

**Análise Comparativa do
Uso dos Modelos
Relacional e Orientado a
Objetos em Sistemas de
Informações Geográficas**

**Raimundo Claudio da
Silva Vasconcelos**

7615026

Análise Comparativa do Uso dos Modelos Relacional e Orientado a Objetos em Sistemas de Informações Geográficas¹

Raimundo Claudio da Silva Vasconcelos²

Departamento de Ciência da Computação
IMECC – UNICAMP

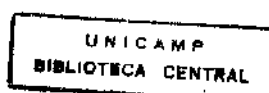
Banca Examinadora:

- Geovane Cayres Magalhães³ (Orientador)
- Edmundo R. M. Madeira³
- Cláudia M. B. Medeiros³
- Cecília Baranauskas³ (Suplente)

¹Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação da UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

²O autor é Bacharel em Ciência da Computação pela Universidade Federal do Ceará.

³Professor do Departamento de Ciência da Computação - IMECC - UNICAMP.



UNIDADE	BC
N.º CHAMADA:	UNICAMP
	V441a
Ex.	
LIBRO BC/28621	
NUM. 667126	
C	<input type="checkbox"/>
D	<input checked="" type="checkbox"/>
CO R\$ 11,00	
19-09-96	
CPD	

CM-00092062-D

**FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DO IMECC DA UNICAMP**

Vasconcelos, Raimundo Claudio da Silva

V441a Análise comparativa do uso dos modelos relacional e orientado a objetos em sistemas de informações geográficas / Raimundo Claudio da Silva Vasconcelos -- Campinas, [S.P. :s.n.], 1996.

Orientador : Geovane Cayres Magalhães

Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Ciência da Computação.

1. Sistemas de informações geográficas. 2. Banco de dados orientado a objetos. 3. Banco de dados relacionais. I. Magalhães, Geovane Cayres. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Ciência da Computação. III. Título.

Análise Comparativa do Uso dos Modelos Relacional e Orientado a Objetos em Sistemas de Informações Geográficas

Este exemplar corresponde à redação final da tese devidamente corrigida e defendida pelo Sr. Raimundo Claudio da Silva Vasconcelos e aprovada pela Comissão Julgadora.


Campinas, 2 de maio de 1.996



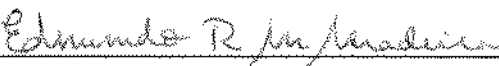
Prof. Dr. Geovane Cayres Magalhães
Orientador

Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

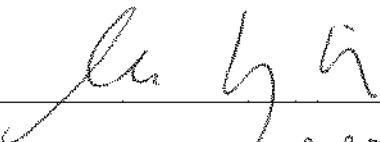
Tese de Mestrado defendida e aprovada em 2 de maio de 1996
pela Banca Examinadora composta pelos Profs. Drs.



Prof (a). Dr (a). Geovane Cayres Magalhães



Prof (a). Dr (a). EDMUNDO ROBERTO MAURO MADEIRA



Prof (a). Dr (a). CLÁUDIA BAUER MEDEIROS

Aos meus pais

Agradecimentos

Aos meus pais Elmo e Lourdes, presenças constantes em minha vida.

Aos meus irmãos Elmo Jr., Ana Lúcia, Ana Célia e Oscar pelo prazer de tê-los sempre ao meu lado. A Baía, pelo amor e dedicação sem fim.

Ao meu orientador Prof. Geovane, pela confiança, incentivo e ensinamentos proporcionados durante esta caminhada. Obrigado, principalmente, pela dupla oportunidade em ser seu orientando.

Ao Flávio Scarelli e à Ivete Lovato pela descrição da Uninet, de grande importância para esta dissertação.

Ao Tutumi e Walter, pela revisão deste trabalho.

Ao grupo de discussão de banco de dados, em especial, à Fátima e ao Juliano pelas discussões sempre frutíferas.

Aos colegas do Clube do Joio Ângelo, Guto e Carrard, companheiros de um período de grandes sonhos.

Aos meus colegas de repúblicas: Nabor, Carrilho, Fátima, Inês e Silvinha, que me ensinaram o valor de uma grande amizade.

Aos meus amigos de Campinas: Kátia, Nilza, Nair e Da. Tereza, pelo carinho e incentivo durante todo este período de convivência.

Aos meus companheiros de turma de mestrado (Fileto, Bacana, Rober, Marcellus, Luiz e Claudião) e das outras turmas (Lincoln, Mário, Marcus Vinícius, Daniel, Anderson, Humberto, Elaine, Nuccio, George, Ana, Karen, Islene, Oliva,...) pela amizade e bons momentos que passamos juntos.

Aos funcionários do DCC (Solange, Luiz, Alda, Roseli, Euclides, Roland e Furlan), pelo carinho com que sempre me atenderam.

À Heloísa do CPqD-Telebrás, pelo seu atendimento sempre em alto astral.

Aos meus amigos de Fortaleza, sempre presentes, apesar da distância.

Este trabalho foi desenvolvido parcialmente dentro do projeto CNPq PROTEM GEOTECH.

a que me foi dada de atingir esta meta.

Resumo

Costuma-se dizer que Sistemas de Informações Geográficas (SIG) são aplicações com características não convencionais e que por isso necessitam de metodologias diferentes das então utilizadas nas aplicações convencionais. Apesar disso, boa parte dos sistemas SIG existentes no mercado utiliza-se, de alguma forma, as ferramentas convencionais. O propósito dessa dissertação é verificar, a partir de um exemplo - rede computadores da Unicamp, o uso da modelagem E-R e orientada a objeto (OMT) em um SIG. As modelagens e implementações nos SGBD relacional (SQL92) e orientado a objetos (O₂) são analisados, obtendo-se os resultados desta tese.

Abstract

Geographical Information Systems (GIS) are known as non-conventional applications, and so they need different methodologies from those used for conventional applications. However, most of existent GIS in the market use conventional tools. The goal of this thesis is to verify, through an example - UNINet, the use of E-R and object oriented models (OMT) to a GIS. The modelings and implementations in relational (SQL92) and object oriented DBMS (O₂) are analyzed deriving the results.

Conteúdo

1	Introdução	1
1.1	Motivação	1
1.2	Objetivos da dissertação	2
1.3	Estrutura da dissertação	2
2	Sistema de informações geográficas - GIS/SIG	4
2.1	Definição de SIG	4
2.2	Histórico	5
2.3	Características de SIG	6
2.4	Tipos de dados	6
2.5	Componentes básicos de um SIG	13
2.6	Aplicações	14
2.7	Alguns SIG protótipos e comerciais	15
2.7.1	PROBE	15
2.7.2	ARC/INFO	16
2.7.3	TIGRIS	18
2.7.4	SIRO-DBMS	19
2.7.5	VISION*	20
2.7.6	GRASS	22
3	Modelagem de dados em SIG	23
3.1	Modelo de dados	23
3.2	Modelo E-R	24
3.2.1	Entidades, atributos e relacionamentos	25
3.2.2	Cardinalidade	26
3.2.3	Chaves	27
3.2.4	Generalização e especialização	28

3.2.5	Agregação	29
3.2.6	Diagrama entidade-relacionamento	29
3.3	Modelo Relacional	31
3.4	Transformação do modelo E-R para tabelas	32
3.5	Modelo orientado a objeto	34
3.6	OMT	35
3.6.1	Modelo de objetos	36
3.7	Modelagem de dados em SIG	39
3.8	Linguagem de consulta	40
3.9	Banco de dados para SIG	41
3.9.1	Linguagem de consulta	44
3.10	SGBD O ₂	44
3.10.1	Modelo de dados	45
3.10.2	Linguagem de consulta	46
3.10.3	O ₂ Look, O ₂ Tools	46
3.11	A utilização de orientação a objeto em SIG	47
3.12	Alguns SIG prototipais e comerciais	48
3.12.1	PROBE	48
3.12.2	ARC/INFO	49
3.12.3	TIGRIS	50
3.12.4	SIRO-DBMS	52
3.12.5	VISION*	53
4	Estudo de caso - UNINet	56
4.1	Características da UNINet	56
4.1.1	TCP/IP	61
4.1.2	Conexão de máquinas de grande porte	61
4.1.3	Componentes da UNINet	62
4.1.4	Possíveis operações entre elementos de rede	65
4.2	Redes departamentais	66
4.3	Modelagem de redes	69
4.4	Modelagem	71
4.4.1	E-R	71
4.4.2	OMT	79
4.5	Considerações sobre modelagem	80

5	Análise comparativa	84
5.1	Comparação dos modelos	84
5.2	Implementação dos modelos	85
5.2.1	Modelo Relacional	85
5.2.2	Modelo orientado a objeto	93
5.2.3	Análise das implementações	98
5.3	Consultas	99
5.3.1	Consultas descritivas	100
5.3.2	Consultas topológicas	102
5.3.3	Consultas métricas	105
5.3.4	Considerações	106
6	Conclusões	107
6.1	Considerações	107
6.2	Contribuições	108
6.3	Trabalhos futuros	109
	Bibliografia	110

Lista de Figuras

2.1	Tipos de dados.	7
2.2	Dados geográficos.	8
2.3	Ponto, linha e polígono.	10
2.4	Modelo espaguete.	11
2.5	Modelo topológico.	12
2.6	Estrutura do banco de dados.	21
3.1	Diagramas do modelo E-R.	26
3.2	Diagrama do modelo E-R.	29
3.3	Cardinalidade de relacionamentos.	30
3.4	Papéis.	30
3.5	Modelo relacional.	31
3.6	Modelo objeto.	38
3.7	Propostas para SIG.	43
3.8	Estrutura do banco de dados.	54
4.1	Backbone da UNINet.	57
4.2	Rede ANSP.	59
4.3	MMAC-3.	59
4.4	FOT-1.	60
4.5	ST-500.	60
4.6	MT-800.	61
4.7	IBM-PC.	61
4.8	FR-3000.	62
4.9	Dois caminhos possíveis entre as caixas 188 e 195.	64
4.10	Percurso redundante.	64
4.11	Rede do centro de computação.	68
4.12	Rede do setor de conectividade.	69
4.13	Modelo E-R para mapeamento urbano da UNINet.	72
4.14	Modelo E-R da rede de dutos e cabos da UNINet.	74
4.15	Modelo E-R da rede departamental da UNINet.	77

4.16	Modelo de objetos do mapeamento urbano.	80
4.17	Modelo de objetos da rede de dutos e cabos.	81
4.18	Modelo de objetos da rede departamental.	82

Capítulo 1

Introdução

O presente trabalho se propõe a verificar e analisar o uso dos modelos relacional e orientado a objetos em sistemas de informações geográficas - (SIG), através de um estudo de caso. Em especial, são feitas análises relativas ao uso de SIG para aplicações de mapeamento automático/gerência de facilidades típicos de concessionárias de serviços públicos.

1.1 Motivação

O uso de computadores em sistemas de informações veio facilitar e melhorar o armazenamento, recuperação e manuseio de grandes volumes de dados. Tal fato está presente em uma grande quantidade de atividades humanas sendo, por esse motivo, muito desenvolvido o uso de computadores em processamento de dados comerciais. Estes sistemas caracterizam-se por um conjunto homogêneo grande orientado a registro, a maioria recuperado em resposta a consultas relativamente simples, como consultas pontuais e por intervalo.

O grande desenvolvimento alcançado nos recursos computacionais permitiu o surgimento de sistemas de computação em áreas díspares das imaginadas quando do início da computação. Pesquisas têm sido desenvolvidas com o intuito de se adequar a aplicação da computação nessas diversas áreas. Dentre essas áreas existem aquelas que manipulam grandes volumes de dados e que por isso utilizam sistemas gerenciadores de banco de dados (SGBD).

Entre essas áreas destacam-se as aplicações espaciais, que vêm tendo uma demanda muito grande (geoprocessamento no gerenciamento de redes de computadores, redes de esgoto, telefonia, recursos naturais). Tais aplicações possuem características que as tornam diferentes das aplicações convencionais. As aplicações convencionais são aquelas que possuem estruturas de armazenamento tradicionais, tipos de dados alfanuméricos, regis-

tros de tamanho fixo e cadeias de caracteres. Aplicações geográficas abrangem, além dos tipos convencionais, outros como imagens, tipos de dados geométricos (pontos, linhas e polígonos), normalmente localizados no espaço bidimensional.

Devido a essas características os sistemas espaciais devem prover um suporte para esses novos tipos de dados, incluindo a necessidade de novas técnicas de armazenamento, algoritmos de busca, novas estratégias para suporte de transações longas, etc.

O que existe hoje no mercado é a utilização de Sistemas Gerenciadores de Banco de Dados convencionais, normalmente relacionais, acoplados a módulos externos para tratamento de relacionamentos espaciais. Tais sistemas espaciais possuem várias restrições e, entre elas, destaca-se a necessidade do usuário ter que conhecer a base de dados e os módulos para poder utilizá-los.

Estudos recentes indicam que o uso de banco de dados orientado a objeto ajudam a simplificar os sistemas geográficos, além de permitir aos usuários maior participação no desenvolvimento de aplicações [dA95].

1.2 Objetivos da dissertação

O assunto dessa dissertação é o estudo de Sistemas de Informações Geográficas (SIG) sob o enfoque de orientação a objeto. Visa, principalmente, comparar este enfoque com o que existe atualmente no mercado, concentrando-se no uso de SGBD relacional para verificar em que aspectos há melhor captura semântica, bem como melhor funcionalidade, com respeito ao funcionamento de uma aplicação de SIG. O termo SIG é aplicado para sistemas que realizam o tratamento computacional de dados geográficos [Cam95]. No segundo capítulo da dissertação é dada uma definição mais abrangente de SIG.

Para tanto foi escolhido uma aplicação de gerenciamento de redes, UNINet. Esta aplicação foi então modelada utilizando-se as técnicas E-R [KS93] [Dat86a] e OMT [RBP⁺91]. Os modelos foram implementados em SGBD relacional e orientado a objeto. Em seguida, foi realizada uma comparação utilizando uma série de consultas para se avaliar o desempenho e captura semântica de cada abordagem. Ao final, são levantados seus pontos positivos e negativos e o que pode ser feito para melhorar o uso dessas abordagens em SIG.

1.3 Estrutura da dissertação

A dissertação está dividida em seis capítulos. Segue-se uma descrição sucinta:

Capítulo 1: Introdução, definições básicas e estrutura da dissertação.

Capítulo 2: contém as definições básicas de SIG, características, principais componentes e cita algumas possíveis aplicações. Em seguida são descritas algumas implementações existentes de SIG.

Capítulo 3: apresenta aspectos relacionados com a modelagem de SIG. Para tanto é feita uma abordagem das técnicas de modelagem E-R e OMT, definições básicas de SGBD, descrição sucinta do SGBD O₂ e do uso de SGBD em SIG. Estas implementações usam tanto a técnica de modelagem relacional quanto a orientada a objeto. A análise é feita procurando verificar em que aspectos essas abordagens podem melhorar a modelagem de uma aplicação SIG.

Capítulo 4: descreve a aplicação exemplo, UNINet, a sua modelagem utilizando as técnicas E-R e OMT.

Capítulo 5: descreve a análise comparativa dos resultados da modelagem E-R e OMT. Em seguida é explanado a passagem do modelo E-R para tabelas no relacional bem como o esquema resultante. O esquema O₂ também é descrito e algumas considerações são feitas. Uma classificação de consultas em SIG é feita e consultas são realizadas utilizando SQL para o relacional e O₂Query para o O₂. A avaliação é feita sobre qual das duas modelagens as consultas ficaram mais "fáceis" de serem entendidas e na quantidade de tabelas/classes acessadas em cada consulta.

Capítulo 6: conclui a dissertação, com contribuições esperadas e sugestões para futuras pesquisas.

Capítulo 2

Sistema de informações geográficas - GIS/SIG

2.1 Definição de SIG

Segundo [Ooi91], sistemas de informações geográficas (SIG) são sistemas de banco de dados que permitem a manipulação, armazenamento, recuperação e análise de dados geográficos, bem como a apresentação dos dados na forma de mapas. Dados geográficos abrangem dados espaciais, geométricos, além dos dados convencionais (alfanuméricos) pertinentes aos objetos geográficos.

Um SIG é uma coleção integrada de hardware, software, dados, princípios e procedimentos para reunir, processar, analisar, arquivar ou mostrar dados ou informações sobre fenômenos humanos ou da natureza, em determinadas regiões do mundo. Aplicações potenciais de SIG podem ser encontradas em campos como geografia, silvicultura, agricultura, planejamento urbano, turismo entre outros. A tecnologia de SIG baseia-se, principalmente, em ciências como geografia e computação, tais como cartografia, sensoriamento remoto, sistemas de banco de dados, engenharia de software e computação gráfica.

Embora as tecnologias de geografia e de computação tenham progredido para um estado de maturidade, sua integração, para capacitar SIG a atender as demandas para gerenciamento e análise de informação geográfica, ainda possui obstáculos importantes para ultrapassar.

2.2 Histórico

Os primeiros SIG foram desenvolvidos na década de 60 no Canadá por Tomlinson [Tom90]. Analisando o desenvolvimento de SIG, pode-se observar que, inicialmente, houve grandes dificuldades a partir das próprias condições tecnológicas da época. Boa parte desses sistemas teve quase nenhum êxito, com exceção de alguns que ainda hoje se encontram em operação, notadamente em versões mais modernas. Esses SIG podem ser considerados de **primeira geração** [Arg95] e tinham como objetivo principal a produção de mapas. Não havia grande necessidade de guardar informações topológicas. Todas as informações eram gráficas e contínuas. Tais ambientes não possuíam suporte adequado para construir grandes bases de dados espaciais.

A década de 70 foi marcada por um grande envolvimento governamental nos assuntos de recursos naturais e de meio-ambiente, implicando em um aumento do manuseio de dados geográficos. O mesmo período coincidiu com os rápidos avanços na tecnologia dos computadores, aumento da capacidade interativa e baixos custos. Não só o governo, mas universidades, agências de pesquisa e companhias comerciais passaram a desenvolver aplicações espaciais. Essa década foi um período de consolidação mais do que inovação.

As origens de SIG comerciais remontam a década de 70. Os líderes daquela época - ESRI (Redlands, CA), Intergraph (Huntsville, AL) e Synercom (Sugarland, TX) - desenvolveram esquemas sofisticados, mas proprietários, para armazenamento e gerenciamento de dados geoespaciais [Sea95]. Os usuários não possuíam qualquer conhecimento sobre essas estruturas, que eram verdadeiras caixas pretas. Nesta **segunda geração**, surgiram várias formas de se representar os dados.

Durante a primeira metade da década de 60, SIG era construído para um usuário específico. No início da década de 70 começaram a aparecer sistemas de propósitos gerais.

Na década de 80, houve avanços significativos em velocidade, facilidade e flexibilidade com que dados geográficos podiam ser tratados. O dilema para os construtores de SIG na década de 80 que queriam utilizar SGBD comerciais era que estes SGBD eram otimizados para tratar dados simples - registros unidimensionais. Dados espaciais eram difíceis de serem adaptados aos SGBD. Problemas antigos continuavam e novos surgiam, de acordo com as possibilidades que estavam sendo abertas. Tais problemas dizem respeito ao incrível volume de dados, a interação entre os tipos de dados e os diferentes SGBD existentes, que não atendiam plenamente às necessidades de SIG, devido, principalmente, aos tipos de dados complexos destes. Outros problemas dizem respeito a segurança e confiança.

Nesta **terceira geração**, a ênfase está na utilização de SGBD relacional. Na década de 80 vários SGBD deram alguns passos em direção a geodados [Sea95]. Intergraph

adotou um SGBD popular como Ingres, Oracle e Sybase. ESRI também passou a oferecer facilidades para ligar dados armazenados em SGBD relacionais a sua estrutura particular ARC, inicialmente através do sistema INFO, depois diretamente ao sistema ARC.

Proposta diferente foi adotada pelo produto VISION* e System/9. As estruturas de dados desses produtos passaram a se localizar diretamente no SGBD, sem utilizar estruturas adicionais.

Na década de 90 deu-se o início da utilização da tecnologia orientada a objetos em SIG (**quarta geração**). Experiências passaram a ser realizadas tanto com a utilização de SGBD relacional implementando um modelo orientado a objeto, bem como diretamente em um SGBD orientado a objeto. Nesta geração que se anuncia novos requisitos estão sendo acoplados: disponibilidade de metadados, acesso via WWW e interoperabilidade [Cam95]. Metadado diz respeito ao ambiente apresentar descrições dos dados disponíveis, seja localmente ou não. A atual disponibilidade de interfaces multimídia, via Internet, deve permitir também o acesso a banco de dados geográficos. Interoperabilidade é o compartilhamento de dados e procedimentos entre SIG distintos. É um desafio considerável.

2.3 Características de SIG

Informações geográficas, ao contrário de informações comuns, possuem requisitos mais difíceis de serem atendidos. Sistemas convencionais de informações consistem de registros simples, contendo nomes, descrições e datas, normalmente unidimensionais, significando que cada campo pode ser expresso por um único valor. Já as informações geográficas são multi-dimensionais, multi-temporais e bastante volumosas. Podem ter propriedades estruturais complexas e, por isso, problemas de processamento. Um SIG pode dar suporte a informações sobre muitos temas dentro de uma região, tais como topografia, geologia, vegetação, população, economia, engenharia. Cada tema relacionado a tipos diferentes de variáveis contínuas e características discretas. Por exemplo, para efeito de alguma análise espacial pode-se ter mapas temáticos para cada assunto ou mesmo a superposição de vários assuntos.

2.4 Tipos de dados

Os dados que representam informação geográfica caracterizam-se por apresentarem quatro componentes fundamentais [Aro89]:

1. O local do fenômeno, normalmente através de um sistema de coordenadas como latitude e longitude.

2. Atributos (população da cidade, tipo de rocha, vegetação, nome da cidade, etc.).
3. Relacionamentos topológicos.
4. Componentes temporais: informações geográficas que descrevem um fenômeno em um local em um ponto específico do tempo.

Esses componentes são armazenados como dados convencionais, dados espaciais e dados pictóricos. Dados convencionais são bem resolvidos por SGBD convencionais.

Dados geográficos são inerentemente uma forma de dado espacial, que podem ser representados por pontos, linhas e polígonos [NW79], conforme Figura 2.1.

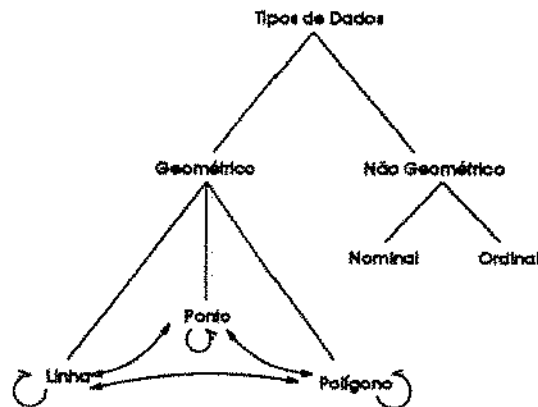


Figura 2.1: Tipos de dados.

Pontos são usados para representar um fenômeno geográfico em um local ou para representar alguma característica em um mapa que é muito pequeno para mostrar essa característica como um polígono ou uma linha. Uma linha consiste em um conjunto ordenado de pontos conectados. Linhas são usadas para representar características muito estreitas para serem mostradas como polígono ou para mostrar características que, teoricamente, não possuem largura, como fronteiras e estradas.

Existem duas propostas para representação do componente espacial de informação geográfica: modelo vetor e modelo raster (Figura 2.2) [LODL91] [Fra90].

No modelo vetor, objetos (ou condições) do mundo real são representados por pontos e linhas, que definem suas fronteiras como se eles fossem desenhados em um mapa. A posição de cada objeto é definida pelo seu posicionamento em um mapa, que é organizado por um sistema de coordenadas. Toda posição no mapa tem um único valor. Pontos, linhas e polígonos são usados para representar objetos geográficos distribuídos irregularmente.

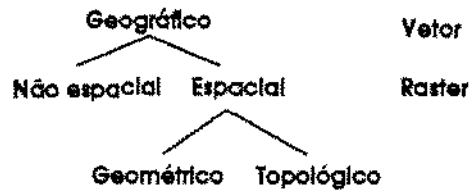


Figura 2.2: Dados geográficos.

Uma linha pode representar uma estrada, um polígono pode representar uma floresta e assim por diante.

No modelo raster, o espaço é subdividido em células (usualmente quadrados). O local de objetos é definido pelo número de linhas e colunas das células que eles ocupam. A área que cada célula representa define a resolução espacial disponível. O valor armazenado para cada célula indica o tipo de objeto que é encontrado naquele local. A cada célula é designado somente um valor. Desta forma, diferentes atributos são armazenados em arquivos separados. Portanto, na proposta raster, o espaço é ocupado por um grande número de células distribuídas regularmente, cada uma podendo ter um valor diferente. Diferentemente do modelo vetor, as unidades do modelo raster não correspondem às entidades espaciais que elas representam no mundo real. Elas são células individuais. Uma estrada não existe como uma entidade raster distinta. As células representando a estrada são as entidades.

Tabela I. Comparação dos modelos de dados vetor e raster	
Modelo Raster	Modelo Vetor
Vantagens	Vantagens
1. Simplicidade.	1. Estrutura de dados compacta
2. Operações de sobreposição são fáceis e eficientemente implementadas.	2. Permite codificação eficiente de topologia, e, como resultado, implementação mais eficiente de operações que requerem informação topológica, tais como análise de redes.
3. Alta variabilidade espacial é eficientemente representada.	3. O modelo vetor é mais adequado para suporte a gráficos que se aproximam a mapas feitos a mão.
4. É possível utilizar o formato raster para manipulação e melhoramento de imagens digitais.	
Desvantagens	Desvantagens
1. A estrutura raster é menos compacta. Técnicas de compressão de dados podem superar este problema.	1. É uma estrutura de dados mais complexa que raster.
2. Relacionamentos topológicos são mais difíceis de representar.	2. Operações de sobreposição são mais difíceis de implementar.
3. A saída de gráficos é menos estética já que fronteiras tendem a ter uma aparência quebrada, enquanto mapas desenhados a mão tem linhas suaves. Isto pode ser superado usando um grande número de células, mas pode resultar em arquivos de tamanho inaceitável.	3. A representação de alta variabilidade espacial é ineficiente.
	4. Manipulação e melhoramento de imagens digitais não podem ser feitas utilizando-se vetor.

Em ambos os modelos, a informação espacial é representada usando unidades homogêneas. No modelo raster, as unidades são células. Arquivos de dados raster comumente contém milhões de células e a posição de cada unidade é definida rigidamente. No modelo vetor, as unidades homogêneas são pontos, linhas e polígonos. No modelo vetor, os elementos podem chegar a centenas, mas não milhões, como comumente ocorre no raster.

Ambas as propostas possuem vantagens e desvantagens (Tabela I) [Aro89]. Cada modelo tende a funcionar melhor em situações onde a informação espacial é tratada na maneira mais próxima do modelo utilizado.

Modelo vetor

O modelo vetor permite o posicionamento preciso de características no espaço. O mapa é assumido como um espaço de coordenadas contínuas, onde uma posição pode ser definida tão precisa quanto o desejado. O modelo vetor assume que as coordenadas de posição são matematicamente exatas.

Uma característica de ponto é gravada como um único par de coordenadas XY, uma linha como uma série de coordenadas XY e uma área como um laço fechado de pares de coordenadas XY, que definem a fronteira da área (Figura 2.3) [Aro89].

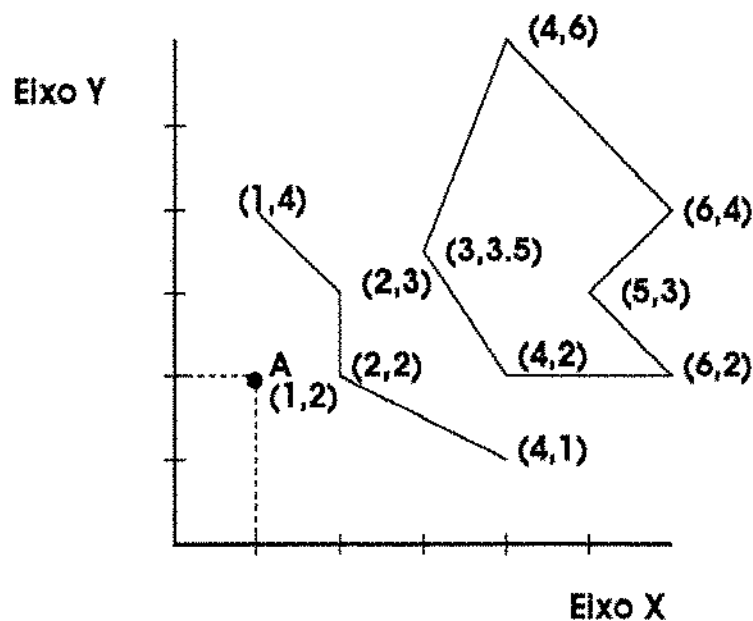


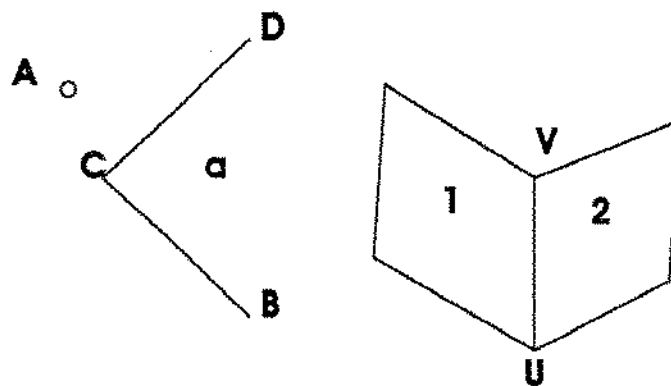
Figura 2.3: Ponto, linha e polígono.

Na Figura 2.3, as coordenadas estão em unidades arbitrárias. Em SIG, posições são comumente armazenadas usando um sistema de coordenadas geográficas padrão como UTM, Latitude x Longitude, e outros.

Uma descrição abrangente de vetores é dada em [ROG89](tabela II.).

Tabela II. Uma classificação de estruturas de dados vetor para SIG
Estruturas simples de vetor
Estruturas não topológicas (espaguete)
Estruturas topológicas simples
Estruturas topológicas orientadas
Estruturas topológicas indexadas hierarquicamente
Estruturas de vetor híbridas
Estruturas de rede indexadas
Estruturas geo-relacional

No modelo espaguete, o mapa é convertido linha a linha para uma lista de coordenadas XY. Um ponto é codificado como um único par de coordenadas XY e uma linha como uma cadeia de pares de coordenadas XY. Embora todas as características espaciais sejam gravadas, os relacionamentos espaciais entre essas características não são codificados (Figura 2.4).

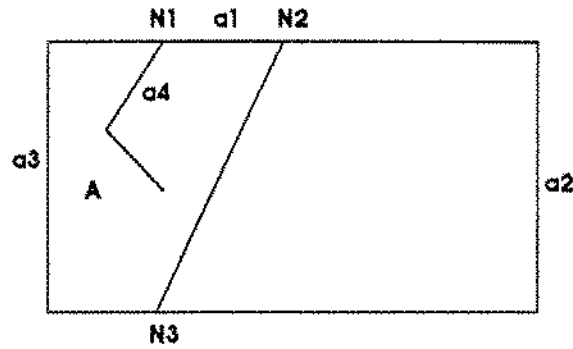


Tipo	Nome	Coordenada
ponto	A	Xa, Ya
Reta	a	B,C,D
Polígono	1	..., U, V
	2	...,U, V

Figura 2.4: Modelo espaguete.

