UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

Unificação Semântica de Esquemas Conceituais de Banco de Dados Geográficos

por

GUILLERMO NUDELMAN HESS

Dissertação submetida à avaliação, como
requisito parcial, para a obtenção do grau
de Mestre em Ciência da Computação.

Prof. Dr. Cirano Iochpe
Orientador

Porto Alegre, 08 de setembro de 2004.

CIP – CATÁLOGAÇÃO DE PUBLICAÇÃO

Hess, Guillermo Nudelman

Unificação semântica de esquemas conceituais de banco de dados geográficos para obtenção se candidatos a padrão de análise.
Por Guillermo Nudelman Hess. - Porto Alegre: PPGC da UFRGS, 2004.

110f.:il

Dissertação (mestrado) - Universidade Federal do Rio Grande do Sul. Programa de Pós Graduação em Computação. Porto Alegre, RS – Brasil, 2004. Orientador: Iochpe, Cirano.

1. Modelagem Conceitual. 2. Sistemas de Informação Geográfica. 3. Ontologias. 4. Integração semântica. I. Iochpe, Cirano. II. Título.

AGRADECIMENTOS

A minha mãe Marion, pelo amor, carinho, dedicação e incentivo que sempre me deu e pelas oportunidades que me propiciou, sem medir esforços.

A minha irmã Carolina, pelo apoio, carinho e torcida que sempre teve por mim, sendo um exemplo de esforço e perseverança, além de uma amiga para todas as horas.

A Renata, pelo amor, apoio, companheirismo e incentivos permanentes, entendendo dias de mau humor, ausências e atrasos em função das atividades para o cumprimento desta etapa.

A minha tia Mônica e avó Margot, pelo carinho, atenção e permanente tietagem.

Ao meu orientador, Prof. Dr. Cirano Iochpe, pelos ensinamentos, discussões, oportunidades e atenção dispensadas.

Aos meus amigos de mais de quinze anos, que souberam entender quando não pude acompanhá-los, mas que principalmente nunca me faltaram quando deles precisei. Em especial ao Pocz, Pri, Artur, Betinho, Tiago, Celinho, Matone e Felipe.

Aos meus amigos que fiz durante a graduação e aqueles que fiz durante o mestrado, que me apoiaram, me incentivaram e tornaram mais fácil a travessia desta etapa. Em especial ao D'Avila, Drebes, Carolina, André, Leonardo, Lucinéia, companheiros do Deslocamento 0 e pessoal do Futinf.

A família da Renata, pelo carinho com que sempre me tratou e apoio que me deu.

Aos professores do PPGC, pela dedicação em fazer deste programa não somente um curso de excelência, mas também um ambiente de trabalho e pesquisa extremamente agradável. Em especial ao professor Palazzo sempre disponível para conselhos e orientações.

Ao Instituto de Informática, pela infra-estrutura disponibilizada e aos funcionários que foram solícitos e ajudaram sempre que possível.

Por fim, a todos que, de alguma forma, colaboraram para o desenvolvimento desta pesquisa.

SUMÁRIO

LISTA DE ABREVIATURAS

BDG	Banco de Dados Geográfico
DCBD	Descoberta de Conhecimento em Banco de Dados
FDE	Formato de Dados de Entrada
GML	*Geography Markup Language*
KDD	*Knowledge Discovery in Databases*
MD	Mineração de Dados
OGC	Open GIS Consortium
RDF	*Resource Description Framework*
SIG	Sistemas de Informação Geográfica
UML	*Unified Modeling Language*
XML	*Extensible Markup Language*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A modelagem conceitual de banco de dados geográficos (BDG) é um aspecto fundamental para o reuso, uma vez que a realidade geográfica é bastante complexa e, mais que isso, parte dela é utilizada recorrentemente na maioria dos projetos de BDG. A modelagem conceitual garante a independência da implementação do banco de dados e melhora a documentação do projeto, evitando que esta seja apenas um conjunto de documentos escritos no jargão da aplicação. Um modelo conceitual bem definido oferece uma representação canônica da realidade geográfica, possibilitando o reuso de sub-esquemas. Para a obtenção dos sub-esquemas a serem reutilizados, o processo de Descoberta de Conhecimento em Bancos de Dados (DCBD – KDD) pode ser aplicado. O resultado final do DCBD produz os chamados padrões de análise. No escopo deste trabalho os padrões de análise constituem os sub-esquemas reutilizáveis da modelagem conceitual de um banco de dados. O processo de DCBD possui várias etapas, desde a seleção e preparação de dados até a mineração e pós-processamento (análise dos resultados). Na preparação dos dados, um dos principais problemas a serem enfrentados é a possível heterogeneidade de dados. Neste trabalho, visto que os dados de entrada são os esquemas conceituais de BDG, e devido à inexistência de um padrão de modelagem de BDG largamente aceito, as heterogeneidades tendem a aumentar. A preparação dos dados deve integrar diferentes esquemas conceituais, baseados em diferentes modelos de dados e projetados por diferentes grupos, trabalhando autonomamente como uma comunidade distribuída. Para solucionar os conflitos entre esquemas conceituais foi desenvolvida uma metodologia, suportada por uma arquitetura de software, a qual divide a fase de pré-processamento em duas etapas, uma sintática e uma semântica. A fase sintática visa converter os esquemas em um formato canônico, a Geographic Markup Language (GML). Um número razoável de modelos de dados deve ser considerado, em conseqüência da inexistência de um modelo de dados largamente aceito como padrão para o projeto de BDG. Para cada um dos diferentes modelos de dados um conjunto de regras foi desenvolvido e um w*rapper* implementado. Para suportar a etapa semântica da integração uma ontologia é utilizada para integrar semanticamente os esquemas conceituais dos diferentes projetos. O algoritmo para consulta e atualização da base de conhecimento consiste em métodos matemáticos de medida de similaridade entre os conceitos. Uma vez os padrões de análise tendo sido identificados eles são armazenados em uma base de conhecimento que deve ser de fácil consulta e atualização. Novamente a ontologia pode ser utilizada como a base de conhecimento, armazenando os padrões de análise e possibilitando que projetistas a consultem durante a modelagem de suas aplicações. Os resultados da consulta ajudam a comparar o esquema conceitual em construção com soluções passadas, aceitas como corretas.

Geographic Database's Conceptual Schema Semantic Integration

ABSTRACT

In Geographic Databases (GDB) conceptual modeling is a key issue for reuse, since the geographic reality is quite complex and, moreover, part of it is recurrently represented in most GDB projects. Conceptual modeling guarantees independence from the database software implementation and improves project documentation preventing it to become only a set of documents written in application-related jargon. A well-defined conceptual model offers a canonical representation of the geographical reality allowing for sub-schemas reuse. To obtain the sub-schemas for reuse, Knowledge Discovery in Databases (KDD) may be applied. The final result of the KDD process produces what is known as analysis patterns. In the scope of this work analysis patterns constitute the reusable sub-schemas of a database design. A KDD process has several phases, from the data selection and preparation to the data mining and post-processing (analysis of the results). In the data preparation, or pre-processing phase, one of the main problems to be handled is the possible data heterogeneities. In our work, since the input data are GDB conceptual schemas, and since there are no widely accepted standards for GDB modeling, the heterogeneities tend to increase. Data preparation must integrate different conceptual schemas, based on different data models and designed by different groups working autonomously as a distributed community. To solve conflicts between conceptual schemas we developed a methodology supported by a software architecture, which divides the pre-processing phase in two steps, a syntactic one and a semantic one. The syntactic step intends to convert the schemas into a canonical format, the Geographic Markup Language (GML). A number of different data models must be taken into consideration as a consequence of the absence of a widely accepted standard data model for GDB conceptual design. For each different data model a set of rules was developed and a wrapper has been implemented. To support the semantic step we use an ontology to semantically integrate conceptual schemas of different projects. The algorithm to search and update the knowledge base relies on mathematical methods of similarity measurement between concepts. Once the analysis patterns are identified they are stored in a knowledge base that can be easily searched and updated. Again the use of ontology supports the knowledge base, storing the analysis patterns and allowing designers to query it during the modeling of new applications. Query results help to compare the conceptual design under construction with past solutions accepted as correct. Furthermore by processing the ontology one can suggest design alternatives when the application domain is not well known by the designer.

1 Introdução

1.1 Motivação

Um sistema de informação geográfica (SIG) é um conjunto de ferramentas computacionais composto de equipamentos e programas que, por meio de técnicas, integra dados, pessoas e instituições, de forma a tornar possível a coleta, o armazenamento, o processamento, a análise e a disponibilização, a partir de dados georreferenciados, de informação produzida por meio das aplicações disponíveis, visando maior facilidade, segurança e agilidade nas atividades humanas referentes ao monitoramento, planejamento e tomada de decisão relativas ao espaço geográfico (TEIXEIRA; CHRISTOFOLETTI, 1997). Ferramentas de SIG têm ganhado cada vez mais espaço no mercado, no sentido de auxiliar no processo de tomada de decisão em áreas como, por exemplo, saúde, segurança, planejamento urbano e controle ambiental (BURROUGH; MCDONNEL, 1997; LISBOA, 2002).

Dados georreferenciados, ou geográficos, se diferem dos dados ditos convencionais por apresentarem componentes espaciais e temporais, além das não-espaciais (ditas descritivas) (ARONOFF, 1989). O objetivo dos dados georreferenciados é o de responder as seguintes questões: o quê?, onde? e quando?. Visando descrever os fenômenos do mundo real, a componente não-espacial apresenta suas características descritivas (tais como nome, extensão e vazão de um rio). A componente espacial descreve a geometria do dado, bem como sua topologia, tendo sempre como referência de sua localização algum ponto sobre a superfície da Terra, expressa por meio de coordenas geográficas. Informações tais como a data de coleta do dado e seu período de validade são informados pela componente temporal.

Os dados geográficos costumam ser armazenados em um banco de dados geográficos (BDG), sendo este um dos principais componentes do SIG. Devido à complexidade dos dados geográficos, o projeto de um BDG torna-se bastante complexo e ao mesmo tempo fundamental para o bom funcionamento do sistema. Assim como um banco de dados convencional, o projeto de um BDG deve passar pelas etapas conceitual, lógica e física (ELMASRI; NAVATHE, 2000).

A modelagem conceitual objetiva descrever, em um alto nível de abstração, independente de plataforma de software e hardware a ser utilizada, os elementos do mundo real que estarão presentes no banco de dados, bem como os relacionamentos entre eles. Esta etapa demanda muito tempo, pois requer que o projeto de um novo sistema seja iniciado do conhecimento do domínio e do levantamento dos requisitos da aplicação.

Com o objetivo de acelerar o processo de modelagem conceitual de banco de dados, a utilização de padrões de análise tem sido amplamente explorada, uma vez que apresentam a essência da modelagem conceitual de uma solução para um problema recorrente, em um contexto específico (FOWLER, 1997; LISBOA, 2002). Um padrão de análise consiste em um sub-esquema conceitual de bancos de dados freqüentemente utilizado na modelagem conceitual de aplicações distintas (SILVA, 2003). Esta reutilização é especialmente interessante no que diz respeito aos BDGs, uma vez que sua modelagem é bastante complexa, e parte da realidade geográfica modelada se repete para diferentes aplicações.

O processo clássico de identificação de padrões é feito de forma totalmente manual e dependente de um especialista do domínio da aplicação. Desta forma, além da pouca disponibilidade de padrões, existe também o problema de sua aceitação, visto que sua aquisição foi baseada em projetos específicos.

A dificuldade em se provar que um determinado (sub-)esquema é realmente um padrão não está somente no fato dos especialistas poderem discordar quanto ao (sub-)esquema em sí, mas também no seu entendimento dos construtores presentes nele. Pode haver um conflito de opiniões entre os especialistas devido a experiências diferentes em modelagem da realidade. Os conflitos podem ser ocorrer no nível de nomenclatura, composição (características) e relacionamentos dos fenômenos modelados, mas também no nível de entendimento do conceito representado pelo fenômeno. Os dois níveis constituem os conflitos semânticos. Não obstante, a sintaxe para a modelagem também pode variar, por exemplo, em termos de idioma e modelo utilizado (STOIMENOV; DJORDJEVIC-KAJAN, 2003; VERSCHELDE et al., 2004)..

Uma forma de tornar a identificação de padrões mais rápida e mais automática é o processo de Descoberta de Conhecimento em Banco de Dados (DCBD). A DCBD ou *Knowledge Discovery in Databases (KDD)* é a área de estudo relacionada ao desenvolvimento de métodos e técnicas para descobrir algum conhecimento ou significado implícito em bases de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Através da (semi-) automatização do processo de identificação de padrões de análise, um número maior de esquemas conceituais de BDG pode ser analisado, resultando em estatísticas mais confiáveis que possam validar o conhecimento extraído (ELDER; KRIEGER; XU, 1996).

DCBD (ou KDD) é o processo não-trivial de extração de conhecimento a partir de um grande volume de dados. O processo é considerado não-trivial, pois o conhecimento que se busca está representado de forma implícita, não sendo possível obtê-lo diretamente a partir de consultas à base de dados. Assim, a pesquisa no contexto de DCBD busca desenvolver métodos e técnicas capazes de obter conhecimento a partir de dados brutos, automatizando, ao menos parcialmente, a tarefa de análise dos dados (SILVA, 2003).

Um processo de DCBD consiste em uma seqüência de etapas, ilustradas na Figura 1.1, as quais são a seleção, o pré-processamento, a transformação (preparação dos dados), a mineração de dados (processamento) e a interpretação/avaliação (pós-processamento).

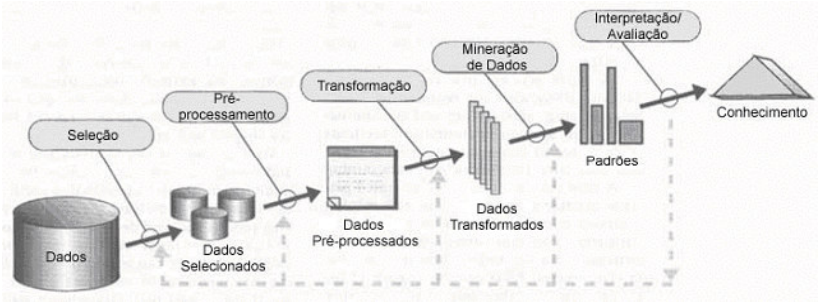


Figura 1.1 - O processo de KDD (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996)

A etapa de mineração de dados (MD) é o centro do processo. Inclusive, comumente, a MD é confundida com todo o processo de DCDB. Em realidade, KDD refere-se ao processo de descoberta de conhecimento útil a partir de bases de dados e MD é um passo do todo, no qual ocorre a aplicação de um algoritmo específico para extração de padrões a partir dos dados existentes (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Contudo, para se chegar a ela, é necessário o pré-processamento e a preparação dos dados de entrada.

Em (SILVA; IOCHPE; ENGEL, 2003) foram investigadas e desenvolvidas técnicas de Mineração de Dados (MD) e pós-processamento para a inferência de candidatos a padrão de análise, partindo-se de esquemas conceituais de BDG. Dentro do processo de DCBD estas são duas das etapas envolvidas. Contudo, como aquele trabalho não tratou da preparação de esquemas reais de BDG para mineração, utilizou-se, como entrada da MD, esquemas criados artificialmente. Não foi, portanto, possível (e nem era este o objetivo do trabalho) encontrar candidatos a padrão de análise, baseados em esquemas reais de BDG.

A preparação dos dados, que neste caso são esquemas conceituais de BDG, engloba as fases de seleção, pré-processamento e transformação dos dados a serem utilizados no processo de DCBD. A preparação deve garantir a correção dos dados, para evitar erros durante a mineração ou durante o processo de avaliação dos resultados obtidos com a MD. Por correção entende-se tanto a utilização dos dados adequados quanto o tratamento de diferenças entre os mesmos.

A etapa de seleção consiste na escolha dos dados a serem utilizados no processo de DBCD. Dependendo da fonte de onde os dados são buscados, diferentes técnicas podem ser utilizadas. Se a origem for um banco de dados, podem-se utilizar consultas em linguagem SQL para obter os dados (BIGOLIN, 1999).

Uma vez tendo os dados de entrada reunidos, é necessário que eles sejam pré-processados. O pré-processamento tem por objetivo a identificação de erros e inconsistências nos dados selecionados. É nesta fase que as heterogeneidades são tratadas, sejam elas sintáticas ou semânticas. Nesta fase é necessário o apoio de um especialista do domínio da aplicação ou de um sistema de apoio para que possam ser corrigidas as incorreções encontradas.

A última das etapas da preparação dos dados é a transformação, que consiste em modificar os dados pré-processados de modo que estes fiquem em um formato interpretável pelas ferramentas de MD. Como produto final da etapa de preparação obtém-se um conjunto de dados armazenados em um único repositório, livre de erros e inconsistências, e em um formato adequado para serem minerados.

Para que seja possível minerar esquemas conceituais de aplicações reais, estes devem estar todos no mesmo formato, o qual deve interpretável pelas ferramentas de MD.

Contudo, devido ao fato de não existir para BDG um modelo de dados padrão, tal como ocorre com a orientação a objeto e o modelo E-R na modelagem conceitual de banco de dados convencionais, a etapa de preparação (pré-processamento) torna-se mais importante e mais complexa quando se trata de BDG.

Diversos modelos conceituais para projeto de BDG têm sido propostos, com o objetivo de tornar a modelagem independente de plataforma de implementação. Dentre eles, pode-se citar o UML-GeoFrame LISBOA; IOCHPE, 1999; ROCHA; EDELWEISS; IOCHPE, 2001), o MADS (PARENT et al., 1999), o OMT-G (BORGES, 1997), o padrão canadense SAIF (SAIF, 2001) e o GeoOOA (KÖESTERS, 1997). Entretanto, nenhum destes formatos é compatível com as ferramentas de mineração de dados. Mais que isso, os formatos são semelhantes, mas não iguais. Cada um representa os conceitos de uma forma distinta e possui uma abrangência (poder de expressão) diferente. Em outras palavras, existem conflitos sintáticos entre os esquemas conceituais baseados nos diferentes modelos de dados. Um estudo comparativo entre eles é apresentado em (BASSALO; IOCHPE; BIGOLIN, 2002), bem como é apresentado o conjunto *união* de construtores. Este conjunto *união* busca identificar as equivalências em termos de construtores entre os modelos, de forma a tratar a integração sintática de esquemas.

1.2 Objetivo do trabalho

O conjunto *união* (BASSALO; IOCHPE; BIGOLIN, 2002) tratou da unificação de construtores dos modelos de dados para BDG em nível sintático, ou seja, apenas identificando equivalências entre os componentes de cada modelo, e não entre os conceitos modelados em cada esquema. Desta forma, a unificação dos esquemas conceituais ficou incompleta, uma vez que nenhuma semântica foi considerada. Conceitos com nomes diferentes e significados iguais (sinônimos) não eram identificados como representando o mesmo fenômeno do mundo real. Por outro lado, para conceitos de nomes iguais, mas com significados distintos (homônimos) não era possível identificar a qual fenômeno do mundo real eles se referiam.

Para possibilitar a mineração de esquemas de aplicações reais de BDG, baseados em diferentes modelos de dados, o principal objetivo deste trabalho é propor uma técnica de integração semântica de esquemas conceituais de BDG. Partindo do conjunto *união* de construtores identificado no trabalho de Bassalo, Iochpe e Bigolin (2002), esquemas conceituais de BDG serão preparados para serem minerados pelas ferramentas que produzem as regras associativas utilizadas em Silva, Iochpe e Engel (2003), conforme apresentado na arquitetura genérica (Figura 1.2).

Segundo Batini, Lenserini e Navathe (1986), é bastante conveniente que os esquemas conceituais a serem comparados sejam descritos em um mesmo modelo de dados, eliminando assim as heterogeneidades entre modelos. É importante também, que o modelo de dados canônico (comum) adotado tenha uma expressividade similar ou superior à dos demais modelos de dados utilizados. Um formato canônico consiste no esquema original do BDG expresso em uma representação sintática padrão (comum). Se este modelo já é aceito como um padrão de intercâmbio e armazenamento, mais indicada é sua utilização.

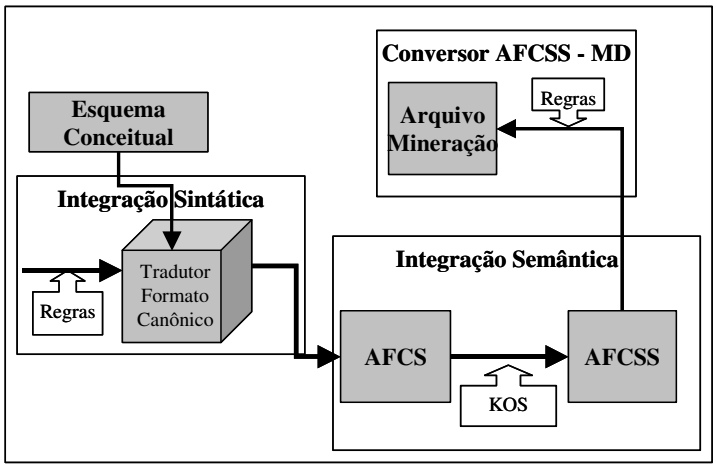


Figura 1.2 - Arquitetura genérica do pré-processamento

Conforme ilustrado, um esquema conceitual é convertido, primeiramente, para um arquivo em formato canônico sintático (AFCS), ou seja, somente no nível sintático. Isto quer dizer que, independente do modelo de dados nos quais os esquemas estão baseados e em que formato são descritos, eles são convertidos para o formato comum, de modo a facilitar o processamento nas etapas subseqüentes. Há apenas uma conversão dos construtores utilizados no modelo de dados original para o formato canônico. De acordo com o modelo de dados no qual o esquema conceitual está baseado, um determinado conjunto de regras de tradução é aplicado.

A segunda etapa do processo consiste em passar o AFCS por uma sistema de organização do conhecimentos (*Knowledge Organization System* – KOS) (HODGE, 2000), tal como vocabulário controlado, taxonomia, thesaurus ou ontologia (QIN; PALING, 2001), de modo a garantir o nível semântico da preparação de dados. Nesta etapa, que constitui a principal contribuição deste trabalho, as heterogeneidades de nomenclatura, estrutura, relacionamentos e conceitualização são tratadas. O resultado é um arquivo no formato canônico sintático e semântico (AFCSS). A AFCSS consiste no esquema original do BDG expresso em uma representação sintática padrão, semanticamente unificada, com outros esquemas conceituais de BDG. A resolução de conflitos semânticos é fundamental na preparação de dados e para todo o processo de DCBD, uma vez que as ferramentas de MD não são capazes de tratá-los, como por exemplo distinguir dois conceitos representados com o mesmo nome (homônimos), nem de reconhecer quando duas representações diferentes se referem ao mesmo conceito (sinônimos). Para o sucesso da integração semântica, faz-se necessária a utilização de algoritmos de casamento (*matching*) para a correta interpretação de variações dos termos já conhecidos, e para a classificação de novos.

A última etapa da preparação dos esquemas conceituais para mineração consiste em transformar o arquivo AFCSS em um formato de dados suportado por ferramentas de MD.

Para alcançar o objetivo desta pesquisa, que é o desenvolvimento do módulo de resolução de conflitos semânticos, alguns sub-produtos devem ser gerados. O primeiro é a proposta de uma ontologia geográfica, como forma de armazenar os conceitos do domínio da aplicação. Esta ontologia não tem a pretensão de ser uma base de conhecimento completa para representar os fenômenos geográficos do mundo real. Baseado na ontologia, um algoritmo é proposto para confrontar os elementos modelados nos esquemas conceituais

de entrada com os conceitos presentes na ontologia. Este algoritmo consiste na aplicação de um conjunto de regras de casamento por similaridade (*similarity matching*), as quais devem ser ponderadas, uma vez que os esquemas conceituais de entrada podem possuir características bastante diferentes.

1.3 Trabalhos relacionados

Philips e Buchanan (2001) utilizam ontologias no na fase de pré-processamento do processo de DCBD. A ontologia é aplicada na preparação dos dados para mineração. O trabalho foi desenvolvido para ser aplicado no campo médico, especificamente para guardar os dados de pacientes em tratamento ortopédico. O trabalho apresenta algumas limitações, tais como aceitar somente uma linguagem natural (inglês), assim como resolve somente os conflitos do tipo sinônimos.

Smith e Mark (1998) apresentam um estudo diferenciando uma ontologia para dados convencionais de uma ontologia para dados geográficos. Eles apontam como questão fundamental para o desenvolvimento de uma ontologia geográfica a consideração dos aspectos de topologia, geometria e de mereologia dos dados. Topologia diz respeito aos limites, geometria à forma e mereologia às relações de todo-parte.

Fileto (2003), em sua tese de doutorado, criou uma ontologia geográfica contendo conceitos de divisão territorial e de produtos agrícolas. Seu objetivo era a integração de dados geográficos provenientes de diferentes fontes.

A criação de um modelo conceitual global geográfico foi proposta por Estrada et. al (2001). O objetivo é a integração de esquemas conceituais de BDG para intercâmbio de dados. Para tanto, o sistema apresentado vale-se de uma ontologia geográfica para resolver conflitos semânticos. Esta ontologia armazena os conceitos com seus atributos e relacionamentos hierárquicos e de agregação. A metodologia de utilização da ontologia não está especificada, e o sistema está restrito aos SIGs MGE da empresa Intergraph e ao SPRING, do INPE.

Também do INPE, o trabalho de Lima, Câmara e Queiroz (2002) propõe o GeoBR, um formato para intercâmbio sintático e semântico de dados geográficos. Este formato é baseado em XML e define um esquema próprio para codificação, diferente do esquema GML. O formato GeoBR se faz uso da linguagem RDF para descrever metadados e da linguagem DAML para descrever os dados da ontologia. Um protótipo de uma ferramenta que use o GeoBR começou a ser desenvolvido e foi abandonado. Até o momento, nenhuma ontologia havia sido escrita em GeoBR.

Hakimpour and Timpf (2001) propõem o uso de ontologias na resolução de heterogeneidades semânticas em SIG. Eles especificam as funções da ontologia, apresentam a definição de um pequeno conjunto de conceitos em lógica de descrição (*description logics* – DL) e as propriedades dos conceitos que devem ser consideradas para o tratamento de heterogeneidades: nomes, atributos e relacionamentos taxonômicos.

Uitermark, van Oosterom, Mars e Molenaar (1999) apresentam um *framework* para auxiliar a integração de dados geográficos. O trabalho utiliza-se de uma ontologia de domínio específica para dados topográficos. O objetivo da pesquisa era possibilitar consultas em um ambiente distribuído de conjuntos de dados heterogêneos.

Stoimenov e Djordjevic-Kajan (2003) propuseram o *framework* GeoNis para alcançar a interoperabilidade semântica entre dados de SIG. A utilização de ontologias foi proposta no contexto de servir como base de conhecimento para a integração e para a solução de

heterogeneidades tais como homônimos e sinônimos, assim como heterogeneidades taxonômicas. A necessidade de algum tipo de integração sintática também é levantada, com o objetivo de unificar os modelos de dados, possivelmente diferentes.

Doan et. al (2002) apresenta o GLUE, uma ferramenta para mapeamento entre ontologias. Este trabalho está focado na identificação de conceitos similares através da comparação das taxonomias das ontologias comparadas. O trabalho não tem como objetivo integrar ontologias, mas encontrar similaridades.

Especificamente no que diz respeito à integração de esquemas conceituais de bancos de dados assistida por uma ontologia, há o trabalho de Verschelde et. al. (2004). O foco do trabalho foi a integração de diferentes fontes de dados biológicas. Entretanto, os tipos de heterogeneidades apontadas são independentes do domínio da aplicação: homônimos, sinônimos e associações.

Ainda no ramo de integração de esquemas conceituais de bancos de dados, há trabalhos que propõem a utilização de uma arquitetura do tipo banco de dados federados (RIBEIRO, 1995; PARK, 2001). Ribeiro (1995) propôs um formato de dados específico para integração e armazenamento de informações semânticas. Park (2001) sugere um dicionário de metadados para armazenamento de atributos padrão. Não há, nestas propostas, um esquema conceitual único.

Outros trabalhos a respeito de integração semântica de esquemas conceituais encontrados sugerem a adoção de ontologias como um esquema conceitual global (MELLO; HEUSER, 2001; DORNELES, 2002), descrito em XML. Os esquemas conceituais locais também devem ser descritos em uma DTD (BRADLEY, 2002) XML e algoritmos de integração semântica são aplicados para unificar os esquemas.

Fonseca et al. (2002) propuseram uma arquitetura de SIG para integração de informações geográficas dirigida por uma ontologia. Nesse trabalho, a ontologia atua como um integrador de sistemas independente do modelo, utilizando os atributos e papéis dos conceitos como campos a serem considerados na integração.

Botêlho, Strauch e De Souza (2003) propõem um sistema para integração e publicação de dados convencionais e georreferenciados na *Web*. O sistema chamado WISE (*Web data Integration SystEm*) utiliza um modelo de dados canônico para representar dados convencionais e georreferenciados, chamado XML Schema Semântico (SXMLS), o qual utiliza as tecnologias XML e GML, bem como um conjunto de elementos semânticos, para representar os esquemas de fontes de dados locais em um modelo comum que armazene informações semânticas. Os SXMLSs são integrados pelo WISE utilizando estas informações semânticas com o auxílio de ontologias da aplicação. (BOTÊLHO, 2004). Nesse trabalho os conflitos semânticos a serem tratados são levantados, mas apenas as heterogeneidades em termos de sistema de projeções e coordenadas, unidades de medidas e nomes foram tratados. Os dois primeiros são exclusivos para dados, enquanto conflitos de nome podem (e devem) ser tratados em nível de esquema conceitual.

Apesar dos trabalhos supra citados levantarem os aspectos a serem tratados na resolução de conflitos semânticos e até sugerirem abordagens a serem utilizadas, nenhum deles apresenta uma metodologia de resolução de conflitos, ou seja, um algoritmo que demonstre como as heterogeneidades podem ser resolvidas.

Em termos de sistemas que apresentam soluções para encontrar similaridades entre conceitos, existe uma gama de trabalhos. Richardson, Smeaton e Murphy (1994) utilizam a base de conhecimento do WorNet para calcular a similaridade entre conceitos somente em função da taxonomia. Dados dois conceitos, sua similaridade é calculada encontrando-se o

primeiro conceito que generaliza os dois comparados. De forma semelhante, a ferramenta TransScm (MILO; ZOHAR, 1998). Os conceitos são identificados como similares através de sua hierarquia, mas além disso mecanismos de tradução de um esquema para outro são fornecidos.

O SKAT (MITRA; WIEDERHOLD; JANNIK, 1999) (Semantic Knowledge Articulation Tool), o DIKE (PALOPOLI; SACCA; URSINO, 1998) e o CUPID (MADHAVAN; BERNSTEIN; RAHN, 2001) são *matchers* híbridos, que fazem integração por similaridade de nomes e similaridade estrutural (relacionamentos taxonômicos).

O sistema ARTEMIS (CASTANO; De ANTONELLIS; De CAPITANI di VEMERCATI, 2001), é um *matcher* híbrido para integração de esquemas. Sua implementação considera nomes, cardinalidades e tipos de atributos de um elemento. Cada um dos aspectos utilizados na integração é ponderado. Este sistema utiliza um *thesaurus* como base de conhecimento auxiliar a ser consultada na integração.

Os sistemas citados acima não têm como propósito integrar esquemas de bancos de dados, mas são aproximações de interesse de como comparar conceitos isolados. Nenhum deles, contudo, possui métodos de identificação de similaridades para todas as heterogeneidades possíveis. Cada um aborda alguma(s) delas.

1.4 Estrutura do trabalho

O texto da dissertação está estruturado em sete capítulos, e dois anexos, além das referências bibliográficas.

O Capítulo 2 aborda as questões relevantes à integração semântica de esquemas conceituais de BDG. Primeiramente é apresentado o levantamento bibliográfico a respeito dos tipos de heterogeneidades semânticas. Também é apresentado um estudo comparativo entre os sistemas de organização de conhecimento que se mostraram alternativas para servirem de base de conhecimento. Um conjunto de critérios foi levantado para justificar a escolha por um dos sistemas de KOS.

O Capítulo 3 apresenta o detalhamento da funcionalidade de um módulo de integração semântica de esquemas conceituais de BDG com base no tipo de KOS escolhido no capítulo 2. É proposto um algoritmo para processamento dos esquemas conceituais a serem integrados, associado a fórmulas matemáticas para cálculo de similaridade.

O Capítulo 4 demonstra a funcionalidade do esquema de integração semântica proposto através da implementação de um protótipo. Um estudo de caso é apresentado para ilustrar o funcionamento da proposta.

Finalmente, o Capítulo 5 apresenta as conclusões e possíveis trabalhos futuros.

O anexo 1 apresenta os modelos de dados utilizados na integração sintática dos esquemas conceituais. O anexo 2 apresenta as ferramentas construídas para a integração sintática entre os modelos, assim como a ferramenta para conversão do modelo canônico para o formato de mineração de dados.

2 A Integração Semântica de Esquemas Conceituais de BDG

O nível semântico de heterogeneidade diz respeito aos aspectos relacionados à compreensão e utilização de dados relacionados a diferentes aplicações e usuários, envolvendo diferentes modelos de dados e diferentes interpretações obtidas a partir destes distintos modelos de dados. A explicação para tal fato é bastante simples. Uma mesma entidade do mundo real, modelada por pessoas de diferentes áreas de atuação, áreas de formação ou culturas, pode ser representada de formas diversas (por exemplo, como atributo ou classe) e/ou identificada em diferentes idiomas (por exemplo, rio e river) ou ainda com termos diferentes (por exemplo, rio e curso d'água), embora representando um mesmo fenômeno do domínio da aplicação. Nestes casos, ocorre o que se chama de conflito. Um conflito, nada mais é que uma diferença na representação de um mesmo conceito da realidade.

2.1 Classificação das heterogeneidades

Partdridge (2002) classifica a heterogeneidade semântica em desacordo entre comunidades e desacordo em forma.

Desacordo entre comunidades ocorre quando duas ou mais comunidades não concordam com o significado dos dados, ou parte deles, em um banco de dado. Estas comunidades acabam por empregar palavras diferentes para representar o mesmo conceito do mundo real, ou seja, com o mesmo significado.

Desacordo em forma ocorre quando o mesmo conjunto de dados em diferentes bancos de dados tem diferenças semânticas, ou seja, possuem diferentes significados, diferentes definições. Isto gera o que é conhecido por homônimos.

Bergamaschi et al. (1998) vai além desta definição, classificando a heterogeneidade em termos de nomenclatura e de estrutura. O primeiro caso engloba os dois aspectos apresentados por Partridge (2002), enquanto heterogeneidade de estrutura diz respeito às diferenças existentes no modelo conceitual utilizado, em termos de relacionamentos e atributos dos conceitos modelados.

Para Park (2001), a heterogeneidade semântica pode ser categorizada, genericamente, em dois níveis diferentes: nível de esquema e nível de dados. No caso de nível de esquema, a heterogeneidade é resultante das diferenças nas estruturas lógicas e/ou inconsistências nos metadados de um mesmo domínio usado em diferentes bancos de dados. Isto é devido ao uso de diferentes estruturas (por exemplo, tabelas e atributos) para uma mesma informação e pelo uso de diferentes especificações (por exemplo, nomes, tipos de dados, restrições)

para uma mesma estrutura. A heterogeneidade em nível de esquema pode, ainda, ser classificada em seis diferentes tipos de conflitos. Conflitos de nomenclatura (homônimos e sinônimos), de identificação de entidades, de isomorfismo de esquemas, de generalização, de agregação e de discrepâncias esquemáticas.

As heterogeneidades de dados resultam nas diferenças nos domínios dos dados, causadas pelas múltiplas representações e interpretações semânticas de um dado. Esta heterogeneidade também pode ser dividida em seis categorias: conflitos de valores, de representação, de unidade, de precisão (incluindo granularidade e resolução espacial), de confiança nos valores de dados conhecidos e de domínios espaciais.

Visser et al. (1997) enfocam o problema da heterogeneidade dividindo-a em quatro categorias distintas. A heterogeneidade de paradigma ocorre quando dois sistemas expressam seu conhecimento usando diferentes paradigmas de modelagens (por exemplo, um orientado a objetos e outro baseado em entidade-relacionamento). A heterogeneidade de linguagem existe se dois sistemas expressam seu conhecimento em diferentes linguagens. A heterogeneidade ontológica ocorre quando dois sistemas divergem quanto ao significado e estrutura dos elementos existentes no seu domínio de aplicação. Por fim, a heterogeneidade de conteúdo existe se dois sistemas representam conteúdos diferentes. As duas últimas categorias formam, juntas, a chamada heterogeneidade semântica.

