FILIPE RIBEIRO NALON

ADEQUAÇÃO DE UM PERFIL UML PARA MODELAGEM CONCEITUAL DE BANCOS DE DADOS GEOGRÁFICOS AOS PADRÕES ISO E OGC USANDO MDA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

VIÇOSA
MINAS GERAIS - BRASIL
2010

FILIPE RIBEIRO NALON

ADEQUAÇÃO DE UM PERFIL UML PARA MODELAGEM CONCEITUAL DE BANCOS DE DADOS GEOGRÁFICOS AOS PADRÕES ISO E OGC USANDO MDA

Dissertação apresentada à Universidade Federal de Viçosa, como parte das exigências do Programa de Pós-Graduação em Ciência da Computação, para obtenção do título de *Magister Scientiae.*

APROVADA: 30 de agosto de 2010.

Karla Albuquerque de V. Borges (Co-Orientador)	José Luís Braga (Co-Orientador)
Alcione de Paiva Oliveira	José Marinaldo Gleriani

Jugurta Lisboa Filho
(Orientador)

AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus por me sustentar até aqui e por me dar condições de realizar este trabalho. A Ele todo o louvor e glória.

Agradeço também aos meus pais, Alberto e Márcia, que com amor e dedicação me criaram e sempre me incentivaram com relação aos estudos, não medindo esforços para que chegasse até aqui.

Agradeço à Laís, que esteve sempre ao meu lado durante a realização deste trabalho. Seu amor, carinho e compreensão foram muito importantes durante este período. Um agradecimento especial ao Marco, Dorinha, Calebe, Camila e D. Maria, que me acolheram de forma bastante carinhosa e sempre me apoiaram também.

Aos amigos da república "Os 300 de Viçosa", pela excelente convivência durante esse tempo e pelos momentos descontraídos e aos irmãos da Igreja Cristã Maranata, com os quais também compartilho essa conquista.

Um agradecimento em especial ao meu orientador Jugurta Lisboa Filho, com o qual pude aprender muito. Seus ensinamentos e conselhos foram fundamentais para que conseguisse realizar este trabalho. Agradeço também aos meus co-orientadores, o professor José Luís Braga e a pesquisadora Karla A. V. Borges, como também aos demais professores do DPI, com os quais pude aprender muito durante a graduação e o mestrado.

Agradeço à CAPES, pelo apoio financeiro e à Universidade Federal de Viçosa.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

ATL - Atlas Transformation Language

BDGeo - Banco de Dados Geográficos

CASE - Computer-Aided Software Engineering

CIM - Computation Independent Model

CORBA - Common Object Request Broker Architecture

EAI - Enterprise Application Integration

EDOC - Enterprise Distributed Object Computing

EPL - Eclipse Public License

ER - Entidade-Relacionamento

GeoOOA - Geographic Object-Oriented Analysis

GPL - General Public License

ISO - International Organization for Standardization

MADS - Modeling for Application Data with Spatio-temporal features

MDA - Model Driven Architecture

MOF - Meta Object Facility

OCL - Object Constraint Language

OGC - Open Geospatial Consortium

OMG - Object Management Group

OMT-G - Object Modeling Technique for Geographic Applications

OOSE - Object-Oriented Software Engineering

PIM - Platform Independent Model

PSM - Platform Specific Model

RSM - Rational Software Modeler

SFA - OpenGIS Simple Feature Access

SGBD - Sistema de Gerenciamento de Banco de Dados

SIG - Sistemas de Informação Geográfica

SQL - Structure Query Language

TC - Technical Committee

UML - Unified Modeling Language

W3C - World Wide Web Consortium

XMI - XML Metadata Interchange

XML - eXtensible Markup Language

RESUMO

NALON, Filipe Ribeiro, M.Sc., Universidade Federal de Viçosa, agosto de 2010. **Adequação de um perfil UML para modelagem conceitual de bancos de dados geográficos aos padrões ISO e OGC usando MDA**. Orientador: Jugurta Lisboa Filho. Co-orientadores: José Luís Braga e Karla Albuquerque de Vasconcelos Borges.

Nos últimos 20 anos, diversos modelos conceituais de dados específicos para modelagem de *Sistemas de Informação Geográfica* (SIG) foram propostos. Porém, ainda não há um modelo de consenso, o que tem gerado vários problemas para a área de SIG, como a falta de interoperabilidade entre ferramentas CASE que dão suporte a estes modelos. Um perfil UML, chamado GeoProfile, foi proposto para padronizar a tarefa de modelagem de dados geográficos. O GeoProfile apresenta características dos principais modelos existentes, procurando, dessa forma, dar suporte a todos os requisitos para modelagem de aplicações geográficas. Porém, não foi levada em consideração na sua versão inicial a utilização de padrões internacionais da área. Este trabalho mostra a integração do GeoProfile com os padrões internacionais publicados por organizações como a *International Organization for Standardization* (ISO) e *Open Geospatial Consortium* (OGC), os quais estão relacionados à informação geográfica. Como o GeoProfile é usado em um nível de abstração mais alto, esta integração é apresentada através dos diferentes níveis de abstração de modelos da abordagem *Model Driven Architecture* (MDA). Os padrões internacionais atuam em nível de abstração mais baixo que o GeoProfile. Com a integração feita, foi possível realizar a transformação de modelos de forma automatizada, utilizando a linguagem de transformação de modelos *Atlas Transformation Language* (ATL).

ABSTRACT

NALON, Filipe Ribeiro, M.Sc., Universidade Federal de Viçosa, August of 2010. **Adequacy of a UML profile for geograpic databases conceptual modeling to ISO and OGC Standards using MDA**. Adviser: Jugurta Lisboa Filho. Co-Advisers: José Luís Braga and Karla Albuquerque de Vasconcelos Borges.

In the last 20 years, several conceptual data models specific for *Geographic Information Systems* (GIS) have been proposed. However, so far there isn't a consensus model, which has generated several problems for the GIS area, such as the lack of interoperability among CASE tools that give support to these models. A UML profile, called GeoProfile, was proposed to standardize the task of geographic data modeling. The GeoProfile presents characteristics of the main existent models, thus trying to give support to all the requirements for geographic applications modeling. However, it was not taken into account in initial version the use of international standards of the area. This work shows the integration of the GeoProfile with the international standards published by organizations as *International Organization for Standardization* (ISO) and *Open Geospatial Consortium* (OGC), which are related to the geographic information. Since the GeoProfile is used in a higher abstraction level, this integration is presented through the different abstraction levels of the *Model Driven Architecture* (MDA) approach models. The international standards act in a lower abstraction level than the GeoProfile. With the integration done, it was possible to transform the models in an automated way, using the *Atlas Transformation Language* (ATL).

1 Introdução

A atividade de desenvolvimento de *software* é uma tarefa que requer, cada vez mais, o uso de metodologias e técnicas padronizadas e amplamente conhecidas. Até pouco tempo atrás, a principal preocupação dos desenvolvedores de *software* estava relacionada à estrutura e qualidade dos códigos-fonte produzidos em determinada linguagem de programação. Atualmente, a principal preocupação está em entender bem o domínio do problema, a fim de se gerar soluções que atendam às reais necessidades dos clientes e usuários finais.

Para auxiliar nessa tarefa de compreensão do problema, a principal técnica utilizada é a modelagem. Um modelo é uma simplificação da realidade. Modelos são utilizados em várias áreas como, por exemplo, na indústria automobilística, na construção civil, na construção de dispositivos eletro-eletrônicos, entre outras. No desenvolvimento de sistemas de *software*, modelos são construídos para compreender melhor o sistema que está sendo desenvolvido (BOOCH; RUMBAUGH; JACOBSON, 2005).

Uma das técnicas que surgiram recentemente e que possibilita o desenvolvimento de sistemas usando modelos em diferentes níveis de abstração é a abordagem MDA (*Model Driven Architecture*) (OMG, 2003). Segundo OMG (2003), inicialmente são construídos modelos de níveis de abstração mais altos, os quais, posteriormente, passam por transformações, até atingir detalhes específicos de implementação. Esses diferentes níveis de abstração, conforme especificado em OMG (2003), são: Modelo Computacionalmente Independente (CIM ou *Computation Independent Model*), Modelo Independente de Plataforma (PIM ou *Platform Independent Model*) e Modelo de Plataforma Específica (PSM ou *Platform Specific Model*). A abordagem MDA pode ser utilizada também para a modelagem de bancos de dados geográficos como sugere, por exemplo, Bédard e Larrivée (2008).

