a observação a um tema ou classe. A *escala nominal* classifica objetos em classes distintas sem ordem inerente, como rótulos que podem ser quaisquer símbolos. As possíveis relações entre os valores são identidade ($a = b$) e dessemelhança ($a \neq b$). Um exemplo é a cobertura do solo, com rótulos como "*floresta*", "*área urbana*" e "*área agrícola*".

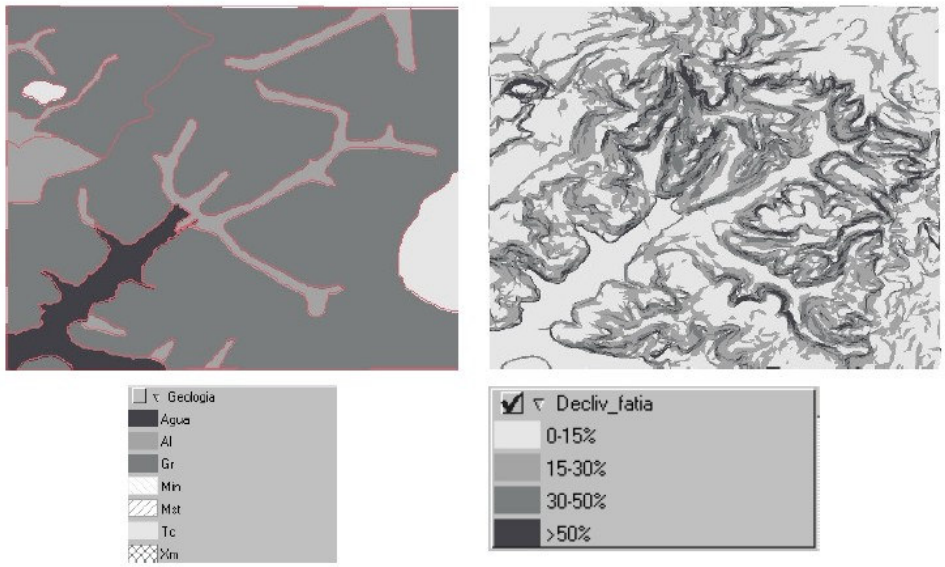


Figura 1.3 – Exemplos de medida nominal (mapa geológico) e medida ordinal (mapa de classes de declividade) .

A *escala ordinal* introduz a idéia de ordenação, caracterizando os objetos em classes distintas que possuem uma ordem natural (por exemplo 1 – ruim, 2 – bom, 3 – ótimo ou "0-10%", "11-20%", "mais que 20%"). A distância definida entre os elementos não é significativa. Nesta escala são evidenciadas as relações "<" ou ">", isto implica que para todo a e b, as relações a < b, a > b ou a = b são possíveis. Um exemplo é a aptidão agrícola de solos, com rótulos como "muito apto", "apto", "pouco apto", e "inapto" (ver Figura 1.3).

As medidas temáticas não estão associadas à magnitude do fenômeno. Quando o estudo necessita de uma descrição mais detalhada, que permita comparar intervalo e ordem de grandeza entre eventos, recorre-se aos níveis de medidas denominados de *numéricos*, onde as regras de atribuição de valores baseiam-se em uma escala de números reais.

Existem dois níveis de medidas baseados em escalas de números reais: *escala por intervalo* e o *escala por razão*. A *escala por intervalo* possui um ponto zero arbitrário, uma distância proporcional entre os intervalos e uma faixa de medidas entre [-∞ , ∞]. A temperatura em graus Celsius é exemplo de medida por intervalo, onde o ponto zero corresponde a uma convenção (a fusão do gelo em água). Por ter uma referência zero arbitrária, valores medidos no nível por intervalo não podem ser usados para estimar proporções. Operações aritméticas elementares (adição e subtração) são válidas, porém multiplicação e divisão não são apropriadas. Por exemplo, dados *a* e *b*, pode-se ter $a = b + c$, onde *c* é a diferença entre *a* e *b* em alguma unidade padrão. Assim, a temperatura em São Paulo pode ser *c* graus mais baixa do que a temperatura em Campos de Jordão.

A *escala de razão* permite um tratamento mais analítico da informação, pois nela o ponto de referência zero não é arbitrário, mas determinado por alguma condição natural. Sua faixa de valores é limitada entre [0, ∞]. Nesta escala existe um ponto zero absoluto que não pode ser alterado e um intervalo arbitrário com distâncias proporcionais entre os intervalos. Números negativos não são permitidos, pois o número zero representa ausência total daquilo que está sendo medido. Por exemplo, na descrição de atributos como peso e volume de objetos não há valores negativos. No caso de temperatura em graus Kelvin, a condição natural é o ponto de

repouso dos átomos da matéria, a partir do qual não se consegue temperaturas menores. Este ponto é o zero absoluto para temperatura, zero graus Kelvin. O fato de ponto de referência zero ser absoluto permite afirmações tais como *a* é duas vezes mais pesado que *b*. Desta forma, dado *a* e *b* pode-se ter $a = c \times b$, onde *c* indica o número de vezes que *b* vai até *a*, a relação de *a* para *b*. Operações matemáticas de adição, subtração, multiplicação e divisão são suportadas nesta escala.

A Tabela 1.2 apresenta um resumo das escalas de medidas, destaca a característica principal, apresenta algumas operações admitidas e exemplos para cada uma delas.

Tabela 1-2 - Tipos de medidas de dados geográficos

Escala	*Características*	*Exemplos*	*Operações possíveis*
Nominal	Descrição	Tipo de solo, vegetação, uso do solo	Seleção, Comparação
Ordinal	Ordem	Classes de declividade, aptidão de uso	Mediana, Máximo, Mínimo
Intervalo	Distância	Altimetria	Diferença, Soma
Razão	Valores absolutos	Renda, população, taxa de natalidade	Operações aritméticas

1.5.2 Espaço absoluto e espaço relativo

Antes de considerar os diferentes modelos formais para dados geográficos, é necessário analisarmos brevemente os conceitos de *espaço absoluto* e *espaço relativo*. Esta distinção decorre da possibilidade de representarmos no computador a localização dos objetos no espaço ou apenas o posicionamento relativo entre eles, como ilustrado na Figura 1.4. Nesta figura, mostramos à esquerda os distritos da cidade de São Paulo, identificados por suas fronteiras. Neste caso, trata-se de uma representação no espaço absoluto, na qual as coordenadas das fronteiras devem corresponder às estabelecidas na legislação. Do lado direito, mostramos um grafo com as conexões dos distritos, que formam uma rede (repetimos a imagem dos distritos por razões de melhor legibilidade da figura). No modelo de redes, a localização exata de cada distrito não é armazenada, pois a rede só captura as relações de adjacência. Dizemos então que a rede de conexões dos distritos é um modelo de *espaço relativo*.

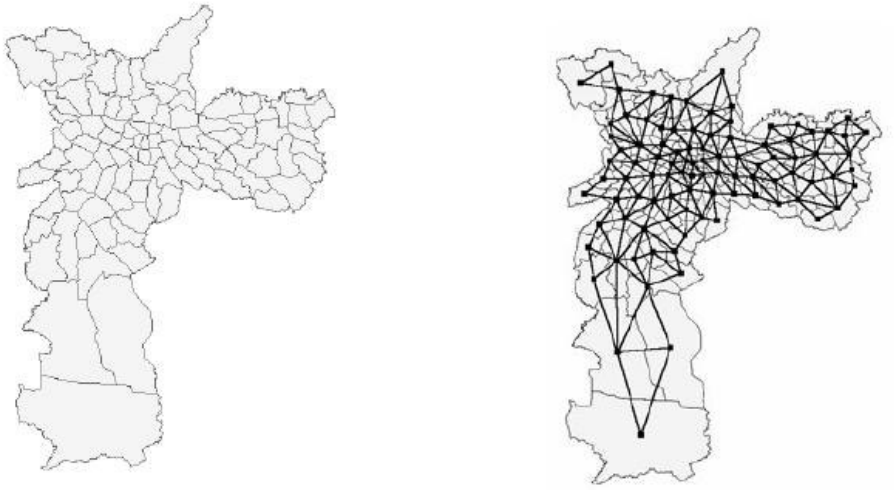


Figura 1.4 - Dualidade entre espaço absoluto e espaço relativo. À esquerda, distritos de São Paulo com suas fronteiras. À direita, grafo mostrando a rede de conectividade entre os distritos (espaço relativo). O mapa da esquerda foi repetido por razões de melhor legibilidade.

A distinção entre espaço absoluto e espaço relativo é de grande importância para a Geografia. Milton Santos (Santos) refere-se ao "espaço dos fixos" e ao "espaço dos fluxos". Castells (Castells) fala em "espaço de lugares" e "espaços de fluxos". Vejam o que Helen Couclelis comenta a respeito do tema:
"*Espaço absoluto, também chamado cartesiano, é um container de coisas e eventos, uma estrutura para localizar pontos, trajetórias e objetos. Espaço relativo, ou leibnitziano, é o espaço constituído pelas relações espaciais entre coisas"* (Couclelis, 1997)

Uma das escolhas básicas que fazemos na modelagem dos fenômenos geográficos é definir se utilizaremos representações no espaço absoluto ou no espaço relativo. Esta escolha depende primordialmente do tipo de análise que queremos realizar. Usualmente, consultas espaciais que envolvem dois tipos de entidades ("*quais os rios que cruzam esta estação ecológica?*") requerem a representação no espaço absoluto. O mesmo vale para questões de álgebra de mapas ("*áreas inaptas tem declividade maior que 15% ou solos arenosos*"). Quando os procedimentos de análise envolvem apenas as relações de conectividade ("*como chegar na estação de metrô Clínicas, partindo da estação Liberdade?*" ou "*qual é a média da mortalidade infantil de meus vizinhos?*") podemos utilizar representações no espaço relativo. Quando falamos em entidades como estradas, linhas de transmissão, conexões de água e esgoto, cadeias de mercado e linhas de comunicação, o espaço relativo é na maioria das vezes plenamente adequado.

1.5.3 Modelos no espaço absoluto: geo-campos e geo-objetos

Existem dois modelos formais para entidades geográficos no espaço absoluto: *geo-campos* e *geo-objetos*. O modelo de geo-campos enxerga o espaço geográfico como uma superfície contínua, sobre a qual variam os fenômenos a serem observados. Por exemplo, um mapa de vegetação associa a cada ponto do mapa um tipo específico de cobertura vegetal, enquanto um mapa geoquímico associa o teor de um mineral a cada ponto. O modelo de geo-objetos representa o espaço geográfico como uma coleção de entidades distintas e identificáveis, onde cada entidade é definida por uma

fronteira fechada. Por exemplo, um cadastro urbano identifica cada lote como um dado individual, com atributos que o distinguem dos demais.

Definição 1.1. Geo-Campo. Um geo-campo representa um atributo que possui valores em todos os pontos pertencentes a uma região geográfica. Um geo-campo *gc* é uma relação $gc = [R, A, f]$, onde $R \subset \Re^2$ é uma partição conexa do espaço, A é um atributo cujo domínio é $D(A)$, e a função de atributo $f: R \rightarrow A$ é tal que, dado $p \in R$, $f(p) = a$, onde $a \in D(A)$.

A noção de geo-campo decorre da definição física associada (segundo o Aurélio, "campo é um conjunto de valores de uma grandeza física que, numa região do espaço, dependem só das coordenadas dos pontos pertencentes a essa região"). Em outras palavras, para cada ponto do espaço, um campo terá um valor diferente.

Definição 1.2 Geo-Objeto. Um geo-objeto é uma entidade geográfica singular e indivisível, caracterizada por sua identidade, suas fronteiras, e seus atributos. Um *geo-objeto* é uma relação $go = [id, a_1,...a_n, G]$, onde *id* é um identificador único, G é um conjunto de partições 2D conexas e distintas $\{R_1,...,R_n\}$ do espaço $\Re^2$, e a_i são os valores dos atributos $A_1,...,A_n$. Note-se que um geo-objeto pode ser composto por diferentes geometrias, onde cada geometria tem uma fronteira fechada (e.g., o Japão com suas diferentes ilhas).

Um exemplo de geo-campo (uma imagem IKONOS da cidade do Rio de Janeiro) e de um conjunto de geo-objetos (os distritos dessa cidade) é apresentado na Figura 1.5. A variável associada à imagem é a reflectância do solo, medida pelo sensor óptico do satélite. Os geo-objetos associados aos distritos de São Paulo são mostrados numa gradação de tons de cinza, cuja intensidade é proporcional ao índice de exclusão social (Sposati, 1996); quanto mais escuro, mais o distrito possui moradores em situação de exclusão social. Os dados na Figura 1.3 acima (geologia e declividade) também são exemplos de geo-campos.

A Figura 1.5 também ilustra uma questão importante: *existem diferenças fundamentais entre geo-campos e geo-objetos? Ou seriam apenas duas maneiras de ver o mesmo tipo de dado?* Considere os retângulos desenhados no interior das duas representações mostradas. Na figura à esquerda, o interior do retângulo tem as mesmas propriedades do geo-campo que o contém. Para cada ponto interior ao retângulo, podemos recuperar o valor do atributo (neste caso, a reflectância da imagem). Verificamos que uma partição espacial genérica de *um geo-campo compõe outro geo-campo com as mesmas propriedades*.

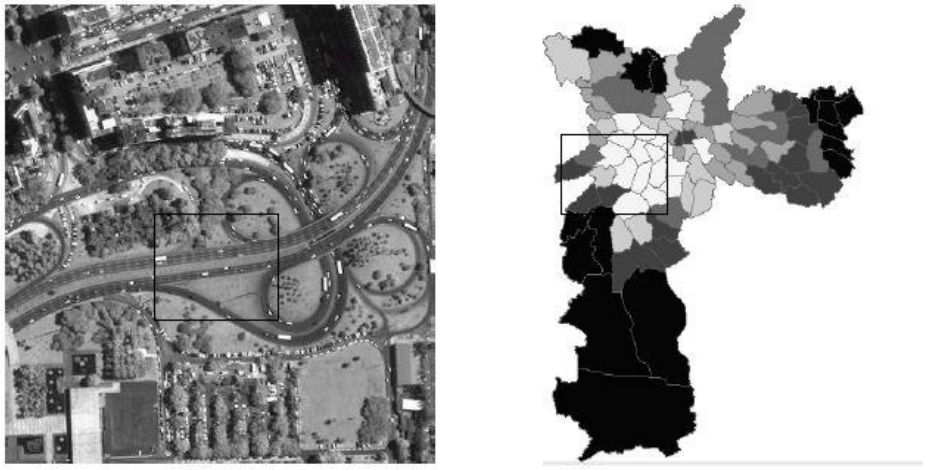


Figura 1.5 - Exemplo de geo-campo (imagem IKONOS do Rio de Janeiro) e de

conjunto de geo-objetos (distritos da cidade de São Paulo).

Considere agora a figura da direita (distritos de São Paulo). O interior do retângulo mostrado não define mais um conjunto de geo-objetos com as mesmas propriedades do conjunto completo. O retângulo intercepta parcialmente alguns objetos. Como cada objeto é único e não pode ser dividido sem perder suas características originais, verificamos que *uma partição espacial genérica de um conjunto de geo-objetos não compõe outro conjunto de geo-objetos com as mesmas propriedades*.

A diferença essencial entre um geo-campo e um geo-objeto é o papel da fronteira. A fronteira de um geo-campo é uma divisão arbitrária relacionada apenas com nossa capacidade de medida. Na Figura 1.5, os limites da imagem correspondem apenas a eventuais limitações do instrumento sensor e não do fenômeno medido. Assim, o geo-campo pode ser divido em partes e ainda assim manter sua propriedade essencial (que é sua função de atributo).

Por contraste, um geo-objeto é essencialmente definido por sua fronteira, que o separa do mundo exterior; ele não pode ser dividido e manter suas propriedades essenciais. Dentro da fronteira, todas as propriedades do objeto são constantes. Tomemos um distrito de São Paulo, como a Sé, que tem um código único de identificação no censo do IBGE. Se dividirmos a Sé em duas partes, precisamos de dois novos códigos de identificação para caracterizar os dois novos distritos.

O exame da Figura 1.5 ilustra outra propriedade dos geo-objetos. É bastante comum lidarmos com um conjunto de geo-objetos que representam uma partição consistente do espaço; isto é, os recobrimentos espaciais destes objetos não se interceptam e eles possuem o mesmo conjunto de atributos. Estas características fazem com que possamos agrupar estes objetos numa coleção.

Definição 1.3 Coleção de geo-objetos. Uma coleção de geo-objetos é relação $cgo = [id, o_1,...o_n, A_1,..., A_n]$, onde *id* é um identificador único, e $o_1,...o_n$ são geo-objetos que possuem os atributos $A_1,..., A_n$. Usualmente, se R_i for a região geográfica associada a o_i, temos $R_i \cap R_j = \emptyset$, $\forall\ i \neq j$. Deste modo, uma coleção reúne geo-objetos cujas fronteiras não se interceptam, e têm o mesmo conjunto de atributos.

O uso de coleções de geo-objetos é bastante freqüente em bancos de dados geográficos, pois é muito conveniente tratar geo-objetos similares de forma consistente. Por exemplo, falamos dos distritos da cidade de São Paulo, dos municípios do estado do Ceará, e das reservas indígenas da Amazônia. A idéia de coleções de geo-objetos é ainda útil para propormos um modelo orientado-a-objetos para dados geográficos, discutido a seguir.

1.5.4 Modelos no espaço relativo: redes

O modelo de redes concebe o espaço geográfico como um conjunto de pontos no espaço (chamados de nós), conectados por linhas (chamados arcos), onde tanto os nós quanto os arcos possuem atributos. Os fenômenos modelados por redes incluem fluxo de pessoas ou materiais, conexões de influência, linhas de comunicação e acessibilidade. Um dos atrativos do modelo de redes é que o suporte matemático para este modelo (*a teoria de grafos*) é uma área de pesquisa consolidada (Bondy and Murty, 1977) (Gross and Yellen, 1998).

O problema que deu início à teoria dos grafos foi uma questão espacial. Em 1736, o matemático Leonard Euler vivia na cidade de Königsberg (na época parte da Prússia; hoje chamada Kaliningrad e pertencente à Rússia) onde haviam duas ilhas

próximas no meio da cidade, cruzadas por sete pontes (ver Figura 1.6 à esquerda). Euler se perguntou se havia uma maneira de fazer um circuito fechado (sair e voltar para um mesmo lugar), cruzando cada uma das pontes apenas uma vez. Ele construiu um grafo equivalente (ver Figura 1.6 à direita) e demonstrou que o problema era insolúvel.

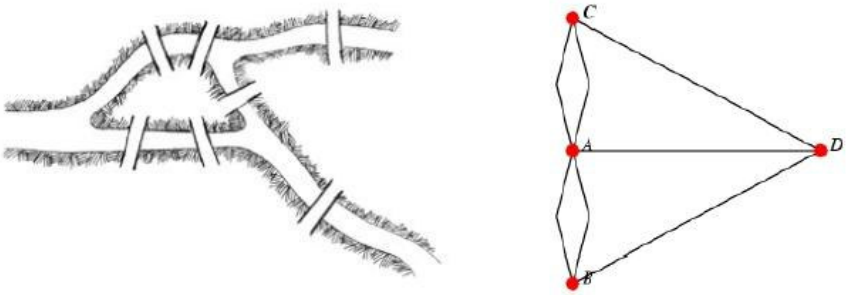


Figura 1.6 - As sete pontes de Königsberg e o grafo equivalente.

Definição 1.4 Redes. Uma rede é uma estrutura geográfica que tem como suporte um grafo $G = [N, A, \varphi]$, onde N é um conjunto de nós, A é um conjunto de arcos (arestas), e $\varphi(a)=(u,v)$ é uma função de incidência que associa cada arco $a \in A$ a um par de nós $(u, v) \in N$. No caso geográfico, os nós podem estar associados a uma localização (x,y) do espaço para fins de referência.

Como os nós de uma rede são abstrações de entidades existentes no espaço, eles podem estar associados aos seus atributos descritivos. Por exemplo, na rede mostrada na Figura 1.4, cada nó está associado a um distrito de São Paulo, e poderia ter diferentes atributos que descrevem este distrito. Também os arcos de uma rede podem ter propriedades, como o custo de percorrimento de um nó a outro. As propriedades mensuráveis das redes incluem operações diretas computáveis sobre a topologia do grafo, como qual o caminho ótimo entre dois nós. Também podemos computar operações matemáticas que envolvem apenas as relações de conectividade, como os indicadores locais de autocorrelação espacial (Druck et al., 2004).

A definição de redes pode ser estendida para considerar o caso de conexões bidirecionais, como no caso de redes de transporte, onde as relações entre os nós não são simétricas, pois os fluxos em sentidos opostos podem ser diferentes. A Figura 1.7 ilustra uma rede simples e uma rede com conexões bidirecionais.

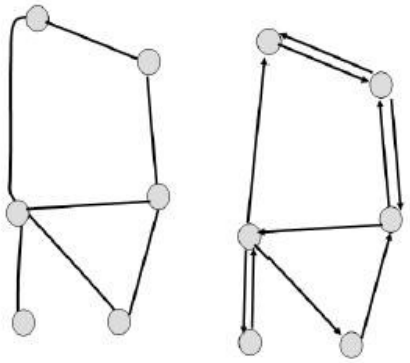


Figura 1.7 – Exemplos de redes simples e de redes com conexões bidirecionais.

Os modelos de rede têm grande utilidade em problemas de geoinformação, incluindo assuntos como gerenciamento de serviços como água, esgoto, eletricidade e telefonia. Para maiores referências, deve-se consultar Birkin et al (Birkin et al.) e Godin (Godin).

1.5.5 Um modelo orientado-a-objetos para dados geográficos

As seções anteriores nos permitem apresentar um modelo orientado-a-objetos que apresenta uma versão unificada dos dados geográficos, com base nos conceitos básicos de *geo-campo*, *coleção de geo-objetos* e *rede*. Para fins de organização lógica, o modelo considera a existência de uma classe genérica, chamada de *plano de informação* (ou *layer*), que é uma generalização destes dois conceitos. O conceito de *plano de informação* captura uma característica comum essencial dos três conceitos básicos: cada instância deles é referente a uma localização no espaço e tem um identificador único. Assim, o uso do conceito de *plano de informação* permite organizar o banco de dados geográfico e responder a perguntas como: "*Quais são os dados presentes no banco, qual o modelo associado a cada um e qual a região geográfica associada?*" Adicionalmente, como cada geo-campo está associado a uma única função de atributo, ele pode ser especializado em *geo-campo temático* (associado a medidas nominais ou ordinais) e *geo-campo numérico* (associados a medidas por intervalo ou por razão). Com estes seis conceitos, construímos um modelo formal básico para dados geográficos, mostrado na Figura 1.8.

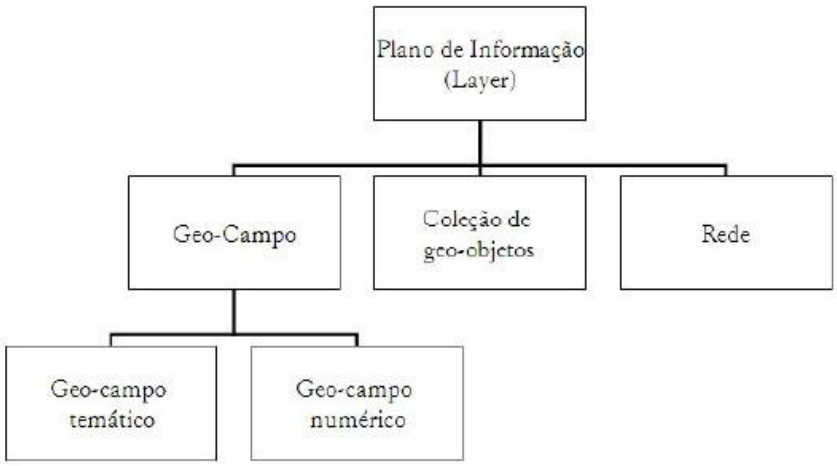


Figura 1.8 – Modelo OO básico para dados geográficos.

O modelo mostrado na Figura 1.8 serve de base para a maioria dos modelos de dados orientados-a-objetos adotados atualmente em geoinformação:

- O software SPRING (Câmara et al., 1996) inclui os conceitos de *rede*, *geo-campo numérico* e *geo-campo temático*, *coleção de geo-objetos* (chamada de *mapa cadastral*). Os geo-campos numéricos admitem as *imagens* como caso particular.
- No ArcGIS (ESRI, 2000b), a coleção de geo-objetos é chamada de *features* (*feições*). Os geo-campos numéricos são chamados de *surfaces* (*superfícies*), e as imagens também são modeladas como caso particular de geo-campos numéricos. As redes (*networks*) também são incluídas.

- No modelo OpenGIS (OGC, 1998), os geo-campos são chamados de *coverage*, e a coleção de objetos é chamada de *feature collection*. O modelo OpenGIS não tem o conceito explícito de *layer*, mas considera que as visões de *feature collection* e *coverage* são complementares.
- Na TerraLib, o conceito de plano de informação (*layer*) é um conceito usado para organizar a informação no banco de dados. Os conceitos de *geo-campos* e de coleções de *geo-objetos* são implícitos. Como se trata de uma biblioteca, os designers da TerraLib quiseram permitir diferentes alternativas de projeto de sistema.

1.6 Do universo ontológico ao universo formal

Para passar do universo ontológico para o universo formal, precisamos responder à pergunta: *como os conceitos da ontologia de aplicação são formalizados?* Colocando o problema de forma mais geral: *Que critérios deve satisfazer um conceito para que seja utilizável em estudos quantitativos associados à geoinformação?* Tais critérios são:

- O conceito deve ser passível de ser associado a propriedades mensuráveis.
- Estas propriedades devem ser medidas no território e devem permitir diferenciar as diferentes localizações.
- Os resultados quantitativos e os modelos matemáticos utilizados devem ser validados em estudos de campo, que devem incluir dimensões objetivas e subjetivas do fenômeno em questão.

Para representar um conceito genérico como "exclusão social", precisamos definir precisamente quais atributos caracterizam a exclusão social e como podemos medi-los no território. Esta caracterização realiza a passagem do universo ontológico para o universo formal. Com base em conceitos bem estabelecidos e associados a medidas quantitativas no espaço, podemos construir *territórios digitais*. O processo pode ser resumido na Figura 1.9.

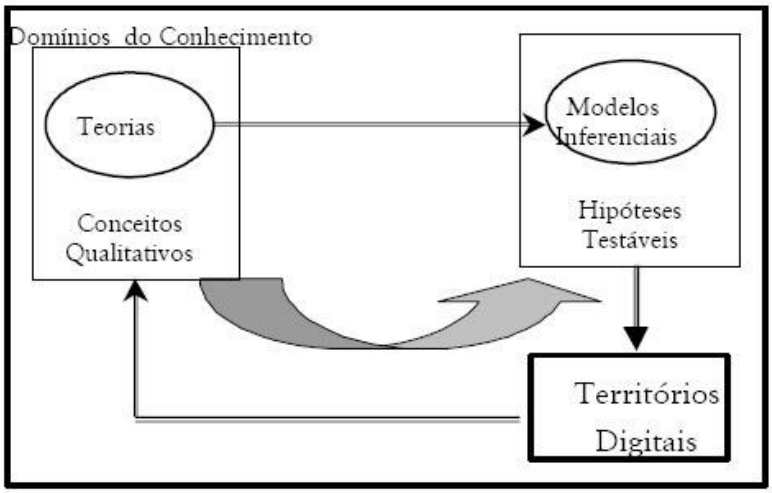


Figura 1.9 – Relação entre a construção dos territórios digitais e as teorias disciplinares (cortesia de Silvana Amaral Kampel).

Os especialistas desenvolvem teorias gerais sobre os fenômenos, que incluem o estabelecimento de conceitos organizadores de sua pesquisa (como "exclusão" ou "vulnerabilidade"). Para passar destas teorias para a construção computacional, é

necessário que o especialista formule modelos inferenciais quantitativos. Estes modelos devem ser submetidos a testes de validação e de corroboração, através dos procedimentos de análise quantitativa. Os resultados numéricos podem então dar suporte ou ajudar a rejeitar conceitos qualitativos.

Após definir como que atributos mensuráveis serão associados ao conceito, o projetista do sistema de informação deverá decidir se este conceito será modelado no *espaço absoluto* ou no *espaço relativo*. A decisão deve-se dar essencialmente em função das propriedades que queremos medir. Se a localização exata é fundamental, ou se precisamos saber o valor do fenômeno em todos os pontos da região de estudo, então é necessário usar os modelos de espaço absoluto. Se o fluxo e as conexões são essenciais, então podemos usar o modelo de rede.

Se precisamos dos dados expressos no espaço absoluto, então devemos escolher ainda qual o modelo apropriado (geo-campo ou geo-objeto). Para isto, a decisão depende essencialmente do papel da fronteira. Se as fronteiras são parte essencial das entidades modeladas, estamos tratando com *indivíduos* e não com *topografias* (vide Tabela 1.1) e o modelo de geo-objetos é o mais adequado. Senão, usaremos os modelos de geo-campos.

1.7 Universo estrutural

As estruturas de dados utilizadas em bancos de dados geográficos podem ser divididas em duas grandes classes: *estruturas vetoriais* e *estruturas matriciais*.

1.7.1 Estruturas de dados vetoriais

As estruturas vetoriais são utilizadas para representar as coordenadas das fronteiras de cada entidade geográfica, através de três formas básicas: pontos, linhas, e áreas (ou polígonos), definidas por suas coordenadas cartesianas, como mostrado na Figura 1.10.

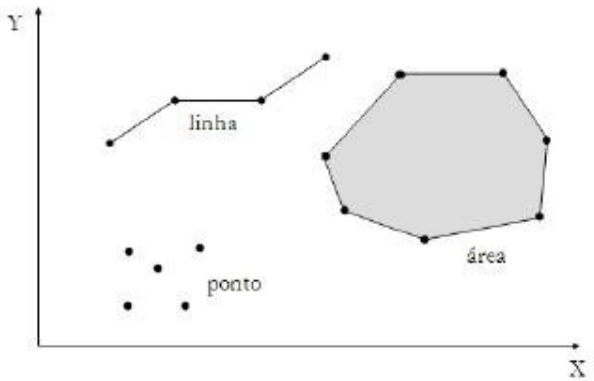


Figura 1.10 – Representações vetoriais em duas dimensões.

Um *ponto* é um par ordenado *(x, y)* de coordenadas espaciais. O ponto pode ser utilizado para identificar localizações ou ocorrências no espaço. São exemplos: localização de crimes, ocorrências de doenças, e localização de espécies vegetais. Uma *linha* é um conjunto de pontos conectados. A linha é utilizada para guardar feições unidimensionais. De uma forma geral, as linhas estão associadas a uma topologia arco-nó, descrita a seguir. Uma *área* (ou *polígono*) é a região do plano limitada por uma ou mais linhas poligonais conectadas de tal forma que o último

ponto de uma linha seja idêntico ao primeiro da próxima. Observe-se também que a fronteira do *polígono* divide o plano em duas regiões: o interior e o exterior. Os polígonos são usados para representar unidades de dados geográficos espaciais individuais (setores censitários, distritos, zonas de endereçamento postal, municípios). Para cada unidade, são associados dados oriundos de levantamentos como censos e estatísticas de saúde.

1.7.2 Vetores e topologia: o caso dos geo-objetos

A topologia é a parte da matemática na qual se investigam as propriedades das configurações que permanecem invariantes nas transformações de rotação, translação e escala. No caso de dados geográficos, é útil ser capaz de determinar relações como adjacência ("vizinho de"), pertinência ("vizinho de"), intersecção, e cruzamento.

Objetos de área podem ter duas formas diferentes de utilização: como objetos *isolados* ou objetos *adjacentes*. O caso de objetos isolados é bastante comum em SIG urbanos, e ocorre no caso em que os objetos da mesma classe em geral não se tocam. Por exemplo, edificações, piscinas, e mesmo as quadras das aplicações cadastrais ocorrem isoladamente, não existindo segmentos poligonais compartilhados entre os objetos. Finalmente, temos objetos adjacentes, e os exemplos típicos são todas as modalidades de divisão territorial: bairros, setores censitários, municípios e outros. Neste caso, pode-se ter o compartilhamento de fronteiras entre objetos adjacentes, gerando a necessidade por estruturas topológicas. Estes também são os casos em que recursos de representação de buracos e ilhas são mais necessários.

Quando queremos armazenar as estruturas de dados do tipo polígono no caso de *objetos adjacentes*, temos uma decisão básica a tomar: guardamos as coordenadas de cada objeto isoladamente, e assim duplicamos as fronteiras em comum com outros objetos, ou armazenamos cada fronteira comum uma única vez, indicando a que objetos elas estão associadas? No primeiro caso é chamado de *polígonos sem topologia* e o segundo, de *topologia arco-nó-polígono*, comparados na Figura 1.11.

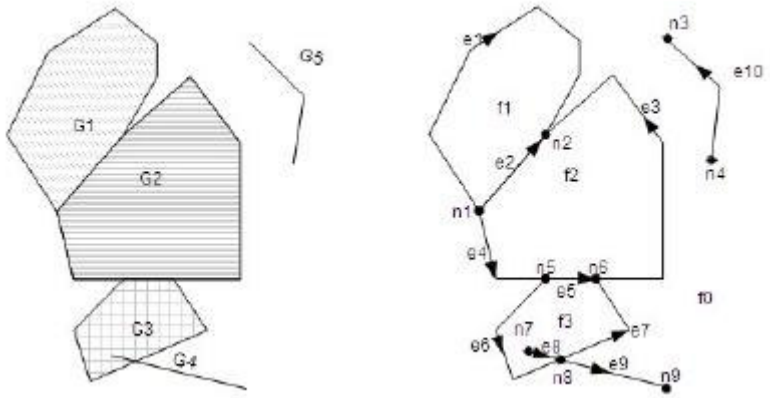
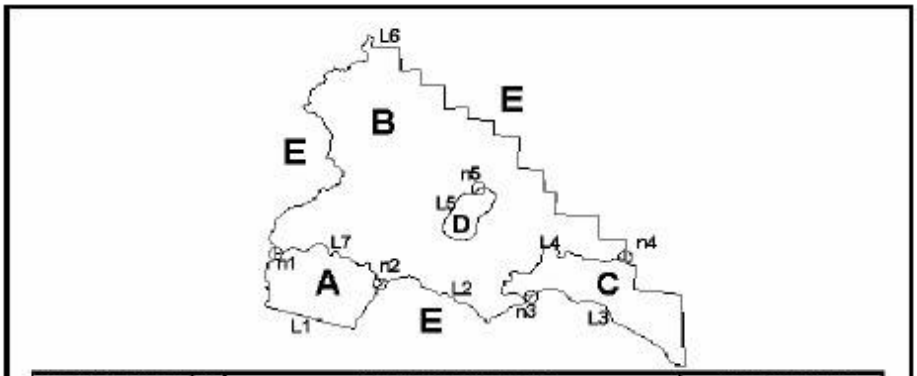


Figura 1.11 – Polígonos sem topologia (à esquerda) e topologia arco-nó-polígono (à direita). (Fonte: Ravada, 2003).



TOPOLOGIA DOS NÓS		TOPOLOGIA DAS LINHAS					TOPOLOGIA DOS POLÍGONOS	
NÓ	LINHAS	LINHAS	NÓ INICIAL	NÓ FINAL	POLÍGONO ESQUERDO	POLÍGONO DIREITO	LINHAS	POLÍGONO
n1	L1, L7, L6	L1	n1	n2	E	B	L1, L7	A
n2	L1, L2, L7	L2	n2	n3	E	B	L2, L4, L6,L7	B
n3	L2, L3, L4	L3	n3	n4	E	C	L4, L3	C
n4	L3, L4, L6	L4	n3	n4	B	C	L5	D
n5	L5	L5	n5	n5	B	D		
		L6	n4	n1	B	E		
		L7	n1	n2	A	B		

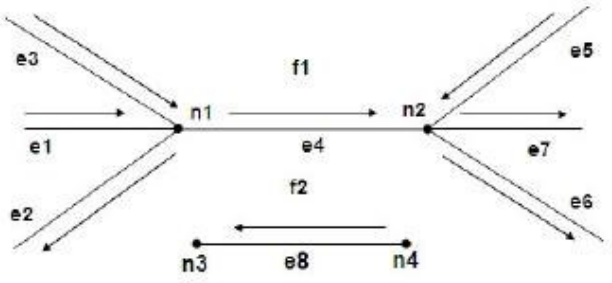
Figura 1.12 – Topologia arco-nó-polígono.