Há dois tipos básicos de heterogeneidade ontológica, de conceitualização e de explicação (VISSER et al., 1997). Desencontro em termos de conceitualização ocorre entre duas ou mais conceitualizações de um domínio. Elas diferem em termos de conceitos (ou classes) englobados ou no modo como estes conceitos estão relacionados. Descasamento de explicação se refere ao modo como a conceitualização está especificada, ou seja, quando dois esquemas possuem diferentes definições, mas seus termos, significados ou descrições são os mesmos.

Conforme citado acima, a heterogeneidade de conceitualização pode ser em termos de classes ou de relações. No caso de classes, o que ocorre é um conflito relacionado às classes diferenciadas na conceitualização. Um primeiro caso ocorre em termos de categorização, que existe quando a hierarquia de uma mesma classe difere em dois esquemas, pois suas subclasses são diferentes. Também no nível de classes, podem ocorrer desacordos em nível de agregação, ou seja, um mesmo conceito pode ser modelado em diferentes níveis de abstração.

Um conflito de relação existe em relação aos relacionamentos entre os conceitos de diferentes esquemas. Eles podem ser estruturais, de nomes de atributos ou de tipos de atributos. O primeiro caso diz respeito às associações entre dois ou mais conceitos. Os conflitos em termos de atributo dizem respeito tanto aos atributos utilizados para caracterizar um conceito (por exemplo, um esquema pode ter os atributos nome e profissão para o conceito pessoa, enquanto outro possui os atributos nome, idade e sexo para o mesmo conceito pessoa), quanto ao seu domínio.

A Tabela 2.1 apresenta uma comparação entre as diferentes classificações encontradas.

Tabela 2.1 - Comparação das classificações de heterogeneidades semânticas

Autor / Tipo	**Partridge**	**Bergamaschi**	**Park**	**Visser**
Sinônimos	Comunidade	Nomenclatura	Nomenclatura	Explicação
Homônimos	Forma	Nomenclatura	Nomenclatura	Explicação
Atributos	X	Forma	Esquema	Relação

Taxonomia	X	X	Generalização	Categorização
Associações	X	Forma	X	Relação
Construtores	X	X	Esquema	Paradigma
Dados	X	X	Dados	X

Os conflitos abordados neste trabalho são aqueles que envolvem esquemas conceituais de BDG. Desta forma, a heterogeneidades de dados não precisa ser tratada. Em função da complexidade em tratar conflitos entre construtores, ou seja, conseguir detectar quando, por exemplo, um determinado conceito representado como uma classe em um esquema equivale a um atributo em outro esquema, este tipo de heterogeneidade também não foi considerado. Desta forma, a classificação das heterogeneidades adotadas neste trabalho é aquela definida por Visser et al. (1997), que englobam aquelas descritas por Bergamaschi et al. (1998) e Park (2001), sem a parte que envolve os dados. Ou seja, os conflitos a serem tratados são os de denominação (explicação) dos conceitos (homônimos e sinônimos) e os de estrutura (atributos, taxonomias e associações). Também em função da quantidade de associações poder ser bastante grande, somente as de agregação e composição são consideradas.

Para entender os diferentes tipos de heterogeneidade em termos de explicação (definição) de um conceito, consideramos uma definição de um conceito como sendo uma tripla $Def = <T,D,C>$, onde T é um termo, D é a explicação do termo e C é o conceito ontológico que está sendo conceitualizado (VISSER et al., 1997). A heterogeneidade em termos de explicação ocorre quando pelo menos um dos três componentes da tripla conflita em duas explicações.

Um conflito CT ocorre quando dois esquemas utilizam a mesma explicação (descrição) para conceitos diferentes, com nomes diferentes. Por exemplo, um sapo = ser vivo + verde. Por outro lado, grama = ser vivo + verde. Nestes casos, mesmo a descrição sendo a mesma, nem o conceito que está sendo modelado, nem o termo utilizado para nomear o conceito são os mesmos.

Um conflito CD ocorre quando dois esquemas utilizam um mesmo termo para dois conceitos diferentes, com descrições diferentes. Este é o caso de palavras homônimas, como, por exemplo, Laranja = fruta, mas também Laranja = cor.

Um conflito C é um caso particular do conflito CD, no qual não apenas o termo utilizado é o mesmo, mas a descrição também é. Apenas o conceito, ou seja, o significado é diferente.

Um conflito TD ocorre quando dois esquemas utilizam termos e descrições diferentes para um mesmo conceito. Neste caso ocorre o que se conhece por sinônimo. Um caso particular deste tipo de heterogeneidade é o conflito T, no qual apenas o termo utilizado para dar nome ao conceito difere. Um exemplo comum ocorre para termos regionais, como é o caso de sinaleira = sinal de trânsito e semáforo = sinal de trânsito.

Por último, um conflito D ocorre quando dois esquemas definem o mesmo conceito e com o mesmo termo, mas com diferentes explicações.

No caso de esquemas conceituais de BDG, os conflitos TD e T podem ser unificados, bem como os conflitos C e CD. Além disso, o conflito D pode ser ignorado, uma vez que a descrição das classes não é um ponto relevante na unificação, ao menos em um primeiro momento.

2.1.1 A formalização dos conflitos

Para formalizar os diferentes conflitos semânticos, é possível dividi-los em quarto tipos: igualdade, dissimilaridade, interseção e continência (STOIMENOV; DJORDJEVIC-KAJAN, 2003). As fórmulas a seguir são adaptações das originais, descritas em (STOIMENOV; DJORDJEVIC-KAJAN, 2003).

A Igualdade semântica (equivalência) Sequ(c1,c2) ocorre quando há um mapeamento 1:1 exato entre um conceito c1 de um esquema conceitual (S) e um conceito c2 de uma ontologia (O) em termos de significado e estrutura (atributos e relacionamentos). Os conceitos que se enquadram neste tipo de heterogeneidade semântica são chamados sinônimos, e podem ser definidos por SEqu(c1,c2)={(c1,c2)|c1∈S ∧ c2 ∈ O ∧ E(c1) = E (c2) ∧ S(c1) = S(c2)}. E(ci) é o significado do conceito e S(ci) é a estrutura do conceito.

A dissimilaridade semântica SNEqu(c1,c2) ocorre quando não há um mapeamento entre a descrição de um conceito c1 de um esquema conceitual S e um conceito c2 de uma ontologia O. Adicionalmente, se o nome Name(c1) for igual ao nome Name(c2), a heterogeneidade semântica recebe o nome de homonímia. A dissimilaridade semântica pode ser definida por SNEqu(c1,c2) = {(c1,c2)|c1∈S ∧ c2∈O ∧ E(s1)≠E(s2) ∧ Name(c1)=Name(c2)}.

A interseção semântica SIntersec(c1,c2) ocorre quando há um mapeamento 1:1 parcial entre um conceito c1 de um esquema conceitual S e um conceito c2 de uma ontologia O, tanto em termos de estrutura quanto em termos de significado. O último caso pode ocorrer quando um mesmo conceito está sendo modelado para diferentes aplicações, e desta forma alguns dos atributos e relacionamentos são os mesmos, mas outros não são. Esta heterogeneidade semântica pode ser definida por SIntersec(c1,c2) = {(c1,c2)| |c1∈S ∧ c2∈O ∧ S(c1) ∩S(c2) ∧ S(c1) ⊄S(c2) ∧ S(c2) ⊄S(c1)}.

Por ultimo, o quarto tipo de conflito semântico é de continência SContain(c1,c2), o qual ocorre quando a estrutura de um conceito c1 de um esquema conceitual S está contida na estrutura de um conceito c2 de uma ontologia O, e vice-versa. Este conflito ocorre quando um conceito é uma especialização ou generalização de outro, sendo definido por SContain(c1,c2) = {(c1,c2)| |c1∈S ∧ c2∈O ∧ S(c1) ⊂ S(c2) ∧ S(c2) ⊄ S(c1)}.

2.2 Técnicas existentes para integração semântica

Para que a integração semântica possa ser realizada de forma (semi-) automatizada e que permita o reuso dos elementos modelados, uma das soluções possíveis é a aplicação de um KOS (*Knowledge Organization System*). Qualquer KOS é uma tentativa de aproximação para auxiliar a estruturar, classificar, modelar e representar os conceitos e relacionamentos pertencentes a algum assunto de interesse de uma comunidade. Diversos são os tipos de KOS existentes, e, desta forma, cada um representa uma técnica possível de ser utilizada para a unificação dos conceitos dos diferentes esquemas conceituais de BDG.

Algumas características básicas todos os KOS apresentam, independente de sua abrangência, finalidade ou classe (HODGE, 2000):

- Um KOS impõe uma visão particular do mundo, sobre uma coleção e sobre os itens nela contidos;
- A caracterização de uma mesma entidade pode variar, de acordo com o KOS utilizado;

- Deve haver, obrigatoriamente, um certo grau de similaridade entre os atributos de um conceito expresso em um KOS e seu correspondente no mundo real.

De acordo com Hodge (2000), os diferentes KOS podem ser agrupados em três classes, de acordo com algumas características, tais como estrutura e complexidade, relacionamento entre os termos e funções históricas: listas de termos, classificações e categorias, e listas de relacionamentos. Cada uma delas será descrita a seguir.

2.2.1 Listas de termos (LT)

São listas enfatizam os termos, geralmente com suas definições. Também são conhecidas por vocabulário controlado (*Controlled Vocabulary*) (QIN; PALING, 2001). Uma lista de termos não relaciona os termos nela contidos, cada um existe individualmente. De um modo geral, são definidos por algum tipo de autoridade no assunto, e possui duas regras básicas:

- Se um mesmo termo é utilizado para representar conceitos diferentes em diferentes contextos, então seu nome é explicitamente qualificado para resolver tal ambigüidade.
- Se múltiplos termos são utilizados para um mesmo significado, um dos termos é identificado como preferencial, e os outros são classificados como sendo sinônimos.

a) Authority Files

São listas de termos utilizadas para controlas as variantes de nomes para uma entidade ou o domínio de valores de um campo em particular. Este tipo de KOS contém uma lista de sinônimos para os termos, com uma pequena hierarquia, mas sem se aprofundar no significado dos termos. Exemplos incluem nomes de países, de pessoas, de organizações.

b) Glossários (*Glossaries*)

Lista de termos, geralmente com significados únicos, e suas descrições. São dependentes do domínio ao qual estão inseridos. Geralmente não contemplam sinônimos. A lista de palavras no final de um livro técnico é um exemplo de glossário.

c) Dicionários

Lista genérica de termos, com definições e sinônimos, mas sem hierarquias. Geralmente segue uma ordenação alfabética. Um exemplo é dicionário online ultralingua (www.ultralingua.net/index.htm).

2.2.2 Classificações e categorias

Basicamente, são listas de termos com hierarquia.

d) Subject Heading

Conjunto de termos controlados, para representar os assuntos dos itens em uma dada coleção. Apresenta uma limitada estrutura hierárquica. Como exemplo, pode-se citar o conjunto de termos médicos MeSH (HODGE, 2000).

e) Taxonomias (TX)

Conjunto de termos classificados em uma rígida estrutura hierárquica. Cada termo em uma taxonomia está presente em, ao menos, um relacionamento de herança do tipo "pai-filho" (*parent-child*) com outros termos da taxonomia. Como exemplo, qualquer modelagem orientada a objetos, na qual os termos são agrupados taxonomicamente, com base em algumas características.

2.2.3 Listas de relacionamentos

Listas de relacionamentos são taxonomias enriquecidas de outros tipos de relacionamentos. Através destes relacionamentos é possível resolver problemas de sinônimos e de homônimos.

f) Thesaurus (TH)

É um vocabulário controlado de termos, com relacionamentos associativos, além dos relacionamentos hierárquicos presentes em uma taxonomia. Os relacionamentos em um *thesaurus* são representados pelas seguintes notações:

- BT (broader term): Indica que um termo é mais genérico que outro, em uma relação de hierarquia;
- NT (narrower term): Indica que um termo é mais especializado que outro, em uma relação de hierarquia;
- SY (synonym): Indica que dois termos presentes no *thesaurus* são sinônimos;
- RT (associative or related term): Indica que dois termos presentes no *thesaurus* estão relacionados através de uma associação;

  Um exemplo de thesaurus é o getty online thesaurus de conceitos geográficos (GETTY, 2003). Mais completo que este é o *thesaurus* das nações unidas para os refugiados (UNHCR, 2003).

g) Redes semâticas

Redes semânticas são como *thesaurus* em termos de poder de expressão. Possuem os mesmos relacionamentos quem um *thesaurus*. A diferença básica está no fato que uma rede semântica não é hierárquica. Seus termos são os nodos de uma rede. A rede semântica mais conhecida é o Wordnet (MILLER, 1995), da universidade de Princeton.

h) Ontologias (ONT)

Uma ontologia é uma especificação explícita de uma conceituação (GRUBER, 1993). Mais especificamente, uma ontologia é uma teoria lógica que corresponde ao significado intencional de um vocabulário formal, ou seja, um comprometimento ontológico com uma conceituação específica do mundo (GUARINO, 1998).

Uma ontologia consiste em axiomas lógicos que contêm os significado dos termos dentro de uma comunidade. Os axiomas lógicos representam hierarquias de conceitos e relações entre eles. Uma ontologia é específica para uma comunidade e deve ser aceita em comum acordo entre os membros desta comunidade (BISHR et al., 1999).

Uma ontologia tem o mesmo poder de expressão que um *thesaurus*, mas baseia-se em um ou mais axiomas, de forma que é possível realizar inferências sobre ela. Uma ontologia deve, obrigatoriamente, ser formalizada em alguma linguagem, o que nem sempre ocorre com um *thesaurus* ou com uma taxonomia. Mais ainda, a linguagem na qual é descrita a ontologia é, por princípio, interpretável por um computador.

2.2.4 Comparação das técnicas

Dentre as principais técnicas apresentadas no capítulo 3, é possível inferir quatro diferentes possibilidades a serem empregadas na integração semântica, ou seja, serem utilizadas como o *Knowledge Organizational System* (KOS). A Figura 2.1 apresenta a relação entre as técnicas apresentadas.

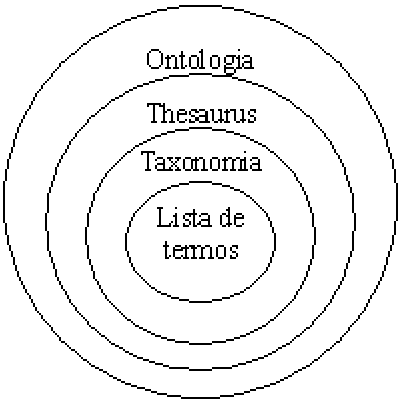


Figura 2.1 - Hierarquia das técnicas de KOS

Para decidir qual técnica utilizar, alguns critérios devem ser levados em consideração, separados em duas classes: poder de expressão e usabilidade.

2.2.4.1 Poder de expressão

O primeiro conjunto de critérios compara os diferentes KOS em termos de poder de expressão. Cada um dos critérios a seguir representa uma importante característica que o KOS deve possuir para alcançar as necessidades da proposta deste trabalho.

a) Definição conceitos

Consiste na capacidade do sistema de poder criar novos termos, com suas propriedades e sua descrição;

b) Definição de sinônimos

O sistema deve ser capaz de representar um mesmo conceito do mundo real, mas que pode ter diferentes nomes (termos).

c) Tratamento de homônimos

O sistema deve ter mecanismos para diferenciar quando dois conceitos diferentes são descritos por um mesmo termo.

d) Construção de hierarquias

Deve ser possível descrever hierarquias de generalização e especialização entre os conceitos modelados, uma vez que assim pode-se saber que uma instância de um conceito também é uma instância do(s) conceito(s) que é(são) superclasse(s) dele.

e) Relacionamentos entre os termos

Além do relacionamento de hierarquias, o sistema deve ser capaz de descrever relacionamentos de outros tipos, tais como associação, agregação e composição. Desta forma, a escolha por algum conceito dependente do contexto pode ser feita.

f) Linguagem para formalizar a técnica

Para que a técnica de organização do conhecimento possa ser utilizada de forma (semi-) automatizada, é necessário que exista uma linguagem que a descreva e que possa ser interpretada por computador.

A Tabela 2.2 apresenta uma comparação entre as técnicas, baseadas nos critérios acima estabelecidos.

Tabela 2.2 - Comparativo do poder de expressão das técnicas de KOS

Critério	LT	TX	TH	ONT
A	Sim	Sim	Sim	Sim
B	Sim	Sim	Sim	Sim
C	Sim	Sim	Sim	Sim
D	Não	Sim	Sim	Sim
E	Não	Não	Sim	Sim
F	Não	Não	Sim	Sim

A aplicação do KOS a ser implementado no sistema é para a integração semântica de esquemas conceituais de BDG, e para o suporte à modelagem de novos esquemas. Desta forma, é importante que todos os critérios sejam satisfeitos. Assim, as técnicas de lista de termos e taxonomias não são adequadas. As opções restantes são *thesaurus*, com algumas limitações, e ontologias.

2.2.4.2 Usabilidade

O segundo conjunto de critérios diz respeito à facilidade de implementação da técnica de KOS em um ambiente computacional. Cada um dos critérios aponta para um requisito que facilite a utilização do KOS.

g) Padronização da linguagem de descrição

Um critério bastante importante, mas não exclusivo, diz respeito à linguagem que descreve os conceitos modelados. Preferencialmente ela deve ser um padrão aceito para tal finalidade, pois facilita o intercâmbio de informações e é mais provável que existam ferramentas para gerenciar e processar.

h) Independência da linguagem

A linguagem na qual os conceitos são descritos não deve ser proprietária de nenhuma ferramenta. Ela deve ser independente.

i) Ferramentas computacionais para gerenciar e processar

Uma vez que é necessário processar a ontologia de forma (semi-)automática, é desejável que existam ferramentas computacionais capazes de manipular os conceitos do sistema de organização de conhecimento, ou que ferramentas possam ser desenvolvidas. Por questões de tempo, se as ferramentas já estiverem disponíveis, é mais um critério a ser levado em consideração. A manipulação deve ser em nível de consulta e atualização.

j) Custo das ferramentas computacionais

Este é outro critério bastante importante, uma vez que, por se tratar de um trabalho acadêmico, se a ferramenta for grátis é mais provável que seja utilizada.

A Tabela 2.3 apresenta a comparação de usabilidade das técnicas de KOS pré-selecionadas, ou seja, ontologias e *thesaurus*.

Tabela 2.3 - Comparação da usabilidade de *Thesaurus* X Ontologias

Critério	TH	ONT
G	Não	Sim
H	Não	Sim

I	Sim	Sim
J	Sim	Sim

2.2.5 A escolha por ontologias

A partir da comparação das técnicas de KOS apresentadas neste capítulo, a técnica escolhida para ser utilizada na integração semântica a qual este trabalho se propõe é a de ontologias. A ontologia é a única das técnicas que satisfez a todos os requisitos. Como pode ser inferido pela Figura 2.1, tudo que é possível fazer com qualquer uma das outras técnicas, pode ser feito com a utilização de ontologias.

Quando se trata de modelagem de banco de dados geográficos, a ontologia apresenta ainda uma outra vantagem, que é o fato de poder ser utilizada também como suporte a esta tarefa. BDGs visam à modelagem de fenômenos da realidade, ou seja, conceitos existentes no mundo real. Desta forma, o conjunto de elementos a serem modelados é bastante restrito e concreto. Os atributos e relacionamentos entre os elementos geográficos são, muitas vezes, os mesmos. O que muda é a abordagem feita, dependendo da aplicação e do conhecimento do projetista, e também os nomes empregados para representar as mesmas coisas.

A criação de uma ontologia contendo os fatos (fenômenos, no caso de BDG) e relacionamentos do domínio de interesse pode contribuir para o projeto de banco de dados em geral, e, por conseguinte, de BDG (SUGUMARAN; STOREY, 2002):

- O projetista pode comparar sua modelagem com a ontologia existente, de forma a detectar possíveis inconsistências e não completudes, tais como a falta de entidades, atributos e relacionamentos;
- Um projetista não muito familiarizado com o domínio da aplicação pode basear sua modelagem nos conceitos presentes na ontologia, desde o princípio.

A Figura 2.2 ilustra o processo de utilização da ontologia na modelagem conceitual de banco de dados geográficos.

Neste ponto cabe uma discussão a respeito das diferenças entre esquemas conceituais e ontologias. Embora alguns autores considerem que um esquema conceitual é uma ontologia, com menor nível de detalhamento semântico, e outros autores considerem ainda que a ontologia é um tipo de esquema conceitual, a abordagem aqui adotada é que esquemas conceituais e ontologias não são a mesma coisa.

De acordo com Cui, Jones e O'Brien (2002), uma ontologia é desenvolvida com o objetivo de definir o significado dos termos utilizados em um determinado domínio, enquanto um esquema conceitual é desenvolvido com o intuito de modelar alguns dados. Fonseca, Davis e Câmara (2003) adicionam o fato de que uma ontologia é externa aos sistemas de informação, e busca especificar várias visões possíveis do mundo. Já um esquema conceitual é interno ao sistema de informação e é modelado como sendo a especificação de uma única visão possível do mundo.

Uma ontologia é semanticamente mais rica que um esquema conceitual, e assim mais próxima ao modelo cognitivo do ser humano. Uma ontologia representa conceitos do mundo real. Um esquema conceitual é desenvolvido para organizar o que será armazenado no banco de dados (FONSECA ;DAVIS; CÂMARA, 2003).

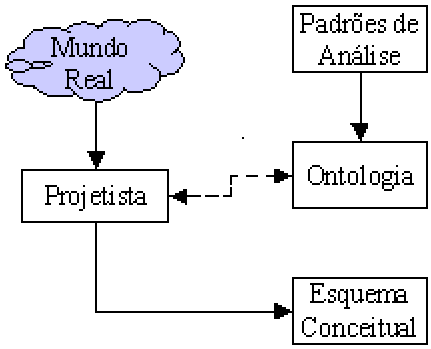


Figura 2.2 - Aplicação de KOS na modelagem conceitual de BDG

2.3 Detalhando ontologias

De uma forma genérica, a ontologia pode ser utilizada tanto em tempo de desenvolvimento (*development time)* quanto em tempo de execução (*run time*) (GUARINO, 1998). No primeiro caso, o projetista de uma aplicação pode valer-se da ontologia como forma de armazenamento dos componentes do sistema de informação. Segundo Guarino (1998), em um cenário onde haja uma ontologia de aplicação (veja seção 2.3.1), a qual contenha todos os elementos necessários ao sistema, o projetista simplesmente traduz os conceitos da ontologia para componentes do banco de dados. Em um segundo cenário, a ontologia existente é chamada de ontologia de domínio, e serve como um *framework* para o projetista criar seus próprios elementos, utilizando a ontologia como um guia, tal qual uma ferramenta CASE, mas mais poderosa, com aspectos semânticos.

A utilização da ontologia em tempo de desenvolvimento, também chamado de Engenharia de Sistemas, traz um conjunto de benefícios (USCHOLD; GRUNINGER, 1996):

- Reuso: A partir do armazenamento do entendimento compartilhado a respeito das entidades, atributos, métodos e relacionamentos dos conceitos de um dado domínio, de modo formal e que permita a recuperação automática, é possível reutiliza-los várias vezes;
- Correção: Uma representação formal do conhecimento permite a verificação automática da consistência dos componentes do sistema;
- Especificação: Define uma conceitualização formal e única para os conceitos armazenados, resolvendo conflitos terminológicos entre, por exemplo, os membros de uma equipe de desenvolvimento de software.

A utilização de ontologias em tempo de execução diz respeito à comunicação entre diferentes sistemas. Nestes casos, agentes de software ficam encarregados de acessar a base de conhecimento, com o objetivo de garantir a comunicação entre os sistemas, de forma correta e não ambígua (GRUBER, 1998; USCHOLD; GRUNINGER, 1996).

A integração de sistemas heterogêneos é principal das aplicações da ontologia. Diversos são os casos de integração que podem valer-se da ontologia: sistemas, domínios e aplicações distintas (USCHOLD; GRUNINGER, 1996). Em todos os casos, a idéia principal é a ontologia servir como um mediador terminológico (MUÑOZ, 2002), ou seja,

através da ontologia deve ser possível fazer os mapeamentos entre os termos dos diferentes sistemas.

O caso de sistemas distintos divide-se em interoperabilidade interna e externa. A primeira ocorre em sistemas de uma mesma empresa, na qual as heterogeneidades ocorrem por razões históricas, como em sistemas legados que não são mais alterados. Necessidade de interoperabilidade externa ocorre quando os sistemas não pertencem a uma mesma organização, ou a diferentes departamentos de uma mesma empresa, desenvolvidos de forma independente.

A ontologia deve tratar de integrar de domínios distintos quando as áreas de aplicação dos sistemas não forem as mesmas. A interoperabilidade entre ferramentas distintas é auxiliada por uma ontologia de forma a resolver os conflitos gerados pelas diferentes bases de dados das ferramentas, as quais geralmente não tem o conhecimento armazenado formalmente.

Outras aplicações de ontologias que merecem ser citadas são processamento de linguagem natural, modelagem de empresas e indexação de *sites* na internet, entre outros (MUÑOZ, 2002). A aplicação que é o foco deste trabalho é a interoperabilidade de sistemas (interna e externa), que no contexto deste trabalho são os esquemas conceituais.

2.3.1 Classificação de ontologia

De acordo com a generalidade da ontologia, ela pode ser definida em diferentes tipos, conforme ilustrado na Figura 2.3 (GUARINO, 1998).

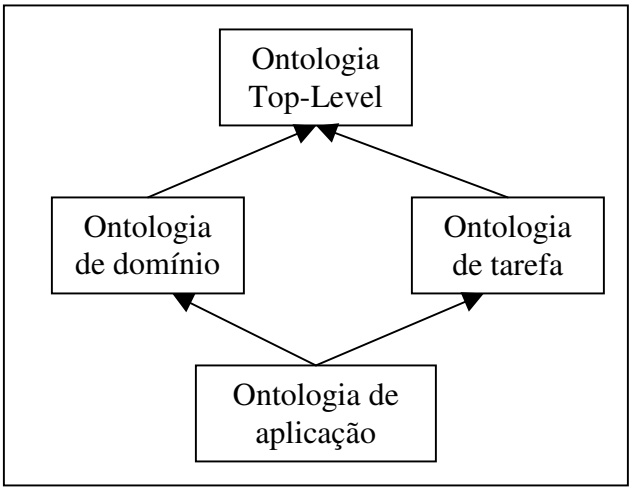


Figura 2.3 - Classes de ontologias

Uma ontologia do tipo *Top-level* descreve os conceitos mais genéricos e abrangentes, tais como espaço, tempo, objeto, evento, ação, etc. Estes conceitos são totalmente independentes do domínio da aplicação, ou seja, valem sempre. Este tipo de ontologia é útil para, por exemplo, unificar o vocabulário de determinada comunidade.

Ontologias de domínio descrevem o vocabulário de um domínio genérico, tal como medicina, geografia, automobilismo, entre outros (GUARINO, 1998). Define restrições na estrutura e conteúdo do domínio particular ao qual se refere (MUÑOZ, 2002). Para este domínio, qualquer que seja a aplicação pode fazer uso de seus conceitos.

A ontologia de tarefa descreve o vocabulário de uma tarefa ou atividade genérica em um contexto específico, como, por exemplo, vendas, diagnóstico, etc. (GUARINO, 1998).

Estas tarefas podem ser dependentes do domínio ou não, como é o caso do diagnóstico, que pode ser médico ou de mecânica de automóveis. O domínio não é o mesmo, mas as atividades de exame e hipótese ocorrem em ambos os casos.

Ontologias de aplicação descrevem conceitos que dependem tanto de um domínio específico quanto de uma tarefa particular. Estes conceitos costumam ser os papéis desempenhados pelas entidades do domínio quando da execução de uma determinada tarefa (GUARINO, 1998). Por exemplo, o conceito de GPS (*Global Position System)* é típico de uma tarefa de posicionamento, no domínio geográfico.

2.3.2 Componentes de uma ontologia

Uma ontologia é composta, basicamente, por conceitos, atributos dos conceitos, relacionamentos entre os conceitos, funções (ou métodos) dos conceitos, axiomas aplicados sobre os conceitos e instâncias dos conceitos. Os conceitos estão organizados de forma taxonômica, ou seja, sob uma hierarquia (NOY; McGUINNESS, 2001).

O conceito, também chamado de classe, é o elemento central de uma ontologia. Um conceito é qualquer fenômeno que se deseje caracterizar e armazenar na ontologia. Cada conceito possui um termo que o identifica, e um conjunto de outros componentes que o caracterizam, que são os atributos, relacionamentos, funções e axiomas. Os atributos (ou *slots*) descrevem as propriedades das classes e das instâncias das classes (NOY; McGUINNESS, 2001).

Os relacionamentos especificam como os conceitos de uma ontologia interagem entre si. Uma vez que a ontologia é, por natureza, hierárquica, obrigatoriamente deve conter relacionamentos taxonômicos, do tipo generalização-especialização (*IS-A*), no qual um conceito é sub-conceito de outro mais genérico. O relacionamento de mereologia ocorre quando um ou mais conceitos agregados (ou compostos), chamados partes (*part*), formam um outro conceito, chamado todo (*whole*). Tipicamente este relacionamento é conhecido como *part-of*. Os relacionamentos binários simples também estão presentes em uma ontologia, tendo a mesma semântica das associações binárias em esquemas conceituais. Outros relacionamentos também podem ser definidos, especialmente quando da criação de uma ontologia do domínio. Por exemplo, no caso de uma ontologia geográfica, relacionamentos espaciais, tais como interseção, pertinência e vizinhança, podem ser introduzidos.

As funções definidas para um conceito são operações a serem realizadas sobre as suas instâncias, podendo ou não envolver instâncias de outros conceitos. Os axiomas, por sua vez, são restrições que devem sempre ser verdades na ontologia, e nenhuma instância do conceito pode violar. Um exemplo de axioma é a determinação de que a população de uma cidade não pode ser maior que a população do estado ao qual a cidade pertence.

Por fim, as instâncias são os objetos que fazem parte de um determinado conceito da ontologia. Por exemplo, Porto Alegre é uma instância do conceito cidade.

3 O módulo de integração semântica

Para que a integração de aplicações geográficas seja possível, é necessário que três requisitos sejam satisfeitos (BERGAMASCHI et al., 1998):

- Os esquemas conceituais de cada fonte de dados estejam disponíveis;
- Haja informação semântica associada a cada esquema;
- Haja um modelo de dados comum, canônico, que seja capaz de descrever os esquemas a serem integrados.

Uma vez que a integração objetivada na pesquisa a qual este trabalho está vinculado é, justamente, de esquemas conceituais, o primeiro requisito é, automaticamente, satisfeito. Os demais requisitos são preenchidos através da utilização do trabalho desenvolvido por (BASSALO; IOCHPE; BIGOLIN, 2002) e da utilização de um formato de dados geográficos padrão.

Através da utilização da ontologia para eliminação de heterogeneidades semânticas, além de poder minerar os esquemas conceituais para obtenção de candidatos a padrão de análise de BDG, serão alcançadas também, pelo menos, outras três capacidades (SHETH, 2000):

- Transparência terminológica: É a característica mais elementar. Ambigüidades geradas por homônimos e sinônimos são eliminadas;
- Processamento sensível ao contexto: De acordo com o contexto (relacionamentos e atributos) no qual um conceito está inserido, é possível inferir sua semântica;
- Correlação semântica: Integração entre esquemas conceituais, combinando os dois aspectos acima mencionados.

Para realizar a integração semântica, primeiramente foi necessário desenvolver uma ontologia geográfica, a qual deu-se o nome de ontoGeo. A idéia não era criar uma ontologia de domínio completa, mas sim uma estrutura que pudesse ser utilizada como ponto de partida para a integração dos esquemas conceituais. Através da execução do algoritmo de comparação dos conceitos de um esquema conceitual de entrada frente a ontoGeo, a integração semântica é alcançada, conforme será descrito neste capítulo, tornando os esquemas livres de heterogeneidades. Adicionalmente, a ontologia pode ser atualizada automaticamente durante o processamento do algoritmo, quando um conceito não for nela encontrado.

3.1 OntoGeo: a ontologia geográfica

Para a criação de uma ontologia geográfica, a qual deu-se o nome de ontoGeo, a classe escolhida foi a de ontologias de domínio. Desta forma, qualquer aplicação geográfica pode valer-se dos mesmos conceitos armazenados.

Integrando os conceitos genéricos de alguns dos modelos de dados propostos, é possível englobar praticamente todos os elementos existentes para aplicações geográficas. Desta forma, cinco conjuntos de conceitos podem ser definidos: classes, representações espaciais, relacionamentos, temporalidade e redes. Nas subseções a seguir será descrito cada um destes conjuntos de classes.

3.1.1 Classes

Todo e qualquer conceito que representa uma classe em uma modelagem conceitual de BDG é descrito como sendo uma instância desta meta-classe. Como sub-classes da classe abstrata *Classes*, foram criados os conceitos de *FenGeo*, para representar fenômenos geográficos, e *FenNGeo*, para conceitos que representam classes de fenômenos convencionais, ou seja, não geográficos. Um *FenGeo* também é uma classe abstrata, e se especializa nas classes geográficas de aplicação propriamente ditas. A Figura 3.1 apresenta a hierarquia de C*lasses* da ontologia.

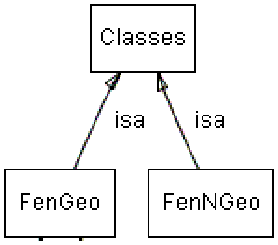


Figura 3.1 - Hierarquia de *Classes*

Com estes meta-conceitos é possível descrever qualquer um dos fenômenos geográficos descritos pelos modelos UML-GeoFrame (LISBOA; IOCHPE, 1999; ROCHA; EDELWEISS; IOCHPE, 2001), o MADS (PARENT et al., 1999), o OMT-G (BORGES, 1997).

3.1.2 Representações espaciais

Cada fenômeno geográfico possui uma geometria associada. As representações espaciais são descritas também como conceitos na ontologia. Estas geometrias são subclasses de *RepEspacial*, a qual é abstrata. Cada geometria contínua é uma subclasse de *RepCampo*, ao passo que geometrias discretas são subclasses de *RepObj*. As classes geométricas adicionadas inicialmente são aquelas presentes nos modelos geográficos MADS, UML-GeoFrame e OMT-G.

A Figura 3.2 ilustra as hierarquias de representações espaciais, construídas com base no software Protégé (NOY et. al, 2000).

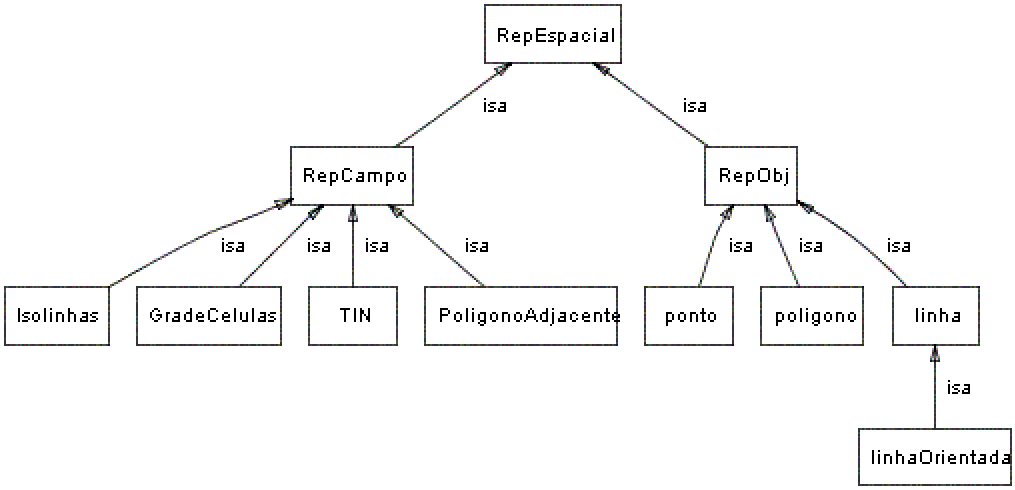


Figura 3.2 - Conceitos geométricos da ontologia

A associação entre fenômenos geográficos e suas representações espaciais é feita através de *slots*. Cada *slot* pode ser visto como um atributo da classe de fenômeno geográfico, cujo domínio é uma subclasse de *RepCampo* ou *RepObj*.