No projeto de banco de dados, a utilização de modelos ajuda a descrever os dados sem se preocupar com detalhes de implementação. No projeto clássico de banco de dados (ELMASRI; NAVATHE, 2003), o modelo de nível de abstração mais alto é chamado de modelo conceitual. A realização da modelagem conceitual é feita com o auxílio de linguagens próprias de modelagem, que são linguagens cuja sintaxe e semântica têm seu foco voltado para a representação conceitual e física de

um sistema (FUENTES; VALLECILLO, 2004). Atualmente, uma das linguagens mais usadas e aceitas é a UML (*Unified Modeling Language*), que é extensível, para atender alguns domínios específicos, utilizando, para isso, o mecanismo chamado perfil.

Um domínio de aplicação que vem recebendo bastante atenção na atualidade é o de aplicações geográficas, devido à maior disponibilidade de dados espaciais, e cujos sistemas possuem características particulares que precisam ser levadas em consideração no desenvolvimento de tais aplicações.

Nos últimos 20 anos, várias pesquisas foram realizadas visando criar ou adaptar modelos conceituais de dados para aplicações geográficas. Porém, o surgimento desses modelos trouxe um problema para a área, que é a falta de um padrão de modelagem. Ferramentas foram criadas para os diversos modelos e há dificuldade de se obter interoperabilidade entre as soluções criadas, tornando, dessa forma, inviável o reuso de soluções em outros projetos. Além disso, há certos requisitos da modelagem de aplicações geográficas que uns modelos suportam e outros não.

Para a padronização desses modelos, foi especificado o GeoProfile (SAMPAIO, 2009). GeoProfile é um perfil UML para modelagem conceitual de bancos de dados geográficos (BDGeo) que reúne as características dos principais modelos conceituais existentes. O uso de perfis UML permite a extensão da sintaxe e semântica dos elementos da UML e possibilita o aproveitamento de toda a estrutura da UML como, por exemplo, de ferramentas CASE (*Computer-Aided Software Engineering*). Dessa forma, não é necessário criar ferramentas específicas para o modelo.

Como parte do esforço para a padronização da informação geográfica, algumas organizações, como a ISO (*International Organization for Standardization*) e OGC (*Open Geospatial Consortium*), têm publicado padrões internacionais para auxiliar na construção de aplicações geográficas padronizadas.

1.1 Motivação

Na especificação inicial do GeoProfile não foi levada em consideração a utilização de notação gráfica para os estereótipos, a qual torna a modelagem mais intuitiva para o projetista, nem os padrões internacionais publicados pelas

organizações ISO e OGC, as quais têm realizados esforços para a padronização da informação geográfica.

Como citado por Brodeur e Badard (2008), alguns dos padrões da série ISO 19100 podem contribuir com a modelagem conceitual de BDGeo, porém eles carregam alguns detalhes técnicos que não seriam interessante tê-los em um modelo mais próximo do usuário, como foi a proposta do GeoProfile. No entanto, a correspondência entre os elementos do GeoProfile e os padrões ISO e OGC pode ser feita e transformações entre esses modelos poderão ser executadas, buscando, dessa forma, aproximar o modelo de detalhes de implementação.

Utilizando a abordagem MDA, será possível adequar o GeoProfile aos padrões ISO/OGC. Dessa forma, ficará mais fácil, posteriormente, transformar um modelo enriquecido com os padrões ISO/OGC em modelos de plataformas específicas. Além disso, a adequação aos padrões internacionais viabilizará a aceitação do GeoProfile pela comunidade científica e pelos projetistas de BDGeo.

1.2 Objetivos

O objetivo deste trabalho é dar continuidade à especificação do perfil UML GeoProfile, proposto inicialmente por Sampaio (2009), e adequá-lo aos padrões internacionais publicados por organizações como ISO e OGC, utilizando, para isso, a abordagem MDA. Especificamente, pretende-se:

a) Estudar o perfil UML GeoProfile, proposto para a modelagem conceitual de BDGeo e propor notação gráfica para os estereótipos;

b) Implementar o GeoProfile em outras ferramentas CASE atuais que dão suporte a perfis UML;

c) Investigar os padrões internacionais propostos pelas organizações ISO e OGC que podem ser utilizados na modelagem de BDGeo;

d) Comparar e adequar o GeoProfile com as especificações de padronização especificadas pelo Comitê Técnico ISO/TC 211 e pelo OGC;

e) Estabelecer regras de transformação de esquemas conceituais de dados elaborados com base no GeoProfile, para esquemas compatíveis com as especificações ISO/TC 211 e OGC utilizando a abordagem MDA.

1.3 Estrutura da dissertação

No Capítulo 2 é apresentado um referencial teórico, cujos temas são utilizados no decorrer da dissertação. São descritos, por exemplo, conceitos básicos sobre modelagem conceitual de BDGeo, os mecanismos de extensão da UML, o conceito de perfil UML e a abordagem MDA.

No Capítulo 3 são descritos os padrões internacionais publicados por organizações como a ISO e OGC, os quais podem ser utilizados na modelagem de BDGeo. São descritos apenas alguns desses padrões que serão utilizados nesta dissertação.

No Capítulo 4 é descrita a estrutura do GeoProfile proposta por Sampaio (2009), como também é apresentada a notação gráfica proposta para os estereótipos e uma análise da implementação do GeoProfile em algumas ferramentas CASE.

No Capítulo 5 é realizada a integração do GeoProfile com os padrões internacionais, mostrando a diferença de níveis de abstração entre ambos, utilizando a abordagem MDA. É apresentado, também, um exemplo de transformação de modelos com a linguagem ATL.

Já no Capítulo 6 são apresentadas as conclusões e propostos alguns trabalhos futuros.

Ao final, também são apresentados alguns apêndices. No apêndice A é apresentado um breve guia prático sobre como criar e utilizar o GeoProfile na ferramenta CASE *Rational Software Modeler* (RSM). No apêndice B é descrita a lista completa de padrões publicados pelo Comitê Técnico ISO/TC 211. E no apêndice C é apresentado o código implementado na linguagem ATL para transformações dos exemplos de modelos mostrados no capítulo 5.

2 Modelos e padrões de modelagem de dados

Neste capítulo são abordadas as técnicas que podem ser utilizadas na modelagem conceitual de banco de dados geográficos (BDGeo). São descritos os conceitos básicos de modelagem conceitual de BDGeo, assim como os principais modelos existentes, os mecanismos de extensão da UML, bem como o conceito de perfil UML, que foi usado em Sampaio (2009) para a construção do GeoProfile. É descrito também a abordagem MDA, que utiliza modelos em diferentes níveis de abstração para construção de aplicações, assim como o formato padrão para intercâmbio de modelos, o XML *Metadata Interchange* (XMI).

2.1 Modelagem conceitual de banco de dados geográficos

Um sistema de informação pode ser definido tecnicamente como um conjunto de componentes inter-relacionados que coleta (ou recupera), processa, armazena e distribui informações destinadas a apoiar a tomada de decisões, a coordenação e o controle de uma organização. Além de dar suporte à tomada de decisões, à coordenação e ao controle, esses sistemas também auxiliam os gerentes e trabalhadores a analisar problemas, visualizar assuntos complexos e criar novos produtos (LAUDON; LAUDON, 2007).

"O termo *Sistemas de Informação Geográfica* (SIG) é aplicado a sistemas que realizam um tratamento computacional de dados geográficos. A principal diferença de um SIG para um sistema de informação convencional é sua capacidade de armazenar tanto os atributos descritivos como as geometrias dos diferentes tipos de dados geográficos" (CASANOVA et al., 2005).

A utilização de SIG cresceu e continua crescendo rapidamente em todo o mundo, dados os avanços de *hardware* e *software* e o acesso cada vez mais facilitado a essas tecnologias. O uso de SIG traz inúmeros benefícios ao seu usuário, uma vez que possibilita um melhor gerenciamento de informações e consequente melhoria nos processos de tomada de decisões em áreas de grande complexidade como planejamento municipal, estadual e federal, proteção ambiental, rede de utilidade pública, agricultura, exploração de petróleo, etc.

Worboys e Duckhan (2004) destacam que entre os principais componentes de um SIG está o componente de armazenamento, que é o banco de dados geográficos

(BDGeo). Sua função é estruturar e armazenar os dados de forma a possibilitar a realização das operações de análise envolvendo dados espaciais.

BDGeo pertence à categoria dos bancos de dados não convencionais. Os dados nele tratados possuem, além dos atributos descritivos, uma representação geométrica no espaço geográfico; esses dados são conhecidos como dados geográficos.