A topologia arco-nó-polígono, como mostrado na Figura 1.12, requer três listas separadas. Os pontos inicial e final de cada linha são chamados de nós. Para cada nó, armazenamos as linhas nele incidentes. Para cada linha, armazenamos os nós inicial e final, permitindo assim que a linha esteja associada a um sentido de percorrimento; guardamos ainda os dois polígonos separados por cada linha (à esquerda e à direita, considerando o sentido de percorrimento). Para cada polígono, guardamos as linhas que definem sua fronteira.

1.7.3 Vetores e topologia: o caso das redes

Objetos de linha podem ter variadas formas de utilização. Analogamente aos objetos de área, podemos ter objetos de linha isolados, em árvore e em rede. Objetos de linha isolados ocorrem, por exemplo, na representação de muros e cercas em mapas urbanos. Objetos de linha organizados em uma árvore podem ser encontrados nas representações de rios e seus afluentes, e também em redes de esgotos e drenagem pluvial. E podem ser organizados em rede, nos casos de redes elétricas, telefônicas, de água ou mesmo na malha viária urbana e nas malhas rodoviária e ferroviária.

No caso das redes, é fundamental armazenar explicitamente as relações de adjacência, utilizamos a *topologia arco-nó*. Um *nó* pode ser definido como o ponto de intersecção entre duas ou mais linhas, correspondente ao ponto inicial ou final de cada linha. Nenhuma linha poderá estar desconectada das demais para que a topologia da rede possa ficar totalmente definida. Para exemplificar, considere-se a Figura 1.13, que mostra um exemplo de como a topologia arco-nó pode ser armazenada.



EDGE_ ID	START _NODE _ID	END_ NODE_ ID	NEXT_ LEFT_ EDGE_ ID	PREV_ LEFT_ EDGE_ ID	NEXT_ RIGHT _EDGE _ID	PREV_ RIGHT _EDGE _ID	LEFT_ FACE_ ID	RIGHT _FACE _ID	GEOM ETRY
E4	N1	N2	-E5	E3	E2	-E6	F1	F2	(...)
E8	N4	N3	-E8	-E8	E8	E8	F2	F2	(...)

Figura 1.13 – Estrutura de dados para topologia arco-nó no Oracle Spatial SGBD (Fonte: Ravada, 2003).

1.7.4 Vetores e topologia: o caso dos dados 2,5 D

Uma das possibilidades associadas a dados vetoriais é a associação de valores que denotem a variação espacial de uma grandeza numérica. No caso mais simples, associamos a cada localização no espaço um valor numérico de atributo. Neste caso, como os valores de localização estão no plano e o valor adicional descreve uma superfície sobre este plano. Os dados resultantes são chamados de dimensão "dois e meio", pois não se tratam estritamente de dados tridimensionais, pois o suporte espacial ainda são localizações 2D. A Figura 1.14 ilustra exemplo de dados de dimensão 2,5.

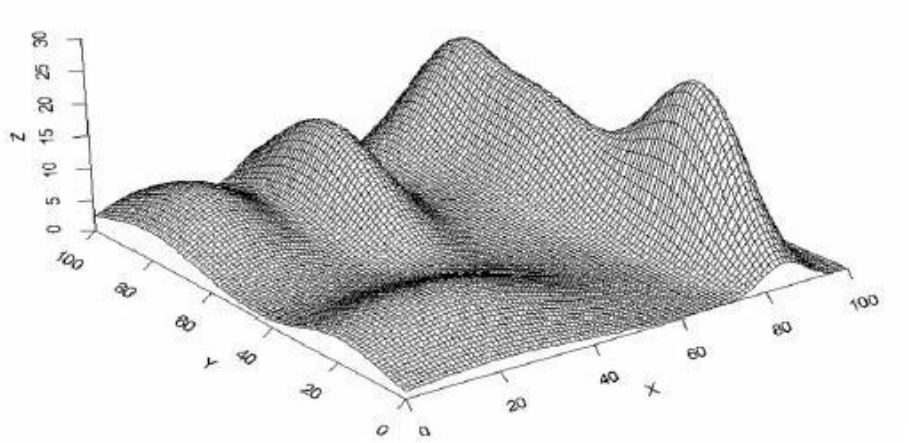


Figura 1.14 – Exemplo de dado com dimensão 2,5 (cortesia de Renato Assunção).

A maneira mais comum de armazenar estes dados é através de estruturas matriciais (vide próxima seção). Temos três alternativas que usam estruturas vetoriais:

- Conjunto de amostras esparsas 2,5D, constituído de pares ordenados *(x,y,z)*, onde *(x,y)* é uma localização no plano e z um valor numérico de atributo.
- Conjunto de isolinhas (curvas de nível), que são linhas às quais estão associados valores numéricos. As isolinhas não se cruzam, e são entendidas como estando "empilhadas" umas sobre as outras.
- A *malha triangular* ou TIN (do inglês "triangular irregular network") é uma estrutura do tipo vetorial com topologia do tipo *nó-arco* e representa uma superfície através de um conjunto de faces triangulares interligadas.

A malha triangular é a estrutura vetorial mais utilizada para armazenar dados 2,5D. Cada um dos três vértices da face do triângulo armazenados as coordenadas de localização *(x, y)* e o atributo *z*, com o valor de elevação ou altitude. Em geral, nos SIGs que possuem pacotes para MNT, os algoritmos para geração da malha triangular baseiam-se na triangulação de Delaunay com restrição de região. Quanto mais equiláteras forem as faces triangulares, maior a exatidão com que se descreve a superfície. O valor de elevação em qualquer ponto dentro da superfície pode ser estimado a partir das faces triangulares, utilizando-se interpoladores. A Figura 1.15 mostra uma superfície tridimensional e a grade triangular correspondente.

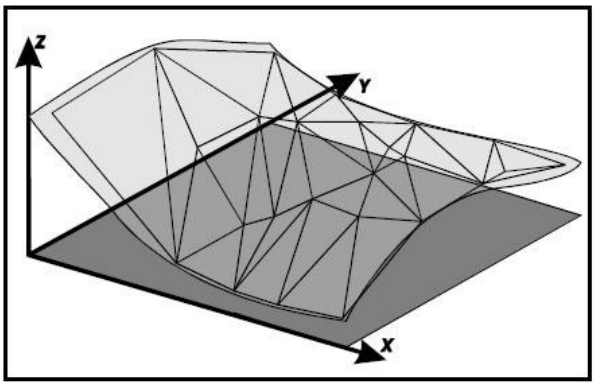


Figura 1.15 – Superfície e malha triangular correspondente. (cortesia de Laércio Namikawa).

1.7.5 Hierarquia de representações vetoriais

Para um entendimento mais detalhado das representações vetoriais em GIS, deve-se inicialmente precisar o que se entende por primitivas geométricas: *coordenadas 2D*, *coordenadas 2,5D*, *nó 2D*, *nó 2,5D*, *nó de rede*, *arcos*, *arcos orientados*, *isolinhas* e *polígonos*. Dada uma região $R \subset \Re^2$, pode-se definir:

- COORDENADA_2D – Uma coordenada 2D é um objeto composto por uma localização singular $(x_i, y_j) \in R$.
- COORDENADA_2,5D – Uma coordenada 2,5D é um objeto composto por uma localização singular (x_i, y_j, z), onde $(x_i, y_j) \in R$.
- PONTO2D – Um ponto 2D é um objeto que possui atributos descritivos e uma coordenada 2D.
- LINHA2D – Uma linha 2D possui atributos e inclui um conjunto de coordenadas 2D.

- ISOLINHA – uma isolinha contém uma linha 2D associada a um valor real (cota).
- ARCO ORIENTADO – um arco orientado contém uma linha 2D associada a uma orientação de percorrimento.
- NÓ2D – um nó 2D inclui uma coordenada2D $(x_i, y_j) \in R$ e uma lista L de linhas 2D (trata-se da conexão entre duas ou mais linhas, utilizada para manter a topologia da estrutura).
- NÓ REDE – um nó de rede contém um nó 2D e uma lista de arcos orientados.
- NÓ 2,5D – um nó 2,5D instância desta classe contém uma coordenada 2,5D (x_i, y_j, z_j) e um lista L de linhas 2D (trata-se da conexão entre três ou mais linhas de uma grade triangular).
- POLÍGONO – um polígono pode ser armazenado como uma lista de coordenadas 2D (caso dos geo-objetos sem topologia) ou por uma uma lista de linhas 2D e uma lista de nós 2D (caso de topologia *arconó-polígono*).

Uma vez definidas as primitivas geométricas vetoriais, pode ser estabelecida a hierarquia de representações geométricas vetoriais, como mostrado na Figura 1.16, onde distinguem-se os relacionamentos de especialização *é-um* ("is-a"), inclusão de uma instância *parte-de* ("part-of"), inclusão de um conjunto de instâncias *conjunto-de* ("set-of") e inclusão de uma lista de identificadores de instâncias *lista-de* ("list-of").

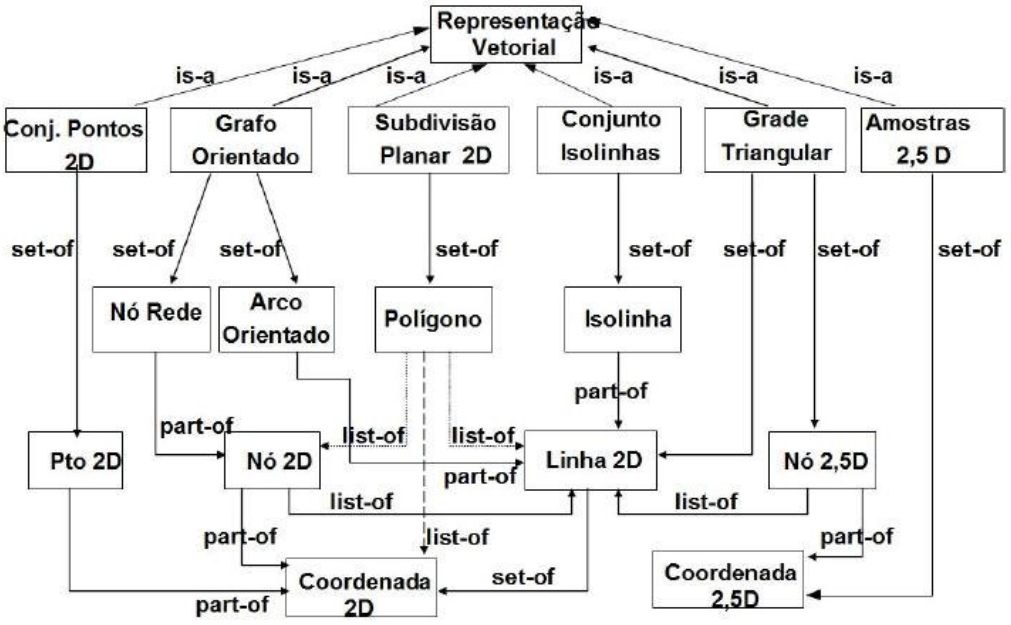


Figura 1.16 – Hierarquia de classes para estruturas vetoriais.

Distinguimos os seguintes tipos de estruturas de dados vetoriais:

- CONJUNTO DE PONTOS 2D – uma instância desta classe é um conjunto de pontos 2D utilizados para guardar localizações isoladas no espaço (p.ex. no caso de poços de petróleo).
- CONJUNTO DE ISOLINHAS – uma instância desta classe é um conjunto de linhas, onde cada linha possui uma cota e as linhas não se interceptam.

- SUBDIVISÃO PLANAR – para uma região geográfica *R* qualquer, uma subdivisão planar contém um conjunto *Pg* de polígonos que não se sobrepõem.
- GRAFO ORIENTADO – uma instância desta classe é uma representação composta de um conjunto de nó de rede e de um conjunto de arco orientado 2D.
- MALHA TRIANGULAR – uma instância desta classe contém um conjunto de nós 2,5D e um conjunto L de linhas 2D tal que todas as linhas se interseptam, mas apenas em seus pontos iniciais e finais.
- MAPA PONTOS 2,5D – uma instância desta classe é um conjunto de coordenadas 2,5D. Trata-se de um conjunto de amostras 2,5D.

1.7.6 Representação matricial

As estruturas matriciais usam uma grade regular sobre a qual se representa, célula a célula, o elemento que está sendo representado. A cada célula, atribui-se um código referente ao atributo estudado, de tal forma que o computador saiba a que elemento ou objeto pertence determinada célula. Nesta representação, o espaço é representado como uma matriz *P(m, n)* composto de *m* colunas e *n* linhas, onde cada célula possui um número de linha, um número de coluna e um valor correspondente ao atributo estudado e cada célula é individualmente acessada pelas suas coordenadas.

A representação matricial supõe que o espaço pode ser tratado como uma superfície plana, onde cada célula está associada a uma porção do terreno. A resolução do sistema é dada pela relação entre o tamanho da célula no mapa ou documento e a área por ela coberta no terreno, como mostrado na Figura 1.17.

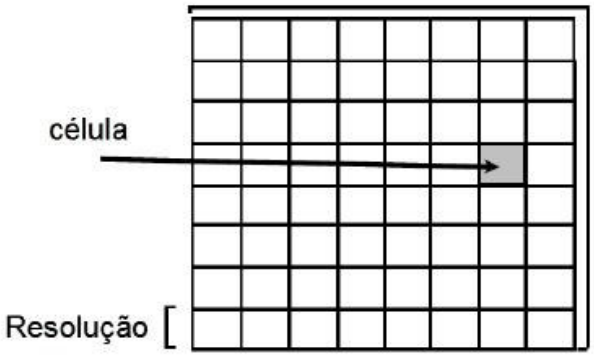


Figura 1.17 – Estrutura matricial.

A estrutura matricial pode ser utilizada para representar diferentes tipos de dados:

- *Grade regular*: representação matricial de dimensão “dois e meio” na qual cada elemento da matriz está associado a um valor numérico, como mostra a Figura 1.18 à esquerda.
- *Matriz temática*: representação matricial 2D na qual cada valor da matriz é um código correspondente à uma classe do fenômeno estudado, como mostra a Figura 1.18 à direita.

15	17	20	21	23
16	18	22	23	24
19	19	22	23	24
23	23	27	28	28
22	22	31	32	33

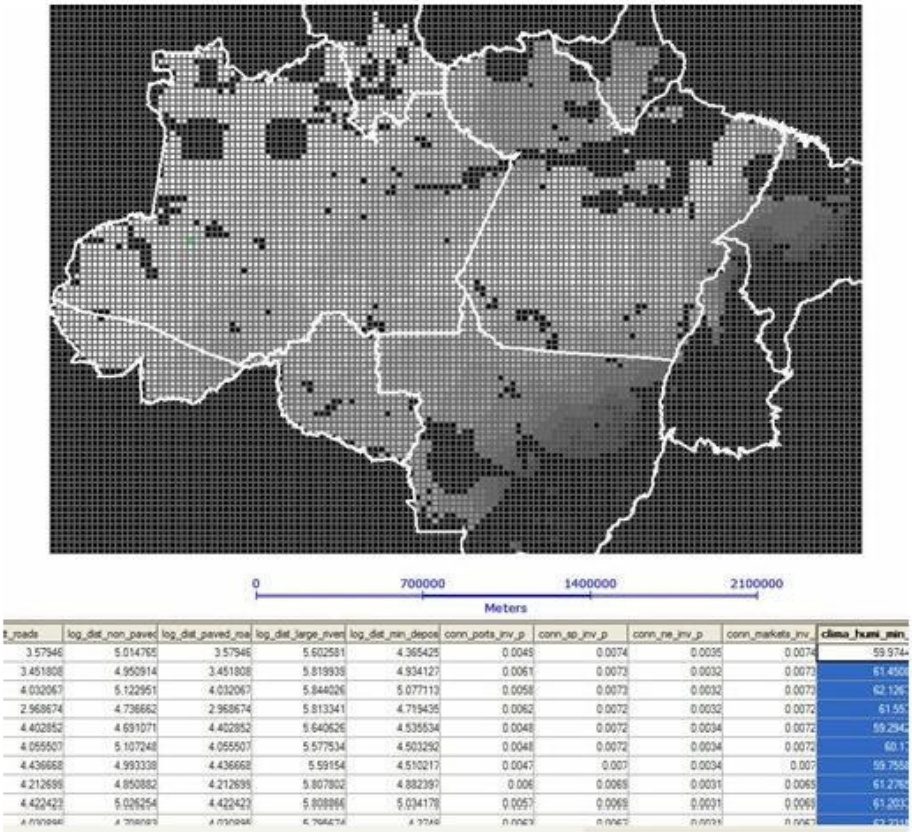
1	1	2	2	2
1	1	2	2	2
1	1	2	2	2
2	2	3	3	3
2	2	3	3	3

Figura 1.18 – À esquerda, grade regular com valores de temperatura em graus Celsius e, à direita, matriz temática com dados classificados (1 = "15-20 graus", 2 = "20-25 graus", 3 = "25-35 graus").

1.7.7 Espaços celulares: generalização de estruturas matriciais

Um *espaço celular* é uma estrutura matricial generalizada onde cada célula está associada a vários tipos de atributos. Os espaços celulares têm várias vantagens sobre estruturas matriciais simples. Usando matrizes com um único atributo (como o caso dos dados mostrados na Figura 1.18), um fenômeno espaço-temporal complexo precisa de várias matrizes separadas para ser representado, o que resulta em maior dificuldade de gerência e de interface. Num espaço celular, a mesma célula está associada a diferentes informações, com ganhos significativos de manuseio dos dados.

Os espaços celulares são muito convenientes para armazenamento em bancos de dados objeto-relacionais. Toda a estrutura de um espaço celular pode ser armazenada numa única tabela, o que faz o manuseio dos dados ser bem mais simples que os dados vetoriais ou mesmo que os dados matriciais indexados. Aplicações como álgebra de mapas e modelagem dinâmica ficam mais simples de implementar e operar. Um exemplo de espaço celular é mostrado na Figura 1.19, onde mostramos uma parte de um banco de dados onde há um espaço celular onde a Amazônia foi dividida em células de 25 x 25 km2; cada uma delas está associada a diferentes atributos socioeconômicos e ambientais (na Figura 1.19, o atributo visualizado é umidade média nos três meses mais secos do ano). Os espaços celulares ainda não são estruturas de dados comuns nos bancos de dados geográficos, e atualmente apenas a TerraLib tem suporte para este tipo de estrutura. Com a ênfase crescente dos SIG em modelos dinâmicos, podemos prever que esta estrutura será futuramente amplamente disponível nas diferentes implementações de bancos de dados geográficos.



t_roads	log_dist_non_paved	log_dist_paved_roa	log_dist_large_river	log_dist_min_depos	conn_ports_inv_p	conn_sp_inv_p	conn_ne_inv_p	conn_markets_inv_	clima_humi_min_
3.57946	5.014765	3.57946	5.602581	4.365425	0.0049	0.0074	0.0035	0.0074	59.974
3.451808	4.950914	3.451808	5.819939	4.934127	0.0061	0.0073	0.0032	0.0073	61.4568
4.032067	5.122951	4.032067	5.844026	5.077113	0.0058	0.0073	0.0032	0.0073	62.126
2.968674	4.736662	2.968674	5.813341	4.719435	0.0062	0.0072	0.0032	0.0072	61.55
4.402852	4.691071	4.402852	5.640626	4.535534	0.0048	0.0072	0.0034	0.0072	59.2942
4.055507	5.107248	4.055507	5.577534	4.503292	0.0048	0.0072	0.0034	0.0072	60.17
4.436668	4.993338	4.436668	5.59154	4.510217	0.0047	0.007	0.0034	0.007	59.7568
4.212698	4.850882	4.212698	5.807802	4.882397	0.006	0.0069	0.0031	0.0069	61.2762
4.422423	5.026254	4.422423	5.808866	5.034178	0.0057	0.0069	0.0031	0.0069	61.2033

Figura 1.19 – Espaço celular com a Amazônia dividida em células de 25 x 25 km2; o atributo visualizado é umidade média nos três meses mais secos do ano (cortesia: Ana Paula Dutra de Aguiar).

1.8 Do universo formal para o universo estrutural

A passagem do universo formal (geo-campos, geo-objetos e redes) para o universo estrutural não é unívoca. Para cada tipo de entidade do modelo formal, há diferentes possibilidades de uso de estruturas de dados, a saber:

- *Geo-objetos*: como as fronteiras são elementos essenciais, são usualmente armazenados em estruturas poligonais, com as opções *polígonos sem topologia* ou *topologia arco-nó-polígono*.
- *Redes*: como a topologia é parte essencial, as redes devem ser armazenadas como um grafo orientado.
- *Geo-campos numéricos*: podem ser armazenados como amostras 2,5D, malhas triangulares ou grades regulares.
- *Geo-campos temáticos*: admitem o armazenamento como estruturas vetoriais (polígonos) ou matriciais (matrizes temáticas).

Os diferentes compromissos de armazenamento para as entidades do modelo formal são discutidos a seguir. Note-se que um espaço celular (discutido na Seção 1.7.7) pode guardar uma combinação arbitrária de geo-campos numéricos e temáticos.

1.8.1 Estruturas de dados para geo-objetos

A escolha entre estruturas topológicas ou não-topológicas para geo-objetos em bancos de dados geográficos depende também do suporte oferecido pelo SGBD. Nos SIG cujas estruturas de dados geométricas são manuseadas fora do SGBD (como o SPRING e o Arc/Info), é comum a escolha da topologia arco-nó-polígono. No caso dos bancos de dados geográficos, a maneira mais simples de armazenar geo-objetos é guardando cada um deles separadamente, o que implica em estruturas não-topológicas. Esta forma de trabalho foi sancionada pelo consórcio Open GIS e é suportada pelos diferentes SGBDs (Oracle, PostgreSQL,MySQL). No entanto, várias aplicações requerem o uso da topologia arco-nó-polígono, e alguns SGBDs com suporte espacial já estão incluindo esta opção, com o Oracle Spatial (Ravada, 2003).

1.8.2 Estruturas de dados para geo-campos temáticos

Geo-campos temáticos admitem tanto a representação matricial quanto a vetorial. Para a produção de cartas e em operações onde se requer maior precisão, a representação vetorial é mais adequada. As operações de álgebra de mapas são mais facilmente realizadas no formato matricial. No entanto, para um mesmo grau de precisão, o espaço de armazenamento requerido por uma representação matricial é substancialmente maior. Isto é ilustrado na Figura 1.20.

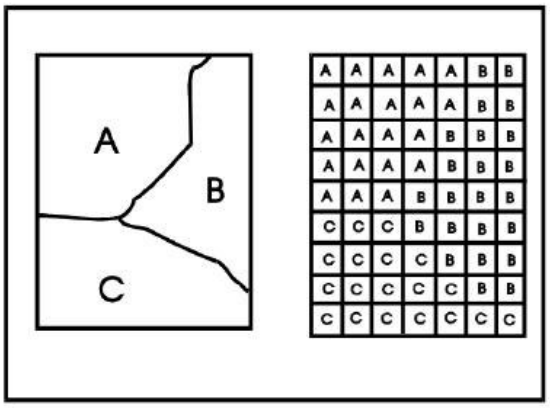


Figura 1.20 – Geo-campo temático em estruturas vetorial e matricial.

A Tabela 1.3 apresenta uma comparação entre as vantagens e desvantagens de armazenamento matricial e vetorial para geo-campos temáticos. Esta comparação leva em conta os vários aspectos: relacionamentos espaciais, análise, armazenamento. Nesta tabela, o formato mais vantajoso para cada caso é apresentado em destaque.

O armazenamento de geo-campos temáticos em estruturas vetoriais é uma herança da cartografia, onde limites entre classes temáticas eram desenhados com precisão em mapas. No entanto, sabemos que estes limites são imprecisos, na grande maioria dos casos. Assim, como nos ensina Peter Burrough, as estruturas matriciais são mais adequadas:

"*Os limites desenhados em mapas temáticos (como solo, vegetação, ou geologia) raramente são precisos e desenha-los como linhas finas muitas vezes não representa adequadamente seu caráter. Assim, talvez não nos devamos preocupar tanto com localizações exatas e representações gráficas elegantes. Se pudermos aceitar que limites precisos entre padrões de vegetação e solo raramente ocorrem, nós*

estaríamos livres dos problemas de erros topológicos associados como superposição e interseção de mapas" (Burrough, 1986).

Tabela 1.3 – Comparação entre estruturas vetoriais e matriciais para mapas temáticos

Aspecto	*Vetorial*	*Matricial*
Armazenamento	Por coordenadas (mais eficiente)	Requer mais espaço de armazenamento
Algoritmos	Problemas com erros geométricos	Processsamento mais rápido e eficiente.
Escalas de trabalho	Adequado tanto a grandes quanto a pequenas escalas	Mais adequado para pequenas escalas (1:25.000 e menores)
Análise, Simulação e Modelagem	Representação indireta de fenômenos contínuos Álgebra de mapas é limitada	Representa melhor fenômenos com variação contínua no espaço Simulação e modelagem mais fáceis

1.8.3 Estruturas de dados para geo-campos numéricos

Para geo-campos numéricos, a escolha básica se dá entre malhas triangulares e grades regulares. As demais estruturas de dados (amostras 2,5D e isolinhas) são formatos intermediários, utilizados para entrada ou saída de dados, mas não adequadas para análise.

As malhas triangulares são normalmente melhores para representar a variação do terreno, pois capturam a complexidade do relevo sem a necessidade de grande quantidade de dados redundantes. As grades regulares têm grande redundância em terrenos uniformes e dificuldade de adaptação a relevos de natureza distinta no mesmo mapa, por causa da grade de amostragem fixa.

Para o caso de variáveis geofísicas e para operações como visualização 3D, as grades regulares são preferíveis, principalmente pela maior facilidade de manuseio computacional. A Tabela 1.4 resume as principais vantagens e desvantagens de grades regulares e malhas triangulares.

Tabela 1.4 – Estruturas para geo-campos numéricos

	Malha triangular	Grade regular
Vantagens	Melhor representação de relevo complexo Incorporação de restrições como linhas de crista	Facilita manuseio e conversão Adequada para dados não-altimétricos
Problemas	Complexidade de manuseio	Representação relevo complexo Cálculo de declividade

1.8.4 Representações computacionais de atributos de objetos

Entende-se por atributo qualquer informação descritiva (nomes, números, tabelas e textos) relacionada com um único objeto, elemento, entidade gráfica ou um conjunto deles, que caracteriza um dado fenômeno geográfico. Nos bancos de dados geográficos, os atributos de objetos geográficos são armazenados em relações convencionais. As representações geométricas destes objetos podem ser armazenadas na mesma tabela que os atributos ou em tabelas separadas, mas ligadas por identificadores únicos.

1.9 Universo de implementação

No universo de implementação, são tomadas as decisões concretas de programação e que podem admitir número muito grande de variações. Estas decisões podem levar em conta as aplicações às quais o sistema é voltado, a disponibilidade de algoritmos para tratamento de dados geográficos e o desempenho do hardware.

2 Modelagem Conceitual de Dados Geográficos

Um modelo de dados é um conjunto de conceitos que podem ser usados para descrever a estrutura e as operações em um banco de dados (Elmasri, 2004). O modelo busca sistematizar o entendimento a respeito de objetos e fenômenos que serão representados em um sistema informatizado. No processo de modelagem é necessário construir uma abstração dos objetos e fenômenos do mundo real, de modo a obter uma forma de representação conveniente, embora simplificada, que seja adequada às finalidades das aplicações. A modelagem de dados geográficos é uma atividade complexa porque envolve a discretização do espaço como parte do processo de abstração, visando obter representações adequadas aos fenômenos geográficos.

A comunidade de banco de dados estabelece claramente uma distinção entre **modelo de dados conceitual** e **esquema conceitual**. Um modelo conceitual se refere a uma técnica usada para modelar um banco de dados, incluindo suas notações. Esquemas conceituais, por outro lado, se referem ao resultado de uma modelagem, ou seja, um conjunto de diagramas que usa um determinado modelo conceitual como uma linguagem para expressar estruturas de dados específicas para uma aplicação. Esquemas conceituais são construídos para abstrair partes específicas do mundo real

e representar, esquematicamente, quais os dados devem ser coletados, como eles serão organizados e relacionados entre si. Esses esquemas servem também como uma documentação da base de dados.

Os primeiros modelos de dados para as aplicações geográficas eram voltados para as estruturas internas dos SIG. O usuário era forçado a adequar os fenômenos espaciais às estruturas disponíveis no SIG a ser utilizado. Conseqüentemente, o processo de modelagem não oferecia mecanismos para a representação da realidade de forma mais próxima ao modelo mental do usuário. Ficava evidente que a modelagem de aplicações geográficas necessitava de modelos mais adequados, capazes de capturar a semântica dos dados geográficos, oferecendo mecanismos de abstração mais elevados e independência de implementação.

Alguns modelos tradicionais de modelagem de dados para aplicações convencionais, como por exemplo, Entidade-Relacionamento (ER) (Chen, 1976), OMT (Rumbaugh et al., 1991), IFO (Abiteboul and Hull, 1987) e UML (Corporation, 1997), têm sido largamente utilizados para a modelagem de aplicações geográficas. Apesar da grande expressividade desses modelos, eles apresentam limitações para a adequada modelagem de aplicações geográficas, já que não possuem primitivas apropriadas para a representação de dados espaciais. Modelos de dados para aplicações geográficas têm necessidades adicionais, tanto com relação à abstração de conceitos e entidades, quanto ao tipo de entidades representáveis e seu inter-relacionamento.

Diversas propostas existem atualmente, principalmente focalizadas em estender os modelos tradicionais, como GeoOOA (Kösters, 1997), MODUL-R (Bédard, 1996), GMOD (Oliveira, 1997), IFO para aplicações geográficas (Worboys et al., 1990), GISER (Shekhar, 1997), OMT-G (Borges, 2001), GeoFrame (Lisboa and Iochpe, 1999) e MADS (Parent, 1999). Todos esses modelos procuram refletir melhor as necessidades de aplicações geográficas. Este capítulo apresenta dois modelos, OMT-G e GeoFrame, ambos baseados na linguagem UML (*Unified Modeling Language*) (Corporation, 1997) (Corporation, 1997).

2.1 Modelo de dados OMT-G

O modelo OMT-G (*Object Modeling Technique for Geographic Applications*) parte das primitivas definidas para o diagrama de classes da UML, introduzindo primitivas geográficas com o objetivo de aumentar a capacidade de representação semântica do espaço (Borges, 2001). O modelo OMT-G provê primitivas para modelar a geometria e a topologia dos dados geográficos, oferecendo suporte a estruturas topológicas "todo-parte", estruturas de rede, múltiplas representações de objetos e relacionamentos espaciais. Além disso, o modelo permite a especificação de atributos alfanuméricos e métodos associados para cada classe.

Esse modelo é baseado em três conceitos principais: **classes**, **relacionamentos** e **restrições de integridade espaciais**. Classes e relacionamentos definem as primitivas básicas usadas para criar esquemas estáticos de aplicação. Além disso, OMT-G propõe o uso de três diferentes diagramas no processo de desenvolvimento de uma aplicação geográfica: **diagrama de classes**, **diagrama de transformação** e **diagrama de apresentação**. O diagrama de classes é o mais usual e contém as classes especificadas junto com suas representações e relacionamentos. A partir do diagrama de classes é possível derivar um conjunto de restrições de integridade espaciais, que deve ser considerado na implementação.

Quando o diagrama de classes especifica múltiplas representações ou a derivação de uma classe a partir de outra, é necessário desenvolver um diagrama de transformação. Nele todo o processo de transformação pode ser especificado, permitindo a identificação dos métodos necessários para a implementação. Finalmente, para especificar as alternativas de visualização que cada representação pode assumir, é necessário desenvolver um diagrama de apresentação. As primitivas para cada um desses diagramas são detalhadas nas próximas seções.

2.1.1 Diagrama de classes

O diagrama de classes é usado para descrever a estrutura e o conteúdo de um banco de dados geográfico. Ele contém elementos específicos da estrutura de um banco de dados, em especial classes de objetos e seus relacionamentos. O diagrama de classes contém apenas regras e descrições que definem conceitualmente como os dados serão estruturados, incluindo informações sobre o tipo de representação que será adotada para cada classe. Por esta razão, o diagrama de classe é o produto fundamental do nível da modelagem conceitual. A seguir estão descritas as primitivas do modelo OMT-G que são usadas para criar o diagrama de classes para as aplicações geográficas.

2.1.1.1 Classes

As classes definidas pelo modelo OMT-G representam os três grandes grupos de dados (contínuos, discretos e não-espaciais) que podem ser encontrados nas aplicações geográficas, proporcionando assim, uma visão integrada do espaço modelado. Suas classes podem ser *georreferenciadas* ou *convencionais*. A distinção entre classes convencionais e georreferenciadas permite que aplicações diferentes compartilhem dados não espaciais, facilitando o desenvolvimento de aplicações integradas e a reutilização de dados.

A classe *georreferenciada* descreve um conjunto de objetos que possuem representação espacial e estão associados a regiões da superfície da terra, representando as visões de geo-campos e geo-objetos mostradas no Capítulo 1. A classe convencional descreve um conjunto de objetos com propriedades, comportamento, relacionamentos e semânticas semelhantes, e que possuem alguma relação com os objetos espaciais, mas que não possuem propriedades geográficas. As classes georreferenciadas são especializadas em classes do tipo *geo-campo* e *geo-objeto*.

As classes covencionais são simbolizadas exatamente como na UML. As classes georreferenciadas são simbolizadas no modelo OMT-G de forma semelhante, incluindo no canto superior esquerdo um retângulo que é usado para indicar a forma geométrica da representação. Em ambos os casos, símbolos simplificados podem ser usados. Os objetos podem ou não ter atributos não espaciais associados, listados na seção central da representação completa. Métodos ou operações são especificados na seção inferior do retângulo. Essas diferenças são mostradas na Figura 2.1.

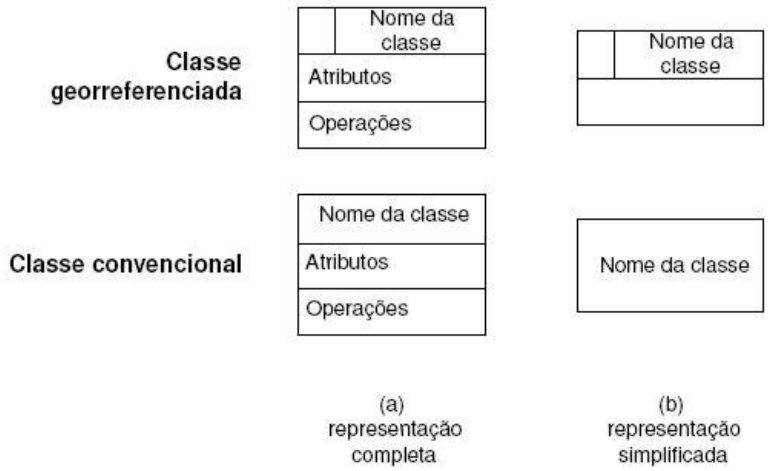


Figura 2.1 – Classes georreferenciadas e convencionais no OMT-G.
Fonte: (Borges, 2005)

O modelo OMT-G apresenta um conjunto fixo de alternativas de representação geométrica, usando uma simbologia que distingue geo-objetos e geo-campos, mostrados na Figura 2.2 e na Figura 2.3. O modelo OMT-G define cinco classes descendentes de geo-campos: *isolinhas*, *subdivisão planar*, *tesselação*, *amostragem* e *malha triangular* (*triangulated irregular network*, *TIN*), e duas classes descendentes de geo-objeto: *geo-objeto com geometria* e *geo-objeto com geometria e topologia*.