3.1.3 Relacionamentos

A ontologia geográfica definida neste trabalho possui os relacionamentos convencionais (não espaciais) entre conceitos, mas também é possível representar os relacionamentos espaciais, tais como interseção, vizinhança, sobreposição, etc., bem como agregações e composições. Para tanto, simplesmente os relacionamentos recebem o nome do tipo de associação que eles devem representar, conforme ilustrado na Figura 3.3. O domínio de cada associação é uma outra classe.

As associações ditas convencionais entre os conceitos da ontologia recebem um nome específico, assim como ocorre em um esquema conceitual. Não há nomes fixos. Inicialmente os nomes são dados concatenando-se os nomes dos dois conceitos associados, separados por um caracter '_'. Assim como entre os conceitos e suas geometrias, uma associação entre dois conceitos é feita através de *slots*. Cada *slot* pode ser visto como um atributo da classe de fenômeno geográfico, cujo domínio é uma subclasse de *Classe* (se for genérica) ou *FenGeo*, se for um relacionamento espacial.

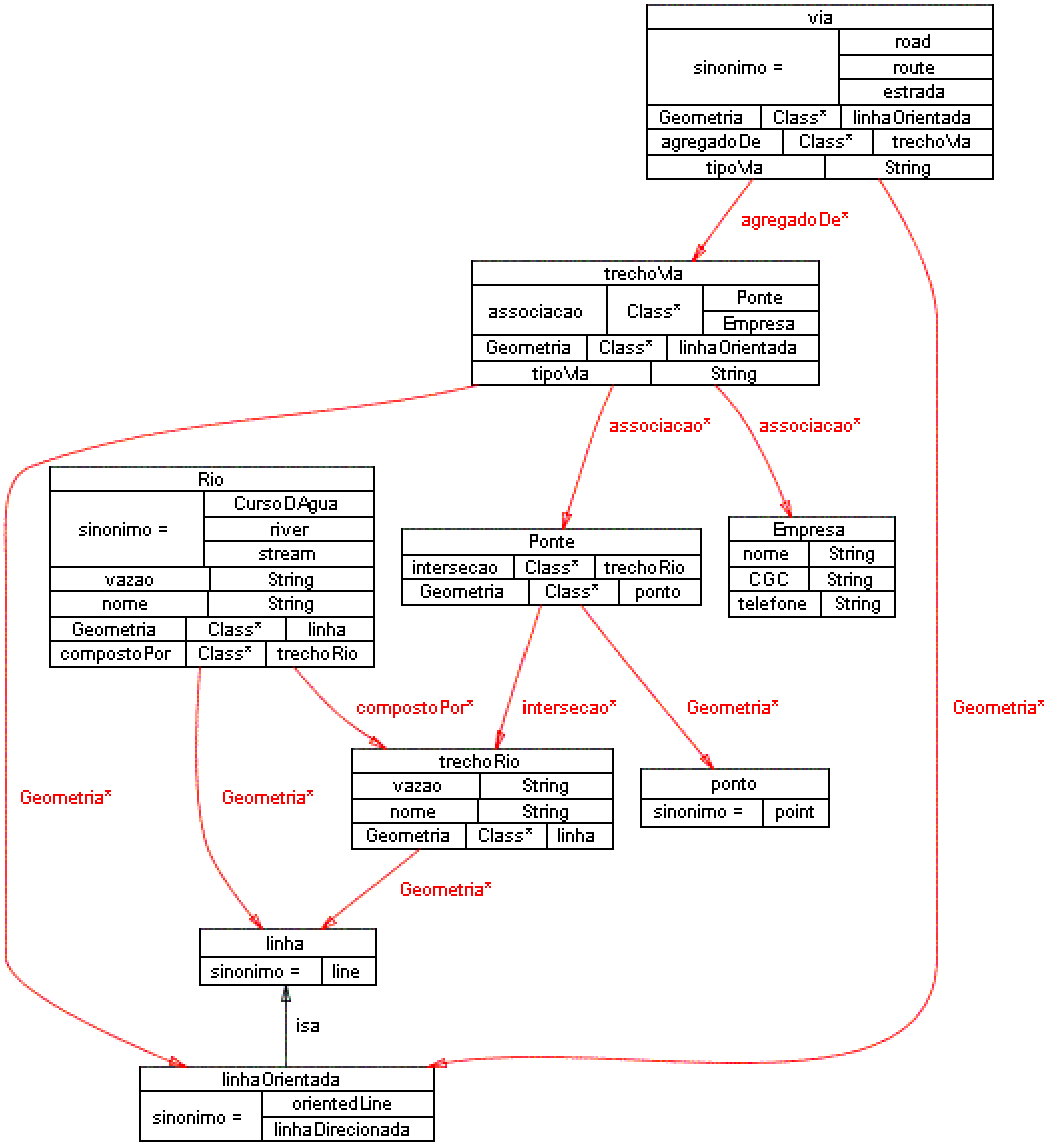


Figura 3.3 - Exemplo de hierarquia de classes e relacionamentos na ontologia

3.1.4 Temporalidade

A medida do tempo pode ser feita em intervalos (períodos) ou em instantes. Desta forma, assim como no MADS e no GeoFrame, foi criado um conceito chamado *Tempo*, que possui como sub-conceitos *ITempo e PTempo*, para instantes e períodos, respectivamente. Adicionalmente, devido à existência do tempo de transação e tempo de validade, um atributo chamado *TipoTempo* foi criado para *Tempo*, podendo assumir os valores 0 para transação e 1 para validade. A Figura 3.4 apresenta a hierarquia temporal.

A temporalidade pode ocorrer tanto em classes geográficas quanto em classes não geográficas. Para associar-se um conceito que representa uma classe da modelagem conceitual aos conceitos que descrevem tempo, foi definido um outro relacionamento, como pode ser visto na Figura 3.5 *ClTempo* associa uma especialização de *Classes* com uma das classes da hierarquia de *Tempo*.

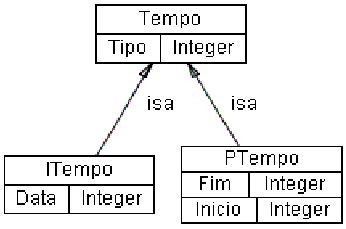


Figura 3.4 - Hierarquia temporal da ontologia

Em nível de atributos, a temporalidade é dada através do domínio do atributo, o qual é especificado como sendo uma referência a uma das classes temporais (instante ou período).

Relacionamentos temporais são criados através de *slots* com dois domínios. O primeiro é a outra classe associada, e o segundo é o elemento temporal que representa a associação.

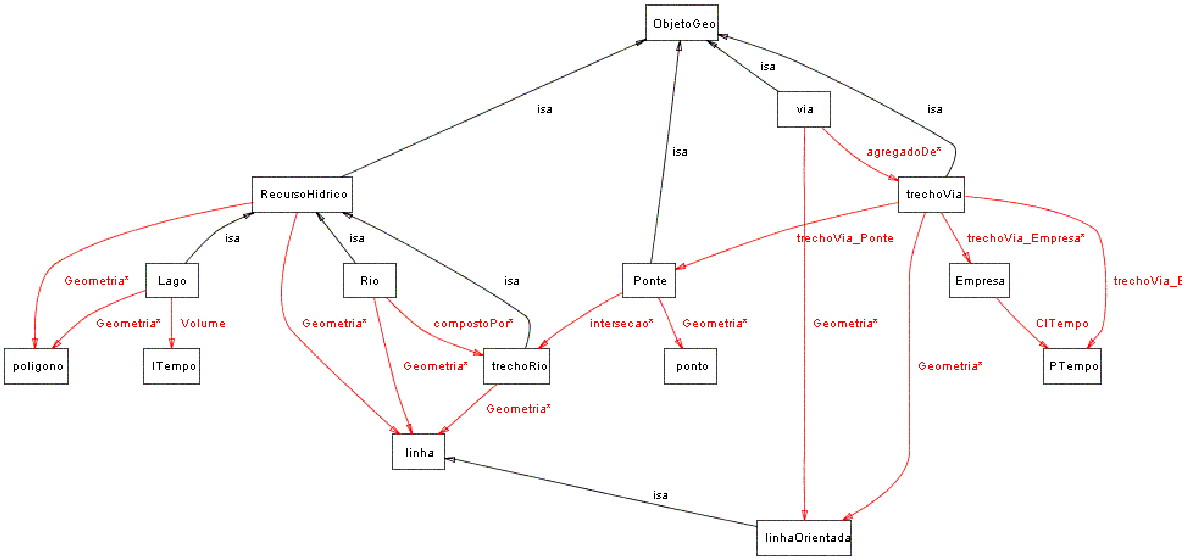


Figura 3.5 - Classes, atributos e associações temporais

Na Figura 3.5, o relacionamento *ClTempo* entre empresa e *PTempo* significa que o conceito empresa é temporal. A associação *Volume* entre *Lago* e *ITempo* significa que o atributo chamado volume do conceito lago é um instante temporal. A associação *trechoVia_Empresa* entre *trechoVia* e *PTempo* e entre *trechoVia* e *Empresa* significa que a associação de uma empresa com um trecho de via é temporal, valendo por um determinado período de tempo.

Os relacionamentos dinâmicos existentes no MADS podem facilmente ser implementados na ontologia, através da definição de *slots* com os nomes destas associações, cujos domínios sejam outros conceitos geográficos existentes na ontologia.

3.1.5 Topologia de redes

Para a definição de topologia de redes, foi definido um novo conceito na ontologia chamado *rede*. Este conceito especializa-se em um chamado *nodo*, o qual também é uma especialização de *FenGeo*, conforme ilustra a Figura 3.6. Um *arco* é um conceito que também se especializa de *rede* e de *FenGeo*. A diferença para um *nodo* está no fato que sua geometria está restrita a uma linha ou alguma especialização dela. Além dos atributos

herdados de *FenGeo*, um *arco* possui um *slot* chamado *assocRede*, cuja semântica é associa-lo a dois conceitos do tipo *nodo*.

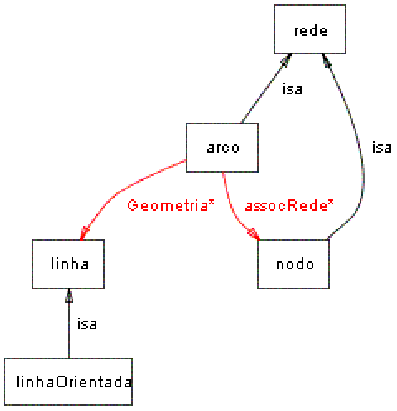


Figura 3.6 - Topologia de redes na ontologia

3.2 O conceitos descritos na ontoGeo

A ontologia ontoGeo foi criada, em sua primeira versão, com os conceitos pertencentes à cartografia básica, tais como hidrografia, relevo e vegetação. Alguns conceitos relativos à infra-estrutura também foram armazenados, especialmente no que diz respeito aos transportes. O tema de localidade também é representado na ontoGeo com conceitos como cidades, distritos, bairros e cidades, entre outros. A Figura 3.7 apresenta os temas englobados pela ontoGeo.

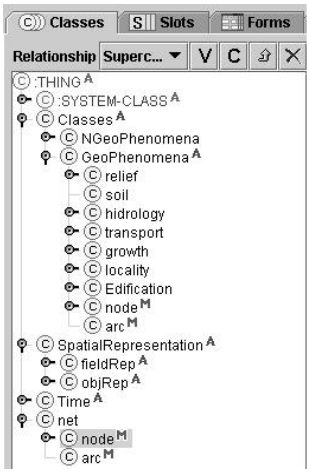


Figura 3.7 - A estrutura de classes da ontoGeo

A versão inicial da ontologia foi construída no software Protégé (NOY et. al, 2000) e seu código é escrito na linguagem semi-estruturada RDF Schema (W3C 2004). Conforme ilustrado pela Figura 3.8 a ontoGeo possuía 173 classes, das quais 14 eram classes de sistema e 159 definidas especificamente para o propósito da ontologia geográfica. Em termos de atributos e relacionamentos, descritos como *slots*, havia um total de 137 diferentes, dos quais 33 eram de sistema e 104 definidos neste trabalho.

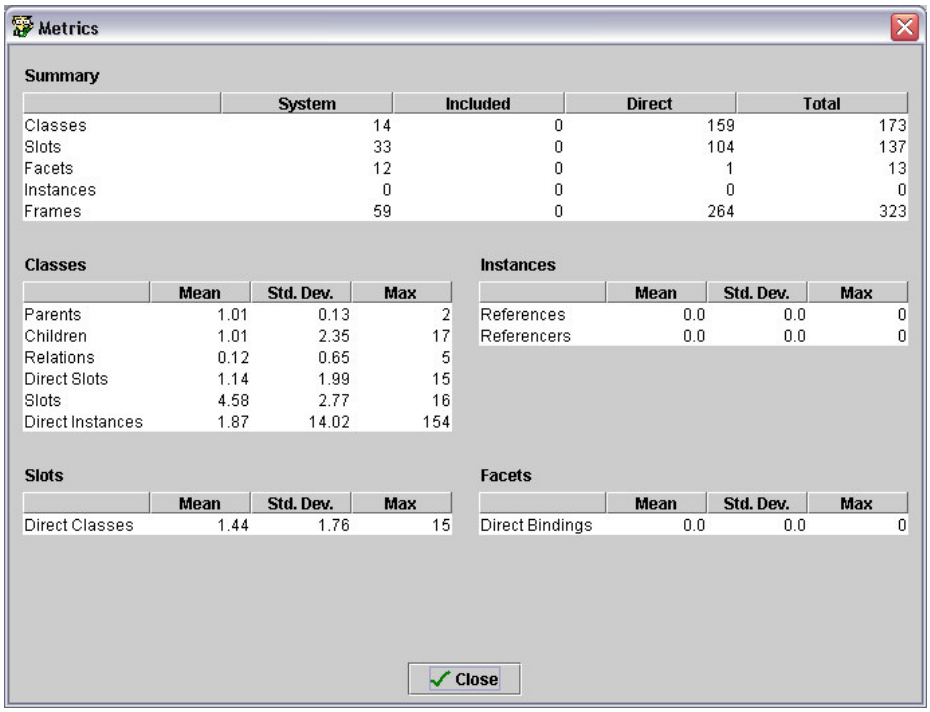


Figura 3.8 - métricas da ontoGeo

Os conceitos representados na ontoGeo foram retirados de diferentes esquemas conceituais de BDG, baseados em diversos modelos de dados e modelados por diferentes pessoas.

Um aspecto importante de salientar é que o modelo de conhecimento utilizado na ontologia está baseado no modelo de conhecimento do software Protégé (NOY et. al, 2000), o qual é, por sua vez, compatível com o modelo OKBC (Open Knowldge Base Conectivity) (CHAUDHRI et. al, 1998). O modelo do Protégé foi estendido, de modo a aceitar a definição de sinônimos, tanto para conceitos quanto para *slots*. A Figura 3.9 apresenta a definição do conceito *Rio*, com seus *slots* e sinônimos.

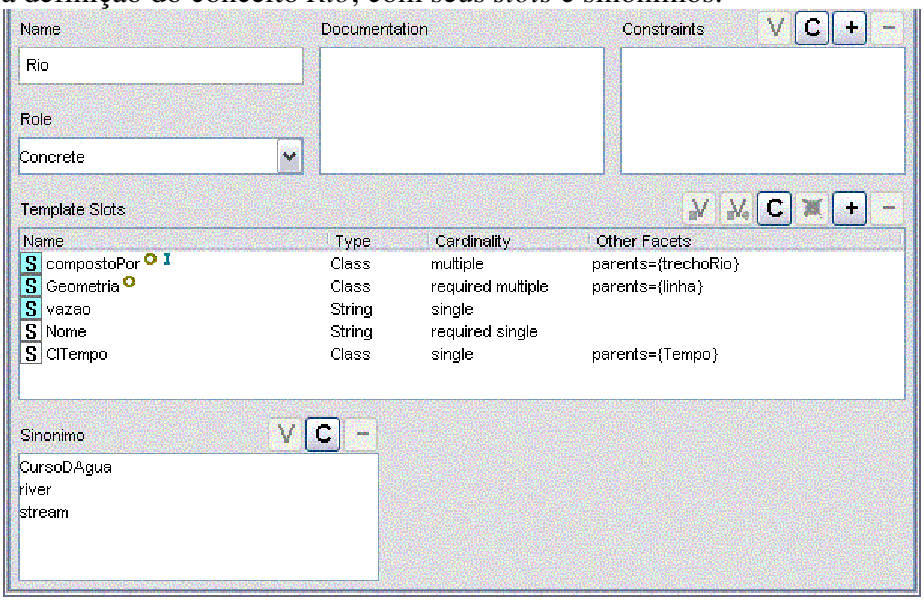


Figura 3.9 - Definição do conceito *Rio*

3.3 O processo de consulta e atualização da ontologia

De acordo com Gotthard, Lockemann e Neufeld (1992), a identificação de conflitos baseia-se no reconhecimento de que diferentes representações, em diferentes esquemas, referem-se a uma mesma entidade do mundo real. Ou seja, apenas após a identificação de alguma similaridade torna-se possível relacionar conceitos equivalentes e identificar divergências e complementações nas representações de um mesmo objeto do mundo real.

O processo de integração de esquemas deve combinar diferentes visões de usuários em uma única visão. O propósito da integração de esquemas é encontrar todas as partes dos esquemas conceituais de entrada que se referem a uma mesma porção da realidade, e unificar sua representação. Através do processo de integração de esquemas, várias visões conflitantes de usuários devem ser mescladas e integradas em um ou mais esquemas globais (PARK, 2001). No caso da integração proposta para este trabalho, o esquema conceitual global é a ontologia. A Figura 3.10 apresenta o diagrama de atividades do algoritmo de busca e atualização dos termos na ontologia.

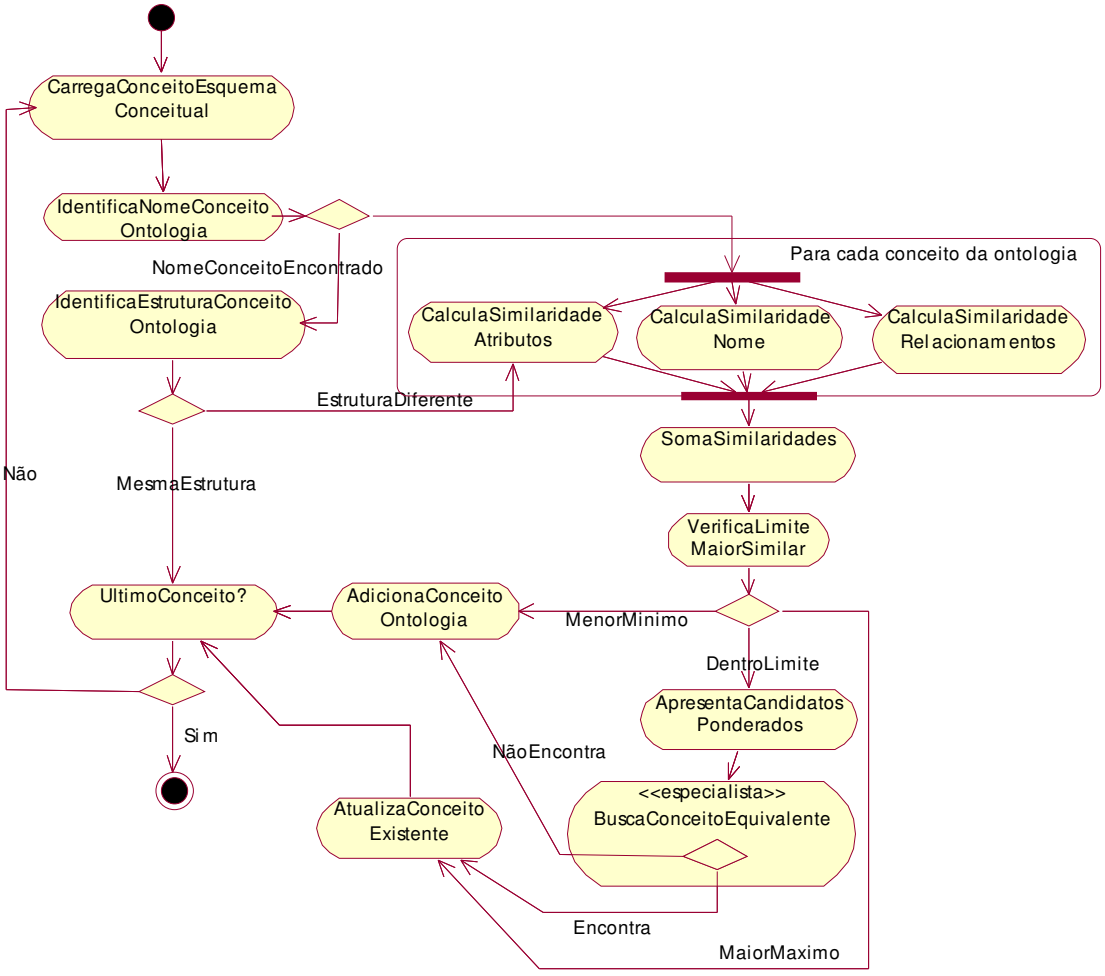


Figura 3.10 - Diagrama de atividades para consulta e atualização da ontologia

3.3.1 O algoritmo proposto para consulta e atualização da ontologia

O algoritmo descrito a seguir detalha, em um alto nível de abstração, a seqüência de passos para a consulta e atualização da ontologia. O algoritmo é semi-automatizado. A completa automatização não é possível em função da necessidade de intervenção do especialista em momentos de decisão nos quais o algoritmo não tenha obtido resultados conclusivos.

Para diminuir a necessidade de intervenção do especialista, dois parâmetros devem ser informados no início do processo, o limiar de análise e o limiar de aceitação. O limiar de análise serve para limitar o número de candidatos possíveis que devem ser avaliados pelo especialista. Se nenhum candidato alcançar o limiar de análise, o conceito analisado será considerado como não existente na ontologia. O limiar de aceitação serve para, se algum candidato ultrapassá-lo, o sistema considera automaticamente que este conceito da ontologia é o equivalente ao conceito que está sendo processado. Um terceiro parâmetro deve ser indicado pelo usuário, para aumentar a confiabilidade do algoritmo. Trata-se de um valor de "confiança", ou seja, um *delta* que indica a margem de erro aceita no cálculo da similaridade. No caso do limite de análise, são considerados todos os candidatos que estiverem acima do valor de "limiar de análise – *delta*". Da mesma forma, se um candidato ultrapassar a similaridade de aceitação, ele não será automaticamente considerado como sendo o correto. Serão considerados todos os candidatos que apresentarem similaridade igual ou superior a "maior similaridade obtida – *delta*". A Figura 3.11 ilustra o funcionamento do fator *delta*.

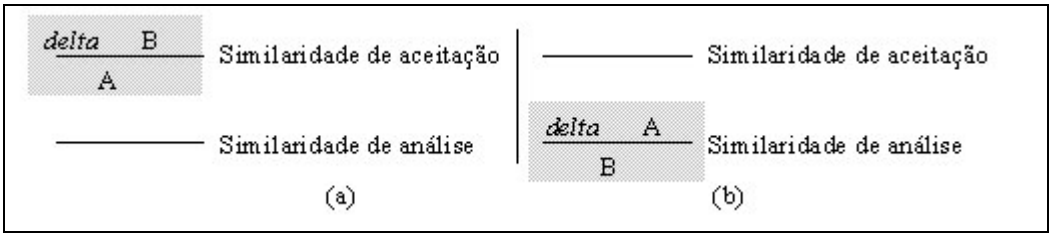


Figura 3.11 – Aplicação do parâmetro *delta*

Para que o algoritmo funcione corretamente, faz-se necessário que todo esquema conceitual a ser processado contenha um metadado indicando o idioma base da modelagem.

Passo 0 – tradução do esquema conceitual para o idioma da ontologia: Caso o idioma da ontologia não seja o mesmo indicado pelo metadado do esquema conceitual, ele deve ser traduzido, com o auxílio de um dicionário. Uma vez estando no idioma correto da ontologia, cada termo presente no esquema conceitual é buscado na ontologia.

Passo 1 – Consulta nome do conceito na ontologia: Se o nome do conceito for encontrado na ontologia, ou algum de seus sinônimos ou acrônimos (abreviaturas), passa para o passo 2. Caso o termo que dá nome ao conceito não seja encontrado, passa para os passos 4, 5 e 6, em paralelo.

Passo 2 – Consulta estrutura conceito na ontologia: Uma vez tendo o termo que nomeia o conceito, varre atributo por atributo do conceito no esquema conceitual e verifica se todos possuem correspondente na ontologia. Em caso positivo, passa para o passo 3. Se houver diferença na estrutura do conceito (algum atributo), vai para o passo 5.

Passo 3 – Testa se há mais conceitos: Procura o próximo conceito (elemento) do esquema conceitual. Caso haja mais conceitos, volta para o passo 1. Caso contrário, passa para o fim.

Passo 4 – Calcula similaridade do termo que dá nome ao conceito: Baseada em algum método, é calculada a similaridade do nome do conceito de entrada com relação aos demais conceitos presentes na ontologia. Passa para o passo 7.

Passo 5 – Calcula similaridade estrutural do conceito: Baseada em algum método, a similaridade da estrutura de um conceito do esquema conceitual de entrada é calculada, em termos de seus atributos, frente aos demais conceitos presentes na ontologia. Passa para o passo 7.

Passo 6 – Calcula similaridade dos relacionamentos do conceito: Baseada em algum método, a similaridade dos relacionamentos de um conceito é calculada em termos de taxonomia (especializações / generalizações) e de agregação e composição, frente aos demais conceitos presentes na ontologia. Passa para o passo 7.

Passo 7 – Soma das similaridades: Baseada em algum método de ponderação, a similaridade de nome e de estrutura de um conceito presente no esquema conceitual de entrada é somada, resultando em uma probabilidade de similaridade geral, com relação a cada conceito da ontologia. Vai para o passo 8.

Passo 8 – Verifica limite similaridade candidatos: Seleciona-se o conceito candidato que apresentar a maior probabilidade de similaridade com o conceito de entrada. Caso seu valor de similaridade for menor que o limite de análise estabelecido pelo usuário menos o parâmetro *delta*, vai ao passo 12. Se for superior ao limite de aceitação estabelecido pelo usuário e não houver nenhum outro candidato cuja probabilidade seja maior que a maior probabilidade encontrada menos o parâmetro *delta*, vai para o passo 11. Caso contrário, vai para o passo 9.

Passo 9 – Apresenta candidatos ponderados: Apresenta todos os candidatos encontrados, com a probabilidade de similaridade ponderada. Caso haja candidatos com similaridade superior ao limite de aceitação, somente são apresentados os candidatos dentro da variação *delta* de probabilidade. A ordem é decrescente, ou seja, aqueles com probabilidade maior primeiro. Somente são mostrados aqueles com probabilidade maior que o limite de análise especificado pelo usuário. Vai para o passo 10.

Passo 10 – Seleção do termo: Aqui se faz necessária a intervenção do especialista do domínio. O especialista seleciona o conceito que ele julga ser o mais adequado para representar o conceito do esquema de entrada, dentre os candidatos apresentados. Caso tenha sido selecionado um conceito existente para representar o conceito do esquema de entrada, vai para o passo 11. Se for definido que é um novo conceito a ser adicionado na ontologia, pula para o passo 12.

Passo 11 – Atualização de um conceito existente: Dependendo da proveniência da chamada desta etapa, uma atualização diferente na ontologia é feita. Esta pode ser a adição de um novo sinônimo ou acrônimo para um conceito já existente, a adição de algum novo atributo à sua estrutura, ou ainda a criação de um novo relacionamento entre o conceito e algum outro presente na ontologia. Volta para o passo 3.

Passo 12 – Adição de um novo conceito na ontologia: Um novo conceito é adicionado na ontologia, com todos os seus atributos e relacionamentos. Volta ao passo 3.

3.4 Técnicas complementares do processo

A utilização de ontologias não resolve, de um todo, o problema da integração semântica. Não é possível que a ontologia represente toda a semântica existente no mundo real, tanto pelas restrições inerentes à ontologia ou a qualquer outra técnica de KOS, como pelas diferenças decorrentes do processo individual de interpretação da realidade, tal como percebida por cada projetista (SHETH, 2000). Por exemplo, dependendo da localização geográfica do projetista os nomes utilizados variam (ex.: aipim, mandioca) , e também porque ocorrem variações na forma de escrever um mesmo conceito, como é o caso das abreviaturas (ex.: curso de água, curso d'água, c. água).

A intervenção humana na resolução de conflitos é, praticamente, indispensável no processo de identificação de correspondência entre diferentes esquemas. O que se pode alcançar, no máximo, é que a ontologia sugira as melhores soluções, de acordo com cálculos de probabilidade e similaridade (CUI; JONES; O'BRIEN, 2002). Para minimizar a necessidade da interação com um especialista do domínio, duas medidas devem ser tomadas. A primeira delas diz respeito à investigação e implementação de técnicas de casamento por similaridade (*similarity matching*) (COHEN, 1998). Este casamento deve dar-se tanto em nível de nomes dos termos quanto em nível de estrutura, levando em consideração hierarquias, relacionamentos e os atributos dos termos candidatos presentes na ontologia (BERGAMASCHI et al. 1998; HOLT, 2000).

A segunda parte da solução diz respeito à atualização da ontologia, de modo "online". Cada vez que um termo é pesquisado na ontologia e não é encontrado, o especialista deve escolher, entre os candidatos apresentados, selecionados através de similaridade, qual deles é um sinônimo daquele utilizado como entrada, ou ainda classificá-lo como novo, inserindo-o em uma nova posição da ontologia. Desta forma, a ontologia é atualizada e torna-se mais rica.

3.5 Consulta por similaridade

Para a identificação de equivalência ou similaridade entre esquemas conceituais existem várias propostas de metodologias (*matchers*) (RAHN; BERNSTEIN, 2001). Cada uma destas propostas possui características particulares e se aplica melhor a um domínio de integração, bem como sua eficiência depende da categoria dos esquemas conceituais a serem integrados.

Quando se trabalha com dados armazenados de forma estruturada, consultas com predicados booleanos são convenientes, visto que os dados resultantes são relativos a um único objeto. Nestes casos, uma das abordagens pode ser de mapeamento de igualdade de valores, tais como ocorre nos sistemas Rufus (SHOENS, 1993) e Pegasus (AHMED, 1991) nos quais são comparados os valores de chaves primárias. Esta abordagem, contudo, é dependente da intervenção do usuário, pois ele deve especificar a equivalência de instância, ou seja, o usuário necessita indicar que instâncias são representações de um objeto indicado por uma chave global (DORNELES, 2002). Uma vez que o esquema global deve ser capaz de representar todos os construtores dos demais esquemas, com identificação e resolução de conflitos semânticos, o esquema global é a ontologia (MELLO; HEUSER, 2001). O mapeamento entre diferentes esquemas, com ou sem o apoio de um esquema global, recebe o nome de *schema matching* (MILO; ZOHAR, 1998).

Uma variação do mapeamento por igualdade de valores, mais flexível, que minimiza a necessidade de intervenção do usuário, consiste na utilização não apenas da chave primária para a integração, mas também da correspondência entre os atributos comuns. Um trabalho que segue esta metodologia é o desenvolvido por Chatterjee e Segev (1991). O sistema gera um valor para determinar a igualdade de tuplas de tabelas de fontes de dados diferentes. Os atributos são comparados para cada par de tuplas de fontes diferentes, e a probabilidade de duas instâncias serem iguais é dada pelo número de atributos iguais entre elas.

Uma outra abordagem, que também trabalha sobre as chaves é o mapeamento por uso de funções. Trabalhos nessa área podem ser encontrados, entre outros, em (HERNÁNDEZ; STOLFO, 1995; SACCOL; HEUSER, 2002).

Ocorre que, uma vez integrados em termos de construtores, os esquemas conceituais são todos descritos na Geographic Markup Language (GML)(OGC, 2001). Sendo baseada em XML, trata-se de uma linguagem semi-estruturada. Desta forma, nenhuma das técnicas acima se mostra satisfatória.

Em dados não-estruturados ou semi-estruturados, a área de pesquisa que trabalha com a questão da similaridade é a de recuperação de informação (*information retrieval – IR*) (BAEZA-YATES, 1999). Sistemas tradicionais de recuperação de informação são particularmente suscetíveis aos problemas gerados pela riqueza das linguagens naturais, especialmente no que diz respeito a sinônimos e homônimos. A maioria destes sistemas não é muito mais que sistemas de casamento de padrões (*pattern matching*), desconsiderando aspectos do contexto global no qual o conceito está inserido (RICHARDSON; SMEATON; MURPHY, 1994). Nesta seção serão apresentadas algumas técnicas de *similarity matching*, desde o nível mais simples de comparação de caracteres, até mais completas semanticamente.

Os *matchers* de esquema consideram somente a informação em nível de esquema, e não em termos de instâncias (dados). Esta classe de metodologia geralmente encontra vários candidatos a similar para cada elemento de entrada, atribuindo um valor no intervalo [0..1] para mensurar o grau de semelhança (BERGAMASCHI; CASTANO; VINCINI, 1999). Este tipo de *matcher* pode realizar a medida de similaridade em diferentes perspectivas, tais como granularidade, cardinalidade, lingüística e restrições (RAHN; BERNSTEIN, 2001).

Uma abordagem de integração semântica pode utilizar um dos tipos de *matcher* individualmente, ou seja, através de uma única perspectiva, ou combinando mais de uma. No segundo caso, esta combinação pode ser simplesmente pela soma dos resultados de cada *matcher* individual ou ainda através da implementação híbrida de vários *matchers* (RAHN; BERNSTEIN, 2001).

3.5.1 Similaridade entre cadeias de caracteres

Uma das técnicas comumente utilizada para cálculo de similaridade entre cadeias de caracteres, ou *strings*, é a medida da distância entre os caracteres de um par de palavras. A esta técnica dá-se o nome de *similarity matching*. Uma função de distância mapeia um par de *strings s* e *t* para um numero real. Quanto menor for este número, mais similares são as palavras (COHEN; RAVIKUMAR; FIENBERG, 2003). As distâncias freqüentemente usadas em recuperação de informação são a distância de Levenshtein e a distância de Hamming.

3.5.1.1 Distância de Levenshtein

É definida pela quantidade de caracteres que é necessário inserir, excluir ou substituir para transformar uma *string* em outra. Por exemplo, d(lagoa, laguna) = 2. Partindo de lagoa, a seqüência "lag" é mantida, "o" é substituído por "u", e "n" é inserido. O "a" final é mantido.

3.5.1.2 Distância de Hamming

É definida sobre seqüências com o mesmo número de caracteres. O valor da distância é dado pelo número de símbolos diferentes entre as duas *strings* e que estão nas mesmas posições. Por exemplo, d(lago, mato) = 2.

3.5.1.3 Métrica de Jaro

A métrica de similaridade de Jaro (JARO, 1989; WINKLER, 1999) é mais abrangente que as distâncias de Hamming e de Levenshtein, uma vez que consideram o fato de que a pessoa que escreveu as palavras pode ter cometido erros de deleção, inserção e transposição (troca na ordem) de caracteres, sem que isso seja considerado uma total disparidade entre as *strings* comparadas. Desta forma, a similaridade de Jaro é dada pela fórmula (JARO, 1989):

$$\Phi(s1,s2)=(1/3)((c/|s1|)+(c/|s2|)+((1/2)(1-T/c)))$$

onde s1 e s1 são as *strings* a serem comparadas, c é o número de caracteres iguais e nas mesmas posições entre as duas *strings* e T é o número de caracteres transpostos (WINKLER, 1999).

O software comercial LikeIt (YANILOS; KANZELBERGER, 1997) implementa um algoritmo da mesma classe que o de Jaro, ou seja, que considera variações nas palavras. Por se tratar de um software comercial, sua especificação não está disponível.

3.5.2 Similaridade semântica

As técnicas de cálculo de distância entre duas cadeias de caracteres (*similarity matching*) puras utilizam apenas aspectos sintáticos das *strings* envolvidas. Elas se aplicam para os casos de abreviações de nomes ou eventuais erros ortográficos (MONGE; ELKAN, 1996). Nenhuma semântica é considerada nestas funções. Desta forma, estas técnicas isoladas não são apropriadas para a unificação semântica necessária neste trabalho, pois elas não detectam sinônimos nem levam em consideração o contexto no qual a palavra (ou conceito) está inserida.