Os modelos topológicos são os mais utilizados para representar relacionamentos espaciais. Topologia é o método matemático para definir relacionamentos espaciais. São

armazenados arcos e nós. Um arco é um conjunto de pontos que começam e terminam em um nó. Um nó é um ponto de interseção onde dois ou mais arcos se encontram. Nós isolados, não conectados a arcos, representam pontos. Um polígono é compreendido de uma cadeia fechada de arcos, que representam as fronteiras da área (figura 2.5).



Topologia

Polígono	Arco	Nós	Arco
A	a1a2a3	N1	a3a4a1

Arco	Nó Inicial	Nó Final	Pol.Esq.	Pol.Dir
a1	N1	N2	-	A

Figura 2.5: Modelo topológico.

O modelo orientado consiste em, explicitamente, atribuir direção a segmentos de linhas codificando nós “destino” e “origem”. O sistema GBF/DIME (Geographic Base File/Dual Encoding Map Encoding), desenvolvido pelo órgão americano de censo, utiliza esta estrutura de dados.

O modelo de Peucker e Chrisman, POLYGON (POLYgon conVeRTer), é um exemplo de indexado. Consiste em armazenar cada estrutura de dados de forma separada. Desta estrutura partem ponteiros para uma lista encadeada, onde constam as fronteiras de cada polígono.

Um dos maiores problemas no uso de vetor hierarquicamente estruturado é que não há provisão para ligação de relacionamentos físicos entre itens. Em adição, são utilizados esquemas de banco de dados relacional ou e de redes, que são os híbridos.

A vantagem do modelo de rede são relacionamentos físicos estabelecidos entre entidades, possibilitando navegação rápida. A desvantagem inerente ao modelo em rede é a estrutura com pouca flexibilidade. Uma aplicação é o sistema PRE-TIGRIS INTERGRAPH.

O relacional é um dos mais utilizados e a principal vantagem é que relacionamentos lógicos podem ser estabelecidos prontamente entre entidades. A desvantagem é o gerenciamento pesado de tabelas. Uma aplicação existente é o sistema VISION*.

2.5 Componentes básicos de um SIG

Os componentes básicos de um SIG são [Aro89] [Dan90]:

- entrada;
- gerenciamento de dados (armazenamento e recuperação);
- manipulação e análise; e
- saída.

O componente de entrada de dados converte dados das suas formas existentes para uma que possa ser usada pelo SIG. O processo de captura é o mais caro, tanto em custo monetário quanto em tempo gasto. O processo é composto de duas fases: aquisição dos dados espaciais e pré-processamento. Os dados normalmente estão na forma de mapas, tabelas, fotos de satélites. As técnicas utilizadas nesta fase são: digitalização manual, *scanning* de fotos, utilização de procedimentos geométricos para cálculo e entrada das coordenadas dos objetos espaciais e tradução de arquivos digitais existentes.

A fase de pré-processamento consiste na conversão dos dados já adquiridos para um formato aceito pelo sistema. Consiste em conversão de formatos, generalização e redução de dados, detecção de erros, entre outros [SE90]. Este é o maior gargalo na implementação de um SIG.

O gerenciamento inclui as funções necessárias para o armazenamento e recuperação dos dados da base de dados, de forma a torná-los disponíveis para o usuário. O modo como os dados estão estruturados e o modo como os arquivos podem estar relacionados irão definir a forma e a velocidade da recuperação.

As funções de manipulação e análise determinam que a informação pode ser gerada por um SIG, sendo essas as mais importantes do ponto de vista dos usuários do sistema. Restrições de performance, de tempo e custos financeiros decidem quais alternativas serão seguidas no processo de manipulação dos dados.

A saída varia em qualidade, precisão e facilidade de uso, podendo ser na forma de imagens, mapas, tabelas e/ou texto. Como um dos principais objetivos de SIG é a manipulação de dados espaciais, uma atenção especial deve ser dada à sua visualização espacial.

2.6 Aplicações

SIG são usados, principalmente, em aplicações de planejamento e administração, as quais podem ser divididas em dois grupos distintos: cadastral (urbano e rural) e ambiental. Outras aplicações como cartas náuticas eletrônicas [dO95] podem ser consideradas como mistas. O grupo de cadastro visa manter informações relacionadas a ruas, avenidas, escolas, hospitais, propriedades particulares, acidentes geográficos (rios, córregos), para que atividades de planejamento urbano e de administração pública possam ser desenvolvidas. O objetivo do segundo grupo (ambiental) é o de manter informações relacionadas a recursos naturais renováveis a fim de facilitar a sua exploração racional.

As aplicações do primeiro grupo (cadastral) são geralmente representadas através de vetor, enquanto a do segundo são representadas através de raster.

Mapeamento Urbano é uma aplicação do primeiro grupo. Os dados do mapeamento urbano são logradouros, linhas centrais, quadras, lotes e prédios. Várias aplicações se utilizam deste mapeamento como ponto de partida, necessitando que estes dados estejam bem gerenciados.

Este mapeamento é utilizado como forma de ajudar no planejamento, organização e controle da cidade [JdVB94] [JdVB95], permitindo o acompanhamento do crescimento da cidade e direcionando a expansão da malha urbana, bem como o gerenciamento da utilização dos recursos públicos.

Outras aplicações do primeiro grupo incluem o zoneamento urbano, planejamento viário, de saúde, distribuição de energia elétrica e telecomunicações. Também são realizadas em cima do mapeamento urbano.

O zoneamento urbano é a divisão da cidade em setores, de acordo com as suas características, de forma a se ter em uma cidade regiões industriais, residenciais, de lazer, etc. Esta divisão permite um melhor planejamento e atendimento das necessidades de cada zona.

O planejamento de saúde está relacionado à distribuição de hospitais e postos de saúde, de acordo com a necessidade do local. Vias de acesso e todos os dados dos hospitais devem também fazer parte do SIG. A finalidade pode ser a pesquisa de quais os hospitais com leito disponível está mais próximo de um determinado local.

Como um exemplo de aplicações ambientais, pode-se citar a conservação de recursos naturais (fauna e flora), através do mapeamento das áreas devastadas e outras que devem ser preservadas, bem como identificação de espécimes ameaçadas de extinção, a fim de colocar as suas áreas como de preservação.

As formas de representação podem ser feitas através de mapas temáticos, mapas ca-

dastrais e redes. Mapas temáticos contêm regiões geográficas definidas por um ou mais polígonos. Mapas de solos e regiões agrícolas são exemplos típicos. No mapa cadastral, os elementos são objetos geográficos, que possuem atributos e podem estar associados a várias representações gráficas.

O exemplo a ser utilizado nessa dissertação trata de uma rede de computadores distribuídos geograficamente. Os objetivos são o planejamento e a administração desta rede. Dentre os aspectos que interessam está o mapeamento urbano, sobre o qual a rede será representada.

Redes são um tipo especial de mapas cadastrais, onde os objetos geográficos possuem uma localização geográfica exata que está sempre associada a atributos descritivos, presentes no banco de dados.

2.7 Alguns SIG protótipos e comerciais

Nesta seção será feita uma pequena descrição de alguns SIG mais conhecidos e que utilizam de alguma forma um SGBD.

A análise será dividida em protótipos e produtos existentes no mercado. Os protótipos são basicamente desenvolvimentos de universidades e centros de pesquisa. Entre os protótipos a serem analisados estão: PROBE, TIGRIS, STARBURST, SIRO-DBMS e EXODUS. Os produtos comerciais analisados são: ARC/INFO, VISION.

2.7.1 PROBE

O PROBE [MO86] [Ooi91] [Feu93] é um projeto de pesquisa de um SGBD orientado a objeto, voltado para aplicações não tradicionais, muitas das quais envolve dados espaciais e dados com estrutura complexa.

O objetivo é prover um sistema de banco de dados de propósito geral para aplicações envolvendo dados temporais e espaciais e outros tipos de dados com estrutura complexa.

O modelo de dados do PROBE tem dois tipos básicos de objetos: entidade e funções. Uma entidade é um objeto que possui identidade. Entidades com características similares são agrupadas em coleções chamadas tipos de entidade. Função é geralmente definida como um relacionamento entre coleções de entidades e valores escalares. Existem duas classes genéricas de funções: funções computadas, com valores de saída calculadas por procedimentos, e funções armazenadas, com valores de saída determinados por uma pesquisa convencional de banco de dados de uma função armazenada *extend*. Tipos de entidade podem ser divididos em subtipos, formando hierarquias de generalização. No topo da hie-

rarquia, ambas entidades e funções são membros do tipo genérico ENTIDADE. Operações genéricas sobre objetos (entidades e funções) como seleção, aplicação de função, operações de conjunto e formação de novas funções estendidas, foram definidas na forma de uma álgebra bem similar em alguns aspectos a álgebra definida para o modelo relacional.

A arquitetura do SGBD PROBE consiste em um núcleo que trata conjuntos de objetos genéricos, enquanto as classes de objeto tratam objetos individuais de tipos especializados.

Os objetos básicos no modelo de dados espacial são conjuntos de pontos (*point sets*). *Point set* de um objeto é o conjunto de pontos no espaço ocupado por aquele objeto. Um conjunto de pontos é um membro do tipo CONJUNTO-DE-PONTOS. Este tipo é uma especialização do tipo ENTIDADE que introduz uma coleção de operadores especiais como união, interseção, diferença, bem como, predicados especiais como sobreposição, está contido e proximidade. Conjunto de pontos exhibe uma estrutura hierárquica, pois um conjunto de pontos contido em um espaço pode conter outros conjuntos de pontos.

PROBE dispõe de um filtro geométrico para otimizar consultas espaciais. Como estruturas do tipo R-trees não são usadas, a maioria das operações geométricas são do tipo laço aninhado (*nested loop*). A função deste filtro é otimizar laços. O problema com algoritmos com laços aninhados é desempenho. Os laços aninhados levam algoritmos de tempo polinomial. A saída produzida do filtro é um conjunto de objetos candidatos (ou conjunto de grupos de objetos - um membro do grupo vem de cada laço). Eles são prováveis de satisfazer a consulta. O conjunto de candidatos será refinado para produzir a resposta precisa aplicando a cada candidato um predicado determinado da classe de objeto espacial suprido.

Uma versão estendida do DAPLEX é utilizada como linguagem de consulta.

Um protótipo do PROBE está em experiência [Ooi91].

Para implementar um modelo de dados de SIG sobre o POBRE ter-se-ia facilidades derivadas de construtos já existentes na linguagem, como conjuntos-de-pontos que já tem embutido operadores espaciais (união interseção e diferença). Permite também hierarquias de generalização, podendo-se definir tipos genéricos e a partir desses ir refinando sucessivamente em subtipos mais especializados.

2.7.2 ARC/INFO

É um produto comercial. Dados não espaciais suportados por SGBD e dados espaciais suportados por um sistema adicional, construído especificamente para isso [Mor92] [Aro89] [Ooi91].

É um sistema genérico para modelagem e análise de dados espaciais armazenados, desenvolvido pela ESRI (Environment Systems Research Institute). ARC refere-se às es-

truturas de dados topológicos e algoritmos. INFO refere-se ao modelo de dados compostos e processos associados.

O modelo de dados georrelacional combina uma visão geográfica especializada dos dados com um modelo de dados relacional convencional. Dados de localização e temáticos são representados como conjuntos de tabela.

O modelo de dados do ARC/INFO é baseado na idéia que dados geográficos podem ser representados usando um conjunto genérico de *features*: ponto, linhas, áreas, superfícies, etc. Cada feature genérico tem associado informação de localização e temática. As features definidas pelo usuário, em termos de características genéricas, é feita pela associação de identificadores de feature como atributos de features genéricos.

No ARC/INFO dados de localização são representados usando um modelo de dados topológico. Dados temáticos usam o modelo relacional.

Ele define atributos de local e temáticos das características de mapa em uma dada área. Uma cobertura é definida com um conjunto de características, onde cada característica tem um local (definida pelas coordenadas e ponteiros topológicos para outras características) e, possivelmente, atributos (definidos com um conjunto de itens ou variáveis).

Existem vários tipos de features que podem estar presentes em uma cobertura. Cada uma dessas classes de features pode ter associado uma informação de local e temático. Essas features podem ser arcos, nós, polígonos. A cada feature podem estar associadas tabelas de atributo. No ARC/INFO as linhas da tabela são denominados registros e as colunas são denominados itens.

O ARC/INFO possui uma biblioteca de mapas, que consiste em um dispositivo que armazena as coberturas. As coberturas são armazenadas simultaneamente em duas dimensões: por assunto ou conteúdo dentro de camadas e por local dentro de *tiles*. A área geográfica representada por um mapa é dividida em um conjunto de tiles não sobrepostos. Uma camada é um tipo de cobertura dentro da biblioteca. Todos os dados em uma mesma camada têm as mesmas características de cobertura e atributos.

O banco de dados ARC/INFO é implementado utilizando técnicas de modelagem de banco de dados relacional. Uma cobertura é definida por um conjunto de relações. Essas relações definem valores geométricos, topológicos e de atributos das várias características na cobertura.

O ARC/INFO possui um conjunto extenso de operadores para a cobertura: sobreposição, geração de polígono de Thiessen, interpolação de contorno, transformação de coordenadas.

Além do SGBD INFO, existem implementações utilizando os SGBD relacionais ORA-

CLE e INGRES.

ARC/INFO é um sistema híbrido, ou seja, a parte espacial é armazenada em estruturas situadas fora do SGBD. A modelagem de dados pode ser considerada mais simples do que a realizada no POBRE, pois a parte não-espacial pode ser modelada via relações (tabelas), bastante conhecida.

2.7.3 TIGRIS

O sistema TIGRIS [Her92] e o seu descendente DYNAMO são implementados com o paradigma de orientação a objeto. O modelo sobre o qual TIGRIS é construído é uma combinação de vários modelos bem conhecidos, como modelo relacional, modelo relacional estendido e orientado a objeto. O modelo OO como utilizado no TIGRIS é uma extensão do modelo relacional padrão (RM/T), introduzido por Codd em 1979. Na implementação do TIGRIS, podemos ver classes de objeto como uma tabela primária no modelo relacional, cada objeto correspondendo a uma tupla dentro daquela tabela e cada pedaço de dados da instância correspondendo a um atributo da tabela. Relacionamentos podem ser vistos como tabelas separadas com chaves compostas consistindo de identificadores de objeto como chaves estrangeiras para tabelas descritivas.

Dentro do TIGRIS características (*features*) podem ser qualquer item de interesse para o usuário como estradas, rios, riachos. Além disso, cada característica pode ser de qualquer complexidade e ter relacionamentos explícitos com outras características.

A estrutura dos dados que o usuário vê pode ser quebrada em três categorias: temas, características compostas e características base. Um tema é uma cobertura completa de uma única característica em um conjunto de características relacionadas da superfície da terra.

Características de base podem ser representações de entidades geográficas separadas ocorrendo na superfície da Terra. Características de base contêm somente aqueles valores de atributo que são precisos sobre sua extensão. Atributos que são precisos para somente uma parte da característica implica que a característica deve ser subdividida.

TIGRIS contém um modelo topológico bidimensional consistindo de pontos (nós), curvas entres esses pontos (arestas) e áreas conectadas limitadas por essas curvas (faces). Todas as relações topológicas entre esses elementos são armazenados ou explicitamente ou podem ser derivados facilmente de relacionamentos armazenados explicitamente.

O modelo topológico é construído e mantido na base de coordenadas (x,y). Cada coordenada (x,y) no mapa será associada somente a um objeto topológico, embora uma posição possa ser usada para definir mais de um objeto topológico.

O TIGRIS utiliza SQL com extensões. O SQL padrão opera sobre relações armaze-

nadas como tabelas de tuplas. No TIGRIS as tuplas consistem em identidades de objeto. As extensões do O-SQL (Object-SQL) incluem:

- a capacidade para chamar qualquer método de qualquer objeto diretamente da linguagem de consulta para uso em predicado em *selects*, em operadores de relação, ou para alterações.
- a capacidade para definir marcos que auxiliem a interpretação de declarações de consulta de acordo com o contexto da classe.
- a capacidade de agrupar declarações de consulta juntas em unidades únicas para execução, com comunicação interconsulta via relações temporais gerenciados pelo sistema.
- a capacidade para armazenar, acessar, e consultar tipos de dados abstratos complexos dentro dos atributos.

2.7.4 SIRO-DBMS

SIRO-DBMS (Spatial Information in Relational Open-architecture DataBase Management System) [Ooi91] [Abe89] é um *toolkit* de banco de dados geográficos que provê alguns tipos de dados espaciais e métodos de acesso espaciais como acessórios (extensões) ao núcleo de um SGBD relacional. Foi desenvolvido no Centro para Sistemas de Informações Espaciais, CSIRO Divisão de Tecnologia da Informação, como um sistema protótipo para explorar o potencial de modelos de dados relacional estendido para banco de dados espaciais. SIRO-DBMS provê um alto nível de integração de imagens, dados vetor, texto e, adicionalmente, métodos de acesso especialistas e opções para representação interna da geometria de objetos para ajustar um projeto de banco de dados aos requerimentos de uma aplicação.

As extensões ao modelo relacional consistem em tipos de dados definidos pelo usuário e extensões para gerenciamento de dados.

SIRO-DBMS suplementa as facilidades padrões de SGBD relacionais com visões de mais alto nível de entidades e operações para módulos escritos em C. As facilidades do SIRO-DBMS são limitadas a criação, alteração e consulta do banco de dados, com a análise, display e outras operações de interface de usuário do sistema.

A interface entre SIRO-DBMS e SGBD relacionais é programado no SQL de modo que o SIRO-DBMS é potencialmente portátil entre SGBD relacionais que ofereçam SQL.

Duas extensões para o modelo são providos: um conjunto de tipos de dados geométricos e uma forma limitada de hierarquia "IS_A".

Cinco tipos de dados de localização são providos: para ponto, retângulos, polígonos e imagem. O efeito é que o desenvolvedor de aplicação pode tratar uma descrição especial de uma entidade com apenas um outro atributo. Uma forma de hierarquia "IS_A" é provido através de classes, denominados conjuntos de relações de entidades espaciais, com uma classe capaz de ser tratada como sujeito de uma pesquisa.

Existem dois elementos para implementação de tipos de dados espaciais. O primeiro procura prover uma fundação para a extensão simplificada do conjunto de tipos de dados através do mapeamento de cada tipo de dados geográfico e um conjunto de atributos atômicos cujos tipos são projetados do conjunto de tipos de dados providos pelo SGBD relacional e pela linguagem C. O tipo ponto, por exemplo, é mapeado como dois atributos, um atributo interpretado como uma coordenada X e outro interpretado como coordenada Y, ambos "número" para o SGBD relacional e "float" para C.

A linguagem utilizada é uma extensão do SQL - SESQL (Spatial Extended SQL).

2.7.5 VISION*

VISION* [Geo92] é um SIG desenvolvido pela Geovision em 1985. Todos os dados estão armazenados em um SGBD relacional (Oracle ou Ingres). É usado para criar, manter, mostrar e analisar informação geográfica.

O VISION pode tratar dados espaciais, dados convencionais e imagens. Dados espaciais identificam o local geográfico e seus relacionamentos (topológicos e de conectividade) entre características geográficas. Dados convencionais são dados descritivos, que podem ser associados a um local específico a um banco de dados como um todo. Dado imagem é uma informação pictórica como fotos, esquemas e gráficos, que podem estar associados a uma característica¹ ou grupo de características ou a um banco de dados como um todo. Todos os dados (espacial, convencional e imagem) são mantidos juntos no mesmo banco de dados lógico.

Os dados são armazenados em uma hierarquia de estruturas, que ajudam a simplificar e a esclarecer os relacionamentos e facilitam o entendimento das interações dos dados.

Muitas estruturas lógicas definidas pelo usuário e estruturas topológicas podem ser desenvolvidas e armazenadas dentro do banco de dados VISION*. Estas estruturas são: features, redes, camadas e grupos (Figura 3.8).

Uma entidade geográfica que tem um ou mais pontos de coordenada é chamada de *feature*. Uma feature é a unidade básica no banco de dados. Uma feature pode ser composta de um número ilimitado de coordenadas (x,y) ou (x,y,z), se é bidimensional ou tridimensional, respectivamente. Existem três tipos de features: pontos, linhas e

¹feature

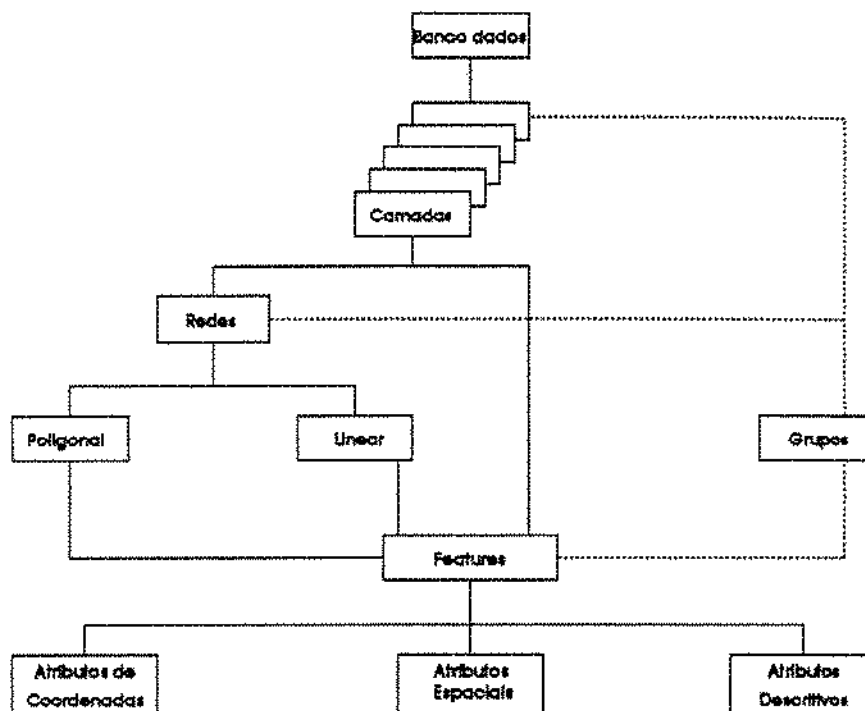


Figura 2.6: Estrutura do banco de dados.

polígonos.

Um feature do tipo ponto deve ter no mínimo um ponto de coordenada para localizá-lo na superfície da terra. Uma linha deve ter um ponto inicial e um final. Pode conter um número ilimitado de outros pontos para rastrear sua rota. Um polígono é um feature que compreende uma área e é identificado por um ponto chamado centróide localizado em algum lugar dentro da área.

Nós são features definidos pelo usuário. São um caso especial de ponto. Eles podem existir em uma rede com ou sem linhas ligando o ponto.

Uma rede é uma estrutura que contém features topologicamente relacionados. Os relacionamentos topológicos são armazenados e mantidos dentro do banco de dados e são usados pelo sistema em consultas avançadas e funções de análise. Existem quatro tipos de redes: poligonal, linear, definida pelo usuário e não conectada.

Redes de polígono contém um número ilimitado de polígonos conectados e “livres”. Cada linha na rede tem um nó em cada ponto terminal.

Rede linear contém features de linha, que são conectados em seus extremos ou são separados geograficamente. Nós podem existir independente da existência de uma linha conectando-os. Nas redes definidas pelo usuário é ele quem define a tabela de conectividade.

Redes não conectadas podem conter qualquer tipo e combinação de features. Camadas são estruturas do banco de dados que tematicamente agrupam features e redes que estão relacionados logicamente (rios, lagos e oceanos podem ser agrupados em uma camada de hidrologia).

A estrutura grupo permite a criação de uma estrutura temporária para realizar manipulações. Grupos são freqüentemente utilizados para executar alguma manipulação sobre um número de features, que de outro modelo não seriam associados.

Aplicações típicas de VISION* incluem gerenciamento de redes de distribuição de gás, transmissão de eletricidade, telefonia, gerenciamento de recursos naturais, entre outras.

A riqueza dos tipos de dados em VISION* e a possibilidade de construir novos tipos mais complexos permite uma modelagem eficiente do fenômeno a ser analisado. Tipos como camadas e redes ampliam o leque e permitem que o usuário possa utilizá-los diretamente.

2.7.6 GRASS

GRASS (Sistema de Suporte de Análise de Recursos Geográficos)[?] é um SIG desenvolvido pelo exército americano para prover ferramentas de gerenciamento de tarefas militares. Da mesma forma, GRASS tem muitas aplicações em área civil, em projetos de planejamento e projeto.

Algumas propriedades são:

- representação de dados na forma raster, vetor e pontos;
- manuseio de imagens (mostrar, geo-referenciar, comparar e classificar imagens de satélite e de fotografias aéreas);
- possibilidade de se ligar a um SGBD para ajudar no gerenciamento dos dados.

Capítulo 3

Modelagem de dados em SIG

3.1 Modelo de dados

O núcleo de qualquer SIG é o modelo de dados que consiste em um conjunto de ferramentas conceituais utilizadas para descrever a estrutura do banco de dados. A estrutura do banco de dados engloba a descrição dos seus dados, relacionamentos entre eles, semântica e restrições que atuam sobre esses dados. Pode também incluir um conjunto de operações, utilizadas para especificar consultas e atualizações no banco de dados.

Os vários modelos de dados que têm sido propostos dividem-se em três diferentes grupos:

- modelos lógicos baseados em objeto;
- modelos lógicos baseados em registros;
- modelos de dados físicos.

Modelos lógicos baseados em objetos são usados na descrição de dados nos níveis conceitual e visual. Eles se caracterizam pelo fato de fornecerem, de forma conveniente, capacidades de estruturação flexíveis e admitem restrições de dados para serem explicitamente especificados. Dentre esses modelos estão o modelo entidade-relacionamento (E-R) e o orientado a objeto (OO). O modelo E-R tem grande aceitação no projeto de banco de dados e é bastante utilizado na prática. O modelo orientado a objeto inclui muitos dos conceitos de E-R e permite tanto a representação de código executável como de dados.

Modelos lógicos baseados em registros são usados nas descrições de dados nos níveis conceitual e visual. Ambos (em registros e em objetos) são usados para especificar a

estrutura lógica geral do banco de dados e para fornecer uma descrição de alto nível da implementação.

Modelos baseados em registros são assim chamados porque o banco de dados é estruturado em registros de formato fixo de diversos tipos. Cada tipo de registro define um número fixo de campos ou atributos e cada campo é usualmente de um tamanho fixo. Os modelos baseados em registros não incluem um mecanismo para representação direta do código dentro do banco de dados. No entanto, existem linguagens separadas que são associadas ao modelo para expressar as consultas e atualizações no banco de dados. Os três modelos mais aceitos são: relacional, em rede e hierárquico.

SGBD específicos são desenvolvidos a fim de melhor capturar as características de técnicas de modelagem de dados. Desta forma, existem gerenciadores hierárquico, em rede e relacional, com o objetivo de melhor capturar os referidos modelos, embora, na verdade, possa-se implementar qualquer modelo em qualquer SGBD, necessitando para isso, apenas um mapeamento adicional do modelo para o SGBD.

A seguir, serão descritos, os modelos entidade-relacionamento (E-R), relacional, orientado a objeto e o método OMT, e os SGBD a serem utilizados.

3.2 Modelo E-R

O modelo entidade-relacionamento (E-R) é bem conhecido. Foi descrito por Chen(1976) [KS93], com a intenção de ser uma ferramenta de projeto de banco de dados lógico e não é implementado como um SGBD. É muito utilizado na modelagem conceitual de sistemas sendo, em seguida, mapeado para o modelo relacional e implementado em um SGBD relacional.

Essencialmente, o modelo E-R consiste em “entidades”, “relacionamentos” entre essas entidades e “atributos” que fazem parte das entidades e dos relacionamentos. Não possui componentes de operação de dados. O conjunto inicial de construtores do modelo E-R foi ampliado. Dentre as extensões propostas estão: generalização, especialização, agregação e associação ordenada.

Muitos sistemas podem ser modelados usando entidades, atributos e relacionamentos, incluindo sistemas com componentes espaciais. Uma importante característica da modelagem E-R é a passagem rápida para o modelo relacional, de forma natural e compreensível.

3.2.1 Entidades, atributos e relacionamentos

Uma entidade é um objeto que existe e é distinguível dos outros objetos. Uma entidade pode ser concreta, como uma pessoa ou um livro, ou pode ser abstrata como um conceito ou um feriado [Aro89].

Um conjunto-entidade é um conjunto de entidades do mesmo tipo. O conjunto de pessoas matriculados em uma universidade pode considerar alunos e as pessoas que trabalham nesta universidade como sendo funcionários. Esses conjuntos-entidade não são necessariamente disjuntos, pois uma pessoa pode ser aluno e funcionário da mesma universidade.

Uma entidade é representada por um conjunto de atributos. Atributos possíveis para aluno podem ser nome-aluno, matrícula, e endereço. Para disciplina podem ser nome-disciplina, horário e local. Para cada atributo, existe um conjunto de valores permitidos, chamado domínio daquele atributo. O domínio do atributo nome-aluno pode ser o conjunto de todas as cadeias de texto de um certo tamanho. Da mesma forma, o domínio do atributo matrícula pode ser o conjunto de todos os inteiros positivos.

Pode-se considerar um atributo como uma função que mapeia um conjunto de entidades em um domínio. Dessa forma, uma entidade pode ser descrita como um conjunto de pares (atributo, valores de dados), um para cada atributo do conjunto-entidade. A entidade aluno pode ser descrita como (nome, Claudio), (matrícula, 925316), (endereço, Rua José da Silva 999), que significa que a entidade descreve um aluno chamado Claudio, com matrícula 925316 e que mora na rua José da Silva 999.

Um relacionamento é uma associação entre diversas entidades. Um conjunto-relacionamento é um conjunto de relacionamentos do mesmo tipo. Formalmente é a relação matemática entre $n \geq 2$ (possivelmente não-distintos) conjuntos-entidade. Se E_1, E_2, \dots, E_n são conjuntos-entidade então o conjunto-relacionamento R é um subconjunto de

$$\{e_1, e_2, \dots, e_n\}, \text{ onde } e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n$$

Considere os conjuntos-entidade Aluno e Disciplina. O conjunto-relacionamento AlunoDisciplina descreve a associação entre alunos e disciplinas.

O relacionamento AlunoDisciplina é binário, ou seja, envolve apenas dois conjuntos entidade. A maioria dos conjuntos-relacionamento em um banco de dados é binária. Como exemplo, pode-se considerar o relacionamento ternário entre as entidades Aluno, Professor e Disciplina. Este relacionamento especifica que um aluno cursa uma determinada disciplina ministrada por um determinado professor. Sempre é possível substituir um conjunto-relacionamento não-binário por uma série de conjuntos-relacionamento binários.

O atributo é denominado simples quando assume um valor único e indivisível para

cada entidade. Quando assume um valor único, mas o valor pode ser dividido em valores significativos menores ele é chamado de **atributo composto**. Um atributo é monovalorado se assume um único valor para uma entidade e multivalorado, caso contrário.