Segundo Lisboa Filho e Iochpe (1999), devido à complexidade das aplicações que são desenvolvidas a partir de um SIG, um dos grandes problemas no desenvolvimento desses sistemas tem sido projetar o BDGeo.

O projeto do banco de dados requer o uso de diferentes instrumentos, uma vez que as atividades necessárias à sua elaboração variam de acordo com a complexidade do sistema, com o tipo de pessoal envolvido, o sistema de gerenciamento de banco de dados (SGBD) utilizado, etc. Desta forma, o desenvolvimento de sistemas de banco de dados deve estar baseado em uma metodologia, a partir da qual são empregados instrumentos específicos de apoio às diferentes etapas do projeto (LISBOA FILHO; IOCHPE, 1999).

É comum na comunidade de banco de dados distinguirem o processo de projeto de banco de dados em três etapas: projeto conceitual, projeto lógico e projeto físico (ELMASRI; NAVATHE, 2003), sendo que na fase de projeto conceitual é elaborado o esquema conceitual do banco de dados, com base em modelos conceituais, que são modelos que fornecem construtores de abstração de alto nível para descrever os requisitos de dados da aplicação, sem se preocupar com detalhes de implementação. Essa fase é apresentada por Elmasri e Navathe (2003) como uma das fases mais importantes no planejamento de uma aplicação de banco de dados bem-sucedida.

A modelagem conceitual apresenta diversas vantagens para modelagem de aplicações geográficas. Primeiro, por facilitar a execução do projeto lógico, o qual necessita atender às particularidades de um SIG específico. Os usuários podem expressar seus conhecimentos sobre a aplicação usando conceitos que estão mais próximos a eles sem a necessidade de utilizar expressões computacionais. Como a modelagem conceitual independe do *software* no qual o sistema é implementado, o projeto resultante se mantém válido caso ocorram mudanças de tecnologia. Neste caso, apenas a transformação entre os esquemas conceitual e lógico é afetada. No caso da tecnologia de SIG, isso se torna um fator muito importante, uma vez que

grandes investimentos são preservados e há uma redução de custos e aumento das chances de sucesso em caso de mudança para tecnologias mais modernas. Por último, a modelagem conceitual facilita a troca de informações entre parceiros de diferentes organizações, uma vez que aumenta a capacidade de entendimento da semântica da informação, facilitando o uso correto da mesma (PARENT; SPACCAPIETRA; ZIMÁNYI, 1999).

Um dos princípios da modelagem conceitual é que um esquema conceitual deve conter apenas os elementos do domínio, desconsiderando os aspectos de implementação. O processo de modelagem conceitual de banco de dados compreende a descrição e definição dos possíveis conteúdos dos dados, além de estruturas e de regras a eles aplicáveis (LISBOA FILHO et al., 2000).

No caso de BDGeo, as particularidades da natureza da informação geográfica provocou o desenvolvimento de soluções específicas para a modelagem de dados geográficos. Segundo Chrisman (1997), a informação geográfica possui três componentes básicos: atributo, espaço e tempo, os quais possibilitam responder, respectivamente, a três perguntas: o que? onde? quando?.

O componente espacial descreve a localização geográfica e a forma geométrica do fenômeno descrito pela informação geográfica, além de relacionamentos com outros fenômenos geográficos. O componente atributo, também conhecido por atributo descritivo ou atributo não espacial, descreve as características não espaciais de um fenômeno geográfico (LISBOA FILHO et al., 2000).

Todo fenômeno geográfico é eminentemente temporal, ou seja, está associado a um instante ou intervalo de tempo em que este ocorre ou em que é observado (PEUQUET; DUAN, 1995 *apud* LISBOA FILHO et al., 2000). O componente tempo pode ser crítico para a informação geográfica, dependendo do tipo de fenômeno e do tipo de aplicação em que este está sendo utilizado.

Friis-Christensen et al. (2001) fizeram um levantamento dos requisitos de modelagem de dados geográficos, os quais foram classificados em cinco grupos. São eles: propriedades espaço-temporais; papéis; associações; restrições; e qualidade dos dados. Outra lista de requisitos é exibida em Lisboa Filho e Iochpe (1999). São eles: fenômenos geográficos e objetos convencionais; visões de campo e objeto; aspectos temáticos; aspectos espaciais; múltiplas representações; relacionamentos espaciais; e aspectos temporais. Friis-Christensen et al. (2001), assim como Lisboa Filho e

Iochpe (1999), também fizeram uma comparação entre alguns modelos e estes requisitos para mostrar as vantagens e desvantagens de cada modelo.

Existem atualmente diversas propostas de modelos específicos para dados geográficos. Alguns dos mais conhecidos são: GeoOOA (KÖSTERS; PAGEL; SIX, 1997), OMT-G (BORGES; DAVIS; LAENDER, 2001), MADS (PARENT; SPACCAPIETRA; ZIMÁNUI, 2008), UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008) e *Perceptory's model* (BÉDARD; LARRIVÉE, 2008). Cada um desses modelos apresenta características particulares e procuram implementar os requisitos da modelagem de aplicações geográficas. Uma comparação de quanto cada um implementa esses requisitos foi feita também em Sampaio (2009).

2.2 Mecanismos de extensão da UML

A UML é uma linguagem visual para especificar, construir, visualizar e documentar artefatos de sistemas (OMG, 2007). Ela é uma linguagem orientada a objetos, que captura aspectos dinâmicos e estruturais de um sistema (ERIKSSON et al., 2004).

As linguagens de modelagem orientadas a objetos surgiram nos anos 80. À medida que as aplicações foram tornando-se cada vez mais complexas, e diante da programação orientada a objetos, foi necessária a introdução de métodos alternativos de análise e projeto (BOOCH; RUMBAUGH; JACOBSON, 2005).

Alguns métodos foram criados e passaram a ser utilizados pela comunidade de *software*, sendo que três desses se destacaram, a saber: os métodos OOD (BOOCH, 1992), OOSE (JACOBSON et al., 1992) e o OMT (RUMBAUGH et al., 1990). Algum tempo depois, os autores desses três métodos se juntaram para unificar seus métodos, criando uma linguagem de modelagem unificada. Dessa forma surgiu a UML, que alguns anos depois foi oferecida para padronização ao *Object Management Group* (OMG) e desde então, passou por várias melhorias até atingir a atual versão 2.1.2 (OMG, 2007).

A especificação da UML é definida usando uma abordagem de metamodelagem, isto é, um metamodelo é utilizado para especificar o modelo que compõe a UML. O OMG padronizou uma arquitetura de quatro camadas que organiza os diferentes níveis conceituais que compõe um modelo: as instâncias, o modelo do sistema, a linguagem de modelagem e o metamodelo dessa linguagem

(OMG, 2007; FUENTES; VALLECILLO, 2004; FRANKEL, 2003). A Tabela 2.1 resume esta hierarquia definida pelo OMG.

Tabela 2.1. Arquitetura de metamodelagem do OMG

METANÍVEL	DESCRIÇÃO	ELEMENTOS
M3	MOF	Classe MOF, Atributo MOF, Associação MOF, etc.
M2	Metamodelo	Classe UML, Associação UML, Atributo UML, etc. Tabela CWM, Coluna CWM, etc.
M1	Modelos	Classe "Cliente", Classe "Conta" Tabela "Empregado", Tabela "Vendedor"
M0	Instâncias	Cliente "José da Silva", Cliente "Roberto Carlos", Conta "1234", Conta "5678", Empregado "A35", Vendedor "54321"

Fonte: Adaptado de (FRANKEL, 2003)

No nível mais baixo, chamado de M0, encontram-se as instâncias atuais que existem no sistema em execução. Por exemplo, o cliente "José da Silva" e dono da conta "1234". No nível acima, que é a camada M1, encontram-se os modelos. Por exemplo, é nessa camada que é definida a classe "Cliente" e a classe "Conta". Há uma relação próxima entre a camada M0 e a camada M1. Os elementos da camada M0 são instâncias dos elementos da camada M1. Para descrever o modelo do sistema é usada uma linguagem definida na camada M2, que é a linguagem de modelagem. Por exemplo, neste nível, são definidos os conceitos usados para modelar os elementos da camada M1. No caso da UML, a camada M2 define "Classe", "Atributo", "Associação", etc.. E da mesma maneira que houve um relacionamento próximo entre as camadas M0 e M1, também há entre as camadas M1 e M2. Todo elemento na camada M1 é uma instância de um elemento da camada M2 e todo elemento da camada M2 categoriza elementos da camada M1. O modelo que reside na camada M2 é chamado de metamodelo. Finalmente, a camada M3 define os conceitos que podem ser usados para definir linguagens de modelagem, que é o meta-metamodelo. Neste nível se encontra a linguagem *Meta Object Facility* (MOF), que é usada para definir a UML e ela própria (FUENTES; VALLECILLO, 2004; OMG, 2007; FRANKEL, 2003).