Tuney (2001) apresenta a proposta de utilização de um algoritmo para encontrar sinônimos de um conceito, a partir de uma lista de candidatos. Seu trabalho baseia-se não na proximidade sintática entre as palavras, mas sim na proximidade em termos da ocorrência das mesmas. No cálculo mais completo é considerado inclusive o contexto no qual as palavras estão inseridas, ou seja, as palavras vizinhas. Seu trabalho utiliza o algoritmo PMI-IR (*Pointwise Mutual Information – Information Retrieval*), e é aplicado sobre textos buscados na Internet. Esta é uma técnica estatística de cálculo de similaridade entre conceitos (palavras), na qual os mesmos não estão organizados de uma forma estruturada, ou seja, ele não está armazenado em único meio.

A seguir são apresentadas as técnicas léxicas utilizadas para realizar os cálculos de similaridade entre conceitos. Nestes casos, os conceitos estão organizados segundo uma hierarquia, construída de forma assistida.

3.5.2.1 Conceptual Similarity

Também chamado de *node-based approach* (JIANG; CONRATH, 1997) considera que os conceitos estão todos definidos em uma hierarquia, no qual os nodos da árvore são os conceitos e os arcos são os relacionamentos entre eles. O trabalho de Resnik (1998) apresenta uma abordagem na qual apenas os relacionamentos taxonômicos (IS-A) são considerados como critério de cálculo de similaridade entre dois conceitos. Os demais relacionamentos, tais como agregação, associação e composição são ignorados. Esta abordagem considera que a similaridade de dois conceitos (c1 e c2) é inversamente proporcional à probabilidade de eles ocorrerem na taxonomia. Uma vez que essa probabilidade é calculada pelo número de elementos da hierarquia que descrevem ambos os conceitos, quanto mais especializado for o conceito comum, mais similares estes dois conceitos são. O cálculo da similaridade é dado pela fórmula a seguir.

$$Sim(c1,c2) = \max_{c \in S(c1,c2)} [-\log p(c)]$$

onde S(c1,c2) é o conjunto de conceitos que englobam tanto c1 quanto c2 (mais genéricos), e p(c) é a probabilidade deste conceito ocorrer. O resultado de –log p(c) é chamado de *information content (IC)*, e representa a relevância do conceito. Quanto mais especializado, mais relevante, e menor a probabilidade de ocorrência.

O cálculo de p(c) é dado através da razão entre a freqüência com que este conceito ocorre e o número total de palavras existentes no texto que está sendo examinado frente à taxonomia. Assim, a fórmula para o cálculo da probabilidade de ocorrência do conceito é:

$$p(c) = \frac{freq(c)}{N}$$

Para calcular freq(c) devem ser considerados outros dois conjuntos de conceitos (RESNIK, 1998): *words(c)* e *classes(w)*. *Words(c)* é o conjunto de palavras pertencentes, direta ou indiretamente, à classe *c*. *Classes(w)* é definido como sendo o número de classes (conceitos) aos quais a palavra pode pertencer, no caso de ela ter seu significado variando de acordo com o contexto (homônimos). Desta maneira, o cálculo da freqüência do conceito *c* (que engloba os dois conceitos que estão sendo comparados), pode ser feito de através de duas formas, considerando ou não o caso de homônimos:

$$f(c) = \sum_{w \in words(c)} f(w) \quad \text{ou} \quad f(c) = \sum_{w \in words(c)} \frac{f(w)}{|classes(w)|}$$

onde f(w) é a freqüência com que a palavra buscada aparece no texto.

3.5.2.2 Distance similarity

Também chamado de *edge-based approach* (JIANG; CONRATH, 1997), considera que a base de conhecimento está descrita por uma taxonomia. Nesta proposta (RICHARDSON; SMEATON; MURPHY, 1994), a similaridade é estimada através da distância entre os

nodos que correspondem aos conceitos que estão sendo comparados. Esta distância é dada pelo número de arcos que os separam.

Para tornar esta abordagem mais realística em termos de distância entre os conceitos, os pesos arcos devem ser ponderados, segundo três aspectos (RICHARDSON; SMEATON; MURPHY, 1994).

- Densidade da hierarquia no nodo: Quanto mais nodos filhos e irmãos houver, mais similares eles são naquele ponto da hierarquia;
- Profundidade na hierarquia: Quanto mais se desce na árvore, menor é a distância entre os nodos, uma vez que a diferenciação é baseada em detalhes cada vez menores;
- A força dos arcos de especialização (*child links*): Para diferenciar o peso dos diferentes arcos que especializam uma mesma superclasse, a força de cada um deles deve ser considerada individualmente. Ela é estimada como uma função dependente do *information content* do nodo e da profundidade do arco na hierarquia (RICHARDSON; SMEATON; MURPHY, 1994). Na proposta desta metodologia a função para o cálculo da força de cada arco não foi especificada.

3.5.2.3 TF-IDF

A técnica de ponderação TF-IDF (*Term Frequency – Inverse Document Frequency*)(COHEN, 1998) consiste em atribuir pesos para cada um dos termos dos documentos de uma coleção. Desta forma, os termos passam a ter uma relevância maior ou menor, tanto em função do número de documentos em que aparece, quanto também pelas vezes que aparece no documento. O peso de um termo varia em cada documento, e seu valor está no intervalo [0,1], sendo 0 para termo ausente e 1 para máxima relevância.

Para cada documento é construído um vetor de pesos, e cada posição do vetor é preenchida com um dos termos a serem ponderados. A fórmula para cálculo do peso de cada termo (V_t) é dada por (COHEN, 1998)

$$V_t = (\log(TF_{v,t})+1).(\log(IDF_t))$$

onde $TF_{v,t}$ é chamada freqüência do termo (*term frequency*) e representa o número de vezes que o termo aparece no documento em questão. IDF_t é a freqüência inversa dos documentos (*Inverse Document Frequency*), e representa a razão inversa do número de documentos D_t que possuem o termo pelo número total de documentos D da coleção (D/D_t).

3.5.2.4 Técnicas combinadas

Combinando as técnicas de distância e de similaridade conceptual, é possível calcular a força de cada arco. Para tanto, considera-se que a força de um arco em uma relação pai-filho é dada pela probabilidade de uma instância de um conceito pertencente à superclasse *p* também pertencer ao conceito da subclasse c_i, segundo a fórmula (JIANG; CONRATH, 1997):

$$p(c_i/p) = \frac{P(c_i)}{P(p)}$$

O valor da força de um arco ($LS(c_i,p)$) é dado pela diferença entre os valores de *information content* dos conceitos que são seus nodos.

$$LS(c_i/p) = IC(c_i) - IC(p)$$

Os demais aspectos a serem considerados, profundidade e densidade não precisam ser calculados. Desta forma, o peso de um arco ligando dois conceitos (wt(c,p)) é dado por

$$Wt(c,p) = (\beta + (1 - \beta).\frac{E}{E(p)}).(\frac{d(p)+1}{d(p)})^{\alpha}[IC(c) - IC(p)].T(c,p)$$

onde d(p) representa a profundidade do nodo p na hierarquia, E(p) o número de nodos filhos do nodo p (densidade), e E a densidade total da hierarquia. α e β são utilizados para controlar a influência dos parâmetros de densidade e profundidade. T(c,p) representa o valor para o tipo de relação. Este valor deve ser considerado sempre unitário quando não se desejam diferenciar os relacionamentos, ou no caso de considerar-se apenas relacionamentos do tipo IS-A. A distância final entre dois conceitos é dada, finalmente, pela soma simples dos pesos de cada um dos arcos que os separam.

3.5.3 O cálculo da similaridade usando a ontologia

A similaridade dos nomes dos atributos do conceito pode ser calculada pela métrica de Jaro (JARO, 1989) ou através da distância de Levenshtein, uma vez que somente interessa saber se o nome do atributo é o mesmo. Em (LAWRENCE; GILES; BOLLACKER, 1999) uma comparação mostra que a técnica de Jaro apresenta os melhores resultados para comparação de textos longos, pois considera erros de transposição de caracteres. Contudo, visto que no caso de esquemas conceituais a quantidade de texto é pequena e, geralmente, não apresenta erros, optou-se por aplicar a técnica de Levenshtein, devido à simplicidade de implementação. Cada atributo do esquema conceitual é confrontado com cada atributo do conceito que se está buscando a similaridade. A escolha por qual atributo é o mais similar é dada pelo valor máximo de probabilidade de similaridade encontrado.

Cada um dos atributos de um conceito também pode ser ponderado, uma vez que alguns são mais significativos que outros. Para tanto, pode-se adaptar a técnica de ponderação TF-IDF (COHEN, 1998). Considera-se que a ontologia é o conjunto de documentos, e que cada conceito representa um documento. Os atributos do conceito fazem o papel dos termos. Uma vez que não é possível que uma classe (ou conceito) possua dois atributos iguais (com o mesmo nome), a parte referente à freqüência do termo no documento não precisa ser considerada, pois seu resultado é sempre igual a 1. Desta forma, a fórmula para o cálculo do peso de cada atributo de um conceito fica restrito a:

$$V_t = (\log(IDF_t))$$

Para simplificar o cálculo, limitando o peso de um atributo ao intervalo [0,1], a fórmula pode ser transformada para:

$$P_{at} = 1 - (C_a/C) \qquad (1)$$

onde P_{at} é o peso do atributo, C_a é o número de conceitos que possuem o atributo e C é o número total de conceito.

É necessário esclarecer que a soma dos pesos dos atributos e relacionamentos de um conceito não pode ultrapassar o valor 1 (DORNELES, 2002), pois este é o valor máximo de similaridade entre dois conceitos.

O cálculo para similaridade estrutural entre dois conceitos, em termos apenas de atributos, pode ser através do algoritmo do vizinho mais próximo (*nearest neighbor*) (HOLT, 2000), dado pela fórmula:

$$SimEst(Cc,Co) = (\sum_{i=1}^{n} f(Cc_i,Co_i) x Pat_i)/Nat \quad (2)$$

onde Cc e Co são os conceitos do esquema conceitual e da ontologia, respectivamente, n é o número de atributos considerados do esquema conceitual, i é o atributo individualmente (de [1..n]), f é a função de similaridade utilizada, Pat_i é o peso do atributo, previamente calculado e Nat é o número total de atributos do elemento do esquema conceitual. A função de similaridade utilizada é a de Levenshtein.

A similaridade SimNome(Cc,Co) dos nomes dos conceitos pode ser calculada utilizando-se a métrica de Levenshtein, aplicando-se a mesma metodologia utilizada para encontrar similares de nomes de atributos.

Para o cálculo ponderado da similaridade entre os relacionamentos, por questões de simplicidade e de relevância, somente os relacionamentos do tipo IS-A (generalização/especialização) são considerados na maioria dos trabalhos existentes (DOAN et al., 2002; MILO; ZOHAR, 1998; RICHARDSON; SMEATON; MURPHY, 1994). Nesta dissertação não apenas a taxonomia é considerada, com também os relacionamentos de agregação e composição.

Para hierarquias, considerando que a influência (β) da profundidade de um nodo na hierarquia é a mesma que a influência (α) da densidade deste mesmo nodo, e que somente utiliza-se os relacionamentos de hierarquia, a fórmula de JIANG e CONRATH (1997) de ponderação dos arcos fica resumida a:

$$Wt(c,p) = \frac{E}{E(p)} \cdot \frac{d(p)+1}{d(p)} \cdot (IC(c) - IC(p)) \quad (3)$$

onde o *information content* (IC) de uma classe é dado pela fórmula:

$$IC(c) = -\log((\sum(1/sup(w))).1/N) \quad (4)$$

onde sup(w) é o número de superclasses de cada uma das subclasses w, diretas ou indiretas, da classe c, e N é o número total de classes existentes. A fórmula original possuía freq(w) no numerador, ao invés de 1. Contudo, como cada classe somente pode existir uma única vez em um esquema conceitual, freq(w) foi suprimido.

O valor final da similaridade dos relacionamentos de generalização especialização de um conceito do esquema conceitual e da ontologia é calculado pela fórmula:

$$SimHier(Cc,Co) = (\sum(Hier(Cc,Pc).Wt(c,p))/ NHier(Cc,Pc)) \quad (5)$$

onde Hier(Cc,Pc) é o cada um dos relacionamentos taxonômicos do conceito do esquema conceitual presente também na ontologia, e NHier é o número total de relacionamentos taxonômicos do conceito do esquema conceitual. Se o relacionamento taxonômico na ontologia for mais especializado que a hierarquia apresentada pelo esquema conceitual, a similaridade para a taxonomia em questão é dada pela soma dos arcos que são necessários percorrer para se chegar ao nodo filho ao nodo pai do esquema conceitual.

O cálculo da similaridade em termos de relacionamentos de agregação e composição é bastante mais simples, sendo dado pela fórmula:

$$SimRel(Cc,Co) = (\sum(Rel(Cc,Co))/Rel(Cc)) \quad (6)$$

onde Rel(Cc,Co) representa um relacionamento de agregação ou composição presente na ontologia e no esquema conceitual de entrada, e Rel(Cc) representa um relacionamentos de agregação ou composição presente no esquema conceitual e ausente na ontologia.Por fim, a similaridade entre um conceito presente em um esquema conceitual frente a outro conceito presente na ontologia é dada por:

```
Sim(Cc,Co)=PN.SimNome(Cc,Co)+PA.SimEst(Cc,Co)+                    (7)
PH.SimHier(Cc,Co)+PR.SimRel(Cc,Co)
```

onde PN, PA e PH e PR são os pesos para a similaridade de nome, estrutura (atributos), hierarquia e relacionamentos, respectivamente. Os valores para os pesos podem depender do domínio da aplicação, ou seja, talvez a melhor combinação para os esquemas conceituais de um tipo de aplicação não seja a melhor se aplicada sobre os esquemas conceituais de outro tipo de aplicação.

4 O protótipo integrador

A execução dos casos de teste foi dividida em três etapas, conforme proposto na arquitetura do sistema. Primeiramente, para que se pudesse alcançar a etapa de integração semântica, o módulo de integração sintática também foi construído, mas apenas para alguns modelos de dados geográficos. A seleção dos modelos de dados deu-se por três fatores, nesta ordem de importância:

- Grau de abrangência conseguida com os modelos, ou seja, qual o poder de expressão do modelo frente à totalidade dos aspectos espaço-temporais necessários no escopo deste trabalho;
- Disponibilidade de esquemas conceituais de BDG no modelo;
- Aceitação do modelo por alguma comunidade.

De posse destes três critérios, os modelos de dados selecionados para serem integrados foram o MADS, o GeoFrame-T e o OMT-G. Cada um deles está descrito em detalhes no anexo A decorrer deste volume.

A segunda etapa consistiu na implementação do algoritmo de consulta e atualização da ontologia. No final do processo de integração semântica, para tornar possível a mineração de esquemas conceituais de BDG de aplicações reais, também foi implementado o módulo gerador de arquivos para mineração. Nesta etapa utilizou-se o Formato de Dados de Entrada (FDE) (SILVA; IOCHPE; ENGEL, 2003) como sendo o formato de dados aceito pelas ferramentas de MD. A Figura 4.1 apresenta a arquitetura de implementação do protótipo.

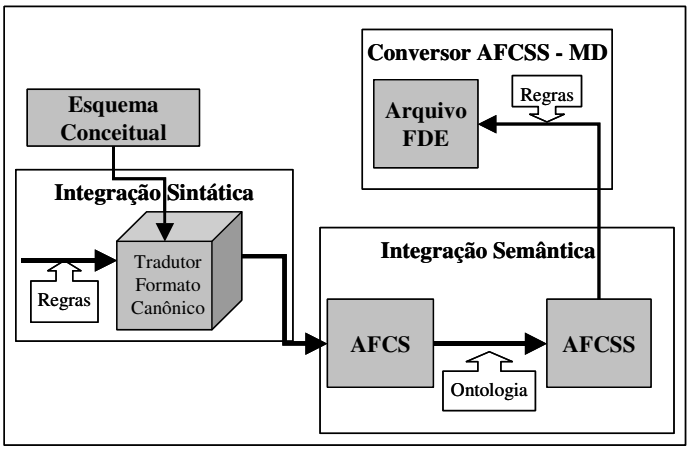


Figura 4.1 - Arquitetura do protótipo implementado

4.1 Mapeamento UML para GML

A GML foi escolhida como sendo o formato de dados comum, por se tratar de um padrão para armazenamento e intercâmbio de dados geográficos adotado pelo Open GIS Consortium, que é a organização que delibera sobre os aspectos relacionados aos sistemas de informações geográficas. Além disso, ela engloba boa parte dos construtores presentes nos demais modelos de dados geográficos e, aqueles que não possui, podem ser implementados, uma vez que trata-se de uma linguagem extensível.

Visto que tanto o GeoFrame quanto o MADS e o OMT-G são descritos a partir de construtores existentes na UML, devem ser especificadas regras de tradução deste padrão para a GML. Uma vez que a GML se baseia em XML-Schema, com extensão para encapsular aspectos geométricos e geo-espacial dos objetos, pode-se utilizar algumas regras aplicáveis à tradução de esquemas UML para XML-Schema (CARLSON, 2002), fazendo as adaptações e extensões necessárias. Estas regras são descritas no decorrer deste capítulo.

Também foi adotado como base para especificação das regras de conversão e, principalmente, para posterior construção do protótipo, os trabalhos de Xavier (2002) e Hess (2003).

4.1.1 Comparação entre os modelos

A Tabela 4.1 apresenta a comparação entre os modelos de dados geográficos escolhidos para a integração.

Tabela 4.1 - Comparação entre os modelos de dados geográficos

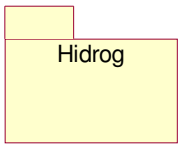
	GeoFrame	**MADS**	**OMT-G**	**GML**
Paradigma	Orientado a objetos	Orientado a objetos	Orientado a objetos	Semi-estruturado
Objetos não-espaciais	SIM	SIM	SIM	SIM
Objetos espaciais	SIM	SIM	SIM	SIM
Classes na visão de objeto (discreta)	SIM	SIM	SIM	SIM (elementos)
Classes na visão de campo (contínua)	SIM	SIM (atributo espaço-variante)	SIM	SIM
Atributos não-espaciais	SIM	SIM	SIM	SIM
Atributos espaciais	NÃO	NÃO	NÃO	SIM
Declaração de métodos nas classes	NÃO	SIM	SIM	NÃO
Temas	SIM	NÃO	NÃO	Estendido
Metadados	SIM	NÃO	NÃO	SIM
GeoMetadados	SIM	NÃO	NÃO	NÃO
Herança				
Herança simples	SIM	SIM	SIM	SIM
Herança múltipla	SIM	SIM	SIM	Indiretamente
Herança objeto	SIM	SIM	SIM	SIM

não-espacial → objeto espacial				
superclasse e subclasses com geometria diferente	SIM	SIM	SIM	SIM
Geometrias				
Ponto	SIM	SIM	SIM	SIM
Linha	SIM	SIM	SIM	SIM
Polígono	SIM	SIM	SIM	SIM
Nodo (de rede)	NÃO	NÃO	SIM	SIM
Linha direcional	NÃO	SIM	SIM	SIM
Múltipla representação espacial	SIM	NÃO	SIM	SIM
Conjunto de pontos	SIM	SIM	NÃO	SIM
Conjunto de linhas	SIM	SIM	NÃO	SIM
Conjunto de linhas direcionais	NÃO	SIM	NÃO	NÃO
Grade de pontos	SIM	SIM	NÃO	SIM
Isolinhas	SIM	NÃO	SIM	NÃO
Rede irregular de triângulos (TIN)	SIM	NÃO	SIM	NÃO
Pontos irregulares	SIM	NÃO	SIM	SIM
Polígonos adjacentes	SIM	NÃO	SIM	SIM
Grades de células	SIM	NÃO	SIM	SIM
Temporalidade*				
Instante temporal	SIM	SIM	NÃO	SIM
Intervalo temporal	SIM	SIM	NÃO	SIM
Temporalidade de objetos	SIM	SIM	NÃO	SIM
Temporalidade de atributos	SIM	SIM	NÃO	SIM
Relacionamentos temporais	SIM	SIM	NÃO	SIM
Aspectos dinâmicos	NÃO	SIM	SIM	SIM
Relacionamentos				
Associação entre objetos não-espaciais	SIM	SIM	SIM	SIM

Associações entre objeto não-espacial e objeto espacial	SIM	SIM	SIM	SIM
Associações entre objetos espaciais	SIM	SIM	SIM	SIM
Agregação	SIM	SIM	SIM	SIM
Composição	SIM	NÃO	SIM	SIM
Relacionamentos espaciais (topológicos)	NÃO	SIM	SIM	NÃO
Relacionamentos temporais	SIM	SIM	NÃO	SIM
Relacionamentos dinâmicos	NÃO	SIM	NÃO	SIM
Relacionamentos de Rede	NÃO	NÃO	SIM	SIM

4.1.2 Temas

Um tema (package) existente no modelo conceitual é mapeado em GML como sendo um elemento, cujo atributo *name* é o próprio nome do pacote, e o atributo *type* é um novo tipo de dados, definido pelo usuário. O nome deste novo tipo de dados é o nome do pacote concatenado com *type*, e é um tipo complexo, de conteúdo complexo. Ele deve ser uma extensão da classe básica *FeatureCollection*, original da GML. Todos os componentes do diagrama de classe que estiverem contidos no tema, sejam eles classes, outros temas ou associações, serão membros dessa coleção (extensões de *FeatureMember*). A Figura 4.2 ilustra a codificação GML para o construtor pacote da UML.




```
<element name="Hidrog" type="hid:HidrogType"
substitutionGroup="gml:_FeatureCollection" />
<complexType name="HidrogType">
 <complexContent>
   <extension base="gml:AbstractFeatureCollectionType"/>
 </complexContent>
</complexType>

<element name="HidrogMember" type="hid:HidrogPropertyType"
substitutionGroup="gml:FeatureMember"/>
<element name="_HidrogFeature" type="gml:AbstractFeatureType"
abstract="true"/>

<ComplexType name="HidrogPropertyType">
 <ComplexContent>
  <restriction base="gml:featurePropertyType">
   <sequence minOccurs="0">
    <element ref="hid:_HidrogFeature"/>
   </sequence>
   <attributeGroup ref="gml:AssociationAttributeGroup"/>
  </restriction>
 </ComplexContent>
</ComplexType>
```

Figura 4.2 - Exemplo de mapeamento de tema da UML para GML

Caso exista, no modelo conceitual, uma hierarquia de pacotes, ela é mapeada para a GML através do atributo *extension base.* Isto é feito na declaração do tipo do elemento do pacote, assim como para uma classe. Uma vez que um pacote pode estar contido apenas em um, e somente um, outro construtor deste tipo, a restrição de que cada elemento pode ser de extensão a apenas um outro não é um problema.

4.1.3 Classes

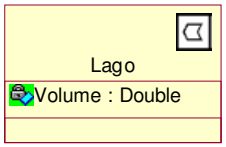
Cada classe do diagrama UML é convertida para um elemento, cujo atributo *name* é o próprio nome da classe e o tipo é definido como sendo a concatenação do nome da classe com *type*. Este é de tipo complexo (*complextype*), e de conteúdo complexo (*complexcontent*). As classes são consideradas especializações de *AbstractFeatureType* Assim, herdam os atributos *fid*, *name*, *description* e *boundedby*.

No contexto deste trabalho, na definição do elemento que representa cada classe, é especificado que o atributo *SubstitutionGroup* dela é aquele elemento que representa os componentes do pacote ao qual a classe pertence. Isto é feito para poder associar uma classe ao seu tema.

4.1.4 Atributos

Cada atributo de uma classe do diagrama UML é mapeado em GML como sendo um sub-elemento do tipo complexo definido pela classe a qual o atributo pertence. O tipo do atributo mantém-se o mesmo.

A Figura 4.3 ilustra a codificação GML para os construtores classe e atributo da UML.




```
<element name="_lago" type="gml:AbstractFeatureType"
abstract="true" substitutionGroup="hid:_HidrogFeature"/>
<element name="lago" type="hid:lagoType"
substitutionGroup="hid:_lago"/>

<complexType name="lagoType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:Polygon" />
        <element name="volume" type="number" />
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

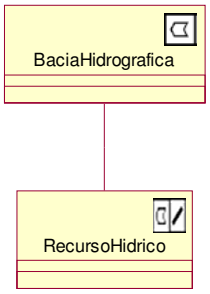
Figura 4.3 - Exemplo de mapeamento de classe e atributo da UML para GML

4.1.5 Associação

As associações simples (não espaciais e não temporais) existentes em um diagrama UML (associações binárias, composição e agregação) são mapeadas para um novo tipo complexo (*complextype*), de conteúdo complexo (*complexcontent*), com a restrição de ser uma associação de feições (*restriction base = "gml:AssociationTypeMember"*). Este novo tipo é formado por dois elementos, que são referências a cada um dos elementos

correspondentes às classes associadas. A cardinalidade da associação é definida pelos atributos *minoccurs* e *maxoccurs*, em cada uma das referências.

A Figura 4.4 ilustra a codificação GML para associações UML.



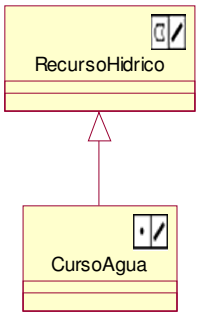

```
<element name="baciaHidrograficaId" type="hid:idPropertyType"
substitutionGroup="rc:baciaHidrografica"/>
<element name="recursoHidricoId" type="hid:idPropertyType"
substitutionGroup="rc:RecursoHidrico"/>

<complexType name="bacia_recursoType">
  <complexContent>
    <restriction base="gml:FeatureAssociationType">
      <sequence>
        <element ref="hid:baciaHidroId" />
        <element ref="hid:recursoHidricoId" minOccurs="0"
maxOccurs="unbounded"/>
      </sequence>
      <attributeGroup ref="gml:AssociationAttributeGroup" />
    </restriction>
  </complexContent>
</complexType>
```

Figura 4.4 - Exemplo de mapeamento de associação da UML para GML

Cabe salientar que os objetos referenciados não são, na realidade, os elementos que representam as classes, mas sim identificadores indicados pela concatenação do nome da classe e *Id*. Estes identificadores pertencem ao tipo *IdPropertyType*, o qual permite que o identificador seja um *ObjectIdentifier* (OID) existente no mesmo arquivo ou uma referência a um OID existente em outro arquivo de dados.

4.1.6 Herança

Uma hierarquia de classes de um diagrama UML é convertida para uma hierarquia de tipos em GML, onde cada classe do modelo conceitual é codificada como um elemento. O tipo do elemento filho é sempre uma extensão baseada no tipo do elemento pai, ou seja, as classes especializadas são declaradas como extensão da classe genérica. Assim, os atributos da classe pai passam automaticamente para as classes filhas. Como em GML cada tipo pode estender apenas um outro tipo, herança múltipla não é permitida. A Figura 4.5 ilustra a codificação de hierarquias em GML.




```
<complexType name="recursoHidricoType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <choice>
          <element ref= "gml:Polygon"/>
          <element ref= "gml:Curve"/>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<complexType name="cursoAguaType">
  <complexContent>
    <extension base="hid:recursoHidricoType">
      <sequence>
        <choice>
          <element ref="gml:Point" />
          <element ref="gml:Curve" />
        </choice>
    </extension>
  </complexContent>
</complexType>
```

Figura 4.5 - Exemplo de mapeamento de herança da UML para GML

4.1.7 Aspectos geométricos

Por se tratarem de *frameworks* para aplicações geográficas, todos os modelos de dados permitem a representação das geometrias das entidades geográficas. Isto é feito através da utilização de estereótipos nas classes espaciais do modelo conceitual. Em GML, um conjunto de propriedades, associadas às formas geométricas, pode ser utilizado. Quando houver a ocorrência de múltiplas geometrias para uma determinada classe, essa característica é mapeada em GML como uma escolha (*choice*) dentre as representações, visto que, na instanciação, somente pode haver uma geometria.

4.1.8 Campos geográficos

No GeoFrame e no OMT-G, campos geográficos são representados pela associação da classe que o representa com alguma(s) das subclasses de representação campo, que contém a geometria. Para fins de limpeza do diagrama, esta associação é representada por um pictograma, chamado estereótipo, cuja semântica é a representação do campo geográfico.

No MADS o conceito de campo geográfico inexiste. É enfocada a visão discreta do espaço, em nível de objetos geográficos. Desta forma, a uma classe geográfica que representa um fenômeno contínuo da realidade é acrescentado um atributo espaço variante. Este atributo não possui geometria definida, apenas representa um atributo da classe cujo valor é dado por uma função de cobertura.

4.1.8.1 Campos geográficos em nível de classes

A partir da especificação 3.0 da GML, é possível codificar fenômenos contínuos da realidade. Contudo, a atual versão da GML ainda é restrita em termos de geometrias possíveis. Desta forma, nem todas as subclasses de representação campo são contempladas.

Qualquer campo geográfico em GML é derivado de *Coverage*, conforme ilustra a Figura 4.6.

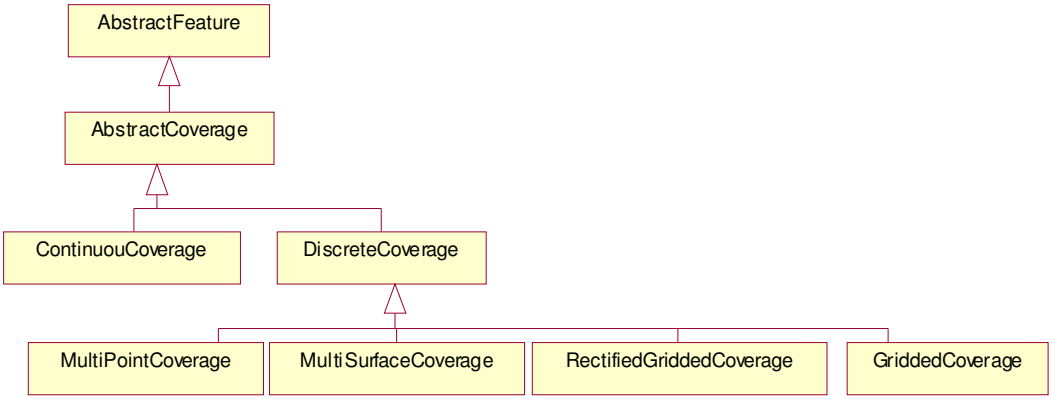
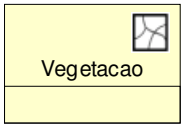


Figura 4.6 - Hierarquia de Coverage

Até o momento da escrita deste trabalho, *ContinuousCoverage* não possuía nem especificação nem subclasses.

Para codificar um campo geográfico em GML 3.0, simplesmente sua geometria é referenciada como sendo um dos sub-elementos do elemento que representa a classe, conforme apresentado Figura 4.7.




```
<element name="Vegetacao" type="VegetacaoType"/>
<complexType name="VegetacaoType">
  <complexContent>
    <sequence>
      <element ref="gml:MultiSurfaceCoverage"/>
    </sequence>
  </complexContent>
</complexType>
```

Figura 4.7 - Exemplo de mapeamento de classe de campo geofráfico da UML para GML

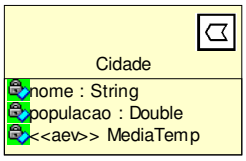
4.1.8.2 Campos geográficos em nível de atributos

Campos geográficos em nível de atributos existem apenas no MADS, sendo chamado de atributo espaço-variante (*space varying attribute* – SVA). Ele é apresentado como sendo um pictograma, sem geometria definida.

Para codificar em GML 3.0 um SVA, assume-se que sua geometria é do tipo *MultiPointCoverageType*, por ser o tipo mais simples derivado de *coverage* e, adicionalmente, representar múltiplos pontos de amostragem. A codificação da classe é igual a de uma classe sem este tipo de atributo, e o elemento que o representa é uma referência a um elemento de um novo tipo, no qual é definida esta "função de cobertura".

Uma vez que os tipos derivados de *Coverage* possuem um sub-elemento *rangeSetType*, cuja uma das possibilidades é o tipo abstrato "gml:_value", é necessário especificar um elemento que o substitua, para os casos em que os valores são fornecidos "manualmente".

Este elemento tem o mesmo nome que a classe, com a palavra “val” concatenada na frente. A Figura 4.8 ilustra o mapeamento descrito.




```
<element     name="_cidade"     type="gml:AbstractFeatureType"
abstract="true"/>
<element  name="cidade"  type="cidadeType"  substitutionGroup=
"_cidade">
<complexType name="cidadeType">
  <complexContent>
    <extension base="gml:AbstractFeatureType>
      <sequence>
        <element ref="gml:Polygon"/>
        <element ref="MediaTemp"/>
        <element name="nome" type="string"/>
        <element name= "populacao" type="number"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="MediaTemp" type="MediaTempType"/>
<complexType name="MediaTempType">
  <complexContent>
    <extension base="gml:MultiPointCoverageType"/>
  </complexContent>
</complexType>

<element    name=    "valMediaTemp"    type="valMediaTempType"
substitutionGroup="gml:_Value"/>
<complexType name="valMediaTempType">
  <simpleContent>
    <restriction base="gml:quantityType">
      <attribute name="val" type="anyURI" use="optional"/>
    </restriction>
  </simpleContent>
</complexType>
```

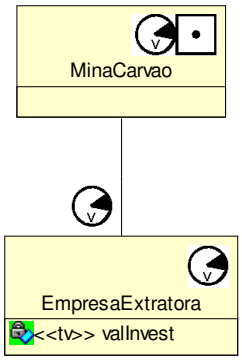
Figura 4.8 - Exemplo de mapeamento de atributo espaço-variante da UML para GML

4.1.9 Aspectos temporais

Os diferentes modelos de dados estudados definem a temporalidade em três níveis: classes, atributos e relacionamentos. A GML 3.0 introduz o conceito de temporalidade, sendo possível estabelecê-la nos três níveis necessários.

4.1.9.1 Temporalidade em classes

Tanto no MADS quanto no GeoFrame-T a temporalidade em classes é dada por um estereótipo temporal. Se for um instante temporal, é criado um novo elemento para o tipo complexo da classe, sendo uma referência para “gml:TimeInstant”. Se for um intervalo temporal, é criado um novo elemento para o tipo complexo da classe, sendo uma referência a “gml:TimePeriod”, conforme apresenta a Figura 4.9.




```
<element name="_MinaCarvao" type="gml:AbstractFeatureType"
abstract="true"/>
<element name="MinaCarvao" type="MinaCarvaoType"
substitutionGroup="_MinaCarvao"/>

<complexType name="MinaCarvaoType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:Point"/>
        <element ref="gml:TimePeriod"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

Figura 4.9 - Exemplo de mapeamento de temporalidade da UML para GML

4.1.9.2 Temporalidade em atributos

Tanto no MADS quanto no GeoFrame-T a temporalidade em atributos é dada por um estereótipo temporal.

O atributo temporal é criado como sendo um grupo, cujo nome é formado pela palavra "group" concatenada com o nome do atributo. Ele é composto de dois outros elementos. Um deles é o próprio atributo da classe, com o tipo nela definido e o outro é uma referência a "gml:TimeInstantType" para o caso de instante temporal ou "gml:TimePeriodType" para o caso de intervalo temporal.

```
<element name="_EmpresaExtratora"
type="gml:AbstractFeatureType" abstract="true"/>
<element name="EmpresaExtratora" type="EmpresaExtratoraType"
substitutionGroup="_ExpresaExtratora"/>

<complexType name="EmpresaExtratoraType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:TimePeriod"/>
        <group ref="groupvalInvest"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<group name="groupvalInvest">
  <sequence>
    <element name="valInvest" type="number"/>
    <element ref="gml:TimePeriod"/>
  </sequence>
</group>
```

4.1.9.3 Temporalidade em associações

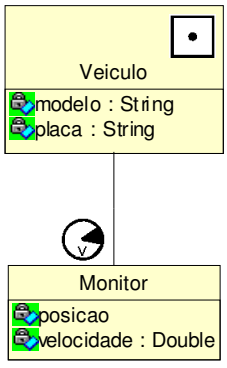
Ao tipo complexo que representa a associação é adicionado um novo elemento, que é uma referência ao tipo temporal da associação, ou seja, instante temporal ou intervalo temporal.

```
<element ref="gml:TimeInstant"/> ou
<element ref="gml:TimePeriod"/>
```

4.1.10 Aspectos dinâmicos

O GeoFrame-T não possui o conceito de modelagem de aspectos dinâmicos. O MADS e o OMT-G possuem esta funcionalidade.