Um relacionamento pode também ter atributos decorrentes do relacionamento entre as entidades. O papel que uma entidade exerce em um relacionamento é chamado de **função**. Funções são normalmente implícitas e não são usualmente especificadas. São úteis quando o sentido de um relacionamento necessita de esclarecimentos, por exemplo, quando os conjuntos-entidade de um conjunto-relacionamento não são distintos. Um caso seria o conjunto-relacionamento *Chefia* entre funcionários, que necessita ser esclarecido pois não há como deduzir a partir dos conjuntos-entidades. A Figura 3.1 mostra alguns exemplos de notação gráfica no modelo E-R.

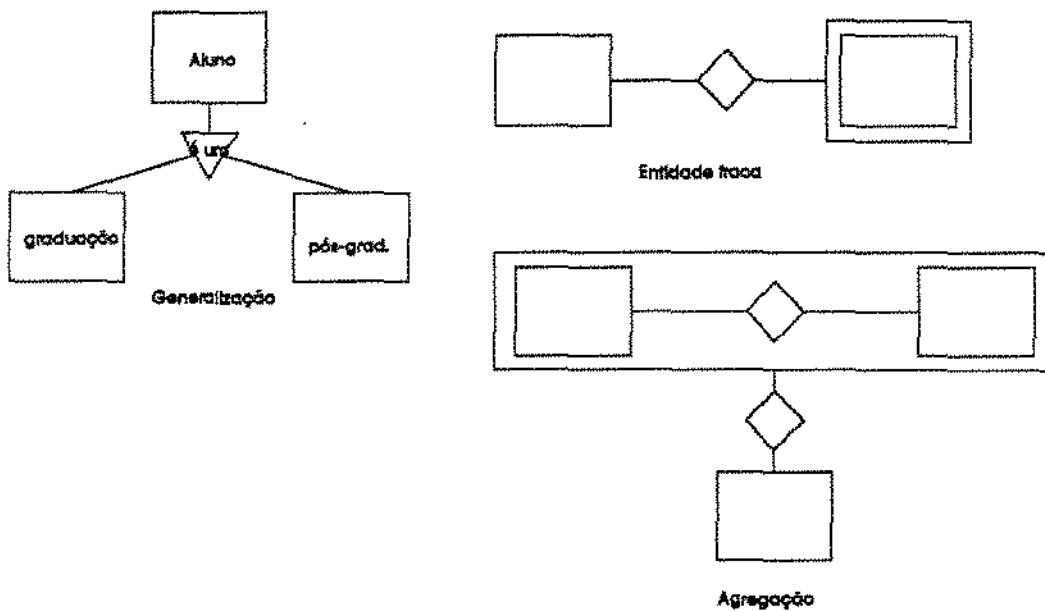


Figura 3.1: Diagramas do modelo E-R.

3.2.2 Cardinalidade

Um esquema E-R pode definir certas **restrições** com as quais o conteúdo do banco de dados tem que estar de acordo. Um restrição importante é o *mapeamento de cardinalidade*, que expressa o número de entidades às quais outra entidade pode ser associada via um conjunto-relacionamento.

Para um conjunto-relacionamento binário R entre conjuntos-entidade A e B , o mapeamento de cardinalidade pode ser:

- Um-para-um. Uma entidade em A está associada a, no máximo, uma entidade em B , e uma entidade em B está associada a, no máximo, uma entidade em A .
- Um-para-muitos. Uma entidade em A está associada a qualquer número de entidades em B . Entretanto, uma entidade em B está associada a, no máximo, uma entidade em A .
- Muitos-para-muitos. Uma entidade em A está associada a qualquer número de entidades em B e uma entidade em B está associada a qualquer número de entidades em A .

Como exemplo, pode-se considerar o conjunto-relacionamento Aluno-Curso. Se em uma universidade um aluno só pode estar matriculado em um curso, o relacionamento é muitos-para-um. Se o aluno pode se matricular em mais de um curso então o conjunto-relacionamento é muitos-para-muitos.

3.2.3 Chaves

Entidades pertencentes a um conjunto-entidade são distintas, conceitualmente são diferentes, mas no banco de dados a diferença precisa ser expressa em termos de seus atributos. Uma **superchave** é um conjunto de um ou mais atributos que, tomados coletivamente, permite identificar unicamente uma entidade no conjunto-entidade. No conjunto-entidade Aluno, matrícula é uma superchave, matrícula e nome-aluno também é superchave, mas nome-aluno não é uma superchave, pois vários alunos podem ter o mesmo nome.

O importante é obter uma superchave mínima, ou seja, um subconjunto da superchave capaz de identificar a entidade. Chama-se as superchaves mínimas de **chaves candidatas**.

Utiliza-se o termo **chave primária** para identificar uma chave candidata que é escolhida como principal significado de identificação de entidades dentro de um conjunto-entidade.

A chave primária serve para identificar as várias entidades dentro de um conjunto. Da mesma forma, são necessárias chaves para identificar os vários relacionamentos dentro de um conjunto-relacionamento.

Seja R um relacionamento envolvendo o conjunto-entidade $\{E_1, E_2, \dots, E_n\}$. Assuma que a chave primária (E_i) designa o conjunto de atributos que forma a chave primária do conjunto-entidade E_i , que os atributos nomes de todas as chaves primárias sejam únicos e que R não tem atributos. Então os atributos que descrevem relacionamentos individuais do conjunto R , representado por $\text{atributo}(R)$, são:

$$\text{chave-primária}(E_1) \cup \text{chave-primária}(E_2) \cup \dots \cup \text{chave-primária}(E_n)$$

Caso R tenha atributos descritivos $\{a_1, a_2, \dots, a_m\}$, então o conjunto atributo(R) consiste em:

$$\text{chave-primária}(E_1) \cup \dots \cup \text{chave-primária}(E_n) \cup \{a_1, a_2, \dots, a_m\}$$

Considere o conjunto-relacionamento Aluno-Disciplina e as chaves matrícula para Aluno e código-disciplina para Disciplina. Uma vez que o conjunto-relacionamento tem atributos como nota, período e presença o conjunto atributo (Aluno-Disciplina) consiste nos atributos matrícula, código-disciplina, nota, período e presença.

Se o conjunto-relacionamento R não possui atributos associados a ele, então o conjunto atributo(R) forma uma superchave. Esta superchave é uma chave primária, caso o mapeamento de cardinalidade seja muitos-para-muitos.

Se o conjunto-relacionamento R tiver diversos atributos associados a ele, então uma superchave é formada como antes, com a possível adição de um ou mais desses atributos. No caso do conjunto-relacionamento Aluno-Disciplina, a chave primária é (matrícula, código-disciplina e período).

Quando um conjunto-entidade não tem atributos suficientes para formar uma chave primária, ele é denominado **conjunto-entidade fraco**. Para que um conjunto-entidade fraco seja significativo, ele deve ser parte de um conjunto-relacionamento um-para-muitos. Este conjunto-relacionamento não deve ter atributos descritivos, uma vez que qualquer atributo requerido pode estar associado ao conjunto-entidade fraco.

A chave primária de um conjunto-entidade fraco é formado pela chave primária do conjunto-entidade forte, no qual ele é dependente, mais um discriminador.

3.2.4 Generalização e especialização

Os conceitos de modelo E-R apresentados até agora são suficientes para representar a maioria dos esquemas para aplicações tradicionais de bancos de dados, em sistemas administrativos. Com a evolução dos SGBD e a procura por melhorar a captura semântica foram feitas extensões ao modelo E-R e, entre os quais pode-se citar **generalização e especialização**.

Quando duas entidades possuem semelhanças, elas possuem atributos em comum e, semanticamente estão próximas, então podem fazer parte de uma generalização. Aluno de graduação e de pós-graduação podem ser especializações (subtipos) de Aluno. Aluno está em um nível superior.

Generalização é o contrário de **especialização**. O processo de especialização permite definir um conjunto de subtipos de um conjunto de entidades e associar atributos específicos adicionais para cada um dos subtipos. Generalização, ao invés de adicionar,

suprime as diferenças entre vários conjuntos de entidades, identificando seus aspectos comuns, para generalizá-los num único supertipo.

3.2.5 Agregação

Uma limitação do modelo E-R é que não é possível expressar relacionamentos entre relacionamentos. A solução para isso é usar **agregação**. A agregação é uma abstração através da qual relacionamentos são tratados como entidades de nível superior.

3.2.6 Diagrama entidade-relacionamento

O diagrama entidade-relacionamento consiste dos seguintes símbolos:(Figura 3.2).

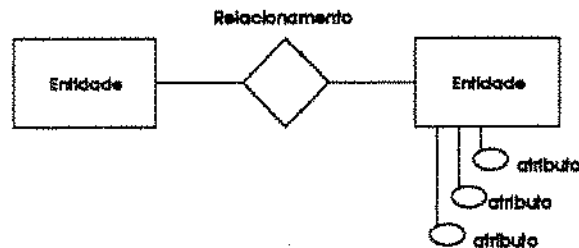


Figura 3.2: Diagrama do modelo E-R.

- retângulos que representam conjunto-entidade;
- elipses que representam atributos;
- losangos que representam conjuntos-relacionamento;
- linhas que ligam atributos a conjunto-entidade e conjuntos-entidade a conjunto-relacionamento.

Um conjunto-relacionamento R pode ser muitos-para-muitos, um-para-muitos ou um-para-um. Para distinguir entre eles deve-se desenhar uma linha direcionada (\rightarrow) ou uma linha não-direcionada (-) entre o conjunto-relacionamento e conjunto-entidade em questão. Uma linha direcionada do conjunto-relacionamento Aluno-Disciplina para o conjunto Aluno especifica que conjunto-entidade Aluno participa ou de um relacionamento um-para-um ou um-para-muitos com o conjunto entidade Disciplina. Uma linha não-direcionada do conjunto-relacionamento Aluno participa de um relacionamento

muitos-para-muitos ou de um relacionamento muitos-para-um com o conjunto-entidade Disciplina(ver Figura 3.3). Outra forma de descrever a cardinalidade é colocar valores próximos das entidades, indicando a sua forma de participação: N para muitos e um para um.

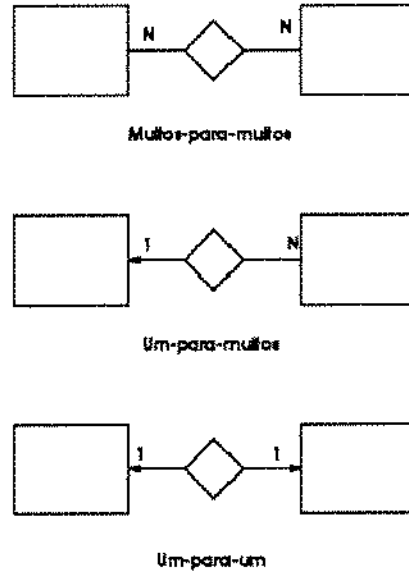


Figura 3.3: Cardinalidade de relacionamentos.

Os papéis são indicados nos diagramas E-R, rotulando as linhas que conectam losangos e retângulos (Figura 3.4).

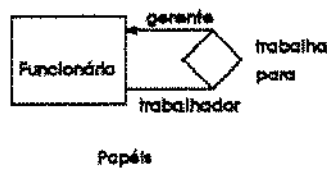


Figura 3.4: Papéis.

Um conjunto entidade fraco é indicado nos diagramas E-R por uma caixa com um duplo contorno (Figura 3.1).

A generalização é representada por um triângulo rotulado “é um”. Agregação é representada conforme ilustrado Figura 3.1.

3.3 Modelo Relacional

Este modelo talvez seja o mais conhecido e o mais implementado [Dat86b] [Ull82] [KS93]. O modelo relacional é baseado em princípios matemáticos de relações e tuplas. Segundo o modelo, um banco de dados consiste em um número de diferentes relações que são essencialmente arquivos do tipo tabela de dados. Cada relação é um conjunto de tabelas, de tipos similares, onde cada tupla é uma linha de dados da tabela. Por exemplo, uma base cadastral pode ter tabelas para lotes de terra, cidade, estrada. Cada linha em uma das tabelas representa um único lote, cidade, estrada (Figura 3.5). O esquema consiste no nome, tipo e tamanho de cada campo da tabela.

Lote de terra	Cidade	Estrada
01	Fortaleza	BR-116
05	Pacajus	BR-116

Figura 3.5: Modelo relacional.

Pode-se considerar um banco de dados relacional como um conjunto de tabelas, logicamente relacionadas entre si. Quando se fala em banco de dados é necessário diferenciar entre o esquema de um banco de dados, ou o projeto lógico de um banco de dados, e uma instância de um banco de dados, que é o dado dentro do banco de dados em um determinado instante. Um esquema nada mais é do que uma lista de atributos e seus domínios correspondentes. Um esquema define os atributos que compõem uma entidade/relacionamento.

Uma linha em uma tabela é chamada de **tupla** de uma relação. Cada tupla é, por sua vez, definida como um conjunto de atributos, que podem ser vistos como as colunas da tabela. Cada entidade ou cada relacionamento entre entidades é representado por uma tupla (um conjunto de valores de uma entidade específica) de uma relação.

Exemplos de esquema para entidade e relacionamento:

```
Aluno_esquema(nome_aluno:cadeia, matricula_aluno:inteiro, endereco_aluno:cadeia)
Curso_esquema(nome_curso:cadeia, codigo_curso:cadeia)
```

```
Aluno_curso(matricula_aluno:inteiro, codigo_curso:cadeia)
```

Neste modelo não há hierarquia de dados e qualquer atributo pode ser usado como chave.

Uma estrutura lógica relacional é definida por um grupo de atributos, onde um atributo é essencialmente uma coluna título na tabela. Todos atributos são **atômicos**, significando que são únicos, ou seja, unidades não passíveis de decomposição. Cada atributo pertence a um domínio que define o tipo de dado (binário, inteiro, real, cadeia, data) e talvez o intervalo de valores permitido para aquele atributo. Ligações entre duas tabelas são permitidas quando elas tem domínios de atributos comuns, por exemplo, uma linha na tabela de cursos está relacionada a uma da tabela de professores, se o atributo de identificação do professor em curso corresponde ao atributo professor-ID da tabela de professor.

Usando o modelo relacional, uma pesquisa pode ser feita sobre qualquer tabela usando qualquer atributo. Pesquisas de atributos relacionados que são armazenados em diferentes tabelas podem ser feitas ligando duas ou mais tabelas usando qualquer atributo que eles tenham em comum. Esta operação denomina-se **junção natural**¹. Como resultado, obtém-se uma tabela, que não necessariamente precisa ser armazenada, chamada **tabela virtual**, que é definida e pode ser consultada. Esta junção dá uma grande flexibilidade ao modelo, sendo capaz de acomodar diversas consultas para as quais o banco de dados não foi especificamente projetado.

As relações não são codificadas explicitamente na base de dados, portanto, o usuário não precisa conhecer a estrutura da base de dados para construir uma consulta.

A maioria dos sistemas de banco de dados relacionais comerciais oferece uma linguagem de consultas que já inclui elementos de abordagem procedurais e não-procedurais. Dentre as principais linguagens destacam-se SQL, QBE e Quel.

3.4 Transformação do modelo E-R para tabelas

Um banco de dados, definido através de um diagrama E-R, pode ser representado por uma coleção de tabelas para cada conjunto-entidade e para cada conjunto-relacionamento. No banco de dados, existe uma única tabela que é designada com o mesmo nome do conjunto-entidade ou do conjunto-relacionamento correspondente.

Uma entidade E é transformada para uma relação R(E). Cada atributo de E, que não seja composto, torna-se um atributo em R(E) e a chave de E torna-se chave primária de R(E). Os atributos compostos podem ser resolvidos de duas formas:

¹join

- usa-se apenas os atributos folhas e ignora-se o atributo composto. Por exemplo : Nome(Primeiro nome, Último nome) - somente primeiro e ultimo nome serão usados na relação.
- transforma-se o atributo composto numa relação separada B. A chave de R(E) é adicionada à relação e é definida como a chave(B).

Um conjunto-entidade ALUNOS com os atributos (nome, matrícula, endereço) é traduzido para uma tabela ALUNOS com uma coluna para cada atributo.

Seja A um conjunto-entidade fraco com atributos $\{a_1, a_2, \dots, a_n\}$. Seja B um conjunto-entidade forte do qual A é dependente. Seja a chave primária de B composta pelos atributos $\{b_1, b_2, \dots, b_s\}$. Representa-se o conjunto-entidade A por uma tabela chamada A com uma coluna para cada atributo do conjunto: $\{a_1, a_2, \dots, a_n\} \cup \{b_1, b_2, \dots, b_s\}$.

Seja R um conjunto-relacionamento binário, ou seja, envolvendo apenas dois conjuntos-entidade. Se o relacionamento é muitos-para-muitos, o conjunto-relacionamento R será representado por uma tabela chamada R, com uma coluna para cada atributo do relacionamento R, se R possuir atributos, bem como para cada atributo que constitua a chave primária de cada conjunto-entidade envolvido.

Se o relacionamento é muitos-para-um o conjunto-relacionamento não será materializado em uma tabela. A chave primária do conjunto-entidade no lado "um" do relacionamento será passado para o outro conjunto-entidade, passando a se chamar neste conjunto de "chave estrangeira". Este conjunto-entidade possuirá uma coluna adicional para cada atributo que componha a chave estrangeira.

Se o relacionamento é um-para-um o conjunto-relacionamento não será materializado. Da mesma forma que um-para-um, a chave primária de um conjunto-entidade será passado para o outro conjunto-entidade e será chamada de "chave estrangeira". O conjunto-entidade que passará a chave pode ser qualquer um dos dois.

Com relação a generalização, existem dois métodos para transformá-la em uma forma tabular:

- cria-se uma tabela para o conjunto-entidade de nível superior. Para cada conjunto-entidade de nível inferior, cria-se uma tabela que inclua uma coluna para cada um dos atributos daquele conjunto-entidade, mais uma coluna para cada atributo da chave primária do conjunto-entidade de nível superior.
- não se cria uma tabela para o conjunto-entidade de nível superior. Em vez disso, para cada conjunto-entidade de nível inferior, cria-se uma tabela que inclua uma coluna para cada um dos atributos daquela entidade e mais uma coluna para cada atributo de conjunto-entidade de nível superior.

3.5 Modelo orientado a objeto

O modelo orientado a objeto representa a confluência de idéias das linguagens de programação orientada a objeto e dos sistemas de gerenciamento de banco de dados. Linguagens orientadas a objeto proporcionam riqueza de dados, incluindo capacidade de modelagem poderosa baseada na habilidade de definir tipos abstratos de dados e construir hierarquias de tipos que permitam herança. SGBD possibilitam armazenamento confiável de dados por longos períodos, acesso multiusuário, controle de concorrência, facilidades para consulta, recuperação e segurança dos dados. Um SGBDOO é a combinação das vantagens das linguagens de programação orientadas a objeto e SGBD.

Essa abordagem foi introduzida por Atkinson et al. em [ABD⁺89]. Desde então, muito já foi desenvolvido na busca de se chegar a um SGBDOO.

O modelo orientado a objeto tem como conceitos principais [JTTW91] [KL89] [Pet87]:

- objetos: utilizados para representar entidades concretas ou abstratas do mundo real no domínio da aplicação sendo modelada;
- classe de objeto: agrupamento de objetos que compartilham os mesmos atributos e mesmo comportamento. Membros de uma classe são chamados de instâncias;
- métodos: operações definidas nas classes;
- encapsulamento: conceito de empacotar um conjunto de métodos de qualquer item de dado, onde somente os métodos serão visíveis ao usuário. Como o usuário não pode fazer suposições sobre a implementação e a representação interna do objeto, as formas de implementação podem ser modificadas de forma transparente para o usuário;
- subclasse: especialização da classe hierarquicamente acima, chamada de pai. Toda instância de uma subclasse é também uma instância da classe pai;
- extensão da classe: conjunto de objetos que estão numa mesma classe;
- identidade de objeto (OID): um objeto existe independente do seu valor. O OID não pode ser alterado da mesma forma que os atributos do objeto. Objetos idênticos possuem o mesmo OID, objetos iguais possuem o mesmo conteúdo. Objetos isomorfos possuem o mesmo grafo de composição;
- herança: na maioria das vezes, é um mecanismo de reusabilidade por compartilhamento de comportamento entre objetos. A idéia chave de herança de classe é prover

um mecanismo simples e poderoso para definição de novas classes, que herdaram propriedades de classes existentes. Herança simples ocorre quando uma subclasse herda variáveis de instância e métodos de uma única classe pai, podendo adicionar métodos e variáveis. Uma extensão natural da herança simples é a herança múltipla, quando uma subclasse herda variáveis e métodos de múltiplas classes pais;

- sobrecarga: redefinição uma operação já definida na classe pai. Neste caso há uma sobrecarga do nome da operação, que irá variar dependendo da classe;
- mensagem: solicitação externa aos objetos.

Em SGBD orientado a objeto tudo é considerado como objeto. Objetos podem ser tão simples e pequenos como números e tão grandes quanto aviões. A todo objeto, independente de tamanho e complexidade, está associado um estado e um comportamento.

O estado é definido pelos valores dos atributos do objeto (também chamados de propriedades ou variáveis de instância). O comportamento é definido pelos métodos que atuam sobre o objeto. Um método consiste de assinatura e corpo. A assinatura especifica o nome do método, os nomes e as classes dos parâmetros e a classe do resultado. O conjunto de assinaturas dos métodos de uma classe representa a interface pública dos objetos daquela classe. O corpo representa a implementação do método e consiste de um conjunto de instruções expressas em uma linguagem de programação.

3.6 OMT

Este método foi proposto por Rumbaugh et al. em [RBP⁺91] e procura estender notações, bastante difundidas na área de desenvolvimento de software, para capturar os conceitos de orientação a objeto. O método OMT usa três tipos de modelos para descrever um sistema:

- modelo de objetos - descreve os objetos no sistema e o seus relacionamentos;
- modelo dinâmico - descreve as interações entre os objetos do sistema;
- modelo funcional - descreve as transformações dos dados do sistema.

Os três modelos são partes ortogonais e complementares de um sistema.

A notação usada é a seguinte:

- diagrama de objetos e classes - representa a estrutura estática do sistema;

- diagrama de estados - representa o aspecto dinâmico do sistema.
- diagrama de fluxo de dados - representa os aspectos funcionais do sistema;
- diagrama de arquitetura do sistema - captura o relacionamento estático entre os subsistemas.

Um sistema incorpora os três aspectos: ele usa as estruturas de dados (modelo de objetos), ele serializa operações no tempo (modelo dinâmico) e ele transforma valores (modelo funcional).

Neste trabalho só será descrito o modelo de objetos.

3.6.1 Modelo de objetos

O modelo de objetos descreve a estrutura estática dos objetos no sistema: sua identificação, seu relacionamento com outros objetos, seus atributos e suas operações no mundo real. A sua notação consiste em um grafo onde os nós são classes de objetos e os arcos representam relacionamentos entre esses objetos.

Objetos, classes e atributos

A representação gráfica de classes e objetos pode ser observada na fig.3.6.

Um **objeto** é simplesmente algo que faz sentido no contexto de uma aplicação. Objetos facilitam o entendimento do mundo real e provêem uma base prática para implementação em computador.

Todo objeto possui uma identidade única. Não existem dois objetos com a mesma identidade.

Uma **classe** descreve um grupo de objetos com propriedades similares (atributos), comportamento comum (operações), relacionamentos comuns a outros objetos e semântica em comum. O diagrama de objeto é a representação gráfica formal do modelo de objetos. Um diagrama de classe é um esquema, modelo para descrição de instâncias de dados. Um diagrama de classes define classes. Um diagrama de instância descreve como um conjunto particular de objetos que se relacionam entre si. A notação OMT para instância é uma caixa com cantos arredondados. O símbolo OMT para classe é uma caixa como o nome da classe em negrito.

Um **atributo** é um valor de dados suportado pelos objetos em uma classe. Atributos são colocados na segunda parte da caixa de classe. Cada nome de atributo pode ser seguido de detalhes opcionais, como tipo e valor-padrão.

Uma **operação** é uma função ou transformação aplicada a/por objetos em uma classe. Todos os objetos em uma classe compartilham as mesmas operações. Operações podem ser aplicadas a muitas classes diferentes. Tal operação é dita polimórfica, isto é, a mesma operação pode ter diferentes implementações em diferentes classes. Um método é a implementação de uma operação para uma classe.

Operações são listadas no terceiro quadro da caixa de classe. Cada nome de operação pode ser seguido por detalhes opcionais, tais como lista de argumentos e tipo de resultado (Figura 3.6).

Um elo² é uma conexão física ou conceitual entre instâncias de objeto. Uma associação descreve um grupo de elos com estrutura e semântica comuns. Um elo é uma instância de uma associação. A notação OMT para uma associação é uma linha entre as classes. Os nomes da associação são escritos em itálico.

Uma instância de uma classe pode estar associada a várias instâncias de uma classe associada. Cardinalidade é freqüentemente descrita como “um” ou “muitos”, mas é possível ter um subconjunto de inteiros não-negativos. Existem símbolos de término de linha para indicar valores de cardinalidade. Uma bola cheia é o símbolo para “muitos”, significando zero ou mais. Uma bola não preenchida indica “opcional”, significando zero ou um. Pode-se ter também valores numéricos especificando a cardinalidade do relacionamento. Uma linha sem símbolos indica associação um-a-um.

Quando há hierarquia, as classes podem ser organizadas em hierarquias.

As associações podem ser de vários tipos: agregação, ternária, ou associação com classe. (ver fig. 3.6).

Um **atributo** é uma propriedade dos objetos em uma classe. Da mesma forma, um atributo de um elo é uma propriedade dos elos de uma associação. A notação OMT para atributo de elo é uma caixa ligada a associação por um laço. Um ou mais atributos podem aparecer na segunda região da caixa. Uma associação pode ser modelada como uma classe, isto é, pode ser útil quando elos participam de associações com outros objetos ou quando elos são sujeitos a operações.

Papéis

Um papel é um nome que identifica um lado de uma associação. Papéis permitem um modo de ver uma associação binária como um caminho de um objeto para um conjunto de objetos associados. O papel distingue os lados da associação e se faz necessário se uma classe participa de uma associação de cardinalidade N mais de uma vez.

²link

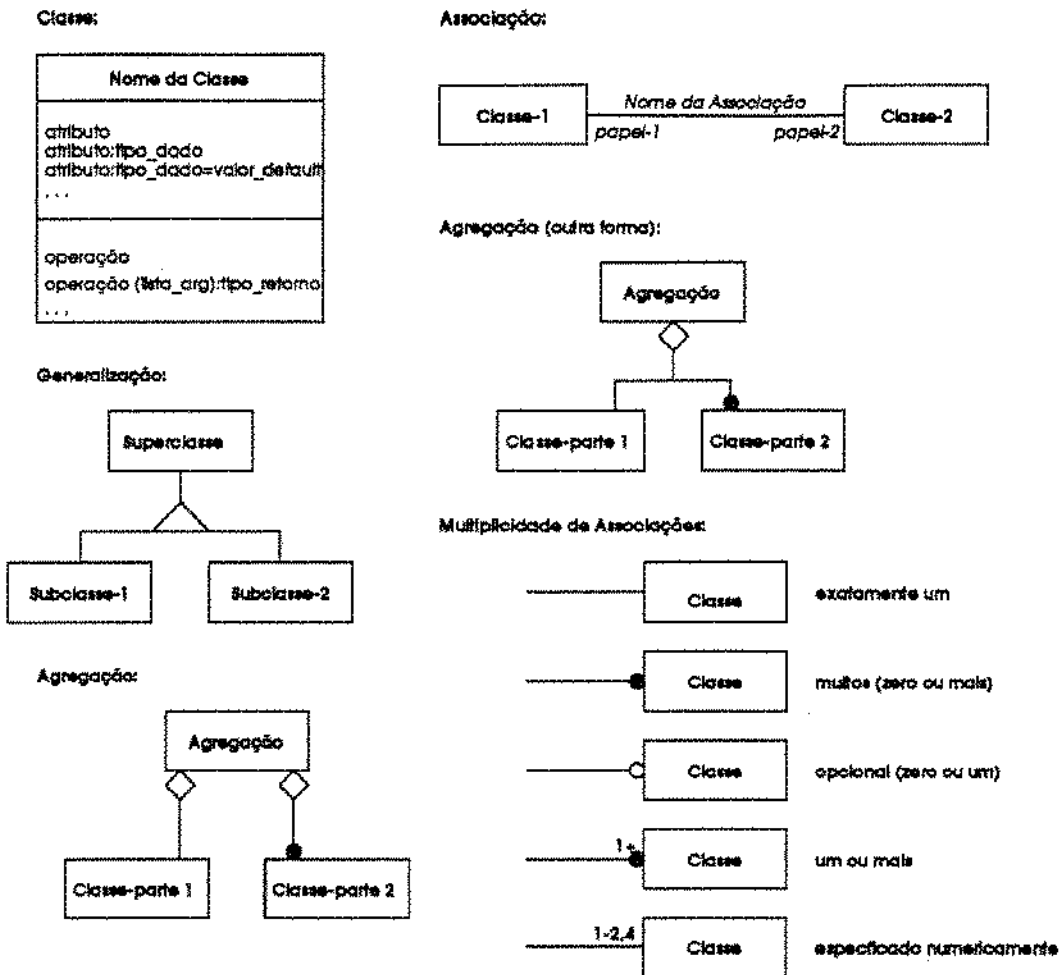


Figura 3.6: Modelo objeto.

Agregação

É um relacionamento do tipo “é uma parte de”, no qual objetos componentes são associados a um objeto que representa a montagem inteira. Define-se agregação como uma relação de uma classe de montagem da classe(s) componente(s). A agregação é representada como associação, exceto por um pequeno losango indicando a classe de montagem do relacionamento.

Generalização e herança

Generalização e herança são mecanismos poderosos para compartilhamento de similaridades entre classes, preservando suas diferenças. Generalização é o relacionamento entre uma classe e uma ou mais versões refinadas dele. A classe sendo refinada é chamada

de *superclasse* e cada versão refinada é chamada de *subclasse*. Atributos e operações comuns a um grupo de subclasses são ligadas a superclasse e compartilhados para cada subclasse. Cada subclasse *herda* as características de sua superclasse. Generalização é um relacionamento do tipo “é um” porque cada instância da subclasse é uma instância da superclasse.

A notação para generalização é um triângulo conectando uma superclasse a suas subclasses. A superclasse está conectado por uma linha ao ápice do triângulo. As subclasses são conectadas por linhas a uma barra horizontal ligada a base do triângulo.

A generalização é um construtor útil para modelagem conceitual e implementação, facilitando modelagem, estruturando classes e capturando semelhanças. A herança de operações é útil durante implementação como um veículo para reutilização de código.

Uma subclasse pode redefinir uma característica (atributo ou operação) de uma superclasse pela definição de uma característica com o mesmo nome. A característica que está se sobrepondo refina e substitui a característica anterior.

Pode-se redefinir valores padrão de atributos e métodos das operações. Uma redefinição deve preservar tipo do atributo, número e tipo de argumentos para uma operação e o tipo do retorno da operação.