A UML é uma linguagem de modelagem de propósito geral que pode ser usada em diversos domínios de aplicação (OMG, 2007). Ela se tornou um dos

padrões mais usados para especificar e documentar sistemas de informação e, com isso, ajudou a melhorar o processo de modelagem de sistemas, tornando a engenharia de *software* uma disciplina mais madura (ERIKSSON et al., 2004).

Existem, porém, situações nas quais os construtores da UML não são capazes de expressar todos os conceitos de determinados domínios. Sendo assim, como citam Eriksson et al. (2004), para evitar que a UML se tornasse complexa demais, seus criadores tornaram-na extensível, ou seja, é possível adaptá-la a um domínio ou plataforma específica. Uma linguagem extensível permite definir novos conceitos, similar à introdução de novas palavras e a extensão do vocabulário de uma linguagem natural (ALHIR, 2003).

O OMG define duas formas de extensão da UML. A primeira é baseada na modificação do metamodelo da UML, criando, assim, uma nova linguagem, na qual a sintaxe e semântica dos novos elementos criados são adaptadas ao domínio pretendido. A segunda forma consiste em adaptar a UML a domínios ou plataformas específicas através do mecanismo de perfis. Nessa segunda alternativa, os elementos da linguagem são especializados, impondo algumas restrições a eles, porém respeitando o metamodelo da UML e mantendo a semântica original dos elementos inalterada (FUENTES; VALLECILLO, 2004; OMG, 2007; ZUBCOFF; PARDILLO; TRUJILLO, 2009).

Na primeira forma de extensão da UML, a nova linguagem é criada utilizando o MOF. Já na segunda alternativa, os elementos da linguagem são especializados utilizando os mecanismos de extensão providos pela UML, que são os estereótipos, *tagged values* e *constraints*.

Em OMG (2007) é descrito que não há uma resposta simples em relação à quando se deve criar um novo metamodelo ou quando se deve utilizar o mecanismo de perfis UML. Cada uma das alternativas apresenta suas vantagens e desvantagens. Definir uma nova linguagem através do MOF pode apresentar um resultado mais próximo da realidade do domínio pretendido, porém como a semântica da UML não é preservada, não é possível aproveitar as ferramentas comerciais UML para construir os diagramas. Já a utilização de perfis UML pode até não se ajustar perfeitamente ao domínio pretendido, porém, é possível aproveitar toda a estrutura da UML, como as ferramentas CASE e materiais de treinamento, proporcionando, assim, um resultado mais rápido para adequação ao perfil. Fuentes e Vallecillo

(2004) citam que os benefícios de usar perfis UML excedem indubitavelmente suas limitações.

Os mecanismos de extensão da UML existem desde suas versões iniciais, porém somente a partir da sua segunda versão é que a noção de perfil foi definida para proporcionar a extensão da linguagem de forma mais estruturada e precisa (OMG, 2007). Em Sampaio (2009) foi utilizado o mecanismo de perfis UML para gerar uma linguagem de domínio específico, em que o domínio pretendido foi o de aplicações geográficas.

Nas sub-seções seguintes são explicados os mecanismos de extensão da UML (estereótipos, *tagged values* e *constraints*), assim como perfis UML.

2.2.1 Estereótipos

Estereótipo é um mecanismo de extensão da UML que define como uma metaclasse existente pode ser estendida e habilita o uso de uma terminologia específica para um domínio ou plataforma específicos em lugar da, ou em adição à, terminologia usada para a metaclasse estendida (OMG, 2007).

Eriksson et al. (2004), assim como OMG (2007), caracterizam estereótipos como um dos principais veículos para a customização da UML. Um estereótipo pode estender uma metaclasse ou outro estereótipo, permitindo assim, que sejam criados novos elementos de modelagem. Não há limite no número de vezes que uma metaclasse pode ser estendida por um estereótipo, nem há regras que limitem o número de classes que podem aplicar um estereótipo.

Um estereótipo é definido por um nome e pelo conjunto de elementos do metamodelo com os quais ele pode ser conectado. Graficamente, eles são definidos em caixas estereotipadas como `<<stereotype>>` (FUENTES; VALLECILLO, 2004). Os elementos do metamodelo são indicados por classes estereotipadas como `<<metaclass>>`. Eles podem também mudar a aparência visual dos elementos do modelo estendidos usando ícones gráficos (OMG, 2007).

2.2.2 *Tagged Values*

Um *tagged value* é um meta-atributo adicional que é vinculado a uma metaclasse do metamodelo estendido por um perfil. Eles adicionam informações aos

elementos do modelo, as quais podem ser utilizadas por seres humanos e por máquinas, podendo assim ser expressos por alguma linguagem natural ou computacional. Seres humanos podem usar esses meta-atributos para adicionar informações administrativas sobre o modelo como, por exemplo, o autor do modelo e a data e hora da última modificação. Máquinas podem usá-los, por exemplo, para a geração de código (ERIKSSON et al., 2004). *Tagged values* têm um nome e um tipo, e são associados a um estereótipo específico (OMG, 2007).

2.2.3 *Constraints*

Constraints são restrições em forma de regras ou definição de condições que são aplicadas a elementos do modelo. A todo elemento da UML está associada alguma semântica. Isso quer dizer que cada elemento gráfico dessa linguagem possui um significado bem definido que, uma vez entendido, fica implícito na utilização do elemento em algum diagrama. As restrições permitem estender ou alterar a semântica natural de um elemento gráfico (BEZERRA, 2002).

As restrições podem ser associadas aos estereótipos, impondo, assim, restrições aos elementos do metamodelo correspondente. As regras definidas para um estereótipo se aplicam a cada elemento do modelo para os quais o estereótipo é aplicado e cada elemento do modelo para os quais o estereótipo precisa aderir às regras (ALHIR, 2003).

Constraints podem ser expressas usando uma linguagem natural, ou a *Object Constraint Language* (OCL), que é adotada pelo OMG para expressar restrições e propriedades de elementos do modelo (FUENTES; VALLECILLO, 2004).

2.2.4 Perfil UML

Um perfil UML é um conjunto dos mecanismos de extensão da UML (estereótipos, *tagged values* e *constraints*), agrupados em um pacote UML estereotipado como `<<profile>>`. Como mencionado anteriormente, esses mecanismos permitem a extensão da sintaxe e semântica dos elementos da UML, porém sem violar a semântica original desses elementos.

A intenção é fornecer um mecanismo direto para adaptar um metamodelo existente com construtores que são específicos para um domínio, plataforma ou método em particular. Tais adaptações são agrupadas em um perfil (OMG, 2007).

O pacote de perfis contém mecanismos que permitem metaclasses de metamodelos existentes serem estendidas para adaptá-las a diferentes propósitos. Isso inclui a habilidade para customizar a UML a diferentes plataformas ou domínios específicos como, por exemplo, aplicações em tempo-real ou modelagem de processos de negócios. O mecanismo de perfis é consistente com o MOF (OMG, 2007).

Apesar do OMG fornecer a especificação da UML, que contém os conceitos de perfil UML e os elementos que compõem uma definição de perfil, não há um método ou um processo estabelecido de como definir perfis UML. Algumas propostas surgiram para suprir essa necessidade como, por exemplo, as de Fuentes e Vallecillo (2004) e Selic (2007).

Alguns perfis UML já existem e estão disponíveis para uso público. Alguns desses já foram adotados e padronizados pelo OMG, tais como o perfil UML para CORBA (*Common Object Request Broker Architecture*), o perfil UML para EDOC (*Enterprise Distributed Object Computing*), o perfil UML para EAI (*Enterprise Application Integration*) e outros. Alguns outros perfis UML existentes estão em processo de serem adotados pelo OMG e outros estão sendo criados por organizações privadas, companhias de *software* e centros de pesquisa.

2.3 XML *Metadata Interchange* (XMI)

A linguagem XML (*eXtensible Markup Language*) se tornou popular depois que a especificação do MOF foi escrita. Como a popularidade de XML cresceu, ela se tornou um excelente meio para ferramentas trocarem modelos de uma maneira padrão. O crescimento de XML foi um caso clássico de uma nova e imprevista tecnologia emergente (FRANKEL, 2003).