Qualquer evento dinâmico a ser modelado sobre uma classe é codificado em duas etapas. Na primeira é criado um tipo da mesma forma que para classes estáticas. Na segunda fase é criado um novo tipo, com o mesmo nome daquele criado na primeira etapa, acrescido da palavra "Dynamic" no início. Este novo tipo é uma extensão do primeiro. Adicionalmente, este tipo de dado tem uma referência ao grupo "gml:dynamicProperties". A Figura 4.10 apresenta o mapeamento de aspectos dinâmicos da UML para a GML.




```
<element name="_veiculo" type="gml:AbstractFeatureType"
abstract="true"/>
<element name="veiculo" type="veiculoType" substitutionGroup=
"_veiculo">
<complexType name="veiculoType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:Point"/>
        <element name="modelo" type="string"/>
        <element name= "placa" type="number"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>

<element name="DynamicVeiculo" type="DynamicVeiculoType"
substitutionGroup="veiculo"/>
<complexType name="DynamicVeiculoType">
  <complexContent>
    <extension base="gml:VeiculoType">
      <group ref="gml:dynamicProperties"/>
    </extension>
  </complexContent>
</complexType>
```

Figura 4.10 - Exemplo de mapeamento de aspectos dinâmicos da UML para GML

A classe que dá a dinamicidade da classe espaço-temporal é convertida para um elemento de substituição a "gml:_TimeSlice". Seu tipo estende *AbstractTimeSliceType*, adicionando os atributos próprios. Desta forma, em cada instante de tempo são armazenados os atributos desejados.

```
<element name="monitor" type="monitorType substitutionGroup=
"gml:_TimeSlice">
<complexType name="monitorType">
  <complexContent>
    <extension base="gml:AbstractTimeSliceType">
      <sequence>
        <element name="posicao" type="gml:location"/>
        <element name="velocidade" type="number"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

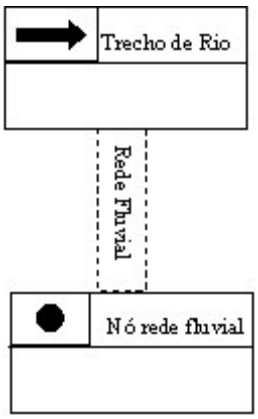
Por último, é criado um tipo que substitui "gml:history", e que referencia o tipo definido para a dinamicidade. Como um dos elementos de d*ynamicProperties* é *history*, é desta forma que se associa o elemento dinâmico ao elemento que registra os estados pelos quais ele passou.

```
<element    name="monitorVeiculo"    type="monitorVeiculoType
substitutionGroup= "gml:history">
<complexType name="monitorVeiculoType">
  <complexContent>
    <restriction base="gml:HistoryPropertyType">
      <sequence>
        <element ref="monitor" maxOcuurs="unbounded">
      </sequence>
    </restriction>
  </complexContent>
</complexType>
```

4.1.11 Aspectos de redes

Nem o MADS nem o GeoFrame modelam redes, ou seja, esquemas geográficos formados por objetos geográficos que são nós (nodos) e objetos geográficos que são arcos, unindo os nós. O modelo OMT-G possui esta expressividade, sendo este o motivo da escolha dele para completar os aspectos geográficos a serem convertidos para GML. Desta forma, todas as potencialidades dos modelos estudados são contempladas.

Na conversão de uma topologia de redes, tanto arcos quanto nodos são convertidos para tipos que são extensões de "gml:AbstractFeatureType". A única coisa que caracteriza a topologia de redes é que arcos possuem uma referência à geometria "gml:directedEdge" e nodos à geometria "gml:directedNode". Os demais atributos são convertidos da mesma forma que para as classes que representam objetos geográficos. Um exemplo de conversão é apresentado na Figura 4.11.




```
<element name="_TrechodeRio" type="gml:AbstractFeatureType"
abstract="true"/>
<element name="TrechodeRio" type="TrechideRioType"
substitutionGroup= "_TrechodeRio">
<complexType name="TrechodeRioType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:directedEdge"/>
      </sequence>
  </complexContent>
</complexType>

<element name="_NoRedeFluvial" type="gml:AbstractFeatureType"
abstract="true"/>
<element name="NoRedeFluvial" type="NoRedeFluvialType"
substitutionGroup= "_NoRedeFluvial">
<complexType name="NoRedeFluvialType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:directedNode"/>
      </sequence>
  </complexContent>
</complexType>
```

Figura 4.11 – Exemplo de conversão de topologia de redes da UML para GML

4.1.12 Aspectos topológicos (relacionamentos espaciais)

Na atual especificação da GML (3.0), aspectos de relacionamentos espaciais, tais como interseção, adjacência e continência ainda não são suportados. Estas funcionalidades estão previstas para serem contempladas a partir da versão 4 da linguagem.

4.2 A implementação do módulo sintático

Em (HESS; IOCHPE, 2003) foi apresentada uma metodologia para conversão de esquemas conceituais baseados no *framework* conceitual GeoFrame para a GML. Naquele trabalho um conjunto de regras foi definido para cada um dos construtores presentes no GeoFrame. Na seção anterior foi apresentada a extensão da integração sintática, englobando mais aspectos do GeoFrame, tais como visão de campo e temporalidade, bem como foram criadas regras para os construtores dos demais modelos de dados utilizados (MADS e OMT-G) que não têm equivalentes no GeoFrame.

Para cada um destes modelos de dados foi implementado um programa conversor de esquemas para GML. No caso do GeoFrame, o programa, chamado RoseGIS (HESS; IOCHPE, SILVA, 2003), foi desenvolvido dentro do próprio ambiente da ferramenta CASE Rational Rose, conforme ilustra a Figura 4.12. Utilizando programação em linguagem Visual Basic for Applications, o conversor GeoFrame para GML 3.0 foi executado sobre vários esquemas conceituais, com as mais variadas quantidades de classes, atributos e relacionamentos. Em todas elas a conversão foi correta. Apenas alguns cuidados

devem ser tomados durante a modelagem, tais como evitar acentos nos nomes de classes e atributos e nomes com mais de uma palavra separados por espaço.

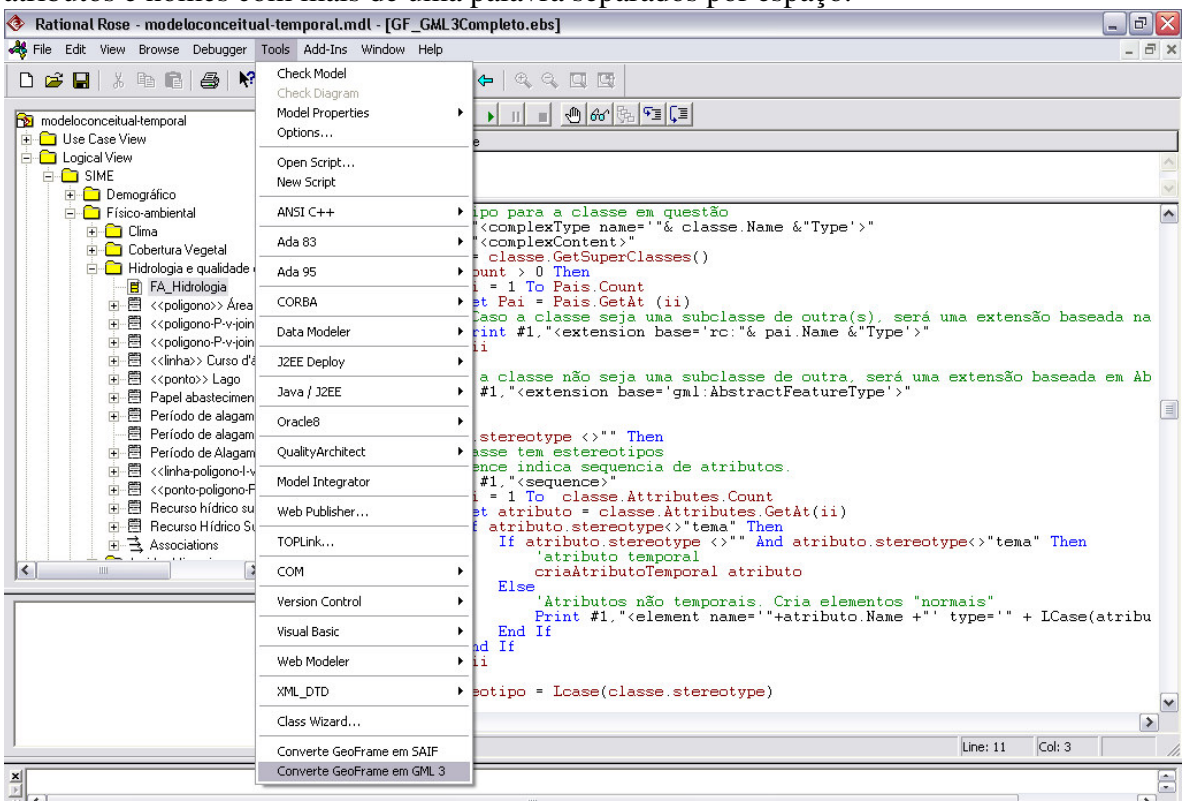


Figura 4.12 - Programa de conversão GeoFrame - GML v3

A implementação das regras de conversão do MADS para GML 3 foi feita em Java, tendo como entrada não o diagrama de classes do esquema conceitual, mas sim um documento XML gerado pela ferramenta MADSEditor (PARENT at. al, 1998). Novamente o programa foi testado com esquemas conceituais de aplicações reais, obtidas junto aos criadores do modelo e da ferramenta e os resultados alcançados foram corretos. A Figura 4.13 mostra a tela de execução da ferramenta de conversão do MADS para GML 3.

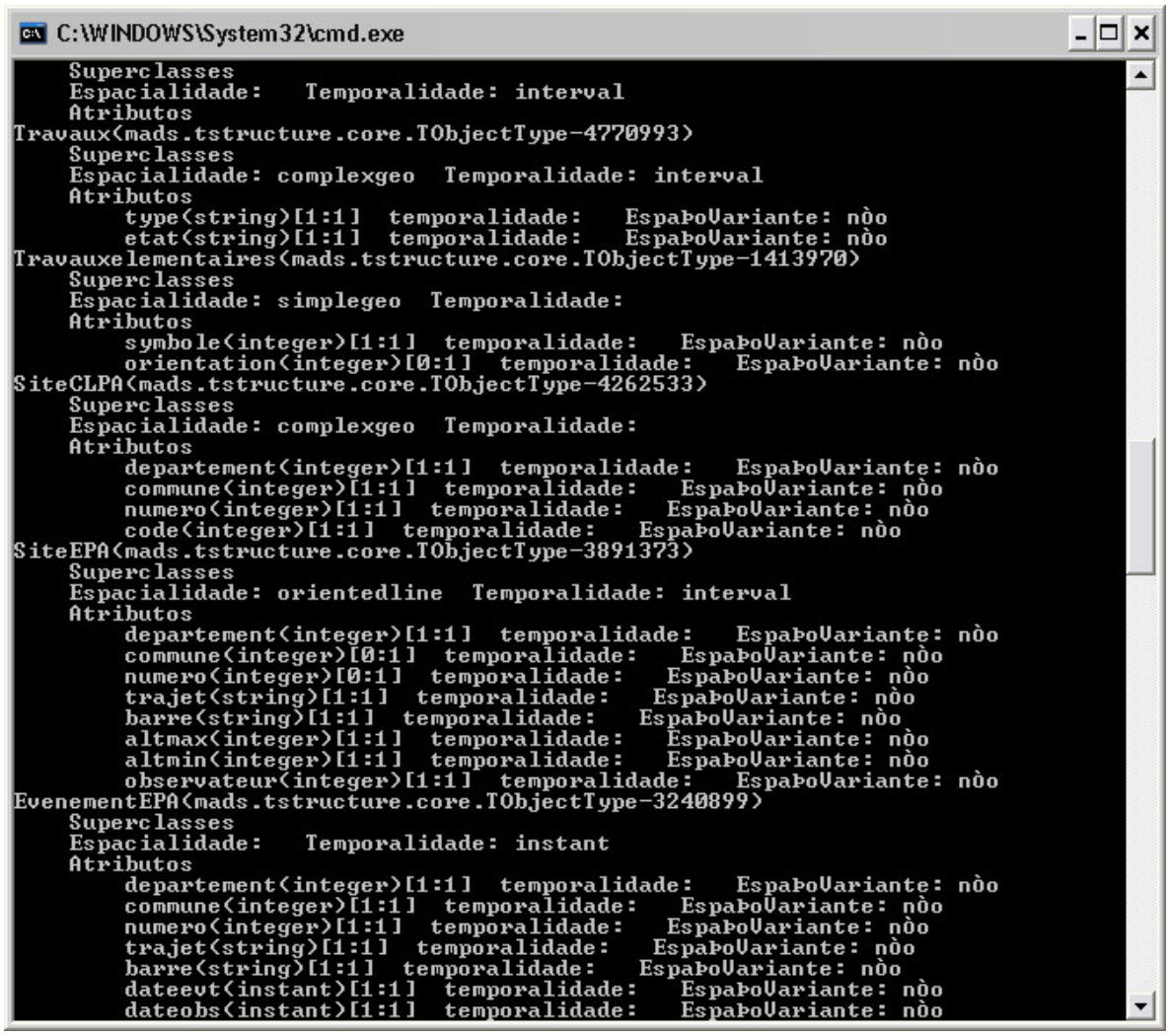


Figura 4.13 - Log da conversão MADS – GML v3

Um aspecto interessante de se ressaltar é que foi testada a conversão de dois esquemas idênticos, um modelado com base no GeoFrame e um modelado baseado no MADS. O resultado foi extremamente satisfatório, pois o código gerado para os dois modelos foi equivalente. Somente não foi igual em função da ordem com que os elementos dos esquemas conceituais são tratados em cada um dos modelos de dados.

4.3 A execução do algoritmo da ontologia

Para a execução do algoritmo de pesquisa e possível atualização da ontologia foi utilizado apenas um subconjunto dos conceitos da ontoGeo. O primeiro subconjunto escolhido foi o dos conceitos de hidrologia. A escolha deu-se em função da disponibilidade deste tema nos esquemas conceituais de teste adquiridos. A Figura 4.14 apresenta os conceitos da ontologia para este domínio, bem como atributos e relacionamentos hierárquicos. Embora os relacionamentos de agregação e composição sejam considerados no algoritmo, eles não estão visíveis para deixar o diagrama mais limpo, visto que se eles fossem mostrados, todos os demais relacionamentos também seriam.

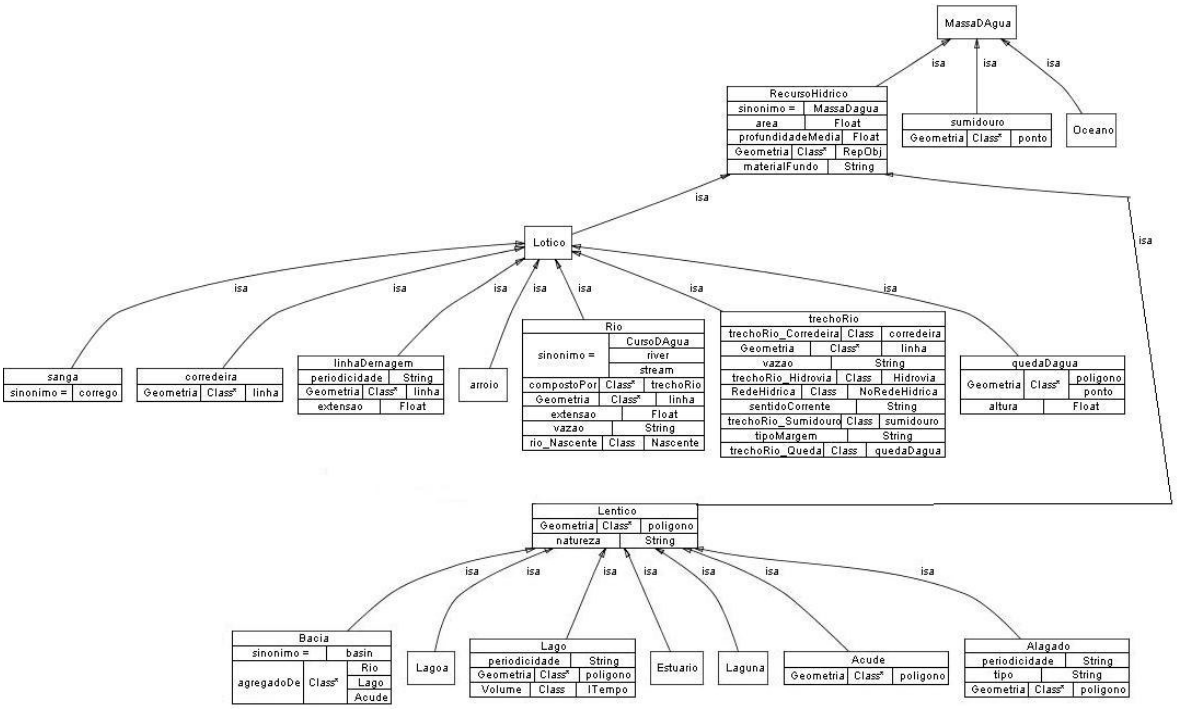


Figura 4.14 - A parte de hidrografia da ontoGeo

É importante esclarecer que os esquemas conceituais de entrada do algoritmo não são os mesmos que foram utilizados para construir a ontologia. São esquemas de fontes diversas, mas todos modelados para aplicações reais.

Este experimento foi executado utilizando-se três diferentes tipos de esquemas conceituais. O primeiro tipo representa os esquemas que modelam apenas classes e relacionamentos taxonômicos. Esquemas com classes, atributos para as classes e relacionamentos de generalização/especialização são considerados como pertencentes ao segundo tipo. O último tipo representa os esquemas que podem ter classes com atributos e classes sem atributos, além de hierarquias. Neste último caso, adicionalmente, a ontologia já fora atualizada.

Como demonstrado pela Tabela 4.2, para cada esquema o algoritmo de casamento por similaridade foi executado duas vezes, cada uma com um conjunto de valores diferentes para os parâmetros PN, PA, PH e PR. Na seqüência, cada um destes conjuntos é chamado de cenário.

Para os três esquemas de BDG processados, no primeiro cenário foi atribuído o valor 0,25 para todos os pesos. No segundo cenários PN, PA, PH e PR assumiram diferentes valores, dependendo das características do esquema conceitual de entrada.

Tabela 4.2 – Esquemas conceituais x pesos dos parâmetros

Esquema	Cenário	PN	PA	PH	PR
Classes e hierarquias	1	0,25	0,25	0,25	0,25
Classes e hierarquias	2	0,70	0,00	0,30	0,00
Classes, atributos e hierarquias	1	0,25	0,25	0,25	0,25
Classes, atributos e hierarquias	2	0,45	0,35	0,20	0,00
Classes, atributos e hierarquias. Ontologia atualizada	1	0,25	0,25	0,25	0,25
Classes, atributos e hierarquias. Ontologia atualizada	2	0,50	0,25	0,25	0,00

Neste primeiro estudo de caso apresentado aqui, o objetivo principal foi determinar um cenário no qual o peso para os parâmetros dos diferentes tipos de fórmulas para o cálculo de similaridade expressasse a sua importância relativa dentro da fórmula global de cálculo de similaridade. Um próximo passo consiste na investigação do melhor conjunto de valores para o conjunto de pesos em função tanto do algoritmo quanto da ontologia.

Inicialmente o valor para a probabilidade de aceitação foi fixado em 0,75 (75%), enquanto a probabilidade de análise foi estipulada em 0,40 (40%) e ao parâmetro *delta* foi dado o valor de 0,10 (10%).

4.3.1.1 Primeiro caso: esquema conceitual somente com classes

O primeiro experimento consistiu em comparar os conceitos da ontologia e do esquema conceitual de entrada somente em termos de nomenclatura. Apenas relacionamentos hierárquicos foram considerados. Atributos também inexistiam no esquema conceitual.

Na primeira rodada do experimento, os pesos PN, PA, PR e PH foram todos considerados iguais, com o valor 0,25. Como resultado, nenhum dos elementos do esquema conceitual de entrada foi casado automaticamente com algum conceito da ontologia, mesmo tendo nomes iguais. A explicação para isso é dada pelo fato que SimRel e SimAt eram sempre zero (0), restringindo o valor máximo de similaridade possível em 0,5 (50%), se ambos nomes e taxonomias fossem iguais.

Em uma segunda execução do algoritmo os valores de PA e PR foram zerados e o de WR baixado para 0,2. Adicionalmente, PN foi aumentado para 0,8. Como resultado, o casamento correto dos elementos do esquema conceitual com seus respectivos pares na ontologia atingiu a taxa de 100% (5 de 5) para similaridades maiores de 0,75 (75%). Para conceitos com similaridade calculada entre 0,4 e 0,75 o algoritmo identificou candidatos para seis elementos do esquema conceitual. Em dois destes casos (33%), o candidato com maior valor calculado para similaridade era o par correto. Para os outros quatro elementos (66%) nenhum dos candidatos apresentados era o par correto, mas em toda a ontologia não havia um conceito equivalente. Não ocorreram casos em havia conceitos na ontoGeo equivalentes a elementos do esquema conceitual e o algoritmo não os identificou.

4.3.1.2 Segundo caso: esquema conceitual com classes e atributos

No segundo experimento foi utilizado um esquema conceitual mais completo que o utilizado no primeiro experimento. O esquema conceitual de entrada possuía classes, atributos para algumas destas e relacionamentos taxonômicos. Associações de agregação e composição inexistiam.

A primeira rodada deste experimento foi feita com PN=PA=PH=PR=0,25. Os resultados obtidos não foram totalmente corretos. Um elemento seria casado automaticamente com um conceito da ontologia se seus nomes, atributos e taxonomias fossem iguais, pois o cálculo da similaridade resultaria em 0,75 (75%). Neste ponto o algoritmo cometeu um erro. Apesar do elemento do esquema conceitual “baia” ter uma similaridade de nome de 90% com o conceito da ontologia “bacia” e especializarem a mesma superclasse, eles não representam o mesmo fenômeno do mundo real. Para os elementos com candidatos entre os limites máximo e mínimo de similaridade a taxa de sucesso na indicação de candidatos a par dos elementos do esquema conceitual foi de 100%. Nenhum casamento foi perdido.

Na segunda rodada da execução do algoritmo o valor de PR foi colocado em zero (0), em virtude do esquema conceitual não apresentar, em sua modelagem, relacionamentos de composição ou agregação. PN foi aumentado para 0,45 e PA para 0,35. PH foi reduzido para 0,2. Como resultado, apenas um elemento foi automaticamente casado ao seu par correspondente na ontoGeo. E este casamento estava correto. O erro ocorrido na primeira execução do algoritmo não se repetiu, e novamente nenhuma identificação de possíveis pares equivalentes foi perdida. A taxa de sucesso na identificação de candidatos foi de 100%. Para dois dos elementos do esquema conceitual não havia conceitos equivalentes na ontologia. Desta forma, para estes elementos foram criados conceitos novos na ontoGeo. Experimentos com outros valores para os pesos PA, PN, PR e PH foram feitos e apresentaram resultados bastante similares.

4.3.1.3 Terceiro caso: esquema conceitual com classes e atributos, e com a ontologia atualizada

Os primeiros dois experimentos foram feitos com a versão original da ontologia ontoGeo. Após o processamento dos dois esquemas conceituais dos experimentos anteriores, o algoritmo atualizou a ontologia. Foram criados novos conceitos, alguns relacionamentos taxonômicos foram modificados e alguns conceitos tiveram atributos adicionados. Como resultado, ontoGeo cresceu para 170 conceitos definidos (contra os 159 anteriores) e 106 *slots*.

Assim como nos experimentos anteriores, o algoritmo foi rodado duas vezes com os mesmos valores utilizados para ponderar PN, PA, PH e PR. Neste experimento os resultados obtidos nas duas rodadas foram praticamente os mesmos, especialmente, pois a maioria dos elementos do esquema conceitual já possuía um conceito equivalente na ontologia. Novamente, o algoritmo produziu resultados corretos com apenas uma exceção, a qual ocorreu quando um elemento do esquema conceitual foi automaticamente casado com um conceito de forma incorreta. Neste caso o elemento do esquema conceitual não possuía nem atributos nem hierarquia definida, o que fez com que somente a similaridade de nome diferenciasse-o dos conceitos da ontologia.

4.4 O formato de dados para mineração

Em Silva, Iochpe e Engel (2003) foi proposto um formato de dados para ser usado como entrada no processo de mineração de esquemas conceituais de BDG, com o intuito de reconhecer candidatos a padrões de análise. Este formato de dados considera que esquemas conceituais são definidos como transações no arquivo de entrada.

Cada esquema deve, na medida do possível, ser decomposto em sub-esquemas, pois quanto menor o número de elementos que formam o sub-esquema, maior a chance desta estrutura se repetir em esquemas de aplicações distintas (LISBOA; IOCHPE, 1999). Os elementos contemplados neste trabalho são alguns dos construtores da UML presentes no UML-GeoFrame: Pacote, classe, atributo e associação binária simples.

Foram definidos dois grupos de construtores, os fortes e os fracos. Os fortes, formados por classe e pacote, são aqueles que têm significado próprio, ou seja, podem aparecer sozinhos ou associados a algum outro construtor. Já os elementos pertencentes ao grupo dos fracos, que são atributo e associação binária simples, não têm sentido soltos. Seu

significado é dado por seu contexto, ou seja, somente aparecem relacionados ao(s) construtor(es) forte(s) ao(s) qual(is) pertence.

A seguir são apresentados os sub-esquemas reconhecidos pelo formato de dados proposto (SILVA; IOCHPE; ENGEL, 2003):

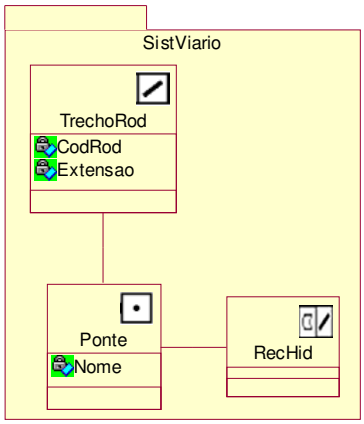
- Pacote: Apresenta apenas uma instância do construtor pacote. Cada pacote do modelo conceitual origina um sub-esquema deste tipo.
- Classe: Apresenta apenas uma instância do construtor classe. Cada classe do modelo conceitual origina um sub-esquema deste tipo.
- Atributo-Classe: Apresenta apenas uma instância do construtor atributo, associado à classe a qual pertence. Apenas o nome do atributo é armazenado, seu tipo é ignorado. A representação espacial da classe também é tratada como sendo um atributo. Cada atributo de cada classe presente no modelo conceitual origina um sub-esquema deste tipo.
- Classe-Pacote: Apresenta apenas uma instância do construtor classe, associada ao pacote ao qual pertence. Cada classe do modelo conceitual que está contida num pacote origina um sub-esquema deste tipo.
- Associação: Representa as associações binárias simples, sendo que cada instância deste construtor, juntamente com as classes que ele relaciona no modelo conceitual, gera um sub-esquema deste tipo.
- Atributo-Classe-Pacote: Análogo ao sub-esquema Atributo-Classe, mas também apresenta o pacote ao qual a classe, da qual o atributo é parte, pertence.

O arquivo de entrada para mineração, contendo os sub-esquemas, deve ser plano (*flat file*). Ele é formado por vários esquemas, cada um correspondendo a uma transação. Cada transação é composta por uma marca de esquema, e por um ou mais itens de transação, os quais são os sub-esquemas gerados a partir da decomposição do esquema.

Com o objetivo de garantir a semântica de cada elemento, e de possibilitar que ao final da mineração os tipos de cada elemento sejam identificados, os dados de entrada são descritos segundo notação própria:

```
MARCAESQ = "E*";
SUBESQ = Pacote | Classe | Atributo-Classe | Associação |
   Classe-Pacote | Atributo-Classe-Pacote;
Pacote = <nome_pacote>,":";
Classe = <nome_classe>;
Atributo-Classe = ": ",<nome_classe>,": "<nome_atributo>;
Associação = <nome_classe>,"#",<nome_classe>,"#",<nome_associação>;
Classe-Pacote = <nome_pacote>,": "<nome_classe>;
Atributo-Classe-Pacote = <nome_pacote>": ",<nome_classe>":"
 <nome_atributo>;
```

A Figura 4.15 apresenta um diagrama de classes UML-GeoFrame e a correspondente representação do mesmo no formato de dados aqui apresentado




```
0001      *E
0001      SistViario:
0001      TrechoRod
0001      Ponte
0001      RecHid
0001      TrechoRod: Linha
0001      TrechoRod: CodRod
0001      TrechoRod: Extensao
0001      Ponte: Ponto
0001      Ponte: Nome
0001      RecHid: Poligono
0001      RecHid: Linha
0001      SistViario: TrechoRod
0001      SistViario: TrechoRod: Linha
0001      SistViario: TrechoRod: CodRod
0001      SistViario: TrechoRod: Extensao
0001      SistViario: Ponte
0001      SistViario: Ponte: Ponto
0001      SistViario: Ponte: Nome
0001      SistViario: RecHid
0001      SistViario: RecHid: Poligono
0001      SistViario: RecHid: Linha
0001      TrechoRod#Ponte#assoc
0001      RecHid#Ponte#assoc
```

Figura 4.15 - Exemplo de arquivo de entrada para mineração a partir de um diagrama UML-GeoFrame

4.4.1 Regras de conversão GML para o formato de dados de entrada suportado pelas ferramentas existentes

Nem todos os elementos presentes na GML são suportados pelo formato de dados descrito acima. Apenas os elementos tema, classe, atributos, associações binárias e aspectos geográficos são contemplados. A herança, aspecto muito comum em modelagem de SIG, é desconsiderada neste mapeamento. As regras de conversão GML para este formato de dados de entrada são descritas a seguir.

4.4.2 Temas

Cada tema (pacote) existente na GML está codificado como sendo um elemento de substituição a *FeatureCollection*, direta ou indiretamente. Se o pacote for raiz de sua hierarquia, será de substituição direta a *FeatureCollection*. No caso deste tipo de consrutor estar contido em outro tema, a substituição será indireta, ou seja, o atributo *SubstitutionGroup* apontará para o pacote que o engloba. Assim, sempre que um elemento tiver o atributo *SubstitutionGroup = "gml:_FeatureCollection"* ou a algum elemento que o possua, ele deve ser mapeado como sendo um tema no formato de dados de entrada.

O exemplo a seguir ilustra a conversão do tema da Figura 4.2, da GML para o formato de dados de entrada para mineração.

```
<element          name="Hidrog"          type="hid:HidrogType"
substitutionGroup="gml:_FeatureCollection" />
 <complexType name="HidrogType">
  <complexContent>
   <extension base="gml:AbstractFeatureCollectionType"/>
</complexContent>
</complexType>
```

```
0001      Hidrog:
```

4.4.3 Classes

Cada classe existente na GML está codificada como sendo um elemento de substituição a *Feature*. Contudo, para poder relacioná-lo ao tema ao qual pertence, o elemento que representa a classe não substitui diretamente a *Feature,* e sim ao elemento que representa, genericamente, os membros do tema. Assim, sempre que o elemento possuir o atributo *SubstitutionGroup = "NometemaFeature"*, ele deve ser mapeado como sendo uma classe no formato de dados de entrada para mineração. Adicionalmente, também é criada uma entrada neste arquivo contendo *tema*: *classe*.

4.4.4 Atributos

Cada atributo está, na GML, obrigatoriamente relacionado a uma classe. Ele é definido como sendo um elemento, e sua declaração é feita internamente à classe a qual pertence, ou seja, dentro dos limites da definição do tipo da classe <complexType></complexType>. Deste modo, sempre que for encontrada a ocorrência de um *element* dentro de um *ComplexType,* mais especificamente delimitado pelas *tags* <extension></extension> e não for uma referência a um OID, este elemento deve ser mapeado como sendo um atributo no formato de dados de entrada para mineração, associado à classe a qual pertence.

O exemplo a seguir ilustra a conversão da classe da Figura 4.3, da GML para o formato de dados de entrada para mineração.

```
<element   name="_lago"   type="gml:AbstractFeatureType"
abstract="true" substitutionGroup="hid:_HidrogFeature"/>
<element      name="lago"      type="hid:lagoType"
substitutionGroup="hid:_lago"/>

<complexType name="lagoType">
  <complexContent>
    <extension base="gml:AbstractFeatureType">
      <sequence>
        <element ref="gml:extentOf" />
      </sequence>
      <element name="volume" type="number" />
    </extension>
  </complexContent>
</complexType>
```

```
0001 lago
0001 lago: polig
0001 lago: volum
0001 Hidrog: lago
0001 Hidrog: lago: polig
0001 Hidrog: lago: volum
```

4.4.5 Associações binárias

Associações binárias são os únicos tipos de relacionamentos suportados pelo formato de dados de entrada aqui utilizado. No arquivo GML, cada associação está identificada como sendo um tipo complexo com a restrição de ser uma *AssociationTypeMember*. Assim, sempre que um elemento possuir um tipo complexo com *restriction base = "AssociationTypeMember"*, ele será mapeado como uma associação entre as classes especificadas pelos OIDs referenciados dentro dos limites <sequence></sequence>. É importante observar que a declaração dos membros de um tema também usa *AssociationTypeMember*, então o filtro é feito pelo texto *Member*, que, se presente, é considerado uma membro de tema, e não uma associação. A cardinalidade é ignorada.

O exemplo a seguir, baseado na Figura 4.4 ilustra o mapeamento de associações da GML para o formato de dados de entrada.