3.7 Modelagem de dados em SIG

Um SIG deve representar fenômenos espaciais através da criação de um modelo que permita ao usuário manipular o fenômeno armazenado no sistema [RM92]. Fenômenos espaciais variam de acordo com o contexto do estudo: podem ser lotes de terra, porções de árvores em uma floresta, redes de duto para água e esgoto, cada um com características próprias.

A modelagem de dados em SIG é governada pelas estruturas disponíveis para representar os fenômenos espaciais. Muito da história de SIG está relacionada ao modo com que os modelos de dados são implementados em termos de software. Com o crescente desenvolvimento do hardware, tornou-se possível desenvolver sistemas com representações mais ricas, e diversas propostas diferentes proliferaram.

Dados geográficos devem ser considerados com respeito ao local ao que se referem (coordenadas geográficas). A modelagem de dados georreferenciados é percebida ou como campos ou como objetos. Estes modelos são representados como vetor ou como raster [MP94].

A visão de campos entende o mundo como uma superfície contínua, sobre a qual características variam em uma distribuição contínua. A superfície pode ser modelada

em camadas. Cada camada³ corresponde a um tema diferente. Dados de campo são representados no formato tesserar (unidades poligonais do espaço, células em uma matriz), onde cada célula tem um valor temático. Células podem ter diferentes formas. Células quadradas são chamadas de pixels.

A visão por objetos trata o mundo como uma superfície com objetos, que existem independente de qualquer definição e que podem ocupar um mesmo local. Dados do tipo objeto são representados como pontos, linhas e polígonos, usando listas de pares de coordenadas. Esta é a chamada representação vetorial. Redes são um caso especial de dados vetor, onde elementos são conjunto de arcos e nós.

A escolha da representação dos dados é importante. O propósito da modelagem de dados é realizar um projeto de um banco de dados que capture a semântica da aplicação, seja fácil de preencher com dados, estender e simples de entender.

Tendo em vista que será feita implementação em SGBD relacional e orientado a objeto e como há um consenso com relação às características de um SGBD relacional optou-se por não descrever nenhum SGBD específico. Como os conceitos de OO ainda estão em consolidação e não existe, até o momento, uma padronização, será descrito o SGBD orientado a objeto a ser utilizado, no caso o O₂.

3.8 Linguagem de consulta

A linguagem de consulta é parte integrante de um SGBD, dando condições aos usuários de interagir com ele. Deve ser simples de aprender e usar, a fim de facilitar a tarefa de formular consultas, e ainda deve ser poderosa o bastante para ser útil.

Os tipos de consultas suportados pelos modelos hierárquicos e em rede são definidos quando o banco de dados é construído. Ligações físicas embutidas nos registros de dados são utilizados para percorrer o banco de dados. Linguagens que requerem o conhecimento da hierarquia são denominados *linguagens de consulta procedurais*.

No modelo relacional, mais flexibilidade é alcançada pela abolição da hierarquia de atributos. Diferente do que ocorre nos modelos hierárquico e em rede, no relacional as relações não são explicitamente codificadas no banco de dados. O usuário não precisa conhecer a estrutura de dados para construir a consulta. Uma linguagem de consulta que não dependente da estrutura do banco de dados é denominada *linguagem não procedural*. A linguagem SQL (Structured Query Language), desenvolvida pela IBM, é largamente utilizada.

Embora originalmente desenvolvida para SGBD relacionais, linguagens de consulta

³layer

não procedural são também disponíveis para sistema em redes.

Comumente utiliza-se para SIG a linguagem de consulta do SGBD sobre o qual foi implementado, com algumas extensões. Como SIG são em geral escritos sobre SGBD relacionais, a linguagem muitas vezes utiliza SQL como base. Mesmo com algumas extensões o SQL apresenta restrições quanto à manipulação de objetos complexos.

Uma das dificuldades inicialmente encontradas para desenvolver aplicações sobre SGBD extensíveis foi a falta de uma padronização dos mecanismos para definição e consulta. Devido a isto, vários fabricantes de SGBD orientado a objeto (O₂, Object Design, Objectivity, Ontos e Versant) formaram o ODMG (Object Data Management Group), que publicou o padrão ODMG'93 [BF94] [Cam95] [Kim95]. Este padrão inclui uma definição de uma linguagem de definição de objetos (ODL - Object Definition Language) e de uma linguagem de consultas a objetos (OQL - Object Query Language). O padrão ODMG garante que uma aplicação escrita em um sistema X possa ser facilmente transferida para um sistema Y.

Todos esses fabricantes se comprometem a cumprir os padrões a partir de versões futuras de seus produtos. A versão atual do O₂ segue a linguagem de consulta ODMG OQL [BF94].

O modelo objeto proposto pela OMG (Object Management Group) suporta a noção de classe, de objetos com métodos e atributos, herança e especialização. O modelo objeto recebeu duas extensões do ODMG: relacionamentos uma-um através do uso de referência *ref*, que se comporta como um ponteiro C++; e coleções. ODMG-93 introduz um conjunto de classes genéricas: *set*< T >, *bag*< T >, *varray*< T >, *lst*< T >.

ODMG-93 introduz uma linguagem de consulta OQL. OQL é um superconjunto da parte de consulta do SQL'92. Ele permite o acesso a objetos. OQL, diferentemente de SQL, pode manipular objetos complexos. Para construir objetos complexos utiliza os construtos *struct*, *set*, *bag*, *list* e *array*.

3.9 Banco de dados para SIG

Os SIG até bem pouco tempo eram caracterizados por serem produtos proprietários, ou seja, um produto desenvolvido por determinadas empresas sem que se conhecesse nada da sua estruturação interna e do modo de funcionamento.

Os SGBD sempre serviram para o armazenamento de dados tradicionais, onde se situam os dados não espaciais de SIG. O desenvolvimento de sistemas para tratar informação espacial é recente. A informação espacial é mais complexa de armazenar e manusear do que os dados não espaciais. Aplicando os conceitos de SGBD para SIG

temos:

- as visões de dados são independentes do modo com que os dados são armazenados. Por exemplo, ao invés de armazenar diferentes mapas, os dados que descrevem os elementos geográficos (informação espacial e não espacial) são armazenados com redundância mínima, e mapas ou outros tipos de saída são gerados à medida que forem solicitados e na forma mais adequada para análise específica;
- é possível a atualização automática de arquivos de dados interrelacionados;
- os relacionamentos entre toda informação espacial e não espacial são definidos explicitamente. Chaves são utilizadas para relacionar a informação de atributos a características espaciais correspondentes e topologia é utilizada para relacionar todos os elementos espaciais um com o outro.
- o controle central do SGBD permite melhor controle da integridade da base de dados por meio de segurança e verificação de consistência para prevenir mau emprego ou degradação da informação.

Ainda assim, os SGBD de maneira geral têm limitações para aplicações de SIG. O SGBD mais utilizado tem sido o relacional, por suas facilidades em tratar dados não espaciais e por ser bem adaptado à natureza imprevisível da análise de SIG. O problema reside nos dados espaciais.

Informação espacial é mais complexa e as transações executadas sobre ela são mais complicadas. Uma única mudança/alteração pode envolver a alteração simultânea de grande número de registros e arquivos. As principais dificuldades são:

- os registros de dados espaciais usados em SIG são de tamanho variável para permitir o armazenamento de números variáveis de pontos de coordenadas, visto que SGBD de propósito geral são projetados para tratar registros de tamanho fixo;
- a manipulação de dados geográficos envolve conceitos espaciais, tais como proximidade, conectividade, inclusão e cobertura, que não são facilmente resolvidos por linguagens de consulta de SGBD de propósito geral;
- um SIG requer capacidade gráfica que não é normalmente suportada por SGBD;
- a natureza altamente interrelacionada de dados de SIG requer um sistema de segurança mais sofisticado do que o bloqueio de registro usado pelos SGBD de propósito geral.

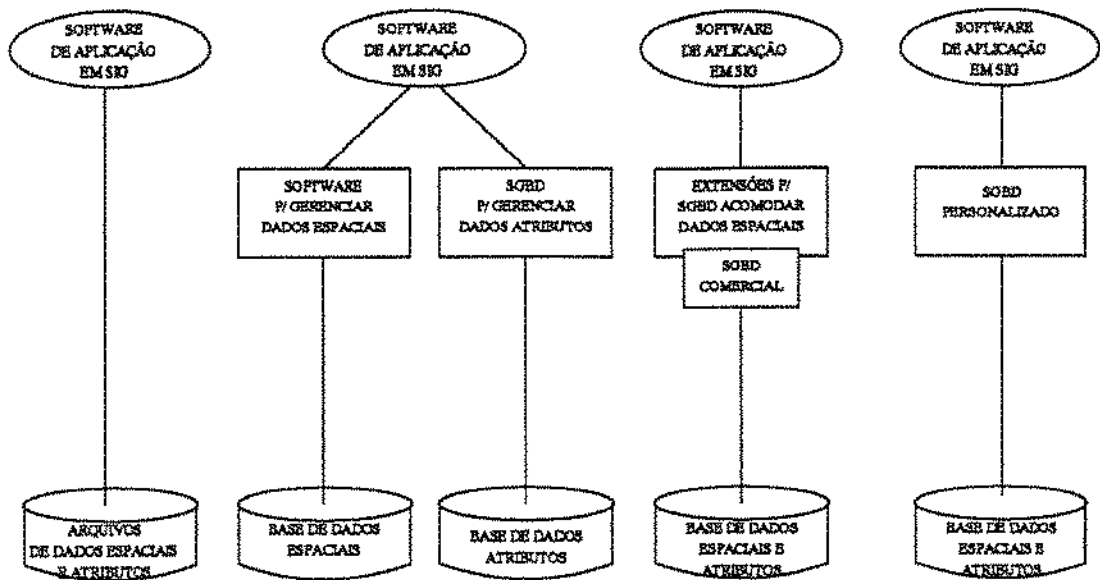


Figura 3.7: Propostas para SIG.

Nos últimos anos várias propostas foram sugeridas para permitir um melhor gerenciamento dos dados de SIG. Pode-se citar (ver fig. 3.7):

- desenvolver um sistema proprietário provendo serviços de gerenciamento de dados solicitados pelos módulos de aplicações diferentes. Esta é a proposta de processamento de arquivos tradicionais;
- desenvolver um sistema híbrido usando um SGBD disponível comercialmente para armazenamento dos atributos não espaciais. Desenvolver em separado um módulo para gerenciamento, armazenamento e análise de dados espaciais, usando os serviços do SGBD para acessar os dados do tipo atributo não espacial;
- usar o SGBD, usualmente relacional, como núcleo do SIG. A partir dele desenvolver extensões para o sistema quando necessário. Uma quantidade significativa de software é geralmente adicionada ao SGBD para prover funções espaciais e visualização gráfica usada na análise gráfica;
- começar do zero e desenvolver uma base de dados espaciais capaz de tratar os dados espaciais e não espaciais de uma forma integrada.

Na primeira categoria, encontram-se a maioria dos SIG antigos. Os SIG comerciais utilizam as outras três propostas.

Exemplos de SGBD extensíveis são os sistemas chamados de pós-relacionais, como o POSTGRES, STARBURST, O₂ e OBJECTSTORE [Cam95].

3.9.1 Linguagem de consulta

Para ter acesso aos atributos de dados geográficos, a maior parte das implementações de linguagem de consulta utiliza o padrão SQL. Entretanto, não é possível lidar com informações topológicas através do SQL, o que tem levado ao surgimento de muitas propostas de linguagem de consulta espacial [Ege94]. Geralmente essas propostas são extensões do SQL.

A proposta do novo padrão para a linguagem SQL (SQL 3) contém um conjunto de extensões para lidar com dados espaciais. Inclui tipos de dados para representar dados espaço-temporais e dados geométricos bem como operadores topológicos [Cam95].

SQL 3 pode ser descrito como [Mel94]:

- a primitiva básica de modelagem é o objeto;
- objetos podem ser reunidos em tipos. Objetos de um mesmo tipo exibem comportamento comum e estados comuns;
- o comportamento de objetos é definido por um conjunto de operações, que podem ser executadas sobre um objeto daquele tipo;
- o estado dos objetos é definido pelos valores que eles carregam para um conjunto de propriedades. Estas propriedades podem ser atributos do próprio objeto ou relacionamentos entre o objeto e um ou mais objetos;

SQL 3 procura incorporar todas as características de orientação a objeto, sem perder as suas características relacionais. Objetos são considerados como ADT, tipos abstratos de dados, definidos pelo usuário, com estrutura e métodos, semelhantes a C++. Só instâncias do ADT são objetos. Isso permite que se possa a continuar a dar suporte ao modelo relacional.

Com relação a SIG, as características geométricas dos dados espaciais podem ser definidos na estrutura e métodos dos ADT. Cabe ao usuário esta tarefa.

3.10 SGBD O₂

Baseado no OMG, SQL, C++ e Smalltalk foi desenvolvido o O₂ [BF94] [Tec93]. O sistema O₂ [Deu91] é um SGBD orientado a objeto com um ambiente de desenvolvimento completo e um conjunto de ferramentas de interface com o usuário. O O₂ consiste de um núcleo central (O₂Engine) que cuida do gerenciamento de disco, distribuição, gerenciamento de transação, concorrência, recuperação, segurança e administração dos dados.

O₂Engine suporta interface com as linguagens C e C++ e o ambiente O₂. Este ambiente oferece:

- uma linguagem de consulta O₂Query;
- um gerador de interfaces O₂Look;
- uma linguagem orientada a objeto O₂C;
- um ambiente de programação gráfica, incluindo um depurador e consulta para banco de dados e esquemas.

3.10.1 Modelo de dados

No O₂, um sistema consiste de um conjunto de esquemas e bancos de dados. Um esquema define tipos de dados, enquanto um banco de dados contém os dados. O₂ manuseia valores e objetos. Um valor tem um tipo. Este tipo é definido recursivamente, a partir de tipos atômicos e construtores de tipo. Um objeto tem uma identidade, um valor e um comportamento definido por seus métodos. Ele pertence a uma classe.

Os tipos atômicos são lógico, caracter, inteiro, real, cadeia⁴ e bits. Tipos complexos podem ser definidos recursivamente usando construtores *tuple*, *list*, *set* e *unique set*. *List* é uma coleção ordenada, cujos elementos são acessíveis através de um índice. Comporta-se como um vetor, isto é, pode ter elementos adicionados e uma sublista pode ser inserida ou removida da lista. O tipo cadeia se comporta como uma lista de caracteres, enquanto um valor de bits age como uma lista de bytes. *Set* é uma coleção sem ordem.

Uma classe é definida pelo seu tipo e seus métodos. A implementação é separada de sua especificação dentro de uma classe. Uma classe pode herdar seu tipo e métodos de outras classes. Colisões de nomes de métodos ou de atributos são solucionados através de renomeação.

Para se tornar persistente, um objeto ou valor deve ser fixado diretamente ou transitivamente a uma raiz persistente. Essas raízes são declaradas no esquema dando-se um nome a elas.

O₂ provê encapsulamento em três níveis: classe, esquema e banco de dados. A primeira forma é clássica em modelos orientado a objeto. Um atributo é *privativo* a sua classe, por definição, mas pode ser acessível fora da classe como um atributo público ou só para leitura⁵. A segunda forma estende essa noção a um conjunto de definições de classe, isto é, um esquema. Um esquema pode exportar algumas classes permitindo que outros

⁴string

⁵read-only

esquemas as reutilizem (importem). A última forma permite encapsulamento de dados. Uma aplicação executando sobre uma base de dados particular pode acessar uma outra base chamando um método que executará sobre essa base “remota”.

3.10.2 Linguagem de consulta

O₂QL é uma linguagem de consulta do tipo SQL estendida para lidar com valores e objetos complexos. É um subconjunto do O₂C, mas pode ser usada independente como uma linguagem de consulta interativa ou como uma função chamada de C ou C++.

A criação de classes C++ pode ser feita através de comandos, permitindo a manipulação dos correspondentes dados em O₂.

3.10.3 O₂Look, O₂Tools

O gerador de interface com o usuário O₂Look suporta o *display* e manipulação de objetos grandes, complexos e multimídia na tela. Junto com O₂C, O₂Look constitui um poderoso gerador de aplicações.

O₂Tools é um ambiente gráfico de programação de suporte ao desenvolvimento de aplicações O₂. Permite ao programador navegar⁶, editar e consultar dados e esquema, e editar, testar e depurar métodos e programas. Provê também ferramentas que simplificam o trabalho do programador.

O O₂System armazena todas as informações (dados, métodos, programas e aplicações) como objetos. O₂Tools também representa as informações como objetos (classes, métodos).

O₂ é tanto sistema de banco de dados como um sistema orientado a objeto. Como sistema de banco de dados, deve prover suporte para acessar e alterar grandes quantidades de dados compartilhados, seguros e confiáveis. Como sistema orientado a objeto, deve suportar todas as características do modelo de dados orientado a objeto como objetos complexos com identidade, classes, tipos, encapsulamento, métodos, herança múltipla, sobrecarga e ligação tardia de métodos⁷.

⁶browse

⁷dynamic link

3.11 A utilização de orientação a objeto em SIG

Com o surgimento do modelo orientado a objeto começaram a ser desenvolvidas atividades relacionadas com os novos conceitos na busca de se criar um SGBD puramente orientado a objeto e, ao mesmo tempo, houve tentativas de se estender os SGBD convencionais, a fim de se criar uma camada orientada a objeto [Oli92]. A segunda alternativa caracteriza-se por ser uma alternativa de baixo custo e, até certo ponto, efetiva, no sentido de melhor capturar a semântica e o aprimoramento da relação entre usuários e SGBD. Isto começou, de forma reduzida, no sistema POSTGRES, que implementou mecanismos de abstração de dados, herança, entre outros.

Alguns SIG foram desenvolvidos a partir desses SGBD relacionais, que possuíam uma camada superior que dava suporte ao conceito de orientação a objeto.

O modelo orientado a objeto possibilita um melhor atendimento às necessidades de SIG. Isto se deduz a partir do próprio modelo orientado a objeto, que através do conceito de abstração permite a solução de problemas mais complexos, como os elementos em SIG. Por isto, o enfoque orientado a objeto destina-se justamente a aplicações não atendidas pelos outros modelos.

As vantagens inerentes ao modelo orientado a objeto para desenvolvimento de SIG, entre outras, são:

1. suporta o conceito de abstração de dados em vários níveis;
2. uma representação objeto permite especializações, que também não é disponível no modelo relacional;
3. fácil captura de restrições de integridade nos dados;
4. permite representação, via classes, de objetos mutuamente exclusivos e sobrepostos.

As diferenças entre relacional e orientado a objeto dizem respeito a:

- extensibilidade de classe. No relacional há um único tipo parametrizado: relação. Operações se limitam a ler e escrever valores. No orientado a objeto há conceito de classes, definidos pelo usuário e com operações definidas;
- abstração. O comportamento de um objeto é descrito na classe.
- capacidade de armazenar dados ativos. Todas as operações (métodos) são gravadas no banco de dados e podem ser chamados a qualquer momento. Esses métodos são expressos em linguagem de programação de propósito geral e qualquer operação pode ser construída e armazenada no banco de dados;

- *gap* semântico. O modelo orientado a objeto alcança uma compreensão dos objetos armazenados, a nível do usuário, bem maior do que a conseguida pelo relacional.

3.12 Alguns SIG prototipais e comerciais

Vários SIG foram desenvolvidos seguindo as propostas apresentadas na seção anterior. Nesta seção será feito uma pequena descrição de alguns SIG mais conhecidos que utilizam de alguma forma um SGBD.

A análise será dividida em protótipos e produtos existentes no mercado. Os protótipos são basicamente desenvolvimentos de universidades e centros de pesquisa. Entre os protótipos a serem analisados estão: PROBE, TIGRIS e SIRO-DBMS. Os produtos comerciais analisados são: ARC/INFO, VISION. Foram analisados dois SGBD voltados para aplicações não convencionais: STARBURST e EXODUS.

3.12.1 PROBE

O PROBE [MO86] [Ooi91] [Feu93] é um projeto de pesquisa de um SGBD orientado a objeto, voltado para aplicações não tradicionais, muitas das quais envolve dados espaciais e dados com estrutura complexa.

O objetivo é prover um sistema de banco de dados de propósito geral para aplicações envolvendo dados temporais e espaciais e outros tipos de dados com estrutura complexa.

O modelo de dados do PROBE tem dois tipos básicos de objetos: entidades e funções. Uma entidade é um objeto que possui identidade. Entidades com características similares são agrupadas em coleções chamadas tipos de entidade. Função é geralmente definida como um relacionamento entre coleções de entidades e valores escalares. Existem duas classes genéricas de funções: funções computadas, com valores de saída calculadas por procedimentos, e funções armazenadas, com valores de saída determinados por uma pesquisa convencional de banco de dados de uma função armazenada *extend*. Tipos de entidade podem ser divididos em subtipos, formando hierarquias de generalização. No topo da hierarquia, ambas entidades e funções são membros do tipo genérico ENTIDADE. Operações genéricas sobre objetos (entidades e funções) como seleção, aplicação de função, operações de conjunto e formação de novas funções estendidas, foram definidas na forma de uma álgebra bem similar em alguns aspectos a álgebra definida para o modelo relacional.

A arquitetura do SGBD PROBE consiste em um núcleo que trata conjuntos de objetos genéricos, enquanto as classes de objeto tratam objetos individuais de tipos especializados.

Os objetos básicos no modelo de dados espacial são conjuntos de pontos (*point sets*).

Point set de um objeto é o conjunto de pontos no espaço ocupado por aquele objeto. Um conjunto de pontos é um membro do tipo CONJUNTO-DE-PONTOS. Este tipo é uma especialização do tipo ENTIDADE que introduz uma coleção de operadores especiais como união, interseção, diferença, bem como, predicados especiais como sobreposição, está contido e proximidade. Conjunto de pontos exibe uma estrutura hierárquica, pois um conjunto de pontos contido em um espaço pode conter outros conjuntos de pontos.

PROBE dispõe de um filtro geométrico para otimizar consultas espaciais. Como estruturas do tipo R-trees não são usadas, a maioria das operações geométricas são do tipo laço aninhado (*nested loop*). A função deste filtro é otimizar laços. O problema com algoritmos com laços aninhados é desempenho. Os laços aninhados levam algoritmos de tempo polinomial. A saída produzida do filtro é um conjunto de objetos candidatos (ou conjunto de grupos de objetos - um membro do grupo vem de cada laço). Eles são prováveis de satisfazer a consulta. O conjunto de candidatos será refinado para produzir a resposta precisa aplicando a cada candidato um predicado determinado da classe de objeto espacial suprido.

Uma versão estendida do DAPLEX é utilizada como linguagem de consulta.

Um protótipo do PROBE está em experiência [Ooi91].

Para implementar um modelo de dados de SIG sobre o PROBE ter-se-ia facilidades derivadas de construtores já existentes na linguagem, como conjuntos-de-pontos, que já tem embutido operadores espaciais (união interseção e diferença). Permite também hierarquias de generalização, podendo-se definir tipos genéricos e a partir desses ir refinando sucessivamente em subtipos mais especializados.

3.12.2 ARC/INFO

O ARC-INFO é um sistema genérico comercial para modelagem e análise de dados espaciais armazenados, desenvolvido pela ESRI (Environment Systems Research Institute). ARC refere-se às estruturas de dados topológicos e algoritmos. INFO refere-se ao modelo de dados compostos e processos associados.

Dados não espaciais são suportados por SGBD e dados espaciais suportados por um sistema adicional, construído especificamente para isso [Mor92] [Aro89] [Ooi91].

O modelo de dados, georrelacional, combina uma visão geográfica especializada dos dados com um modelo de dados relacional convencional. Dados de localização e temáticos são representados como conjuntos de tabela.

O modelo de dados do ARC/INFO é baseado na idéia que dados geográficos podem ser representados usando um conjunto genérico de *features*: ponto, linhas, áreas, superfícies, etc. Cada feature genérico tem associado informação de localização e temática. As featu-

res definidas pelo usuário, em termos de características genéricas, é feita pela associação de identificadores de feature como atributos de features genéricos.

No ARC/INFO dados de localização são representados usando um modelo de dados topológico. Dados temáticos usam o modelo relacional.

O sistema define atributos de local e temáticos das características de mapa em uma dada área. Uma cobertura é definida com um conjunto de características, onde cada característica tem um local (definida pelas coordenadas e ponteiros topológicos para outras características) e, possivelmente, atributos (definidos com um conjunto de itens ou variáveis).

Existem vários tipos de features que podem estar presentes em uma cobertura. Cada uma dessas classes de features pode ter associado uma informação de local e temático. Essas features podem ser arcos, nós, polígonos. A cada feature podem estar associadas tabelas de atributo. No ARC/INFO as linhas da tabela são denominados registros e as colunas são denominados itens.

O ARC/INFO possui um conjunto extenso de operadores para a cobertura: sobreposição, geração de polígono de Thiessen, interpolação de contorno, transformação de coordenadas.

O ARC/INFO possui uma biblioteca de mapas, que consiste em um dispositivo que armazena as coberturas. As coberturas são armazenadas simultaneamente em duas dimensões: por assunto ou conteúdo dentro de camadas e por local dentro de *tiles*. A área geográfica representada por um mapa é dividida em um conjunto de *tiles* não sobrepostos. Uma camada é um tipo de cobertura dentro da biblioteca. Todos os dados em uma mesma camada têm as mesmas características de cobertura e atributos.

O banco de dados ARC/INFO é implementado utilizando técnicas de modelagem de banco de dados relacional. Uma cobertura é definida por um conjunto de relações. Essas relações definem valores geométricos, topológicos e de atributos das várias características na cobertura. Além do SGBD INFO, existem implementações utilizando os SGBD relacionais ORACLE e INGRES.

ARC/INFO é um sistema híbrido, ou seja, a parte espacial é armazenada em estruturas situadas fora do SGBD. A modelagem de dados pode ser considerada mais simples do que a realizada no PROBE, pois a parte não-espacial pode ser modelada via relações (tabelas), bastante conhecida.

3.12.3 TIGRIS

O sistema TIGRIS [Her92] é um SIG comercial. Tanto o TIGRIS quanto o seu descendente DYNAMO são implementados com o paradigma de orientação a objeto. O modelo

sobre o qual TIGRIS é construído é uma combinação de vários modelos bem conhecidos: o modelo relacional, o modelo relacional estendido e o orientado a objeto. O modelo OO como utilizado no TIGRIS é uma extensão do modelo relacional padrão (RM/T), introduzido por Codd em 1979. Na implementação do TIGRIS, podemos ver classes de objeto como uma tabela primária no modelo relacional, cada objeto correspondendo a uma tupla dentro daquela tabela e cada dado da instância correspondendo a um atributo da tabela. Relacionamentos podem ser vistos como tabelas separadas com chaves compostas consistindo de identificadores de objeto como chaves estrangeiras para tabelas descritivas.

Dentro do TIGRIS características (*features*) podem ser qualquer item de interesse para o usuário como estradas, rios, cidades. Além disso, cada característica pode ser de qualquer complexidade e ter relacionamentos explícitos com outras características.

A estrutura dos dados que o usuário vê pode ser quebrada em três categorias: temas, características compostas e características base. Um tema é uma cobertura completa de uma única característica em um conjunto de características relacionadas da superfície da terra.

As características de base podem ser representações de entidades geográficas separadas ocorrendo na superfície da Terra. As características de base contêm somente aqueles valores de atributo que são precisos sobre sua extensão. Atributos que são precisos para somente uma parte da característica implica que a característica deve ser subdividida.

TIGRIS contém um modelo topológico bidimensional consistindo de pontos (nós), curvas entre esses pontos (arestas) e áreas conectadas limitadas por essas curvas (faces). Todas as relações topológicas entre esses elementos são ou armazenadas explicitamente ou podem ser derivadas a partir de relacionamentos armazenados explicitamente.

O modelo topológico é construído e mantido na base de coordenadas (x,y). Cada coordenada (x,y) no mapa será associada somente a um objeto topológico, embora uma posição possa ser usada para definir mais de um objeto topológico.

O TIGRIS utiliza SQL com extensões. O SQL padrão opera sobre relações armazenadas como tabelas de tuplas. No TIGRIS, as tuplas consistem em identidades de objeto. As extensões do O-SQL (Object-SQL) incluem:

- a capacidade para chamar qualquer método de qualquer objeto diretamente da linguagem de consulta para uso em predicado em *selects*, em operadores de relação, ou para alterações.
- a capacidade para definir marcos que auxiliem a interpretação de declarações de consulta de acordo com o contexto da classe.
- a capacidade de agrupar declarações de consulta juntas em unidades únicas para

execução, com comunicação interconsulta via relações temporais gerenciadas pelo sistema.

- a capacidade para armazenar, acessar, e consultar tipos de dados abstratos complexos dentro dos atributos.

3.12.4 SIRO-DBMS

SIRO-DBMS (Spatial Information in Relational Open-architecture DataBase Management System) [Ooi91] [Abe89] é um *toolkit* de banco de dados geográficos que provê alguns tipos de dados espaciais e métodos de acesso espaciais como acessórios (extensões) ao núcleo de um SGBD relacional. Foi desenvolvido no Centro para Sistemas de Informações Espaciais, CSIRO Divisão de Tecnologia da Informação, como um protótipo para explorar o potencial de modelos de dados relacional estendido para banco de dados espaciais. SIRO-DBMS provê um alto nível de integração de imagens, dados vetor, texto e, adicionalmente, métodos de acesso especialistas e opções para representação interna da geometria de objetos para ajustar um projeto de banco de dados aos requerimentos de uma aplicação.

As extensões ao modelo relacional consistem em tipos de dados definidos pelo usuário e extensões para gerenciamento de dados.

SIRO-DBMS suplementa as facilidades padrão de SGBD relacionais com visões de mais alto nível de entidades e operações para módulos escritos em C. As facilidades do SIRO-DBMS são limitadas a criação, alteração e consulta do banco de dados, com a análise, display e outras operações de interface de usuário do sistema.

A interface entre SIRO-DBMS e SGBD relacionais é programado no SQL de modo que o SIRO-DBMS é potencialmente portátil para SGBD relacionais que ofereçam SQL.

Duas extensões para o modelo são providas: um conjunto de tipos de dados geométricos e uma forma limitada de hierarquia "IS_A".

Quatro tipos de dados de localização são providos: para ponto, retângulos, polígonos e imagem. O efeito é que o desenvolvedor de aplicação pode tratar uma descrição especial de uma entidade com apenas um outro atributo. Uma forma de hierarquia "IS_A" é provido através de classes, denominados conjuntos de relações de entidades espaciais, com uma classe capaz de ser tratada como sujeito de uma pesquisa.

Existem dois elementos para implementação de tipos de dados espaciais. O primeiro procura prover uma fundação para a extensão simplificada do conjunto de tipos de dados através do mapeamento de cada tipo de dados geográfico e um conjunto de atributos atômicos cujos tipos são projetados do conjunto de tipos de dados providos pelo SGBD relacional e pela linguagem C. O tipo ponto, por exemplo, é mapeado como dois atributos,

um atributo interpretado como uma coordenada X e outro interpretado como coordenada Y, ambos "número" para o SGBD relacional e "float" para C.

A linguagem utilizada é uma extensão do SQL - SESQL (Spatial Extended SQL).