O MOF também é utilizado para definir um formato de intercâmbio para os modelos da camada M1 (Tabela 2.1). Sempre que uma linguagem de modelagem é definida utilizando-se o metamodelo MOF, este define uma maneira padronizada de geração de formato de intercâmbio para modelos naquela linguagem. No final de 1998, o OMG adotou um mapeamento MOF-XML, o qual é conhecido como XML

Metadata Interchange (XMI). Este formato de intercâmbio é baseado em XML. Como o MOF é definido com base em si próprio, XMI também pode ser utilizado para gerar formatos de intercâmbio padrão para metamodelos (KLEPPE; WARMER; BAST, 2003).

Os diagramas UML, assim como perfis UML, devem ser interoperáveis entre as ferramentas CASE através do formato XMI (OMG, 2007). Várias ferramentas CASE já permitem o uso desse formato como, por exemplo, a ferramenta *Rational Software Modeler* (RSM), *Papyrus UML2 Modeler*, entre outras. Uma lista das ferramentas CASE utilizam esse formato pode ser encontrada em UML Forum (2009).

2.4 *Model Driven Architecture* (MDA)

Com novas tecnologias que surgem a cada ano e também as constantes mudanças nos requisitos, torna-se necessário o surgimento de soluções no desenvolvimento de *software* que proporcionem uma maior interoperabilidade entre as aplicações para que os esforços gastos com a substituição de uma tecnologia por outra seja minimizado. Isso evitaria que muito do trabalho já realizado seja repetido, garantindo, assim, reusabilidade e aumento de produtividade.

Se há alguns anos a principal preocupação dos desenvolvedores estava relacionada com as linguagens e os paradigmas de programação, atualmente considera-se que houve uma mudança de pensamento no mercado de tecnologia da informação, que é o aumento do interesse em modelagem de dados e aplicações.

Com a promessa de melhorar o desenvolvimento de *software*, o OMG adotou, então, a abordagem MDA (*Model Driven Architecture*), que tem como ponto chave a importância dada aos modelos. Nela, o processo de desenvolvimento de *software* é dirigido pela atividade de modelagem do sistema. Um modelo de um sistema é uma descrição ou especificação do sistema. Os artefatos produzidos em MDA são modelos formais, isto é, modelos que podem ser entendidos por computadores (OMG, 2003).

A promessa da abordagem MDA é permitir a definição de aplicações legíveis por máquinas e modelos de dados que permitem flexibilidade a longo prazo (OMG, 2003). Entre essas flexibilidades desejadas, pode-se citar:

- **Implementação**: novas infra-estruturas de implementação podem ser integradas por projetos existentes;
- **Integração**: pode-se automatizar a produção de pontes para integração de dados e a conexão às novas infra-estruturas de integração;
- **Manutenção**: a disponibilidade do projeto em uma forma legível por máquina permite aos desenvolvedores acesso direto à especificação do sistema, tornando a manutenção muito mais simples;
- **Teste e simulação**: uma vez que os modelos desenvolvidos podem ser utilizados para gerar código, eles também podem ser validados conforme os requisitos, testados conforme as várias infra-estruturas e podem ser usados para simular diretamente o comportamento do sistema que está sendo projetado.

A abordagem MDA prevê o desenvolvimento de um modelo inicial, livre de considerações sobre a plataforma na qual o sistema será implantado. Este modelo inicial é então transformado – segundo um conjunto de regras de transformação – em um modelo que considera características mais técnicas, porém ainda independente de qualquer plataforma, e depois, este último é transformado em um modelo com características próprias de alguma plataforma específica (OMG, 2003).

As subseções seguintes descrevem essas etapas da abordagem MDA, assim como as transformações que são realizadas nos modelos.

2.4.1 Níveis de abstração de modelos em MDA

Os requisitos para o sistema em MDA são modelados em um modelo computacionalmente independente, o CIM (*Computation Independent Model*). Tal modelo é, algumas vezes, chamado modelo de domínio ou modelo de negócio e usa um vocabulário familiar aos especialistas do domínio em questão. Um CIM não mostra detalhes da estrutura dos sistemas, mas o ambiente em que o sistema vai operar, sendo útil para entender o problema em mãos (OMG, 2003).

Em um nível de abstração mais baixo está o PIM (*Platform Independent Model*), que ainda é um modelo independente de qualquer tecnologia de implementação (KLEPPE; WARMER; BAST, 2003). Esse modelo pode usar uma

linguagem de modelagem de propósito geral ou uma linguagem específica para a área em que o domínio será utilizado (OMG, 2003).

Posteriormente, o PIM é transformado em um modelo de plataforma específica (PSM ou *Platform Specific Model*). Um PSM é customizado para especificar o sistema em termos dos construtores de implementação que estão disponíveis em uma tecnologia de implementação específica. Por exemplo, um PSM para banco de dados relacional inclui termos como "tabela", "coluna", "chave estrangeira" e outros. Um PIM pode ser transformado em um ou mais PSMs. Para cada plataforma de tecnologia específica um PSM separado é gerado. O passo seguinte é a transformação de cada PSM para código-fonte. Essa transformação é relativamente direta pelo fato do PSM ser ajustado a uma tecnologia específica.

A Figura 2.1 ilustra os principais passos no processo de desenvolvimento MDA, a saber: CIM, PIM, PSM. Conforme mostrado, o CIM é o modelo com o mais alto nível de abstração, seguido pelo PIM e posteriormente pelo PSM.

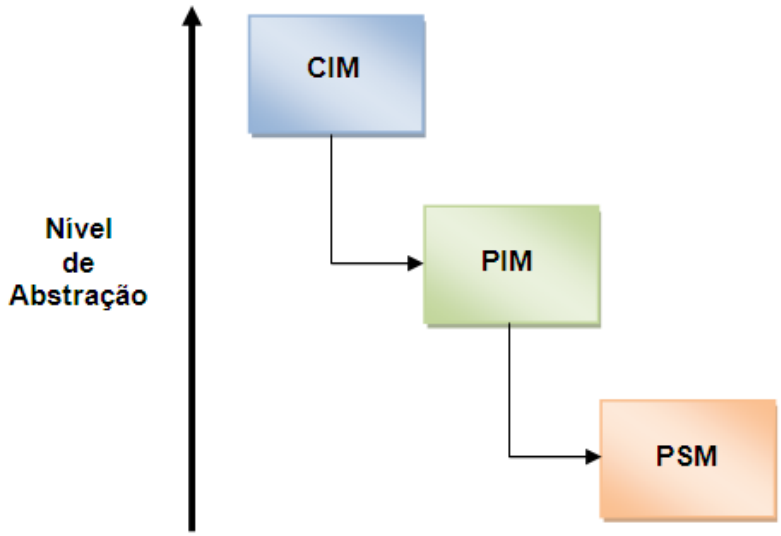


Figura 2.1. Os principais passos no processo de desenvolvimento MDA

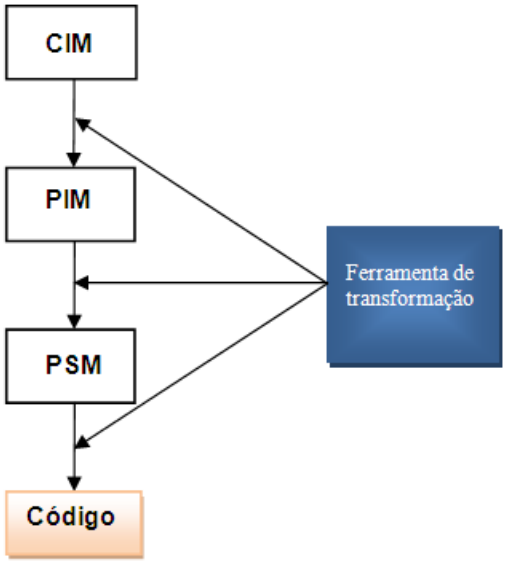
O CIM, PIM e PSM são mostrados como artefatos em diferentes passos no ciclo de vida de desenvolvimento e representam diferentes níveis de abstração na especificação de sistema. A habilidade para transformar um CIM de alto nível em um PIM e posteriormente, transformar um PIM em um PSM eleva o nível de abstração em que um desenvolvedor pode trabalhar. Isso permite a um desenvolvedor enfrentar sistemas mais complexos com menos esforço (OMG, 2003).