Figura 2.2 – Geo-campos
Fonte: (Borges, 2005)

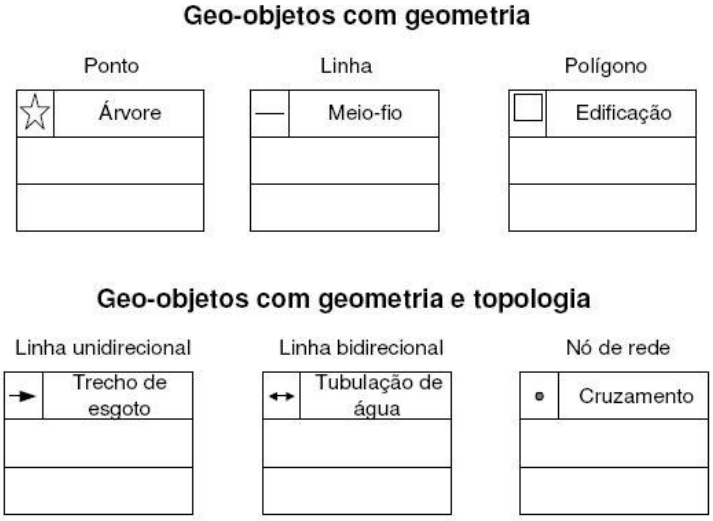


Figura 2.3 – Geo-objetos

Fonte: (Borges, 2005)

A classe geo-objeto com geometria representa objetos que possuem apenas propriedades geométricas, e é especializada nas classes *Ponto*, *Linha* e *Polígono*, como por exemplo, respectivamente, árvore, meio fio e edificação (Figura 2.3). A classe geo-objeto com geometria e topologia representa objetos que possuem, além das propriedades geométricas, propriedades de conectividade topológica, sendo especificamente voltadas para a representação de estruturas em rede, como por exemplo, sistemas de abastecimento de água ou fornecimento de energia elétrica.

Os arcos podem ser unidirecionais, como em redes de esgoto, ou bidirecionais, como em redes de telecomunicações. Assim, as especializações previstas são denominadas *nó de rede*, *arco unidirecional* e *arco bidirecional*. Os segmentos orientados traduzem o sentido do fluxo da rede, se unidirecional ou bidirecional, dando mais semântica à representação. O foco do modelo OMT-G com respeito a redes não está concentrado na implementação do relacionamento entre seus elementos, mas sim na semântica da conexão entre elementos de rede, que é um fator relevante para o estabelecimento de regras que garantam a integridade do banco de dados. Nas aplicações de rede os relacionamentos do tipo conectividade e adjacência são fundamentais.

2.1.1.2 Relacionamentos

Considerando a importância das relações espaciais e não espaciais na compreensão do espaço modelado, o modelo OMT-G representa três tipos de relacionamentos entre suas classes: associações simples, relacionamentos topológicos em rede e relacionamentos espaciais. A discriminação de tais relacionamentos tem o objetivo de definir explicitamente o tipo de interação que ocorre entre as classes.

Associações simples representam relacionamentos estruturais entre objetos de classes diferentes, convencionais ou georreferenciadas. Relacionamentos espaciais representam relações topológicas, métricas, de ordem e fuzzy. Algumas relações podem ser derivadas automaticamente, a partir da forma geométrica do objeto, no momento da entrada de dados ou da execução de alguma análise espacial. Relacionamentos topológicos são exemplos dessa possibilidade.

No modelo OMT-G, associações simples são indicadas por linhas contínuas, enquanto relacionamentos espaciais são indicados por linhas pontilhadas, como ilustrado na Figura 2.4a/b. Isso torna fácil a distinção visual entre relacionamentos baseados em atributos alfanuméricos e baseados na localização e forma geométrica dos objetos. O nome do relacionamento é anotado sobre a linha, e uma seta usada para deixar clara a direção de leitura (por exemplo, na Figura 2.4b, lê-se "lote contém edificação").

Os relacionamentos de rede são relacionamentos entre objetos que estão conectados uns com os outros. Relacionamentos de rede são indicados por duas linhas pontilhadas paralelas, entre as quais o nome do relacionamento é anotado, como mostrado na Figura 2.4c. Os relacionamentos são em geral especificados entre uma classe de nós e uma classe de arcos. No entanto, estruturas de redes sem nós podem ser definidas, especificando um relacionamento recursivo sobre uma classe de arcos (Figura 2.4d).

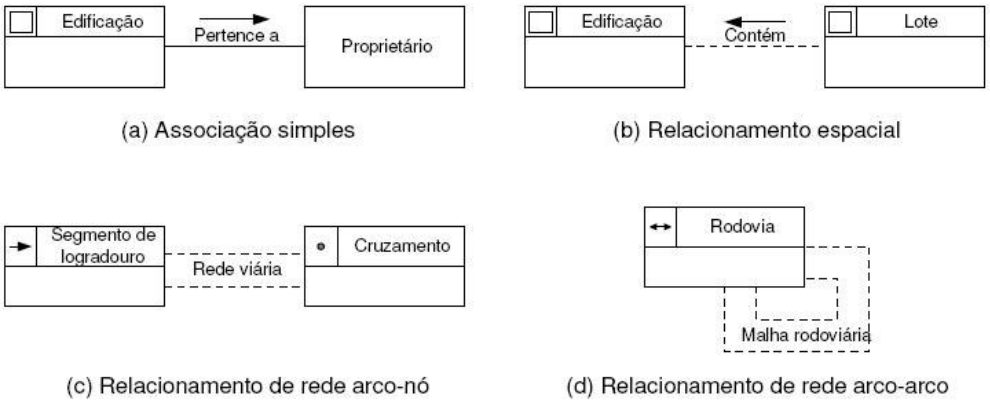


Figura 2.4 – Relacionamentos
Fonte: (Borges, 2005)

2.1.1.3 Cardinalidade

Os relacionamentos são caracterizados por sua cardinalidade. A cardinalidade representa o número de instâncias de uma classe que podem estar associadas a instâncias da outra classe. A notação de cardinalidade adotada pelo modelo OMT-G é a mesma usada na UML, como mostrado na Figura 2.5.

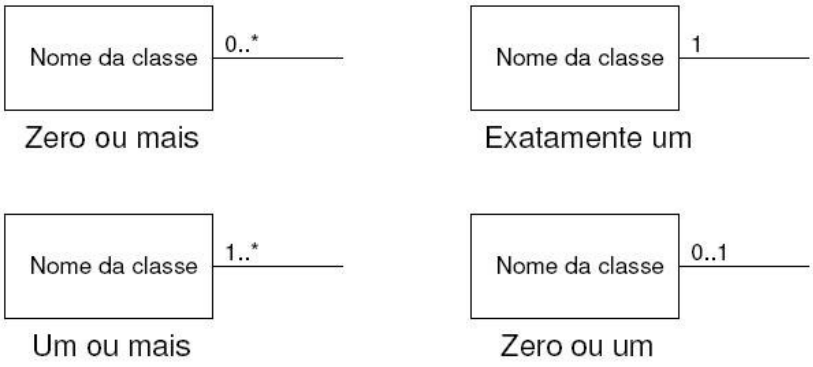


Figura 2.5 – Cardinalidade
Fonte: (Borges, 2005)

2.1.1.4 Generalização e especialização

Generalização é o processo de definição de classes mais genéricas (superclasses) a partir de classes com características semelhantes (subclasses). A especialização é o processo inverso, no qual classes mais específicas são detalhadas a partir de classes genéricas, adicionando novas propriedades na forma de atributos. Cada subclasse herda atributos, operações e associações da superclasse.

No modelo OMT-G, as abstrações de generalização e especialização se aplicam tanto às classes georreferenciadas quanto às classes convencionais, seguindo as definições e a notação propostas na UML, como mostrado na Figura 2.6. Cada generalização pode ter um discriminador associado, que indica qual propriedade ou característica está sendo abstraída pelo relacionamento de generalização.

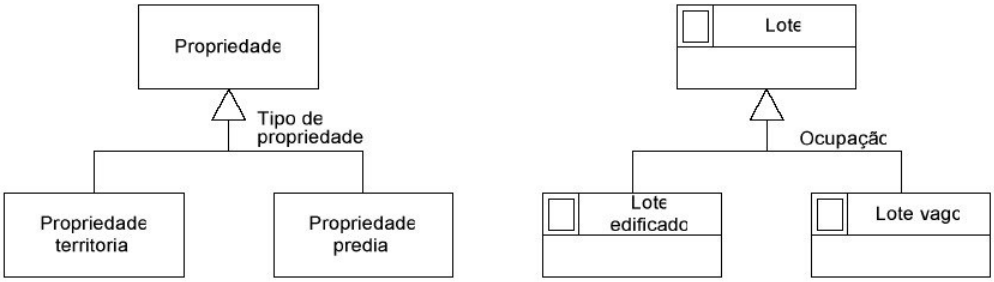


Figura 2.6 – Generalização/Especialização
Fonte: (Borges, 2005)

Uma generalização (espacial ou não) pode ser especificada como total ou parcial (Laender, 1994) (Corporation, 1997). Uma generalização é total quando a união de todas as instâncias das subclasses equivale ao conjunto completo de instâncias da superclasse. A UML representa a totalidade através do uso dos elementos de restrição predefinidos como completo e incompleto, mas no modelo OMT-G foi adotada a notação introduzida em (Laender, 1994), na qual um ponto é colocado no ápice do triângulo para denotar a totalidade. Além disso, o modelo OMT-G também adota a notação OMT (Rumbaugh et al., 1991) para os elementos de restrição predefinidos como disjunto e sobreposto da UML, ou seja, em uma generalização disjunta o triângulo é deixado em branco e em uma generalização sobreposta o triângulo é preenchido. Portanto, a combinação de disjunção e totalidade gera quatro tipos de restrições aplicáveis a generalização/especialização. A Figura 2.7 apresenta exemplos de cada combinação.

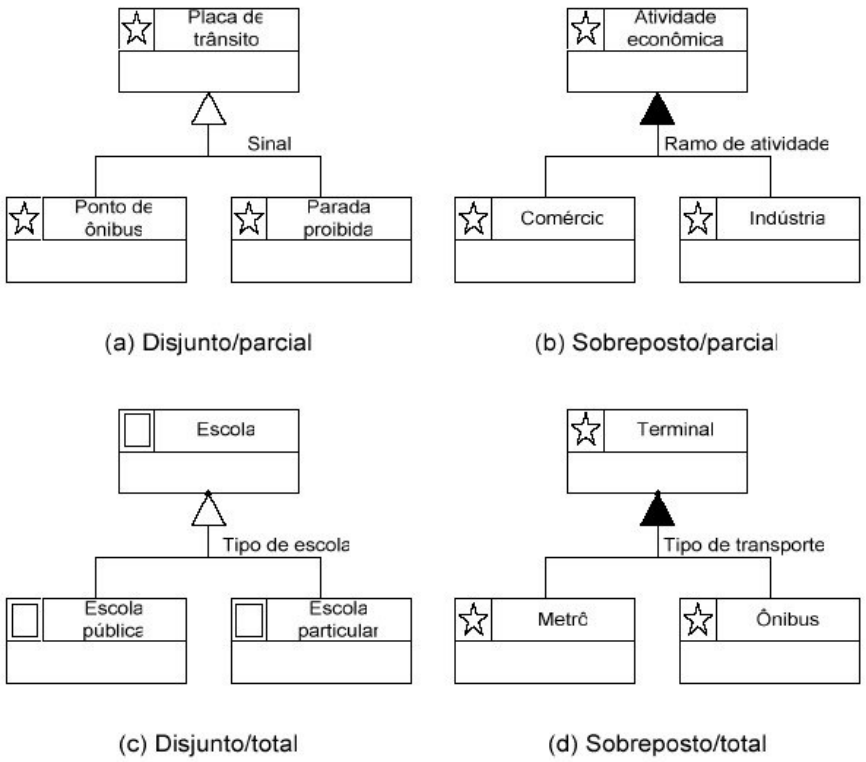


Figura 2.7 – Exemplos de generalização espacial
Fonte: (Borges, 2005)

2.1.1.5 Agregação

A agregação é uma forma especial de associação entre objetos, onde se considera que um deles é formado a partir de outros. A notação gráfica usada no modelo OMT-G segue a empregada na UML. Uma agregação pode ocorrer entre classes convencionais, entre classes georreferenciadas ou entre uma classe convencional e uma classe georreferenciada. A Figura 2.8 ilustra uma agregação entre uma classe convencional e uma georrefenciada. Quando a agregação ocorre entre classes georreferenciadas, é necessário usar a agregação espacial.

Figura 2.8 - Agregação entre uma classe convencional e uma georreferenciada.
Fonte: (Borges, 2005)

A agregação espacial é um caso especial de agregação na qual são explicitados relacionamentos topológicos "todo-parte" (Abrantes, 1994) (Kösters, 1997). A utilização desse tipo de agregação impõe restrições de integridade espacial no que diz respeito à existência do objeto agregado e dos sub-objetos. Além de o modelo ganhar mais clareza e expressividade, a observação dessas regras contribui para a manutenção da integridade semântica do banco de dados geográfico. Muitos erros no processo de entrada de dados podem ser evitados, se procedimentos baseados nessas restrições forem implementados.

A agregação espacial indica que a geometria de cada parte deve estar contida na geometria do todo. Não é permitida a superposição entre geometria das partes, a geometria do todo deve ser totalmente coberta pela geometria das partes, configurando assim, uma partição do plano ou subdivisão planar (Preparata and Shamos, 1985) (Davis Jr., 2000). A notação para essa primitiva é apresentada na Figura 2.9, onde mostra uma situação em que quadras são compostas de lotes, ou seja, as quadras são geometricamente equivalentes à união dos lotes contidos nelas.

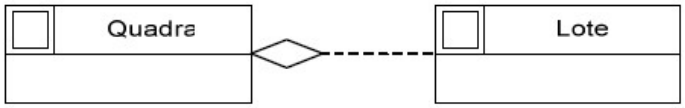


Figura 2.9 - Agregação espacial ("todo-parte")
Fonte: (Borges, 2005)

2.1.1.6 Generalização conceitual

A generalização conceitual, no sentido cartográfico, pode ser definida como uma série de transformações que são realizadas sobre a representação da informação espacial, cujo objetivo é melhorar a legibilidade e aumentar a facilidade de compreensão dos dados por parte do usuário do mapa. Por exemplo, um objeto do mundo real pode ter diversas representações espaciais, de acordo com a escala de visualização. Uma cidade pode ser representada em um mapa de escala pequena por um ponto, e como um polígono em um mapa de escala maior (Davis and Laender, 1999). Neste sentido, o termo representação é usado no sentido de representação da forma geométrica do objeto geográfico.

Definir se a representação deve ser simples ou mais elaborada depende da percepção que o usuário tem do objeto correspondente no mundo real, e como essa

representação afeta os relacionamentos espaciais que podem ser estabelecidos com outros objetos modelados.

Considerando a necessidade de tais relacionamentos, pode haver a demanda para mais de uma representação para um dado objeto. Isso acontece, por exemplo, quando a informação geográfica precisa ser compartilhada entre diversas aplicações em um ambiente corporativo (ou cooperativo). Portanto, no desenvolvimento de aplicações geográficas, existem situações em que duas ou mais representações para um objeto do mundo real precisam coexistir. Isso significa que, dependendo da visão do usuário, é necessário ter formas geométricas distintas para representar o mesmo objeto geográfico, com a mesma resolução e ao mesmo tempo. Além disso, é freqüente a necessidade de se representar o mesmo objeto com graus variáveis de resolução e detalhamento, configurando representações adequadas para diferentes faixas de escalas.

A primitiva de generalização conceitual foi incluída no modelo OMT-G para registrar a necessidade de representações diferentes para um mesmo objeto. Nesse tipo de relacionamento, a superclasse não tem uma representação específica, já que poderá ser percebida de maneiras diferentes, conforme especificado nas subclasses. Essas são representadas por formas geométricas distintas, podendo herdar os atributos alfanuméricos da superclasse e ainda possuir atributos próprios. O objetivo é permitir a especificação de relacionamentos independentes envolvendo cada alternativa de representação considerada. A generalização conceitual pode ocorrer em duas variações: de acordo com a forma geométrica (Figura 2.10a) ou de acordo com a escala (Figura 2.10b).

A variação de acordo com a forma é utilizada para registrar a existência de múltiplas representações para uma classe, independente de escala. A descrição geométrica da superclasse é deduzida a partir do uso das subclasses. Por exemplo, um rio pode ser percebido como um espaço entre suas margens, como um polígono de água ou como um fluxo (linha direcionada), formando a rede hidrográfica (Figura 2.10a). A variação de acordo com a escala é usada na representação de diferentes aspectos geométricos de uma classe, cada aspecto corresponde a uma faixa de escalas. Uma cidade pode ser representada por suas fronteiras políticas (um polígono) em uma escala maior, e por um símbolo (um ponto) em uma escala menor (Figura 2.10b).

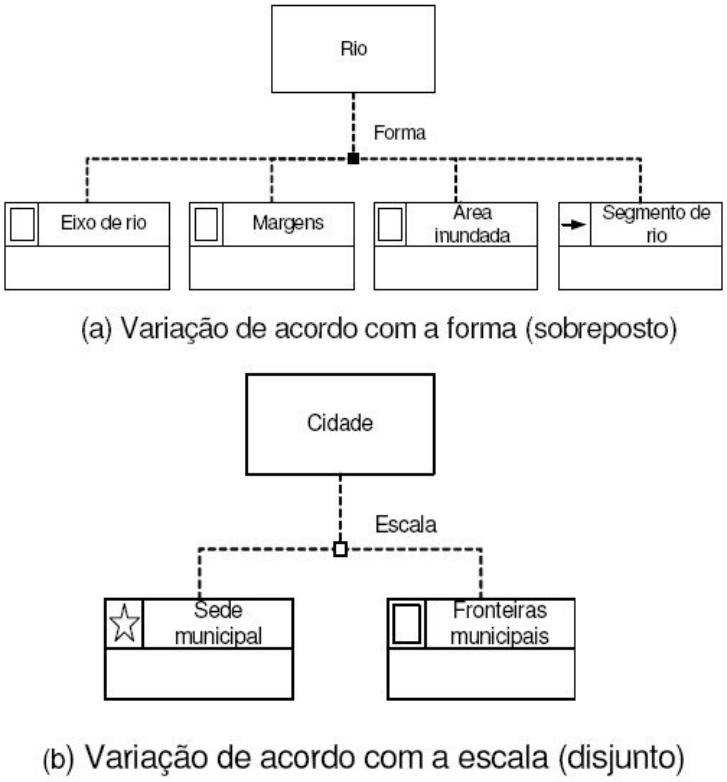


Figura 2.10 - Generalização conceitual
Fonte: (Borges, 2005)

Uma estrutura como a apresentada na Figura 2.10 é rara em esquemas de aplicações geográficas, porque as decisões quanto à modelagem são freqüentemente (e erroneamente) tomadas já pensando na apresentação final, conforme exigido pela aplicação que está sendo modelada. Ou seja, o esquema é muitas vezes concebido visando um tipo específico de visualização, antecipando uma exigência da aplicação. Esta tendência acaba por inibir usos que exijam representações alternativas, ou aplicações que compartilhem dados geográficos (Davis Jr., 2000).

2.1.1.7 Restrições de integridades

No modelo OMT-G, existem diversas restrições de integridade que são implícitas às primitivas do modelo ou que podem ser deduzidas a partir da análise dos diagramas. Assim, restrições de integridade topológica são definidas através de regras para geo-campos, relacionamentos espaciais, relacionamentos em rede e para agregação espacial. Da mesma forma, restrições de integridade semântica são definidas através de regras associadas a relacionamentos espaciais. Já as restrições de integridade definidas pelo usuário podem ser modeladas como métodos associados a cada classe. Mais informações sobre restrições de integridade no modelo OMT-G podem ser encontradas em (Borges, 1999) (Davis Jr., 2001) e (Davis Jr., 2005).

2.1.2 Diagrama de transformação

O diagrama de transformação, proposto para o modelo OMT-G em (Davis and Laender, 1999), adota uma notação semelhante à proposta na UML para os diagramas de estados e de atividades, e é usado para especificar transformações entre classes. Como tanto a origem quanto o resultado das transformações são sempre as

representações de cada classe, o diagrama de transformação também está no nível conceitual de representação.

Observe que o diagrama de transformação não pretende descrever aspectos dinâmicos da aplicação, como a interface com o usuário e a execução de consultas, restringindo-se à manipulação de representações. Os diagramas de transformação são baseados nas primitivas de classe, conforme definidas para os diagramas de classes. As classes que estão envolvidas em algum tipo de transformação são conectadas por meio de linhas contínuas, com setas que indicam a direção da transformação. Os operadores de transformação (TR) envolvidos e seus parâmetros, quando houver, são indicados por meio de texto sobre a linha que indica a transformação.

No diagrama de transformação, pode-se indicar se o resultado da transformação precisa ou não ser materializado. Classes resultantes muito simples, ou que são passos intermediários em uma transformação mais complexa, freqüentemente não precisam ser materializadas, e podem ser armazenadas apenas temporariamente. Tais classes temporárias são indicadas usando linhas tracejadas em seu contorno. As classes que são resultantes de alguma transformação e que precisam ser materializadas (devido à complexidade do processo ou às necessidades específicas da aplicação) são denotadas com linhas contínuas, exatamente como no diagrama de classes.

As transformações indicadas no diagrama de classes podem relacionar qualquer número de classes originais, bem como qualquer número de classes resultantes, dependendo da natureza da operação de transformação. Cadeias de transformações também podem ser definidas, permitindo, dessa forma, a especificação de processos complexos de análise espacial.

Um operador de transformação adequado para o diagrama de transformação pode ser basicamente qualquer algoritmo que manipula e modifica a representação de um objeto. Algumas operações podem ser melhor caracterizadas como operações TR quando existe apenas uma classe de origem e uma classe resultante, e a classe resultante é ou (1) de natureza diferente da classe original (ou seja, pertence a uma classe georreferenciada diferente), ou (2) menos detalhada que a classe original, mantendo a natureza da representação (Davis and Laender, 1999).

A especificação de transformações no diagrama de transformação é em geral exigida quando as primitivas de generalização conceitual e de agregação espacial são usadas no diagrama de classes. Essas duas primitivas são indicativas da possibilidade de produzir uma representação a partir de outras. Um estudo das possíveis transformações entre representações de geo-objetos e geo-campos pode ser visto em (Davis Jr., 2000) (Davis and Laender, 1999). Os operadores aplicados para cada transformação são baseados em algoritmos definidos nas áreas de geometria computacional, generalização cartográfica e análise espacial.

2.1.3 Diagrama de apresentação

O diagrama de apresentação para o modelo OMT-G pertence ao nível de apresentação. Em contraste com o conceito de representação, o termo apresentação é usado no sentido de determinar o aspecto visual ou gráfico (envolvendo parâmetros como cor, tipo de linha, espessura da linha e padrão de hachura), de geo-objetos e geo-campos, no papel ou na tela do computador.

No diagrama de apresentação estão reunidos os requisitos definidos pelo usuário quanto às alternativas de apresentação e saída para cada objeto geográfico. Essas alternativas podem incluir apresentações criadas especificamente para visualização em tela, para impressão na forma de mapas ou cartas, para interpretação visual em um processo de análise, e outras.

Cada apresentação é definida a partir de uma representação contida no diagrama de classes ou no diagrama de transformação do nível de representação. Operações de transformação para apresentação (TA) são especificadas, permitindo obter o aspecto visual desejado a partir da simples forma geométrica, definida para a representação. Observe-se que a operação TA não modifica a alternativa de representação definida previamente, nem muda o detalhamento definido no nível de representação. Se isso for necessário, uma nova representação tem de ser criada a partir de uma representação existente, usando as ferramentas de especificação de múltiplas representações (como a primitiva de generalização conceitual) e registrando essa demanda nos diagramas de classes e de transformação.

O diagrama de apresentação necessita de apenas três primitivas. A primeira é a própria primitiva de classes, definida para os diagramas de classes e de transformação. A segunda é usada para indicar a operação TA, de maneira semelhante à usada para denotar as transformações no diagrama de transformação. É composta de uma linha tracejada simples, com uma seta que indica o sentido da operação, sobre a qual é especificado o operador a ser usado. No processo de especificação dessa expressão de transformação, quaisquer características geométricas ou atributos alfanuméricos que foram definidos no nível de representação para a classe podem ser usados como parâmetros. As linhas indicando operações TA são tracejadas para distingui-las visualmente das operações TR, especificadas no diagrama de transformação com linhas contínuas. A terceira primitiva serve para especificar uma apresentação, e contém duas seções.

A seção superior indica o nome da classe, o nome da apresentação, e a aplicação na qual é usada. Nessa seção pode-se especificar uma faixa de escalas onde a apresentação será usada. A segunda é dividida em duas partes: à esquerda, um pictograma indica o aspecto visual dos objetos após a transformação e à direita são lançadas especificações mais precisas quanto aos atributos gráficos, incluindo cor da linha, tipo e espessura de linha, padrão de preenchimento, cor de preenchimento, e nome do símbolo, como mostrado na Figura 2.11. A especificação dos atributos gráficos pode ser feita já considerando a codificação de símbolos usada pelo sistema de informação geográfica subjacente. Pode existir qualquer número de pictogramas na seção esquerda da primitiva de especificação de apresentações, cada qual associada a um valor ou faixa de valores obtidos a partir das características de cada objeto. Nesse caso, a seção da direita deve detalhar os atributos gráficos de cada apresentação gerada. Atributos comuns podem ser especificados apenas uma vez, enquanto atributos variáveis são especificados como listas de valores individuais. Como no caso do diagrama de transformação, os resultados das transformações (ou seja, as apresentações) são indicados com linhas tracejadas quando não precisam ser materializados no banco de dados e com linhas contínuas no caso contrário.

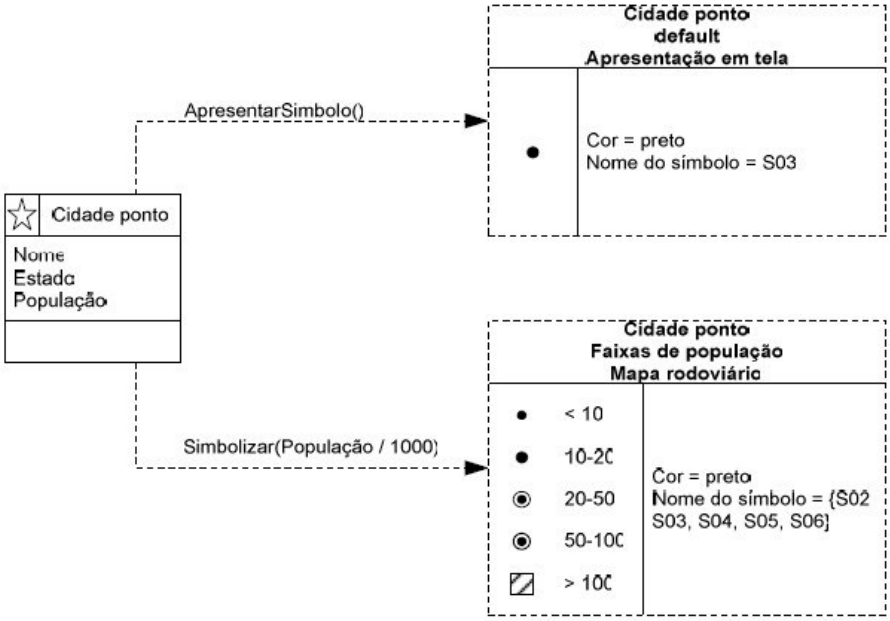


Figura 2.11- Diagrama de apresentação para a classe cidade ponto
Fonte: (Borges, 2005)

Cada classe georreferenciada especificada no diagrama de classes precisa ter pelo menos uma apresentação correspondente especificada no diagrama de apresentação. Caso exista mais de uma apresentação para uma dada representação, uma delas deve ser identificada como a *default*. Alternativamente, cada usuário ou aplicação pode eleger sua apresentação *default*.

As operações TA mais comuns envolvem a simples definição de atributos gráficos. No entanto, outros operadores mais sofisticados, muitos dos quais derivados de operações da cartografia temática (classificação, simbolização, exagero, deslocamento, destaque) também podem ser empregados. Uma descrição detalhada dos operadores TA pode ser encontrada em (Davis Jr., 2000) (Davis and Laender, 1999).

2.1.4 Ferramenta CASE

Para construir esquemas no modelo OMT-G, foi desenvolvida uma extensão (*Stencil*) para o software Microsoft Visio 2000 por Karla Borges (Borges), mostrado na Figura 2.12.

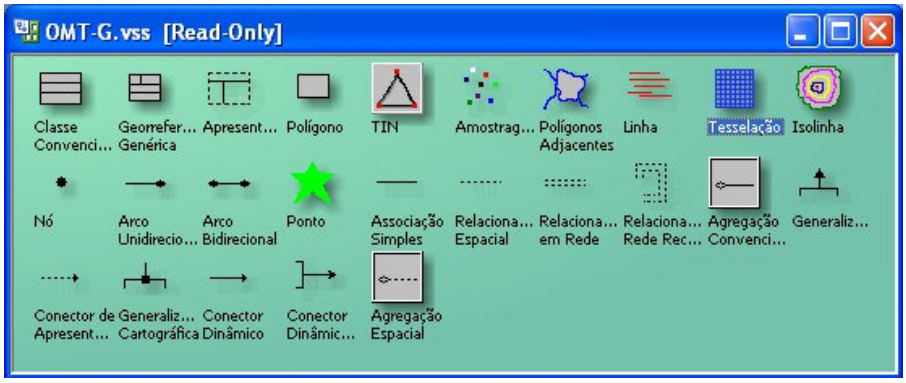


Figura 2.12 – Extensão OMT-G para o software Microsoft Visio

2.2 Framework GeoFrame

GeoFrame é um framework conceitual proposto por Lisboa e Ioche (Lisboa and Iochpe), baseado no paradigma de orientação a objetos, para ser usado na modelagem conceitual de bancos de dados geográficos. Este framework utiliza as notações e conceitos da linguagem UML. O objetivo de um framework conceitual é fornecer um conjunto de classes genéricas para um determinado domínio de aplicação para ser usadas como base ou molde para modelar aplicações específicas dentro desse domínio. O produto final de um framework conceitual é um esquema conceitual de dados.

2.2.1 Diagrama de classes GeoFrame

O diagrama de classes GeoFrame, mostrado na Figura 2.13, utiliza as mesmas notações gráficas do diagrama de classes UML e é formado por quatro classes principais: `RegiãoGeográfica` (`GeographicRegion`); `Tema` (`Theme`); `ObjetoNãoGeográfico` (`NonGeographicObject`) e `FenômenoGeográfico` (`GeographicPhenomenon`). Essas classes generalizam, em um alto nível de abstração, os elementos de um esquema de dados geográficos.

As classes `Tema` e `RegiãoGeográfica` são básicas para qualquer aplicação geográfica. A grande maioria das aplicações geográficas tem como principal objetivo gerenciar e manipular um conjunto de dados de uma determinada região de interesse, gerando uma base de dados geográficos referentes a essa região. Por exemplo, em uma aplicação urbana a região geográfica de interesse pode ser a área da cidade. Para essa mesma região geográfica podem ser definidos diferentes temas, por exemplo, limites das zonas urbanas, limites das zonas rurais, ruas, construções (escolas, hospitais, etc) e rede de transporte público. É possível que alguns temas sejam definidos para algumas subáreas, por exemplo, zonas de segurança e de perigo da área central da cidade. Portanto, uma coleção de temas pode ser definida para cada área geográfica.

Os temas definidos em um esquema GeoFrame não são necessariamente implementados como um plano de informação ou *layer* em um SIG. Dependendo do aplicativo usado e das necessidades do usuário, um tema definido durante a modelagem conceitual pode ser implementado como vários *layers*. Por exemplo, um tema de rios pode ser implementado em dois *layers* diferentes, um contendo somente objetos espaciais com representação geométrica linear e outro contendo objetos espaciais com representação geométrica poligonal.

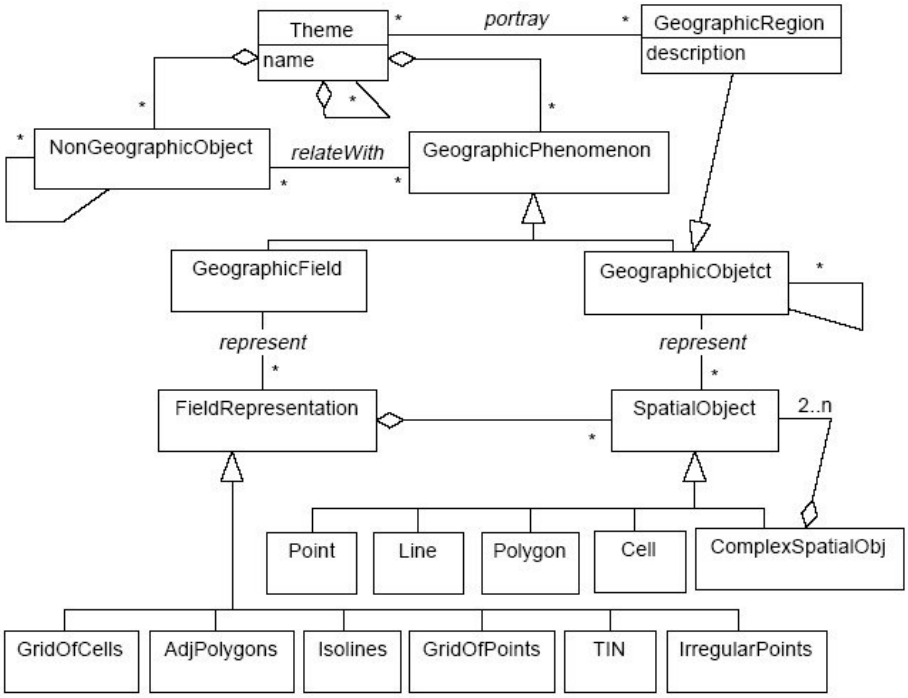


Figura 2.13 – Framework GeoFrame
Fonte: (Lisboa and Iochpe, 1999)

Em bancos de dados geográficos, como em qualquer sistema de informação, existem objetos não geográficos ou convencionais. Em GeoFrame, esses objetos são considerados como instâncias da classe `ObjetoNãoGeográfico` (`NonGeographicObject`). A classe abstrata `FenômenoGeográfico` (`GeographicPhenomenon`) generaliza qualquer fenômeno cuja localização em relação a superfície da Terra é considerada, chamado então de fenômeno geográfico. Por exemplo, um lote pode ser considerado um fenômeno geográfico, e ser representado como uma instância de `FenômenoGeográfico`, se seus atributos espaciais devem ser representados na base de dados.

As classes `CampoGeográfico` (`GeographicField`) e `ObjetoGeográfico` (`GeographicObject`) especializam a classe `FenômenoGeográfico` segundo os modelos geo-campo e geo-objeto apresentados no capítulo 1. A classe abstrata `ObjetoGeográfico` é uma generalização de todas as classes que representam geo-objetos, ou seja, que representam fenômenos geográficos que podem ser individualizados (possuindo uma identidade e características que podem ser descritas através de um conjunto de atributos). Por exemplo, as classes Lotes, Rios, Estradas e Cidades podem ser modeladas como uma especialização da classe `ObjetoGeográfico`. Por outro lado, a classe abstrata `CampoGeográfico` generaliza todas as classes que representam geo-campos.

Fenômenos geográficos e objetos não geográficos podem ser relacionados entre si. Para isso a associação *relateWith* (Figura 2.13), herdada da classe `FenômenoGeográfico`, pode ser especializada para representar esse relacionamento. A Figura 2.14 mostra um exemplo de um esquema conceitual de uma aplicação urbana onde a classe *Parcel*, uma subclasse de `ObjetoGeográfico`, está associada com as classes *Owner* e *TaxType* subclasses de `ObjetoNãoGeográfico`, e com as classes *Block* e *Street*, subclasses de

`ObjetoGeográfico`. Neste exemplo, para representar que todo lote pertence a algum proprietário, foi definido um relacionamento chamado *own* entre as classes *Parcel* e *Owner*.