```
<element                name="baciaHidroId"
type="hid:idPropertyType"
```

```
0001 baciaHidrog#recursoHidrico#assoc
```

```
substitutionGroup="gml:_Feature"/>
<element            name="recursoHidricoId"
type="hid:idPropertyType"
substitutionGroup="gml:_Feature"/>

<complexType name="bacia_recursoType">
 <complexContent>
  <restriction
base="gml:FeatureAssociationType">
   <sequence>
    <element ref="hid:baciaHidroId" />
    <element
ref="hid:recursoHidricoId"
minOccurs="0" maxOccurs="unbounded"/>
   </sequence>
   <attributeGroup
ref="gml:AssociationAttributeGroup" />
  </restriction>
 </complexContent>
</complexType>
```

4.4.6 Aspectos geométricos

Os aspectos geométricos presentes na GML estão codificados como elementos, representando um atributo da classe. No mapeamento para o formato de dados de entrada para mineração esta estrutura é mantida, ou seja, a representação geométrica é tratada simplesmente como mais um atributo da classe.

Contudo, como os nomes não são iguais para representar uma mesma geometria, a Tabela 4.3 apresenta a geometria GML e a correspondente geometria no formato de dados de entrada.

Tabela 4.3 - Correspondência entre as geometrias da GML e do formato de dados de entrada

GML	**Formato de dados de entrada**
Point	Ponto
Curve	Linha
LineString	Linha
OrientableCurve	Linha
CompositeCurve	Linha
Solid	Polígono
CompositeSolid	Polígono
Polygon	Polígono

5 Conclusões

Em banco de dados geográficos (BDG) a modelagem conceitual é um aspecto chave para possibilitar o reuso de construções, uma vez que a realidade geográfica é bastante complexa e, sobretudo, parte dela é representada de forma recorrente na maioria dos projetos de BDG. Um modelo conceitual bem definido fornece uma representação canônica da realidade geográfica, possibilitando a reutilização de sub-esquemas.

A utilização de padrões de análise pode contribuir para a melhoria da modelagem conceitual de BDG, uma vez que padrões são soluções previamente testadas e aprovadas (FOWLER, 1997). Isto pode reduzir o tempo necessário para o projeto conceitual e também reduzir a possibilidade de erros. Para obter os sub-esquemas de forma (semi-) automática, o processo de descoberta de conhecimento em banco de dados (DCBD) (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996) é uma alternativa possível. Uma das fases deste processo é o pré-processamente. Especificamente no caso de esquemas conceituais de banco de dados geográficos o pré-processamento dos dados consiste na integração de esquemas conceituais modelados com base em diferentes modelos de dados e com variações em termos de nomenclatura e estrutura para um mesmo conceito do mundo real.

A fase de pré-processamento do processo de DCBD é fundamental para o sucesso da fase subseqüente, a mineração de dados e, conseqüentemente, para todo o processo de DCBD. Os dados (esquemas) de entrada devem estar sintática e semanticamente corretos para produzir resultados satisfatórios. Desta forma, para esquemas conceituais de BDG a fase de pré-processamento deve tratar heterogeneidade em termos de modelos de dados (sintática) e também em termos de conceitos (semântica).

O nível sintático visa converter os esquemas em um formato canônico Um modelo conceitual bem definido fornece uma representação canônica da realidade geográfica, possibilitando a reutilização de sub-esquemas. O uso de um modelo de dados canônico para armazenar os esquemas conceituais a serem comparados frente a uma base de conhecimento minimiza o esforço da consulta, mas por outro lado impõe a necessidade de se converter cada um dos modelos de dados a ser utilizado no formato de dados canônico. Para alcançar os benefícios acima apontados, primeiramente, este trabalho especificou um conjunto de regras para a codificação de um esquema conceitual de BDG baseado em cada um dos modelos de dados apresentado no capítulo 4 para a linguagem de marcação GML, a qual foi escolhida por ser o padrão recomendado pelo OpenGIS para armazenamento de informações geográficas.

A integração semântica entre diferentes esquemas conceituais pode ser apoiada por uma ontologia, a qual possibilita a consulta por nomes, bem como a consulta por estruturas tais como atributos e associações. Associada ao modelo de dados canônico, a ontologia auxilia não somente o intercambio de esquemas, mas também o entendimento e solução de conflitos, tais como heterogeneidades e redundância. Para explorar todas as potencialidades das ontologias de uma forma eficiente, é necessário combiná-la com outra técnica de bastante utilizada em bancos de dados heterogêneos, conhecida por casamento por similaridade (*similarity matching*).

Tendo um algoritmo definido, assim como a ontologia ontoGeo construída e as técnicas de medida de similaridade determinadas, um conjunto de esquemas conceituais foi

confrontado com a ontologia. Os resultados foram bastante satisfatórios, com um alto índice de acertos na identificação de pares de conceitos iguais ou equivalentes entre a ontologia e os esquemas conceituais de entrada.

Para que fosse possível processar esquemas conceituais de BDG através do algoritmo, comparando com os conceitos armazenados na ontologia, esta teve que ser construída antes da primeira rodada de execuções. Contudo, para verificar se a ontoGeo seria criada de forma semelhante partindo-se de uma ontologia vazia, foi feito um teste rodando o algoritmo para os esquemas conceituais utilizados como base para a construção dela. O resultado não foi uma ontologia idêntica, mas semanticamente equivalente. As diferenças podem ser explicadas em função da ordem de processamento dos esquemas. A primeira versão da ontologia correspondia exatamente ao primeiro esquema conceitual processado. Cada execução subseqüente do algoritmo sofreu influência das execuções anteriores. Também é importante ressaltar que a ontologia geográfica ontoGeo não tem a ambição de ser uma ontologia completa. Ela apenas serve como base para a definição de outros conceitos. A idéia é que a ontologia seja completada à medida que os esquemas conceituais sejam processados.

Após algumas rodadas de execução do algoritmo algumas considerações devem ser feitas. Primeiramente, o peso dos fatores multiplicativos PN, PA, PR e PH exerce uma considerável influência sobre a medida final da similaridade. Não existe uma combinação de valores para estes pesos que esteja sempre correta. A melhor combinação depende das características do esquema conceitual a ser comparado com a ontologia. Mesmo que os valores absolutos para os parâmetros não foram totalmente corretos, a partir dos experimentos foi possível constatar que a razão estabelecida entre eles foi bastante razoável, para os esquemas conceituais testados.

Um outro aspecto relevante ocorre em relação à quantidade de informação do esquema conceitual. Quanto mais detalhado for o esquema, ou seja, quanto maior o número de atributos por classe, hierarquias e relacionamentos, mais preciso se torna o algoritmo. Se somente classes são modeladas, provavelmente um número bastante elevado de conceitos será indicado como candidatos a serem o par do elemento de cada elemento do esquema conceitual, uma vez que somente os nomes dos conceitos serão diferentes. Por outro lado, se os esquemas são modelados com uma taxonomia completa e com uma quantidade de atributos suficiente para discernir as classes (elementos) umas das outras, os conceitos presentes na ontologia terão que satisfazer um número maior de requisitos para serem considerados candidatos a par do elemento de entrada. Como resultado, serão identificados menos candidatos, mas mais corretos. Desta forma a decisão de qual conceito é mais adequado para representar a classe de entrada se tornará mais automática, bem como a decisão do especialista do domínio será facilitada, quando necessária.

Tendo em vista que os esquemas conceituais a serem integrados provêm de diferentes fontes e possuem diferentes níveis de detalhamento, podemos classificá-los quatro diferentes categorias:

- Categoria 1: Esquemas que possuem somente a definição de classes e relacionamentos de generalização e especialização;
- Categoria 2: Esquemas que possuem a definição de classes com seus atributos e relacionamentos de generalização e especialização;
- Categoria 3: Esquemas que possuem a definição de classes com seus atributos, relacionamentos de generalização e especialização e associações binárias;

- Categoria 4: Esquemas que possuem a definição de classes com seus atributos, relacionamentos de generalização e especialização, associações binárias, composição e agregação, e relacionamentos espaciais.

Os esquemas de categoria mais baixa são mais pobres semanticamente que os esquemas de categoria mais alta, uma vez que possuem menos informações para diferenciar um elemento de outro.

É importante ressaltar que neste trabalho não foram testados todos os tipos de esquemas conceituais, ou seja, representantes de todas as categorias acima. Através do estudo de caso apresentado fica evidente a importância de se possibilitar a calibragem dos parâmetros de ponderação das métricas de similaridade para cada tipo de esquema conceitual, de forma a obter a melhor combinação cada vez.

Para dar continuidade à pesquisa aqui desenvolvida alguns trabalhos futuros são propostos:

- Atualmente o algoritmo de busca e atualização da ontologia não é capaz de resolver heterogeneidades em termos de diferenças de construtores utilizados para modelar um mesmo fenômeno do mundo real. Por exemplo, se em um esquema conceitual o fenômeno é modelado como sendo uma classe e em outro como sendo um atributo de uma classe, o algoritmo não consegue identificá-los como sendo equivalentes.
- Em função da diferença de quantidade de informação semântica presente nos esquemas conceituais das categorias elencadas, o enriquecimento semântico (SPACCAPIETRA; PARENT, 2000) dos esquemas conceituais também deve ser estudado, de modo a melhorar a eficácia do processo de identificação de similares entre os elementos presentes no esquema conceitual e aquele armazenados na base de conhecimento.
- Um estudo mais aprofundado sobre o cálculo dos valores para os parâmetros PN, PA, PH e PR deve ser feito, também considerando as diferentes categorias de esquemas conceituais, bem como das fórmulas de *similarity matching*. É possível que as fórmulas e pesos tenham desempenho diferente em função da categoria do esquema conceitual.
- Uma vez tendo os padrões armazenados na ontologia, é necessário desenvolver uma metodologia para consulta e recuperação dos mesmos. Desta forma, a modelagem conceitual pode ser assistida, de modo a oferecer ao projetista os padrões que melhor solucionam as suas necessidades quando da modelagem de novas aplicações.
- À medida que a ontologia cresce e a identificação de equivalências entre elementos dos esquemas conceituais de entrada e os conceitos armazenados na ontoGeo se torna mais automática, é necessário desenvolver um mecanismo de manutenção da ontologia. Isto se faz necessário para manter a correção da ontoGeo bem como garantir que ela representa a realidade geográfica de forma fiel.

6 Referências bibliográficas

AHMED, R. el al. The pegasus heterogeneous database system. **IEEE Computer**, v.24, n.12, p.19-27, 1991.

ARONOFF, S. **Geographic Information Systems.** Canada: WDL Publications, 1989.

BAEZA-YATES, R. **Modern information retrieval**. New York: ACM, c1999. 513 p. : il.

BASSALO, G.H.M.; IOCHPE, C.; BIGOLIN, N. Representando esquemas no Formato Atributo-Valor para a Inferência de Padrões de Análise. **IV Brazilian Symposium on GeoInformatics - GeoInfo 2002**. Caxambu, Brazil, 2002.

BATINI, C.; LENSERINI, M.; NAVATHE, S. A Comparative Analysis Of Methodologies For Database Schema Integration. **ACM Computing Surveys**, New York, v.18, n.4, p.323-364, 1986.

BERGAMASCHI, S. et al. An Intelligent Approach to Information Integration. **International Conference on Formal Ontology in Information Systems (FOIS'98)**. Itália, junho, 1998.

BERGAMASCHI, S.; CASTANO, S.; VINCINI, M. Semantic Integration os semistructured and structured data sources. **ACM SIGMOD Record**. Número 28 (54-59). 1999.

BIGOLIN, N. M.. **Data Mining: conceitos e técnicas**. VII Escola de Informática da SBC Sul. Porto Alegre, RS. 1999.

BISHR, Y. A.; PUNDT, H.; KUHN, W.; RADWAN, M. Probing the Concepts of Information Communities – A First Step Towards Semantic Interoperability. **M. Goodchild, Max Egenhofer, R. Fegeas, and C. Kottman, editors, Interoperating Geographic Information Systems**, pp. 55-69, Kluwer-Academic Publishers: Norwell, MA. 1999.

BORGES, K.V.A. Modelagem de dados geográficos: uma Extensão do Modelo OMT para aplicações geográficas. Escola do governo de MG/FJP. Belo Horizonte, Brasil, 1997.

BOTÊLHO, L. R; STRAUCH, J. C. M; DE SOUZA, J. M. WISE: An Architecture for geo-referenced data integration. In **Proceedings of the International Workshop on Next Generation Geospatial Information**. Boston, EUA. 2003.

BOTÊLHO, L. R. Uma Arquitetura Extensível para Integração e Publicação de Dados Georreferenciados. 2004. Dissertação (Mestrado em Ciências em Engenharia de Sistemas e Computação) – COPPE, UFRJ, Rio de Janeiro.

BRADLEY, N. **The XML Companion**. 3rd ed. Boston: Addison-Wesley, 2002.

BURROUGH, P. A.; MCDONNELL, R. A. **Principles of Geographical Information Systems**. Great Britain: Oxford university Press, 1997.

CARLSON, D. **Modelagem de aplicações XML com UML**. São Paulo, Makron Books, 2002.

CASTANO, S.; De ANTONELLIS, V.; De CAPITANI di VEMERCATI, S. Global viewing of heterogeneous data sources. **In IEEE Transactions on Data and Knowledge Engineering**. Número 13 (277-297). 2001.

CHATTERJEE, A.; SEGEV, A. Data manipulation in heterogeneous databases. **SIGMOD Record**, v.20, n.4, p.64-68, 1991.

CHAUDHRI, V. K.; FARQUHAR, A.; FIKES, R.; KARP, P. D.; RICE, J. P. **Open Knowledge Base Connectivity 2.0.3.**, 1998.

COHEN, W.W., Integration of Heterogeneous Databases Without Common Domains Using Queries Based on Text Similarity. **Proceedings of the 1998 ACM SIGMOD international conference on Management of data**. Estados Unidos, 1998.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A comparison of String Metrics for Matching Names and Records. **IJCAI 2003 – Workshop on Information Integration on the Web**. Acapulco, México, agosto, 2003.

CUI, Z.; JONES, D.; O'BRIEN, P. Semantic B2B Integration: Issues in Ontology-based Approaches. **Sigmod record web edition**. V.31, 2002.

DAVIS JR, C. A.; LEANDER, A. H. F. Extensões ao Modelo OMT-G para Produção de Esquemas Dinâmicos e de Apresentação. **II Workshop Brasileiro de GeoInformática – GeoInfo 2000**. São Paulo, Brasil, 2002.

DOAN, A. et at. Learning to Map between Ontologies on the Semantic Web. **World Wide Web Conference 2002**. Honolulu, Hawaii, Estados Unidos, 2002.

DORNELES, C. F. **Consulta semântica por similaridade a dados semi-estruturados**. 2002. Relatório de Pesquisa RP326 (Doutorado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

ELDER, J., KRIEGEL, H.P., XU, X.. Statistical perspective on knowledge discovery in databases. **In proceedings of Advances in knowledge discovery and data mining**. Pp83-115. 1996

ELMASRI, R; NAVATHE, S. B. **Fundamentals of database systems**. Reading: Addison-Wesley, 2000.

ESTRADA, R. P. D; PAIVA, J. A. C.; CÂMARA, G.; MONTEIRO, A. M. V. Integração Semântica e de Dados entre Sistemas de Informações Geográficas Heterogêneos. **III Workshop Brasileiro de GeoInformática – GeoInfo 2001**. Rio de Janeiro, Brasil, 2001.

FILETO, R. **A Abordagem POESIA para a Integração de Dados e Serviços na Web Semântica**. 2003. Tese (Doutorado em Ciência da Computação) – Instituto de Compitação, UNICAMP, Campinas.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From Data Mining to Knowledge Discovery in Databases. **AI Magazine**, v.17, n.3,p.37-54, 1996

FONSECA, F. T., EGENHOFER, M. J., AGOURIS, P. and CÂMARA, G. Using Ontologies for Integrated Geographic Information Systems. In: **Transactions in GIS 6(3)**, 2002

FONSECA, F.; DAVIS, C.; CÂMARA, G. Bridging Ontologies and Conceptual Schemas in Geographic Information Integration. **GeoInformatica**. v.7, n.4, p.355-378. Kluwer Academic Publishers, 2003.

FOWLER, M. Analysis patterns : reusable object models. Menlo Park: Addison-Wesley, 1997. 357 p. : il.

FOWLER, M.; SCOTT, K. **UML Essencial: Um breve guia para a linguagem padrão da modelagem de objetos**. Segunda Edição. São Paulo: Bookman, 2000.

GETTY, J. P. **Getty Thesaurus of Geographic Names On Line**. Disponível em (http://www.getty.edu/research/conducting_research/vocabularies/tgn/). Ultimo acesso em março, 2004.

GOTTHARD, W.; LOCKEMANN, P.C.; NEUFELD, A. System-Guided View Integration For Object Oriented Databases. **IEEE Transactions on Knowledge and Data Engineering**. V.4, n.1, p.1-22, 1992.

GRUBER, T. R. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. **In Formal Ontology in Concepuatl Analysis and Knowledge Representation**. Agosto, 1993.

GUARINO, N. Formal Ontology and Information Systems. **International Conference on Formal Ontology in Information Systems (FOIS'98)**. Itália, junho, 1998.

HAKIMPOUR, F.; TIMPF, S. Using Ontologies for Resolution of Semantic Heterogeneity in GIS. **4th AGILE Conference on Geographic Information Science**. Brno, República Checa. Abril, 2001.

HERNÁNDEZ, M. A.; STOLFO, S. J. The merge/purge problem for large databases. **SIGMOD CONFERENCE**, [s.l.: s.n.] 1995.

HESS, G. N. **Utilização da GML como interface para o Uso de Modelos Conceituais de Sistemas de Informações Geográficas na Descoberta de Padrões de Modelagem**. 2003. Trabalho Individual (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

HESS, G.N.; IOCHPE, C.; SILVA, C.M.S. RoseGIS: Uma ferramenta CASE para projeto de banco de dados geográficos. **In GISBrasil 2003**. São Paulo, Brasil, agosto, 2003.

HESS, G. N.; IOCHPE, C. Utilizando a GML na Identificação de Candidatos a Padrão de Análise para BDG. **In V Simpósio Brasileiro de Geoinformática (GEOINFO 2003)**. Campos do Jordão, Brasil. Novembro, 2003.

HODGE, G. Knowledge Organization Systems: An Overview. **System of knowledge Organization for Digital Libraries: Beyond Traditional authority files**. Abril, 2000.

HOLT, A. Understanding environment and geographical complexities trough similarity matching. **Complexity International**, number 7, 2000.

JARO, M. A. Advances in Record-Linkage Methodology as Applied toMatching the 1985 Census of Tampa, Florida. **Journal of the American Statistical Association**, número 89, p.414-420, 1989.

JIANG, J. J.; CONRATH, D. W. Semantic Similarity Based in Corpus Statistics and Lexical Taxonomy. **International Conference Reasearch in Computational Linguistics (ROCLING X)**. Taiwan, 1997.

KÖSTERS, G.; PAGEL, B.; SIX, H. GeoOOA: Object Oriented Analysis for Geographic Information Systems. **2nd International Conference on Requirements Engineering (ICRE '96)**. Colorado, Estados Unidos, 1996.

LAWRENCE, S.; GILES, C. L.; BOLLACKER, K. D. Autonomous Citation Matching. **Third International Conference on Autonomous Agents**. Seatle, Estados Unidos, maio, 1999.

LIMA, P.; CÂMARA, G.; QUEIROZ, G. GeoBR: Intercâmbio Sintático e Semântico de Dados Espaciais. **In IV Brazilian Symposium on GeoInformatics - GeoInfo 2002**. Caxambu, Brazil, 2002.

LISBOA J.; IOCHPE, C. Specifying analysis patterns for geographic databases on the basis of a conceptual framework. **7th ACM GIS**. Kansas City, Estados Unidos, 1999.

LISBOA, J. Desenvolvimento de uma ferramenta CASE para o Modelo UML-GeoFrame com suporte para padrões de análise. **IV Brazilian Symposium on GeoInformatics - GeoInfo 2002**. Caxambu, Brasil, 2002.

MADHAVAN, J.; BERNSTEIN, P.; RAHN E. Generic Schema Matching with Cupid. **In 27TH International Conference on Very Large Databases**. 2001.

MELLO, R. S.; HEUSER, C. A. A Bottom-Up Approach for Integration of XML Sources. **International Workshop on Data Integration on the Web**. IME, Rio de Janeiro, Brasil, 2001.

MILLER, G. A. WordNet: A lexical database for English. Communications of the ACM, 38(11):39-41. 1995.

MILO, T.; ZOHAR, S. Using Schema Matching to Simplify Heterogeneous Data Translation. **International Conference on Very Large Databases (VLDB)**. New York, Estados Unidos, 1998.

MITRA, P.; WIEDERHOLD, G.; JANNIK, J. Semi-automatic integration of knowledge sources. **In Fusion'99**. Estados Unidos, 1999.

MONGE, A. E.; ELKAN, C. P. The field matching problem: Algorithms and Applications. **Second International Conference and Data Mining**. 1996.

MUÑOZ, L. S. **Estudo de ontologias para representação de conteúdos de ensino baseado na www**. 2002. Trabalho Individual (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

NOY, Natalya F. et al. **The Knowledge Model of Protégé 2000: Combining Interoperability and Flexibility.** *Stanford Medical Informatics, Standford University, Stanford, CA*. 2000. Disponível em http://citeseer.nj.nec.com/noy01knowledge.html

NOY, N. F.; McGUINNESS, D. L. **Ontology Development 101: A Guide to Creating Your First Ontology**. Technical Report, Stanford University, Estados Unidos. 2001.

OGC - OpenGIS Consortium. **Geography Markup Language (GML) 3.0.** Open GIS Implementation Specification, 2003. Disponível em <http://www.opengis.net>. Acesso em dezembro de 2003.

PALOPOLI, L.; SACCA, D.; URSINO, D. Semi-automatic, semantic discovery of properties from database schemas. **In International Database Engineering and Applications Symposium (IDEAS)**. 1998.

PARENT, C. et al. Modeling Spatial Data in the MADS Conceptual Model. **International Symposium on Spatial Data Handling, SDH 98**. Vancouver, Canadá, 1998.

PARENT, C. et al. Spatio-temporal conceptual models: data structures + space + time. **7th ACM GIS**. Kansas City, Estados Unidos, 1999.

PARK, J. **Schema Integration Methodology and Toolkit for Heterogeneous and Distributed Geographic Databases**. University of Minnesota, Estados Unidos., 2001 (working paper).

PARTRIDGE, C. **The Role of Ontology in Integrating Semantically Heterogeneous Databases**. Technical Report. Itália, junho, 2002.

PHILIPS, J.; BUCHANAN, B. G. Ontology-Guided Knowledge Discovery in Databases. **International Conference on Knowledge Capture**. Victoria, Canada. 2001.

QIN, J.; PALING, S. Converting a controlled vocabulary into an ontology: the case of GEM, **Information Research**, number 6, 2001.

RAHM, E.; BERNSTEIN, P. A survey of approaches to automatic schema matching. **The VLDB Journal**. Número 10 (334-350). 2001.

RATIONAL ROSE 2000e. Rose Extensibility User's Guide, 2000. Disponível em www.cs.hmc.edu/qref/rational/DevelopmentStudioUNIX.2000.02.10/docs/pdf/rose_REI_guide/Rose_REI_guide.pdf. Ultimo acesso em set. 2003.

RESNIK, P. Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. **Journal of Artificial Intelligence Research**, number 11, p. 95-130, 1998.

RIBEIRO, C. H. F. P. **Banco de Dados Heterogêneos: Mapeamento dos Esquemas Conceituais em um Modelo de Dados Orientado a Objetos**. 1995. Tese (Doutorado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

RICHARDSON, R.; SMEATON, A. F.; MURPHY, J. Using WordNet as a Knowledge Base for Measuring Semantic Similarity between Words. **AICS Conference**. Dublin, Irlanda, 1994.

ROCHA, L.V.; IOCHPE, C.; EDELWEISS, N. **O framework conceitual GEOFRAME versão 2.0. 2001**. Relatório de pesquisa (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

ROCHA, L. V.; EDELWEISS, N.; IOCHPE, C. GeoFrame-T: A Temporal Conceptual Framework for Data Modeling. **ACM Symposium on Advances in GIS**. Atlanta, Estados Unidos, 2001

SAIF 3.2. Ministry of sustainable resource management. Geographic Data BC**.** Province of British Columbia, Canadá, 2001. Disponível em <http://home.gdbc.gov.bc.ca/SAIF>. Acesso em abril 2003.

SACCOL, D.; HEUSER, C.A. Integration of XML data. **First VLDB Workshop on Efficiency and Effectiveness of XML tools and Techniques**. Hong Kong, 2002.

SHETH, A. P. Changing focus on interoperability in information systems: From systems, syntax, structure to semantics. **Interoperating Geographic Information Systems**, 2000.

SHOENS, K. A. et al. The rufus system: information organization for semi-structured data. **International Conference on Very Large Databases**. Dublin, Irlanda. Agosto, 1993.

SILVA, C. M. S.; IOCHPE, C.; ENGEL, P. M. Using Knowledge Discovery in Database to Identify Analysis Patterns. **5th International Conference on Enterprise Information System**. Angers, França, 2003.

SILVA, C. M. S. **Utilizando o Processo de Descoberta de Conhecimento em Banco de Dados para Identificar Candidatos a Padrão de Análise para Bancos de Dados Geográficos**. 2003. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

SMITH, B.; MARK, D. M. Ontology and geographic kinds. **In Proceedings of the International Symposium on Spatial Data Handling**. 1998. Vancouver, Canada

SPACAPPIETRA, S.; PARENT, C.; ZIMANYI, E. Modeling Time from a Conceptual Perspective. **International Conference on Information and Knowledge Menagement – CIKM'98**. Washington D.C., Estados Unidos, 1998.

SPACCAPIETRA, S.; PARENT, C. Database Integration: The Key to Data Interoperability. **In Advances in Object-Oriented Data Modeling,** M. P. Papazoglou, S. Spaccapietra, Z. Tari (Eds.). The MIT Press, 2000.

STOIMENOV, L.; DJORDJEVIC-KAJAN, S. Realization of GIS Semantic Interoperability in Local Community Environment. **6th AGILE Conference on Geographic Information Science**. Lyon, França. Abril, 2003.

SUGUMARAN, V.; STOREY, V. C. Ontologies for Conceptual Modeling: their creation, use and management. **Data & Knowledge Engineering**. Elsevier, abril, 2002.

TEIXEIRA, A.L.A.; CHRISTOFOLETTI, A. **Sistemas de Informação Geográfica – Dicionário Ilustrado**. Ed. Hucitec, 1997. Disponível em: http://www.fatorgis.com.br. Acesso em: 01 mar. 2004.

TURNEY, P.D. Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. **European Conference on Machine Learning**. 2001.

UITERMARK, H. T.; VAN OOSTEROM, P. J. M.; MARS, N. J. I.; MOLENAAR, M. Ontology-Based Geographic Data Set Integration. **International Workshop on Spatio-Temporal Database Management (STDBM'99)**. Edinburgo, Escócia. Setembro, 1999.

UNITED NATIONS HIGH COMISSIONER FOR REFUGEES LIBRARY. The International Thesaurus of Refugee Terminology. Disponível em http://www.refugeethesaurus.org. Último acesso em junho de 2004.

USCHOLD, M.; GRUNINGER, M. Ontologies: Principles, Methods and Applications. **Knowledge Engineering Review**, volume 11, number 2. Junho, 1996.

VERSCHELDE, J. L.; SANTOS, M. C.; DERAY, T.; SMITH, B.; CEUSTERS, W. Ontology-assisted database integration to support natural language processing and biomedical data-mining. **Journal of Integrative Bioinformatics**. No 0001. 2004.

VISSER, P.R.S. et al. An Analysis of Ontology Mismathces: Heteroneneity versus Interoperability. **AAAI 1997 Spring Symposium on Ontological Engineering**. Estados Unidos, 1997.

WINKLER, W. E. The State of Record Linkage and Current Research Problems. **Statistics of Incom Division, Internal Revenue**. Service Publication R99/04, Seatle, Estados Unidos, 1999.

WORLD WIDE WEB CONSORTIUM (W3C). **RDF Vocabulary Description Language 1.0: RDF Schema**. February, 2004. Disponível em http://www.w3.org/TR/rdf-schema/. Último acesso em abril de 2004.

XAVIER, A. C. **Geração automática de esquemas GML a partir de modelagens utilizando Geoframe-T**. 2002. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

YANILOS, P. N.; KANZELBERGER, K. G. The LikeIt Intelligent String Comparison Facility. **Technical Report**. NEC Research Institute, maio, 1997.

Anexo 1 – MODELOS DE DADOS GEOGRÁFICOS

A1.1 Modeling of Application Data with Spatio-Temporal features (MADS)

O MADS (PARENT et al., 1998) é um projeto desenvolvido na universidade de Lousanne, na Suíça, que tem como objetivo oferecer um modelo conceitual espaço-temporal e um conjunto de ferramentas para o desenvolvimento de SIG (PARENT et al. 1999). As principais características do modelo de dados são:

- Ser um modelo conceitual;
- Possibilidade de definir esquemas que são legíveis e inteligíveis por usuários;
- Notação visual intuitiva;
- Definição formal de todos os conceitos;
- Suficientemente rico em expressividade para suportar diversos tipos de aplicações;
- Independência entre os conceitos, mas fácil associação entre eles;
- Possibilidade de descrição explícita de um relacionamento espaço-temporal entre elementos do modelo;

O MADS está restrito a modelagem bi-dimensional, e além de aspectos espaciais, é capaz de representar a temporalidade dos elementos do modelo. Outra característica do MADS está na forma como os elementos são considerados. Apesar de poder representar a visão de campo, o modelo favorece a visão discreta do espaço, ou seja, a visão de objetos.

O modelo conceitual do MADS adota o paradigma objeto relacional, o qual inclui as características do padrão ODMG (*Object Database Management Group*) para sistemas orientados a objetos. Deste modo, um conjunto de conceitos bem conhecidos é suportado: objeto, relacionamento, atributo, método, generalização e especialização, e agregação.

Um objeto (*object*) representa uma entidade do mundo real. Um tipo objeto (*objetct type*) é um usado para descrever um conjunto de objetos com estrutura e comportamento similar.

Um relacionamento (*relationship*) é uma associação entre dois ou mais objetos, onde cada um tem um papel (função). Um tipo relacionameno (*relationship type*) é usado para descrever um conjunto de relacionamentos com características similares, ou seja, que associam objetos do mesmo tipo, com os mesmos papéis e com propriedades similares.

Um atributo (*attribute*) representa uma propriedade de um objeto ou relacionamento do mundo real. Podem ser simples ou complexos, mono ou multivalorados, obrigatórios ou opcionais. Estes dois últimos pares de possibilidades representam a cardinalidade. O valor

de um atributo pode ser atribuído diretamente a ele ou ser derivado, ou seja, calculado ou inferido a partir de outros objetos e/ou relacionamentos.

Um método (*method*) é uma operação que pode ser realizada sobre um objeto. Sua especificação envolve uma interface, na qual o nome, o tipo e os parâmetros do método são especificados, e uma ou mais implementações.

A generalização (*generalization*) liga dois tipo objeto (*object type*) relacionados: um supertipo (*supertype*) e um subtipo (*subtype*). Ambos representam a mesma entidade do mundo real, porém em dois níveis de abstração diferentes. Herança múltipla é permitida. As propriedades podem ser refinadas em um subtipo, ou ainda redefinidas.

A agregação é um relacionamento especial, binário, cuja semântica expressa que um objeto de composição é um agregado de objetos de um tipo diferente, chamados objetos componentes.

No MADS, a espacialidade deve estar associada a objetos, atributos, relacionamentos e agregações. Sua representação é feita por meio de ícones, o que possibilita uma interpretação visual, não ambígua e sintética. Este ícone tem a mesma semântica de um atributo, chamado geometria (*geometry*). Os tipos espaciais suportados pelo modelo são ponto (*point*), linha (*line*), linha orientada (*oriented line*) e área simples (*simple area*). Ainda há a possibilidade de representação de conjuntos dos tipos simples. Adicionalmente, o MADS define tipos espaciais mais genéricos, chamados geo, o qual especializa-se em *simple geo*, o qual pode ser usado para qualquer tipo espacial, e *complex geo*, utilizado para representar conjuntos de objetos espaciais de tipos diferentes. A cada tipo espacial é associado um conjunto de métodos para definição e manipulação de instâncias deste tipo. A hierarquia dos tipos espaciais do MADS pode ser visualizada na Figura A1.1.

Um objeto espacial (*spatial object*) representa uma entidade cuja forma (geometria) e localização devem ser representados. A representação da espacialidade é descrita através de um dos atributos espaciais, o qual é um atributo simples, mono ou multivalorado, cujo domínio é um *Abstract Data Type* (ADT) espacial.

É através dos relacionamentos espaciais, além dos objetos espaciais, que se define uma aplicação espacial. Os relacionamentos podem ser do tipo topológico, de orientação, métrico ou de agregação espacial. No MADS, um relacionamento espacial deve ligar ao menos dois objetos espaciais (e possivelmente outros não espaciais), cuja semântica é explicitamente definida através das restrições de integridade espaciais.

O MADS predefine dois tipos de relacionamento espacial: topológico e de agregação. Isto não impede que outros relacionamentos sejam expressos, mas para isso devem ser explicitamente declarados, com os métodos anexados aos ADT.

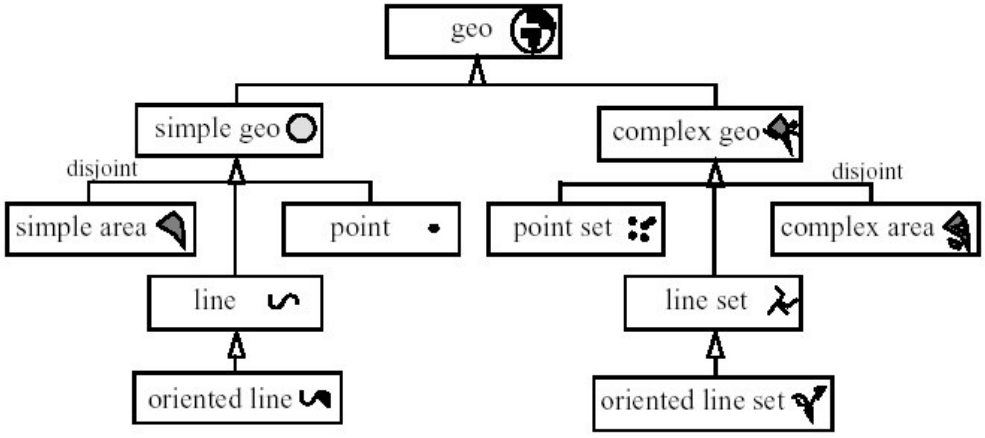


Figura A1.1 - Hierarquia básica dos tipos espaciais abstratos (PARENT et al., 1998)

Os relacionamentos topológicos são utilizados para associar objetos espaciais, sendo necessário especificar o tipo do relacionamento, visto que, conforme demonstra a Tabela A1.1, cada tipo possui uma semântica diferente.

Tabela A1.1 - Tipos de relacionamentos topológicos existentes no MADS (PARENT et al., 1998)

Tipo espacial	**definição**
Disjunction	Objetos espaciais cujas geometrias são disjuntas.
Adjacency	Objetos espaciais que possuem parte da geometria em comum, mas sem se cruzar.
Cruzamento	Objetos espaciais que possuem parte do interior comum, mas na qual a área compartilhada é de dimensão inferior à dimensão do maior dos objetos relacionados.
Overlapping	Objetos espaciais de mesma dimensão dividem uma área comum, que abrange parte da geometria de ambos.
Inclusion	Um objeto espacial está, espacialmente, incluso no outro.
Equality	Objetos espaciais são exatamente iguais em geometria e ocupam o mesmo lugar no espaço.

O relacionamento de agregação espacial é uma ligação binária dirigida, do objeto composto para o componente. Para um objeto composto formado por diversos objetos componentes, há um relacionamento de agregação para cada tipo de objeto componente. Há a possibilidade, neste tipo de relacionamento, de alguns atributos do objeto composto e de sus componentes estarem relacionados. Esta dependência pode ser representada tanto por atributos derivados quanto por restrições de integridade.

O MADS apresenta-se como sendo temporal, aceitando temporalidade em nível de classes, atributos e relacionamentos. Eventos (relacionamentos) dinâmicos também são contemplados (SPACAPPIETRA;PARENT;ZIMANYI, 1998):

- Transição: Caracteriza a mudança de estado de uma instância. Pode ser de dois tipos, evolução ou extensão.
    - Evolução: Um objeto pertencente a uma classe evolui e torna-se uma instância de outra classe, abandonando a classe de origem;
    - Extensão: Um objeto pertencente a uma classe torna-se uma instância de outra classe também, mas sem deixar de pertencer à classe de origem.

- Geração: Representa os processos nos quais um conjunto de instâncias gera uma ou mais novas instâncias de outra classe. Pode ser de transformação ou produção.
    - Transformação: As instâncias que servem de insumo são totalmente consumidas para gerar a(s) nova(s) instância(s), ou seja, deixam de existir;
    - Produção: As instâncias utilizadas como fonte se mantêm vivas após a geração da nova instância de outra classe.

Por último, há a generalização e especialização, as quais podem ocorrer entre objetos espaciais e não espaciais. Um objeto não-espacial pode ser especializado em objetos espaciais, mas objetos espaciais somente poderão ser especializados em outros objetos espaciais. Todas as propriedades essenciais do modelo podem ser refinadas ou redefinidas, conforme já mencionado. A herança múltipla também é permitida no MADS, sendo que o objeto espacial especializado herda as propriedades e a geometria de todos seus ancestrais.

A1.2 OMT-G

O modelo OMT-G (BORGES, 1997) foi concebido na Escola de Governo de MG/Fundação João Pinheiro, no Brasil. É baseado no modelo OMT, com extensão espacial. O OMT-G tem seus alicerces em três conceitos principais, que são classes, relacionamentos e restrições de integridade espacial. Estas primitivas geográficas permitem a modelagem da geometria e da topologia dos objetos. Estruturas do tipo todo-parte, redes, relacionamentos espaciais e múltiplas visões dos objetos são suportadas pelo modelo.

No modelo OMT-G, o diagrama de classes é usado para descrever a estrutura e o conteúdo de um banco de dados geográfico. Ele contém elementos específicos da estrutura do banco de dados, em especial classes de objetos e seus relacionamentos, sendo que transformações e processos dinâmicos não são considerados, na versão original. Uma extensão ao modelo é apresentada em (DAVIS; LEANDER, 2000) para suporte a aspectos dinâmicos e de apresentação. No diagrama de classes é especificada qual será a representação espacial adotada por cada classe, as quais podem ser convencionais, e neste caso não terão representação espacial, ou georeferenciadas.

As classes convencionais são aquelas que não possuem atributos espaciais (geometria) As classes convencionais podem ou não apresentar relacionamentos com classes georeferenciadas, e têm propriedades e comportamentos como qualquer classe não espacial.

As classes georeferenciadas são aquelas com representação espacial e com referência em relação à superfície da Terra. Elas se dividem em geo-objetos e geo-campos. Geo-objetos são aquelas classes usadas para descrever os fenômenos geográficos na visão de objetos. Um geo-objeto especializa-se em ponto, linha, polígono, nó (de rede), arco unidirecional e arco bidirecional. As três primeiras fazem parte do grupo geo-objetos com geometria, e as demais são especializações da classe geo-objeto e com geometria e topologia, usadas para redes.

As classes especializadas a partir de geo-campo são aquelas utilizadas para representar fenômenos geográficos da realidade na visão de campo. Estas classes podem ser isolinhas, subdivisão planar (polígonos adjacentes), tesselação, rede irregular de triângulos (TIN) ou amostragem.

A Figura A1.2 ilustra as diferentes classes que podem ser utilizadas no modelo OMT-G.

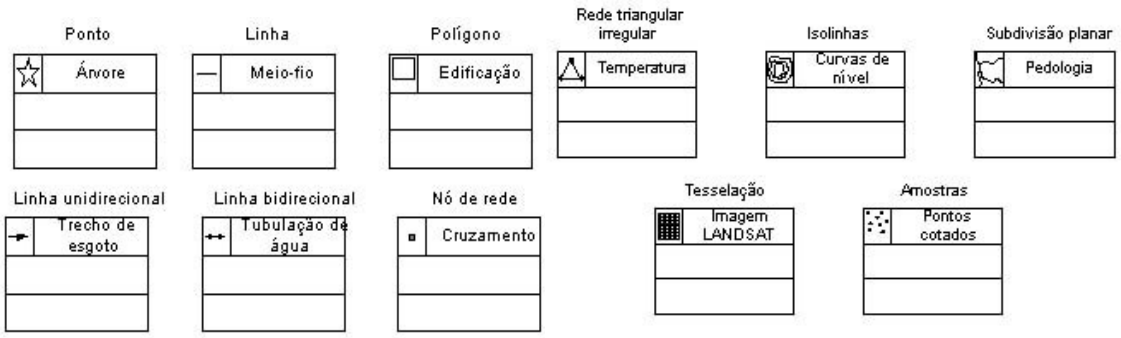


Figura A1.2 - Geo-campos e geo-objetos (BORGES, 1997)

Os relacionamentos são separados em três tipos, no modelo OMT-G: Associação simples, relacionamentos topológicos em rede e relacionamentos espaciais. As associações simples representam os relacionamentos entre classes diferentes, convencionais ou georeferenciadas. Relacionamentos espaciais representam relações topológicas, métricas, ordinais ou *fuzzy*. Por último, relacionamentos topológicos em rede ocorrem entre objetos que estão conectados uns aos outros. A representação de cardinalidade é a mesma adotada na UML.

A agregação é um tipo de associação, na qual um objeto é montado a partir da composição de outros. A agregação pode ocorrer entre classes convencionais, classes georeferciadas e, ainda, entre uma classe convencional e uma classe georeferenciada. Quando a agregação ocorre entre classes georeferenciadas, faz-se uso de uma categoria especial de agregação. Neste caso a geometria da parte deve estar contida na geometria do todo. As agregações espacias, no modelo OMT-G, devem ser por *subdivisão espacial*, *união espacial* ou *pertinência*.

Na subdivisão espacial o todo é formado pela agregação de partes de mesma natureza, e sua geometria é completamente coberta pela geometria das partes. Na união espacial o todo é formado é formado pela união das partes, ao passo que na pertinência a geometria do todo contém a geometria das partes, e objetos com naturezas geométricas diferentes podem estar contidos no todo.

No modelo OMT-G, as abstrações de generalização e especialização se aplicam tanto às classes georreferenciadas quanto às classes convencionais. Graficamente, são representadas segundo a notação UML. As generalizações podem ser totais ou parciais. A generalização total ocorre quando a união de todas as subclasses é equivalente ao conjunto completo das instâncias da superclasse. Para elementos de restrição espacial pré-definidos, o modelo OMT-G adota a notação OMT, que são disjunto e sobreposto. A Figura A1.3 ilustra a hierarquia de dados do OMT-G.

Em virtude da necessidade de poder apresentar um mesmo objeto espacial com múltiplas representações, foi incluída no modelo a primitiva de generalização cartográfica. Esta generalização pode ocorrer em duas variantes, de acordo com a forma e de acordo com a escala. O primeiro caso é utilizado quando houver existências simultâneas de múltiplas representações do objeto, independentemente da escala de visualização. A variação em função da escala é empregada na representação de diferentes aspectos geográficos de uma determinada classe, cada um correspondendo a uma faixa de escala diferente.

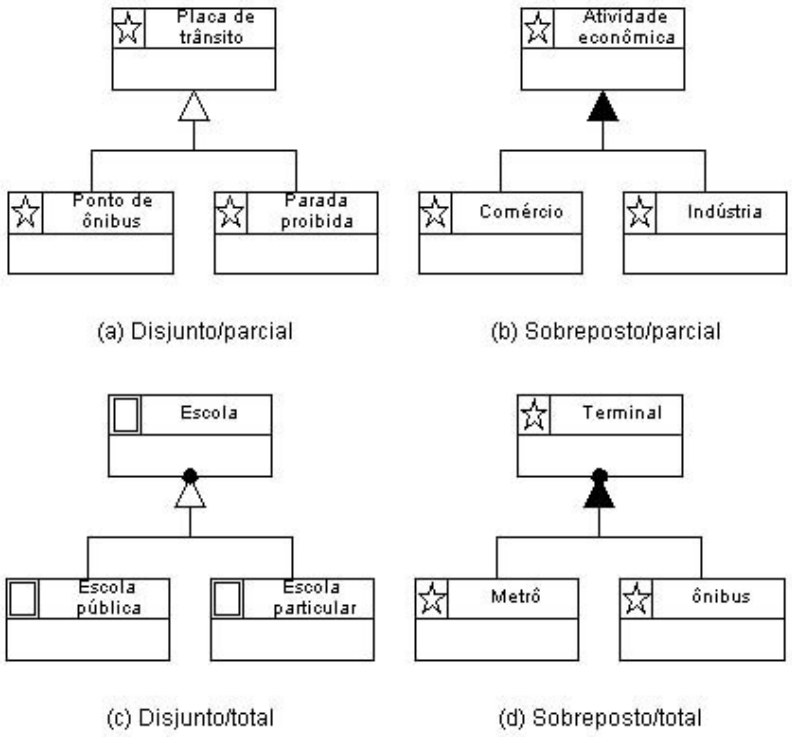


Figura A1.3 - Generalizações espaciais (BORGES, 1997)

No que tange às restrições de integridade, o modelo OMT-G apresenta três conjuntos distintos: semânticas, topológicas e definidas pelo usuário. Restrições semânticas são aquelas que estão diretamente relacionadas ao significado real dos fenômenos geográficos (ex: uma ponte não pode cortar um edifício). As restrições topológicas são aquelas que dizem respeito às propriedades geométricas e aos relacionamentos espaciais entre os objetos (ex: um edifício de um parque industrial não pode ficar fora da área do parque). As restrições definidas pelo usuário são aquelas válidas apenas no escopo da aplicação que está sendo modelada (ex: Um edifício não pode ter mais que três andares, em determinado bairro).

A1.3 UML-GeoFrame

O Geoframe é um framework conceitual que fornece um diagrama de classes básicas para auxiliar o projetista nos primeiros passos da modelagem conceitual de dados de uma nova aplicação de SIG (LISBOA; IOCHPE, 1999). Foi desenvolvido no Instituto de Informática da UFRGS, estando atualmente na segunda versão (ROCHA; IOCHPE; EDELWEISS, 2001). Um outro trabalho deu origem ao GeoFrame-T (ROCHA; EDELWEISS; IOCHPE, 2001), que é uma extensão temporal para o GeoFrame. Este framework foi definido de acordo com as regras do formalismo da orientação a objetos, utilizando a notação gráfica do diagrama de classes da linguagem UML.

Os construtores da UML suportados pelo GeoFrame são Pacote (tema), composição, estereótipos, classes, heranças e associações. Os pacotes são construções da UML cuja semântica é a de particionar o diagrama de classes, através do agrupamento de elementos,

de acordo com o assunto enfocado. A composição é um tipo especial de agregação, na qual o todo somente existe se estiverem presentes todas as suas partes.

Estereótipos são mecanismos de extensão dos construtores, cujo objetivo é representar algo que não tem uma semântica previamente definida, ou seja, cabe a que modela definir. No caso do GeoFrame, os estereótipos são empregados para substituir o relacionamento entre um fenômeno geográfico e sua representação espacial. Sua representação gráfica é de um pictograma.

As classes são os construtores que representam os elementos do mundo real no modelo conceitual. Os conceitos de associação e herança também são suportados pelo GeoFrame.

O GeoFrame está baseado, principalmente, nas classes *ObjetoNãoGeográfico* e *FenômenoGeográfico*, e na utilização de pacotes (Tema), conforme mostra a Figura A1.4. Cada pacote Tema pode ser composto, recursivamente, por outros pacotes Tema.

- *ObjetoNãoGeográfico* – são os objetos convencioanis, ou seja, que não possuem referência em relação à sua posição geográfica.
- *Metadado* – são dados que descrevem os próprios dados, especificamente dos objetos não geográficos.
- *GeoMetadado* – especialização da classe *Metadado*, com o atributo *regiãoGeográfica*.
- *FenômenoGeográfico* – classe que representa qualquer fenômeno de interesse da aplicação, cuja localização geográfica deve ser considerada.
- *CampoGeográfico* – especialização da classe *FenômenoGeográfico*, que generaliza os fenômenos que se enquadram na visão de campo, ou seja, apresentam uma distribuição contínua no espaço. A realidade é modelada por um todo, onde não se consegue analisar individualmente cada parte.
- *ObjetoGeográfico* – especialização da classe *FenômenoGeográfico*, que generaliza os fenômenos que se enquadram na visão de objeto, ou seja, cada objeto é modelado individualmente, sendo que cada um deles apresenta suas características particulares.
- *ObjetoEspacial* – classe que apresenta um conjunto de construtores necessários para representação espacial de objetos geográficos. É especializada nas subclasses *Ponto*, *Linha*, *Polígono* e *ObjetoEspComplexo*.

- *RepresentaçãoCampo* – classe usada para a representação espacial de campos geográficos. É especializada nas subclasses *GradeCélulas*, *PolAdjacentes*, *Isolinhas*, *GradePontos*, *TIN* e *PontosIrregulares*.

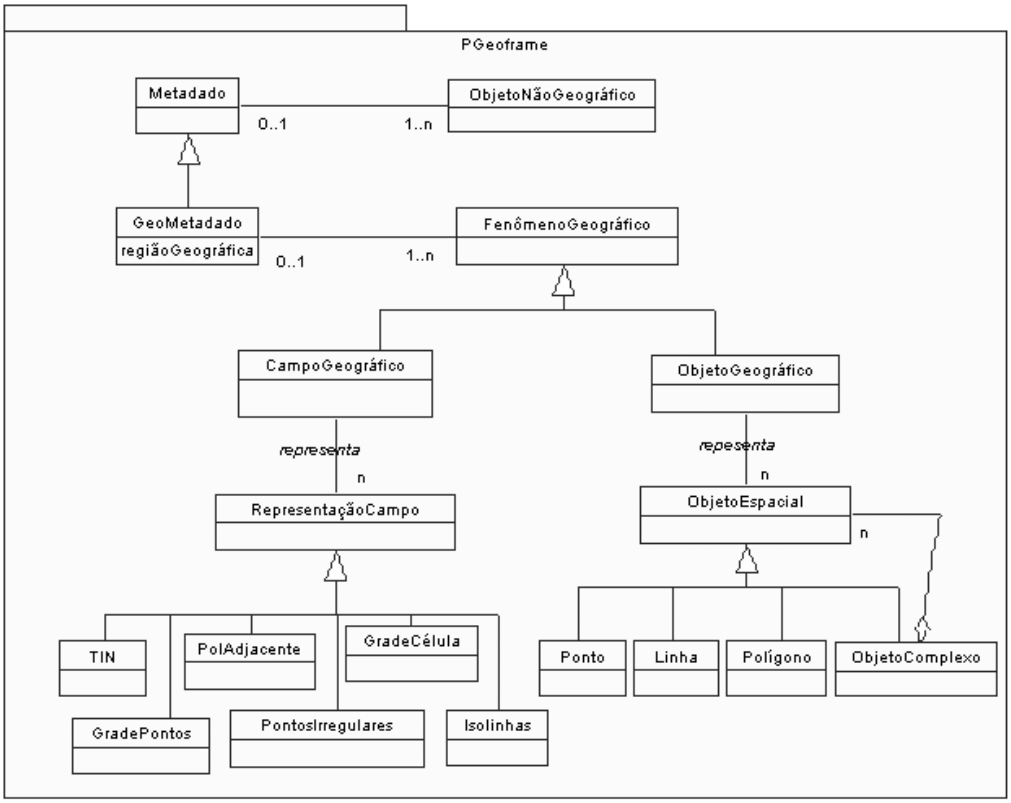


Figura A1.4 - O *framework* GeoFrame V2 (ROCHA; IOCHPE; EDELWEISS, 2001)

O framework Geoframe ainda faz uso de estereótipos, a fim de tornar o diagrama mais limpo. Como todo fenômeno geográfico possui sempre uma geometria associada a ele, cada estereótipo representa essa geometria. A sua semântica é a de substituição da associação entre o fenômeno geográfico e a classe de sua componente espacial, além da indicação da forma geométrica de sua representação. A Figura A1.5 ilustra os estereótipos presentes no GeoFrame.

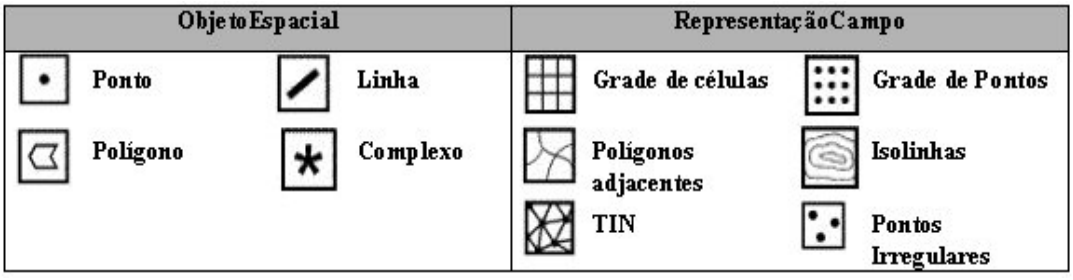


Figura A1.5 - Estereótipos do GeoFrame (ROCHA; IOCHPE; EDELWEISS, 2001)

A1.4 O Formato canônico: GML 3.0

Tendo a GML sido escolhida como modelo dedados canônico para a integração dos modelos conceituais, os demais modelos de dados devem ser mapeados pare a GML. Apesar da GML ainda não ser capaz de descrever todos os construtores existentes nos demais modelos de dados, conforme apresentado na Tabela A1.2, ela foi escolhida pelos seguintes motivos:

- É um padrão estabelecido pelo Open GIS Consortium (OGC), que é o grupo que controla a padronização de todo e qualquer produto (ou especificação) que envolva sistemas de informações geográficas;
- Sendo baseado em XML, é bastante indicado para o armazenamento, transporte e intercâmbio de dados geográficos através da Internet;
- É um padrão em evolução. Várias novas funcionalidades foram adicionadas da especificação 2.0 de 2001 para a versão 3.0 de 2003. Outras potencialidades já estão previstas para uma versão 4.0;

Por ser baseada em XML, é, por princípio, extensível. Deste modo, aquelas funcionalidades não contempladas originalmente podem ser especificadas. No escopo deste trabalho não se estendeu a GML para contemplar as características faltantes devido à dificuldade de padronizar propostas junto ao OGC. Assim, de nada adiantaria criar novos elementos, se outras ferramentas não seriam capazes de reconhecê-los.

A Geography Markup Language (GML) é uma codificação XML (BRADLEY, 2002) para transporte e armazenamento de informação geográfica, incluindo as propriedades espaciais e não espaciais (OGC, 2003; HESS, 2003). É um padrão proposto pelo consórcio OpenGIS (OGC). A especificação da GML está baseada na sintaxe XML-Schema (BRADLEY, 2002), e provê mecanismos e convenções para:

- Geração de um *framework* aberto para definição de esquemas e objetos de aplicações geoespaciais;
- Suporte à descrição de esquemas de aplicações geoespaciais específicas para um domínio;
- Suporte ao armazenamento e transporte de aplicações e conjunto de dados;
- Possibilidade de troca de informações e esquemas de aplicações geoespaciais entre organizações.

A GML está baseada na especificação feita pelo OpenGIS consortium para modelagem dos aspectos geográficos do mundo. Neste sentido, são reconhecidos fenômenos de zero, uma, duas dimensões ou três dimensões. No modelo GML, as geometrias tradicionais de zero, uma, duas e três dimensões são definidas num sistema de referência espacial (SRS) tridimensional, e são representados como pontos, curvas (ou linhas), superfícies (polígonos) e sólidos. Ainda há representação para coleções de outras geometrias.

Os fenômenos do mundo real são tratados, na GML, como *features*. Estas *features* possuem propriedades, tanto espaciais (geométricas) quanto descritivas (simples, como inteiros, strings, etc.).

Os principais objetivos da GML são:

- Prover meios de codificação da informação espacial tanto para transporte quanto para armazenamento de dados, especialmente na *web*;
- Ser suficientemente extensível para suportar uma vasta gama de tarefas espaciais;
- Estabelecer os alicerces para utilização de SIG na Internet;

- Permitir uma eficiente codificação para geometrias geo-espaciais;
- Prover uma codificação fácil de ser entendida, tanto para informações espaciais quanto para relacionamentos espaciais;
- Ser capaz de separar dados espaciais dos dados não espaciais;
- Permitir uma fácil integração entre dados espaciais e não espaciais, especialmente se os não espaciais estão codificados em XML;
- Ser capaz de fazer ligações entre objetos espaciais e outros objetos, sejam eles espaciais ou não;
- Prover um conjunto base de modelos geográficos para permitir a interoperabilidade entre aplicações.

A GML é plenamente compatível com as normas definidas pela ISO para dados geográficos, de acordo com as seguintes definições:

- ISO DIS 19107 – Spatial Schema;
- ISO DIS 19108 – Temporal Schema;
- ISO DIS 19118 – Encoding;
- ISO DIS 19123 – Coverages.

A GML 3.0 contempla tanto a codificação de fenômenos geográficos discretos (objetos) quanto contínuos (campos). Além disso, aspectos de topologia de redes também podem ser descritos em GML. A temporalidade é outra característica presente na especificação 3.0 deste padrão. Além disso, a GML possui as funcionalidades para definição e codificação de sistemas de referências geográficas, sistemas de medidas, observação e descrição de metadados.

A Figura A1.6 ilustra, de um modo genérico, a hierarquia de esquemas da GML 3.0.

Assim como em XML, cada objeto é considerado um elemento. Em GML, todos os elementos são derivados a partir de "gml:_Object". Pode ser um *object* ou uma *property*. Somente uma *property* pode ser um sub-elemento de um *object*.

O primeiro e mais básico dos esquemas da GML 3.0 é o *gml Base Schema* (gmlBase.xsd). Neste esquema estão definidos os tipos básicos da GML e os elementos que servem de suporte aos metadados. Também é este esquema que serve como base para definição de outros tipos, sejam objetos ou propriedades. Os principais tipos gerados a partir de *Base Schema* são:

- Xlinks: forma de referenciar elementos externos ao documento;
- NullType: define os tipos nulos possíveis. São eles: inaplicável (*Inapplicable*), faltante (*missing*), de exemplo (*template*), desconhecido (*unknown*) e não fornecido (*whithheld*);
- SignType: Sinal (+/-), utilizado para definir origem e destino em uma topologia de rede;
- SimpleType: Tipos básicos de dados – string, number, boolean, etc.
- CodeType: utilizado para definição de códigos;
- MeasureType: define as medidas básicas que a GML suporta, tal como comprimento, volume, peso, etc;
- CoordinateType: define os tipos de coordenadas geográficas reconhecidas pela GML.

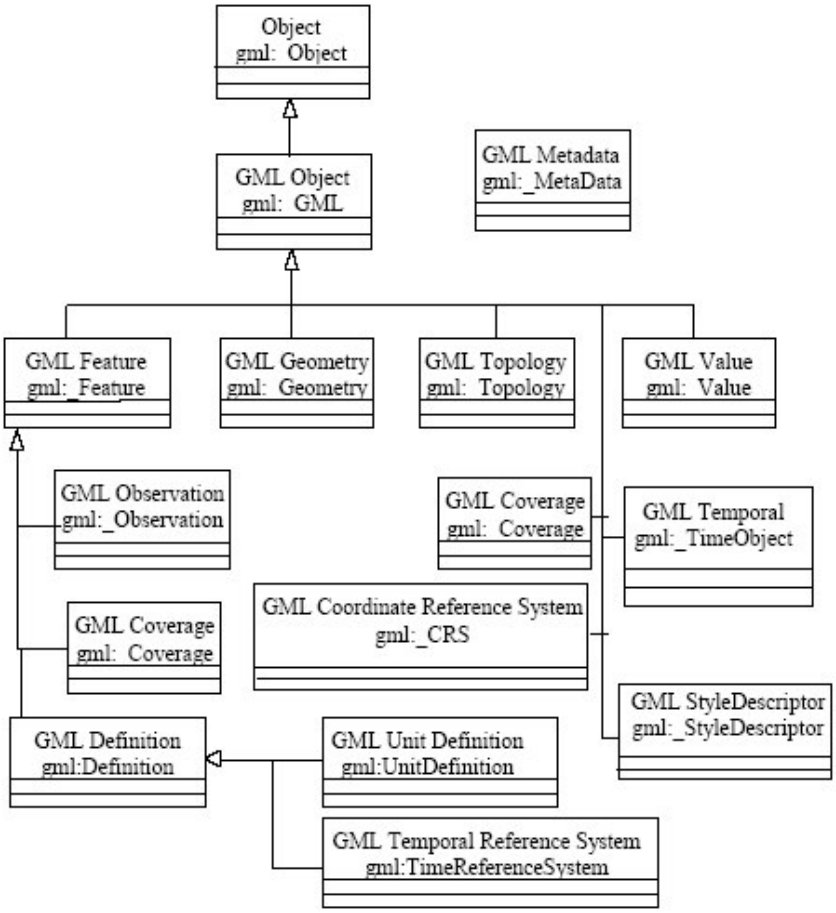


Figura A1.6 - Hierarquia de classes da GML 3.0 (OGC, 2003)

A1.4.1 Objetos Geográficos

O segundo esquema, chamado *FeatureSchema* (feature.xsd) é a base para a codificação dos fenômenos de interesse da modelagem, sejam eles geográficos (espaciais) ou não. Todo e qualquer objeto a ser representado em GML é derivado de feature. Assim, tendo ou não ele geometria, deve ser definido como um tipo de substituição de feature. A Figura A1.7 ilustra, em um diagrama UML, a hierarquia de *featureSchema*.

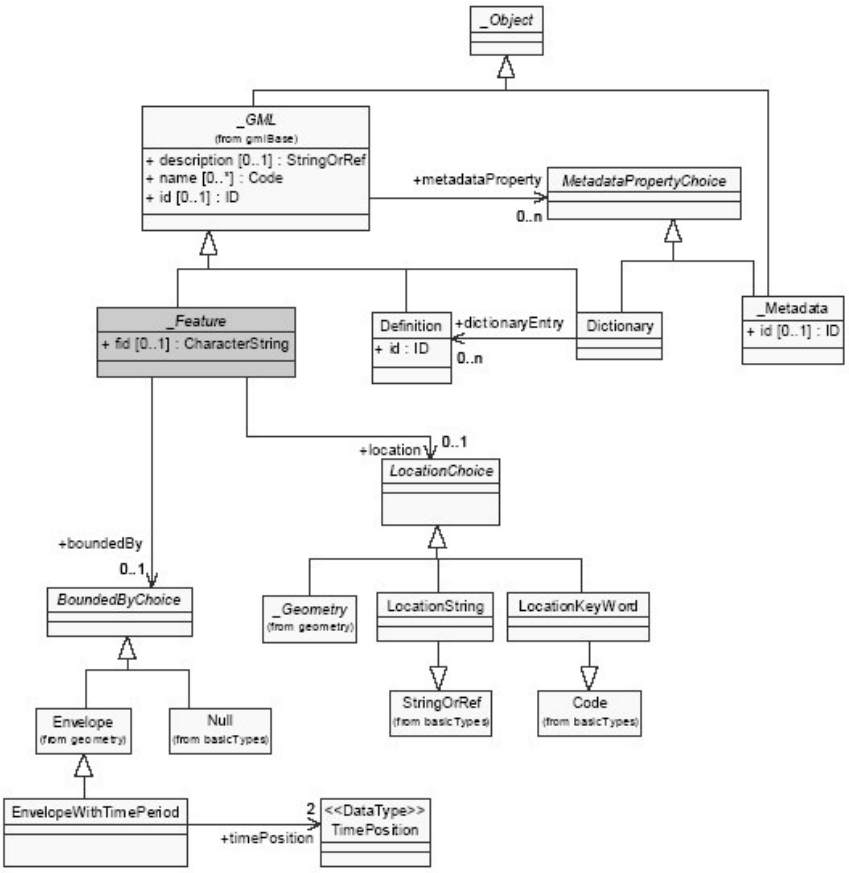


Figura A1.7 - Hierarquia de featureSchema (OGC, 2003)

A base para a definição que qualquer elemento que representa um fenômeno geográfico é *AbstractFeature*, representado no diagrama por "_Feature". Conforme pode ser observado, uma *feature* está associada a um elemento do tipo *BoundedBy*, que representa o mínimo retângulo envolvente (MBR) do objeto. Opcionalmente, por questões de compatibilidade com a versão anterior da GML, este envelope pode ser nulo. A geometria de uma *feature* é dada através da sua associação com *Location*. O esquema que representa a geometria será descrito na seção A1.4.3. Abaixo segue a definição do tipo que descreve *AbstractFeature*.