3.12.5 VISION*

VISION* [Geo92] é um SIG desenvolvido pela Geovision em 1985. Todos os dados estão armazenados em um SGBD relacional (Oracle ou Ingres). É usado para criar, manter, mostrar e analisar informação geográfica.

O VISION pode tratar dados espaciais, dados convencionais e imagens. Dados espaciais identificam o local geográfico e seus relacionamentos (topológicos e de conectividade) entre características geográficas. Dados convencionais são dados descritivos, que podem ser associados a um local específico a um banco de dados como um todo. Dado imagem é uma informação pictórica como fotos, esquemas e gráficos, que podem estar associados a uma característica⁸ ou grupo de características ou a um banco de dados como um todo. Todos os dados (espacial, convencional e imagem) são mantidos juntos no mesmo banco de dados lógico.

Os dados são armazenados em uma hierarquia de estruturas, que ajudam a simplificar e a esclarecer os relacionamentos e facilitam o entendimento das interações dos dados.

Muitas estruturas lógicas definidas pelo usuário e estruturas topológicas podem ser desenvolvidas e armazenadas dentro do banco de dados VISION*. Estas estruturas são: features, redes, camadas e grupos (Figura 3.8).

Uma entidade geográfica que tem um ou mais pontos de coordenada é chamada de *feature*. Uma feature é a unidade básica no banco de dados. Uma feature pode ser composta de um número ilimitado de coordenadas (x,y) ou (x,y,z), se é bidimensional ou tridimensional, respectivamente. Existem três tipos de features: pontos, linhas e polígonos.

Um feature do tipo ponto deve ter no mínimo um ponto de coordenada para localizá-lo na superfície da terra. Uma linha deve ter um ponto inicial e um final. Pode conter um número ilimitado de outros pontos para rastrear sua rota. Um polígono é um feature que compreende uma área e é identificado por um ponto chamado centróide localizado em algum lugar dentro da área.

Uma rede é uma estrutura que contém features topologicamente relacionados. Os relacionamentos topológicos são armazenados e mantidos dentro do banco de dados e são usados pelo sistema em consultas avançadas e funções de análise. Existem quatro tipos

⁸feature

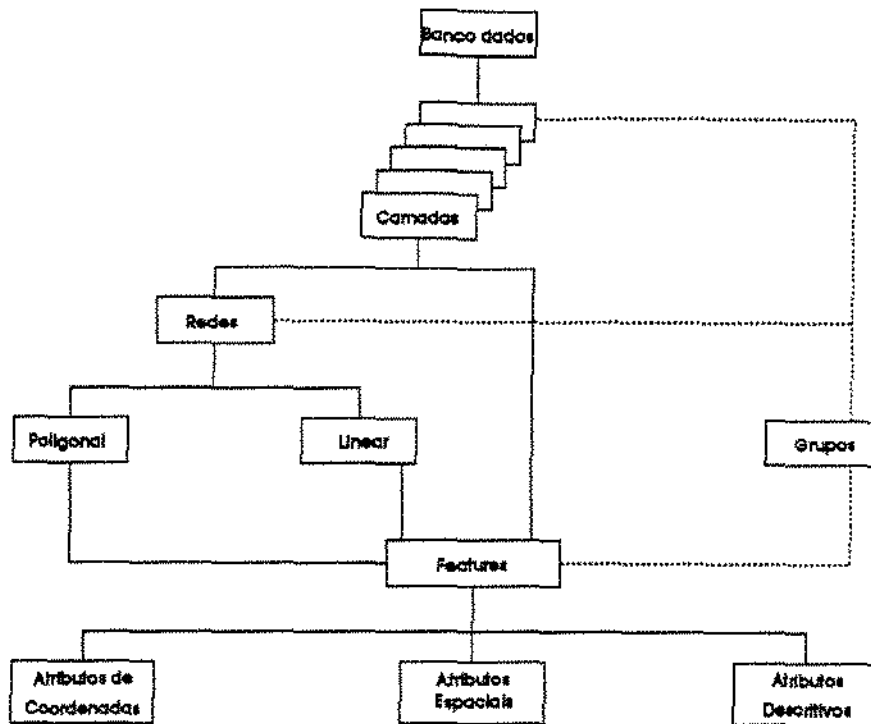


Figura 3.8: Estrutura do banco de dados.

de redes: poligonal, linear, definida pelo usuário e não conectada.

Redes de polígono contém um número ilimitado de polígonos conectados e “livres”. Cada linha na rede tem um nó em cada ponto terminal.

Nós são features definidos pelo usuário, sendo um caso especial de ponto. Eles podem existir em uma rede com ou sem linhas ligando o ponto.

Uma rede linear contém features de linha, que são conectados em seus extremos ou são separados geograficamente. Nós podem existir independente da existência de uma linha conectando-os. Nas redes definidas pelo usuário é ele quem define a tabela de conectividade.

Redes não conectadas podem conter qualquer tipo e combinação de features. Camadas são estruturas do banco de dados que tematicamente agrupam features e redes que estão relacionados logicamente (rios, lagos e oceanos podem ser agrupados em uma camada de hidrologia).

A estrutura grupo permite a criação de uma estrutura temporária para realizar manipulações. Grupos são freqüentemente utilizados para executar alguma manipulação sobre um número de features, que de outro modelo não seriam associados.

Aplicações típicas de VISION* incluem gerenciamento de redes de distribuição de gás, transmissão de eletricidade, telefonia, gerenciamento de recursos naturais, entre outras.

Capítulo 4

Estudo de caso - UNINet

O exemplo a ser utilizado na dissertação é a rede de computadores da Unicamp - UNINet. Suas principais características estão descritas a seguir[San93] [LdA95].

A modelagem foi realizada a partir das informações disponíveis sobre a UNINet, descritas neste capítulo. Os modelos utilizados são o E-R e o Orientado a Objeto, pois existem implementações disponíveis (relacional para o E-R e O₂ para o orientado a objeto) e o objetivo é justamente de comparar esses dois modelos e nada melhor do que comparar as implementações.

Este capítulo está dividido da seguinte forma: inicialmente, é feita uma descrição do funcionamento da rede de computadores da Unicamp, a partir daí, a rede é dividida em três partes para facilitar a análise: mapeamento urbano, rede UNINet, rede departamental.

4.1 Características da UNINet

O *backbone*¹ da rede consiste em uma fibra óptica com extensão de 16 km e par trançado com extensão de 40 Km. Existe por toda a Unicamp uma rede de dutos, caixas e armários que dão o suporte físico a essas duas redes.

A fibra óptica e o par trançado percorrem a mesma malha de dutos e caixas, ou seja, os caminhos possíveis de percurso são os mesmos para ambos. As diferenças são intrínsecas às características de cada um dos materiais.

O *backbone* de par trançado é mais antigo do que a fibra óptica, por razões óbvias. Considerado o seu custo relativamente baixo e a sua existência prévia, preferiu-se continuar a utilizá-los, principalmente para conexão de terminais a equipamentos de grande

¹espinha dorsal

porte como o IBM, que possui problemas de conexão com Ethernet.

O *backbone* de fibra óptica consiste, atualmente, de cinco nós principais, ligados através de fibra óptica e dos quais é feita a ligação com as redes departamentais. Isto foi feito para facilitar o gerenciamento dos recursos da rede (Figura 4.1). Os cinco nós são:

- nó 19, próximo à FEM, conhecido como nó central;
- nó 1, localizado no antigo CCUEC e conhecido como nó do Básico;
- nó 61, localizado no prédio do Hospital das Clínicas, conhecido como nó do HC;
- nó 47, localizado no DGA;
- nó 207, localizado no CCUEC.

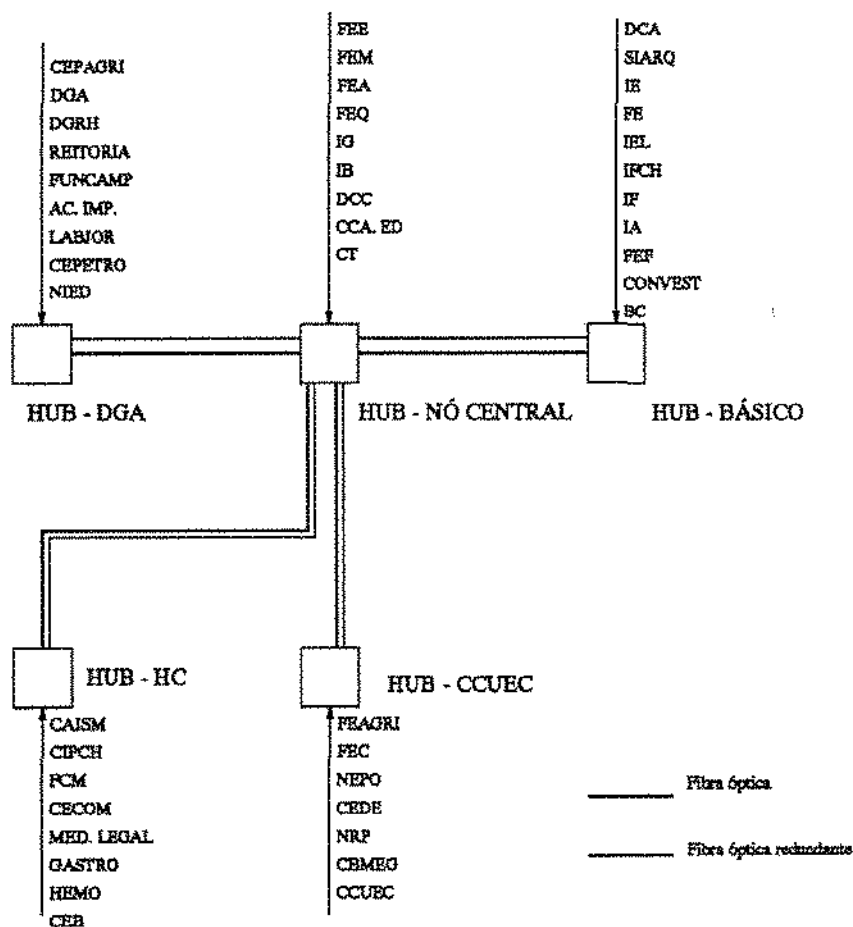


Figura 4.1: Backbone da UNINet.

Pela Figura 4.1 pode-se observar que o nó 19 continua sendo uma espécie de nó central, pelo qual passam todas as mensagens. A rede de fibra óptica é ponto-a-ponto, no sentido que a fibra que sai do nó 19 para cada departamento, em nenhum lugar sofre derivação. Daí se conclui que existem fibras ópticas individuais para cada rede departamental.

Cada nó consiste em um distribuidor central (*hub*). As unidades e institutos da Unicamp são conectados, de acordo com a proximidade física desses *hubs*, através de um único gateway departamental. Dentro de cada unidade existe um *backbone* Ethernet e várias subredes conectadas a estes através de outros gateways. São as redes departamentais.

Para controle e segurança, devido a características próprias e dispersão geográfica, a rede de par trançado necessita de elementos como caixas, colocadas a intervalos máximos de 50 metros. Necessita também de armários, por onde os cabos são distribuídos para conexão às redes departamentais.

As fibras ópticas utilizam as mesmas caixas do par trançado, mas com dutos diferentes. A ligação entre as caixas é feita, na sua maior parte, através de dois dutos. Apenas algumas caixas se ligam através de 3 ou 4 dutos. As caixas servem como segurança para possíveis problemas que possam ocorrer nos cabos e/ou dutos.

Tanto a fibra óptica quanto o par trançado não sofrem ação do repetidores durante o percurso no *backbone*.

O roteador externo é um CISCO. Existem linhas a 64 kbits/s com a FAPESP (de onde sai o link internacional) e a USP, que provê uma ligação redundante com a FAPESP. Instituições regionais (CTI, CPqD Telebrás, EMBRAPA) estão ligadas no roteador CISCO ou via servidores SLIP de terminais (Figura 4.2).

A maioria das redes departamentais usam Ethernet e os protocolos de rede e transporte são o TCP/IP (Transmission Control Protocol/Internet Protocol). O padrão Ethernet utilizado é o IEEE 802.3 e a implementação consiste de uma ligação em uma árvore de barras entre os equipamentos conectados. Neste tipo de rede, um único cabo substitui um grande número de cabos de interconexão utilizados em redes de comunicação tradicionais.

Cada prédio servido pela rede local possui um ou mais ramos que configura sua "rede departamental". As redes departamentais são interligadas por enlaces de cabos de fibra óptica, através de "distribuidores centrais" (*HUB*). Pode haver alguma rede departamental não alcançada pela fibra óptica e, se não houver outro meio como ondas de rádio, ele será alcançado através do par trançado. Neste caso haverá o problema de velocidade, que será reduzida.

O distribuidor central corresponde a um concentrador e controlador óptico. É constituído pelos equipamentos (Figura 4.3):

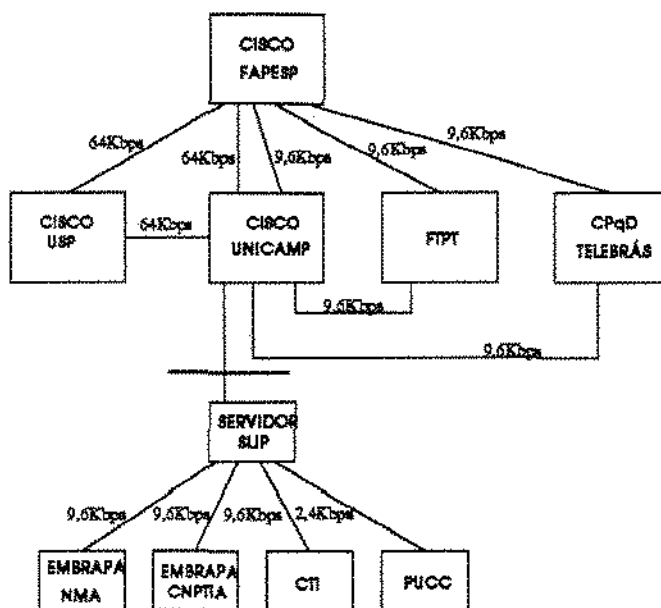


Figura 4.2: Rede ANSP.

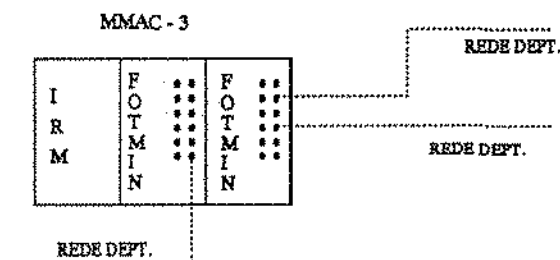


Figura 4.3: MMAC-3.

- MMAC-3² rack para o conjunto
- FOT-MIN³ conjunto para conectar as fibras (2 unidades, total de 24 conectores)
- IRM⁴ controlador inteligente

A cada unidade acadêmica chega um par de fibra óptica ligado ao distribuidor central, que está conectado a uma estação servidora através de um transceptor óptico FOT-1 e uma ponta Ethernet (ver fig. 4.4).

²Multi Media Access Center

³Fiber Optic Media Interface Module

⁴Intelligent Repeater Module

As redes departamentais são *Thick Ethernet* (Cabo Grosso), também chamadas de *Yellow Cable*. Algumas redes também possuem o *Thin Ethernet* (Cabo Fino), que segue o método de acesso IEEE 802.3.

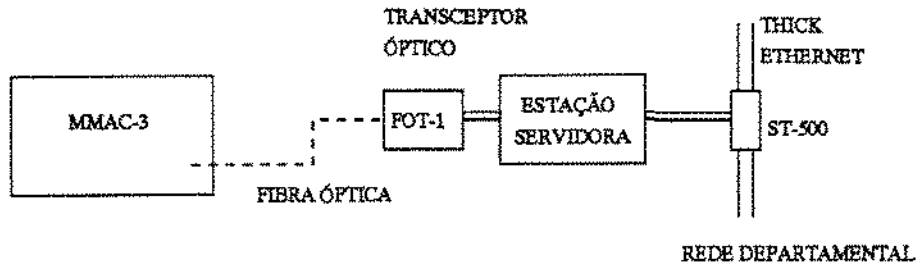


Figura 4.4: FOT-1.

As ligações de equipamentos na rede departamental são de quatro tipos:

- estação ligada ao cabo, via equipamento transceptor ST-500 (Figura 4.5).
- vários equipamentos ligados ao multiplexador MT-800, que se conecta a Ethernet através do transceptor ST-500, para o caso de equipamentos instalados próximos um dos outros (Figura 4.6).
- um microcomputador tipo IBM-PC ligado diretamente ao cabo Ethernet através de uma placa para conexão em rede (Figura 4.7).
- quando uma rede departamental envolve mais de um prédio, utiliza-se um par de repetidores de fibra óptica FR-3000 entre os ramos Ethernet (Figura 4.8).

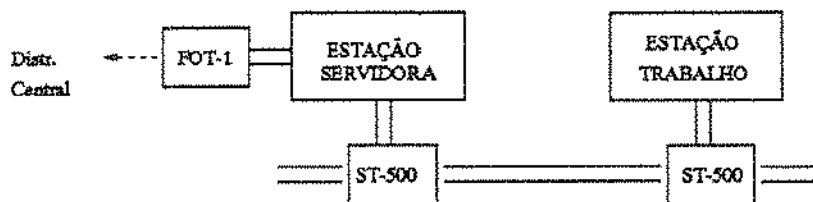


Figura 4.5: ST-500.

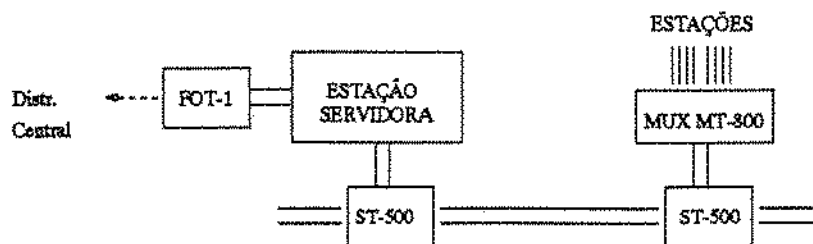


Figura 4.6: MT-800.



Figura 4.7: IBM-PC.

4.1.1 TCP/IP

O software utilizado na rede é o TCP/IP, e todos os equipamentos computacionais da UNICAMP possuem implementação. Algumas facilidades disponíveis são:

- emulação remota de terminais (*telnet*)
- transferência de arquivos (*ftp*)
- correio eletrônico (*SMTP*)

4.1.2 Conexão de máquinas de grande porte

Não há diferenças de conexão entre uma máquina de grande porte VAX, CDC, ServerSUN e uma simples estação de trabalho. Ambas usam um transceptor tipo ST-500 “vampirado” ao cabo Thick Ethernet. Isto é encontrado em equipamentos centrais de conexão de terminais, como servidores de terminais ou roteadores para redes públicas X.25.

A maior diferença encontra-se na conexão do IBM 3090. São duas arquiteturas diferentes: SNA/IBM e Ethernet com TCP/IP. Por isto foi previsto um *Gateway* SNA-Ethernet, equipamento denominado pela IBM de 3172 Interconnect Host Lan. Há outra ligação do IBM 3090-Rede através de uma linha serial Síncrona da Controladora de Comunicação

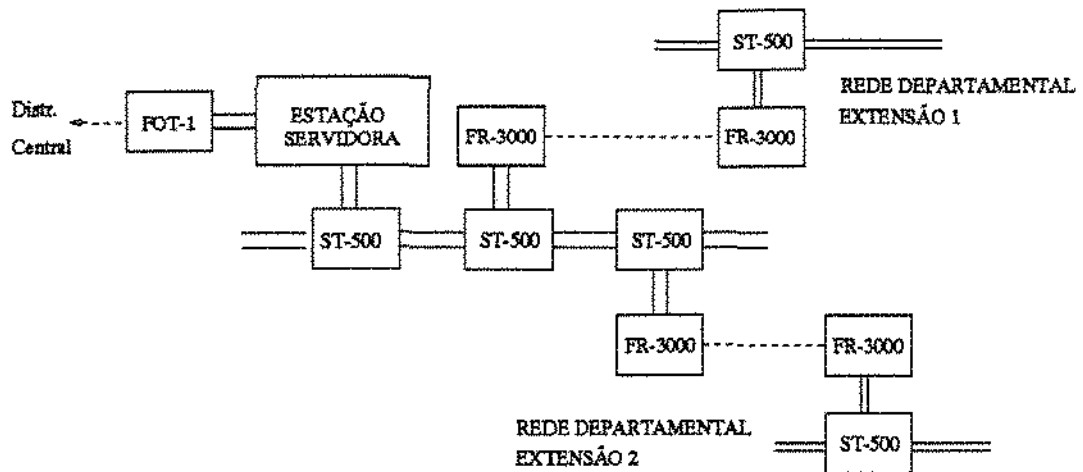


Figura 4.8: FR-3000.

3720 e uma porta servidora SUN. Nesta linha está implementado o protocolo X.25 e sobre ele o TCP/IP. Estas conexões não serão analisadas neste trabalho.

A UNINet se liga com outras redes, como BITNet, Internet e Rempac. Permite acesso remoto a outros computadores, consultas a base de dados e uso residencial.

Este último permite o acesso a qualquer equipamento da rede a partir da rede pública de telefonia.

As ligações com as redes BITNet e Internet são feitas através da FAPESP, que possui equipamentos DEC, que servem como roteadores de mensagens e conexões a BITNet e Internet.

4.1.3 Componentes da UNINet

Podemos citar os componentes físicos da rede como sendo:

- fibra óptica;
- par trançado;
- duto;
- caixa;
- armário;
- FOT-1;

- HUB;
- equipamentos.

Do ponto de vista de SIG, as entidades com características geo-referenciadas possuem representação gráfica de ponto ou linha. Um ponto pode ser considerado como uma entidade contendo coordenadas espaciais como atributos e linha tendo coordenadas iniciais e finais. Caixa, armário, FOT-1, HUB e equipamento possuem características de ponto.

A velocidade do par trançado é de 9,6 Kbps. A velocidade da fibra óptica depende basicamente das interfaces que estão nas pontas. Se a fibra é utilizada com Ethernet a velocidade é de 10 Mbps, se for utilizado FDDI será de 100 Mbps, mas se for utilizada Token Ring será de 4 ou 16 Mbps.

Um cabo do tipo par trançado é composto por um conjunto de pares lógicos. Esses pares lógicos podem ser separados formando outros cabos. Isto é chamado de distribuição. Essa distribuição ocorre nos armários. O cabo original deixa de existir e novos cabos surgem contendo subconjuntos de pares trançados.

Portanto, a extremidade de um par trançado pode ser um armário, onde ele é conectado (*jump*) a outro(s) par(es) trançado(s) ou não, e neste último caso eles ficam estacionados naquele armário. O par pode também ter como extremidade um equipamento: terminal, PC ou impressora.

As extremidades de fibra óptica são dois HUBs ou um HUB e um roteador. HUB no caso dos cabos que fazem a comunicação entre os HUBs, e HUB/roteador quando as fibras se conectam às redes departamentais.

Entre duas caixas pode-se ter dois, três ou quatro dutos. Em um desses dutos sempre estará passando somente fibra óptica, enquanto nos outros dutos passa par trançado.

Para este estudo, deve-se analisar como é feito o percurso de um cabo físico pela rede (*backbone*). Os segmentos de cabo entram na rede a partir de um equipamento/armário e passando através de dutos caminham pela rede. Se existe algum armário no caminho, todos os cabos de par trançado entram no armário, onde podem sofrer distribuição, ser conectados ou ficar neste armário.

Procura-se sempre o menor caminho para os cabos, mas existem casos onde o cabo caminha mais do que o necessário, para que possa atender outros nós, ou simplesmente por alguma outra razão importante. Fica difícil adivinhar qual o caminho realizado por um determinado cabo nesta situação. Na Figura 4.9, por exemplo, existem dois caminhos possíveis entre as caixas 188 e 195.

Um armário sempre está relacionado a uma caixa. Os cabos que passam pela caixa podem sofrer ação desse armário. Se houver uma distribuição ou simplesmente um "*jump*"

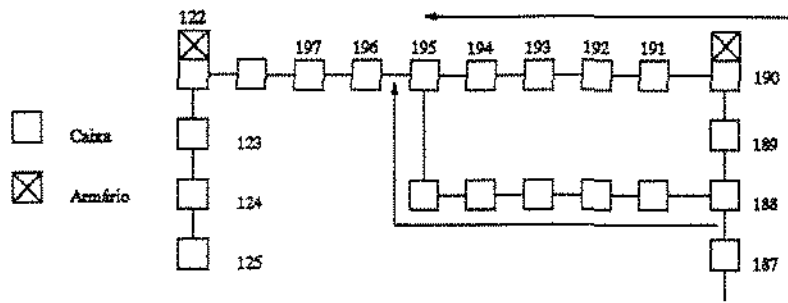


Figura 4.9: Dois caminhos possíveis entre as caixas 188 e 195.

desses cabos, isto implicará na existência de um outro segmento de cabo, que dará continuidade lógica àqueles cabos pois, na verdade, o sinal transmitido é o mesmo, mas o (segmento de) cabo é outro.

Como pode haver mais de um caminho possível entre dois pontos na rede, ou seja, existir mais de um trecho de duto ligando duas caixas, os cabos podem fazer mais de um caminho. Além disso, por outras razões, pode-se ter caminhos de cabos no qual trechos de dutos são percorridos mais de uma vez (Figura 4.10).

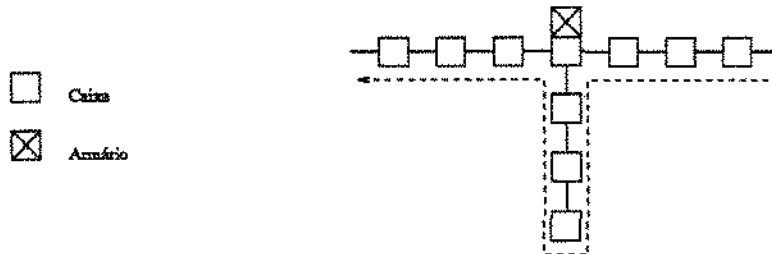


Figura 4.10: Percurso redundante.

Como pode-se observar, o caminho nem sempre será o menor possível.

Quando os cabos chegam nos armários é feita a distribuição de alguns desses cabos. A distribuição nada mais é do que conectá-los a outros cabos, geralmente para ligação às redes departamentais. Estes cabos saem aos pares, fazendo a ligação das redes departamentais. Estes novos (segmentos de) cabos saem em direção à rede departamental e o caminho a ser percorrido pode incluir trechos de dutos da própria rede UNINet. Outros podem caminhar do armário diretamente para a rede departamental. Portanto, a rede também estará sendo utilizada por (segmentos) de cabos com destino definido. A capacidade utilizada dos trechos de dutos deve levar em conta estes cabos adicionais.

A rede UNINet está constantemente sofrendo ampliações, com ligações de novas redes departamentais. Estas novas ligações seguem a lógica da distribuição física atual da rede, ou seja, procura-se o armário mais próximo da nova rede e verifica-se se existe condição de atender a esta nova demanda. Caso haja disponibilidade no armário, verifica-se o percurso que o segmento de cabo irá fazer do armário até este ponto da rede local. Se houver espaço no trecho de dutos para a passagem deste segmento de cabo, a ligação é feita facilmente. Se o armário não tem condições de atender à nova demanda, verifica-se a possibilidade de se criar um novo armário.

Para isto é necessário que se guarde (ou que se calcule) em cada armário a sua capacidade instalada e a sua capacidade utilizada, para que se tenha uma resposta rápida a uma consulta, como a de cima. Esta informação está necessariamente ligada a capacidade do trecho de duto que chega à caixa relacionada ao armário. A capacidade do armário está limitada à capacidade total dos trechos de duto que chegam na caixa.

Os segmentos de cabo poderão ser definidos utilizando o equipamento/armário inicial e equipamento/armário final, desde que não haja outros caminhos entre eles. Se houver, pode-se ter uma outra definição do caminho, descrevendo as caixas importantes do percurso e, para que não haja dúvida, pode-se chegar ao máximo de citar todas as caixas por onde o segmento de cabo está passando.

Existem várias formas de descrever os elementos da rede. Pode-se ter uma representação de trechos de cabos independente de caixas, ou seja, apenas com início e fim do trecho, desde que exista apenas um único caminho entre esses dois pontos. Um segmento de cabo pode também ser representado através dos trechos de dutos que ele atravessa.

No caso de fibra óptica, os atuais cinco nós centrais da rede são considerados equipamentos e a partir deles é feito o controle de cada subrede. Uma mensagem que interessa apenas à subrede que a originou não é transmitida a outras redes (via nó 19). Portanto, pode-se considerar os equipamentos nestes nós como pontes⁵, reduzindo o tráfego no *backbone*.

Os elementos da rede (caixa, armário, duto, HUB, FOT-1, equipamentos - impressora, terminal e PC e cabos - fibra óptica e par trançado) estão todos explicitados acima, e a seguir serão estabelecidas algumas possíveis operações entre eles.

4.1.4 Possíveis operações entre elementos de rede

Podem-se requisitar inúmeras informações de uma rede como a UNINet. Elas podem ir desde informações básicas (quais os segmentos de cabos que passam por um determinado trecho de dutos), a informações mais complexas (sobre onde fica o armário mais próximo

⁵bridge, em inglês

de um determinado ponto na rede, ou quais são os possíveis caminhos que fazem a ligação entre dois pontos). A partir dessas operações fica mais fácil a escolha da representação.

As operações que se fazem comumente na UNINet dizem respeito a obter:

- informações sobre os caminhos possíveis (através de cabos/dutos) para se chegar a um determinado local;
- qual o melhor caminho para se passar um cabo de um determinado ponto a outro da rede;
- como fazer para ligar um determinado ponto fora da rede da Unicamp na rede UNINet, por qual armário, por quais trechos de dutos;
- qual o caminho mínimo para uma determinada informação ser enviada de um nó a outro da rede (inclusive tempo e velocidade de transmissão);
- informações relativas a par trançado: início, fim, por quais caixas passa, por quais armários passa, por quais é “jumpeado”, em qual sofre distribuição, por quais trechos de dutos atravessa;
- cabo: início, fim, por quais dutos passa, por quais caixas passa;
- informações relativas a caixa: localização, se está relacionada a armário, que trechos de dutos está conectada, que segmentos de cabos passam por ela;
- informações globais: quantas redes departamentais existem, quantos nós existem, velocidade (máxima e mínima) existente, área máxima alcançada e alcançável.

Além da difusão por cabo, a rede UNINet dispõe de usuários que fazem parte da rede e que estão ligados através de antenas parabólicas. Estes usuários podem ser considerados como qualquer outro pois esta antena se conecta à rede através de uma caixa. Isto é algo novo e busca atender os usuários situados a grande distância.

4.2 Redes departamentais

Os elementos das redes departamentais são:

- backbone (cabos Ethernet);
- estações de trabalho;

- PC's;
- terminais;
- equipamento de grande porte;
- ponte;
- roteador;
- impressoras;
- transceptor ST-500;
- repetidor de fibra óptica FR-3000;
- multiplexador MT-800.

Para conectar uma rede departamental, um par de fibra óptica deve ser ligado primeiro a um equipamento FOT-1 (transceptor óptico) e a partir deste equipamento, é ligado a um roteador. A partir deste roteador é que está conectado o *backbone*.