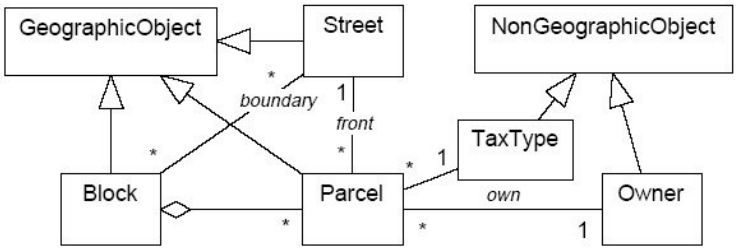


Figura 2.14 –Exemplos de relacionamento *relateWith*
Fonte: (Lisboa and Iochpe, 1999)

Um fenômeno geográfico é formado por três componentes: atributo, espaço e tempo. Essas três componentes devem ser abstraídas do mundo real durante o processo da modelagem conceitual. Porém, a escolha da melhor estrutura de dados para implementar a componente espacial de cada fenômeno geográfico é uma tarefa para ser executada em outros níveis, após a modelagem conceitual. No modelo GeoFrame, quando o projetista associa pontos, polígonos ou grade de células aos fenômenos geográficos, ele está considerando as formas de abstração da componente espacial do fenômeno geográfico, mas não a maneira de como eles serão estruturados no banco de dados. Por exemplo, na modelagem conceitual o principal objetivo é determinar se uma rua terá uma dimensão linear ou poligonal. Portanto, não é necessário, neste nível, definir que essa rua será armazenada como um arco direcionado em uma estrutura vetorial com topologia.

A classe `ObjetoEspacial` (`SpatialObject`) generaliza as classes que são usadas para representar a componente espacial de geo-objetos. Essas classes são: `Ponto` (`Point`), `Linha` (`Line`), `Polígono` (`Polygon`), `Célula` (`Cell`) e `ObjetoEspacialComplexo` (`ComplexSpatialObj`). Para representar a componente espacial de geo-campos, o GeoFrame considera seis modelos espaciais descritos por Goodchild (Goodchild): grade de células, polígonos adjacentes, isolinhas, grade de pontos, rede triangular irregular (TIN) e amostras pontuais irregulares. Esses modelos são representados pelas classes `GradeDeCelulas` (`GridOfCells`), `PolígonosAdjacentes` (`AdjPolygon`), `Isolinas` (`Isolines`), `GradeDePontos` (`GridOfPoints`), `TIN` (`TIN`) e `PontosIrregulares` (`IrregularPoints`), que são subclasses da classe `RepresentaçãoCampo` (`FieldRepresentation`). Na fase de implementação, estes modelos devem ser mapeados para os modelos de representação matricial e vetorial, mostrados no capítulo 1. Cada um dos seis modelos de geo-campos pode ser implementado tanto no modelo vetorial quanto no matricial, embora alguns mapeamentos sejam mais naturais.

Um geo-campo pode ter sua componente espacial abstraída de diferentes maneiras e, portanto, ser modelado através de mais de um modelo de representação. Por exemplo, o geo-campo Temperatura pode ser abstraído como amostras pontuais irregulares ou como isolinhas. Uma situação similar acontece com os geo-objetos quando sua componente espacial deve ser representada através de mais de um tipo de geometria. A possibilidade de ter múltiplas representações é mostrada na Figura 2.13, através da associação *represent* de cardinalidade (1:n).

O modelo GeoFrame usa os relacionamentos entre classes definidos na linguagem UML: associação, especialização, agregação e composição. Os relacionamentos entre classes permitem a definição de restrições de integridade que devem ser implementadas nos bancos de dados para manter a consistência dos dados. Os relacionamentos podem ser divididos em três categorias: semântico, espacial e temporal. Os relacionamentos semânticos complementam a descrição do conhecimento, considerando os aspectos descritivos do fenômeno (por exemplo, um bloco pode conter vários lotes ou cada lote possui um proprietário). Relacionamentos espaciais estabelecem associações entre as localizações dos fenômenos (por exemplo, o limite espacial de um bloco contém os limites espaciais dos seus lotes). Os relacionamentos temporais ainda estão sendo estudados.

2.2.2 Esquema conceitual

Um esquema conceitual é construído através da especialização das classes abstratas do GeoFrame. Porém, para simplificar a construção desses esquemas e torná-los mais legíveis, foi proposto um conjunto de estereótipos que representam as especializações das classes GeoFrame. Os estereótipos utilizados para representar especializações das classes `NonGeographicObject`, `GeographicField` e `GeographicObject` são mostrados na Figura 2.15. E os estereótipos utilizados para representar especializações das classes `SpatialObject` e `FieldRepresentation` são ilustrados na Figura 2.16.

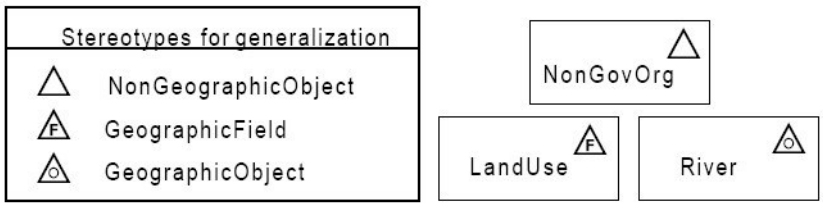


Figura 2.15- Estereótipos para generalização

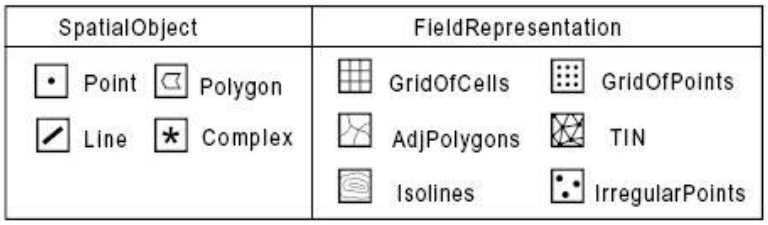


Figura 2.16 - Estereótipos para a associação *represent*
Fonte: (Lisboa and Iochpe, 1999)

A modelagem conceitual de um banco de dados geográficos utilizando o framework GeoFrame é feita considerando uma abordagem *topdown* de três fases. Inicialmente, para cada área geográfica considerada são identificados os possíveis temas e subtemas. Na segunda fase, um diagrama de classes é especificado para cada tema identificado. As associações entre classes de diferentes temas são determinadas também nessa fase. Finalmente, a análise e a modelagem da componente espacial de cada fenômeno geográfico são feitos. A seguir, as três fases são discutidos com mais detalhes.

2.2.2.1 Fase 1) Diagrama de Temas

Cada região geográfica é representada por zero ou mais temas. Porém, um tema pode estar associado com mais de uma região geográfica. Um tema é definido como uma agregação de classes de fenômenos geográficos, de classes não geográficas ou de outros temas. Para aumentar a legibilidade dos esquemas resultantes, temas não são modelados como subclasses da classe `Theme`, mas como um conceito de UML chamado *package*. Uma *package* é composta por um conjunto de elementos UML que podem ser de qualquer tipo, por exemplo, classes, associações e outras *packages*. A Figura 2.17 mostra um exemplo de um diagrama de temas. Como pode ser visto na Figura 2.17, os temas são associados com instâncias da classe `GeographicRegion`. Nesta faze, os temas devem ser definidos para cada região do projeto.

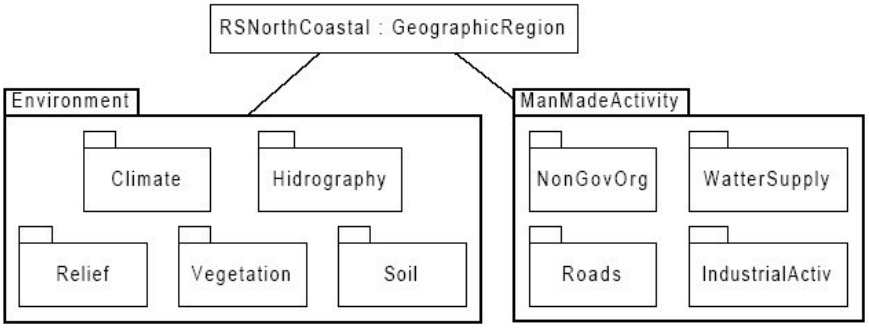


Figura 2.17 – Um exemplo de diagramas de temas
Fonte: (Lisboa and Iochpe, 1999)

2.2.2.2 Fase 2) Identificação dos fenômenos geográficos e objetos não geográficos

Depois da fase de identificação de temas, cada tema deve ser refinado. Na abordagem orientada a objetos, um esquema de banco de dados geográfico é representado pelo diagrama de classes, o qual descreve os fenômenos geográficos, os objetos não geográficos e os possíveis relacionamentos entre eles. Nesta fase, cada classe identificada deve ser modelada como uma subclasse de uma das seguintes classes GeoFrames: `NonGeographicObject`, `GeographicField` e `GeographicObject`. Nesta fase, os estereótipos são usados para evitar uma difícil visualização do diagrama.

2.2.2.3 Fase 2) Representação das componentes espaciais dos fenômenos geográficos

Em GeoFrame, todo geo-campo ou geo-objeto podem ser representado por múltiplas instâncias das classes `FieldRepresentation` e `SpatialObject`, respectivamente. Um fenômeno geográfico pode ter múltiplas representações por várias razões, dentre elas, a necessidade de múltiplas escalas, usuários com diferentes visões de um mesmo fenômeno e versões temporárias. A possibilidade de ter múltiplas representações para um fenômeno geográfico existe em GeoFrame e é modelada através de diferentes associações entre o fenômeno geográfico e as possíveis representações de sua componente espacial (associação *represent* na Figura 2.13). Em GeoFrame, as diferentes representações espaciais são definidas através de uma livre combinação de diferentes estereótipos em uma mesma classe.

A Figura 2.18 mostra um exemplo de um esquema conceitual GeoFrame para uma aplicação urbana. Neste exemplo, os mecanismos para simplificação de esquema são mostrados. Algumas classes não apresentam estereótipos (por exemplo *BuiltParcel*) porque os respectivos relacionamentos são herdados de sua superclasse.

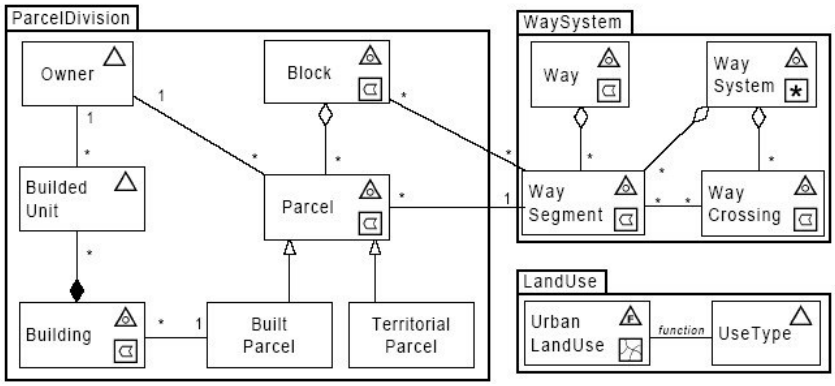


Figura 2.18 – Exemplo de um diagrama de classes usando GeoFrame
Fonte: (Lisboa and Iochpe, 1999)

2.2.3 Ferramenta CASE

O software ArgoCASEGEO (Lisboa, 2004) é um ferramenta CASE livre e de código fonte aberto, disponível no endereço www.dpi.ufv.br/projetos/argocasegeo, que permite a modelagem de banco de dados geográficos com base no modelo conceitual GeoFrame. Essa ferramenta também suporta aspectos simples de modelagem temporal, bem como a geração automática de esquemas lógicos de bancos de dados, em formato Shapefile ou TerraLib. Este aplicativo, mostrado na Figura 2.19, tem como base o software ArgoUML e está sendo desenvolvida no Departamento de Informática da Universidade Federal de Viçosa (UFV).

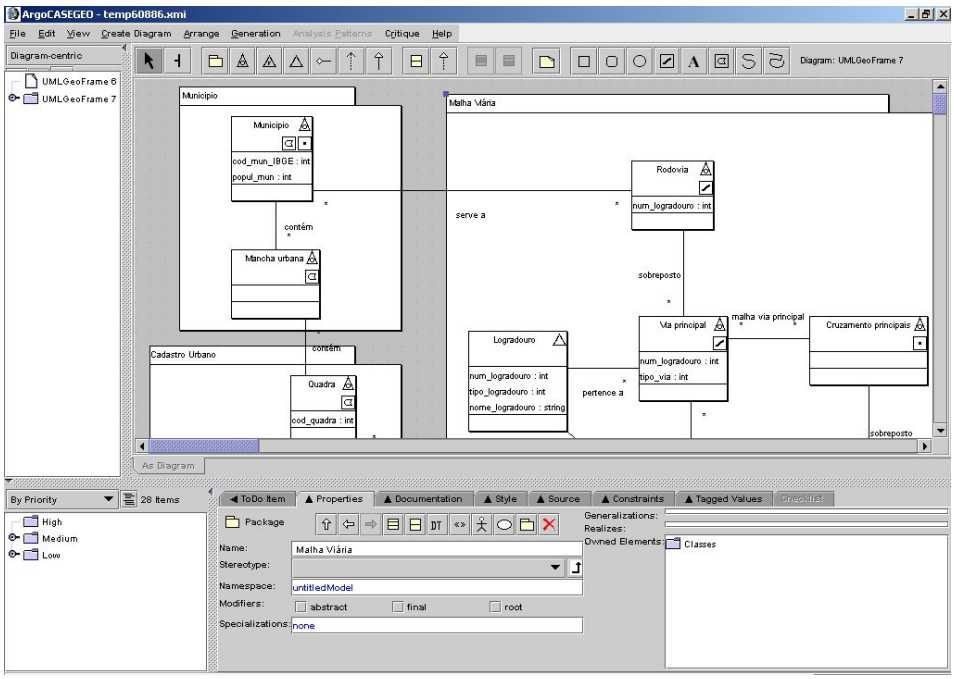


Figura 2.19 – ferramenta ArgoCASEGEO

2.3 Exemplo de modelagem

Para exemplificar o uso das principais primitivas dos modelos OMT-G e GeoFrame, apresentamos nesta seção um exemplo de modelagem. A aplicação proposta combina aspectos de interesse em três diferentes contextos:

- Cadastro municipal: os usuários estão interessados na estruturação da ocupação do solo urbano em quadras, lotes e vias públicas.
- Gerenciamento de transportes e trânsito: interesse está na estruturação do sistema viário.
- Mapeamento em escala regional: os usuários se interessam apenas pelos principais aspectos de ocupação do território e acessos, em especial a malha rodoviária.

Alguns objetos do ambiente urbano são necessários nos três contextos, como por exemplo, o sistema viário. No entanto, cada um deles percebe esses objetos de uma maneira diferente, gerando a necessidade de mais de uma representação.

Para os usuários da área de cadastro municipal, os principais objetos são as quadras e lotes da cidade. No caso dos lotes, adotamos duas diferentes alternativas para representação, pontos e polígonos. Usuários que executam consultas simples e que não dependem da geometria exata de cada lote, como por exemplo, "*quantos lotes existem na quadra Q1?*", utilizam a representação pontual. Usuários que precisam executar consultas baseadas na geometria exata de cada lote, como por exemplo, "*quais os vizinhos do lote L1?*" e "*quais os lotes cuja área é maior que 100.000 m2?*", utilizam a representação poligonal. Além disso, nessa aplicação precisamos das informações dos proprietários dos lotes.

O relacionamento dos lotes e quadras com o sistema de endereçamento da cidade, através da malha viária, é também desejável, para que seja possível simplificar a tarefa de localizá-los em campo e também para facilitar a comunicação com os proprietários e outros cidadãos.

Para que se possa trabalhar com transportes e trânsito, é fundamental poder contar com todas as informações relevantes quanto à malha viária, incluindo a localização de cada logradouro e cada cruzamento entre logradouros. Na presente aplicação, a malha viária básica será representada por uma rede, em que arcos bidirecionais representam os segmentos de logradouro entre cruzamentos, que por sua vez constituem os nós. Cada trecho de logradouro recebe uma classificação de acordo com o Plano de Classificação Viária do município que define vias de ligação regional, arteriais, coletoras e locais. Essa classificação depende basicamente do volume de tráfego e da função primária de cada trecho de logradouro. Portanto, essa classificação deve ser um atributo do trecho e não do logradouro inteiro, pois existem situações em que parte do logradouro recebe tráfego intenso e parte tem características de via local. Considerando apenas as vias mais importantes para a circulação, concebe-se uma nova rede, esta adequada para o planejamento da circulação de veículos entre regiões da cidade.

Por fim, os responsáveis pelo mapeamento regional estão interessados em obter os limites da área urbanizada da cidade, usualmente denominados "mancha urbana", além das ligações da cidade a outras por meio de rodovias e outros meios de transporte de cargas. As ferrovias foram deixadas de fora do problema por simplicidade, mas as rodovias que atravessam a cidade fazem parte da malha viária

concebida para a aplicação de transportes e trânsito. Além disso, considerando as escalas em que se pretende construir o mapeamento regional, é interessante poder contar com (1) uma representação simplificada do polígono que compõe os limites entre a cidade e suas vizinhas, e (2) uma representação da cidade como ponto, para a geração de mapas temáticos sobre transportes rodoviários.

Para modelar a aplicação proposta, foi construído um diagrama de classes OMT-G e GeoFrame, mostrados nas Figuras 2.20 e 2.21, respectivamente.

No diagrama mostrado na Figura 2.20, a primitiva de generalização conceitual do modelo OMT-G foi usada duas vezes, uma para a classe `Município`, que pode ter representações pontuais ou poligonais, e outra para a classe `Lote`, que podem ser representados por pontos ou polígonos. Existe também uma primitiva de agregação, que indica que as instâncias da classe `Quadra` serão criadas pela agregação de instâncias de `Lote`. A classe `Rodovia` está relacionada à classe de `Via Principal`, assumindo a regra de que todos os trechos de rodovia são classificados como vias de ligação regional. A classe `Via principal`, por sua vez, é um subconjunto da classe `Trecho`, pois nem todo trecho de logradouro pertence à malha viária principal. Com isso, nem todos os nós de cruzamento constituem interseções na malha viária principal. O diagrama de classes indica, assim, que existe uma superposição parcial entre a malha de logradouros e a malha viária principal, mas não determina a forma de estruturação do banco de dados geográfico quanto a esse aspecto.

No diagrama GeoFrame mostrado na Figura 2.21, a aplicação foi dividida em três pacotes (*packages*): Malha viária, Cadastro Urbano e Município. No GeoFrame, para identificar que um relacionamento entre duas classes é espacial devemos colocar o estereótipo <<*spatial*>> associado a esse relacionamento. Esse modelo não oferece recursos para representar arcos direcionados, portanto, as classes `Via Principal` e `Trecho` foram representadas como linhas. Por último, múltiplas representações são representadas através de um conjunto de estereótipos associados à classe, como nas classes `Lote` e `Município`.

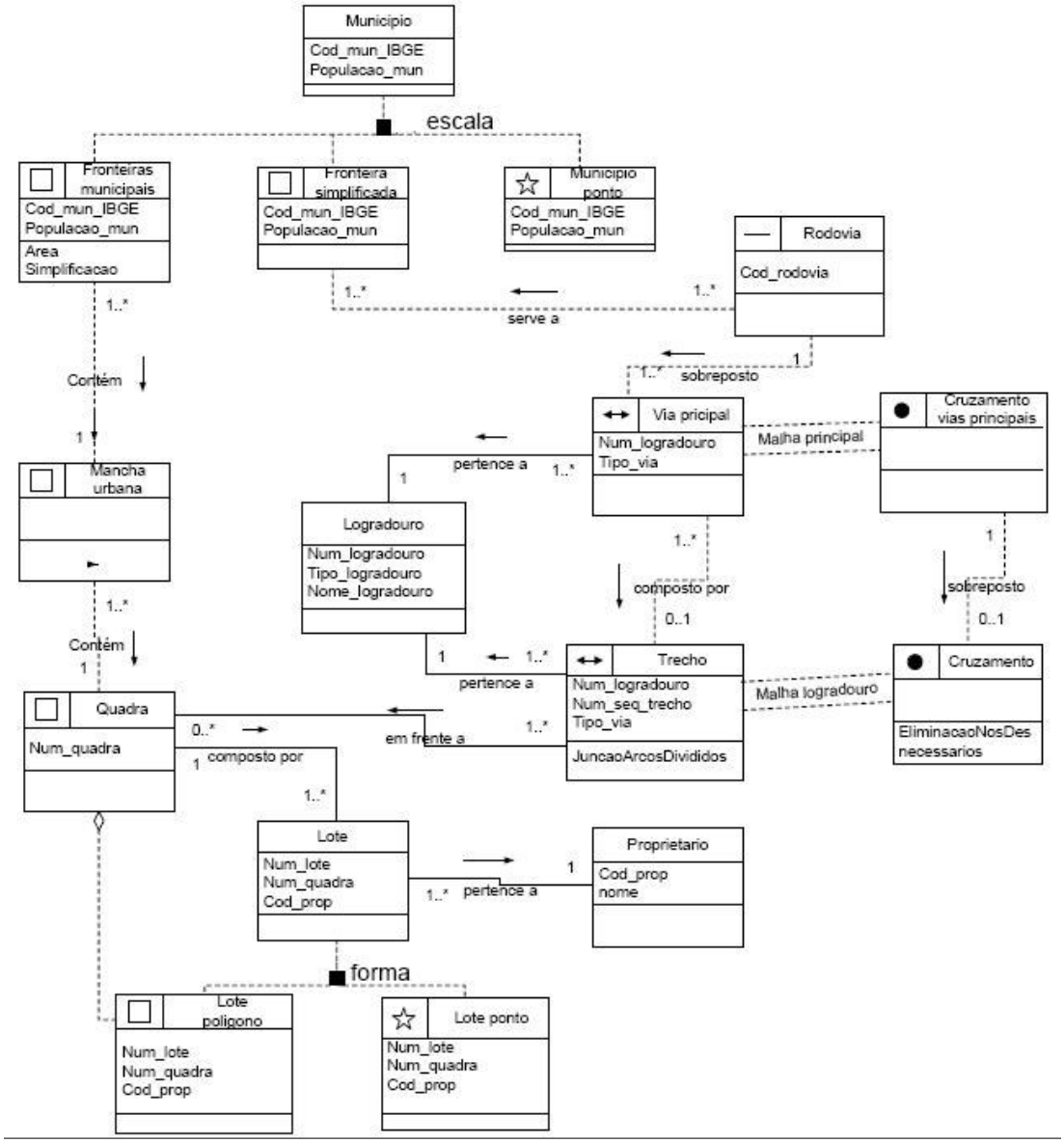


Figura 2.20 – Diagrama de classes OMT-G

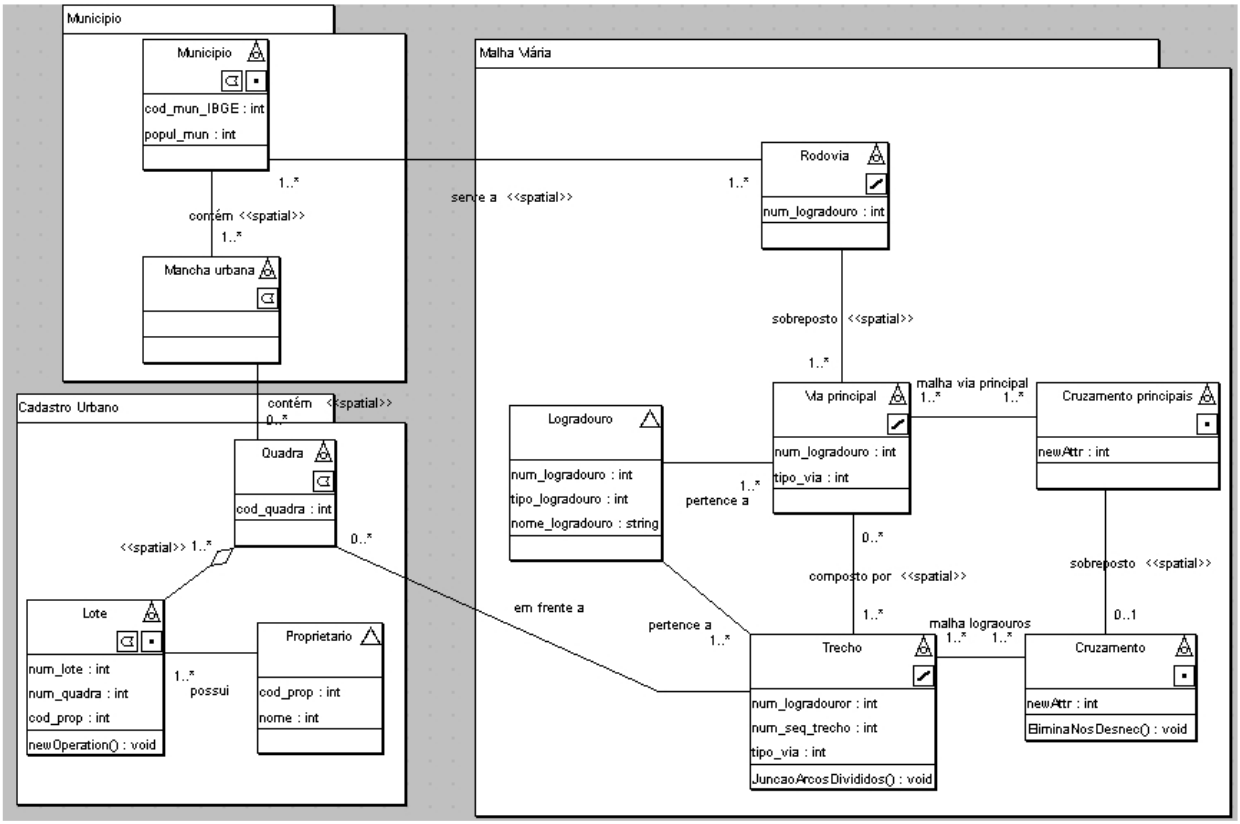


Figura 2.21 – Diagrama de classes GeoFrame

3 Sistemas de Informações Geográficas e Bancos de Dados Geográficos

Ao longo dos anos, as implementações de SIGs seguiram diferentes arquiteturas, distinguindo-se principalmente pela estratégia adotada para armazenar e recuperar dados espaciais. Mais recentemente, tais arquiteturas evoluíram para utilizar, cada vez mais, recursos de SGBDs.

Por seu lado, a pesquisa na área de Banco de Dados passou, já há algum tempo, a preocupar-se com o suporte a aplicações não convencionais (Schneider, 1997), incluindo as aplicações SIG. Uma aplicação é classificada como não convencional quando trabalha com outros tipos de dados, além dos tradicionais, como tipos de dados espaciais, temporais e espaço-temporais.

3.1 Preliminares

3.1.1 Sistemas de gerência de banco de dados

Um sistema de gerência de banco de dados (SGBD) oferece serviços de armazenamento, consulta e atualização de bancos de dados. A Tabela 3.1 resume os requisitos mais importantes para SGBDs e a
Tabela 3-2 lista as principais tecnologias desenvolvidas para atendê-los.

Tabela 3-1 – Principais requisitos para SGBDs

Requisito	*Definição*
Facilidade de uso	a modelagem do banco de dados deve refletir a realidade das aplicações, e o acesso aos dados deve ser feito de forma simples
Correção	os dados armazenados no banco de dados devem refletir um estado correto da realidade modelada
Facilidade de manutenção	alterações na forma de armazenamento dos dados devem afetar as aplicações o mínimo possível
Confiabilidade	atualizações não devem ser perdidas e não devem interferir umas com as outras
Segurança	o acesso aos dados deve ser controlado de acordo com os direitos definidos para cada aplicação ou usuário
Desempenho	o tempo de acesso aos dados deve ser compatível com a complexidade da consulta

Tabela 3-2 – Principais tecnologias para SGBDs

Requisito	Tecnologia
Facilidade de uso	linguagem de definição de dados e linguagem de consulta baseadas em modelo de dados de alto nível
Correção	restrições de integridade, *triggers* e assertivas
Facilidade de manutenção	especificação do banco de dados em níveis, isolando os detalhes de armazenamento das aplicações
Confiabilidade	transações atômicas implementadas através de mecanismos para controle de concorrência e mecanismos de recuperação em caso de falhas
Segurança	níveis de autorização e controle de acesso

Desempenho	otimização de consultas, baseada em métodos de acesso e de armazenamento eficientes, gerência eficaz do buffer pool e modelos de custo, entre outras tecnologias

O mercado para SGBDs concentra-se em duas tecnologias, SGBDs Relacionais (SGBD-R) e SGBDs Objeto-Relacionais (SGBD-OR), com uma pequena fatia para SGBDs Orientados-a-Objeto (SGBD-OO).

Os SGBD-R seguem o modelo relacional de dados, em que um banco de dados é organizado como uma coleção de relações, cada qual com atributos de um tipo específico. Nos sistemas comerciais atuais, os tipos incluem números inteiros, de ponto flutuante, cadeias de caracteres, datas e campos binários longos (BLOBs). Para esses tipos encontram-se disponíveis uma variedade de operações (exceto para o tipo BLOB), como operações aritméticas, de conversão, de manipulação textual e operações com data.

Os SGBD-R foram concebidos para atender as necessidades de aplicações manipulando grandes volumes de dados convencionais. De fato, tais sistemas não oferecem recursos para atender as necessidades de aplicações não convencionais. A mera simulação de tipos de dados não convencionais em um SGBD-R pode ter efeitos colaterais, como queda de desempenho, dificuldade de codificação e posterior manutenção da aplicação (Stonebraker, 1996).(Stonebraker, 1996)

Os SGBD-OR estendem o modelo relacional, entre outras características, com um sistema de tipos de dados rico e estendível, oferecendo operadores que podem ser utilizados na linguagem de consulta. Possibilitam ainda a extensão dos mecanismos de indexação sobre os novos tipos. Essas características reduzem os problemas ocorridos na simulação de tipos de dados pelos SGBD-R, tornando os SGBD-OR uma solução atrativa para aplicações não convencionais.

3.1.2 A linguagem SQL

A linguagem SQL (*Structured Query Language*) é adotada pela maioria dos SGBD-R e SGBD-OR comerciais. Desenvolvida inicialmente pela IBM na década de 70, a linguagem sofreu sucessivas extensões, culminando com a publicação do padrão conhecido por SQL:1999 (ver Tabela 3.3).

Tabela 3-3 – Breve histórico de SQL

Ano	Versão	Características
1974	SEQUEL	linguagem original, adotada no protótipo de mesmo nome, desenvolvido pela IBM
1976	SEQUEL 2	extensão de SEQUEL, adotada no *System R* da IBM
1986	SQL-86 (SQL1)	padrão publicado pela ANSI em 1986 ratificado pela ISO em 1987
1989	SQL-89	extensão do SQL-86
1992	SQL-92 (SQL2)	padrão publicado pela ANSI e pela ISO
1996	SQL-92 / PSM	extensão do SQL-92
2001	SQL:1999	padrão aprovado em 1999 pela ISO, resultado de 7 anos de trabalho, e publicado em maio de 2001

A SQL é formada basicamente por duas sub-linguagens:

- Linguagem de definição de dados (SQL DDL): fornece comandos para definir e modificar esquemas de tabelas, remover tabelas, criar índices e definir restrições de integridade.
- Linguagem de manipulação de dados (SQL DML): fornece comandos para consultar, inserir, modificar e remover dados no banco de dados.

A Tabela 3.4 apresenta alguns comandos em SQL.

Tabela 3-4 – Comandos em SQL

Comando	*Descrição*	*Tipo*
select	Recupera dados de uma ou mais tabelas	DML
insert *update* *delete*	Servem para incluir, alterar e eliminar registros de uma tabela, respectivamente	DML
commit *roolback*	Responsáveis pelo controle de transações, permitem que o usuário desfaça (*rollback*) ou confirme (*commit*) alterações em tabelas	DML
create *alter* *drop*	Usados para definir, alterar e remover tabelas de um banco de dados	DDL

3.2 Arquiteturas de SIGs

A partir desta seção, usaremos os conceitos introduzidos na Tabela 3.5 e definidos em detalhe ao longo das seções seguintes ou em outros capítulos deste texto.

Tabela 3-5 – Resumo dos principais conceitos relativos a bancos de dados espaciais

Conceito	Definição
geometria matricial	uma matriz de elementos, usualmente pertencentes a um sub-conjunto dos números reais $\Re$
geometria vetorial	um elemento do $\Re^2$, considerado como um espaço topológico. Pontos, linhas e regiões são particulares geometrias.
geometria	uma geometria vetorial ou matricial
atributo espacial	um atributo de um objeto cujo domínio seja um conjunto de geometrias.
objeto espacial	qualquer objeto com um atributo espacial. Os geo-objetos são uma classe particular de objetos espaciais.
componente espacial ou geometria de um objeto	valor de um atributo espacial de um objeto.
banco de dados espacial	um banco de dados armazenando, entre outros, objetos espaciais.
consulta espacial	uma consultas definida sobre um banco de dados espacial.

Existem basicamente duas principais formas de integração entre os SIGs e os SGBDs, que são a *arquitetura dual* e a *arquitetura integrada*. A *arquitetura dual*,

mostrada na Figura 3.1 armazena as componentes espaciais dos objetos separadamente. A componente convencional, ou alfanumérica, é armazenada em um SGBD relacional e a componente espacial é armazenada em arquivos com formato proprietário. Os principais problemas dessa arquitetura são:

- Dificuldade no controle e manipulação das componentes espaciais.
- Dificuldade em manter a integridade entre a componente espacial e a componente alfanumérica.
- Separação entre o processamento da parte convencional, realizado pelo SGBD, e o processamento da parte espacial, realizado pelo aplicativo utilizando os arquivos proprietários.
- Dificuldade de interoperabilidade, já que cada sistema trabalha com arquivos com formato proprietário.

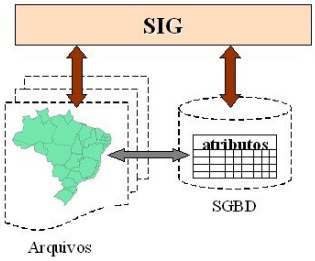


Figura 3.1 – Arquitetura Dual

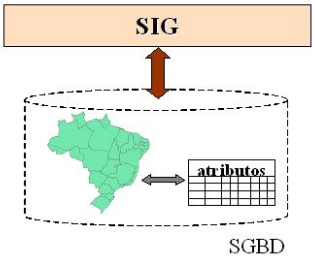


Figura 3.2 – Arquitetura Integrada

A *arquitetura integrada*, mostrada na Figura 3.2, consiste em armazenar todos os dados em um SGBD, ou seja, tanto a componente espacial quanto a alfanumérica. Sua principal vantagem é a utilização dos recursos de um SGBD para controle e manipulação de objetos espaciais, como gerência de transações, controle de integridade, concorrência e linguagens próprias de consulta. Sendo assim, a manutenção de integridade entre a componente espacial e alfanumérica é feita pelo SGBD.

Esta última arquitetura pode ainda ser subdividida em três outras: *baseada em campos longos, em extensões espaciais* e *combinada*.

A *arquitetura integrada baseada em campos longos* utiliza BLOBs para armazenar a componente espacial dos objetos. Como comentado anteriormente, um SGBD-R ou -OR trata um BLOB como uma cadeia de bits sem nenhuma semântica adicional. Portanto, esta arquitetura apresenta algumas desvantagens:

- Um BLOB não possui semântica.
- Um BLOB não possui métodos de acesso.
- SQL oferece apenas operadores elementares de cadeias para tratar BLOBs.

Portanto, ao codificar dados espaciais em BLOBs, esta arquitetura torna a sua semântica opaca para o SGBD. Ou seja, passa a ser responsabilidade do SIG implementar os operadores espaciais, capturando a semântica dos dados, e métodos de acesso que possam ser úteis no processamento de consultas, embora seja bastante difícil incorporá-los ao sistema de forma eficiente.