```
<complexType name="AbstractFeatureType" abstract="true">
  <complexContent>
    <extension base="gml:AbstractGMLType">
      <sequence>
        <element ref="gml:boundedBy" minOccurs="0"/>
        <element ref="gml:location" minOccurs="0"/>
      </sequence>
      <attribute name="id" type="string"/>
    </extension>
  </complexContent>
</complexType>
```

A1.4.2 Campos geográficos

A GML 3.0 introduz o conceito de campos geográficos, presente, por exemplo, no *framework* conceitual GeoFrame. Através da codificação de campos geográficos é possível descrever fenômenos contínuos do mundo real, cujo valor em cada ponto é dado por uma função de cobertura. Na GML um campo geográfico é uma *feature*, mas de um tipo especial, chamado *coverage* (esquema coverage.xsd). A Figura A1.8 apresenta a hierarquia de dos tipos da GML, em forma de classes da UML, para representação de *coverages*.

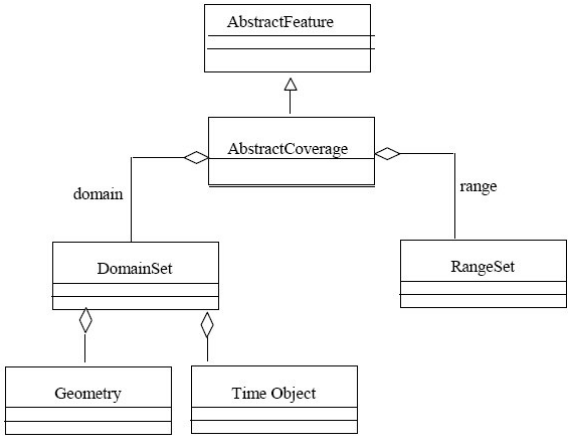


Figura A1.8 - Hierarquia de Coverage (OGC, 2003)

Conforme ilustrado, um campo geográfico (*coverage*) é uma especialização de *feature*, com as propriedades *DomainSet* e *RangeSet*. Há também outros dois componentes, chamado *coverageFuncion* e *dimension*.

- DomainSet: Especifica a região espaço-temporal de interesse, através de coordenadas espaciais e temporais, se for o caso;
- RangeSet: Possui o conjunto de valores a ser aplicado sobre a região de interesse;
- CoverageFunction: Especifica a função de cobertura a ser aplicada sobre a região de interesse;
- Dimension: Especifica o número de dimensões do campo geográfico.

A GML 3.0 suporta, ainda, a definição de *coverages* sobre atributos de uma *feature*. Desta forma, é possível codificar atributos espaço-variantes, como os existentes no modelo MADS apresentado.

A1.4.3 Geometrias

A geometria em GML é definida em *GeometrySchema* (geometry.xsd). A hierarquia deste esquema é apresentada na Figura A1.9.

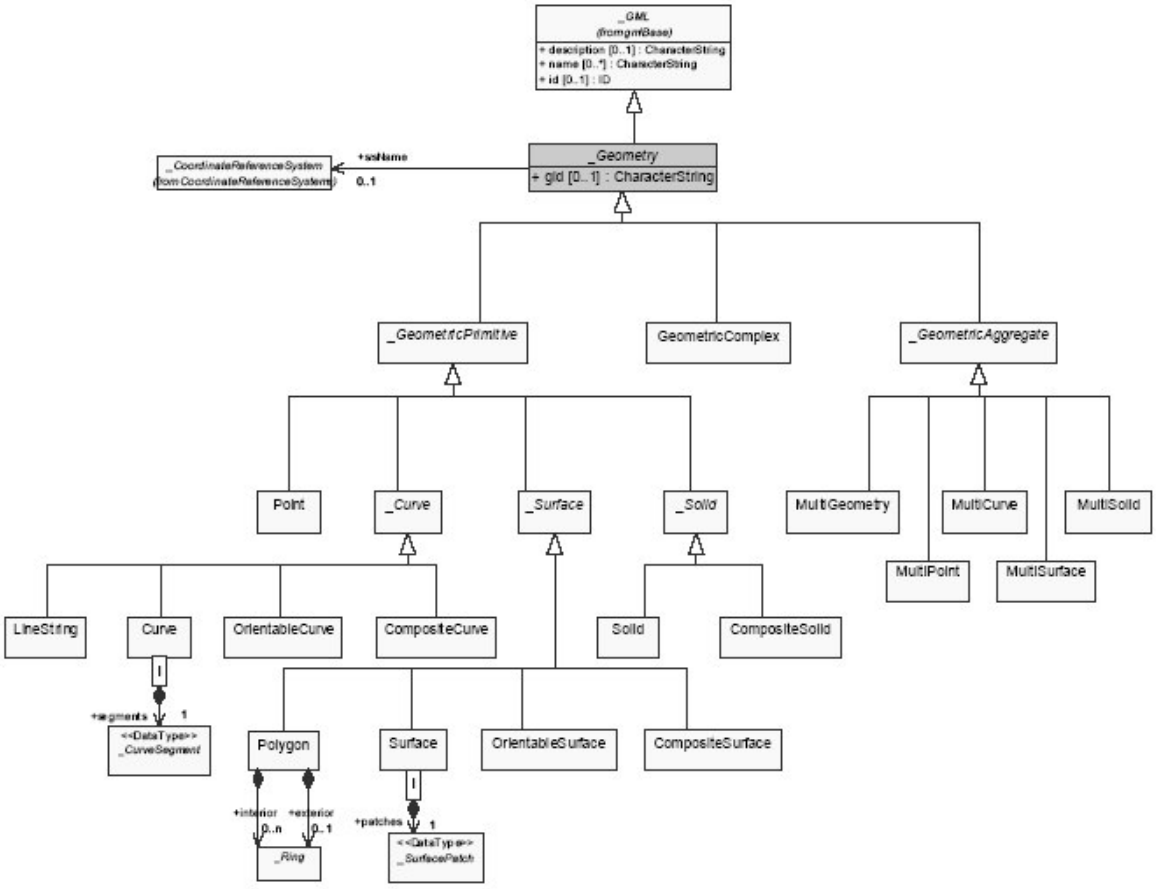


Figura A1.9 - Hierarquia de geometrias da GML 3.0 (OGC, 2003)

Na especificação 3.0 da GML, são contempladas geometrias de nenhuma, uma, duas ou três dimensões. Tanto a visão de campo quanto a de objetos pode ser representada pelas geometrias da GML, bem como objetos formados não por geometrias simples, mas através da agregação delas. Esta agregação pode ser homogênea ou de heterogêneas (geometrias diferentes).

- Ponto: Representa as geometrias dos objetos geográficos de nenhuma (zero) dimensão;
- Curva: Representa as geometrias dos objetos geográficos de uma dimensão. Pode ser uma curva, uma linha, uma linha dirigida ou ainda uma composição de curvas.
- Superfície:
- Sólido:
- Agregação:
- Geometrias complexas:

É no elemento de geometria que é definido o sistema de coordenadas a ser utilizado, bem como os próprios valores das coordenadas geográficas. Isto é feito através da referência a um elemento de um dos tipos de coordenadas que a GML 3.0 define: *coordinates*, *pos* ou *coord*.

Em termos de geometria para representação de campos geográficos (*coverages*) a GML ainda é bastante limitada. Existem tipos já definidos apenas para superfície de pontos, linhas e polígonos, e também para grades (*grids*) de células e de pontos.

A1.4.4 Topologia

A partir da especificação 3.0, a GML possui a capacidade de codificação de topologias de rede. Desta forma, é possível definir alguns tipos de relacionamentos espaciais entre objetos. Atualmente, o único relacionamento pré-definido na GML é o do tipo arco-nodo. Demais relacionamentos espaciais, utilizados para criação de restrições espaciais (cruzamento, adjacência, disjunção, etc.) não são suportados, estando isto previsto para a especificação 4.0 da linguagem.

Todo o esquema topológico da GML está definido nos esquemas topology.xsd e geometryComplexes.xsd, e seus componentes podem ser, assim como as geometrias, de zero, uma, duas ou três dimensões. A base é composta por nodos (0D) e arcos (1D), mas é possível aplicar-se superfícies (2D) e sólidos (3D) sobre qualquer um destes componentes.

A1.4.5 Temporalidade

Aspectos temporais são bastante importantes em termos de sistemas de informações geográficas, uma vez que os fenômenos modelados mudam com o passar do tempo. Desta forma, outra característica presente na GML a partir da especificação 3.0 é o suporte temporal, através do esquema temporal.xsd. Sua hierarquia é apresentada na Figura A1.10.

Os tipos de tempo suportados pela GML 3.0 são instante temporal e intervalo temporal, e qualquer registro temporal é baseado em algum sistema de referência temporal.

- TimeCoordinateSystem: Este é o sistema mais preciso. Utilizado para codificar datas absolutas, baseadas em algum calendário;
- TimeOrdinalReferenceSystem: Este sistema faz uma aproximação temporal, uma vez que codifica o tempo relativo a alguma data (antes de, depois de);
- TimeOrdinalEra: Este é o sistema mais impreciso. Utilizado para codificar datas relativas a algum período conhecido (ex.: idade média).

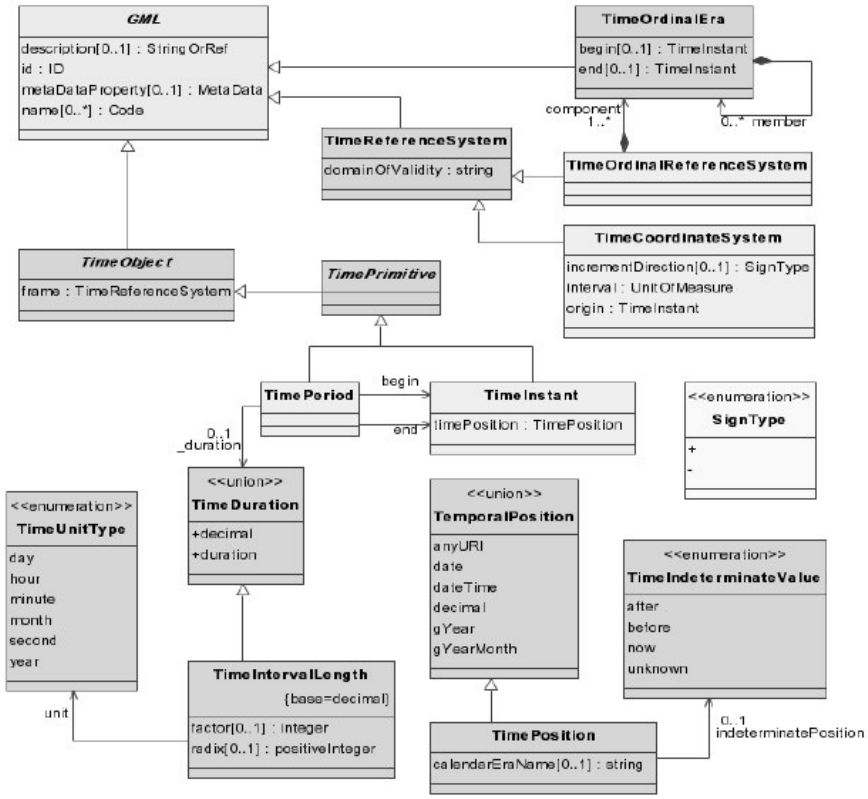


Figura A1.10 - Hierarquia temporal da GML 3.0 (OGC, 2003)

A1.4.6 Outros esquemas

A GML 3.0 define ainda uma série de outros esquemas a serem usados na codificação de aplicações geográficas.

- DynamicFeature: Tipo especial de elemento temporal, utilizado para monitoramento do comportamento e registros históricos de eventos de alguma *feature*;
- Definitions e Dictionaries: Definição de novos conceitos em GML necessários para a aplicação. Não define os elementos propriamente ditos, mas sim seus significados e descrições;
- Units: Esquema utilizado para definições de unidades de medidas a serem utilizadas na aplicação. Algumas já estão pré-definidas, mas é possível definir novas;
- Measures: Esquema da GML utilizado para definições dos tipos de medidas a serem utilizadas na GML, como volume, comprimento, peso, etc. Alguns já estão pré-definidos, mas é possível definir novos;

Anexo 2 – Ferramentas desenvolvidas

A2.1 Ferramenta de conversão UML-GeoFrame para GML 3.0

Um dos objetivos principais com esta ferramenta é tornar a conversão para GML transparente ao projetista da aplicação de SIG. Assim, o sistema é praticamente automatizado, interagindo com o usuário somente quando mais de uma opção de mapeamento é possível.

Para implementar as regras de mapeamento UML-GeoFrame para GML 2.1 foi desenvolvido um protótipo [XAV02]. Este programa foi desenvolvido em linguagem Visual Basic for Applications (VBA), como um *Add In* ao próprio Rational Rose, ambiente usado para modelagem conceitual. Este *script* foi alterado neste trabalho, para contemplar o poder de expressividade da GML 3.0. A utilização do Rational Rose e das suas potencialidades de desenvolvimento de *scripts* permitiu que o desenvolvimento do protótipo ocorresse de forme acelerada, visto que, além das funcionalidades nativas da linguagem Visual Basic, já possui alguns métodos específicos para tratamento dos construtores presentes no diagrama de classes.

Por outro lado, alguns construtores UML não possuem codificação em VBA. Especificamente, isto ocorre no caso dos construtores do tipo pacote. Uma vez que pacotes são elementos da arquitetura para ilustração, em alto nível, de relacionamentos entre sistemas e subsistemas, nenhum código pode ser gerado para eles. Isto prejudica o mapeamento do diagrama de classes para GML, visto que pacotes são construtores largamente usados, e o objetivo principal deste trabalho é o de preparar modelos conceituais reais para mineração na busca de identificação de candidatos à padrões de análise.

Para contornar esta limitação, foi concluído que é necessário alterar o modelo conceitual original desenvolvido pelo projetista. Deste modo, foi especificado um pré-tratamento do diagrama de classes, abaixo descrito:

- Cada construtor do tipo PACOTE existente no modelo conceitual dá origem a uma classe. Esta classe possui o mesmo nome do pacote, e, adicionalmente, um esterótipo TEMA, que é o que identifica que a classe em questão representa, na realidade, um pacote.
- A hierarquia de pacotes, ou seja, um tema é subtema de outro mais abrangente, e mapeada como sendo uma hierarquia simples de classes, todas elas identificadas como pacotes, através do estereótipo TEMA.
- Cada classe recebe um atributo adicional para cada pacote ao qual ela pertence. Este atributo também é identificado pelo estereótipo TEMA, que indica, neste caso, que

este atributo não é da classe, e, portanto não deve ser mapeado para GML, mas sim que a classe pertence a este pacote.

Idealmente, esta conversão deve ser feita de forma automática. O projetista entra com o seu modelo conceitual e a ferramenta transforma-o para que possa ser gerado o código GML. Contudo, por limitações de tempo e por não ser crucial no contexto deste trabalho, esta transformação é feita, no momento, de forma manual.

As Figuras A2.1 e A2.2 apresentam esta transformação de um diagrama de classes a fim de poder mapeá-lo corretamente da UML-GeoFrame para GML.

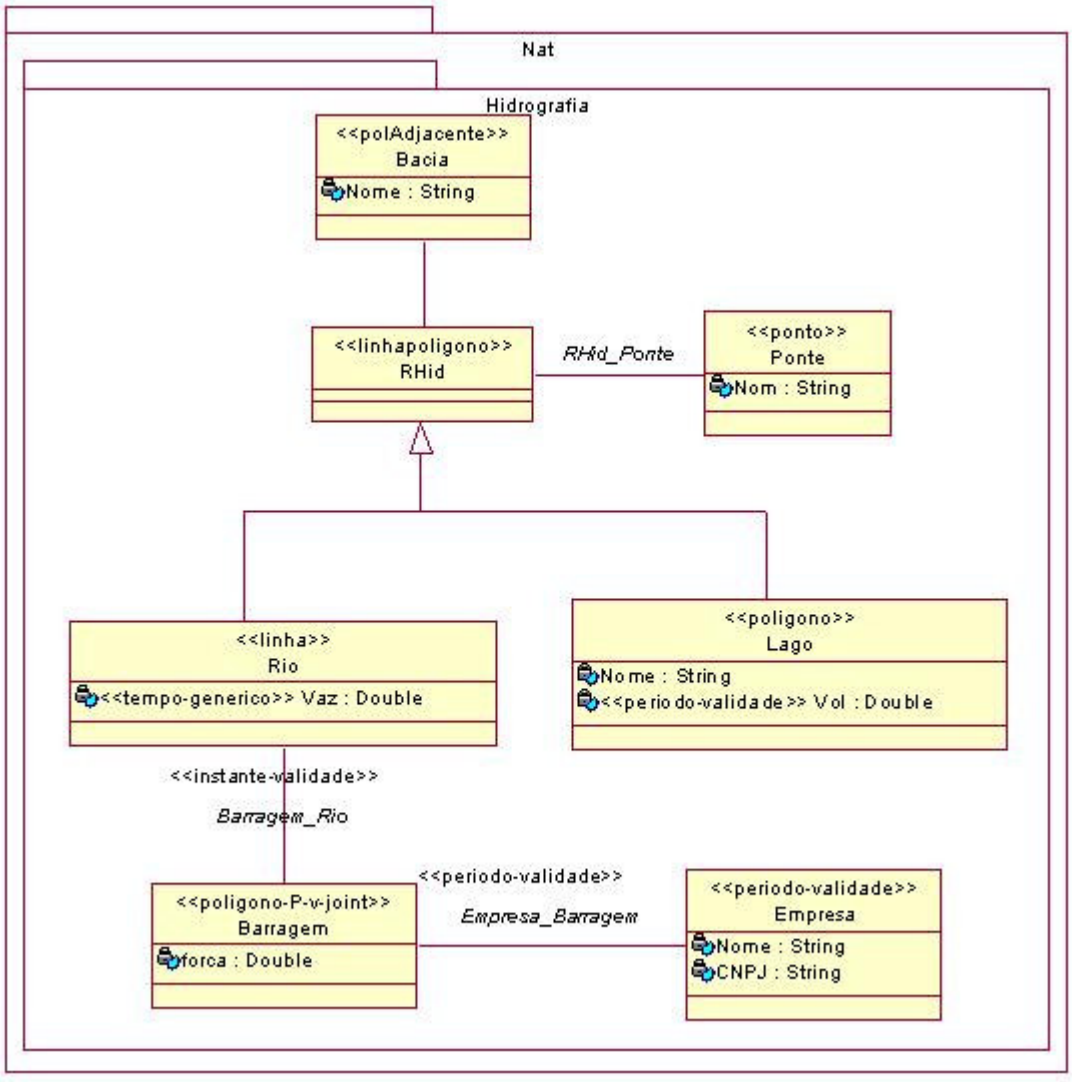


Figura A2.1 - Esquema conceitual original

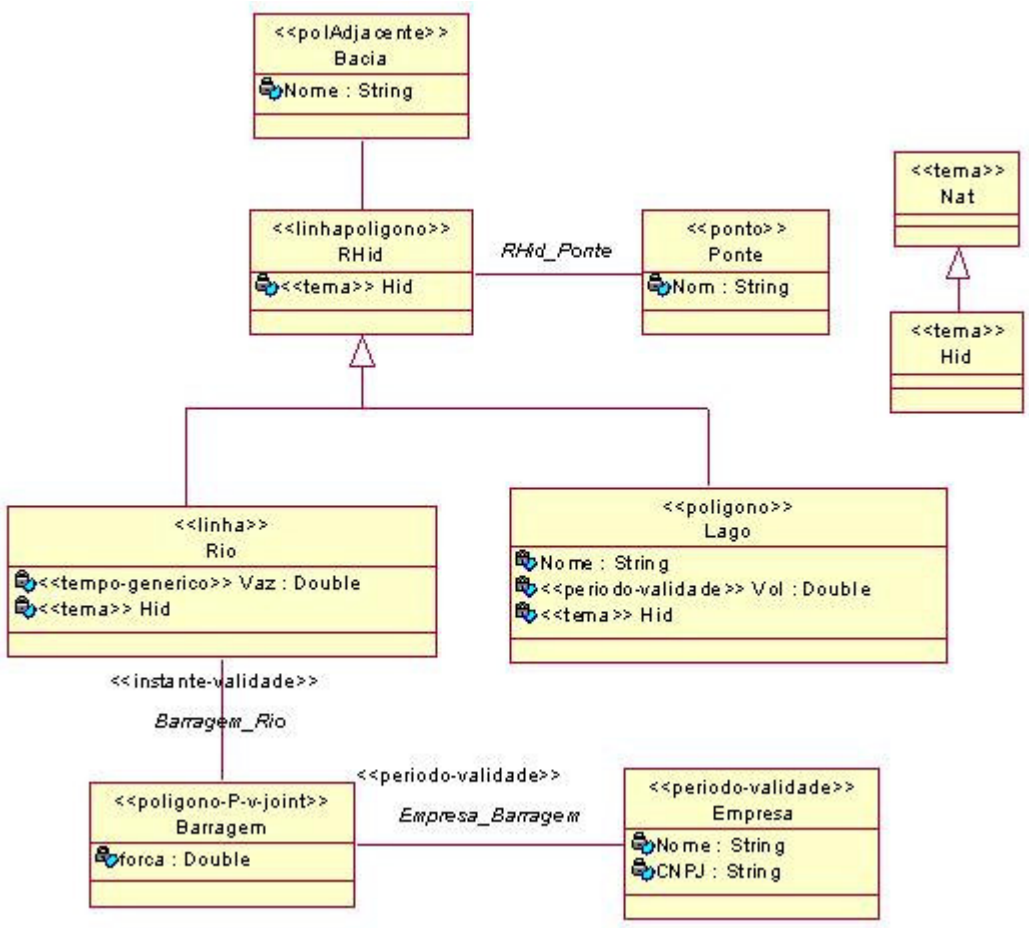


Figura A2.2 – Esquema conceitual Alterado

Uma vez tendo o diagrama de classes pronto para a conversão, o processo de mapeamento da UML-GeoFrame para GML pode ser iniciado. Nesta transformação, são aplicadas as regras descritas no capítulo 4.

A seqüência seguida na geração do código GML é apresentada a seguir, de forma resumida:

- Geração do cabeçalho do arquivo GML, contendo os *namespaces* utilizados, versão e demais informações a respeito do arquivo a ser gerado.
- Definição dos elementos representados no arquivo. Todos os pacotes, classes e associações são definidos nesta fase.
- Definições dos tipos complexos novos. Como especificado no capítulo 4, cada classe tem um tipo próprio. Estes tipos são criados nesta etapa, que também é responsável pela codificação das geometrias dos elementos existentes, tratamento de herança e associações.

A2.2 Ferramenta de Conversão MADS para GML 3.0

A conversão do esquema conceitual baseado no MADS para GML é realizada em duas etapas. Primeiramente, tendo o esquema conceitual construído graficamente através da ferramenta case MADSEditor (PARENT at. al, 1998), conforme ilustra a Figura A2.3, este

é transformado para um documento XML. Esta conversão é uma funcionalidade da própria ferramenta. O arquivo XML gerado não é genérico, segue uma definição específica do MADS.

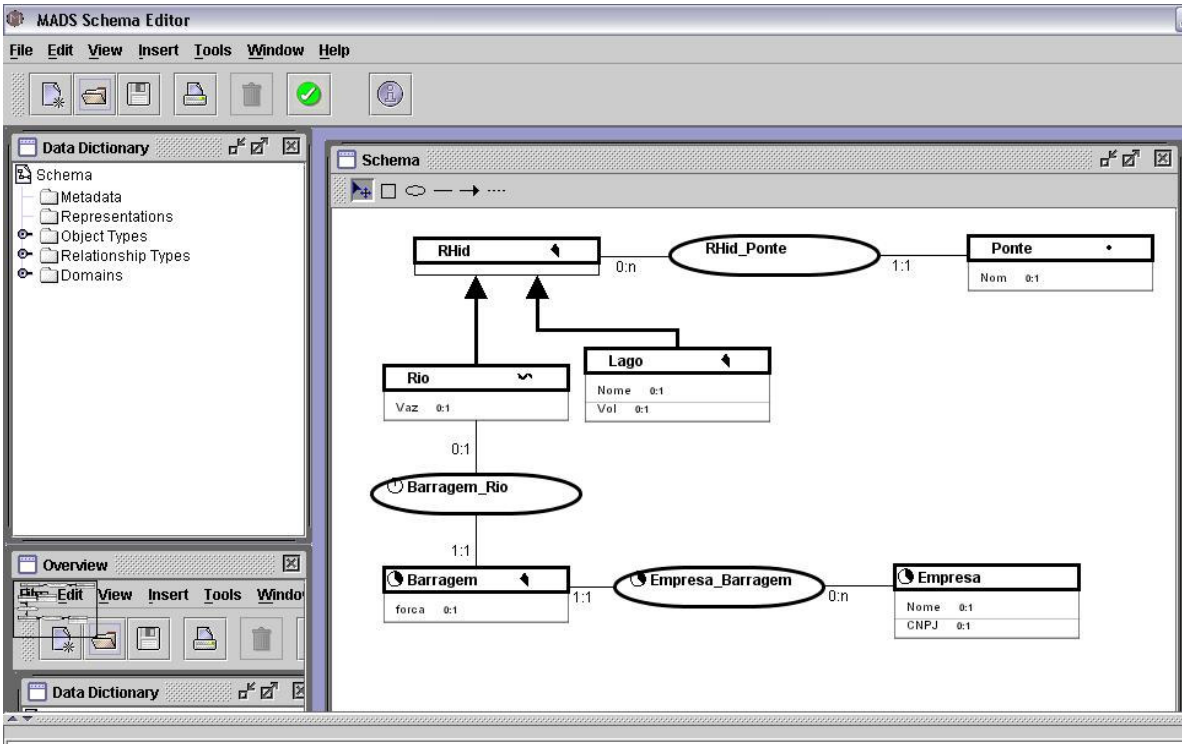


Figura A2.3 – MADS Editor

Este documento XML é processado por um parser XML, escrito em linguagem Java, que gera o como resultado código em GML 3.0. Este programa não possui uma interface gráfica. Os passos da criação do arquivo GML podem ser acompanhados pelo *prompt* de comando, conforme ilustrado na Figura A2.4.

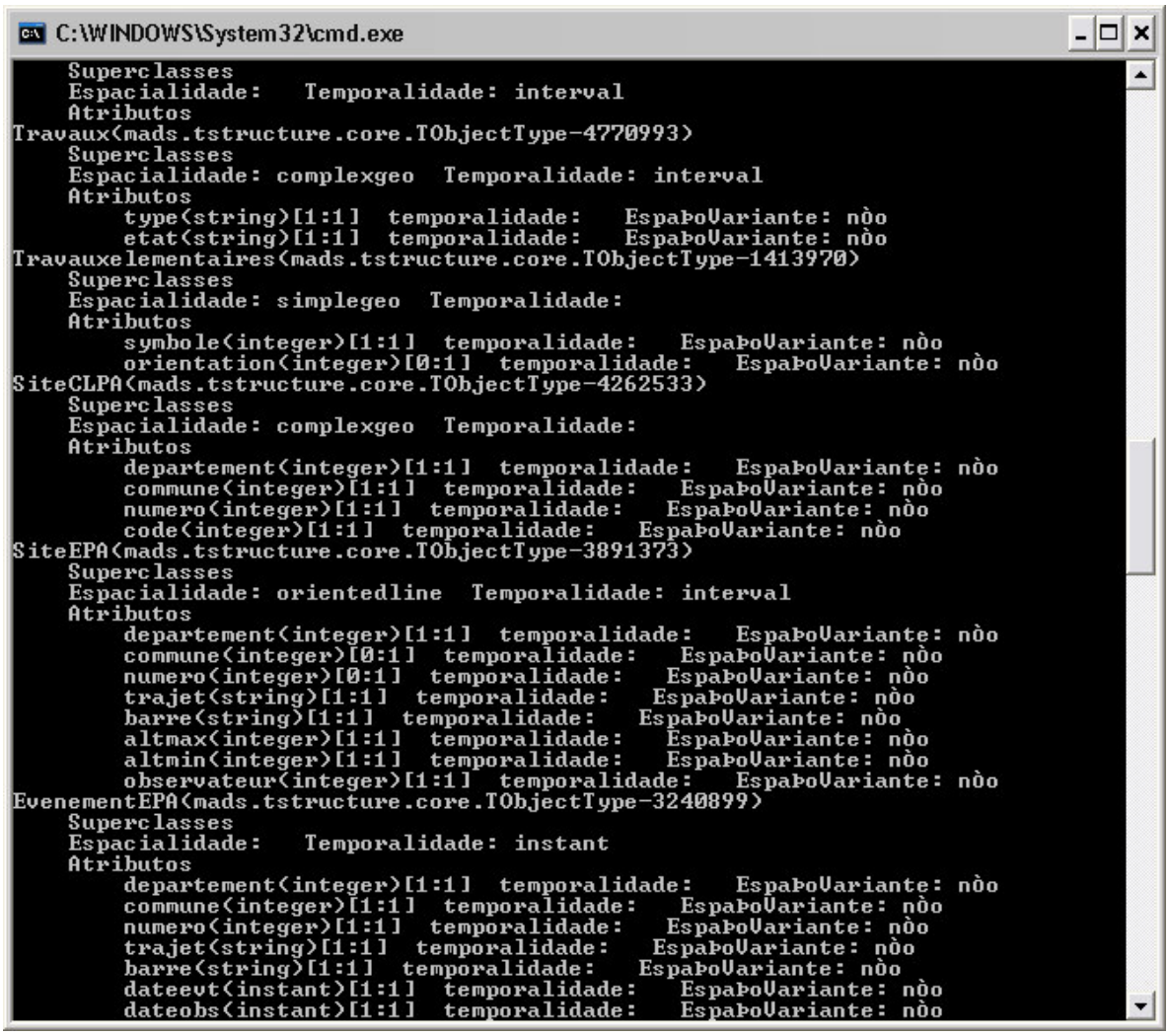


Figura A2.4 - Conversor MADS - GML

A2.3 Ferramenta de conversão GML para formato de dados de entrada suportado pelas ferramentas existentes

A ferramenta desenvolvida para a conversão de um arquivo GML para o formato de dados de entrada (FDE) objetiva gerar, automaticamente, um arquivo para ser utilizado como entrada na mineração de esquemas conceituais de BDG para obtenção de candidatos a padrões de análise. Esta ferramenta implementa as regras definidas no capítulo 4, e apresenta as seguintes funcionalidades:

- Espaço para visualização e edição de arquivos GML. É possível criar um arquivo novo, abrir um existente e salvar em disco.
- Espaço para visualização e edição de arquivos FDE. São arquivos planos, ou seja, ".txt" e é possível criar novos e salvar em disco.
- Verificação de arquivos GML. Esta funcionalidade é responsável por checar se o documento GML é bem formado, ou seja, se segue as regras de marcação XML. Não verifica se é um documento GML válido, apenas se é bem formado.

- Geração do arquivo FDE. A partir de um documento GML o programa faz a conversão para o formato de entrada de dados.

Este aplicativo foi desenvolvido em linguagem Java, utilizando o ambiente de programação Net Beans. A Figura A2.5 apresenta a interface do programa.

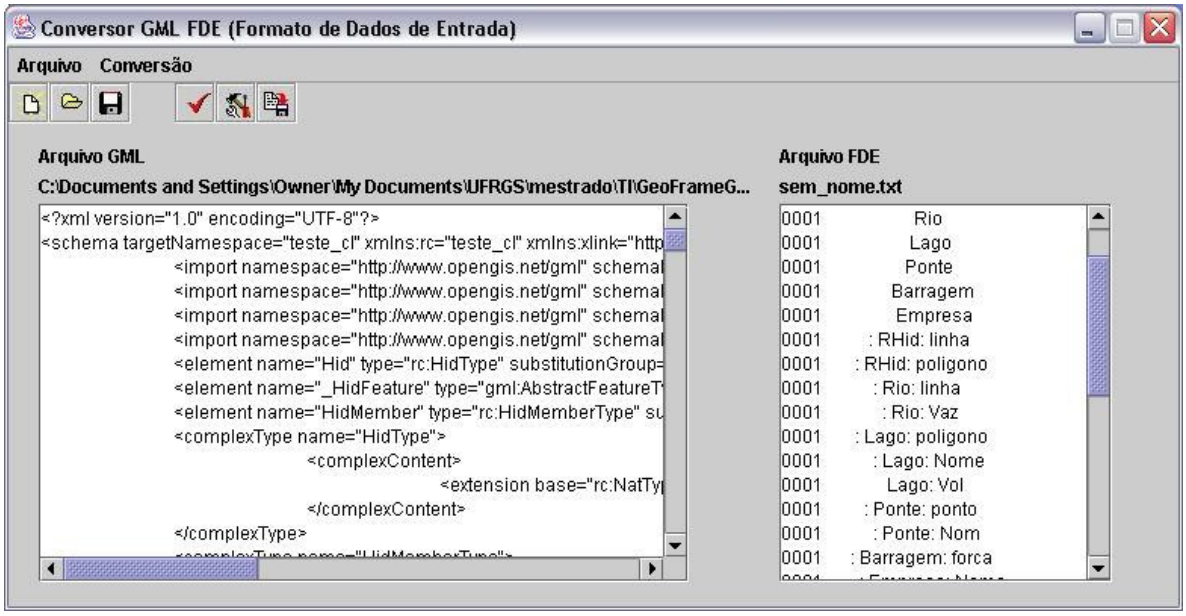


Figura A2.5 - Ferramenta de conversão GML-FDE

Na conversão GML para FDE, o algoritmo desenvolvido está dividido nas seguintes etapas:

- Verificação se o documento GML é bem formado.
- Varredura no documento GML para identificação dos construtores de interesse (classes, atributos, pacotes e associações).
- Geração de seis vetores, contendo os seis elementos diferentes suportados pelo FDE: Classe, Classe:Atributo, Pacote, Pacote:Classe, Pacote:Classe:Atributo, Classe#Classe#Associação. A geração destes vetores é feita conforme as regras descritas no capítulo 4.
- Escrita no arquivo FDE de destino do conteúdo dos cinco vetores acima especificados, na ordem que eles foram apresentados.
- Gravação do arquivo FDE em disco.

A2.4 Exemplo de Conversão do UML-GeoFrame

Para testar a correção das regras de tradução propostas, bem como da eficácia das ferramentas implementadas, foi realizado um estudo de caso. Neste teste, foi utilizado o diagrama de classes apresentado na Figura A2.2.

Após a execução do *script* de conversão UML-GeoFrame para GML, obteve-se um arquivo que foi salvo com o nome de teste_cl.xsd, validado no programa comercial XML Spy. A listagem deste arquivo segue abaixo.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="teste_cl" xmlns:rc="teste_cl" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:ggf="geometry_geoframe" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema">
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/feature.xsd"/>
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/coverage.xsd"/>
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/units.xsd"/>
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/gmlBase.xsd"/>
    <element name="Hid" type="rc:HidType" substitutionGroup="gml:_FeatureCollection"/>
    <element name="_HidFeature" type="gml:AbstractFeatureType" abstract="true"
substitutionGroup="gml:_Feature"/>
    <element name="HidMember" type="rc:HidMemberType" substitutionGroup="gml:featureMember"/>
    <complexType name="HidType">
        <complexContent>
            <extension base="rc:NatType"/>
        </complexContent>
    </complexType>
    <complexType name="HidMemberType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence minOccurs="0">
                    <element ref="rc:_HidFeature"/>
                </sequence>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </restriction>
        </complexContent>
    </complexType>
    <element name="Nat" type="rc:NatType" substitutionGroup="gml:_FeatureCollection"/>
    <element name="_NatFeature" type="gml:AbstractFeatureType" abstract="true"
substitutionGroup="gml:_Feature"/>
    <element name="NatMember" type="rc:NatMemberType" substitutionGroup="gml:featureMember"/>
    <complexType name="NatType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <complexType name="NatMemberType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence minOccurs="0">
                    <element ref="rc:_NatFeature"/>
                </sequence>
                <attributeGroup ref="gml:AssociationAttributeGroup"/>
            </restriction>
        </complexContent>
    </complexType>
    <element name="RHid" type="rc:RHidType" substitutionGroup="rc:_HidFeature"/>
    <complexType name="RHidType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:Curve"/>
                    <element ref="gml:Polygon"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <element name="RHidId" type="rc:idPropertyType" substitutionGroup="rc:RHid"/>
    <element name="Rio" type="rc:RioType" substitutionGroup="rc:_HidFeature"/>
    <complexType name="RioType">
        <complexContent>
            <extension base="rc:RHidType">
```

```
            <sequence>
                <group ref="rc:GroupVaz"/>
                <element ref="gml:Curve"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="RioId" type="rc:idPropertyType" substitutionGroup="rc:Rio"/>
<element name="Lago" type="rc:LagoType" substitutionGroup="rc:_HidFeature"/>
<complexType name="LagoType">
    <complexContent>
        <extension base="rc:RHidType">
            <sequence>
                <element name="Nome" type="string"/>
                <group ref="rc:GroupVol"/>
                <element ref="gml:Polygon"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="Ponte" type="rc:PonteType" substitutionGroup="gml:_Feature"/>
<complexType name="PonteType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element name="Nom" type="string"/>
                <element ref="gml:Point"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="PonteId" type="rc:idPropertyType" substitutionGroup="rc:Ponte"/>
<element name="Barragem" type="rc:BarragemType" substitutionGroup="gml:_Feature"/>
<complexType name="BarragemType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element name="forca" type="double"/>
                <element ref="gml:TimePeriod"/>
                <element ref="gml:Polygon"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="BarragemId" type="rc:idPropertyType" substitutionGroup="rc:Barragem"/>
<element name="Empresa" type="rc:EmpresaType" substitutionGroup="gml:_Feature"/>
<complexType name="EmpresaType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element name="Nome" type="string"/>
                <element name="CNPJ" type="string"/>
                <element ref="gml:TimePeriod"/>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="EmpresaId" type="rc:idPropertyType" substitutionGroup="rc:Empresa"/>
<element name="Bacia" type="rc:BaciaType" substitutionGroup="gml:_Feature"/>
<complexType name="BaciaType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element name="Nome" type="string"/>
                <element ref="gml:MultiSurfaceCoverage"/>
            </sequence>
```

```
            </extension>
        </complexContent>
    </complexType>
    <element name="BaciaId" type="rc:idPropertyType" substitutionGroup="rc:Bacia"/>
    <group name="GroupVaz">
        <sequence>
            <element name="Vaz" type="double"/>
            <element ref="gml:TimeInstant"/>
        </sequence>
    </group>
    <group name="GroupVol">
        <sequence>
            <element name="Vol" type="double"/>
            <element ref="gml:TimePeriod"/>
        </sequence>
    </group>
    <element name="RHid_Ponte" type="rc:RHid_PonteType" substitutionGroup="gml:_Feature"/>
    <complexType name="RHid_PonteType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence>
                    <element ref="rc:PonteId"/>
                    <element ref="rc:side"/>
                    <element ref="rc:RHidId"/>
                    <element ref="rc:side"/>
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
    <element name="Bacia_RHid" type="rc:Bacia_RHidType" substitutionGroup="gml:_Feature"/>
    <complexType name="Bacia_RHidType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence>
                    <element ref="rc:BaciaId"/>
                    <element ref="rc:side"/>
                    <element ref="rc:RHidId"/>
                    <element ref="rc:side"/>
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
    <element name="Barragem_Rio" type="rc:Barragem_RioType" substitutionGroup="gml:_Feature"/>
    <complexType name="Barragem_RioType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence>
                    <element ref="rc:RioId"/>
                    <element ref="rc:side"/>
                    <element ref="rc:BarragemId"/>
                    <element ref="rc:side"/>
                    <element ref="gml:TimeInstant"/>
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
    <element name="Empresa_Barragem" type="rc:Empresa_BarragemType" substitutionGroup="gml:_Feature"/>
    <complexType name="Empresa_BarragemType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence>
                    <element ref="rc:EmpresaId"/>
                    <element ref="rc:side"/>
                    <element ref="rc:BarragemId"/>
                    <element ref="rc:side"/>
                    <element ref="gml:TimePeriod"/>
```

```
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
    <element name="side" type="string"/>
    <element name="part" type="string"/>
    <element name="whole" type="string"/>
    <complexType name="idPropertyType">
        <sequence minOccurs="0">
            <element name="objId" type="gml:id"/>
        </sequence>
        <attributeGroup ref="xlink:simpleLink"/>
    </complexType>
</schema>
```