Este backbone é feito de cabo coaxial Ethernet e a topologia utilizada é o barramento. As ligações ao barramento devem distar de no mínimo 2,5 m. Em cada ponto pode ser conectado quaisquer equipamentos: PC, terminal, estação de trabalho, roteador, ponte, desde que este equipamento tenha uma placa de rede para conexão. Impressoras sempre ficam conectadas em outros equipamentos, com exceção de ponte.

Cada rede na Unicamp pode ter no máximo 62 *hosts*, ou seja, podem ter conectados a ela 62 equipamentos. Uma rede tem um número IP⁶ e pode rotear até 62 números IP (endereços). Para haver essa conexão é necessário um "vampiro", equipamento que capta as mensagens que passam pela rede e possa enviar as suas próprias mensagens. Esse equipamento é o ST-500.

Um roteador pode ser um PC ou uma estação de trabalho. A única característica do roteador é que ele possui um programa para rotear pacotes.

Cada equipamento (roteador, PC, ET, terminal, grande porte) ligado a rede possui um número IP. Se o equipamento faz parte de mais de uma rede, pode possuir mais de um endereço IP, um para cada rede. Neste trabalho, para efeito de simplificação será considerado apenas um único endereço IP.

Um desses equipamentos pode ser outro roteador e, a partir deste, se conecta outro *backbone* e uma nova sub-rede é formada, com um novo número IP, sem qualquer relacionamento com a sub-rede anterior.

⁶*Internet Protocol*

As redes da UNINet possuem algumas regras mínimas que foram estabelecidas para facilitar a sua administração. Os endereços IP codificam a rede e a máquina dentro da rede. Eles não especificam uma máquina, mas sim uma conexão à rede. A rede da Unicamp é da classe B e os endereços nessa classe possuem a máscara 255.255.255.192. Os dois primeiros números são fixos para a Unicamp (143.106) e os outros dois informam a rede e a quantidade de *hosts*. Isto permite 1022 subredes com 62 máquinas cada.

Mais de uma subrede pode existir em apenas um departamento, e atualmente a Unicamp possui cerca de 50 Departamentos e 200 subredes.

Como pode-se observar na Figura 4.11 o CCUEC possui uma grande quantidade de redes/subredes. Tanto a rede quanto a subrede respeitam as mesmas regras citadas anteriormente. Considerando-se apenas uma subrede (Figura 4.12) (conectividade, por exemplo), tem-se:

- 3 estações de trabalho, das quais uma é o roteador para essa rede;
- 6 PCs 486; e
- 1 impressora laser (abaixo de um dos PCs).

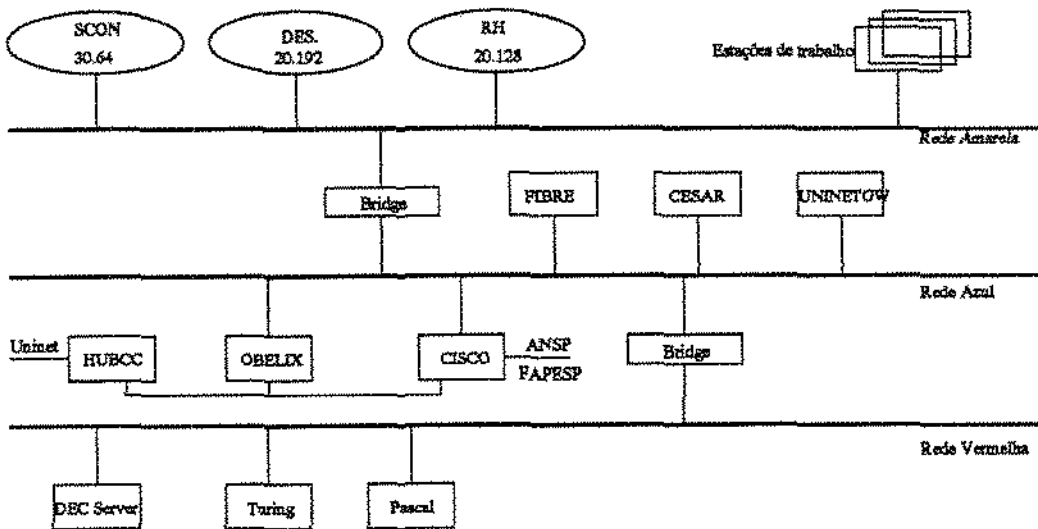


Figura 4.11: Rede do centro de computação.

Esta subrede tem os endereços 143.106.30.(64 - 127). A rede a qual ela está ligada, denominada como rede amarela, tem os endereços 143.106.10.(1 - 63).

Atualmente existem cerca de 200 endereços de rede/subrede já utilizados. Os endereços vagos são utilizados à medida que são solicitados e a administração da rede é responsável pelo controle e distribuição desses endereços.

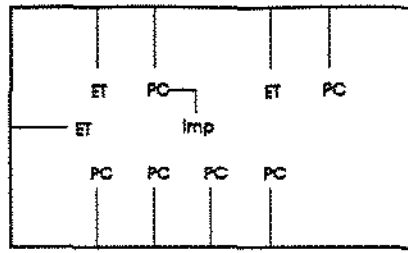


Figura 4.12: Rede do setor de conectividade.

O CCUEC se liga a UNINet através de uma ligação para dois equipamentos (Obelix e Cisco), de forma redundante (Figura. 4.11). A subrede da conectividade (SCON) se liga através de seu roteador a rede amarela, e desta através de um bridge a rede azul e, a partir daí, se liga a uma dessas duas máquinas chegando finalmente a UNINet.

4.3 Modelagem de redes

Aplicações como administração de instalações e redes de utilidade pública, conhecidas como AM/FM (*Automated mapping/facilities management*), são, principalmente, todas as aplicações de gerenciamento de redes para áreas urbanas. Exemplos incluem redes de telefonia, esgoto, água, distribuição de energia elétrica, etc.

Essas aplicações envolvem o armazenamento e o entendimento da divisão urbana com a qual interagem. Este mapeamento urbano diz respeito a ruas (logradouros), quadras, prédios, cruzamentos, etc. Cada rede deve considerar a estruturação urbana pela qual irá passar. Aplicações completas podem definir elementos de hidrografia, altimetria, marcos geodésicos, entre outros.

O exemplo UNINet consiste de um sistema de gerenciamento de rede externa. Por rede externa entenda-se o conjunto de cabos, canalizações subterrâneas, equipamentos de sustentação e proteção a esses cabos, além de outros dispositivos complementares.

Sobre esta camada de mapeamento urbano são desenvolvidas as aplicações de gerenciamento de redes. Como SIG representa um investimento financeiro elevado, apenas as informações necessárias devem ser armazenadas. Em geral, pode-se dizer que são constituídas por ruas, logradouros, quadras e lotes.

Os elementos da rede externa estão, em geral, posicionados sobre o mapeamento, como é o caso de elementos de canalização. Um conjunto de estruturas topológicas, denominadas redes, podem ser definidas e são fundamentais à representação da rede externa. Essas estruturas topológicas permitem a modelagem de várias redes existentes na rede externa,

por exemplo redes de cabos e redes de canalização.

O mapeamento urbano da UNINet conterá apenas as informações necessárias para dar suporte ao modelo de rede. Essas informações são: logradouro, linha central, quadra e prédio.

Um logradouro é qualquer rua, beco ou avenida, presentes no traçado urbano do campus. No mapeamento, um logradouro é representado por vários segmentos de linha central. Os segmentos de linha central são linhas equidistantes de duas faces de quadras opostas, que passam ao longo do eixo dos logradouros e são delimitadas por pontos de interseção (cruzamento de ruas) com outros logradouros.

Modelagem de redes externas

Redes externas possuem características próprias como dutos e cabos. Tanto cabos como dutos formam uma malha, regidos através de determinadas regras. Todas as entidades e relacionamentos podem ser modelados através de quatro tipos de modelos. A escolha do modelo depende de uma análise das especificações da aplicação [Geo92]:

- Arco-nó: define o relacionamento entre um elemento arco e um elemento nó. Cada extremo de um arco é conectado a um nó. Pode-se imaginar nós como sendo cidades e arcos como sendo as estradas ligando as cidades. Pode-se ter fluxo de tráfego associado com a conexão entre cada arco e nó.
- Nó-arco-nó: este modelo define um relacionamento entre dois elementos nós ao longo de um elemento arco. Um bom exemplo é direção de fluxo em canalização de água. É mais natural modelar a direção do fluxo como um atributo do relacionamento entre os dois nós ao longo da canalização.

O relacionamento é modelado de nó-a-nó ao longo do arco, portanto a direção é implícita no relacionamento pela ordem na qual os nós são listados na definição de conectividade.

- arco-nó-arco;
- Elemento-elemento: este modelo define um relacionamento entre elementos mas não faz distinção entre elementos do tipo arco e do tipo nó. Um exemplo é uma rede de telefonia que pode ter segmentos de cabos conectados juntos onde não há dispositivos físicos ou conector que seja interessante modelar.

A Uninet pode ser considerada como uma aplicação nó-arco-nó, considerando cabos e dutos como arcos e todas as outras entidades como nós.

4.4 Modelagem

Aqui propomos a modelagem da rede UNINet usando E-R e metodologia OMT. Apenas por uma questão de melhor visualização e compreensão, as modelagens E-R e OMT têm como resultado três diagramas:

- diagrama do mapeamento urbano;
- diagrama da rede de dutos e cabos;
- diagrama da rede departamental.

Os diagramas dos modelos possuem uma letra no canto inferior direito, informando o tipo de representação gráfica para a entidade (L - linha, PL - polilinha ou P - ponto). Essa informação geométrica é essencial em SIG, e sobre ela são efetuadas consultas.

4.4.1 E-R

Mapeamento urbano

O Mapeamento urbano resultou no diagrama (Figura 4.13).

As entidades linha central, quadra e prédio são geo-referenciadas. Portanto, possuem representação gráfica de polilinha/polígono. Como possuem um posicionamento fixo no espaço, os relacionamentos *Adjacente* e *Contém* se tornam redundantes, pois podem ser deduzidos a partir do posicionamento dessas entidades.

Entidades:

- ponto - composto pelas coordenadas georreferenciadas x e y:
 - x: real;
 - y: real.
- linha - segmento de reta;
- polilinha - conjunto de linhas;
- polígono - conjunto de linhas;
- logradouro - conjunto de linhas centrais que representam uma rua
 - código_logradouro: inteiro;

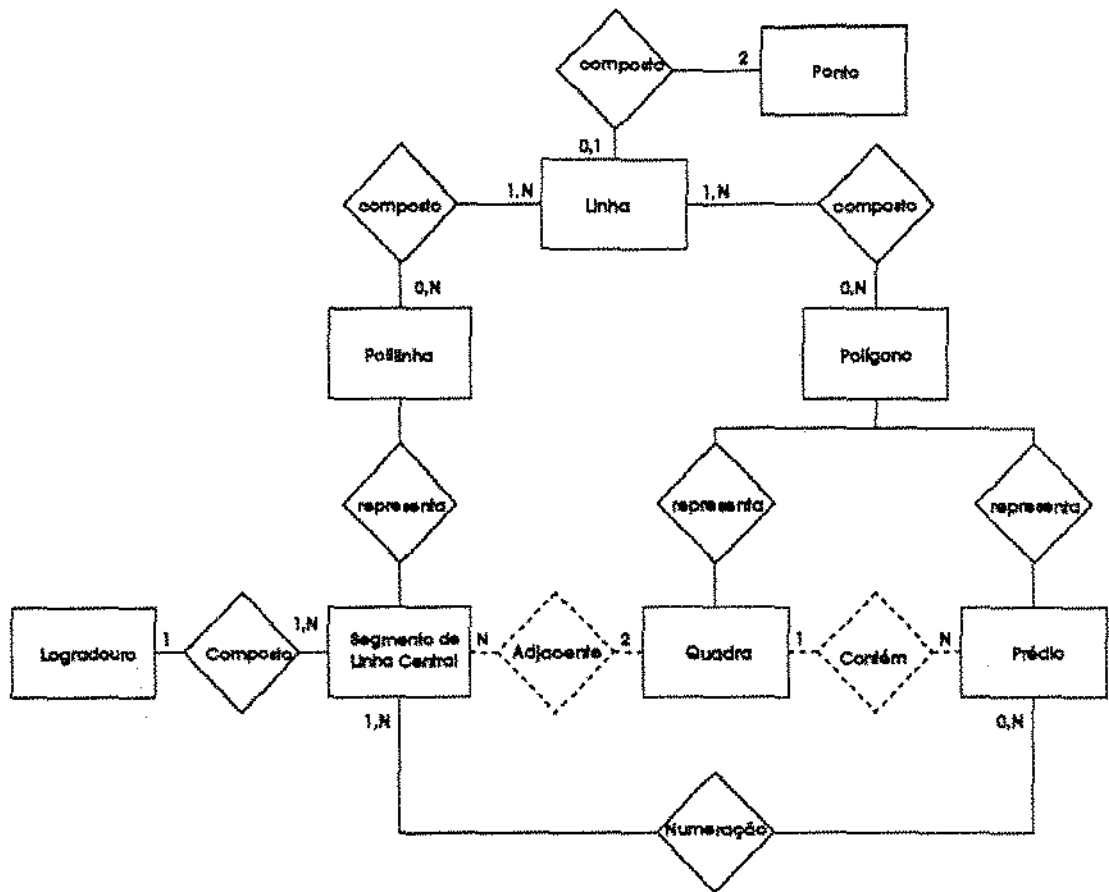


Figura 4.13: Modelo E-R para mapeamento urbano da UNINet.

- nome_logradouro: cadeia;
- segmento de linha central - trecho de logradouro delimitado por pontos de interseção com outros logradouros
 - identificador_linha_central: inteiro;
- quadras - área limitada
 - identificador_quadra: inteiro;
- prédio - edificações significativas na Unicamp
 - identificador_prédio: inteiro;

Os relacionamentos do modelo são:

- numeração (segmento de linha central - prédio) - um prédio situado em um segmento de linha central possui uma numeração, o que configura um endereço. Ele pode ter mais de um endereço, situados em segmentos de linha central diferentes;
 - número: inteiro.
- composto (logradouro - segmentos de linha central) - um logradouro é composto por um ou mais segmentos de linha central;
- adjacente (segmento de linha central - quadra) - uma quadra é adjacente a um conjunto de linhas centrais - pode ser deduzido a partir do posicionamento das entidades;
- contém (quadra - prédio) - uma quadra contém um ou mais prédios, e um prédio só pode estar em apenas uma única quadra - pode ser deduzido a partir do posicionamento das entidades;
- representa(segmento de linha central - polilinha) - um segmento de linha central é representado graficamente como uma polilinha;
- representa(quadra - polígono) - uma quadra é representado graficamente como um polígono;
- representa(prédio - polígono) - um prédio é representado como um polígono;
- composto(polilinha - linha) - uma polilinha é um conjunto de linhas armazenadas de forma a gerar uma única linha contínua;
- composto (polígono - linha) - um polígono é um conjunto de linhas armazenadas de tal forma a gerar uma região fechada.
- composto(linha - ponto) - uma linha é composta por um ponto inicial e um ponto final.

As restrições ao modelo:

- um prédio com numeração em uma linha central tem de estar contido em uma quadra adjacente a esta linha central;
- uma polilinha é composta por linhas que tenham um ponto adjacente. As linhas devem ser armazenadas de forma ordenada;
- um polígono é composto de linhas que se conectam, onde a última linha se conecta a primeira em um ponto. As linhas devem ser armazenadas de forma ordenada.

Rede de dutos e cabos

A rede de dutos e cabos da Uninet, sem incluir as redes departamentais, foi considerada como ilustrado na Figura 4.14.

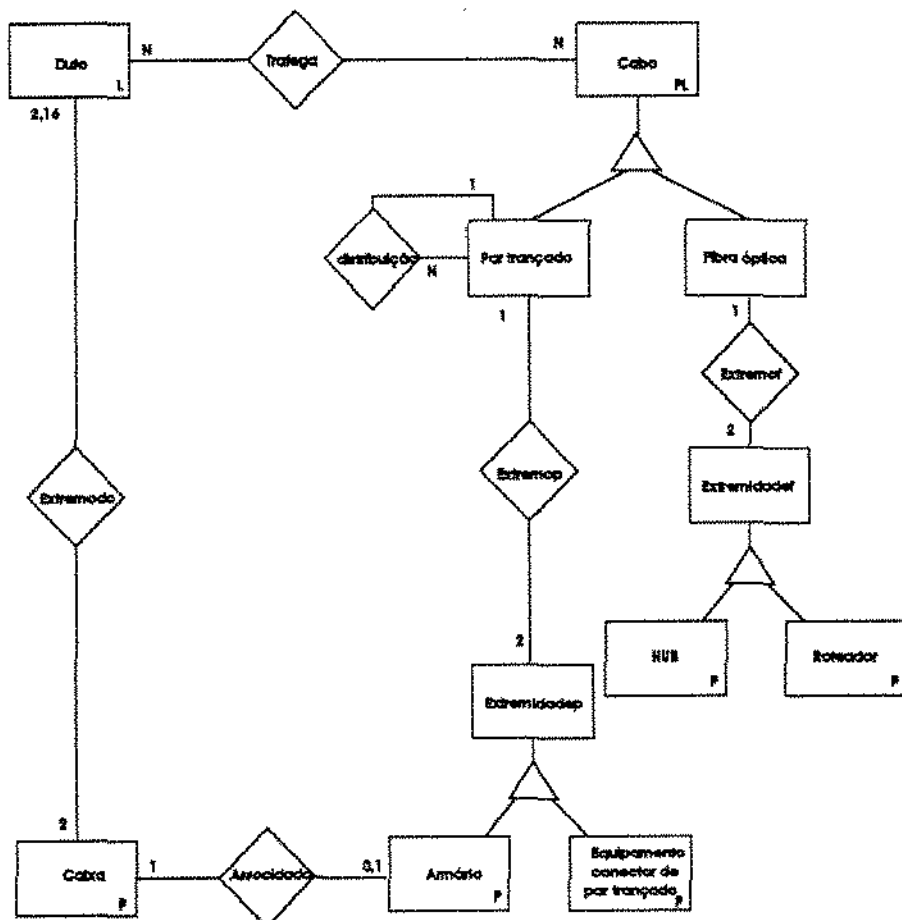


Figura 4.14: Modelo E-R da rede de dutos e cabos da UNINET.

Entidades:

- duto - tubos retilíneos utilizados como passagem para os cabos da Uninet;
 - duto_identificador: inteiro;
 - espessura: inteiro (polegadas);
 - material: cadeia (geralmente PVC);
 - capacidade máxima de cabos: inteiro;
 - capacidade instalada de cabos: inteiro (pode ser derivada);

- tamanho: inteiro (não superior a 50m).
- caixa - interseção de dutos, utilizados para controle e averiguação dos cabos de par trançado. Possui tamanho padronizado e pode ligar de dois a quatro dutos;
 - caixa_identificador: inteiro.
- armário - ponto de distribuição dos cabos de par trançado. Está sempre situado à superfície e está ligado a uma caixa. A sua dimensão é padronizada;
 - armário_identificador: inteiro;
 - capacidade_máxima_de_conexões: inteiro;
 - capacidade_instalada_de_conexões: inteiro (pode ser derivada).
- cabo - generalização de par trançado/fibra óptica para efeito de relacionamento com duto;
 - cabo_identificador: inteiro;
 - cabo_material: inteiro;
 - cabo_espessura: inteiro.
- fibra óptica - filamento onde o sinal transmitido é um pulso de luz;
 - fibra_óptica_identificador: inteiro.
 - tipo: inteiro (monomodo, multimodo);
- par trançado - conjunto de fios de cobre trançado;
 - par_trançado_identificador: inteiro.
- Equipamento - dispositivo que faz parte da rede departamental, podendo ser um PC, terminal ou uma estação de trabalho;
- HUB - concentrador e controlador óptico de onde partem e chegam fibras ópticas;
 - HUB_identificador: inteiro.
- roteador - equipamento que serve de entrada para as redes departamentais.
 - roteador_identificador: inteiro.

Os relacionamentos do modelo são:

- `trafega`(duto - cabo) - um cabo passa por vários dutos e um duto é atravessado por vários cabos;
- `extremodc`(duto - caixa) - um duto tem dois extremos que são duas caixas distintas e uma caixa serve como extremidade para até 04 dutos;
- `associado`(caixa - armário) - um armário está sempre associado a uma caixa, mas o contrário não é verdade;
- `distribuição`(par trançado-par trançado) - um par trançado ao chegar no armário pode sofrer distribuição dando origem a novos pares trançados que têm início neste armário;
- `extremop`(par trançado - extremidadep) - um par trançado tem como início e fim um armário ou um equipamento;
- `extremof`(fibra óptica - extremidadef) - uma fibra óptica tem início em um HUB e fim em um HUB ou em um roteador. Se for roteador haverá o atributo FOT-1 associado ao relacionamento.

As restrições ao modelo:

- um duto tem duas extremidades distintas, ou seja, duas caixas diferentes;
- se um duto é ocupado por uma fibra óptica não poderá ser utilizado por par trançado, e vice-versa;
- em um armário um par trançado sofre distribuição dando origem a n pares trançados;
- as extremidades de uma fibra óptica podem ser: 2 HUB; 1 HUB e um roteador. Nunca podem ser 2 roteadores;
- se uma caixa está associada a um armário o cabo que sofre distribuição neste armário deve passar por um duto que tem como extremidade essa caixa.

Rede departamental

A rede departamental é ilustrada no diagrama da Figura 4.15.

As entidades são as seguintes:

- Equipamento - ponto de ligação do par trançado na rede departamental;
 - `equipamento.identificador`: inteiro;

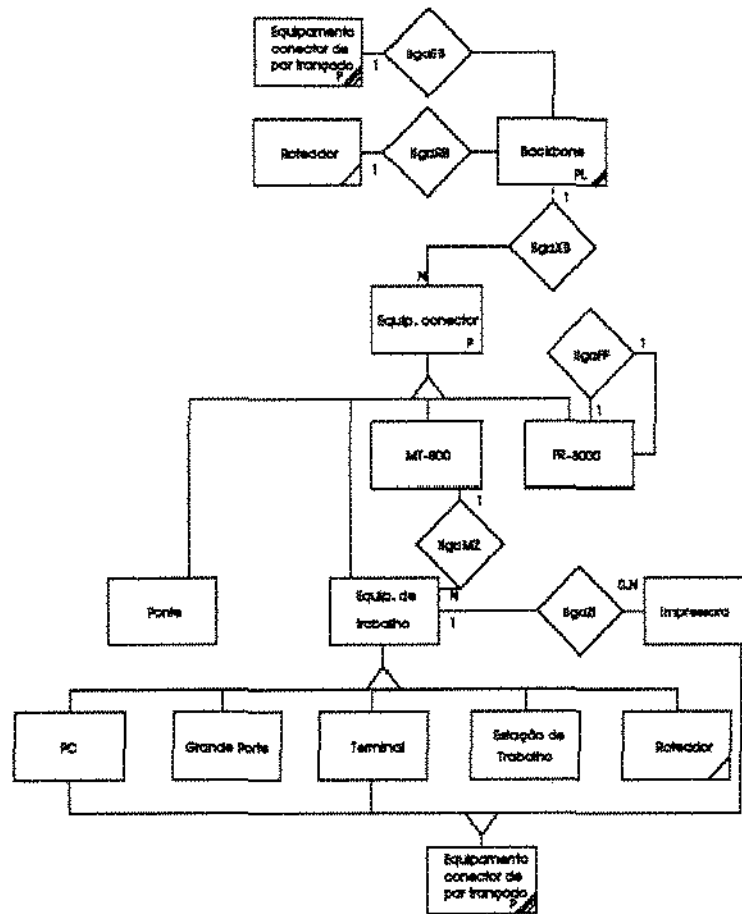


Figura 4.15: Modelo E-R da rede departamental da UNINet.

- roteador - PC ou estação de trabalho que possui um programa para rotear pacotes;
 - roteador_identificador: inteiro;
 - roteador_tipo: cadeia (PC - ET);
 - IP: cadeia.
- Backbone - cabo Ethernet, ao qual se ligam todos os equipamentos que compõem uma subrede. Tem associado um número IP;
 - backbone_identificador: inteiro;
 - IP: cadeia;
 - material: cadeia (cabo coaxial, par trançado).
- MT-800 - multiplexador;

- MT-800_identificador: inteiro.
- FR-3000 - repetidor óptico;
 - MT-800_identificador: inteiro.
- Ponte - equipamento que serve para ligar duas subredes com diferentes camadas de enlace;
 - ponte_identificador: inteiro.
- PC;
 - PC_identificador: inteiro;
 - PC_IP: cadeia.
- Terminal;
 - terminal_identificador: inteiro;
 - terminal_IP: cadeia.
- Estação de trabalho;
 - ET_identificador: inteiro;
 - ET_IP: cadeia.
- Impressora;
 - impressora_identificador: inteiro.

Os relacionamentos são descritos a seguir:

- ligaEB (equipamento - backbone) - o equipamento terminal do par lógico pode estar conectado a uma subrede;
 - ST-500_identificador: inteiro.
- ligaRB (roteador - backbone) - um roteador se liga ao backbone através de um transceptor ST-500;
 - ST-500_identificador: inteiro.
- ligaXB (equipamento conector de par trançado - backbone) - um equipamento conector de par trançado se liga ao backbone através de um transceptor ST-500;

- ST-500_identificador: inteiro.
- ligaFF (FR-300 - FR-300) - um repetidor de fibra óptica FR-300 se liga a outro do seu mesmo tipo;
- ligaMZ (MT-800 - equipamentos de trabalho) - um multiplexador MT-800 pode conectar até 8 equipamentos de trabalho;
- ligaZI (equipamento de trabalho - Impressora) - uma impressora se conecta à rede através de um equipamento de trabalho.

As restrições ao modelo são:

- um roteador é a porta de entrada/saída da rede;
- em cada subrede podem estar conectados 62 *hosts*. Uma rede pode mapear 62 endereços;
- uma ponte conecta dois trechos de backbone, ambos fazem parte da mesma subrede, ou seja, existe apenas um roteador para a subrede;
- a impressora é controlada por um equipamento de trabalho e é enxergada pela rede inteira;
- o equipamento de trabalho ligado a rede através do MT-800 não pode ter outra forma de conexão redundante a rede.

4.4.2 OMT

O modelo orientado a objeto utilizando o método OMT produziu os diagramas ilustrados nas Figuras 4.16, 4.17 e 4.18.

Da mesma forma que no modelo E-R, nos diagramas do modelo objeto foi adicionada no canto inferior direito uma sigla informando o tipo de representação daquela classe (P - ponto, PL - polilinha ou l - linha).

No modelo de objetos do mapeamento urbano, as classes são idênticas às entidades no modelo E-R. O mesmo ocorre no modelo da rede de dutos e cabos. No modelo da rede departamental fica mais claro que para as ligações ao backbone faz-se necessário a existência de um equipamento ST-500. Para a ligação do roteador à fibra óptica o equipamento necessário é o FOT-1. Estes equipamentos não foram considerados importantes e se tornaram atributos da associação entre os dois objetos, do backbone e do equipamento.

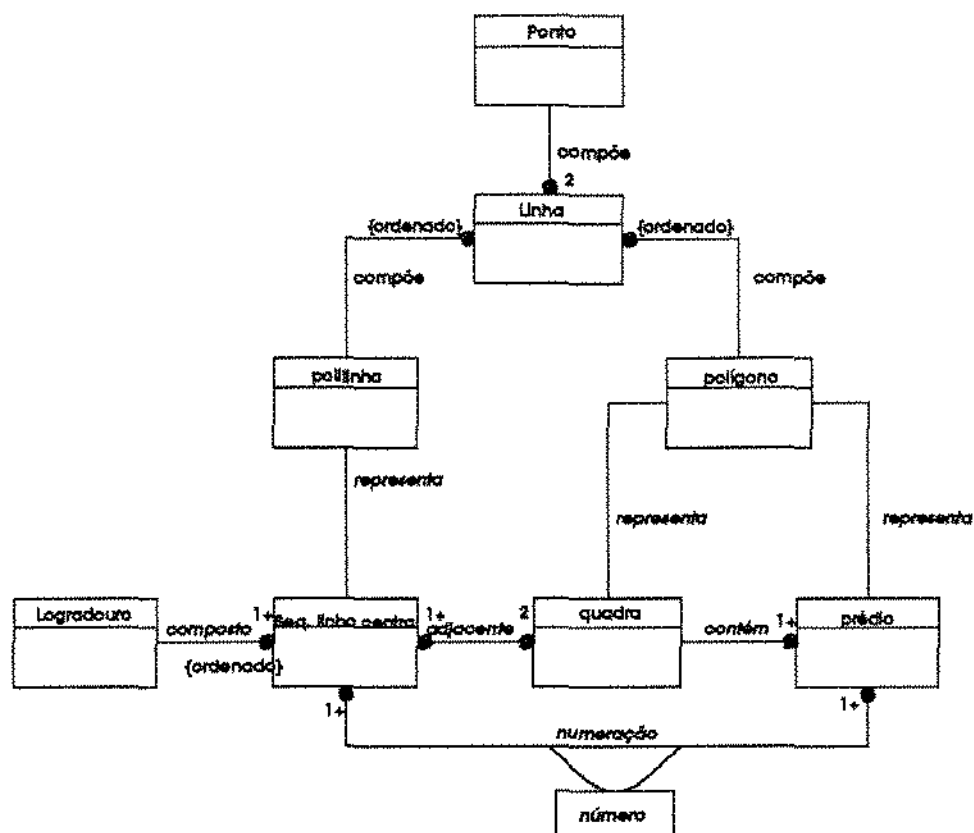


Figura 4.16: Modelo de objetos do mapeamento urbano.

4.5 Considerações sobre modelagem

Em ambos os modelos a maior preocupação era, além de capturar a idéia de uma rede computadores, a de capturar as características geométricas de um SIG. É necessário que se guarde de alguma forma essas informações geográficas a fim de que seja possível a consulta e manipulação rápidas dessas informações.

Estas informações geométricas dizem respeito ao posicionamento das entidades no espaço, suas formas, bem como o relacionamento, daí decorrente, com as outras entidades. As representações geométricas possíveis das entidades são pontos, linhas e polígonos.

Prédio e quadra são polígonos com pontos (coordenadas) que definem o seu posicionamento no espaço. Esses pontos devem ser armazenados de acordo com uma ordem pré-definida, do tipo elemento-elemento, para que se possa saber o que está dentro de um polígono, o que está posicionado à esquerda do polígono, entre algumas formas de possíveis operações. Os relacionamentos Linha Central-Adjacente-Quadradas e Quadradas-Contém-Prédio são executadas a partir das informações geométricas armazenadas.