A *arquitetura integrada com extensões espaciais* consiste em utilizar extensões espaciais desenvolvidas sobre um SGBD-OR. Esta arquitetura oferece algumas vantagens:

- Permite definir tipos de dados espaciais, equipados com operadores específicos (operadores topológicos e métricos).
- Permite definir métodos de acesso específicos para dados espaciais.

Exemplos dessa arquitetura são o Oracle Spatial e o PostGIS, apresentados no Capítulo 5. As extensões espaciais utilizadas nas arquiteturas integradas possuem as seguintes características (Güting, 1994):

- fornecem tipos de dados espaciais (TDEs) em seu modelo de dados, e mecanismos para manipulá-los.
- estendem SQL para incluir operações sobre TDEs, transformando-a de fato em uma linguagem para consultas espaciais.
- adaptam outras funções de nível mais interno ao SGBD para manipular TDEs eficientemente, tais como métodos de armazenamento e acesso, e métodos de otimização de consultas.

Normalmente, essas extensões tratam somente objetos espaciais cuja componente espacial seja uma geometria vetorial, utilizando BLOBs para armazenar dados matriciais, com todos os problemas já citados.

De fato, no caso de aplicativos SIG que manipulam objetos com geometrias tanto matriciais quanto vetoriais, é possível a utilização de uma *arquitetura integrada combinada*, formada pela combinação das duas últimas. Ou seja, as geometrias vetoriais são armazenadas utilizando-se os recursos oferecidos pelas extensões e as geometrias matriciais são armazenadas em BLOBs. As funcionalidades para manipulação de geometrias matriciais são fornecidas por uma camada externa ao SGBD, de modo a complementar os recursos ausentes, até o momento, nas extensões.

3.3 Operações Espaciais

As consultas espaciais baseiam-se em relacionamentos espaciais de vários tipos: métricos, direcionais e topológicos. Por serem dois conceitos de natureza distinta, as operações sobre as componentes espaciais de geo-campos e geo-objetos também são diferentes. Abordaremos nesta seção apenas operações sobre as geometrias vetoriais de geo-objetos. Portanto, no que se segue, omitiremos o adjetivo "vetorial". As operações espaciais podem ser classificadas em (Rigaux et al., 2002):

- *Operação unária booleana:* mapeia geometrias em valores booleanos. Como exemplos, temos: *Convex,* que testa se uma geometria é convexa; e *Connected,* que testa se uma geometria está conectada.
- *Operação unária escalar:* mapeia geometrias em valores escalares. Como exemplos, temos: *Length,* que computa o comprimento ou perímetro de uma geometria; e *Área,* que computa a área de uma geometria.
- *Operação unária espacial:* mapeia geometrias em geometrias. Como exemplos, temos: *Buffer,* que retorna uma nova geometria a partir de uma distância em torno de uma geometria específica; *ConvexHull*, que retorna uma geometria convexa a partir da geometria; MBR, que retorna o

mínimo retângulo envolvente de uma geometria; e *Centroid,* que retorna o centróide de uma geometria.

- *Operação binária booleana:* também chamada de *predicado espacial* ou *relacionamento espacial*, mapeia pares de geometrias em valores booleanos. Esta classe pode ser dividida em:
    - *Relacionamento topológico:* um relacionamento que não é alterado por transformações topológicas, como translação, rotação e mudança de escala. Como exemplos, temos: contém (*contains*), disjunto (*disjoint*), intercepta (*intersects*), cruza (*crosses*), como apresentado na seção seguinte.
    - *Relacionamento direcional:* um relacionamento que expressa uma noção de direção. Como exemplos, temos: acima de (*above*), ao norte de (*northOf*), dentre outras.
    - *Relacionamento métrico:* um relacionamento que expressa uma noção métrica. Por exemplo, o relacionamento que retorna Verdadeiro se duas geometrias estão a menos de uma determinada distância uma da outra.
- *Operação binária escalar:* mapeia pares de geometrias em valores escalares. Por exemplo, *distance* computa a distância entre duas geometrias.
- *Operação binária espacial:* mapeia pares de geometrias em geometrias. Como exemplos, temos as operações de conjunto, como interseção (*Intersection*), união (*Union*) e diferença (*Difference*).
- *Operação n-ária espacial:* mapeia n-tuplas de geometrias em geometrias. Por exemplo, a operação *ConvexHull* pode pertencer a essa classe quando receber mais de uma geometria como parâmetro de entrada.

3.4 Relacionamentos Topológicos

Existem várias propostas de modelos com o objetivo de descrever os possíveis relacionamentos entre dois objetos. Os modelos adotados nas implementações da maioria dos SIGs seguem o paradigma das matrizes de interseção introduzida por Max Egenhofer.

No modelo chamado de *matriz de 4-interseções* (ver a Figura 3.3)*,* oito relações topológicas binárias são consideradas, representando a interseção entre a fronteira e o interior de duas geometrias (Egenhofer and Franzosa, 1991).

Para definir relacionamentos topológicos entre geometrias com estruturas mais complexas, como regiões com ilhas e separações, é necessário estender a matriz de 4-Interseções para também considerar o exterior de uma geometria. O novo modelo, chamado de *matriz de 9-Interseções* (ver Figura 2.20), considera então o resultado da interseção entre as fronteiras, interiores e exteriores de duas geometrias. Maiores detalhes sobre relações topológicas entre regiões com ilhas podem ser encontrado em (Egenhofer and Herring, 1991).

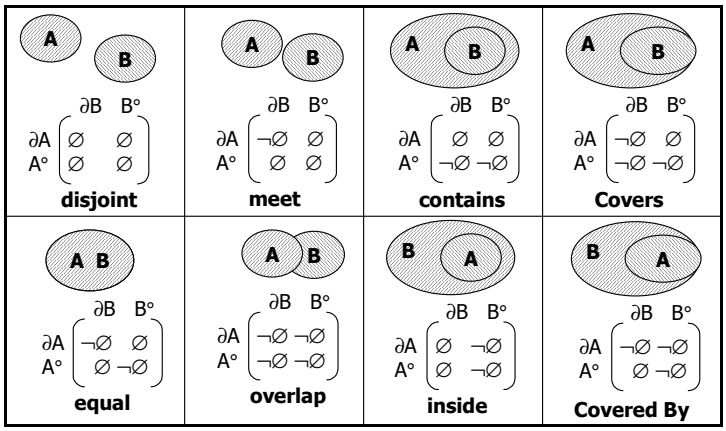


Figura 3.3 – Matriz de 4-Interseções para relações entre duas regiões.
Fonte: (Egenhofer and Franzosa, 1991).

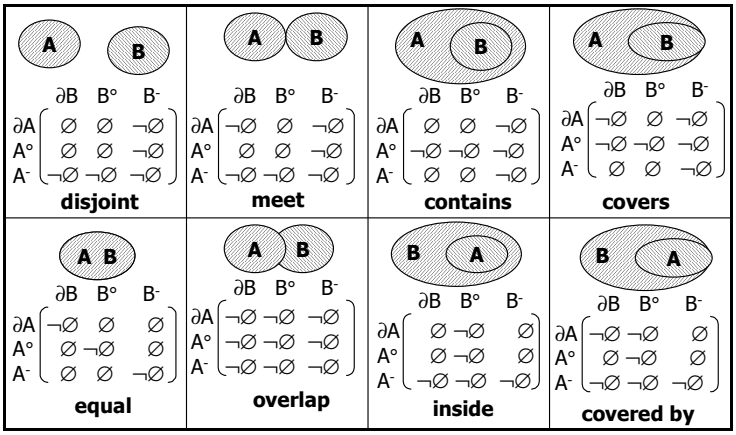


Figura 3.4 - Matriz de 9-Interseções para relações entre duas regiões.
Fonte: (Egenhofer and Herring, 1991).

Nos modelos citados acima, os resultados das intersecções são avaliados considerando os valores vazio ou não-vazio. Há várias situações em que é necessário considerar as dimensões das interseções não vazias. Por exemplo, certo estado *X* só considera um outro estado *Y* como vizinho se eles têm pelo menos uma aresta em comum. Neste caso, para encontrar os vizinhos do estado *X*, não basta saber quais estados "tocam" ou são "adjacentes" a ele, mas sim se o resultado da interseção entre eles é uma aresta.

Para acomodar estas situações, novos modelos foram definidos, levando em consideração as dimensões dos resultados das interseções não vazias. Clementini et al (Clementini et al.) estenderam a abordagem da matriz de 4-interseções de forma a incluir a informação da dimensão da interseção. No espaço bidimensional, a dimensão da interseção pode ser vazia, um ponto, uma linha ou uma região. Este modelo contempla assim um conjunto de 52 relacionamentos topológicos, o que não é conveniente do ponto de vista do usuário. Para equacionar este problema, os relacionamentos topológicos foram agrupados em cinco mais gerais - *touch*, *in*, *cross*, *overlap*, *disjoint* - que são sobrecarregados, ou seja, que podem ser usados indistintamente para ponto, linha e região. Estes relacionamentos são definidos da seguinte forma:

- *Touch:* aplica-se a pares de geometrias dos tipos região/região, linha/linha, linha/região, ponto/região e ponto/linha:

  $\langle \lambda_1, touch, \lambda_2 \rangle \Leftrightarrow (\lambda_1^o \cap \lambda_2^o = \emptyset) \wedge$

  $((\partial\lambda_1 \cap \lambda_2^o \neq \emptyset) \vee (\lambda_1^o \cap \partial\lambda_2 \neq \emptyset) \vee (\partial\lambda_1 \cap \partial\lambda_2 \neq \emptyset))$
- *In:* aplica-se a pares de geometrias com qualquer combinação de tipos:

  $\langle \lambda_1, in, \lambda_2 \rangle \Leftrightarrow (\lambda_1^o \cap \lambda_2^o \neq \emptyset) \wedge (\lambda_1^o \cap \lambda_2^- = \emptyset) \wedge (\partial\lambda_1 \cap \lambda_2^- = \emptyset)$
- *Cross:* aplica-se a pares de geometrias dos tipos linha/linha e linha/região. No caso de linha/região, temos:

  $\langle L, cross, R \rangle \Leftrightarrow (L^o \cap R^o \neq \emptyset) \wedge (L^o \cap R^- \neq \emptyset)$

  No caso de linha/linha, temos:

  $\langle L_1, cross, L_2 \rangle \Leftrightarrow \dim(L_1^o \cap L_2^o) = 0$
- *Overlap:* aplica-se a pares de geometrias dos tipos região/região e linha/linha. No caso de região/região, temos:

  $\langle A_1, overlap, A_2 \rangle \Leftrightarrow (A_1^o \cap A_2^o \neq \emptyset) \wedge (A_1^o \cap A_2^- \neq \emptyset)$

  $\wedge (A_1^- \cap A_2^o \neq \emptyset)$

  No caso de linha/linha, temos:

  $\langle L_1, overlap, L_2 \rangle \Leftrightarrow (\dim(L_1^o \cap L_2^o) = 1) \wedge (L_1^o \cap L_2^- \neq \emptyset)$

  $\wedge (L_1^- \cap L_2^o \neq \emptyset)$
- *Disjoint:* aplica-se a pares de geometrias com qualquer combinação de tipos:

  $\langle \lambda_1, \text{disjoint}, \lambda_2 \rangle \Leftrightarrow (\lambda_1^o \cap \lambda_2^o = \emptyset) \wedge (\partial\lambda_1 \cap \lambda_2^o = \emptyset) \wedge$

  $(\lambda_1^o \cap \partial\lambda_2 = \emptyset) \wedge (\partial\lambda_1 \cap \partial\lambda_2 = \emptyset)$

3.5 Consultas Espaciais

Segundo Brinkhoff et al. (Brinkhoff et al.) as consultas espaciais podem ser classificadas em:

- *Seleção espacial:* dado um conjunto de objetos espaciais D e um predicado de seleção espacial ρ sobre atributos espaciais dos objetos em D, determine todos os objetos em D cujas geometrias satisfazem ρ.
- *Junção espacial:* dados dois conjuntos de dados espaciais, D e D', e um predicado de seleção espacial θ, determine todos os pares $(d,d') \in D \times D'$ cujas geometrias satisfazem θ.

Identificamos ainda os seguintes casos particulares importantes de seleção espacial:

- *Seleção por ponto:* dado um ponto P e um conjunto de objetos espaciais D, determine todos os objetos em D cujas geometrias contêm P.
- *Seleção por região:* dada uma região R e um conjunto de objetos espaciais D, determine todos os objetos em D cujas geometrias estão contidos em R.
- *Seleção por janela:* dado um retângulo R com os lados paralelos aos eixos e um conjunto de objetos espaciais D, determine todos os objetos em D cujas geometrias estão contidos em R.

Um *predicado de seleção espacial* é uma expressão booleana B(x) com uma variável livre, x, varrendo geometrias, tal que B(x) envolve apenas operações espaciais. Semelhantemente, um *predicado de junção espacial J(x,y)* é uma expressão booleana com duas variáveis livres, x e y, varrendo geometrias, tal que *J(x,y)* envolve apenas operações espaciais. Um exemplo de seleção espacial seria:
S1. *Selecione as regiões da França adjacentes à região de Midi-Pirenées.*

A Figura 3.5 ilustra o resultado desta consulta, onde a região de Midi-Pirenées aparece em cinza escuro, e as regiões adjacente, em cinza claro.

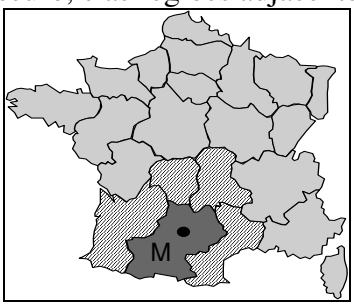


Figura 3.5 – Operação de seleção espacial.

Exemplos de junção espacial seriam:
J1. *Para cada estrada da Amazônia, selecione as reservas indígenas a menos de 5 km de uma estrada.*
J2. *Para as cidades do sertão cearense, selecione quais estão a menos de 10 km de algum açude com capacidade de mais de 50.000 m^3 de água.*

3.6 Métodos de acesso

Uma forma comum de indexarmos um conjunto de dados é através do uso de árvores de pesquisa. As mais simples e conhecidas são as árvores binárias, como: AVL, Red Black e Splay Tree (Cormen, 1990), que são árvores balanceadas, ou seja, todos os caminhos desde a raiz até as folhas possuem o mesmo comprimento. Essas estruturas permitem que algumas operações, como a localização de um elemento, sejam executadas em tempo logarítmico – *O(log n)*. A Figura 3.6 ilustra a representação de uma árvore binária balanceada, onde as cidades estão organizadas segundo a ordem alfabética de seus nomes. Esse tipo de estrutura é empregado para uso em memória principal.

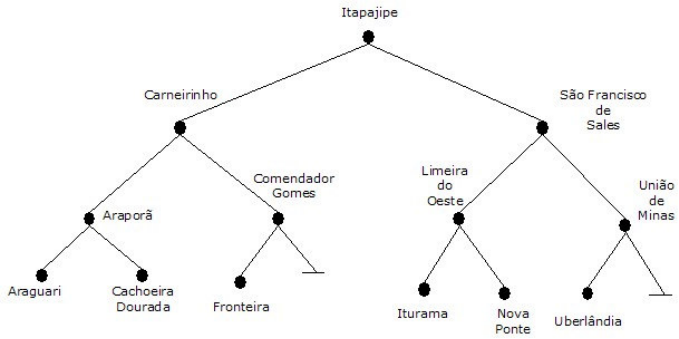


Figura 3.6 – Árvore binária balanceada.

No caso de grandes bancos de dados onde a memória principal pode não ser suficiente para armazenar todos os nós da árvore, é comum armazená-la em disco.

Nesse caso utilizamos uma representação que procura minimizar o acesso a disco. A forma mais comum, e mais largamente empregada pelos sistemas comerciais atuais, é a representação do índice através de uma árvore B^+ (Comer, 1979). Conforme podemos observar na Figura 3.7, essa estrutura tenta minimizar o número de acessos a disco durante o processo de pesquisa agrupando várias chaves em um único nó, denominado de página.

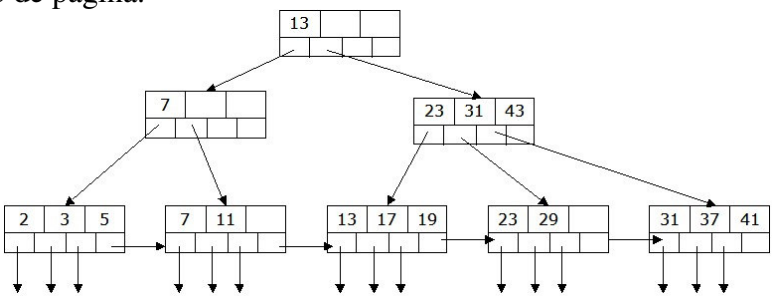


Figura 3.7 – Índice unidimensional B-Tree. Fonte: adaptada de (Garcia-Molina, 2001).

Em comum, tanto a árvore binária quanto a B^+, são estruturas unidimensionais, isto é, elas pressupõem que a chave de pesquisa seja formada por apenas um atributo ou pela concatenação de vários atributos. Em sistemas que trabalham com informações multidimensionais (mais de um atributo), como os sistemas de bancos de dados espaciais, estas estruturas não são diretamente aplicáveis.

Para esses sistemas existe uma classe de métodos conhecidos como *métodos de acesso multidimensionais*. No caso dos bancos de dados espaciais, estes métodos estão ligados ao processamento de consultas típicas como: consulta por apontamento (encontrar os objetos que contém um dado ponto), consultas por região (encontrar os objetos dentro de uma janela ou retângulo) e consultas com predicados topológicos (encontrar os objetos vizinhos de um determinado objeto).

Uma idéia fundamental dos índices espaciais é o uso de aproximações, isto é, a estrutura do índice trabalha com representações mais simples dos objetos, como o menor retângulo envolvente (REM) do objeto. Dessa forma, a maneira mais comum de processar as consultas é dividir o processamento em duas etapas: uma de filtragem e outra de refinamento (Figura 3.8). Na etapa de filtragem são usados métodos de acesso espaciais. O principal objetivo do uso destes métodos é o de reduzir e rapidamente selecionar os possíveis candidatos que satisfaçam a consulta. A redução do espaço de busca é muito importante, pois a etapa seguinte, a de refinamento, envolve a aplicação de algoritmos geométricos computacionalmente complexos e custosos e que são aplicados à geometria exata dos candidatos selecionados na etapa anterior.

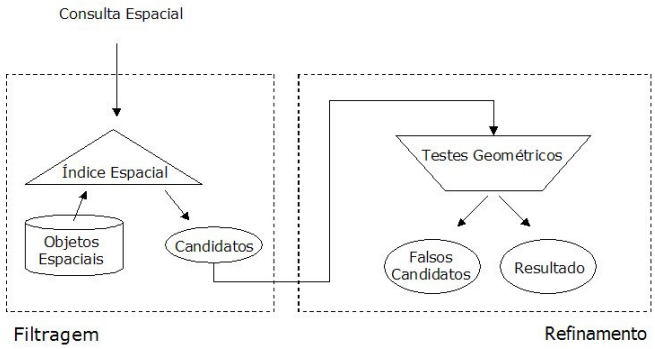


Figura 3.8 – Processamento de consultas espaciais. Fonte: adaptada de (Gaede and Günther, 1998)

Entre os principais métodos de acesso espaciais podemos citar:

- **R-Tree:** este método é uma extensão da B-Tree para o espaço multidimensional, onde os objetos são aproximados através de seus retângulos envolventes mínimos (MBR) (Guttman, 1984). Conforme pode ser visto na Figura 3.9 os nós internos contém entradas da forma <MBR, ptr-nó>, onde MBR é o retângulo que envolve todos os retângulos do nó filho (apontado por ptr-nó). As entradas dos nós folha contêm o MBR do objeto e um ponteiro para ele. A busca nessa árvore difere da B-Tree pela possibilidade de ser necessário pesquisar pelo dado desejado em mais de uma sub-árvore, caso o retângulo de pesquisa sobreponha mais de um retângulo em cada nível. Ciferri e Salgado (Ciferri) apresentam um estudo comparativo de algoritmos dessa família que, geralmente, se diferenciam na estratégia adotada para manter o balanceamento da árvore no caso de divisão de um nó.

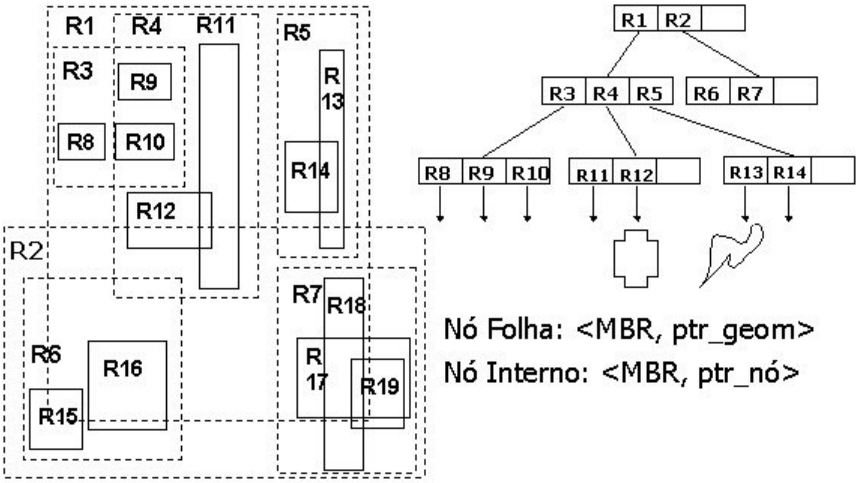


Figura 3.9 – Esquema da R-Tree.

- **Fixed Grid:** neste método de indexação, o espaço é particionado em uma grade retangular, onde cada célula é associada a uma página do disco. Essa associação é feita através de uma matriz bidimensional, conhecida como diretório, onde os elementos possuem o endereço de uma página. A Figura 3.10 ilustra a representação deste tipo de índice utilizada para o caso de pontos. Para objetos mais complexos, tais como polígonos, é utilizada a aproximação pelo MBR, sendo o objeto associado às páginas das células com as quais o seu MBR intercepta. A resolução da grade é um ponto de grande importância para este tipo de indexação.

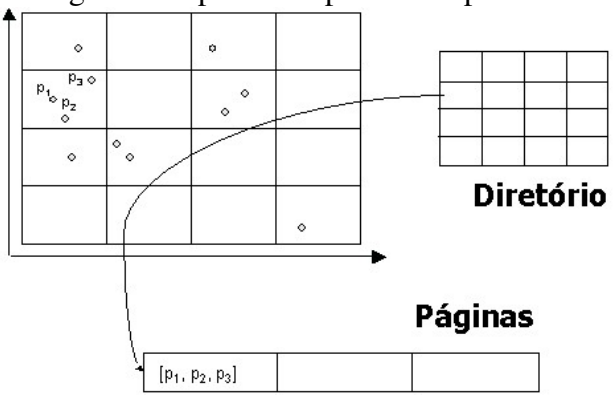


Figura 3.10 – Esquema do *Fixed Grid*.

- **QuadTree:** a idéia básica deste método é subdividir o espaço em quadrantes. A árvore é formada por nós que possuem quatro descendentes que representam os quatro quadrantes nos quais o original do nó foi subdividido. Quadrantes que não são mais subdivididos são armazenados em nós folha. Esse método pode ser aplicado a dados no formato matricial, pontos e objetos mais complexos como polígonos (Samet, 1984). A Figura 3.11 ilustra esse método de indexação.

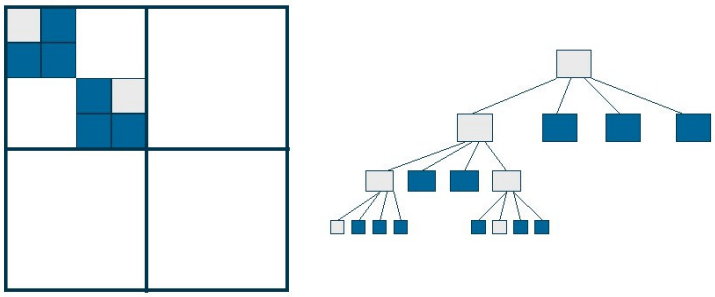


Figura 3.11 – Esquema da QuadTree.

3.7 Dados Geográficos na Web

A World Wide Web (W3C, 2005), abreviadamente Web, se tornou uma das mídias mais importantes e preferidas para disseminação de informações. Ela evoluiu de simples páginas com conteúdo estático para páginas com conteúdos dinâmicos, extraídos, principalmente, de Sistemas Gerenciadores de Bancos de Dados (SGBDs). Para atender à necessidade de construção de páginas com conteúdo dinâmico, diversas tecnologias emergiram no final da década passada e, entre elas, pode-se citar: Common Gateway Interface (CGI) (CGI), Active Server Pages (ASP) (Homer, 1999), Java Server Pages (JSP) (SUN) e Hypertext Preprocessor (PHP) (PHP).

Nesta década, a demanda pela disseminação de dados geográficos, tem motivado os principais fornecedores de Sistemas de Informação Geográficas (SIG) a investirem em ferramentas para construção de aplicações que forneçam acesso a esses tipos de dados via Web. A forma mais comum de publicação de dados geográficos nesse tipo de mídia é através de mapas em formato de imagem PNG, GIF ou JPEG, gerados dinamicamente por um servidor. Esta alternativa é a mais natural do ponto de vista dos navegadores Web, uma vez que requer apenas a apresentação de imagens. No entanto, é uma alternativa limitada, por diversos motivos. Em primeiro lugar, porque não permite que o usuário navegue pelo mapa, interagindo diretamente com os objetos apresentados, sem que haja a necessidade de re-geração da imagem. Além disso, a transmissão de imagens apresenta um compromisso entre tamanho, qualidade e resolução, por um lado, e a velocidade de transmissão, pelo outro. Por fim, existe o problema de sobrecarga no servidor, que precisa construir o mapa em formato matricial, muitas vezes a partir de dados vetoriais, e transmiti-lo para o cliente. Este último fator claramente reduz a escalabilidade dessa solução. Observe que qualquer operação simples, como zoom ou pan, exige a formação de um novo mapa-imagem (sempre com processamento no servidor) e nova transmissão. Um exemplo deste tipo de disseminação pode ser visto na Figura 3.12.

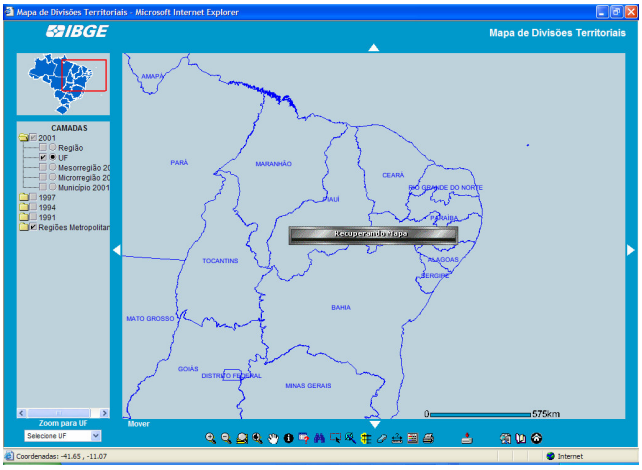


Figura 3.12 - Web site do IBGE, servidor de mapas (www.ibge.gov.br)

Uma alternativa à transmissão de imagens é a transmissão de objetos geográficos com representação vetorial. Os objetos vetoriais transmitidos são mantidos na memória da máquina cliente, para que possam ser reaproveitados no caso de operações de *zoom* ou *pan*, ganhando tempo para aumentar a interatividade. No Capítulo 4, apresentaremos padrões que endereçam o formato de troca de informações e serviços geográficos na Web.

4 Open Geospatial Consortium

O Open Geospatial Consortium (OGC, 2005a) é um consórcio formado por diversas empresas, agências governamentais e universidades, criado para promover o desenvolvimento de tecnologias que facilitem a interoperabilidade entre sistemas envolvendo informação geo-espacial (Gardels, 1996) (Percivall, 2003). Os produtos do trabalho do OGC são apresentados sob forma de especificações de interfaces e padrões de intercâmbio de dados. Dentre as especificações, iremos abordar uma voltada para o intercâmbio de dados (GML), duas voltadas para Web Services (WMS, WFS) e uma para SQL (SFS-SQL).

4.1 Geographic Markup Language (GML)

Seguindo a tendência do uso de padrões para intercâmbio de dados, o OpenGIS usa o padrão XML (eXtensible Markup Language) para definir uma forma de codificar dados geográficos. Para isso especificou a linguagem GML (Geography Markup Language) (OGC, 2004). A GML (Geography Markup Language) foi especificada para o transporte e armazenamento de informação geográfica, incluindo propriedades espaciais e não espaciais das feições geográficas.

O objetivo da GML é oferecer um conjunto de regras com as quais um usuário pode definir sua própria linguagem para descrever seus dados. Para tanto, a GML é baseada em esquemas XML (XML Schema). O esquema XML define os elementos (*tags*) usados em um documento que descreve os dados. Sua versão 3.0 inclui esquemas que contêm os modelos de geometria, feições (features) e superfícies. Os esquemas estão publicados nas especificações do OGC (OGC, 2004), sendo os principais:

- **BasicTypes**: engloba uma série de componentes simples e genéricos para representação arbitraria de atributos, nulos ou não.
- **Topology**: especifica as definições do esquema geométrico dos dados, bem como sua descrição.
- **CoordinateReference Systems**: para sistemas de referência de coordenadas.
- **Temporal Information and Dynamic Feature**: este esquema estende aos elementos características temporais dos dados geográficos e suas funções dinamicamente definidas.
- **Definitions and Dictionaries**: definições das condições de uso dentro de documentos com certas propriedades ou informações de referentes à propriedade padrão.
- **Metadata**: Este esquema é utilizado para definir as propriedades dos pacotes de dados que podem ser utilizados através de outros dados já existentes.

De posse destes esquemas, um usuário pode definir o seu próprio esquema para sua aplicação. Mas há algumas exigências a seguir para obter conformidade:

- Desenvolvedores de esquemas de aplicação devem assegurar que seus tipos são subtipos dos correspondentes tipos da GML: gml:AbstractFeatureType ou gml:AbstractFeatureCollectionType para feições e gml:AbstractGeometryType ou gml:GeometryCollectionType para a geometria.

- Um esquema de aplicação não pode mudar o nome, definição ou tipo de dado dos elementos obrigatórios da GML.
- Definições de tipos abstratos podem ser livremente estendidas ou restritas;
- Esquema de aplicação deve estar disponível a qualquer um que receba o dado estruturado por aquele esquema.
- Os esquemas relevantes devem especificar um "*namespace*" que não deve ser http://www.opengis.net/gml.

Desta forma um desenvolvedor de esquemas pode criar seus próprios tipos e *tags* e uma aplicação GML poderá fazer uso dos dados. Por exemplo, consideramos dois arquivos, um para o esquema (exemplo.xsd) e outro com os dados (exemplo.xml).

```
...
1.<complexType name="hidrografia">
2.     <complexContent>
3.          <extension base="gml:AbstractFeatureType">
4.                  <sequence>
5.                          <element ref="gml:centerLineOf"/>
6.                  </sequence>
7.          </extension>
8.     </complexContent>
9.</complexType>
...
```

Figura 4.1 – Trecho de um esquema de aplicação.

A Figura 4.1 é um fragmento de um arquivo exemplo.xsd que define um esquema de aplicação, mostrando a criação de um tipo, no caso hidrografia. Seguindo as regras, a linha 3 faz com que hidrografia seja subtipo de `gml:AbstractFeatureType`. Este tipo pode ser usado na criação de uma *tag*, por exemplo:

```
...
<element name="rio" type="ex:hidrografia" substitutionGroup="gml:_Feature"/>
...
```

Figura 4.2 – Criação de uma *tag*.