2.4.2 Transformações de Modelos em MDA

Transformação de modelos é o processo de converter um modelo em outro modelo do mesmo sistema (OMG, 2003). A transformação utiliza os mapeamentos, mas também contém outras informações como, por exemplo, a condição no modelo-fonte para disparar a transformação, a linguagem do modelo-alvo, a linguagem-fonte, etc. (KLEPPE; WARMER; BAST, 2003).

Um mapeamento é especificado usando alguma linguagem para descrever uma transformação de um modelo em outro. A descrição pode ser em linguagem natural ou em uma linguagem de transformação de modelos. Uma qualidade desejável de uma linguagem de transformação é a portabilidade. Isso habilita o uso de um mapeamento com diferentes ferramentas (OMG, 2003).

Outro elemento chave de MDA é que as transformações sejam executadas por ferramentas. O objetivo da MDA é automatizar a parte incômoda e trabalhosa do desenvolvimento de *software*. Tradicionalmente, as transformações de modelo para modelo, ou de modelo para código são feitas manualmente. Já, em MDA, transformações são preferencialmente executadas por ferramentas, como ilustrado pela Figura 2.2 (KLEPPE; WARMER; BAST, 2003).



Fonte: Adaptado de (KLEPPE; WARMER; BAST, 2003)

Figura 2.2. Utilização de ferramentas de transformação na abordagem MDA

O modelo de plataforma específica (PSM) produzido pela transformação é um modelo do mesmo sistema especificado pelo PIM; ele também especifica como

aquele sistema faz uso da plataforma escolhida. Um PSM pode prover mais ou menos detalhes, dependendo de seus propósitos. Um PSM será uma implementação se ele provê todas as informações necessárias para construir um sistema e colocá-lo em operação ou ele pode agir como um PIM que é usado para refinamentos para mais adiante gerar um PSM que poderá ser diretamente implementado (OMG, 2003).

O OMG provê também algumas maneiras de transformação de modelos em MDA, sendo que uma delas é a transformação utilizando perfis UML. Segundo o OMG, um PIM pode ser preparado usando um perfil UML independente de plataforma. Esse modelo pode ser transformado em um PSM expresso usando um segundo perfil, de plataforma específica. A transformação poderá ser feita marcando o PIM com elementos de uma plataforma específica (OMG, 2003).

2.4.2.1 Linguagens de Transformação de Modelos

Um dos objetivos da abordagem MDA é diminuir o tempo de desenvolvimento de sistemas usando, para isso, modelos com diferentes níveis de abstração, e em especial os modelos com altos níveis de abstração, os quais permitem uma maior proximidade do problema que se pretende resolver.

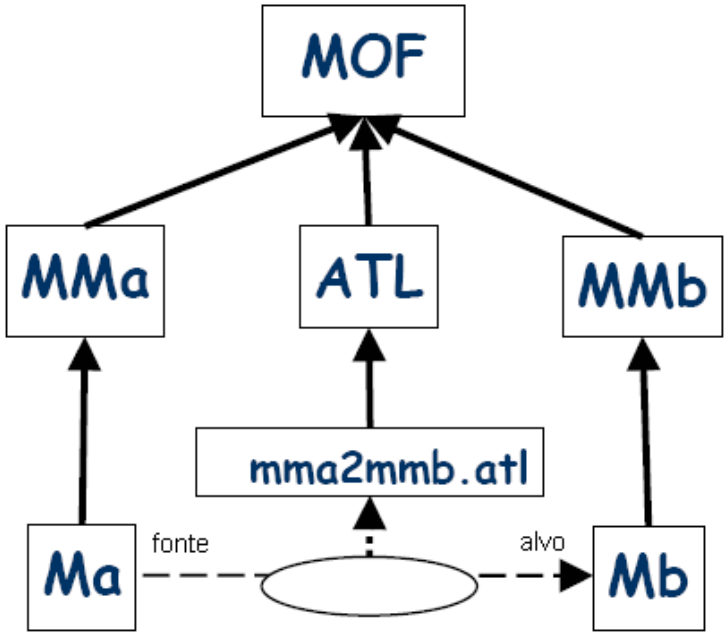
Sendo assim, um dos desafios está na transformação desses modelos de níveis mais altos para modelos de níveis mais baixos. Segundo Sendall e Kozaczynski (2003), as ferramentas de desenvolvimento deveriam oferecer não apenas a possibilidade de aplicar transformações de modelos predefinidas, mas deveriam também oferecer uma linguagem que permita aos usuários definirem seus próprios modelos de transformação e então executá-los sob demanda.

Sendall e Kozaczynski (2003) ainda descrevem algumas características que são desejáveis para essas linguagens de transformação de modelos. A linguagem deve ser implementável de maneira eficiente, ser executável e deve oferecer construtores gráficos, os quais provêem uma maneira mais intuitiva quando comparados com os textuais.

Em geral, essas linguagens de transformação de modelos podem ser imperativas, declarativas ou híbridas. As linguagens imperativas são aquelas que descrevem como atingir determinado objetivo através de uma sequência de ações que devem ser executadas pelo computador. Exemplos são: Cobol, Fortran, C, Pascal, Java, etc.. As linguagens declarativas são aquelas que descrevem o objetivo que deve

ser atingido pelo computador, deixando para o computador escolher a melhor maneira de realizá-lo. Exemplos desse tipo de linguagem são: Lisp, Haskell, SQL, etc.. Já as linguagens híbridas são aquelas combinam características das duas anteriores.

Um exemplo de linguagem de transformação de modelos é a *Atlas Transformation Language* (ATL), desenvolvida pelo grupo de pesquisa ATLAS INRIA & LINA (ATL, 2010). Ela é uma linguagem híbrida, porém o estilo recomendado para escrever as regras de transformação é o declarativo. No entanto, em determinadas situações cuja utilização de regras puramente declarativas seja difícil, é permitido o uso de construções imperativas. A Figura 2.3 apresenta o padrão transformacional em que a linguagem é aplicada (JOUAULT; KURTEV, 2005).



Fonte: Adaptado de (JOUAULT; KURTEV, 2005)

Figura 2.3. Visão geral da abordagem de transformação de modelos da linguagem ATL

Nesse padrão, um modelo fonte (*Ma*) é transformado em um modelo alvo (*Mb*). A transformação é conduzida por uma definição de transformação (*mma2mmb.atl*), que é escrita usando os construtores da linguagem ATL. A definição da transformação também é um modelo. Os modelos fonte, alvo e a definição da transformação devem obedecer aos seus metamodelos (*MMa*, *MMb* e

ATL, respectivamente). Os metamodelos devem obedecer a um meta-metamodelo, que neste caso é o MOF (JOUAULT; KURTEV, 2005).

A linguagem ATL é direcionada a transformações de modelo para modelo. Ela não provê suporte à transformação de modelos para texto. Além disso, as transformações são unidirecionais, não sendo permitida a definição de transformações bidirecionais. Nesse caso, uma transformação bidirecional é implementada como um par de transformações, sendo uma para cada direção.

2.4.3 Benefícios da abordagem MDA

Alguns dos prováveis benefícios proporcionados pela utilização da abordagem MDA, descritos por Kleppe et al. (2003), são citados abaixo:

- **Produtividade**: Com a criação do PIM e das definições de transformação, é possível aproveitar o trabalho feito no desenvolvimento de outros sistemas, evitando, assim, que um mesmo trabalho seja feito novamente. Porém, o ganho de produtividade só pode ser realmente obtido pelo uso de ferramentas que automatizem a geração de um PSM a partir do PIM;
- **Portabilidade**: Em MDA, a portabilidade é alcançada pelo fato do PIM, que por definição é independente de plataforma, poder ser transformado em múltiplos PSMs para diferentes plataformas. Porém, isso só será possível com a disponibilidade de ferramentas que automatizem esse processo;
- **Interoperabilidade**: MDA permite que PSMs gerados a partir de um PIM tenham relacionamentos, os quais são chamados de pontes (*bridges*). Havendo esses relacionamentos, os PSMs gerados para diferentes plataformas poderão comunicar-se entre si. Para isso, as ferramentas de automatização precisam ser capazes de gerar não só os PSMs em diferentes plataformas, mas também as pontes entre eles;
- **Manutenção e Documentação**: Depois de construído, o PIM é utilizado como uma documentação de alto nível para o sistema. Para as mudanças que poderão ocorrer, basta alterar o PIM e gerar novamente o PSM e o código.

3 Padrões internacionais para informação geográfica

O uso de padrões é bastante comum em nossa vida cotidiana como também pelas organizações de modo geral. Um exemplo bastante comum nas organizações é a série ISO 9000 que é utilizada para gerenciamento de qualidade. Essa série define um conjunto de regras para investigar a eficiência do gerenciamento de qualidade em uma organização.