A2.5 Exemplo de Conversão do MADS

Para testar a correção das regras de tradução propostas, bem como da eficácia das ferramentas implementadas, foi realizado um estudo de caso. Neste teste, foi utilizado o diagrama de classes apresentado na Figura A2.2.

Após a execução do *script* de conversão do MADS para GML, obteve-se um arquivo que foi salvo com o nome de mads.xsd, validado no programa comercial XML Spy. A listagem deste arquivo segue abaixo.

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema targetNamespace="mads" xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:gml="http://www.opengis.net/gml" xmlns:ggf="geometry_geoframe"
    mlns:xlink="http://www.w3.org/1999/xlink" xmlns:rc="mads" elementFormDefault="qualified"
    version="2.06">
  <import namespace="http://www.opengis.net/gml"
  schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/feature.xsd" />
  <import namespace="http://www.opengis.net/gml"
  schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/coverage.xsd" />
  <import namespace="http://www.opengis.net/gml"
  schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/units.xsd" />
  <import namespace="http://www.opengis.net/gml"
  schemaLocation="http://schemas.opengis.net/gml/3.0.0/base/gmlBase.xsd" />
  <element name="Lago" type="rc:LagoType" substitutionGroup="gml:_Feature" />
  <complexType name="LagoType">
    <complexContent>
      <extension base="rc:RHidType">
        <sequence>
          <element ref="gml:Polygon" />
          <element name="Nome" type="string" />
          <group ref="rc:groupVolume" />
        </sequence>
       /extension>
    </complexContent>
  </complexType>
  <group name="groupVolume">
    <sequence>
      <element name="Volume" type="double" />
      <element ref="gml:TimeInstant" />
    </sequence>
  </group>
  <element name="Ponte" type="rc:PonteType" substitutionGroup="gml:_Feature" />
  <complexType name="PonteType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element ref="gml:Point" />
          <element name="Nom" type="string" />
```

```
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="RHid" type="rc:RHidType" substitutionGroup="gml:_Feature" />
<complexType name="RHidType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element ref="gml:Polygon" />
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="Rio" type="rc:RioType" substitutionGroup="gml:_Feature" />
<complexType name="RioType">
    <complexContent>
        <extension base="rc:RHidType">
            <sequence>
                <element ref="gml:Curve" />
                <group ref="rc:groupVaz" />
            </sequence>
        </extension>
    </complexContent>
</complexType>
<group name="groupVaz">
    <sequence>
        <element name="Vaz" type="double" />
        <element ref="gml:TimeInstant" />
    </sequence>
</group>
<element name="Empresa" type="rc:EmpresaType" substitutionGroup="gml:_Feature" />
<complexType name="EmpresaType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element ref="gml:TimePeriod" />
                <element name="Nome" type="string" />
                <element name="CNPJ" type="string" />
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="Barragem" type="rc:BarragemType" substitutionGroup="gml:_Feature" />
<complexType name="BarragemType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element ref="gml:Polygon" />
                <element ref="gml:TimePeriod" />
                <element name="forca" type="double" />
            </sequence>
        </extension>
    </complexContent>
</complexType>
<element name="RHid_Ponte" type="rc:RHid_PonteType" substitutionGroup="gml:_Feature" />
<element name="PonteId" type="rc:idPropertyType" substitutionGroup="rc:Ponte" />
<element name="RHidId" type="rc:idPropertyType" substitutionGroup="rc:RHid" />
<complexType name="RHid_PonteType">
    <complexContent>
        <restriction base="gml:FeaturePropertyType">
            <sequence>
                <element ref="rc:PonteId" minOccurs="0" maxOccurs="1" />
                <element ref="rc:RHidId" minOccurs="0" maxOccurs="1" />
            </sequence>
        </restriction>
```

```
        </complexContent>
    </complexType>
    <element name="Barragem_Rio" type="rc:Barragem_RioType" substitutionGroup="gml:_Feature" />
    <element name="RioId" type="rc:idPropertyType" substitutionGroup="rc:Rio" />
    <element name="BarragemId" type="rc:idPropertyType" substitutionGroup="rc:Barragem" />
    <complexType name="Barragem_RioType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence>
                    <element ref="rc:RioId" minOccurs="0" maxOccurs="1" />
                    <element ref="rc:BarragemId" minOccurs="0" maxOccurs="1" />
                    <element ref="gml:TimeInstant" />
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
    <element name="Empresa_Barragem" type="rc:Empresa_BarragemType"
        substitutionGroup="gml:_Feature" />
    <element name="EmpresaId" type="rc:idPropertyType" substitutionGroup="rc:Empresa" />
    <complexType name="Empresa_BarragemType">
        <complexContent>
            <restriction base="gml:FeaturePropertyType">
                <sequence>
                    <element ref="rc:BarragemId" minOccurs="0" maxOccurs="1" />
                    <element ref="rc:EmpresaId" minOccurs="0" maxOccurs="1" />
                    <element ref="gml:TimePeriod" />
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
    <complexType name="idPropertyType">
        <sequence minOccurs="0">
            <element name="objId" type="gml:id" />
        </sequence>
        <attributeGroup ref="xlink:simpleLink" />
    </complexType>
</schema>
```

A2.6 Exemplo de Geração do Arquivo FDE

Após a geração do arquivo GML, utilizou-se a segunda ferramenta construída para geração do arquivo no formato FDE. Como entrada foi passado o arquivo teste_cl.xsd (GeoFrame) e a saída, mostrada a seguir, foi salva como teste_cl.txt. Utilizando o arquivo mads.xsd (MADS) como entrada, a saída foi a mesma.

```
    0001                    *E
    0001                 Bacia
    0001                  RHid
    0001                   Rio
    0001                  Lago
    0001                 Ponte
    0001               Barragem
    0001                Empresa
    0001          : RHid: linha
    0001       : RHid: poligono
    0001           : Rio: linha
    0001             : Rio: Vaz
    0001       : Lago: poligono
    0001           : Lago: Nome
    0001              Lago: Vol
    0001         : Ponte: ponto
    0001           : Ponte: Nom
    0001      : Barragem: forca
```

```
0001         : Empresa: Nome
0001         : Empresa: CNPJ
0001                    Hid:
0001                    Nat:
0001               Hid: RHid
0001                Hid: Rio
0001               Hid: Lago
0001        Hid: RHid: linha
0001     Hid: RHid: poligono
0001         Hid: Rio: linha
0001           Hid: Rio: Vaz
0001     Hid: Lago: poligono
0001         Hid: Lago: Nome
0001          Hid: Lago: Vol
0001        RHid#Ponte#assoc
0001 Barragem#Empresa#assoc
0001     Barragem#Rio#assoc
```