O fragmento mostrado na **Figura 4.2** é parte do mesmo esquema e define a criação de uma *tag* `<rio>` do tipo hidrografia que pode ser usada para descrever um determinado rio no arquivo exemplo.xml. Também é definido que o elemento tem o atributo `substitutionGroup` igual a `gml:_Feature`, o que garante a uma aplicação a leitura dos dados. O "ex" em `ex:hidrografia` indica que o nome hidrografia é do domínio `ex`, declarado no início do arquivo:

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!-- File: city.xsd -->
3. <schema targetNamespace="http://www.opengis.net/examples"
4.      xmlns:ex="http://www.opengis.net/examples"
5.      xmlns:xlink="http://www.w3.org/1999/xlink"
6.      xmlns:gml="http://www.opengis.net/gml"
7.      xmlns="http://www.w3.org/2000/10/XMLSchema"
8.      elementFormDefault="qualified" version="2.03">
...
```

Figura 4.3 – Exemplo de *namespace.*

No início do arquivo foi definido o "*namespace*" `ex`, como mostra a linha 4 da **Figura 4.3**. Um fragmento do arquivo contendo os dados é ilustrado pela **Figura 4.4**:

```
...
1. <rio>
2.     <gml:description>O rio principal</gml:description>
3.     <gml:name>Rio Grande</gml:name>
4.     <gml:centerLineOf>
5.        <gml:LineString  srsName="http://www.opengis.net/gml/srs/epsg.xml#432>
6.           <gml:coord><gml:X>0</gml:X><gml:Y>50</gml:Y></gml:coord>
7.           <gml:coord><gml:X>7</gml:X><gml:Y>60</gml:Y></gml:coord>
8.           <gml:coord><gml:X>1</gml:X><gml:Y>50</gml:Y></gml:coord>
9.        </gml:LineString>
10.    </gml:centerLineOf>
11. </rio>
...
```

Figura 4.4 – Fragmento de um arquivo de dados em XML.

As *tags* `<gml:description>` (linha 2) e `<gml:name>` (linha 3) da **Figura 4.4** não foram criadas no esquema de aplicação, mas sim herdadas do tipo AbstractFeatureType, já que hidrografia é deste tipo e `<rio>` foi definido como hidrografia. Para a transferência de dados é necessária a transferência do arquivo com o esquema também, só assim uma aplicação que procura por `<_Feature>` poderá saber que `<rio>` é `<_Feature>`.

Os esquemas da GML sozinhos não são adequados para criar uma instância de documento. Estes devem ser estendidos pela criação de esquemas de aplicação para domínios específicos, seguindo as regras descritas na especificação. Isto exige um investimento na elaboração de esquemas.

A GML possui pontos, linhas, polígonos e coleções geométricas (MultiPoint, MultiPolygon) definidos por coordenadas cartesianas uni, bi ou tridimensionais associados a eventuais sistemas de referência espacial.

Uma vantagem no uso de XML é a flexibilidade oferecida para criar "tags" que expressam o significado do dado descrito, obtendo-se um documento rico semanticamente. Mas, considere a seguinte situação: dois usuários de domínios diferentes representam uma determinada entidade, pela GML, como <rio> e <curso_de_agua>. Em uma troca de dados entre os usuários, os esquemas também devem ser compartilhados, pois só assim uma aplicação poderá saber que <rio> ou <curso_de_agua> são da classe <_Feature> definida pelo esquema Feature.xsd da GML, e então processá-los adequadamente. Desta forma o problema de acesso aos dados é resolvido. Mas não há como saber que <rio> é <curso_de_agua> e vice-versa. O aspecto semântico não é considerado de forma efetiva a promover a interoperabilidade. Para amenizar este problema, pode-se acrescentar *tags* que descrevem as entidades e suas relações, ou que identifiquem sinônimos. A GML é

largamente empregada pela especificação WFS que será vista na seção de Serviços Web a seguir.

4.2 OGC Web Services (OWS)

Um Web Service (WS) é uma aplicação Web que pode realizar uma tarefa específica ou um conjunto de tarefas, cuja interface é descrita através de uma notação XML (padronizada) que fornece os detalhes necessários para interagir com o serviço. Esta tecnologia, cada vez mais empregada no desenvolvimento de aplicações Web, fornece um modelo de programação independente de linguagem e plataforma, que facilita a integração (interoperabilidade) entre sistemas através da adoção de padrões abertos como HTTP, XML, SOAP, WSDL e UDDI.

Ao contrário das aplicações clientes/servidoras tradicionais (como os sistemas Web de livrarias virtuais) um WS não fornece uma interface gráfica com o usuário. Ao invés disso, compartilha a lógica do negócio e os dados através de uma interface de programação que pode ser utilizada através da rede. Um bom exemplo de WS é o serviço de busca oferecido pelo Google (www.google.com), que fornece uma API através da qual os programadores podem (utilizando sua linguagem de programação favorita) construir aplicativos customizados que realizam remotamente consultas à base de dados do Google (para maiores detalhes acesse http://www.google.com/apis).

A entidade responsável pela regulamentação das diretrizes básicas da arquitetura de Web Services é o W3C (http://www/w3.org). Entre os padrões que desempenham papel fundamental nesta arquitetura podemos citar:

- **XML:** acrônimo de eXtensible Markup Language, trata-se de uma linguagem de marcação projetada para descrever dados. Ela é extensível, possibilitando a criação de linguagens de marcação para necessidades específicas (exemplo: GML e SVG).
- **SOAP:** acrônimo de Simple Object Access Protocol, trata-se de um protocolo de comunicação entre aplicações baseado em XML, independente de protocolo de transporte. No contexto de WS, o protocolo de transporte utilizado é o http.
- **WSDL:** acrônimo de Web Services Description Language, trata-se de uma linguagem baseada em XML para descrição dos WS.
- **UDDI:** acrônimo de Universal Description, Discovery and Integration, trata-se de um serviço de registro e descoberta de Web Services (que devem estar descritos através de WSDL), que utiliza SOAP na comunicação.

Dentro deste contexto encontram-se os serviços definidos pelo OGC, conhecidos como OGC Web Services, que definem um conjunto de interfaces padronizadas com o intuito de promover a interoperabilidade entre softwares geo-espaciais. Originalmente, estas especificações não seguem todas as recomendações do W3C para definição de WS. O principal motivo, é que o OGC foi um dos pioneiros na linha de WS e, portanto, suas especificações foram desenvolvidas anteriormente ao estabelecimento do consenso a respeito de WS. Só para exemplificar, as especificações originais não utilizavam WSDL para descrição da interface e nem SOAP para encapsular as mensagens. No caso específico das mensagens, a forma utilizada são as operações Get e Post do protocolo HTTP. No

entanto, atualmente, está sendo realizado um esforço para maior aderência aos padrões da W3C. Nesta seção iremos apresentar os dois serviços que, atualmente, estão sendo mais utilizados: WMS e WFS.

4.2.1 Web Map Service (WMS)

A especificação OpenGIS WMS (OGC, 2006) define um serviço para a produção de mapas dinâmicos na Web. Neste sentido, o mapa é uma representação visual dos dados geográficos e não os dados de fato. Os mapas produzidos são representações geradas em formatos de imagem, como PNG, GIF e JPEG, ou em formatos vetoriais, como o SVG.

Quando o cliente requisita um mapa utilizando o serviço, um conjunto de parâmetros deve ser informado ao servidor: as camadas desejadas, os estilos que devem ser aplicados sobre as camadas, a área de cobertura do mapa, a projeção ou sistema de coordenadas geográficas, o formato da imagem gerada e também o seu tamanho. As três operações definidas são:

- **GetCapabilities:** obtém os metadados do servidor, que descrevem o conteúdo e os valores dos parâmetros aceitos. A reposta do servidor a esta requisição é um documento XML formatado de acordo com o esquema *capabilities_1_3_0.xsd* disponível em *http://schemas.opengis.net/wms/1.3.0*. A Figura 4.5 ilustra o resultado de uma requisição GetCapabilities a um servidor WMS. Como pode ser observado, as operações definidas nesta especificação podem ser invocadas utilizando um navegador Web comum, bastando submeter as requisições codificadas na forma de URLs.

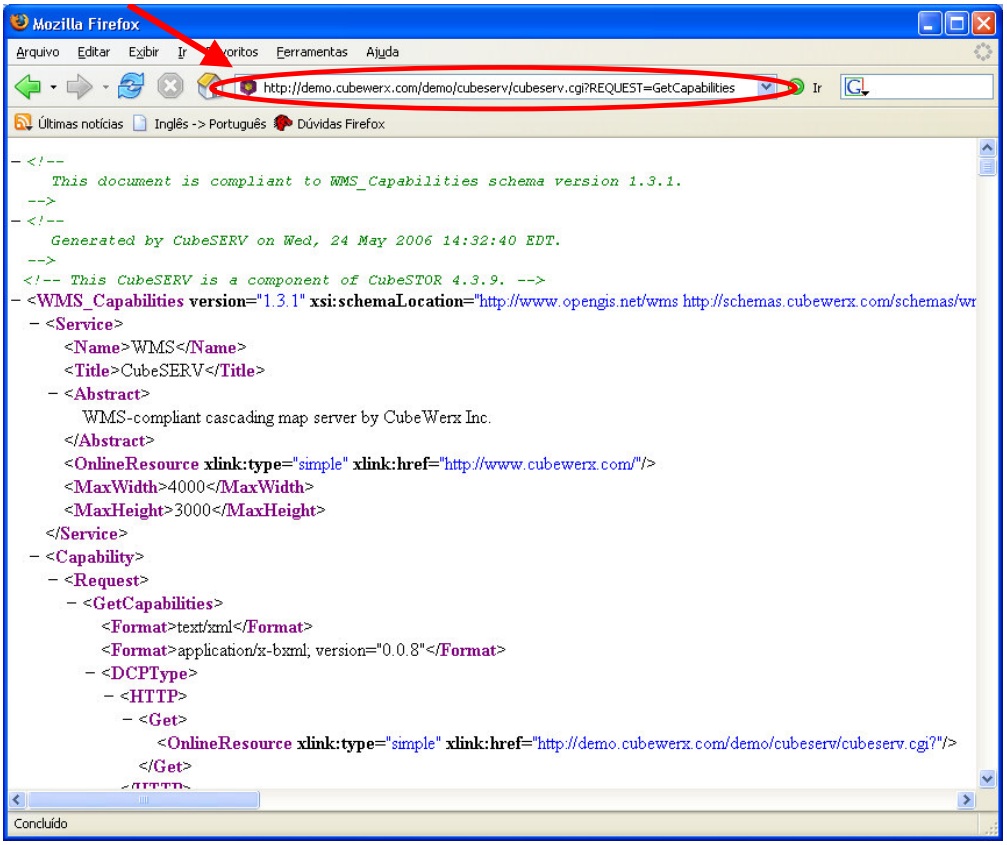


Figura 4.5 – http://demo.cubewerx.com/demo/cubeserv/ cubeserv.cgi?REQUEST=GetCapabilities

- **GetMap:** obtém a imagem do mapa que corresponde aos parâmetros informados. A Figura 4.6 ilustra o resultado da requisição de um mapa constituído de duas camadas de informação[1] a um servidor WMS.

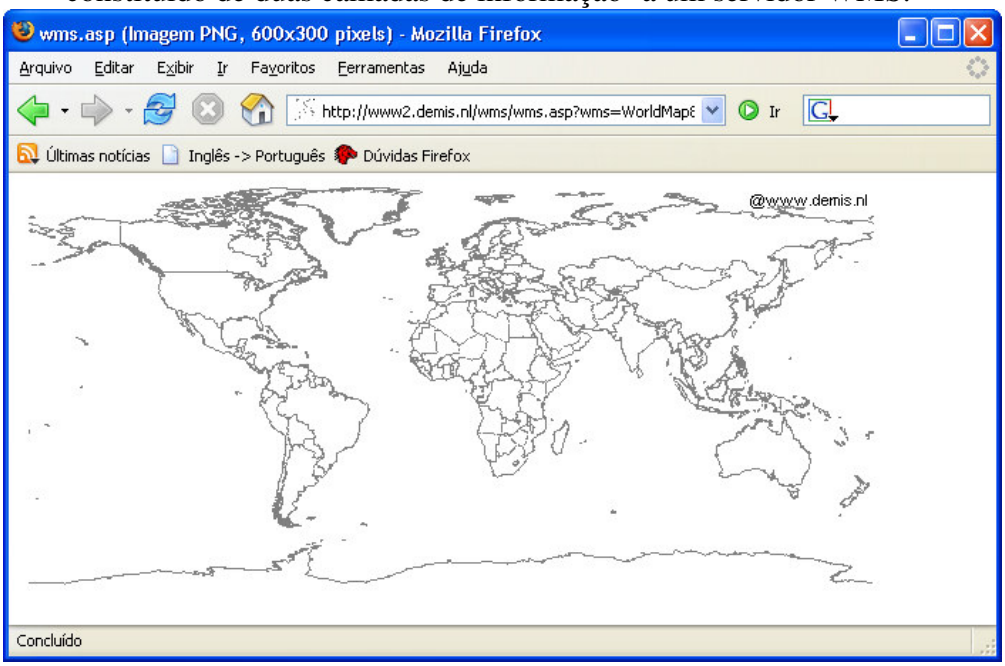


Figura 4.6 - http://www2.demis.nl/wms/wms.asp?wms=WorldMap& REQUEST=GetMap&LAYERS=Coastlines,Borders& BBOX=-184,-90,180,85&SRS=EPSG:4326&WIDTH=600& HEIGHT=300&FORMAT=PNG

- **GetFeatureInfo:** recupera informações sobre um elemento (*feature*) particular de um mapa. Esta operação somente está disponível para as camadas de informação cujo atributo queryable="1" esteja definido.
  Um exemplo de um pedido deste tipo é o URL seguinte:
  http://www.mapsherpa.com/cgi-bin/ wms_iodra?SERVICE=wms&REQUEST=getFeatureInfo& QUERY_LAYERS=locations&X=315&Y=231& LAYERS=Bathymetry,Topography,Hillshading,borders, coastlines,oceans,locations&format=image/png& bbox=21.0123,-34.4092,133.253,49.6596& STYLES=&VERSION=1.1.1&SRS=EPSG:4326& INFO_FORMAT=text/plain.

Os servidores WMS podem ser de dois tipos:

- **WMS Básico:** fornece suporte às operações GetCapabilities e GetMap, que são obrigatórias.
- **WMS Completo:** fornece suporte a operação GetFeatureInfo, que é opcional na implementação de um servidor WMS.

[1] O termo *layer* pode ser usado com sinônimo de camada de informação.

4.2.2 Web Feature Service (WFS)

As três operações mencionadas na seção anterior (WMS) fornecem uma maneira interoperável de combinar e visualizar mapas de fontes diferentes, bem como de consultar informações a respeito de um elemento (*feature*) exibido nos mapas. A especificação WFS (OGC, 2005b) define a interface de um serviço complementar: o acesso e manipulação dos dados geográficos que estão por trás dos mapas, empregando GML como formato de intercâmbio dos dados. As seguintes operações são definidas:

- **GetCapabilities:** retorna um documento com a lista de todos os tipos de objetos que podem ser servidos e as operações suportadas em cada uma delas. Exemplo de requisição:
  http://localhost:8080/wfsserver?REQUEST=GetCapabilities
- **DescribeFeatureType:** descreve a estrutura dos tipos de objeto que podem ser servidos utilizando esquemas GML/XML. Exemplo de requisição:
  http://localhost:8080/wfsserver?REQUEST=DescribeFeatureType&
  typeName=tiger:tiger_roads
- **GetFeature:** retorna instâncias dos objetos disponíveis na base de dados. O cliente pode selecionar quais objetos deseja, por critérios espaciais ou não. Exemplo de requisição:
  http://localhost:8080/wfsserver?REQUEST=GetFeature&
  typeName=topp:states&maxFeatures=3&
  PropertyName=STATE_NAME
- **GetGmlObject:** permite recuperar elementos (features) através de seu identificador, sendo esta a interface de uso do recurso de Xlinks para recuperação de informação.
- **Transaction:** utilizado para a execução de operações de modificação dos objetos (inserção, exclusão e atualização). Exemplo de requisição:
  http://localhost:8080/wfsserver?REQUEST=Transaction&
  Operation=Delete&typeName=topp:states&
  FEATUREID=states.1&RELEASEACTION=ALL&
  LockId=WfsServer_30c69bcc19ecb187
- **LockFeature:** bloqueia uma ou mais instâncias durante uma transação (opicional). Exemplo de requisição:
  http://localhost:8080/wfsserver?REQUEST=LockFeature&typeName=topp:states&FEATUREID=states.1

O serviço pode ser implementado de três formas:

- **WFS Básico:** onde apenas operações de consulta ficam disponíveis.
- **WFS com suporte a XLink:** deve suportar a operação GetGmlObject, com XLinks locais ou remotos.
- **WFS Transacional:** implementa o serviço completo, que inclui operações de inserção, exclusão, atualização, consulta de objetos (features) geográficos, ficando opcional a implementação da operação GetGmlObject.

4.3 Simple Features Specification For SQL (SFS-SQL)

A SFS-SQL (OGC, 1998) endereça o armazenamento e recuperação de feições espaciais pelos sistemas de bancos de dados. Ela define um esquema para o armazenamento de feições, a semântica dos operadores topológicos a serem usados em consultas espaciais e a interface dos demais operadores espaciais (métricos e que geram novas geometrias).

Para o armazenamento dos dados espaciais é introduzido o conceito de "tabelas com feições", onde os atributos não-espaciais são mapeados para colunas de tipos disponíveis na SQL, e os atributos espaciais, para colunas que podem seguir dois modelos de armazenamento: SQL-92 e SQL-92 com Tipos Geométricos. No primeiro modelo, a coluna geométrica de uma tabela com feição é implementada como uma chave estrangeira para uma tabela de geometrias. Esta tabela (de geometrias) possui um esquema capaz de armazenar a componente espacial, podendo ser implementada através de um esquema com campos do tipo numérico ou binário (Figura 4.7). O segundo modelo, utiliza tipos abstratos de dados específicos para as geometrias, estendendo os tipos básicos da SQL.

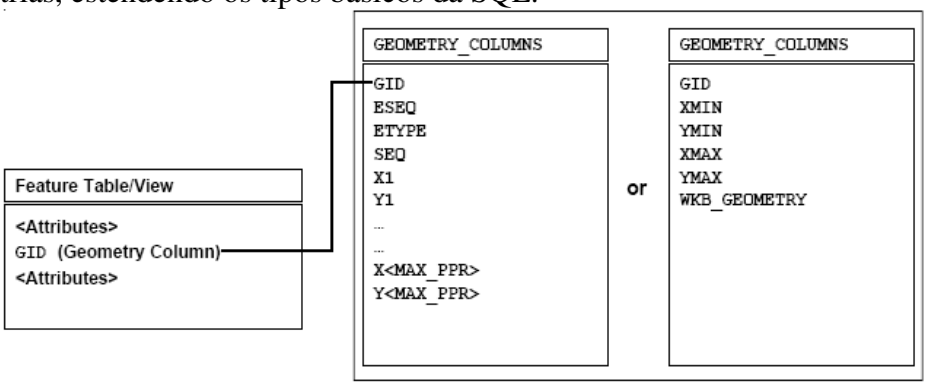


Figura 4.7 – Esquema para tabelas com feições

O modelo das geometrias, tanto da SQL-92 quanto da SQL-92 com Tipos Geométricos, segue a hierarquia mostrada na Figura 4.8. Cada uma dessas classes possui uma série de métodos entre os quais podemos citar:

- **Básicos:** retorno do retângulo envolvente, do identificador da projeção, exportação para WKT e WKB.
- **Teste de relacionamento espacial (topológicos):** toca, cruza, sobrepõe, disjunto, contém, dentro e intercepta. Estes operadores são definidos segundo o modelo formal conhecido como matriz de 9-Interseções estendida dimensionalmente (DE-9IM).
- **Suporte à análise espacial:** distância, buffer, união, interseção, diferença e diferença simétrica.

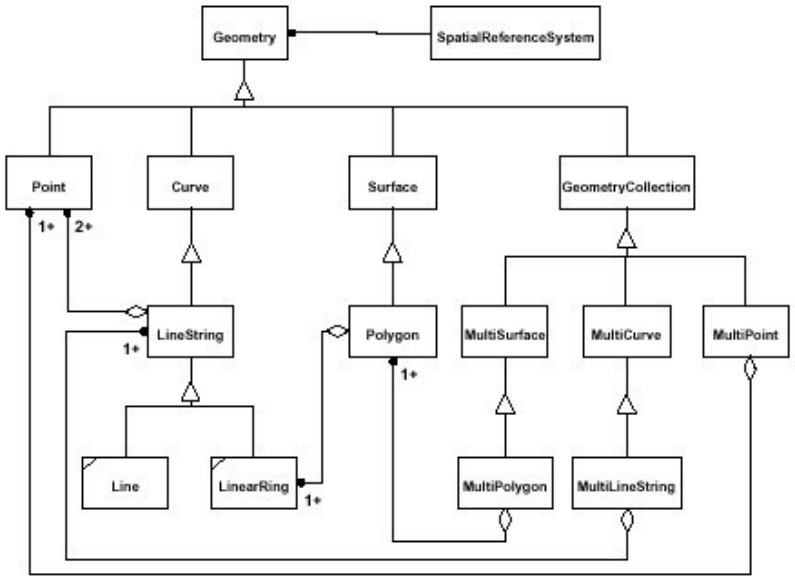


Figura 4.8 – Hierarquia de tipos de geometrias da SFS-SQL(OGIS, 1995)

Além do modelo geométrico acima, é proposto um esquema de metadados como o da **Figura 4.9**. Neste esquema, a tabela *SPATIAL_REF_SYS* armazena dados sobre cada projeção disponível e a tabela *GEOMETRY_COLUMNS* registra quais são as colunas geométricas das tabelas com feição juntamente com a projeção em que se encontram os dados da tabela.

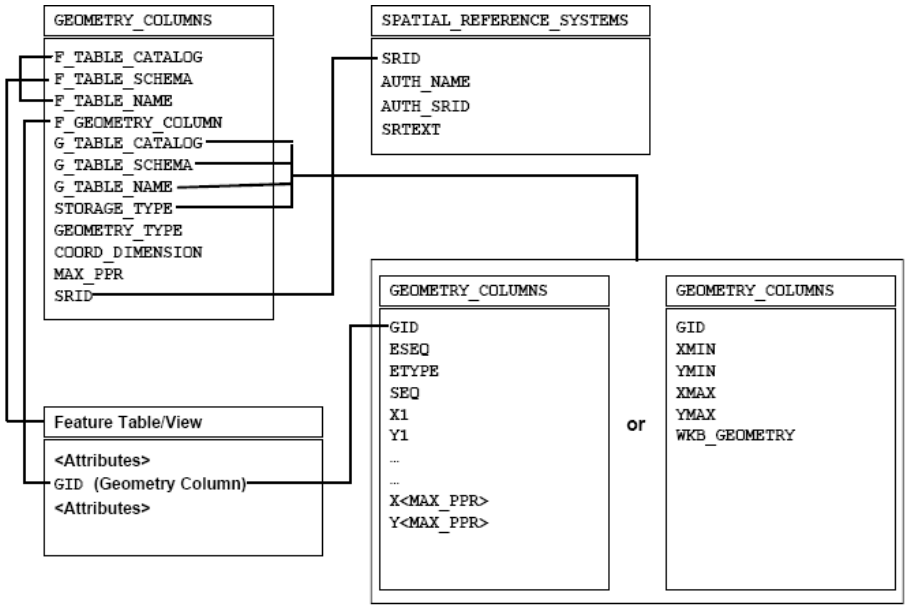


Figura 4.9 – Esquema de Metadados da SFS-SQL(OGIS, 1995)

5 Geo-Tecnologias

Nos últimos anos, além das tecnologias comerciais, vem surgindo um grande número de tecnologias baseadas em software livre para trabalhar com dados geográficos. O objetivo deste capítulo é apresentar alguns sistemas livres e comerciais, com enfoque em Bancos de Dados Geográficos, apresentando suas funcionalidades, arquiteturas, modelos e disponibilidades.

5.1 Sistemas Gerenciadores de Bancos de Dados

5.1.1 PostGIS para PostgreSQL

O PostgreSQL (Stonebraker and Rowe, 1986) é um sistema gerenciador de banco de dados objeto-relacional, gratuito e de código fonte aberto, desenvolvido a partir do projeto Postgres, iniciado em 1986, na Universidade da Califórnia em Berkeley, sob a liderança do professor Michael Stonebraker. Em 1995, quando o suporte a SQL foi incorporado, o código fonte foi disponibilizado na Web (http://www.postgresql.org). Desde então, um grupo de desenvolvedores vem mantendo e aperfeiçoando o código fonte sob o nome de PostgreSQL.

Um dos pontos fortes desse SGBD é seu potencial de extensibilidade, o que possibilitou o desenvolvimento de uma extensão geográfica mais completa, chamada PostGIS (Ramsey, 2002). Esta extensão foi desenvolvida por uma empresa canadense chamada Refractions Research Inc (http://postgis.refractions.net), que mantém a equipe de desenvolvimento. Seu código fonte é aberto (GNU GPL), seguindo a especificação SFS-SQL do OGC.

Os tipos de dados espaciais fornecidos por esta extensão podem ser vistos na Figura 5.1.

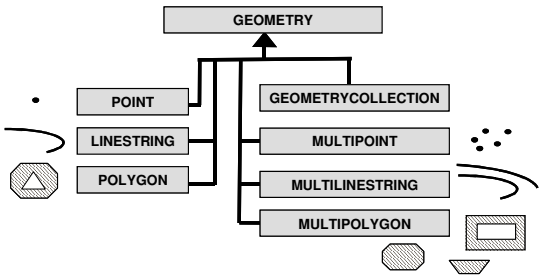


Figura 5.1 - Tipos de dados espaciais do PostGIS.

Esses tipos possuem a seguinte representação textual:

- `Point: (0 0 0)`
- `LineString: (0 0, 1 1, 2 2)`
- `Polygon: ((0 0 0, 4 0 0, 4 4 0, 0 4 0, 0 0 0), ( 1 0 0, ...), ...)`
- `MultiPoint: (0 0 0, 4 4 0)`
- `MultiLineString: ((0 0 0, 1 1 0, 2 2 0), (4 4 0, 5 5 0, 6 6 0))`

- `MultiPolygon: (((0 0 0, 4 0 0, 4 4 0, 0 4 0, 0 0 0), (...), ...), ...)`
- `GeometryCollection: (POINT(2 2 0), LINESTRING((4 4 0, 9 9 0))`

A criação de uma tabela com estes tipos de dados deve ser realizada em duas etapas como mostrado no exemplo abaixo, onde é criada uma tabela para armazenar os distritos da cidade de São Paulo:

```
CREATE TABLE distritossp
( cod           SERIAL,
  sigla            VARCHAR(10),
  denominacao      VARCHAR(50),
  PRIMARY KEY   (cod)
);
```

```
SELECT AddGeometryColumn('terralibdb', 'distritossp',
'spatial_data', -1, 'POLYGON', 2);
```

Observe que na primeira etapa, definimos os atributos básicos (alfanuméricos) e na segunda, usamos a função `AddGeometryColumn` para adicionar a coluna com o tipo espacial. Esta função implementada no PostGIS e especificada no OpenGIS, realiza todo o trabalho de preenchimento da tabela de metadado "`geometry_columns`". Os parâmetros dessa função são:

- nome do banco de dados;
- nome da tabela que irá conter a coluna espacial;
- nome da coluna espacial;
- sistema de coordenadas em que se encontram as geometrias da tabela;
- tipo da coluna espacial, que serve para criar uma restrição que verifica o tipo do objeto sendo inserido na tabela;
- dimensão em que se encontram as coordenadas dos dados.

As tabelas de metadados do PostGIS seguem as especificações da SFS-SQL e são representadas pelas seguintes tabelas:

Tabela 5-1 – Tabela de metadado do sistema de coordenadas

spatial_ref_sys		
Attribute	Type	Modifier
srid	INTEGER	PK
auth_name	VARCHAR(256)	
auth_srid	INTEGER	
srtext	VARCHAR(2048)	
proj4text	VARCHAR(2048)	

Tabela 5-2 – Tabela de metadado das tabelas com colunas espaciais

geometry_columns		
Attribute	Type	**Modifier**
f_table_catalog	VARCHAR(256)	PK
f_table_schema	VARCHAR(256)	PK
f_table_name	VARCHAR(256)	PK
f_geometry_column	VARCHAR(256)	PK
coord_dimension	INTEGER	
srid	INTEGER	FK
type	VARCHAR(30)	

Após criar as tabelas de dados, podemos inserir as informações usando o comando SQL INSERT. Para isso, podemos usar a representação textual das geometrias em conjunto com a função `GeometryFromText` que recebe a representação textual e mais o sistema de coordenadas em que se encontra a geometria:

```
INSERT INTO distritossp (denominacao, sigla, spatial_data)
VALUES('GRAJAU','MAR',
GeometryFromText('POLYGON((335589.530575 7356020.721956,
335773.784959 7355873.470174, …))', -1));
```

Podemos também recuperar as informações em cada tabela. Por exemplo, o comando abaixo seleciona a linha do bairro "Vila Mariana":

```
SELECT bairro, AsText(spatial_data) geom
FROM distritossp WHERE denominacao = 'GRAJAU';
```

```
Resultado:
    bairro    |    geom
--------------+-----------------------------------
   GRAJAU     | POLYGON((335589.530575
(1 row)       |
```

Aqui, empregamos a função `AsText` para obter a representação textual, pois a partir das versões mais recentes, o PostGIS utiliza o formato binário do OpenGIS (WKB) como o padrão nas consultas.

Outra funcionalidade disponível nesta extensão é indexação espacial. As colunas com tipos espaciais podem ser indexadas através de uma R-Tree construída no topo do mecanismo GiST. A sintaxe básica para criação de um índice é a seguinte:

```
CREATE INDEX sp_idx_name ON nome_tabela
USING GIST (coluna_geometrica GIST_GEOMETRY_OPS);
```

Para a tabela do nosso exemplo, poderíamos construir o seguinte índice espacial:

```
CREATE INDEX sp_idx_distritos ON distritossp
USING GIST (SPATIAL_DATA GIST_GEOMETRY_OPS)
```

Os índices espaciais são usados em consultas que envolvam predicados espaciais, como no caso de consultas por janela, onde um retângulo envolvente é informado e as geometrias que interagem com ele devem ser recuperadas rapidamente. O operador **&&** pode ser usado para explorar o índice espacial. Por exemplo, para consultarmos os distritos de São Paulo que interagem com o retângulo envolvente de coordenadas: 438164.882699, 7435582.150681 e 275421.967006, 7337341.000355, podemos construir a seguinte consulta:

```
SELECT * FROM distritossp
 WHERE 'BOX3D(438164.882699 7435582.150681,
             275421.967006 7337341.000355)'::box3d
        && spatial_data);
```

Com o uso do operador **&&**, apenas alguns registros precisarão ser pesquisados para responder à pergunta acima. Este operador é de fundamental importância quando construímos consultas espaciais mais elaboradas, que utilizam um outro recurso muito importante do PostGIS: *os operadores espaciais*. Outro grande destaque desta extensão é o grande número de operadores espaciais disponíveis, entre alguns deles podemos citar:

- Operadores topológicos conforme a Matriz de 9-Interseções DE:

  `equals(geometry, geometry)`

  `disjoint(geometry, geometry)`

  `intersects(geometry, geometry)`

  `touches(geometry, geometry)`

  `crosses(geometry, geometry)`

  `within(geometry, geometry)`

  `overlaps(geometry, geometry)`

  `contains(geometry, geometry)`

  `relate(geometry, geometry)`: retorna a matriz de intersecção.

- Operador de construção de mapas de distância:

  `buffer(geometry, double, [integer])`

- Operador para construção do Fecho Convexo:

  `convexhull(geometry)`

- Operadores de conjunto:

  `intersection(geometry, geometry)`

  `geomUnion(geometry, geometry)`

  `symdifference(geometry, geometry)`

  `difference(geometry, geometry)`

- Operadores Métricos:

  `distance(geometry,geometry)`

  `area(geometry)`

- Centróide de geometrias:

```
Centroid(geometry)
```

- Validação (verifica se a geometria possui auto-interseções):

```
isSimple(geometry)
```

O suporte aos operadores espaciais é fornecido através da integração do PostGIS com a biblioteca GEOS (Geometry Engine Open Source). Esta biblioteca é uma tradução da API Java JTS (Java Topology Suite) para a linguagem C++. A JTS é uma implementação de operadores espaciais que seguem as especificações da SFS-SQL. Abaixo, mostramos um exemplo de consulta espacial, envolvendo o operador *touches*, para descobrir os distritos vizinhos a Grajau:

```
SELECT d2.denominacao
FROM distritossp d1,
     distritossp d2
WHERE touches(d1.spatial_data,
                    d2.spatial_data)
AND (d2. denominacao <> 'GRAJAU')
AND (d1. denominacao = 'GRAJAU');
```

Resultado:

nomemunicp
PARELHEIROS
CIDADE DUTRA
PEDREIRA

(3 rows)

Na consulta acima, o operador *touches* retorna verdadeiro caso as geometrias de *d2* toquem a geometria de Grajau. Esse é um exemplo de junção espacial entre duas relações (no nosso caso a mesma relação foi empregada duas vezes). Todas as geometrias da relação *d1*, com exceção da geometria Grajau, foram avaliadas no teste topológico *touches***,** pois o índice espacial não foi empregado. Em tabelas com grandes números de objetos, é importante a utilização desse índice. Ele pode ser explorado empregando-se o operador && em conjunto com os predicados da consulta anterior. Nossa consulta pode ser reescrita como:

```
SELECT d2. denominacao
FROM distritossp d1, distritossp d2
WHERE touches(d1.spatial_data, d2.spatial_data)
  AND (d2.denominacao <> 'GRAJAU')
  AND (d1.spatial_data && d2.spatial_data)
  AND (d1.denominacao = 'GRAJAU');
```

5.1.2 Oracle Spatial

O Oracle Spatial (Murray, 2003) é uma extensão espacial desenvolvida sobre o modelo objeto-relacional do SGDB Oracle. Este modelo permite definir novos tipos de dados, através da linguagem de definição de dados SQL DDL, e implementar operações sobre esses novos tipos através da linguagem PL/SQL (Urman, 2002), que é uma extensão da SQL. Esta extensão é baseada nas especificações do OpenGIS e contém um conjunto de funcionalidades e procedimentos que permite armazenar, acessar, modificar e consultar dados espaciais**.**

O modelo de dados do Oracle Spatial consiste em uma estrutura hierárquica de elementos, geometrias e planos de informação (*layers*). Cada plano é formado por um conjunto de geometrias, que por sua vez são formadas por um conjunto de elementos. Cada elemento é associado a um tipo espacial primitivo, como ponto, linha ou polígono (com ou sem ilhas), os quais são mostrados na Figura 5.2.

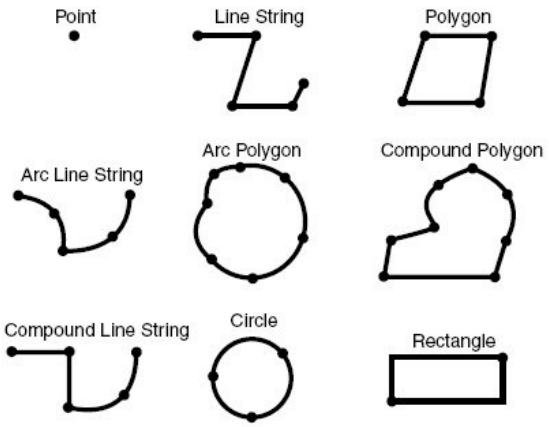


Figura 5.2 – Tipos espaciais primitivos do Oracle Spatial.

Os tipos espaciais bidimensionais são compostos por pontos formados por duas ordenadas X e Y, freqüentemente correspondente à longitude e latitude. A extensão também suporta o armazenamento e indexação de tipos tridimensionais e tetradimensionais, mas as funções e operadores só funcionam para os tipos bidimensionais.

Uma geometria pode ser formada por um único elemento ou por um conjunto homogêneo (multipontos, multilinhas ou multipolígonos) ou heterogêneo (coleção) de elementos. Um plano de informação é formado por uma coleção de geometrias que possuem um mesmo conjunto de atributos.

Baseado no modelo objeto-relacional, o Spatial define um tipo de objeto, para representar os dados vetoriais, chamado `SDO_GEOMETRY`, como mostrado a seguir.

```
CREATE TYPE sdo_geometry AS OBJECT (
SDO_GTYPE         NUMBER,
SDO_SRID          NUMBER,
SDO_POINT         SDO_POINT_TYPE,
SDO_ELEM_INFO     SDO_ELEM_INFO_ARRAY,
SDO_ORDINATES     SDO_ORDINATE_ARRAY);
```

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, DATE, dentre outros) e a geometria, como uma coluna do tipo SDO_GEOMETRY. Em uma tabela espacial cada instância do dado espacial é armazenada em uma linha e o conjunto de todas as instâncias dessa tabela forma um plano de informação.

O objeto SDO_GEOMETRY é composto pelos seguintes atributos:

- **SDO_GTYPE**: formado por quatro números, onde os dois primeiros indicam a dimensão da geometria e os outros dois o seu tipo. Os tipos podem ser: 00 (não conhecido), 01 (ponto), 02 (linha ou curva), 03 (polígono), 04 (coleção), 05 (multipontos), 06 (multilinhas) e 07 (multipolígonos).
- **SDO_SRID**: utilizado para identificar o sistema de coordenadas, ou sistema de referência espacial, associado à geometria.
- **SDO_POINT**: é definido utilizando um objeto do tipo SDO_POINT_TYPE, que contém os atributos X, Y e Z para representar as coordenadas de um ponto. Somente é preenchido se a geometria for do tipo ponto, ou seja, se os dois últimos números do SDO_GTYPE forem iguais a "01".
- **SDO_ELEM_INFO**: é um vetor de tamanho variável que armazena as características dos elementos que compõem a geometria. As coordenadas de cada elemento são armazenadas em um vetor variável chamado SDO_ORDINATES e são interpretadas através de três números armazenados no SDO_ELEM_INFO:
  - SDO_STARTING_OFFSET: indica qual a posição da primeira coordenada do elemento no SDO_ORDINATES.
  - SDO_ETYPE: indica o tipo do elemento.
  - SDO_INTERPRETATION: indica como o elemento deve ser interpretado juntamente com o SDO_ETYPE.
- **SDO_ORDINATES**: é um vetor de tamanho variável que armazena os valores das coordenadas da geometria.

Um exemplo de criação de uma tabela com o tipo SDO_GEOMETRY para armazenar os distritos da cidade de São Paulo é mostrado a seguir:

```
CREATE TABLE  DistritosSP (
cod           NUMBER(32) NOT NULL,
sigla         VARCHAR2(20),
denominacao   VARCHAR2(200),
spatial_data  MDSYS.SDO_GEOMETRY,
PRIMARY KEY (cod))
```

A tabela criada anteriormente contém colunas de tipos básicos, como NUMBER e VARCHAR2, para armazenar atributos e uma coluna do tipo

SDO_GEOMETRY para armazenar geometrias. Note que não é preciso especificar, na criação, o tipo de geometria que será armazenado.

O Oracle Spatial apresenta dois conjuntos de tabelas de metadados que são utilizadas por funcionalidades internas da extensão, como por exemplo, nas consultas espaciais:

- Tabelas de metadados sobre geometrias armazenadas, chamadas `USER_SDO_GEOM_METADATA` e `ALL_SDO_GEOM_METADATA`.
- Tabelas de metadados sobre indexação espacial, chamadas `USER_SDO_INDEX_METADATA` e `ALL_SDO_INDEX_INFO`.

As tabelas de metadados sobre geometrias armazenam para cada tabela espacial, o seu nome (TABLE_NAME); o nome da coluna de tipo geométrico (COLUMN_NAME); todas as dimensões das geometrias, cada uma com um mínimo retângulo envolvente e uma tolerância (DIMINFO) e o sistema de coordenadas (SRID). Sua estrutura é mostrada a seguir:

```
TABLE_NAME    VARCHAR2(32)
COLUMN_NAME   VARCHAR2(32)
DIMINFO       SDO_DIM_ARRAY
SRID          NUMBER
```

Após criar e inserir os dados em uma tabela espacial, o usuário deve registrar seu metadado. A seguir, mostraremos um exemplo de um comando em SQL para inserir o metadado referente à tabela espacial DistritosSP criada anteriormente.