Sob uma perspectiva econômica, os esforços de padronização podem trazer benefícios. Por exemplo, um estudo feito no ano 2000 provou que as vantagens econômicas do uso de padrões para a indústria alemã eram de 15 bilhões de dólares por ano (KRESSE; FADAIE, 2004).

A modelagem de banco de dados geográficos (BDGeo) também pode se beneficiar muito de padrões, principalmente em relação à interoperabilidade de aplicações. Para alcançar isso, os esforços para a padronização internacional na área de informação geográfica começaram durante a última década por meio de organizações como ISO e OGC (BRODEUR; BADARD, 2008).

Essas organizações são exemplos dos dois tipos de grupos em nível internacional que existem para padronização, que são as organizações internacionais e os consórcios internacionais. As organizações internacionais baseiam suas decisões no consenso e são independentes dos interesses individuais de indústrias ou governos. Seus padrões são conhecidos como padrões *de jure*. Um exemplo de organização internacional é a ISO. Já os consórcios internacionais são formados principalmente por membros de indústrias, agências governamentais e universidades. Os padrões desenvolvidos pelos consórcios são chamados de padrões industriais ou padrões *de facto*. O *Open Geospatial Consortium* (OGC) pode ser considerado o consórcio mais importante na comunidade de informação geográfica. Outro exemplo de consórcio internacional é o *World Wide Web Consortium* (W3C), que organiza o trabalho necessário para o desenvolvimento e evolução de tecnologias *Web* (KRESSE; FADAIE, 2004).

Segundo Brodeur e Badard (2008), o desenvolvimento desses padrões para informação geográfica tem como objetivo reduzir a inconsistência entre o *de jure* (i.e., oficial, aprovado por um corpo de padronização, como ISO) e o *de facto* ou padrões industriais (i.e., adotados por usuários, indústria e/ou setor profissional, por causa de sua popularidade).

Nas seções seguintes são descritos alguns dos principais padrões internacionais para informação geográfica publicados pela ISO e pelo OGC e que são utilizados neste trabalho de pesquisa. Na seção 3.1 é descrita a série ISO 19100, publicada pelo Comitê Técnico ISO/TC211, bem como os padrões dessa série que são utilizados neste trabalho, a saber, o padrão ISO 19107 *Spatial Schema*, que é descrito na subseção 3.1.1, o padrão ISO 19108 *Temporal Schema*, descrito na subseção 3.1.2 e o padrão ISO 19123 *Schema for Coverage Geometry and Functions*, descrito na subseção 3.1.3. Na seção 3.2 é descrito o *OpenGIS Simple Feature Access*, publicado pelo consórcio internacional OGC. Esse documento está dividido em duas partes. Na subseção 3.2.1 é descrita a primeira parte e na subseção 3.2.2 a segunda parte.

3.1 A série ISO 19100 do Comitê Técnico ISO/TC 211

ISO/TC211 é o comitê técnico padrão formado dentro da ISO e voltado para a área de informação geográfica digital. Ele é responsável pela preparação de uma série de padrões internacionais e especificações técnicas numeradas pela série 19100. Esses padrões podem especificar, para a informação geográfica, métodos, ferramentas e serviços para gerenciamento de dados (incluindo definição e descrição), aquisição, processamento, análise, acesso, apresentação e transferência de tais dados entre diferentes usuários, sistemas e localizações (ISO/TC211, 2009).

Com esses padrões, o ISO/TC211 objetiva melhorar o entendimento, uso, a disponibilidade, acesso, integração e compartilhamento da informação geográfica, assim como promover o seu uso de forma eficiente, efetivo e econômico, podendo assim, contribuir para o tratamento de problemas humanitários e ecológicos globais.

A família de padrões da série ISO 19100 não forma um modelo completo e hierárquico para todo o universo da informação geográfica. Os padrões são mais uma coleção de padrões abstratos independentes para criar e gerenciar sistemas de informação geográfica (KRESSE; FADAIE, 2004).

Os padrões da série ISO 19100, publicados pelo Comitê Técnico ISO/TC 211, estão divididos em grupos. Como listado em ISO/TC211 (2009), há o grupo de padrões que especificam a infra-estrutura para a padronização geo-espacial, padrões que descrevem modelos de dados para informação geográfica, padrões para gerenciamento da informação geográfica, padrões para serviços de informação

geográfica, padrões para codificação da informação geográfica e padrões para áreas temáticas específicas. Uma lista completa dos padrões pertencentes a cada um desses grupos pode ser encontrada no apêndice B.

Os padrões para informação geográfica contribuem em vários níveis de abstração, desde a modelagem até considerando aspectos de implementação. Mais especificamente, segundo Brodeur e Badard (2008), os seguintes padrões contribuem direta ou indiretamente para a modelagem da informação geográfica:

- ISO/TS 19103 *Conceptual Schema Language*
- ISO 19107 *Spatial Schema*
- ISO 19108 *Temporal Schema*
- ISO 19109 *Rules for Application Schema*
- ISO 19110 *Methodology for Feature Cataloging*
- ISO 19111 *Spatial Referencing by Coordinates*
- ISO 19112 *Spatial Referencing by Geographic Identifiers*
- ISO 19115 *Metadata*
- ISO 19135 *Procedures for Item Registrations*

Nesta dissertação são considerados alguns dos padrões que fazem parte do grupo de padrões que descrevem modelos de dados para informação geográfica. São eles: o padrão ISO 19107 *Spatial Schema*, ISO 19108 *Temporal Schema* e o padrão ISO 19123 *Schema for Coverage Geometry and Functions*. Esses padrões são utilizados para fazer a correspondência com os elementos do GeoProfile (SAMPAIO, 2009), o que será visto no capítulo 5.

3.1.1 ISO 19107 *Spatial schema*

Esse padrão internacional especifica esquemas conceituais para descrever e manipular as características espaciais das feições geográficas. Uma feição é uma abstração de um fenômeno do mundo real. Essa abstração é uma feição geográfica se ela está associada a uma localização relativa na Terra (ISO/TC211, 2003). O padrão utiliza a UML para apresentar os esquemas conceituais, que consistem em definições de classes conceituais que poderão ser usadas em esquemas de aplicações, perfis e especificações de implementação. Ele também define operações espaciais, padrões

para uso no acesso, consulta, gerência, processamento e intercâmbio de dados geográficos.

A representação das características espaciais é fundamental para a descrição das feições geográficas. Esquemas conceituais padronizados para as características espaciais melhoram a habilidade para compartilhar informações geográficas entre aplicações. Esses esquemas poderão ser usados por SIG, desenvolvedores de *software* e usuários de informação geográfica para prover estruturas de dados espaciais consistentes (ISO/TC211, 2003).