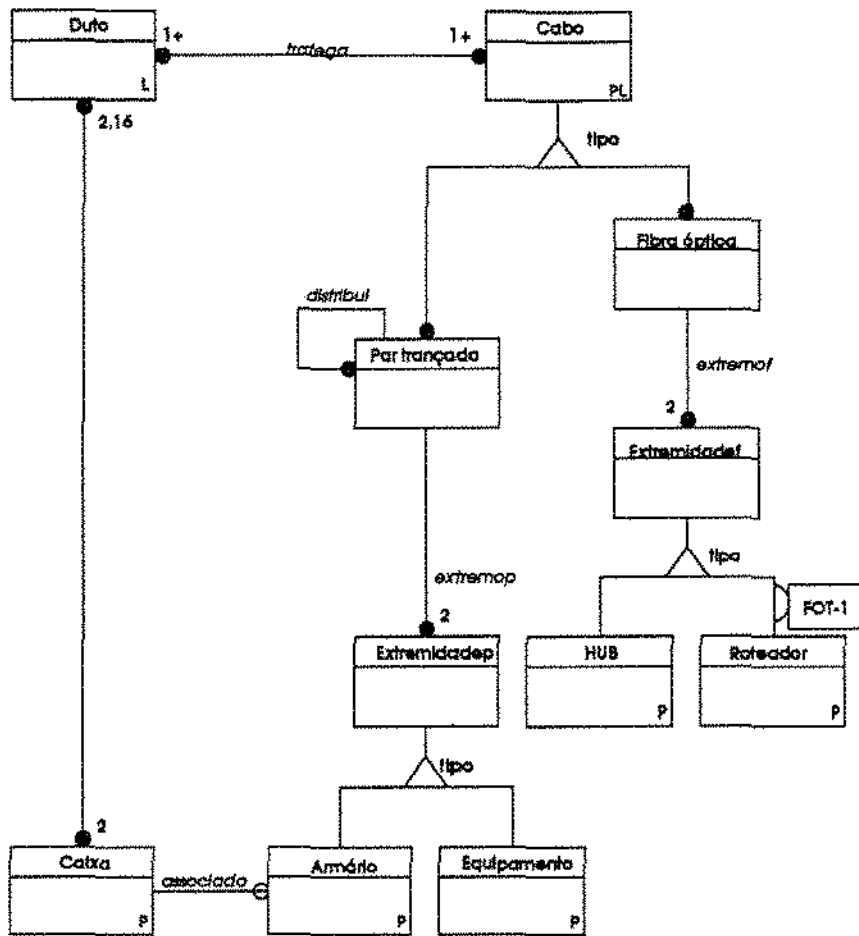


Figura 4.17: Modelo de objetos da rede de dutos e cabos.

Da mesma forma, na modelagem da rede de dutos e cabos, algumas entidades possuem representação geométrica associada. HUB, Roteador, Armário, Equipamento possuem características de ponto, Cabo possui característica de polilinha.

Pontos, linhas e polígonos não existem no SGBD, portanto, é necessário que haja um mecanismo de representação desses dados. É possível a abstração desses dados multidimensionais a partir de dados simples (inteiros e cadeias), mas é necessário um suporte às operações sobre esses dados, acesso e recuperação. Geralmente essas informações necessitam de uma grande quantidade de espaço para representação e, além disso, a determinação de relacionamentos como interseção e proximidade, por exemplo, exigem algoritmos complexos. [Sam90] discorre sobre diversas estruturas para representação de dados espaciais.

No capítulo dois foram descritas diversas formas de se armazenar informações topológicas através de vetores. Uma escolha possível é o formato arco-nó-arco, de forma a guardar o sentido e a direção do objeto. Cabos e dutos, por exemplo, talvez necessitem

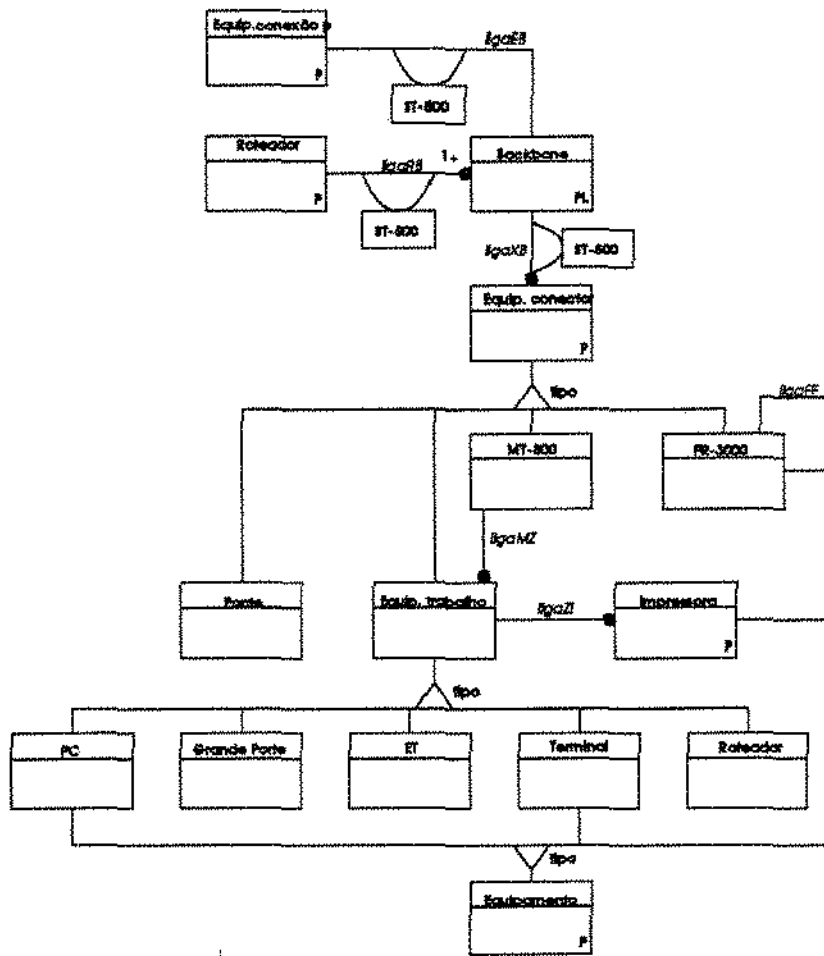


Figura 4.18: Modelo de objetos da rede departamental.

ter armazenadas informações sobre sentido e isto é possível através de arco-nó-arco. O arco seria o duto e as caixas poderiam ser os nós.

Essas informações devem estar armazenadas em estruturas separadas dentro do SGBD. Métodos de acesso foram muito estudados e não são assunto desta dissertação. Maiores informações podem ser obtidas em [Cox91].

No modelo E-R, estas informações devem ser armazenadas em tabelas separadas, dentro do próprio SGBD, com chaves para facilitar o seu rápido acesso. Linhas e polígonos são compostas por pontos. O armazenamento deve guardar a ordem dos pontos, para que se possa ter sentido/direção. As operações (proximidade, contenimento, interseção) são calculados a partir das coordenadas dos pontos. Essas operações devem ser implementadas a partir desses dados armazenados.

No método OMT uma possível forma de se armazenar tais informações seria através

da criação de objetos Ponto, Linha e Polígono, com atributos e métodos definidos. Estes métodos podem ser chamados a partir da linguagem de consultas utilizada no O₂.

Capítulo 5

Análise comparativa

Neste capítulo serão analisados comparativamente os modelos relacional e orientado a objeto descritos no capítulo anterior para o exemplo - UNINet.

Além disso, algumas considerações serão feitas com relação a consultas em SIG e a partir daí serão listadas algumas consultas, avaliadas em cada uma das implementações.

5.1 Comparação dos modelos

O esquema E-R do exemplo UNINet consta de 31 Entidades e 21 Relacionamentos. Os diagramas do MER apresentam um fácil entendimento das entidades e dos relacionamentos. As restrições possíveis de serem especificadas no modelo estão relacionadas à cardinalidade. Outras restrições foram especificadas em linguagem natural e constam do capítulo anterior.

As entidades com características geométricas (pontos, linhas e polígonos) captam toda a informação típica de um SIG. Os seus relacionamentos com as outras entidades são importantes para o sistema, pois é através desses relacionamentos que as outras entidades adquirem características geométricas-topológicas.

Os relacionamentos nem sempre são muito claros. Por exemplo, no relacionamento `par_trançado-extremidadep` não está claro que um par trançado pode ter como extremidade dois armários, dois equipamentos ou um equipamento e um armário. O mesmo pode ser dito do relacionamento `fibra_óptica-extremidadef`, onde não está claro que uma fibra óptica pode ter dois HUB ou um HUB e um roteador como extremidades. Essa informação deve estar contido no texto para que haja esclarecimento do esquema.

No relacionamento linha-polígono é necessário informar que as linhas que compõem um polígono devem ser armazenadas de forma a se guardar a ordem das linhas. O mesmo deve

ser dito com relação a polilinha-linha, as linhas que compõem uma polilinha devem ser mantidas de formar ordenada, de forma que ao se recuperar as linhas de uma determinada polilinha essas linhas sejam devolvidas segundo uma ordem pré-determinada.

O modelo OMT consta de 31 Classes e 21 Associações, idêntico ao obtido com o MER. As restrições que constam dos diagramas são de cardinalidade e ordenação de associação. Outras restrições ao modelo estão no texto em linguagem natural.

Da mesma forma que no MER, foram criadas classes ponto, linha e polígono, sendo que no OMT é possível especificar as operações (métodos) de cada classe. Nessa especificação é possível implementar outras restrições além de operações propriamente ditas. As classes ponto, linha e polígono se relacionam com as outras mediante herança. As classes que possuem características de ponto herdam todos os atributos e métodos da classe ponto e assim por diante.

As classes geométricas (ponto, linha e polígono) podem ter operações (interseção, proximidade, tamanho, área, perímetro, sobreposição) implementadas em métodos.

As associações entre classes são mais claras que os relacionamentos entre entidades do MER, devido ao maior poder de representar restrições via ordenação e através dos métodos.

Uma característica importante do OMT é que há a possibilidade de se construir objetos complexos, através de herança. A classe terminal, por exemplo, herda características de Equipamento e de Equipz, que por sua vez herdam características de ponto. Isso permite que a classe Terminal herde os atributos das classes hierarquicamente acima bem como os métodos. Se houver necessidade, a classe pode redefinir métodos/atributos herdados, desde que mantenha a mesma assinatura.

5.2 Implementação dos modelos

O modelo E-R foi implementado sobre um SGBD relacional e o modelo orientado a objeto sobre o SGBD O₂. As comparações baseiam-se apenas na capacidade de captura semântica, bem como na facilidade de elaboração de consultas.

5.2.1 Modelo Relacional

Passagem do MER para relacional

A passagem do modelo E-R para o modelo relacional (tabelas) é rápida (ver seção 3.1). Neste mapeamento as entidades e alguns relacionamentos são transformados em tabelas.

Além disso há a especificação de chaves primárias e chaves estrangeiras em cada tabela. O resultado está descrito a seguir na definição dos esquemas das tabelas em um SGBD relacional.

Implementação da geometria

As entidades Ponto, Linha, Polilinha e Polígono foram mapeadas em tabelas. Essas entidades possuem características importantes no SIG, pois é sobre elas que as operações geométricas-topológicas são executadas.

As restrições devem ser estabelecidas de alguma forma. Uma polilinha é um conjunto ordenado de linhas, ou seja, é necessário que, ao se guardar as linhas de uma polilinha, essas linha sempre respeitem a ordem de precedência estabelecida entre elas.

Pesquisar e manipular essas tabelas exigem alguma forma de indexação espacial, seja *Quad Tree* ou qualquer outro método, e esta indexação deve ser mapeada também em tabelas.

Mapeamento urbano

Ponto contém um identificador além dos atributos X e Y, que são as suas coordenadas. Linha é considerado como um segmento de reta que liga dois pontos e portanto tem um relacionamento duplo com Ponto. Desta forma Linha possui como atributos dois identificadores de Ponto (chave estrangeira) além do seu próprio identificador.

Polilinha é um conjunto de linhas conectadas. O relacionamento Linha-Polilinha é materializado em uma tabela tendo como atributos os identificadores da entidade Polilinha e Linha. A tabela Polilinha possui um único atributo, que é o seu identificador.

Polígono é um conjunto de linhas conectadas onde a última linha se conecta à primeira. O relacionamento Linha-Polígono se transforma em uma tabela tendo os identificadores de Linha e Polígono como atributos. A tabela Polígono possui o identificador da tabela linha-polígono como atributo.

As entidades Quadra e Prédio possuem uma representação de Polígono, enquanto Segmento de Linha Central pode ser representado como uma Polilinha, ou seja, um conjunto de linhas conectadas.

O relacionamento Logradouro-Segmento de Linha Central é representado mediante a inclusão da chave estrangeira *logradouro_id* na entidade Segmento de Linha Central.

Os relacionamentos Segmento de Linha Central-Adjacente-Quadra e Quadra-contém-Prédio não são materializados pois são deduzíveis a partir da posição das entidades que os representam (Polilinha e Polígono). Estes relacionamentos se tornam uma operação

topológica.

O relacionamento Numeração foi materializado em uma tabela, tendo como chaves estrangeiras os identificadores de Segmento de Linha Central e de Prédio, além do atributo número.

Rede de dutos e cabos

O relacionamento Cabo-Trafega-Duto foi materializado em uma tabela. Esta tabela tem como atributos os identificadores de Cabo e Duto. O relacionamento Duto-Caixa é mapeado através da inclusão da chaves das caixas limites do duto na entidade Duto (caixa1_id, caixa2_id).

Armário-Associado-Caixa foi mapeado através da inclusão do identificador de caixa em armário, pois todo armário possui uma caixa associado mas o contrário nem sempre é verdade.

O relacionamento distribuição (par trançado) foi mapeado através da inclusão de um atributo par_trancado_orig_id na entidade Par Trançado, para informar que par trançado que o originou.

Um par trançado tem como extremidade um Armário ou Equipamento de conexão de par trançado (PC, terminal ou impressora). Houve a criação da tabela Extremidade1 e Extremidade2, cada um com a informação do identificador do par trançado, armário e equipamento.

Da mesma forma, fibra óptica tem extremidade inicial em um HUB e a outra extremidade podendo ser um outro HUB ou um roteador. Foi criado a tabela Extremidadef que contém essa informação, através do identificador de fibra óptica, de HUB e de roteador.

Rede departamental

O relacionamento ligaRB foi resolvido através da inclusão do identificador de Roteador como chave estrangeira na tabela do Backbone.

LigaPB foi mapeado através da inclusão de dois identificadores de Backbone na entidade Ponte.

Às entidades MT-800 e FR-300 foram adicionados o identificador da entidade Backbone para que relacionamento ligaXB fosse mapeado. Liga PB foi mapeado através da inclusão de dois identificadores de Backbone na entidade Ponte. LigaMZ foi mapeado incluindo o identificador de MT-800 na entidade Equipz. LigaZI foi mapeado incluindo o identificador de Equipz na entidade Impressora.

As entidades Terminal, PC e Impressora são Equipamentos de conexão de par trançado e, por isso, possuem um atributo a mais, identificador de Equipamento, para que se possa realizar esse relacionamento. Da mesma forma, as entidades ET, Terminal, PC, Grande Ponte e Impressora possuem o identificador de Equipamento de trabalho para capturar a informação que eles são um tipo de Equipamento de trabalho.

A entidade FR-3000 se liga a outra entidade FR-3000, por isso foi incluído o identificador do outro repetidor óptico na tabela.

Esquema das tabelas

```
CREATE TABLE PONTO (
```

```
    PONTO_ID          INTEGER PRIMARY KEY,  
    X                 REAL,  
    Y                 REAL )
```

```
CREATE TABLE LINHA (
```

```
    LINHA_ID          INTEGER PRIMARY KEY,  
    PONTO1_ID         FOREIGN KEY REFERENCES PONTO (PONTO_ID),  
    PONTO2_ID         FOREIGN KEY REFERENCES PONTO (PONTO_ID) )
```

```
CREATE TABLE LINHA-POLIGONO (
```

```
    POLIGONO_ID       INTEGER FOREIGN KEY REFERENCES POLIGONO,  
    LINHA_ID          INTEGER FOREIGN KEY REFERENCES LINHA,  
    CONSTRAINT LPOLIGONO_PK PRIMARY KEY ( POLIGONO_ID, LINHA_ID ) )
```

```
CREATE TABLE LINHA-POLILINHA (
```

```
    POLILINHA_ID      INTEGER FOREIGN KEY REFERENCES POLILINHA,  
    LINHA_ID          INTEGER FOREIGN KEY REFERENCES LINHA,  
    CONSTRAINT LPOLILINHA_PK PRIMARY KEY (POLILINHA_ID, LINHA_ID) )
```

```
CREATE TABLE LINHA_CENTRAL (
```

```
    LINHA-CENTRAL_ID  INTEGER PRIMARY KEY,  
    LOGRADOURO_ID     INTEGER FOREIGN KEY REFERENCES LOGRADOURO,  
    POLILINHA_ID      INTEGER FOREIGN KEY REFERENCES POLILINHA )
```

```
CREATE TABLE LOGRADOURO (
```

```
    LOGRADOURO_ID     INTEGER PRIMARY KEY,  
    LINHA_CENTRAL_ID  INTEGER FOREIGN KEY REFERENCES LINHA_CENTRAL )
```

```
CREATE TABLE QUADRA (
    QUADRA_ID          INTEGER PRIMARY KEY,
    POLIGONO_ID       INTEGER FOREIGN KEY REFERENCES POLIGONO )

CREATE TABLE PREDIO (
    PREDIO_ID         INTEGER PRIMARY KEY,
    POLIGONO_ID       INTEGER FOREIGN KEY REFERENCES POLIGONO )

CREATE TABLE NUMERACAO
    LINHA_CENTRAL_ID  INTEGER FOREIGN KEY REFERENCES LINHA_CENTRAL,
    PREDIO_ID         INTEGER FOREIGN KEY REFERENCES PREDIO,
    NUMERO            INTEGER,
    NUMERACAO_PK     PRIMARY KEY ( LINHA_CENTRAL_ID, PREDIO_ID ) )

CREATE TABLE ARMARIO (
    ARMARIO_ID        INTEGER PRIMARY KEY,
    CAP_MAX_CABO      INTEGER,
    CAIXA_ID          INTEGER FOREIGN KEY REFERENCES CAIXA,
    PONTO_ID          INTEGER FOREIGN KEY REFERENCES PONTO )

CREATE TABLE BACKBONE (
    BACKBONE_ID       INTEGER PRIMARY KEY,
    BACKBONE_IP       INTEGER,
    MATERIAL           CHARACTER (10),
    ROTEADOR_ID       INTEGER FOREIGN KEY REFERENCES ROTEADOR,
    POLILINHA_ID      INTEGER FOREIGN KEY REFERENCES POLILINHA )

CREATE TABLE CABO (
    CABO_ID           INTEGER PRIMARY KEY,
    ESPESSURA        INTEGER,
    POLILINHA_ID      INTEGER FOREIGN KEY REFERENCES POLILINHA )

CREATE TABLE CAIXA (
    CAIXA_ID          INTEGER PRIMARY KEY,
    PONTO_ID          INTEGER PRIMARY KEY )

CREATE TABLE DUTO (
    DUTO_ID           INTEGER PRIMARY KEY,
```

```
ESPESSURA      INTEGER,
MATERIAL        CHARACTER (10),
CAP_MAX_CABO    INTEGER,
CAIXA1_ID       INTEGER FOREIGN KEY REFERENCES CAIXA,
CAIXA2_ID       INTEGER FOREIGN KEY REFERENCES CAIXA,
LINHA_ID        INTEGER FOREIGN KEY REFERENCES LINHA )
```

```
CREATE TABLE EQUIPAMENTO_CONEX_PT (
  EQUIPAMENTO_ID INTEGER PRIMARY KEY,
  TIPO           INTEGER,
  BACKBONE_ID    INTEGER FOREIGN KEY REFERENCES BACKBONE,
  ST_500_ID      INTEGER,
  PONTO_ID       INTEGER FOREIGN KEY REFERENCES PONTO )
```

```
CREATE TABLE EQUIPAMENTO_TRABALHO (
  EQUIPZ_ID      INTEGER PRIMARY KEY,
  TIPO           INTEGER,
  ST_500_ID      INTEGER,
  MT_800_ID      INTEGER FOREIGN KEY REFERENCES MT_800,
  BACKBONE_ID    INTEGER FOREIGN KEY REFERENCES BACKBONE,
  PONTO_ID       INTEGER FOREIGN KEY REFERENCES PONTO )
```

```
CREATE TABLE ET (
  ET_ID          INTEGER PRIMARY KEY,
  ET_IP          INTEGER,
  EQUIPZ_ID      INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_TRABALHO,
  EQUIPAMENTO_ID INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_CONEX_PT,
  ROTEADOR_ID    INTEGER FOREIGN KEY REFERENCES ROTEADOR )
```

```
CREATE TABLE FIBRA_OPTICA (
  FIBRA_ID       INTEGER PRIMARY KEY,
TIPO            INTEGER,
  CABO_ID        INTEGER FOREIGN KEY REFERENCES CABO )
```

```
CREATE TABLE FR_3000 (
  FR_300_ID      INTEGER PRIMARY KEY,
  BACKBONE_ID    INTEGER FOREIGN KEY REFERENCES BACKBONE,
  FR_300_ID2     INTEGER FOREIGN KEY REFERENCES FR_3000,
  PONTO_ID       INTEGER FOREIGN KEY REFERENCES PONTO )
```

```
CREATE TABLE GRANDE_PORTE (
    GRANDE_ID      INTEGER PRIMARY KEY,
    GRANDE_IP      INTEGER,
    TIPO           INTEGER,
    EQUIPZ_ID      INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_TRABALHO )

CREATE TABLE HUB (
    HUB_ID         INTEGER PRIMARY KEY,
    PONTO_ID       INTEGER FOREIGN KEY REFERENCES PONTO )

CREATE TABLE IMPRESSORA (
    IMP_ID         INTEGER PRIMARY KEY,
    EQUIPZ_ID      INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_TRABALHO,
    PONTO_ID       INTEGER FOREIGN KEY REFERENCES PONTO,
    EQUIPAMENTO_ID INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_CONEX_PT )

CREATE TABLE MT_800 (
    MT_800_ID      INTEGER PRIMARY KEY,
    BACKBONE_ID    INTEGER FOREIGN KEY REFERENCES BACKBONE,
    ST_500_ID      INTEGER,
    PONTO_ID       INTEGER FOREIGN KEY REFERENCES PONTO )

CREATE TABLE PAR_TRANCADO (
    PAR_TRAN_ID    INTEGER PRIMARY KEY,
    PAR_TRAN_ORIG_ID INTEGER FOREIGN KEY REFERENCES PAR_TRANCADO,
    CABO_ID        INTEGER FOREIGN KEY REFERENCES CABO )

CREATE TABLE PC (
    PC_ID          INTEGER PRIMARY KEY,
    PC_IP          INTEGER,
    EQUIPZ_ID      INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_CONEX_PT,
    EQUIPAMENTO_ID INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_TRABALHO,
    ROTEADOR_ID    INTEGER FOREIGN KEY REFERENCES ROTEADOR )

CREATE TABLE POLIGONO (
    POLIGONO_ID    INTEGER PRIMARY KEY )

CREATE TABLE POLILINHA (
```



```
POLILINHA_ID      INTEGER PRIMARY KEY )

CREATE TABLE PONTE (
  PONTE_ID          INTEGER PRIMARY KEY,
  BACKBONE_ID      INTEGER FOREIGN KEY REFERENCES BACKBONE,
  ST_500_ID        INTEGER,
  PONTO_ID         INTEGER FOREIGN KEY REFERENCES PONTO )

CREATE TABLE ROTEADOR (
  ROTEADOR_ID      INTEGER PRIMARY KEY,
  FOT1_ID          INTEGER,
  TIPO             STRING )

CREATE TABLE TERMINAL (
  TERMINAL_ID      INTEGER PRIMARY KEY,
  TERMINAL_IP      INTEGER,
  EQUIPZ_ID        INTEGER FOREIGN KEY REFERENCES EQUIP_TRABALHO,
  EQUIPAMENTO_ID  INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_CONEX_PT )

CREATE TABLE TRAFEGA (
  DUTO_ID          INTEGER FOREIGN KEY REFERENCES DUTO,
  CABO_ID          INTEGER FOREIGN KEY REFERENCES CABO,
  TRAFEGA_PK PRIMARY KEY ( DUTO_ID, CABO_ID ) )

CREATE TABLE EXTREMIDADEP (
  PAR_TRAN_ID      INTEGER FOREIGN KEY REFERENCES PAR_TRANCADO,
  EQUIPAMENTO_ID  INTEGER FOREIGN KEY REFERENCES EQUIPAMENTO_CONEX_PT,
  ARMARIO_ID       INTEGER FOREIGN KEY REFERENCES ARMARIO )

CREATE TABLE EXTREMIDADEF (
  FIBRA_ID         INTEGER FOREIGN KEY REFERENCES FIBRA_OPTICA,
  HUB_ID           INTEGER FOREIGN KEY REFERENCES HUB,
  ROTEADOR_ID      INTEGER FOREIGN KEY REFERENCES ROTEADOR )
```

5.2.2 Modelo orientado a objeto

Passagem do OMT para o O_2

Existem várias regras[DA95] para a passagem do OMT para o O_2 . Serão citadas as regras que foram aplicadas no exemplo.

Uma classe C do modelo de objetos do OMT gera uma definição de classe de mesmo nome no modelo O_2 , denominado C' . Um atributo A da classe C corresponde diretamente a um atributo de mesmo nome na classe C' , denominado A' . Uma operação O da classe C gera uma declaração de método de mesmo nome na classe C' , denominado O' . A declaração deste método pode ser feita dentro da classe ou fora, após a declaração de todas as classes. Neste caso, deve ser especificada a cláusula "in class nome da classe".

Uma generalização entre uma superclasse C e as subclasses C_1, C_2, \dots, C_n é mapeada em $n+1$ definições de classe, denominadas $C', C'_1, C'_2, \dots, C'_n$.

O mapeamento da superclasse C na classe C' segue as mesmas regras definidas anteriormente. Se houver a especificação de um discriminador do tipo enumerado no OMT, ele será um atributo adicional da classe C' .

O mapeamento de cada subclasse C_i na classe C'_i segue as mesmas regras definidas anteriormente. A cláusula "inherits nome da superclasse" deve ser especificada em cada subclasse, para estabelecer que a classe C'_i herda as propriedades (atributos e métodos) da sua superclasse.

No O_2 , as associações entre as classes são representadas através de relacionamentos de herança e/ou composição. Sejam C'_1 e C'_2 duas classes no modelo OMT relacionadas entre si através de uma associação AS . Sejam C'_1 e C'_2 as duas classes correspondentes no O_2 . O mapeamento da associação AS implica na inserção de um atributo adicional na classe C'_1 , denominado AA' . Dependendo da multiplicidade de AS , do fato de AS possuir atributos e da especificação da restrição ordenação, diferentes situações podem acontecer.

Considere-se que AS não possui atributos.

- AS representa uma associação na qual a classe C_1 está associada a nenhuma ou a uma única instância de C_2 . Neste caso, AA' constitui uma referência a esta classe.
- AS representa uma associação na qual a classe C_1 está associada a várias instâncias de C_2 e a restrição de ordenação não foi especificada. Neste caso AA' constitui uma referência a um conjunto de instâncias desta classe (construtor *set*).
- AS representa uma associação na qual a classe C_1 está associada a várias instâncias de C_2 e a restrição de ordenação foi especificada. Neste caso, AA' constitui uma referência a um conjunto ordenado de instâncias desta classe (construtor *list*).

Se *AS* possuir atributos, o atributo *AA'* será constituído a partir de uma tupla. O nome desta tupla será formado pela concatenação do string "tupla_" com o nome da classe *C2*. Se *AS* possuir nome, ele deverá ser usado como nome do atributo *AA'*.

- *AS* representa uma associação na qual a classe *C1* está associada a uma instância de *C2*. Neste caso, *AA'* é declarado como uma tupla de $i+1$ outros atributos. Os primeiros i atributos correspondem aos atributos de *AS* e possuem o mesmo nome e o mesmo tipo de dado que estes. Já o último é um atributo adicional que referencia a classe em *C'2*.
- *AS* representa uma associação na qual a classe *C1* está associada a várias instâncias de *C2* e a restrição de ordenação não foi especificada. Neste caso, *AA'* é declarado como um conjunto de tuplas de $i+1$ atributos. Os primeiros i atributos correspondem aos atributos de *AS* e possuem o mesmo nome e o mesmo tipo de dado que estes. Já o último é um atributo adicional que referencia a classe em *C'2*. O construtor a ser utilizado é *set*.
- *AS* representa uma associação na qual a classe *C1* está associado a várias instâncias de *C2* e a restrição de ordenação foi especificado. Neste caso, *AA'* é declarado como um conjunto de tuplas de $i+1$ atributos. Os primeiros i atributos correspondem aos atributos de *AS* e possuem o mesmo nome e o mesmo tipo de dado que estes. Já o último é um atributo adicional que referencia a classe em *C'2*. O construtor a ser utilizado é *list*.

Implementação da geometria

Ponto, Linha e Polígono são classes no modelo OMT. Sobre elas estão especificadas as principais operações de SIG. Essas operações tornam-se métodos no OMT. Polígono e Polilinha são conjuntos ordenados de linha e para capturar esta restrição essas classes são implementadas como *list* de Linhas.

Da mesma forma que no MER, a indexação espacial será realizadas através de *Quad Tree*, *Árvore R* ou qualquer outra forma, desde seja implementado dentro dos métodos e classes do modelo. A capacidade de especificar as operações em métodos é uma grande vantagem do OMT sobre o MER.

Mapeamento Urbano, rede de dutos e cabos e rede departamental

Todas as classes no OMT deram origem a classes no O_2 . As associações entre classes foram mapeadas através da inclusão de referências nas classes que faziam parte da associação, seguindo as regras especificadas na subseção 2.2.1.