```
INSERT INTO USER_SDO_GEOM_METADATA
VALUES ( 'DistritosSP' ,'spatial_data' ,
MDSYS.SDO_DIM_ARRAY(
MDSYS.SDO_DIM_ELEMENT('X',275.9670,429.567,0.0005),
MDSYS.SDO_DIM_ELEMENT('Y',833.0355,582.15,0.0005)), NULL)
```

Note que as geometrias armazenadas na tabela DistritosSP possuem duas dimensões X e Y, portanto existem duas entradas no vetor `SDO_DIM_ARRAY`, uma pra cada dimensão contendo seu mínimo retângulo envolvente e sua tolerância.

A seguir, mostramos um exemplo de um comando SQL para inserir dados na tabela criada:

```
INSERT INTO DistritosSP (cod, sigla, denominacao,spatial_data)
VALUES (1, 'VMR', 'VILA MARIA'
MDSYS.SDO_GEOMETRY(2003, NULL, NULL,
MDSYS.SDO_ELEM_INFO_ARRAY( 1, 1003, 1 ),
MDSYS.SDO_ORDINATE_ARRAY(6,10, 10,1, 14,10, 10,14, 6,10)))
```

O Oracle Spatial fornece dois tipos de indexação espacial, R-tree e Quadtree, podendo ser utilizados simultaneamente. Porém, a Oracle recomenda fortemente o uso de R-tree ao invés de Quadtree, por causa da performance. O usuário pode criar uma R-tree utilizando os parâmetros *default* ou especificando parâmetros como, por

exemplo, o tamanho da memória utilizada e o número de dimensões a serem indexadas. Como exemplo, mostraremos um comando em SQL para gerar um índice espacial (R-tree) com parâmetros *default* na tabela criada anteriormente:

```
CREATE INDEX DistritosSP_IDX
ON DistritosSP(SPATIAL_DATA)
INDEXTYPE IS MDSYS.SPATIAL_INDEX
```

Ao inserir, remover ou modificar geometrias de uma tabela, a performance do índice espacial gerado inicialmente pode ser degradada. Para isso, a extensão fornece um conjunto de funções para avaliar a performance dos índices, como a função `SDO_TUNE.QUALITY_DEGRADATION`, e para reconstruí-lo, usando o comando `ALTER INDEX REBUILD`.

Após a criação de índices espaciais, a extensão atualiza, automaticamente, as tabelas de metadados sobre indexação. Essas tabelas são mantidas pela extensão e não devem ser alteradas pelos usuários.

Para executar consultas e operações espaciais, o Oracle Spatial fornece um conjunto de operadores e funções que são utilizados juntamente com a linguagem SQL. Os operadores, alguns mostrados na Tabela 5.3, são usados na cláusula WHERE e utilizam indexação espacial. Portanto, só podem ser executados sobre colunas espaciais já indexadas. As funções, algumas mostradas na Tabela 5.4, são definidas como subprogramas em PL/SQL, são utilizadas na cláusula WHERE ou em subconsultas e podem ser executadas sobre colunas espaciais não indexadas.

Tabela 5.3 – Principais operadores espaciais do Oracle Spatial

Operador	*Descrição*
`SDO_FILTER`	Implementa o primeiro filtro do modelo de consulta, ou seja, verifica se os mínimos retângulos envolventes das geometrias têm alguma interação entre si.
`SDO_RELATE`	Avalia se as geometrias possuem uma determinada relação topológica.
`SDO_WITHIN_DISTANCE`	Verifica se duas geometrias estão dentro de uma determinada distância.
`SDO_NN`	Identifica os *n* vizinhos mais próximos de uma geometria

A extensão ainda fornece outros operadores, como por exemplo, `SDO_JOIN`, `SDO_TOUCH`, `SDO_INSIDE` e `SDO_ON`. As funções fornecidas pelo Oracle Spatial podem ser agrupadas em:

- Relação (verdadeiro/falso) entre duas geometrias: `RELATE` e `WITHIN_DISTANCE`.
- Validação: `VALIDATE_GEOMETRY_WITH_CONTEXT`, `VALIDATE_LAYER_WITH_CONTEXT`.
- Operações sobre uma geometria: `SDO_ARC_DENSIFY`, `SDO_AREA`, `SDO_BUFFER`, `SDO_CENTROID`, `SDO_CONVEXHULL`, `SDO_LENGTH`, `SDO_MAX_MBR_ORDINATE`, `SDO_MIN_MBR_ORDINATE`, `SDO_MBR`, `SDO_POINTONSURFACE`.

- Operações sobre duas geometrias: SDO_DISTANCE, SDO_DIFFERENCE, SDO_INTERSECTION, SDO_UNION, SDO_XOR.

Tabela 5.4 – Algumas funções espaciais do Oracle Spatial.

Função	*Descrição*
SDO_BUFFER	Gera uma nova geometria ao redor ou dentro de uma outra, considerando uma distância passada como parâmetro.
SDO_AREA SDO_ LENGTH	Calculam, respectivamente, a área e o perímetro ou comprimento de uma geometria.
SDO_DISTANCE	Calcula a distância entre duas geometrias.
SDO_INTERSECTION SDO_UNION SDO_DIFFERENCE	Geram uma nova geometria resultante da interseção, união e diferença, respectivamente, entre outras duas.

Para exemplificar o uso de operadores e funções, mostraremos a seguir um exemplo de consulta espacial, envolvendo o operador SDO_TOUCH, para descobrir os distritos vizinhos a Grajau:

```
SELECT d2.denominacao
FROM   distritossp d1,
       distritossp d2
WHERE
   SDO_TOUCH(d1.spatial_data,
             d2.spatial_data) = 'TRUE'
   AND (d2.denominacao <> 'GRAJAU')
   AND (d1.denominacao = 'GRAJAU');
```

Resultado:

nomemunicp
PARELHEIROS
CIDADE DUTRA
PEDREIRA

(3 rows)

Na consulta anterior, o operador SDO_TOUCH retorna verdadeiro caso as geometrias de d2 toquem a geometria de Grajau. Esse é um exemplo de junção espacial entre duas relações (no nosso caso a mesma relação foi empregada duas vezes), utilizando o índice espacial.

5.1.3 Outros SGBDs com extensões espaciais

Além das extensões apresentadas nesta seção, ainda temos o IBM DB2 Spatial Extender (IBM, 2002), o Informix Spatial e Geodetic Datablade (IBM, 2003) e a extensão do MySQL (MySQL, 2003).

A extensão espacial deste último SGBD encontra-se em construção. O seu projeto segue as especificações da SFS-SQL, e os únicos recursos disponibilizados até o momento são os tipos de dados espaciais semelhantes aos fornecidos pelo PostGIS (Point, LineString, Polygon, MultiLineString, MultiPolygon, MultiPoint e GeometryCollection) e o mecanismo de indexação através de R-Tree.

O quadro abaixo (Tabela 5.5) apresenta um resumo comparativo entre os SGBDs com suporte espacial:

Tabela 5.5 – Quadro Comparativo entre os SGBDs com Suporte Espacial.

Recurso	**Oracle Spatial**	**PostgreSQL com Tipos Geométricos**	**PostgreSQL com PostGIS**	**MySQL**
Tipos espaciais	SFSSQL	Tipos simples	SFSSQL	SFSSQL
Indexação espacial	R-Tree e QuadTree	R-Tree nativa ou R-Tree sobre GiST	R-Tree sobre GiST	R-Tree
Operadores topológicos	Matriz 9-Interseções	Não	Matriz 9-Interseções DE	Em desenv.
Operadores de conjuntos	Sim	Não	Sim	Em desenv.
Operador de *buffer region*	Sim	Não	Sim	Em desenv.
Transformação entre sistemas de coordenadas	Sim	Não	Sim	Não
Tabelas de metadados das colunas geométricas	Sim (diferente do OGIS)	Não	Sim (conforme OGIS)	Não

5.2 Bibliotecas para desenvolvimento de aplicativos geográficos

5.2.1 TerraLib

TerraLib (Vinhas, 2005) é uma biblioteca de classes escritas em C++ base para a construção de aplicativos geográficos de arquitetura integrada. É distribuída como software livre e com o código fonte aberto, seguindo a licença LGPL - ***G**NU **L**esser **P**ublic **L**icense*, através do endereço www.terralib.org. Essa biblioteca é multiplataforma e desenvolvida pela Divisão de Processamento de Imagens (DPI) do INPE (www.dpi.inpe.br), juntamente com a Funcate (www.funcate.org.br) e a Tecgraf da PUC-RIO (www.tecgraf.puc-rio.br).

A arquitetura da biblioteca é mostrada na Figura 5.3. Existe um módulo central, chamado *kernel*, composto de estruturas de dados espaços-temporais, suporte a projeções cartográficas, operadores espaciais e uma interface para o armazenamento e recuperação de dados espaços-temporais em SGBD, além de mecanismos de controle de visualização. Em um módulo composto de *drivers* a interface de recuperação e armazenamento é implementada. Esse módulo também contém rotinas de decodificação de dados geográficos em formatos abertos e proprietários. Funções para análise espacial como, por exemplo, processamento de

imagens, estatísticas espaciais e operações geográficas são implementadas utilizando as estruturas do *kernel* da biblioteca. Finalmente, sobre esses módulos podem ser construídas diferentes interfaces aos componentes da TerraLib em diferentes ambientes de programação (Java, COM, C++) inclusive para a implementação de serviços OpenGIS como o WMS – Web Map Server.

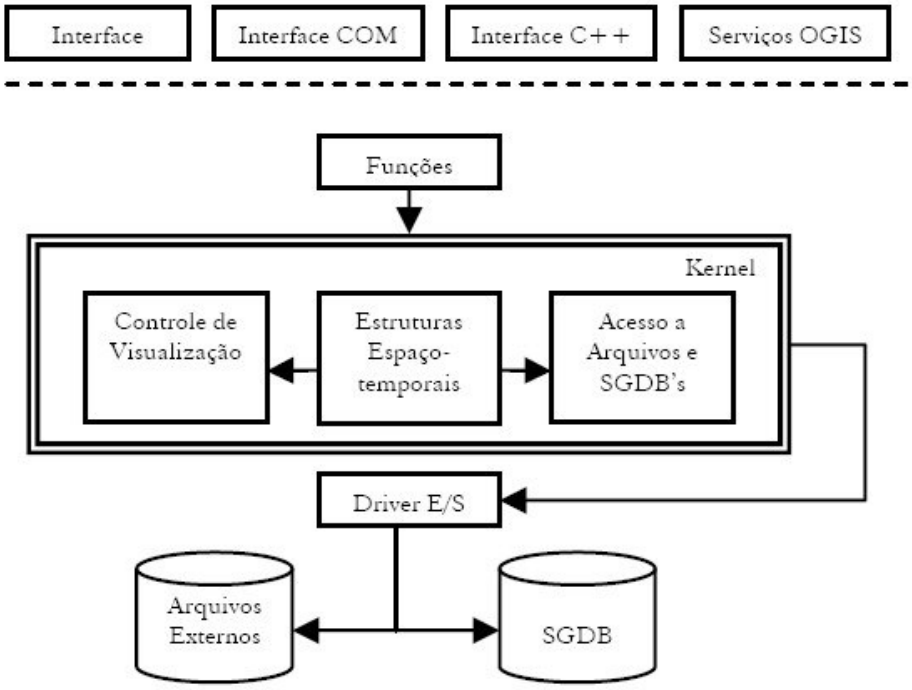


Figura 5.3 – Arquitetura da TerraLib

A TerraLib armazena os dados geográficos (vetoriais e matriciais) em SGBDs, seguindo o conceito de arquitetura integrada apresentada no Capítulo 3. Através dos *drivers*, um banco de dados TerraLib pode ser criado em diferentes tipos de SGBDs, comerciais ou livres, com ou sem extensão espacial. Os SGBDs suportados pela biblioteca são: PortgreSQL, PostGIS, MySQL, Oracle, Oracle Spatial, SQL Server e Microsoft Access.

O modelo para armazenar dados geográficos proposto pela TerraLib é formado por dois tipos de tabelas:

1. **Tabelas de metadados**: possuem nome e formato pré-definido e são usadas para guardar o modelo conceitual da TerraLib.
2. **Tabelas de dados**: são usadas para guardar os dados em si, tanto em sua componente espacial quanto descritiva.

As tabelas de metadados são criadas automaticamente quando se cria um banco de dados TerraLib. Essas tabelas armazenam informações sobre os dados geográficos existentes no banco como, por exemplo, quais os planos de informação ou *layers* existem, qual a projeção cartográfica de cada plano, quais geometrias esse plano contêm (pontos, linhas, polígonos, células, rasters, etc) e onde estão armazenadas e quais suas tabelas de atributos. Essas tabelas refletem o modelo conceitual da TerraLib, que é explicado com maiores detalhes em (Vinhas, 2005), e são mostradas na Figura 5.4.

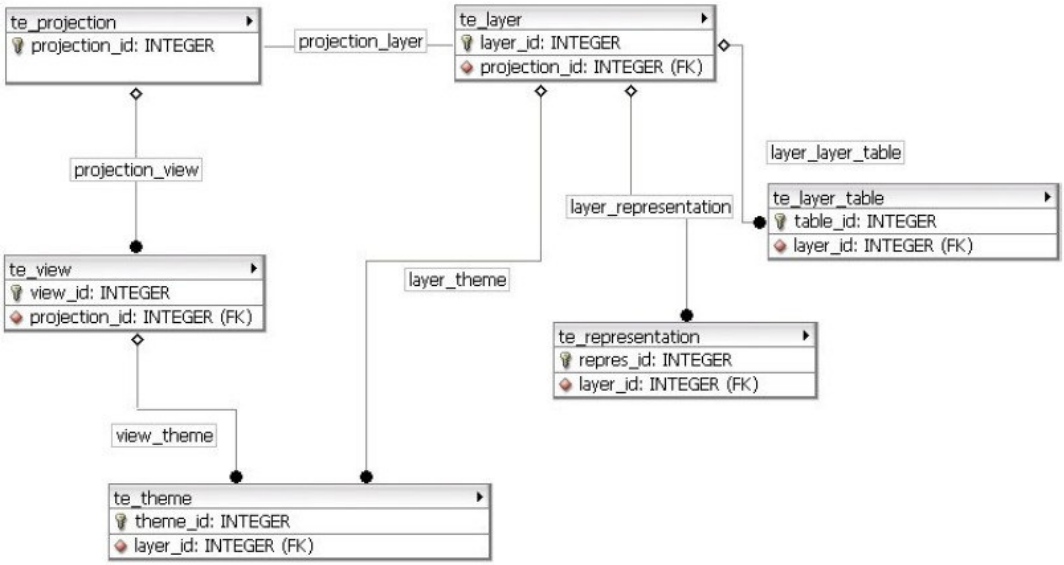


Figura 5.4 – Tabelas de metadados da TerraLib

Em um banco TerraLib, os dados geográficos são armazenados em dois tipos de tabelas: tabelas de atributos e tabelas de geometrias. As tabelas de atributos armazenam a componente alfanumérica ou descritiva e temporal dos dados geográficos, enquanto as tabelas de geometrias armazenam a componente espacial.

Em SGBDs com extensão espacial, como o Oracle Spatial e PostGIS, os dados vetoriais são armazenadas em tipos de dados espaciais. Além disso, todos os recursos oferecidos por essas extensões são explorados, como indexação espacial e operadores espaciais usados com SQL. Por outro lado, nos SGBDs que não possuem extensão espacial, os dados vetoriais são armazenados em campos do tipo binário longo (BLOB). Portanto, em um banco TerraLib, as tabelas que armazenam dados vetoriais possuem estruturas diferentes dependendo do SGBD. Essas diferenças são controladas pelos *driver*, que são responsáveis pelo mapeamento entre o modelo TerraLib e os SGBDs. A Figura 5.5 apresenta a diferença entre uma tabela para armazenar polígonos no MySQL e no Oracle Spatial. No MySQL, algumas informações como o retângulo envolvente (`lower_x`, `lower_y`, `upper_x`, `upper_y`) e o número de coordenadas (`num_coords`) de cada polígono são armazenadas explicitamente. No Oracle Spatial, essas informações fazem parte do tipo `SDO_GEOMETRY`.

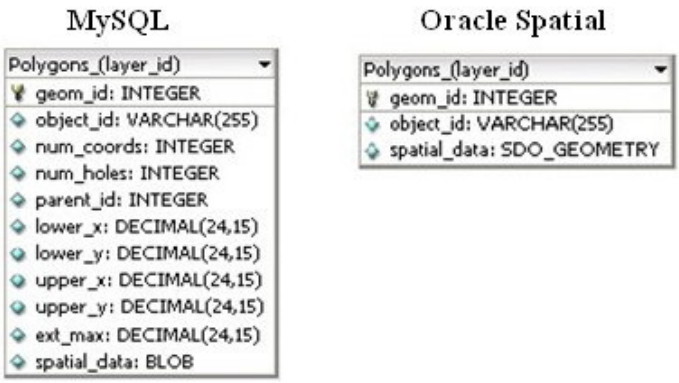


Figura 5.5 – Tabela para armazenar polígonos no MySQL e Oracle Spatial

A TerraLib propõem um modelo para armazenar e manipular dados matriciais baseado em divisão em blocos e pirâmide de multi-resolução (Souza, 2005), com mostrado na Figura 5.6. Alguns trabalhos sobre manipulação de dados matriciais (Patel et al., 1997) (Reiner et al., 2002) mostram que a estratégia mais apropriada para trabalhar com grandes bases de dados de imagens é uma combinação de divisão por blocos, ou *tiling*, e a criação de uma pirâmide de multi-resolução. O esquema de *tiling* é usado como indexação espacial, de forma que quando se deseja recuperar uma parte da imagem apenas os blocos relevantes para essa parte serão recuperados. A pirâmide de multi-resolução é útil na visualização de imagens grandes e evita acessos desnecessários, pois, nos níveis da pirâmide onde a resolução é degradada, menos blocos são recuperados. Em um banco TerraLib, os dados matriciais são armazenados nos SGBDs em tipos longos binários (BLOB).

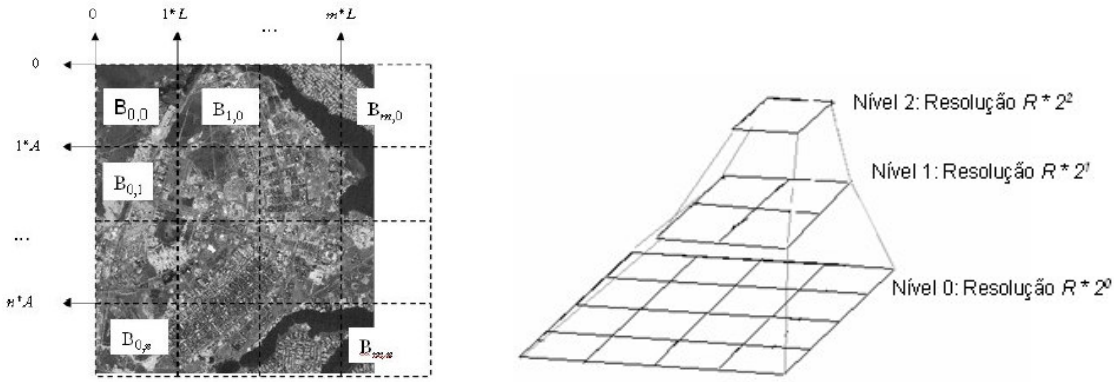


Figura 5.6 – Dados matriciais na TerraLib.

5.2.2 Outras bibliotecas para construção de aplicativos geográficos

O MapObjects é um software da ESRI (www.esri.com) formado por um conjunto de componentes para serem usados por programadores para construir aplicativos geográficos. Esse software foi construído seguindo a arquitetura ActiveX e pode ser usado em diferentes ambientes de programação, como por exemplo, Visual Basic, Visual C++, Delphi, Borland C++ Builder, Visual FoxPro e PowerBuilder. O MapObjects pode trabalhar diretamente com arquivos ou manipular bases de dados geográficos em SGBDs através do ArcSDE, mostrado na seção 5.3.3. Além disso, para construir aplicações WEB, o programador pode utilizar o MapObjects juntamente com a tecnologia ArcIMS.

A GDAL (Geospatrial Data Abstraction Library) é uma biblioteca que fornece uma API única de acesso a diversos formatos de dados espaciais, tanto matriciais quanto vetoriais. Seu código fonte é aberto, escrito em linguagem C++ e pode ser obtida a partir do seguinte endereço: http://www.gdal.org. Ela encontra-se presente na camada de acessoa dados de vários sistemas livres, como o MapServer. O seu grande destaque é a grande variedade de formatos suportados:

- Matriciais: Arc/Info ASCII Grid, Arc/Info Binary Grid (.adf), ERMapper Compressed Wavelets (.ecw), ESRI .hdr Labelled, ENVI .hdr Labelled Raster, Graphics Interchange Format (.gif), GRASS Rasters, TIFF / GeoTIFF (.tif), Hierarchical Data Format Release 4 (HDF4), Hierarchical Data Format Release 5 (HDF5), Erdas Imagine (.img), Vexcel MFF2,

Idrisi Raster, Image Display and Analysis (WinDisp), ILWIS Raster Map (.mpr,.mpl), Japanese DEM (.mem), JPEG (.jpg), JPEG2000 (.jp2, .j2k), Portable Network Graphics (.png), entre outros formatos mariciais.

- Vetoriais: Arc/Info Binary Coverage, Comma Separated Value (.csv), DODS/OPeNDAP, DWG, DXF, ESRI Personal GeoDatabase, ESRI ArcSDE, ESRI Shapefile, FMEObjects Gateway, GML, GRASS, INTERLIS, Mapinfo File, Microstation DGN, MySQL, OGDI Vectors, ODBC, Oracle Spatial, PostgreSQL, S-57 (ENC), SDTS, SQLite, UK .NTF, U.S. Census TIGER/Line, VRT - Virtual Datasource.

5.3 Aplicativos Geográficos

5.3.1 SPRING

O SPRING é um Sistema de Informação Geográfica, desenvolvido pela Divisão de Processamento de Imagens (DPI) do INPE (www.dpi.inpe.br), com funções para processamento de imagens, análise espacial, modelagem numérica de terreno, edição, importação, exportação e consultas a um banco de dados geográficos. É distribuído como software livre para os sistemas operacionais Linux e Windows, através do endereço www.dpi.inpe.br/spring.

O SPRING foi desenvolvido seguindo a arquitetura dual, mostrada no Capítulo 3. Portanto, em um banco SPRING o dado geográfico é armazenado separadamente, sua componente espacial em arquivos no formato ASCII-SPRING e sua componente descritiva em SGBDs. Os SGBDs suportados pelo SPRING são: Access, MySQL e Oracle.

No formato ASCII-SPRING, cada tipo de entidade (pontos, linhas, linhas 3D, etc) é armazenado em um arquivo diferente, refletindo o modelo conceitual do SPRING. Na Tabela 5-6 um mapeamento entre o modelo conceitual e os diferentes arquivos do formato é mostrado.

Tabela 5-6 – Mapeamento entre o modelo conceitual do SPRING e arquivos ASCII-SPRING

Modelo	Arquivos (Extensões)
Temático Pontos	POINT2D (*_P2D.spr)
Temático Linhas e Polígonos	LINES (*_L2D.spr) POINTS (*_LAB.spr) POLYGONS (*_POL.spr)
Cadastral Pontos	POINT2D (*_P2D.spr)
Cadastral linhas e Polígonos	LINES (*_L2D.spr) POINTS (*_LAB.spr) POLYGONS (*_POL.spr)
Numérico Amostras, Grade Retangular e Linhas de Quebra	SAMPLE (*_L3D.spr) GRIDREG (*_GRR.spr) LINES (*_L2D.spr)
Redes Linhas e Nós	NETWORK (*_NET.spr) NETWORK_OBJECTS (*_NETOBJ.spr)
Todos Textos	TEXT (*_TEX.spr)
Objetos Atributos	TABLE (*_TAB.spr)
Não-Espacial Atributos	TABLE (*.TAB.spr)

Para armazenar a componente descritiva no SGBD, o SPRING cria dois tipos de tabelas: tabelas de metadados e tabelas de atributos descritivos. As tabelas de metadados refletem o modelo conceitual do SPRING, armazenando informações sobre os projetos existentes (tabela `projects`), os planos de informações (tabela `infolayer`) e suas representações (tabela `repres`), as projeções cartográficas

(tabela `projecti`), os objetos (tabela `geobject`) e categorias (tabela `category`), dentre outras. Maiores informações sobre o modelo conceitual do SPRING podem ser encontradas no seu manual que pode ser adquirido no endereço www.dpi.inpe.br/spring/portugues/manuais.html. A Figura 5.7, ilustra as tabelas de metadados do SPRING. Além das tabelas de metadados, são também criadas no SGBD as tabelas de atributos e as tabelas externas, que armazenam os dados descritivos em si.

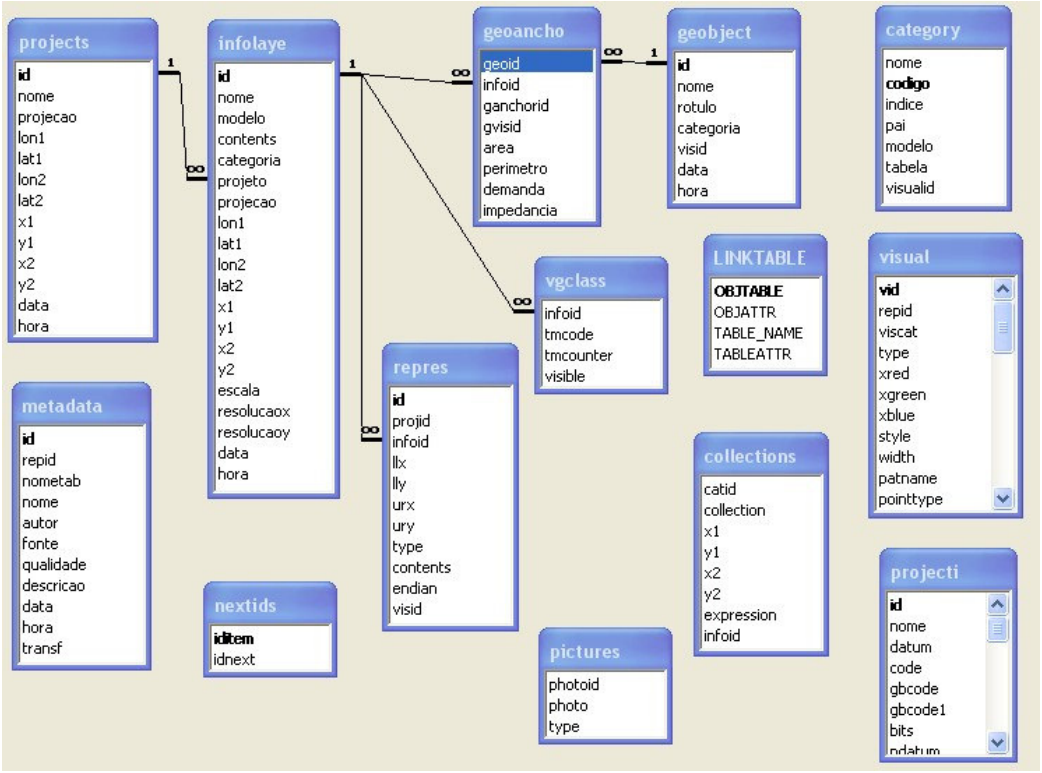


Figura 5.7 – Tabelas de Metadados do SPRING

A associação entre a componente espacial armazenada em arquivos e a componente descritiva armazenada no SGBD é feita através da tabela `geoancho` e os arquivos do tipo "Ancora", com extensões *.an1 e *an2. No SPRING, cada geo-objeto ou geo-campo recebe um identificador único chamado "geoid", que é mantido automaticamente pelo sistema. Portanto, a associação é feita através desses identificadores que são armazenados nos arquivos e no SGBD, no atributo `geoid` da tabela `geoancho`, como mostrado na Figura 5.8.

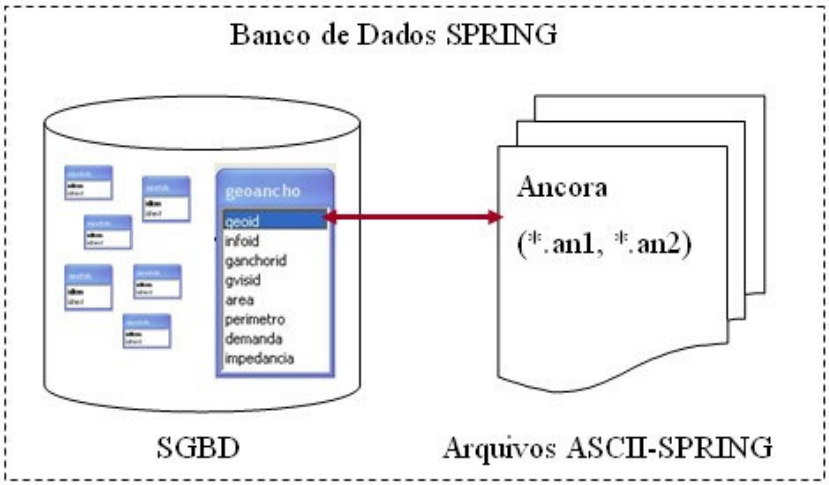


Figura 5.8 – Associação entre as componentes espaciais e descritivas no SPRING

5.3.2 TerraView

O TerraView é um aplicativo geográfico construído sobre a biblioteca TerraLib pela Divisão de Processamento Digital de Imagens (DPI) do INPE (www.dpi.inpe.br). Assim como a biblioteca, esse aplicativo é distribuído como software livre e com o código fonte aberto, seguindo a licença LGPL, através do endereço www.dpi.inpe.br/terraview. Está disponível para os sistemas operacionais Windows e Linux.

Esse sistema é um exemplo de como desenvolver aplicativos geográficos utilizando a biblioteca TerraLib. O TerraView é composto basicamente por um conjunto de interfaces gráficas que são manipuladas por usuários de SIG e que, internamente, utilizam as funcionalidades da TerraLib. Portanto, o TerraView funciona como uma interface gráfica entre os usuários de SIG e a biblioteca, como ilustrado na Figura 5.9.

Um banco de dados criado pelo TerraView possui a mesma estrutura de um banco de dados TerraLib, apresentado na seção 5.2.1, pois o armazenamento e recuperação dos dados geográficos são feitos pela biblioteca. Assim, o aplicativo pode utilizar qualquer SGBD suportado pela biblioteca.

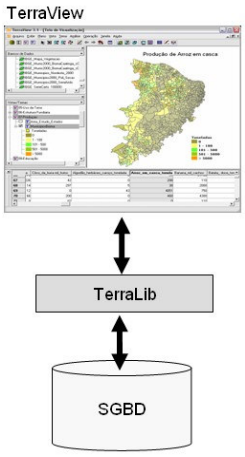


Figura 5.9 - TerraView

5.3.3 ArcGIS/ArcSDE

ArcGIS é uma coleção de softwares geográficos integrados desenvolvidos pela ESRI (www.esri.com). Esses softwares manipulam dados geográficos armazenados tanto em arquivos quanto em SGBDs comerciais, com ou sem extensão espacial. Os SGBDs suportados pelo ArcGIS são: IBM DB2 com extensão espacial, IBM Informix com extensão espacial, Microsoft SQL Server, Oracle, Oracle com extensão espacial ou Locator e Microsoft Access. Esse suporte a diferentes SGBDs é implementado pela ferramenta ArcSDE.

O ArcSDE (***S**patial **D**ata **S**erver*) funciona como um servidor de dados geográficos para os softwares do ArcGIS (ArcReader, ArcView, ArcEditos, ArcInfo, ArcIMS, ArcGIS Server, etc), fazendo a interface entre estes softwares e os SGBDs. Ele é uma ferramenta responsável pelo armazenamento, gerenciamento e recuperação dos dados dos SGBDs, como mostrado na Figura 5.10.

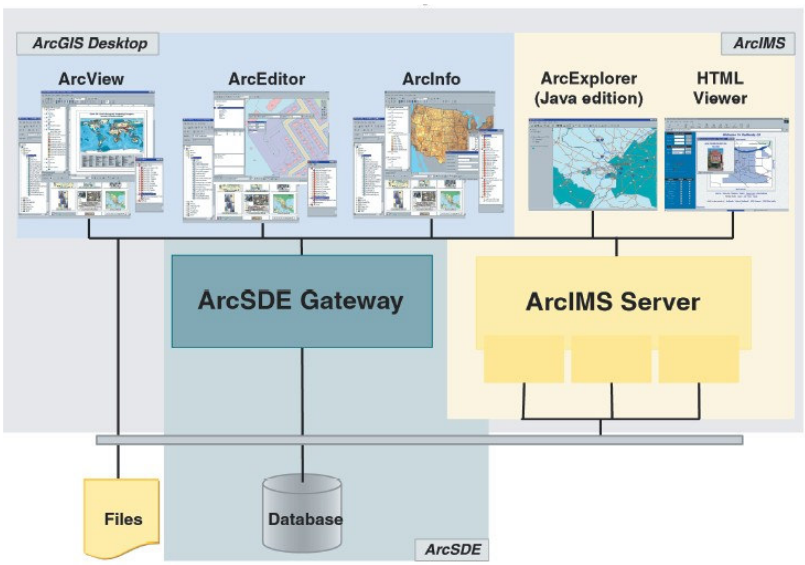


Figura 5.10 – Arquitetura do ArcGIS.
Fonte: www.esri.com

O ArcSDE define um modelo lógico para dados espaciais implementado sobre o modelo físico de cada SGBD. Nos SGBDs com extensão espacial (Oracle Spatial, IBM DB2 Spatial Extender e Informix Spatial Datablade), os dados vetoriais são armazenados em tipos de dados espaciais, como mostrado na Figura 5.11. Além disso, os recursos oferecidos por essas extensões para manipular esses tipos de dados, como indexação espacial e a SQL espacial, são explorados. Nos SGBDs que não possuem essa extensão (SQL Server e Oracle), os dados vetoriais são armazenados em tipos binários longos (BLOB, Long Raw ou Image), seguindo a especificação OpenGIS Simple Features Specification for SQL's Binary Geometry (OGC, 1998).

DBMS	Geometry Storage	DBMS Type
Oracle	ArcSDE Compressed Binary Oracle9*i* Spatial and 9*i* Locator Oracle10*g* Locator and 10*g* Spatial	Long Raw, BLOB SDO_Geometry SDO_Geometry
Microsoft SQL Server	ArcSDE Compressed Binary	Image
Informix	Informix Spatial DataBlade	ST_Geometry
IBM DB2	DB2 Spatial Extender	ST_Geometry

Figura 5.11 – Tipos de dados usados pelo ArcSDE para armazenar dados vetoriais
Fonte: www.esri.com

O dado matricial é armazenado nos SGBDs seguindo um modelo de persistência definido pelo ArcSDE (ESRI, 2005). Esse dado é dividido em blocos com tamanhos definidos pelos usuários e estruturas de indexação espacial são construídas sobre esses blocos. Os blocos são armazenados em tipos binários longos (BLOBs) em todos os SGBDs, com ou sem extensão espacial. Fazendo isso, quando um raster é consultado, somente os blocos necessários para atender a consulta são retornados, ao invés de todo o dado, aumentando a performance.

Além disso, o dado matricial é armazenado em diferentes resoluções (pirâmides). Essa característica otimiza a performance de desenho do dado reduzindo a quantidade de dados transferidos para a aplicação cliente. A Figura 5.12 ilustra o modelo de persistência para dados matriciais do ArcSDE.

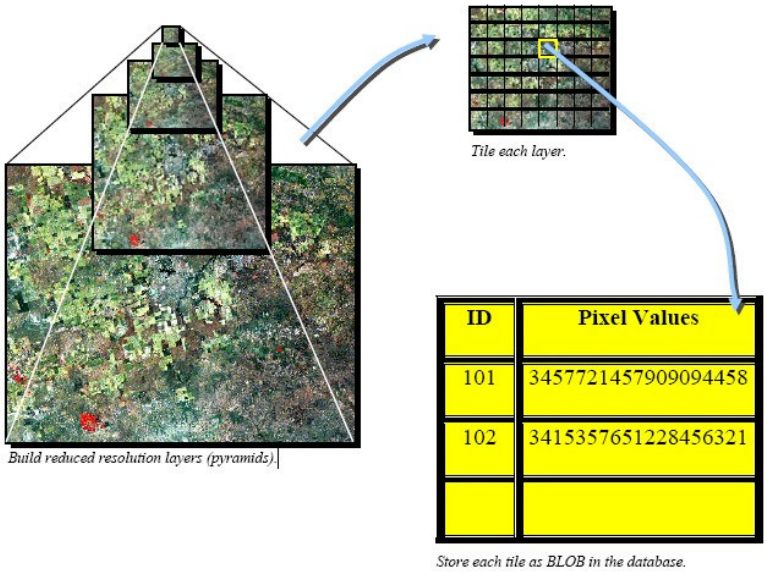


Figura 5.12 – Como um dado matricial é armazenado em um SGBD pelo ArcSDE
Fonte: (ESRI, 2005)

Para cada banco de dados, o ArcSDE cria várias tabelas de sistema, no esquema do usuário ArcSDE, para armazenar metadados sobre os dados geográficos (ESRI, 2000a). Algumas dessas tabelas são mostradas na Figura 5.13.

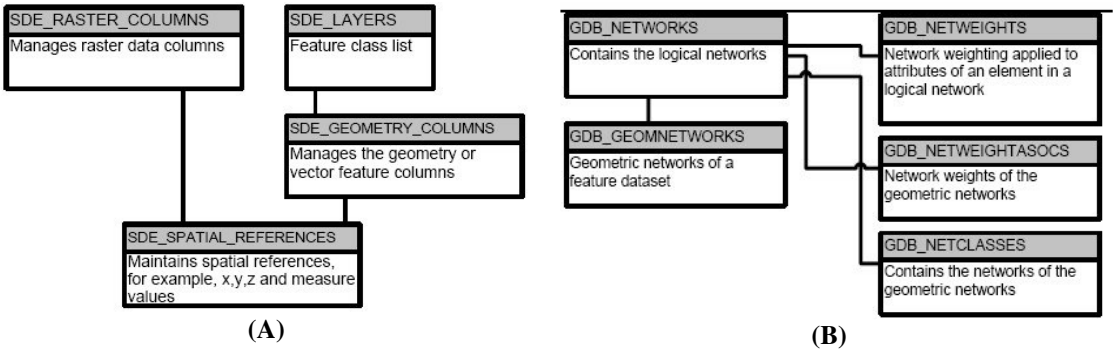


Figura 5.13 – Tabelas de sistema criadas pelo ArcSDE
Fonte: (ESRI, 2000a)

As tabelas apresentadas na Figura 5.13 (A), armazenam informações sobre os *layers*, sobre onde e como os dados vetoriais e matriciais estão armazenados e seus sistemas de coordenadas:

- SDE_LAYERS: mantém informação sobre os *layer* armazenados na base de dados, como por exemplo, o identificador interno de cada *layer* e seu mínimo retângulo envolvente.
- SDE_GEOMETRY_COLUMNS: contém informações sobre os dados vetoriais, como por exemplo, qual o nome da tabela e coluna que armazenam as geometrias, como estas estão armazenadas (BLOB ou tipos de dados espaciais), suas dimensões e qual seu sistema de coordenadas.
- SDE_RASTER_COLUMNS: contém informações sobre os dados matriciais armazenados na base, como por exemplo, o nome das tabelas e das colunas onde estão armazenados e qual seu sistema de coordenadas.
- SDE_SPATIAL_REFERENCES: contém informações sobre os sistemas de coordenadas suportados.

As tabelas apresentadas na Figura 5.13 (B) armazenam metadados sobre redes, como por exemplo, o tipo de rede, qual a tipo de indexação utilizada e quais os pesos associados à rede.

Além de dar suporte aos softwares da ESRI, o ArcSDE fornece duas ferramentas para programadores: uma API de programação em C e JAVA para o desenvolvimento de aplicações customizadas e uma API COM, chamada ArcObjects.

5.4 Tecnologias Web

5.4.1 MapServer

A principal funcionalidade deste sistema é a renderização de mapas a serem exibidos na Web. Permite tanto a criação de aplicativos Web customizados quanto de servidores de mapas. Neste último caso, o MapServer disponibiliza suporte a três serviços da OGC (como cliente ou servidor): WMS, WFS (somente para consulta) e WCS. Originalmente desenvolvido pela Universidade de Minnesota (http://mapserver.gis.umn.edu/), seu código fonte é aberto, escrito em linguagem C e multiplataforma.

Uma de suas características é a diversidade de formatos de entrada suportados, tanto vetorial quanto matricial. Nativamente, possui suporte a arquivos shape da ESRI, WFS, GML, PostGIS, ArcSDE, Oracle Spatial, JPEG, GIF, PNG,

TIFF e GeoTIFF, além de outros formatos suportados por meio da biblioteca GDAL/OGR. A saída pode ser feita em formatos PNG, JPEG, GIF, SVG, PDF, WFS, WMS, GeoTIFF (via GDAL), Flash(swf).

Entre as principais funcionalidades disponíveis neste sistema, podemos citar a geração automática de legendas, barra de escalas, possibilidade de uso de símbolos nos mapas, suporte a fontes true-type, controle de colisão de rótulos e controle de desenho dependente de escala.

Podemos construir um aplicativo, basicamente, de duas maneiras: através da interface CGI ou através da interface MapScript disponibilizada para as linguagens PHP, Perl, Python, Ruby, Tcl, Java, e C#. A primeira maneira de construção é a mais comum de encontrarmos nos aplicativos espalhados pela Web.

Tipicamente, um aplicativo construído com a interface CGI é composto por um arquivo de configuração (MapFile) e arquivos HTML (templates) que controlam a exibição dos mapas e legendas gerados pelo MapServer, bem como as informações a serem enviadas ao CGI. O arquivo de configuração é um documento texto que define, entre outras coisas, a área do mapa, as camadas de informação disponíveis e a localização desses dados, o local onde as imagens de saída podem ser armazenadas temporariamente e o formato de saída.

A **Figura 5.14** ilustra um arquivo de configuração, que contém informações de uma camada com os limites estaduais brasileiros, armazenada em um arquivo shape. Podemos visualizar os dados contidos nesta camada através de um navegador (**Figura 5.15**), bastando digitar uma URL com as informações de localização do arquivo de configuração e as camadas desejadas para que o CGI construa o mapa:

http://localhost/cgi-bin/mapserv.exe?
map=C:\mapserv_exemplo\mapfile.map&
layer=LimitesEstaduais&
mode=map