Esquema

O esquema gerado foi o seguinte:

```
class Ponto public type
  tuple(x: real,
        y: real)
end;
class Linha inherit Object public type
  tuple(a: Ponto,
        b: Ponto)
end;
class Poligono inherit Object public type
  tuple(linhas: list(Linha))
end;
class Polilinha inherit Object public type
  tuple(linhas: list(Linha))
end;
class Logradouro inherit Object private type
  tuple(logradouro_nome: string,
        lin_central: list(Linha_central),
        logradouro_id: integer)
end;
class Linha_central inherit Polilinha public type
  tuple(linha_central_id: integer,
        endereco: set( predio_id: integer))
end;
class Quadra inherit Poligono public type
  tuple(quadra_id: integer)
end;
class Predio inherit Poligono public type
  tuple(predio_id: integer,
        endereco: set(tuple( numero: integer,
                              rua: Linha_central)))
end;
class Duto inherit Linha public type
  tuple(duto_id: integer,
        espessura: real,
        material: string,
        cap_max_cabos: integer,
```

```
        cap_inst_cabos: integer,
        caixa1: Caixa,
        caixa2: Caixa,
        ref_cabo: set(Cabo))
end;
class Cabo inherit Polilinha public type
    tuple(cabo_id: integer,
          espessura: real,
          tipoc: inteiro,
          trafega: list(Duto))
end;
class Caixa inherit Ponto public type
    tuple(caixa_id: integer,
          dutos: list(Duto))
end;
class Par_tranc inherit Cabo public type
    tuple(par_tranc_id: integer,
          distribui: set(Par_tranc),
          extremop: Extremidadep)
end;
class Extremidadep inherit Ponto public type
    tuple(extremid:set(Par_tranc),
          tipop: integer)
end;
class Armario inherit Extremidadep public type
    tuple(armario_id: integer,
          cap_max_conexao: integer,
          cap_inst_conexao: integer,
          caixa_assoc: Caixa)
end;
class Equipamento inherit Extremidadep public type
    tuple(backbone: Backbone,
          ST_500: integer,
          equip_tipo: string,
          tipoe: integer)
end;
class Fibra_optica inherit Cabo public type
    tuple(fibra_id: integer,
          tipo: inteiro,
```

```
        extremof: Extremidadef)
end;
class Extremidadef inherit Ponto public type
    tuple(extremidf:set(Fibra_optica),
        tipof: integer,
        FOT_1: integer)
end;
class HUB inherit Extremidadef public type
    tuple(HUB_id: integer)
end;
class Roteador inherit Equip_trabalho, Extremidadef public type
    tuple(roteador_id: integer,
        tipo: string,
        ref_Backbone: set(Backbone))
end;
class Equip_conexao inherit Ponto public type
    tuple(IP: integer,
        tipox: integer,
        tuple(ligaXB: Backbone,
            ST_500: integer))
end;
class Equip_trabalho inherit Equip_conexao public type
    tuple(imps: set(Impressora),
        tipoz: integer)
end;
class PC inherit Equip_trabalho, Equipamento, Roteador public type
    tuple(PC_id: integer)
end;
class Grande_porte inherit Equip_trabalho public type
    tuple(grande_id: integer)
end;
class ET inherit Equip_trabalho, Equipamento, Roteador public type
    tuple(ET_id: integer)
end;
class Terminal inherit Equip_trabalho, Equipamento public type
    tuple(terminal_id: integer)
end;
class Impressora inherit Equipamento public type
    tuple(impressora_id: integer,
```

```
        ref_Equipz: Equip_trabalho)
end;
class Ponte inherit Equip_conexao public type
    tuple(ponte_id: integer,
        backbone: Backbone,
        ST_500: integer)
end;
class MT_800 inherit Equip_conexao public type
    tuple(MT_800_id: integer,
        equip: set(Equip_trabalho))
end;
class FR_3000 inherit Equip_conexao public type
    tuple(FR_3000_id: integer,
        FR: FR_3000)
end;
class Backbone inherit Polilinha public type
    tuple(backbone_id: integer,
        material: string,
        IP: integer,
        tuple(roteador: Roteador,
            ST_500: integer))
end;
```

5.2.3 Análise das implementações

No modelo relacional algumas relações deram origem a tabelas, o que originou uma maior quantidade de tabelas geradas do que classes no O_2 . Para efeito de mapear os relacionamentos houve a necessidade de se definir uma chave primária para cada entidade bem como incluir chaves estrangeiras em algumas entidades, de acordo com a necessidade.

As Entidades significativas para SIG (Pontos, Linhas, Polilinhas e Polígonos) foram definidas uma em função da outra. Linha se relaciona com Ponto, Polilinha é um conjunto de Linhas e foi criado uma tabela Linha_Polilinha. O mesmo aconteceu com Polígono, que teve o relacionamento com Linha materializado em uma tabela Polígono_Linha.

No caso do O_2 , houve o mesmo procedimento, mas foi implementado de maneira diferente. A classe Linha faz referência à classe Ponto por meio de dois ponteiros. A classe Polilinha (Polígono) faz referência a uma lista de Linhas, o que é muito interessante, já que em lista a ordem dos elementos é importante.

O número de classes no esquema O_2 é relativamente pequeno, porque a criação de classes se beneficia de construtores de objetos complexos, tais como hierarquia de classes, hierarquia de atributos, hierarquia de composições, hierarquias de tuplas e conjuntos de atributos [Kam96].

O O_2 implementa herança adicionando os atributos da classe superior na classe inferior. Isto é transparente para o usuário, ou seja, basta colocar que a classe é uma especialização de uma outra classe. Quadra e Prédio, por exemplo, são polígonos e, na implementação, eles terão os atributos de Polígono incluídos nas suas estruturas. Se houver herança múltipla, os atributos idênticos serão adicionados com a mudança de nome, incluindo ao final a classe de onde está sendo herdado.

Se os atributos herdados de duas classes são de uma outra classe pai, de onde essas duas herdaram e se esse atributo não sofreu alteração nessas duas classes, então é o caso de herança repetida [Tec93]. Neste caso o atributo será herdado apenas uma vez. Os atributos herdados podem também ser apagados ou ser sobrepostos por uma nova definição.

Os métodos são herdados da mesma forma. As classes herdam todos os métodos de suas classes pai. Se houver herança múltipla de métodos de nomes idênticos, um método único será adicionado. Este método precisará ter a mesma assinatura dos métodos das classes pai. Se não for possível a classe não poderá ser criada. Este problema pode ser resolvido renomeando-se os métodos herdados na sub-classe.

Esses casos de herança múltipla podem ser resolvidos trocando-se os nomes dos atributos/métodos idênticos. Outra forma é simplesmente remover os atributos/classes que não se fazem necessários.

O que se observa é que a implementação do O_2 foi mais direta e compacta. Não houve a necessidade de se criar classes a mais que o modelo OMT estipulava, o que não ocorreu no caso do Modelo E-R para o relacional.

Tanto o O_2 quanto o SGBD relacional permite a definição de chaves (indexação) para facilitar a recuperação de objetos/tuplas, evitando desta forma a pesquisa seqüencial.

5.3 Consultas

Existem vários trabalhos que propõem classificação de consultas em SIG. O objetivo destas classificações é obter um conjunto mínimo de consultas, que combinadas, possam representar todas as formas de pesquisa espacial. [Gat91] classifica as consultas em descritivas e analíticas. [dFM93] classifica em consultas topológicas, de conjunto e métricas.

No presente trabalho, as consultas serão classificadas em: descritivas, topológicas

(adjacência, borda e co-borda) e métricas (mínimos/máximos e raios de abrangência).

Consultas descritivas podem ser consideradas como as normais em um SGBD, que buscam obter informação armazenada em um SGBD.

Consultas métricas buscam obter distância, perímetro, comprimento e área [Cif95]. Basicamente, as operações de distância podem ser euclidiana ou a soma de segmentos de uma rede. As operações de comprimento devem calcular a extensão de um objeto linear. As operações de raios de abrangência são operações métricas que calculam um raio, a partir de um determinado ponto e dentro de um intervalo especificado.

Consultas topológicas envolvem relacionamentos topológicos entre os objetos: cruzamento, intersecção, adjacência, inclusão e continência [Cif95].

Nas consultas, a seguir descritas, foi considerado como já existente algumas funções: *IS_IN*, *CROSSING*, *ADJACENT*, *PERIMETER*, *AREA*, *LENGTH*.

A seguir são descritas algumas consultas usando a linguagem SQL[MS93] e a linguagem de consultas do O_2 [Tec93].

5.3.1 Consultas descritivas

- obter todas as informações sobre uma caixa

(SQL)	(O_2)
SELECT *	SELECT c
FROM Caixa	FROM c IN Caixa
WHERE caixa_id = X	WHERE c.caixa_id = X

Nesta consulta serão pesquisadas a tabela Caixa e a classe Caixa. A única diferença é que no O_2 haverá uma maior quantidade de informações recuperadas, já que a classe Caixa é um objeto complexo, possuindo mais informações que a tabela Caixa. A consulta no SQL deveria ser mais elaborada para obter a mesma quantidade de informações que a consulta no O_2 . A chave de pesquisa em ambas implementações é o identificador de Caixa.

- obter as informações sobre os dutos que tem como extremidade uma determinada caixa

(SQL)	(O_2)
SELECT *	SELECT d
FROM Duto	FROM d IN Duto
WHERE caixa1_id = X .OR. caixa2_id = X	WHERE d.caixa1.caixa_id = X OR d.caixa2.caixa_id = X

Nesta consulta a tabela/classe a ser pesquisada é Duto. Da mesma forma que a consulta anterior, o O₂ possui mais informações de Duto a serem recuperadas do que no SGBD relacional.

- qual o roteador de um backbone

<pre>(SQL) SELECT roteador_id FROM backbone WHERE backbone_id = X</pre>	<pre>(O₂) SELECT b.roteador FROM b IN Backbone WHERE b.backbone_id = X</pre>
---	---

Nesta consulta, o O₂ e o SGBD relacional precisam acessar o objeto/tabela Backbone para realizar a consulta.

- quais as impressoras que estão em uma determinada rede

<pre>(SQL) SELECT impressora_id FROM impressora, equipamento_trabalho WHERE impressora.equipamento_id = equi- pamento_trabalho.equipamento_id .AND. equipa- mento_trabalho.backbone_id = X</pre>	<pre>(O₂) SELECT imps FROM e_trab IN Equip_trabalho, e_conex IN Equip_conexao WHERE e.ligaXB.backbone_id = X</pre>
--	---

O O₂ precisa acessar apenas a classe Impressora enquanto o relacional precisa pesquisar duas tabelas: impressora e equipamento.

- quantos backbones existem a partir de um determinado roteador

<pre>(SQL) SELECT count (*) FROM backbone WHERE roteador_id = X</pre>	<pre>(O₂) SELECT b FROM b IN Backbone WHERE b.roteador.roteador_id = X</pre>
---	---

O O₂ e o SGBD relacional precisam acessar uma classe/tabela (Backbone).

- selecione todos os roteadores que são PC

<pre>(SQL) SELECT roteador_id FROM roteador WHERE roteador_tipo = "PC"</pre>	<pre>(O₂) SELECT r.roteador_id FROM r IN Roteador WHERE r.tipo = "PC"</pre>
--	--

Em ambos os SGBD é necessário acessar apenas uma classe/tabela.

- a que equipamento de trabalho está ligado uma determinada impressora

<pre>(SQL) SELECT equipz.id FROM impressora WHERE imp.id = X</pre>	<pre>(O₂) SELECT i.ref_Equipz FROM i IN Impressora WHERE i.impressora.id = X</pre>
--	---

No O₂ e no SGBD relacional se faz necessário a pesquisa de apenas uma class/tabela.

As consultas descritivas são comuns em qualquer banco de dados. Houve poucas diferenças entre as duas implementações, em geral o percurso no O₂ exigiu uma classe a mais para ser pesquisada do que o relacional exigiu em número de tabelas.

5.3.2 Consultas topológicas

São descritas, a seguir, algumas consultas topológicas:

- que prédios estão contidos em uma determinada quadra

<pre>(SQL) SELECT predio.id FROM predio, quadra WHERE predio.poligono.id IS_IN quadra.poligono.id .AND. quadra.id = X</pre>	<pre>(O₂) SELECT q.predio.id FROM q IN Quadra, p IN Predio WHERE q IS_IN p AND q.quadra.id = X</pre>
---	---

- em que quadra está localizado um determinado prédio

<pre>(SQL) SELECT quadra.id FROM predio, quadra WHERE predio.poligono.id IS_IN quadra.poligono.id .AND. predio.id = X</pre>	<pre>(O₂) SELECT q.quadra.id FROM q IN Quadra, p IN Predio WHERE p IS_IN q AND p.predio.id = x</pre>
---	---

A operação *IS_IN* implica em acessos a tabelas/classes. Esta consulta, no modelo relacional, implica percorrer das tabelas prédio, quadra, polígono, linha, ponto, linha-polígono. No O₂ esta operação pode ser implementada como um método sobre a classe polígono, que será herdada pelas classes prédio e quadra. A operação no O₂ envolve apenas as classes Quadra e Prédio, já que são estruturas complexas.

- quais as linhas centrais adjacentes a uma quadra

<pre>(SQL) SELECT linha-central.id FROM linha-central, quadra WHERE linha-central.polilinha.id ADJACENT qua- dra.poligono.id .AND. quadra.id = X</pre>	<pre>(O₂) SELECT l.linha_central FROM l IN Linha_central, q IN Quadra WHERE l ADJACENT q AND q.quadra_id = X</pre>
--	---

A operação *ADJACENT* implica no acesso a tabelas ponto, linha, linha-polilinha, polígono, linha-polígono, além, das tabelas linha-central e quadra. O método *ADJACENT* no O_2 pode ser implementado sobre a classe linha. A consulta irá acessar apenas as classes linha_central e quadra.

- que ruas são cortadas por determinado

<pre>(SQL) SELECT logradouro_id FROM linha-central, duto WHERE duto.linha_id CROSSING linha-central.polilinha_id .AND. duto.id = X</pre>	<pre>(O₂) SELECT l.logradouro_id FROM l IN Logradouro, d IN Duto WHERE l.logradouro CROSSING d.duto AND d.duto_id = X</pre>
--	--

A operação *CROSSING* implica no acesso as tabelas ponto, linha, linha-polilinha, além de linha-central e duto. No O_2 , o método *CROSSING* pode ser implementado sobre as classes linha e polígono. Nesta consulta apenas as classes logradouro e duto são acessados.

- que caixas estão localizadas em um determinado logradouro

<pre>(SQL) SELECT caixa_id FROM logradouro, caixa WHERE caixa_id IS_IN linha- central.polilinha_id .AND. linha- central.logradouro_id = X</pre>	<pre>(O₂) SELECT c.caixa_id WHERE c IN Caixa, l IN Logradouro WHERE c IS_IN l AND l.logradouro_id = X</pre>
--	--

No relacional a consulta exigirá o percurso sobre as tabelas caixa, logradouro, polígono, linha, linha-polígono e ponto. No O_2 apenas as classes já relacionadas (caixa e logradouro).

- selecione todos os roteadores localizados em um determinado prédio

<pre>(SQL) SELECT roteador_id FROM roteador, predio WHERE roteador_id IS_IN pre- dio.poligono_id .AND. predio_id = X</pre>	<pre>(O₂) SELECT r.roteador_id FROM r IN Roteador, p IN Predio WHERE r IS_IN p AND p.predio_id = X</pre>
--	---

As tabelas roteador, prédio, ponto, linha, linha-polígono e polígono terão de ser pesquisadas no relacional. No O₂ é suficiente as classes roteador e prédio.

- selecione todos os equipamentos de trabalho localizados em um determinado prédio

<pre>(SQL) SELECT equipz_id FROM equipamento_trabalho, predio WHERE equipz_id IS_IN predio.poligono_id .AND. pre- dio.id = X</pre>	<pre>(O₂) SELECT e.equipz_id FROM e IN Equip_trabalho, p IN Predio WHERE e IS_IN p AND p.predio_id = X</pre>
--	---

- em que prédio está localizado um determinado equipamento de trabalho

<pre>(SQL) SELECT predio_id FROM equipamento_trabalho, predio WHERE equipz_id IS_IN pre- dio.poligono_id .AND. equipz_id = X</pre>	<pre>(O₂) SELECT p.predio_id FROM p IN Predio, e IN Equip_trabalho WHERE e IS_IN p AND e.equipz_id = X</pre>
--	---

- em qual rua está localizada uma caixa

<pre>(SQL) SELECT logradouro_id FROM logradouro, caixa WHERE caixa_id IS_IN lo- grradouro.linha_id .AND. caixa_id = X</pre>	<pre>(O₂) SELECT l.logradouro_id FROM l IN Logradouro, c IN Caixa WHERE c IS_IN l AND c.caixa_id = X</pre>
---	---

O que vale ser ressaltado é que as operações (*IS_IN*, *ADJACENT*, *CROSSING*) devem estar implementadas de alguma forma no SQL e no O₂Query. O SQL precisa ter extensões que tratem disso e o O₂ pode ter estas operações como métodos de classes. As tabelas/classes a serem acessadas por esses métodos devem ser Ponto, Linha, Polilinha e Polígono.

5.3.3 Consultas métricas

- qual a distância de um determinado prédio a outro prédio

<pre>(SQL) SELECT DISTANCE(P1.predio, P2.predio) FROM predio P1, predio P2 WHERE P1.predio_id = X1 .AND. P2.predio_id = X2</pre>	<pre>(O₂) SELECT DISTANCE(p1, p2) FROM p1 IN Predio, p2 IN Predio WHERE p1.predio_id = X1 AND p2.predio_id= X2</pre>
---	---

No relacional, a operação *DISTANCE* irá acessar as tabelas polígono, linha-polígono, linha e ponto, além de prédio. No *O₂*, o método *DISTANCE* deve ser implementado sobre a classe ponto e a consulta irá acessar a classe prédio.

- calcular o perímetro e a área de uma determinada quadra

<pre>(SQL) SELECT AREA(quadra) FROM quadra WHERE quadra_id = X</pre>	<pre>(O₂) SELECT AREA(q) FROM q IN Quadra WHERE q.quadra_id = X</pre>
--	--

No relacional, a operação *AREA* irá pesquisar as tabelas polígono, linha-polígono, linha e ponto, além da tabela quadra. No *O₂*, o método *AREA* deve ser implementado sobre a classe polígono. A consulta pesquisar a classe quadra.

<pre>(SQL) SELECT PERIMETER(quadra) FROM quadra WHERE quadra_id = X</pre>	<pre>(O₂) SELECT PERIMETER(q) FROM q IN Quadra WHERE q.quadra_id = X</pre>
---	---

No relacional, a operação *PERIMETER* irá pesquisar as tabelas polígono, linha-polígono, linha e ponto. No *O₂* o método *PERIMETER* deve ser implementado sobre a classe polígono.

- qual o comprimento de um determinado cabo

<pre>(SQL) SELECT LENGTH(cabo) FROM cabo WHERE cabo_id = X</pre>	<pre>(O₂) SELECT LENGTH(c) FROM c IN Cabo WHERE c.cabo_id = X</pre>
--	--

No relacional, a operação *LENGTH* irá pesquisar as tabelas polilinha, linha-polilinha, linha e ponto. O método *LENGTH* deve ser implementado sobre a classe linha e polilinha.

5.3.4 Considerações

As consultas escolhidas foram simples, apenas para mostrar como seriam realizadas em cada SGBD. Os problemas existentes dizem respeito a necessidade de funções extras, no tocante às características espaciais de SIG. Em nenhum momento foi levado em consideração problemas relacionados a escala, representação em dispositivos de saída, tempo de execução, tamanho dos dados em cada SGBD, forma de armazenamento dos dados espaciais e métodos de acesso.

Mesmo assim, é possível observar o seguinte:

- os objetos/entidades complexos, como polígonos, são mais facilmente representados pelo O_2 ;
- necessidade de extensão espacial para o SQL e implementação de métodos espaciais no O_2 . Neste aspecto a modelagem orientada a objeto é mais eficiente, pois é possível implementar um novo método de acordo com a necessidade;

Capítulo 6

Conclusões

6.1 Considerações

O objetivo principal desta tese é comparar o uso dos modelos relacional e orientado a objetos na implementação de sistemas de mapeamento automático/gerência de facilidades sobre sistemas de informações geográficas. Atualmente, sistemas de informações geográficas são construídos fazendo uso de modelos de dados proprietários ou do modelo relacional. Sistemas de gerenciamento de banco de dados orientado a objetos estão sendo introduzidos no mercado mas, já há um certo tempo a técnica é vendida como mais adequada para suportar Sistemas de Informações Geográficas.

Embora não haja implementações diretas do modelo E-R, este modelo foi utilizado na fase de modelagem procurando se aproximar com o que acontece na prática - modelagem em M E-R e implementação no modelo relacional.

A passagem do modelo orientado a objeto para o SGBD O₂ foi bastante rápida, o mesmo pode ser dito do modelo E-R para o relacional. Foram criados menos objetos (classes) no orientado a objeto do que no relacional (tabelas).

Se não houvesse um SGBD OO e o mapeamento do OMT fosse em cima de um SGBD relacional, haveria a mesma dificuldade quanto a tradução. O que se pode avaliar é que o uso de software OO, mais do que a metodologia de desenvolvimento OO, é que causou o grande diferencial com relação a captura semântica.

No modelo relacional, os relacionamentos são explícitos, no sentido que se há um relacionamento haverá uma chave estrangeira em uma das tabelas. No O₂, os relacionamentos são realizados através de referência a outras entidades mediante o uso de ponteiros, e tudo nem sempre é visível para o usuário. Além disso, O₂ permite certas construções como *sets*, *lists*, que facilitam a modelagem, mas podem complicar o entendimento do esquema do

banco de dados. Isto pode é mais visível quando da construção de consultas. No percurso de algumas consultas fica visível no modelo relacional quantas tabelas se deve pesquisar. No O₂ isto nem sempre é visível.

6.2 Contribuições

Na fase de modelagem, utilizando os modelos E-R e OMT, pode-se concluir:

- o modelo E-R conseguiu capturar tão bem quanto a OMT a semântica estrutural da aplicação;
- com relação à modelagem comportamental o OMT tem os recursos adicionais de encapsulamento de métodos e herança podendo garantir melhor a integridade da aplicação, principalmente no que se refere à conectividade dos elementos da rede.

No projeto físico, utilizando o O₂ e SQL padrão, as diferen'cas entre os modelos orientado a objetos e relacional são mais salientes:

- a passagem do modelo E-R para tabelas no relacional foi simples, mas a estrutura resultante é mais pobre do que no E-R;
- a passagem do OMT para o O₂ conseguiu manter todas as características do modelo OMT;
- os métodos (operações) implementados nas classes podem ser herdados pelas subclasses, o que facilita a construção de tipos complexos;

Foram observados os tipos de problemas que podem acontecer quando da tentativa de se modelar uma aplicação SIG utilizando as técnicas de modelagem E-R e OMT. Tais problemas estão relacionados aos tipos de complexos de SIG (Pontos, Linhas e Polígonos), operações sobre esses tipos, bem como formas de armazenamento e acesso. Identificado os problemas, passou-se à fase de implementação dos esquemas desenvolvidos a fim de se verificar, agora com consultas, como poderiam ser resolvidos as operações topológicas e métricas no próprio SGBD.

As operações desses tipos devem ser implementadas de alguma forma. No modelo relacional não há possibilidade de se definir essas operações, cabendo ao usuário implementá-las de alguma forma. Existem versões de SQL com extensões espaciais, como o PSQL (*Pictorial SQL*) e o Spatial SQL [dN95]. Entretanto podem existir aplicações

com operações não cobertas por essas extensões, o que pode implicar na necessidade de implementá-las de alguma forma.

No SGBD O_2 , essas operações podem ser implementadas como métodos em O_2C e serem usadas diretamente na linguagem como uma chamada de função. Isto dá uma grande facilidade na construção de sistemas complexos que exijam operadores não convencionais.

6.3 Trabalhos futuros

Os trabalhos futuros incluem:

- avaliar as consultas utilizando a linguagem O_2C - todas as consultas deste trabalho foram implementadas usando o SQL do O_2Query . A linguagem O_2 , por ser um superconjunto de C, com características de orientação a objeto, pode facilitar a manipulação das informações;
- avaliar o desempenho com relação as implementações, avaliando tempo de resposta - neste caso é necessário especificar padrões comuns para ambas as implementações (máquina, memória, consultas, etc);
- adicionar a coordenada z (profundidade) aos objetos geográficos;
- avaliar características de cada modelo com relação à incorporação de imagens e diagramas como atributos;
- permitir e avaliar a possibilidade de multiplicidade de representações dos objetos, de acordo com a escala utilizada;
- avaliar algoritmos de percurso máximo/mínimo e
- avaliar os modelos com relação à utilização de raster.

Bibliografia

- [ABD⁺89] M. Atkinson, F. Bancelhon, D. DeWitt, K. Dittrich, D. Maier, and S. Zdonik. The Object-Oriented System Manifesto. *In Proc. DOOD, Kyoto, Japan*, December 1989.
- [Abe89] D. J. Abel. SIRO-DBMS: A database tool-kit for geographical information systems. *International Journal of Geographical Information Systems*, 3(2):103–116, 1989.
- [Arg95] Argus. The evolution of geographic information systems. Technical report, Argus Technologies Corporation, 1995.
- [Aro89] Stanley Aronoff. *Geographic Information Systems: A Management Perspective*. WDL Publications, 1989.
- [BF94] François Bancelhon and Guy Ferran. Odmg-93: The object database standard on data engineering. *IEEE Bulletin of the Technical Committee on Data Engineering*, 17(4):3–14, December 1994.
- [Cam95] Gilberto Camara. *Modelos, Linguagens e Arquiteturas para Bancos de Dados Geográficos*. PhD thesis, INPE, December 1995.
- [Cif95] Ricardo Rodrigues Ciferri. Um benchmark voltado à análise de desempenho de sistemas de informações geográficas. Master's thesis, DCC-IMECC-UNICAMP, jun 1995.
- [Cox91] Frederico Sidney Cox Jr. Análise de métodos de acesso a dados espaciais aplicados a sistemas gerenciadores de banco de dados. Master's thesis, Universidade Estadual de Campinas, Dec 1991.
- [dA95] Cristina Dutra de Aguiar. Integração de sistemas de banco de dados heterogêneos em aplicações de planejamento urbano. Master's thesis, DCC-IMECC-UNICAMP, apr 1995.

- [Dan90] Jack Dangermond. Classification of software components commonly used in gis. In *Introductory Readings in Geographic Information Systems*, pages 30–51. Taylor & Francis, 1990.
- [Dat86a] C. J. Date. *An Introduction to Database Systems*. M.A.: Addison-Wesley, 4th edition, 1986.
- [Dat86b] C. J. Date. *Relational Databases: Selected Writings*. Addison-Wesley, 1986.
- [Deu91] O. Deux and et all. The O₂ system. *Communications of the ACM*, 34(10):34–48, October 1991.
- [dFM93] L. de Floriani and P. Marzano. Spatial queries and data models in advances in spatial databases. In Springer-Verlag, editor, *Lecture Notes in Computer Science*, volume 716, pages 113–138, September 1993.
- [dN95] Adriana Maria Rebouças do Nascimento. Lingeo - uma linguagem de consulta geográfica. Master's thesis, DI-CCEN-UFPE, dec 1995.
- [dO95] Cleomar Márcio Marques de Oliveira. Cartas náuticas eletrônicas: Operações e estruturas de dados. Master's thesis, DCC-IMECC-UNICAMP, nov 1995.
- [Ege94] Max J. Egenhofer. Spatial sql: A query and presentation language. *IEEE Transactions on Knowledge and Data Engineering*, (6):86–95, 1994.
- [Feu93] Martin Feuchtwanger. *Towards a Geographic Semantic Database Model*. PhD thesis, Simon Fraser University, August 1993.
- [Fra90] Andrew U. Frank. Spatial concepts, geometric data models and data structures. Technical Report 90-11, National Center for Geographic Information Analysis / NCGIA, November 1990.
- [Gat91] A. C. Gatrell. Concepts of space and geographical data. In *Volume 1 of [MGR91]*, chapter 9, pages 119–134. Longman Scientific & Technical, 1991.
- [Geo92] GeoVision Systems. *Introduction to VISION**, 1992.
- [Her92] John R. Herring. Tigris: A data model for an object-oriented geographic information system. *Computer & Geosciences*, 18(4):443–452, 1992.
- [JdVB94] Clodoveu Augusto Davis Jr and Karla Albuquerque de Vasconcelos Borges. Gis orientado a objetos na prática. In *GIS Brasil 94 - Congresso e Feira para Usuários de Geoprocessamento*, pages 18–28, 1994.

- [JdVB95] Clodoveu Augusto Davis Jr and Karla Albuquerque de Vasconcelos Borges. Gis orientado a objetos: Teoria e prática. *Fator GIS - A Revista do Geoprocessamento*, pages 31–34, Abril/Maio/Junho 1995.
- [JTTW91] John V. Joseph, Satish M. Thatte, Craig W. Thompson, and David L. Wells. Object-oriented databases: Design and implementations. In *Proceedings of the IEEE*, pages 42–64, January 1991.
- [Kam96] Aqueo Kamada. Transformação de esquema relacional para esquema orientado a objeto em sistemas de banco de dados herogêneos. Master's thesis, DCA-FEE-Unicamp, January 1996.
- [Kim95] Won Kim, editor. *Modern Database Systems: The Object Model, Interoperability, and Beyond*. ACM Press, 1995.
- [KL89] W. Kim and F. H. Lochovsky. *Object-Oriented Concepts, Databases and Applications*. ACM Press, 1989.
- [KS93] Henry F. Korth and Abraham Silberschatz. *Sistemas de Banco de Dados*. Makron Books, 1993.
- [LdA95] Maria Ivete Lovato and Rubens Queiroz de Almeida. Administração de redes tcp/ip, oct 1995.
- [LODL91] Hongjun Lu, Beng Chin Ooi, Ashvin D'Souza, and Che Chin Low. Storage management in geographic information systems. *LNCS 525 Advances In Spatial Databases*, pages 451–470, November 1991.
- [Mel94] Jim Melton. Object technology and sql: Adding objects to a relational language. *IEEE Bulletin of the Technical Committee on Data Engineering*, 17(4):15–26, December 1994.
- [MGR91] D. J. Maguire, M. F. Goodchild, and D. W. Rhind. *Geographic Information Systems, Principles and Applications*. 1991.
- [MO86] F. Manola and J. Orenstein. Toward a general spatial data model for an object-oriented dbms. In *Proceedings of 12th VLDB*, pages 328–335, August 1986.
- [Mor92] Scott Morehouse. The arc/info geographic information system. *Computer & Geosciences*, 18(4):435–442, 1992.
- [MP94] Cláudia Bauzer Medeiros and Fátima Pires. Database for gis. *SIGMOD RECORD*, 23(1):107–115, March 1994.

- [MS93] Jim Melton and Alan R. Simon. *Understanding the new SQL: a complete guide*. Morgan Kaufmann Publishers, Inc., 1993.
- [NW79] George Nagy and Sharad Wagle. Geographic data processing. *Computer Surveys*, 11(2):139-181, 1979.
- [Oli92] Ricardo César Nery Oliveira. A-sql: Uma interface semântica. Master's thesis, Universidade Estadual de Campinas, Agosto 1992.
- [Ooi91] B. C. Ooi. *Efficient Query Processing in Geographic Information Systems*. Springer-Verlag LNCS 477, 1991.
- [Pet87] Robert W. Peterson. Object-oriented data base design. *AI Expert*, pages 26-31, March 1987.
- [RBP+91] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorenzen. *Object-Oriented Modelling and Design*. Prentice-Hall, 1991.
- [RM92] Jonathan F. Raper and David J. Maguire. Design models and functionality in gis. *Computers & Geosciences*, 18(4):387-394, 1992.
- [ROG89] David Rhind, Stan Openshaw, and Nick Green. The analysis of geographical data: Data rich, technology adequate, theory poor. *LNCS 339*, pages 427-454, 1989.
- [Sam90] Hannah Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley Publishing Company, Inc., 1990.
- [San93] Marçal Santos. Unicamp: Uma universidade conectada ao futuro. *DECUS-COPE*, 5(8):11-19, 1993.
- [SE90] J. Star and J. Estr. *Geographic Information Systems. An Introduction*. Prentice Hall, Englewood Cliffs, 1990.
- [Sea95] Doug Seaborn. Database management in gis - past, present, future: An enterprise perspective for executives. *SHL VISION* Solutions*, pages 1-12, 1995.
- [Tec93] O₂ Technology. The O₂ user manual, version 4.3. Technical report, O₂ Technology, 1993.
- [Tom90] Tomlinson. A classification of software components commonly used in geographic information systems. In *Introductory Readings in Geographic Information Systems*, pages 30-51. Taylor & Francis, 1990.
- [Ull82] J. D. Ullman. Principles of database systems. *Pattern Recognition*, 6(1):29-43, 1982.