```
00   MAP
01     IMAGETYPE      PNG
02     EXTENT         -74 -33.76 -28.9 5.28
03     SIZE           400 300
04     SHAPEPATH      "data"
05     IMAGECOLOR     255 255 255
06
07     LAYER # Inicio da camada com os limites estaduais
08       NAME         LimitesEstaduais
09       DATA         uf_2001
10       STATUS       OFF
11       TYPE         POLYGON
12
13       LABELITEM    "nome"
14
15       CLASS
16         NAME       "Limites Estaduais"
17         STYLE
18           COLOR        232 232 232
19           OUTLINECOLOR 32 32 32
20         END
21         LABEL
22           COLOR 0 0 0
23           SHADOWCOLOR 255 255 255
24           SHADOWSIZE 2 2
```

```
25            TYPE TRUETYPE
26            FONT arial
27            SIZE 8
28            ANTIALIAS TRUE
29            POSITION CC
30            PARTIALS FALSE
31            MINDISTANCE 300
32            BUFFER 4
33          END
34        END
35      END
36    END
```

Figura 5.14 – Exemplo de um arquivo de configuração do MapServer (mapfile.map)

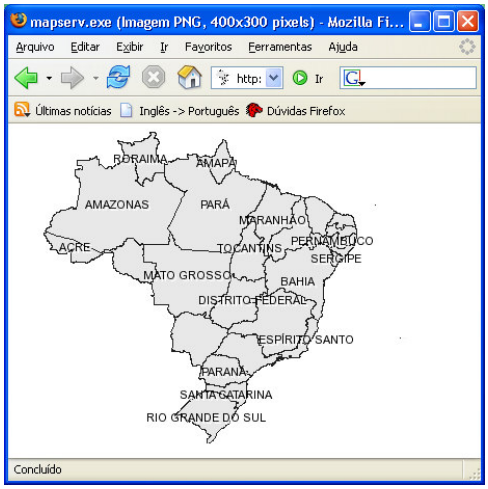


Figura 5.15 – Resultado da chamada ao MapServer

5.4.2 TerraPHP

O PHP (Hypertext Preprocessor) é uma linguagem de *script* bastante utilizada no desenvolvimento de páginas Web dinâmicas. Geralmente, os códigos PHP (chamados de *scripts*) são delimitados pelas *tags* <*?php* e *?*>, sendo embutidos nas páginas HTML. O código entre estas *tags* é processado no servidor (como por exemplo, Apache) antes da página ser enviada ao cliente (navegador ou *webbrowser*).

O código fonte de PHP é portável, aberto e escrito em linguagem C. No seu núcleo, são encontradas as funcionalidades básicas de toda linguagem de programação: estruturas de controle, laços, classes, operações aritméticas, operações relacionais e operações lógicas entre outras. As funcionalidades específicas, como manipulação de documentos (XML, PDF e etc) e comunicação com bancos de dados (PostgreSQL, Oracle e etc), são fornecidas através de extensões. A arquitetura de PHP possibilita a criação de extensões que podem ser utilizadas dentro da linguagem.

O ***TerraPHP*** é uma extensão da linguagem PHP, construída no topo da biblioteca TerraLib, para facilitar o desenvolvimento de aplicativos Web de visualização e consulta a bancos de dados geográficos. As funcionalidades desta extensão estão disponíveis na forma de uma classe interna chamada ***TerraWeb***. A forma de implementação (classe interna) utilizada permite o uso de um estilo de programação orientado a objeto, como o mostrado no fragmento de código abaixo:

```
01  <?
02    $t = TerraWeb();
```

```
03
04    if(!$t->connect("localhost",
05                    "nome_usuario",
06                    "senha",
07                    "nome_banco",
08                    3306, 1))
09    {
10      echo("ERRO!!!");
11      echo($t->errorMessage());
12      exit();
13    }
14    else
15    {
16      echo("A conexao foi um sucesso!");
17    }
18  ?>
```

A classe TerraWeb fornece métodos para estabelecimento de uma conexão a um servidor de bancos de dados, exploração do conteúdo do banco e fornece um *canvas* (abstração de uma área para desenho) que pode ser utilizado para visualizar a componente espacial dos objetos geográficos do banco de dados (Figura 5.16). O desenho sobre o *canvas* pode ser materializado através de imagens no formato PNG ou JPEG.

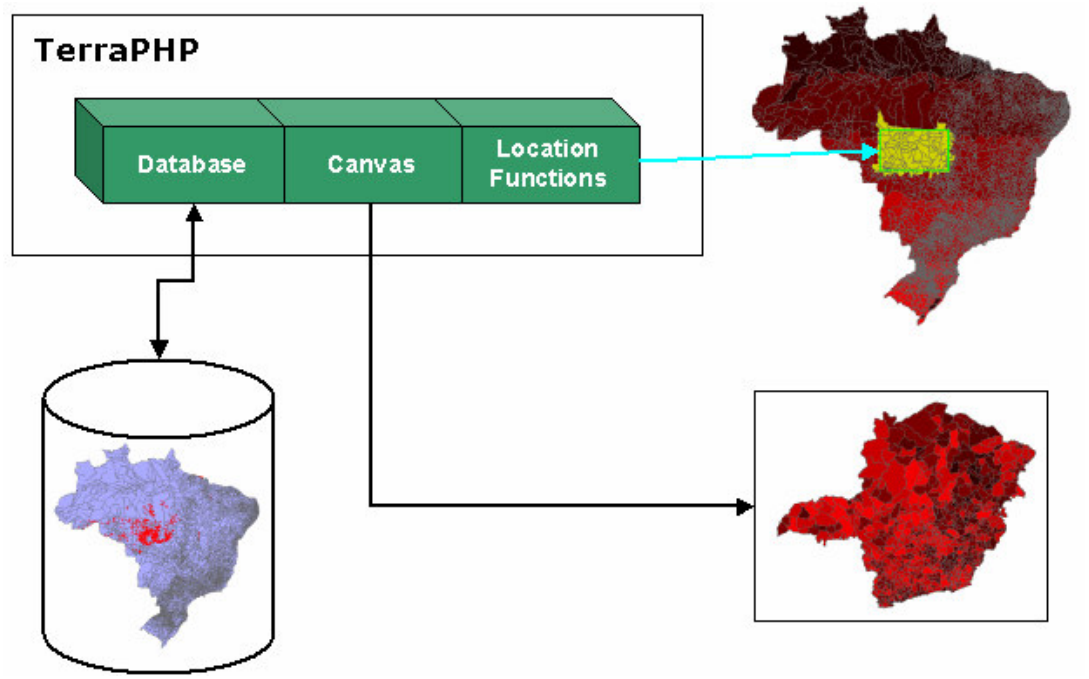


Figura 5.16 - TerraPHP

Apenas para exemplificar a simplicidade de uso desta extensão, apresentaremos um trecho de código que ao se conectar em um servidor de bancos de dados, desenha o conteúdo de uma camada de informação matricial.

Exemplo de Programa

```
01  <?
02    $t = terraweb();
03
04    if(!$t->connect("localhost", "nome_usuario", "senha", "nome_banco", 3306, 1))
05    {
06      echo("Nao foi possivel conectar-se ao banco de dados<BR>");
07      echo($t->errorMessage());
08      exit();
09    }
10
11    if($t->setCurrentView("NATIVIDADE") == false)
```

```
12      {
13        echo($t->errorMessage() . "<BR>");
14        exit();
15      }
16
17      $box = $t->getcurrentviewbox();
18
19      if($box == false)
20      {
21         echo($t->errorMessage() . "<BR>");
22         exit();
23      }
24
25      $t->setWorld($box[0], $box[1], $box[2], $box[3], 800, 600);
26
27      $t->setTheme("tema_raster", 0);
28
29       $result = $t->plotraster();
30
31       if($result == false)
32       {
33          echo($t->errorMessage() . "<BR>");
34          exit();
35       }
36
37
38      $imagemSaida = $t->getjpg();
39      header ("Content-type: image/jpg");
40      echo($imagemSaida);
41   ?>
```

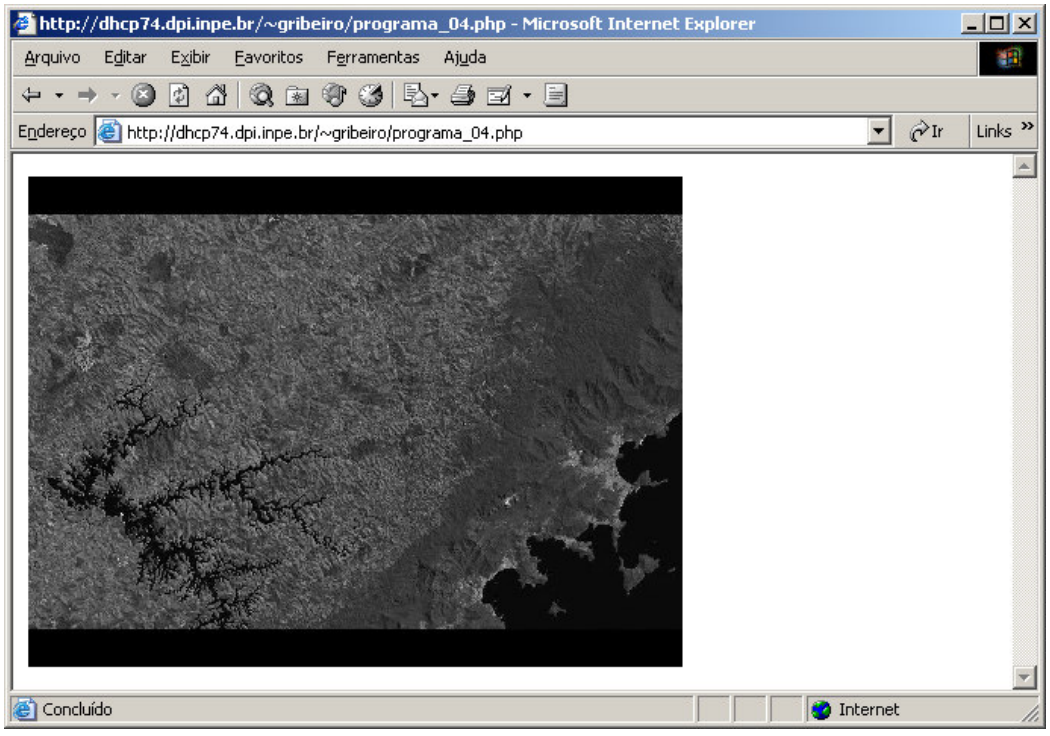


Figura 5.17 – Saída do exemplo anterior.

Referências

ABITEBOUL, S.; HULL, R. IFO: A Formal Semantic Database Model. **ACM Transactions on Database Systems**, v. 12, n.4, p. 525-565, 1987.

ABRANTES, G. C., R. . Explicit representation of data that depend on topological relationships and control over data consistency. In: Fifth European Conference and Exhibition on Geographical Information Systems. 1994. p.

BÉDARD, Y. C., C.; MAAMAR, Z.;MOULIN, B.; VALLIÈRE, D. . Adapting data models for the design of spatio-temporal databases. **Computers, Environment and Urban Systems**, v. 20, p. 19-41, 1996.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The Semantic Web. **Scientific American**, v. May, 2001.

BIRKIN, M.; CLARKE, G.; CLARKE, M. P.; WILSON, A. **Intelligent GIS : Location Decisions and Strategic Planning.** New York: John Wiley, 1996.

BONDY, J. A.; MURTY, U. S. R. **Graph Theory with Applications.** London: The Macmillan Press LTD, 1977.

BÖNISCH, S.; ASSAD, M. L.; CÂMARA, G.; MONTEIRO, A. M. Representação e Propagação de Incertezas em Dados de Solos: I - Atributos Numéricos. **Revista Brasileira de Ciência do Solo**, v. 28, n.1, p. 33-47, 2004.

BORGES, K. A. V. **Modelagem de dados geográficos - uma extensão do modelo OMT para aplicações geográficas.**Belo Horizonte - MG, 1997.Fundação João Pinheiro, 1997.

BORGES, K. A. V. D. J., C. A.; LAENDER, A. H. F., 2005. Modelagem Conceitual de Dados Geográficos. In: CASANOVA, M. A. C., G.; DAVIS JR., C. A.; VINHAS, L.; QUEIROZ, G. R., ed., **Bancos de Dados Geográficos**: Curitiba - PR, MundoGeo, p. 93-146.

BORGES, K. A. V. D. J., C. A.; LAENDER, A. H. F. . OMT-G: an object-oriented data model for geographic applications. **GeoInformatica**, v. 5, p. 221-260, 2001.

BORGES, K. A. V. L., A. H. F.; DAVIS JR., C. A.; . Spatial data integrity constraints in object oriented geographic data modeling. In: 7th International Symposium on Advantaces in Geographic Information Systems - ACMGIS. kansas City, 1999. p. 1-6.

BRINKHOFF, T.; KRIEGEL, H.; SEEGER, B. Efficient Processing of Spatial Joins Using R-Trees. In: ACM SIGMOD Conference. ACM, Washington, 1993. p. 237-246.

BURROUGH, P. **Principles of Geographical Information Systems for Land Resources Assessment.** Oxford, England: Oxford University Press, 1986.

CALDEIRA, T. **Cidade de Muros: Crime, Segregação e Cidadania em São Paulo (City of Walls: Crime, Segregation and Citizenship in Sao Paulo).** São Paulo: Edusp, 2000.

CÂMARA, G. **Modelos, Linguagens e Arquiteturas para Bancos de Dados Geográficos.**São José dos Campos, SP: Instituto Nacional de Pesquisas Espaciais (INPE), 1995.Ph.D., 1995.

CÂMARA, G.; FREITAS, U.; CASANOVA, M. Fields and Objects Algebras for GIS Operations. In: III Brazilian Symposium on Geoprocessing.**Honorary Mention on the I COMPAQ Prize on Research and Development on Computer Science in Brazil, 1995**. USP, São Paulo, 1995. p. 407-424.

CÂMARA, G.; SOUZA, R.; FREITAS, U.; GARRIDO, J. SPRING: Integrating Remote Sensing and GIS with Object-Oriented Data Modelling. **Computers and Graphics**, v. 15, n.6, p. 13-22, 1996.

CÂMARA, G. S., R.C.M.; MONTEIRO, A.M.V.; PAIVA, J.A.C; GARRIDO, J. Handling Complexity in GIS Interface Design. In: I Brazilian Workshop on Geoinformatics. SBC, Campinas, SP, 1999. p.

CASTELLS, M. **A Sociedade em Rede.** São Paulo: Paz e Terra, 1999.

CGI. **The Common Gateway Interface.**http://hoohoo.ncsa.uiuc.edu/cgi/overview.html.

CHEN, P. S. S. The Entity-Relationship Model: Towards a Unified View of Data. **ACM Transactions on Database Systems**, v. 1, n.1, p. 9-36, 1976.

CIFERRI, R. R. S., A. C. . Análise de Eficiência de Métodos de Acesso Espaciais em Termos da Distribuição Espacial dos Dados. In: Workshop Brasilerio de GeoInformática Rio de Janeiro - RJ, 2001. p. 95-101.

CLEMENTINI, E.; DI FELICE, P.; VAN OOSTEROM, P., 1993. A Small Set of Formal Topological Relationships Suitable for End-User Interaction. In: ABEL, D.; OOI, B. C., eds., **Third International Symposium on Large Spatial Databases, SSD '93**: Lecture Notes in Computer Science, v. 692: New York, NY, Springer-Verlag, p. 277-295.

COMER, D. The ubiquitous B-Tree. In: ACM Computing Survey. 1979. p. 121-137.

CORMEN, T. H. L., C. E.; RIVEST, R. L. . **Introduction to algotihms.** E.U.A.: McGraw-Hill, 1990.

CORPORATION, R. S., 1997, The Unified Modeling Language: notation guide, version 1.1

COUCLELIS, H. From Cellular Automata to Urban Models: New Principles for Model Development and Implementation. **Environment and Planning B: Planning and Design**, v. 24, p. 165-174, 1997.

DAVIS, C.; LAENDER, A. Multiple Representations in GIS: Materialization Through Map Generalization, Geometric and Spatial Analysis Operations. In: 7th ACM Symposium on Advances in Geographic Information Systems. ACM Press, N.Y., Kansas City, MO, 1999. p. 60-65.

DAVIS JR., C. A., 2000, Múltiplas representações em sistemas de informação geográficos, Belo Horizonte-MG, Departamento de Ciência da Computação da UFMG.

DAVIS JR., C. A. B., K. A. V.; LAENDER, A. H. F. . Restrições de Integridade em Bancos de Dados Geográficos. In: III Workshop Brasileiro de GeoInformática (GeoInfo 2001). Rio de Janeiro - RJ, 2001. p. 63-70.

DAVIS JR., C. A. B., K. A. V.; LAENDER, A. H. F. . Deriving spatial integrity constraints from geographic application schemas. In: Encyclopedia of Database Technologies and Applications. Idea Group Publishing, Hershey (PA), 2005. p.

DRUCK, S.; CARVALHO, M. S.; CÂMARA, G.; MONTEIRO, A. M. V. **Análise Espacial de Dados Geográficos.** Brasília: EMBRAPA (ISBN 85-7383-260-6), 2004.

EGENHOFER, M. Spatial SQL: A Query and Presentation Language. **IEEE Transactions on Knowledge and Data Engineering**, v. 6, n.1, p. 86-95, 1994.

EGENHOFER, M.; FRANZOSA, R. Point-Set Topological Spatial Relations. **International Journal of Geographical Information Systems**, v. 5, n.2, p. 161-174, 1991.

EGENHOFER, M.; HERRING, J. **Categorizing Binary Topological Relationships Between Regions, Lines, and Points in Geographic Databases.** Orono, ME: Department of Surveying Engineering, University of Maine, 1991.

ELMASRI, R. N., S. . **Fundamentals of Database Systems.** Pearson Education, 2004.

ESRI, 2000a, Managing ArcSDE Services, Redlands, C.A., ESRI.

ESRI, 2000b, Modelling Our World : The ESRI Guide to Geodatabase Design, Redlands, CA.

ESRI, 2005, Raster Data in ArcSDE, Redlands, U.S.A., ESRI.

FONSECA, F.; DAVIS, C.; CAMARA, G. Bridging Ontologies and Conceptual Schemas in Geographic Applications Development. **Geoinformatica**, v. 7, n.4, p. 355-378, 2003.

GAEDE, V.; GÜNTHER, O. Multidimensional Access Methods. **ACM Computing Surveys**, v. 30, p. 170-231, 1998.

GARCIA-MOLINA, H. U., J. D.; WIDOM, J. **Implementação de Bancos de Dados.** Rio de Janeiro-RJ: Campus, 2001.

GARDELS, K. The Open GIS Approach to Distributed Geodata and Geoprocessing. In: Third International Conference/Workshop on Integrating GIS and Environmental Modeling. Santa Fe, NM, 1996. p. 21-25.

GINZBURG, C. **Olhos de Madeira:Nove Reflexões sobre a Distância.** São Paulo: Companhia das Letras, 2001.

GODIN, L. **GIS in Telecommunications.** Redlands, CA: ESRI Press, 2001.

GOMES, J.; VELHO, L. Abstraction Paradigms for Computer Graphics. **The Visual Computer**, v. 11, n.5, p. 227-239, 1995.

GOODCHILD, M. Geographical Data Modeling. **Computers and Geosciences**, v. 18, n.4, p. 401-408, 1992.

GROSS, J.; YELLEN, J. **Graph Theory and Its Applications.** Boca Raton, FL: CRC Press, 1998.

GRUBER, T. R. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. **Int. Journal of Human-Computer Studies**, v. 43, p. 907-928, 1995.

GÜTING, R. An Introduction to Spatial Database Systems. **VLDB Journal**, v. 3, n.4, 1994.

GUTTMAN, A. R-Trees: A Dynamic Index Structure for Spatial Searching. In: Annual Meeting ACM SIGMOD. Boston, MA, 1984. p. 47-57.

HOHL, P. **GIS Data Conversion: Strategies, Techniques, and Management.** Clifton Park, NY: OnWorld Press, 1998.

HOMER, A. E. A. **Professional Active Server Pages 3.0.** Wrox, 1999.

IBM, 2002, IBM DB2 Spatial Extender: user's guide and references.

IBM, 2003, Working with the geodetic and spatial datablade modules

KÖSTERS, G. P., B. SIX, H. . GIS application development with GeoOOA. **International Journal of Geographic Information Science**, v. 11, p. 307-335, 1997.

KRAAK, M.-J.; BROWN, A., eds., 2001, Web Cartography: London, Taylor and Francis.

KUHN, W.; FRANK, A., 1991. A Formalization of Metaphors and Image-Schemas in User Interfaces. In: MARK, D.; FRANK, A., eds., **Cognitive and Linguistic Aspects of Geographic Space**: Dordrecht, Kluwer Academic Publishers, p. 419-434.

LAENDER, A. H. F. F., D. J. A semanti comparison of modelling capabilities of the ER and NIAM models In: Entity-Relationship Approach - ER'93. Springer-Verlag, 1994. p. 242-256.

LI, Z.; ZHU, Q.; GOLD, C. **Digital Terrain Modeling: Principles and Methodology.** London: Taylor and Francis, 2004.

LISBOA, F. J. R. J. M. F. D., J.; SODRÉ, V. F. ArgoCASEGEO - an open source CASE tool for Geographic Information Systems modelling using the UML-GeoFrame model. In: 7th International Conference On Infromation Systems Implementation and Modelling. Czech Republic, 2004. p.

LISBOA, J.; IOCHPE, C. Specifying Analysis Patterns For Geographic Databases on the basis of a Conceptual Framework. In: 7th International Symposium on Advances in Geographic Information Systems. ACM Press, Kansas City, 1999. p.

MACEACHREN, A. M. **How Maps Work : Representation, Visualization, and Design.** New York: Guilford Press, 2004.

MASSEY, D. S.; DENTON, N. A. **American Apartheid: Segregation and the Making of the Underclass.** Cambridge: Harvard University Press, 1993.

MATHER, P. M. **Computer Processing of Remotely-Sensed Images : An Introduction (3rd ed).** New York: John Wiley, 2004.

MONMONIER, M. **Mapping It Out : Expository Cartography for the Humanities and Social Sciences.** Chicago: University of Chicago Press, 1993.

MURRAY, C., 2003, Oracle Spatial User's Guide and Reference 10g Release 1 (10.1), Redwood City, Oracle Corporation.

MYSQL, 2003, MySQL reference manual

NOY, N. F.; SINTEK, M.; DECKER, S.; CRUBEZY, M.; FERGERSON, R. W.; MUSEN, M. A. Creating Semantic Web Contents with Protege-2000. **IEEE Intelligent Systems**, v. 16, n.2, p. 60-71, 2001.

OGC, 1998, OpenGIS Simple Features Specification for SQL, Boston, Open GIS Consortium.

OGC, 2004, Geographic Information — Geography Markup Language (GML).

OGC. **OpenGIS Consortium Inc.**, 2005a.http://www.opengeospatial.org/.

OGC, 2005b, Web Feature Service Implementation Specification.

OGC. OpenGIS Web Map Server Implementation Specification. 2006.

OGIS. **OpenGIS® simple features specification for SQL revision 1.1.** 1995.http://www.ogis.org.

OLIVEIRA, J. L. P., F.; MEDEIROS, C. M. B. . An environment for modeling and design of geographic applications. **GeoInformatica** v. 1, p. 29-58, 1997.

PARDINI, R.; SOUZA, S.; BRAGANETO, R.; METZGER, J.-P. The role of forest structure, fragment size and corridors in maintaining small mammal abundance and diversity in an Atlantic forest landscape. **Biological Conservation**, v. 124, p. 253-266, 2005.

PARENT, C. S., S.; ZIMANYI, E. . Spatio-temporal conceptual models: data structures + space + time. In: 7th International Symposium on Advances in Geographic Infromation Systems (ACMGIS). Kansas City, 1999. p.

PATEL, J.; YU, J.; KABRA, N.; TUFTE, K.; NAG, B.; BURGER, J.; HALL, N.; RAMASAMY, K.; LUEDER, R.; ELLMANN, C.; KUPSCH, J.; GUO, S.; LARSON, J.; DEWITT, D.; NAUGHTON, J. Building a Scalable Geo-Spatial DBMS: Technology, Implementation, and Evaluation. In: SIGMOD Conference. Tucson, Arizona, 1997. p.

PERCIVALL, G., 2003, OpenGIS Reference Model.

PHP. **PHP documentation.**http://www.php.net/docs.php.

PREPARATA, F.; SHAMOS, M. **Computational Geometry.** New York, NY: Springer-Verlag, 1985.

RAMSEY, P. **PostGIS Manual.** 2002.http://postgis.refractions.net.

RAVADA, S. **Topology Management in Oracle Spatial 10g.** Dagstuhl Seminar on Computational Cartography and Spatial Modelling, 2003.http://www.dagstuhl.de/03401/Materials/.

REINER, B.; HAHN, K.; HÖFLING, G.; BAUMANN, P. Hierarchical Storage Support and Management for Large-Scale Multidimensional Array Database Management Systems. In: 3th International Conference on Database and Expert Systems Applications (DEXA). Aix en Provence, France, 2002. p.

RICHARDS, J.; EGENHOFER, M. A Comparison of Two Direct-Manipulation GIS User Interfaces for Map Overlay. **Geographical Systems**, v. 2, n.4, p. 267-290, 1995.

RIGAUX, P.; SCHOLL, M.; VOISARD, A. **Spatial Databases with Application to GIS.** San Francisco: Morgan Kaufman, 2002.

RUMBAUGH, J.; BLAHA, M.; PREMERLANI, W.; EDDY, F.; LORENSEN, W. **Object-Oriented Modeling and Design.** Englewood Cliffs, NJ: Prentice-Hall, 1991.

SAMET, H., 1984. The Quadtree and Related Hierarchical Data Structures. **ACM Computing Surveys**, v. 16, p. 187-260.

SANTOS, M. **Espaço e Método.** São Paulo: Nobel, 1985.

SCHNEIDER, M. **Spatial data types for database systems.** Berlin Hidelberg: Springer-Verlag, 1997.

SEARLE, J. R. Rationality and Realism, What is at Stake? **Dædalus**, v. 122, n.4, 1993.

SEARLE, J. R. **Mind, Language and Society.** New York: Basic Books, 1998.

SHEKHAR, S. C., M.; GOYAL, B.; LIU, D.;SARKAR, S. Data Models in Geographic Infromation Systems. **Communications of the ACM**, v. 40, p. 103-111, 1997.

SMITH, B., 2003. Ontology and Information Systems. In: ZALTA, E. N., ed., **The Stanford Encyclopedia of Philosophy. The Metaphysics Research Lab, Center for the Study of Language and Information.**: Stanford, Stanford University.

SMITH, B.; MARK, D. Ontology and Geographic Kinds. In: International Symposium on Spatial Data Handling. Vancouver, Canada, 1998. p. 308-320.

SOUZA, L. V. R. C. M., 2005. Tratamento de dados matriciais na TerraLib. In: CASANOVA, M. A. C., G.; DAVIS JR., C. A.; VINHAS, L.; QUEIROZ, G. R., ed., **Bancos de Dados Geográficos**: Curitiba - PR, MundoGeo, p. 441-476.

SPOSATI, A. **Mapa de Exclusão/Inclusão Social de São Paulo.** São Paulo: EDUC, 1996.

STEVENS, S. On the Theory of Scales of Measurement. **Science Magazine**, v. 103, n.2684, p. 677-680, 1946.

STONEBRAKER, M. **Object-relational DBMSs: the next great wave** San Francisco Morgan Kaufmann, 1996.

STONEBRAKER, M.; ROWE, L. A., 1986. The Design of POSTGRES. **ACM-SIGMOD International Conference on the Management of Data**: Washington, D.C., p. 340-355.

SUN. **JavaServer Pages technology.**http://java.sun.com/products/jsp/.

TOMLIN, C. D. **Geographic Information Systems and Cartographic Modeling.** Englewood Cliffs, NJ: Prentice-Hall, 1990.

TORRES, H. Segregação residencial e políticas públicas: São Paulo na década de 1990 (Spatial segregation and public policies: São Paulo in the 1990s). **Revista Brasileira de Ciências Sociais (Brazilian Social Sciences Journal)**, v. 54, p. 41-56, 2004.

TUFTE, E. **The Visual Display of Quantitative Information.** Chesire, CT: Graphics Press, 1983.

URMAN, S. **Oracle 9i: Programação PL/SQL.** Rio de Janeiro - RJ: Editora Campos, 2002. 552 p.

VINHAS, L. F., K. R. , 2005. Descrição da TerraLib. In: CASANOVA, M. A. C., G.; DAVIS JR., C. A.; VINHAS, L.; QUEIROZ, G. R., ed., **Bancos de Dados Geográficos**: Curitiba, Editora MundoGeo, p. 397-439.

W3C. **A little history of the World Wide Web.** 2005.http://www.w3.org/History.html.

WHITE, M. J. The measurement of spatial segregation. **American Journal of Sociology**, v. 88, p. 1008-1018, 1983.

WORBOYS, M.; HEARNSHAW, H.; MAGUIRE, D. Object-Oriented Data Modeling for Spatial Databases. **International Journal of Geographical Information Systems**, v. 4, n.4, p. 369-383, 1990.