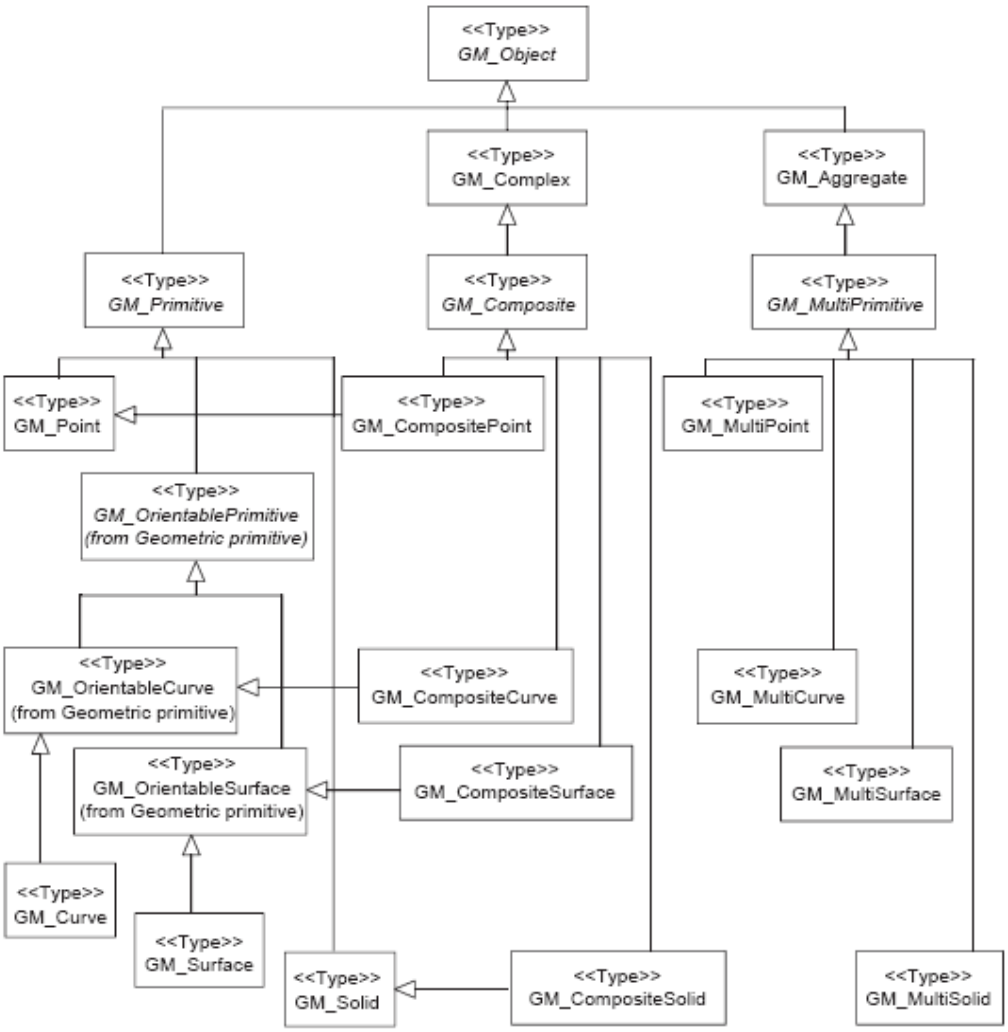
As características espaciais envolvem a geometria da feição, sua localização com relação a um sistema de coordenadas e suas propriedades topológicas. O padrão internacional ISO 19107 define em detalhes as características geométricas e topológicas que são necessárias para descrever as feições geográficas (BRODEUR; BADARD, 2008). As características espaciais são descritas por um ou mais atributos espaciais cujo valor é dado por um objeto geométrico (*GM_Object*) ou um objeto topológico (*TP_Object*). A geometria é o aspecto da informação geográfica que muda quando a informação é transformada de um sistema de coordenadas para outro e a topologia lida com as características das figuras geométricas que permanecem invariantes se o espaço é deformado de algum modo.

A Figura 3.1 mostra as classes de geometria básicas especificadas neste padrão internacional. Conforme a figura, de *GM_Object*, que está no topo da hierarquia de classes, são especializadas as três subclasses principais que representam as características geométricas, que são: *GM_Primitive*, *GM_Complex* e *GM_Aggregate*. Qualquer objeto que herda a semântica de *GM_Object* age como um conjunto de *posições diretas* (posição descrita por um simples conjunto de coordenadas em um sistemas de referência de coordenadas). Seu comportamento será determinado por quais posições diretas ele contém.

GM_Primitive constitui as primitivas geométricas básicas, que são representadas pelas subclasses *GM_Point*, *GM_Curve*, *GM_Surface* e *GM_Solid*. Como descreve Brodeur e Badard (2008), elas provêem todos os componentes necessários para descrever a forma e localização de feições geográficas simples tais como prédios, torres, estradas, pontes, rios, etc.. ISO/TC211 (2003) ainda destaca que os objetos que herdam de *GM_Primitive* serão abertos, ou seja, eles não conterão seus pontos de limite. Assim, por exemplo, curvas não conterão seus pontos finais,

superfícies não conterão suas curvas limites e sólidos não conterão suas superfícies limites.

Isso conduz a alguma ambigüidade aparente. Uma representação de uma linha como uma primitiva deve referenciar seus pontos finais, mas não conter esses pontos como um conjunto de posições diretas. Uma representação de uma linha como um complexo deverá também referenciar seus pontos finais, e deverá conter esses pontos como um conjunto de posições diretas. Isso significa que representações digitais idênticas terão diferentes semânticas dependendo se elas são acessadas como primitivas ou complexos.



Fonte: (ISO/TC211, 2003)

Figura 3.1. Hierarquia de classes geométricas do padrão ISO 19107

A subclasse *GM_Complex* constitui os objetos complexos, que são feições geográficas que têm uma estrutura geométrica mais complicada como o caso de uma

estrada ou uma rede hidrográfica. Esses objetos consistem de um conjunto de *GM_Primitives* com os interiores disjuntos, isto é, o interior não tem interseção com outra geometria (ISO/TC211, 2003). Por exemplo, uma *GM_CompositeCurve* é feita de um conjunto de *GM_OrientableCurves*, onde o primeiro ponto de cada curva corresponde ao último ponto do anterior. Similarmente, uma *GM_CompositePoint* se comporta como uma *GM_Point*, uma *GM_CompositeSurface* como uma *GM_OrientableSurface* e uma *GM_CompositeSolid* como uma *GM_Solid* (BRODEUR; BADARD, 2008; ISO/TC211, 2003).

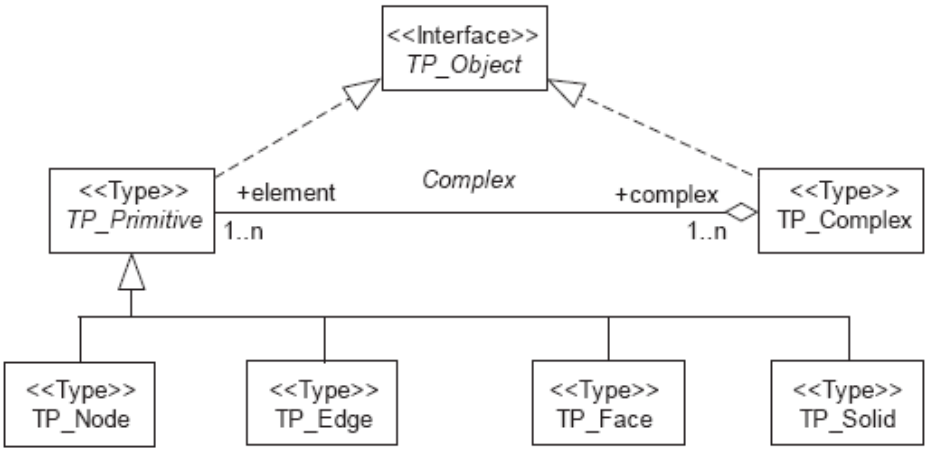
GM_Primitive e *GM_Complex* possuem algumas semelhanças, porém uma diferença básica entre ambas é que uma *GM_Primitive* não deve conter seus limites (exceto no caso de *GM_Point* em que o limite é vazio), enquanto um *GM_Complex* deve conter seus limites em todas as classes. Além disso, as duas classes, *GM_Object* e *GM_Primitive*, são abstratas, ou seja, nenhum objeto ou estrutura de dados de um esquema de aplicação pode instanciá-las diretamente. Instâncias dessas classes devem ser de um de seus subtipos não abstratos, tais como *GM_Point*, *GM_Curve* ou *GM_Surface*. Esse não é o caso para *GM_Complex*, que pode ser diretamente instanciado.

A subclasse *GM_Aggregate* constitui as geometrias agregadas, que são feições compostas de múltiplas primitivas geométricas. Aplicações podem usar agregados para feições que usam múltiplos objetos geométricos em suas representações, tais como uma coleção de pontos para representar um pomar ou um arquipélago. Como ainda destaca Brodeur e Badard (2008), a localização de todos os tipos de primitivas geométricas, agregados e complexos é descrita com coordenadas que são referenciadas a um sistema de referência de coordenadas.

As primitivas topológicas são necessárias para a realização de cálculos geométricos complexos, como cálculos de adjacência, fronteiras e análise de redes entre objetos geométricos (BRODEUR; BADARD, 2008). Esses tipos de cálculos utilizam uma computação intensa e o uso de topologia pode acelerar a realização desses tipos de cálculos.

A Figura 3.2 mostra a hierarquia de classes topológicas do padrão ISO 19107. No topo da hierarquia se encontra a classe *TP_Object*, que é uma classe abstrata e raiz de duas subclasses: *TP_Primitive* e *TP_Complex*. As primitivas topológicas, representadas pela subclasse *TP_Primitive*, podem ser comparadas com as primitivas geométricas apresentadas anteriormente e inclui as seguintes subclasses: *TP_Node*,

TP_Edge, *TP_Face* e *TP_Solid*. Os complexos topológicos são representados pela subclasse *TP_Complex*.



Fonte: (ISO/TC211, 2003)

Figura 3.2. Hierarquia de classes topológicas do padrão ISO 19107

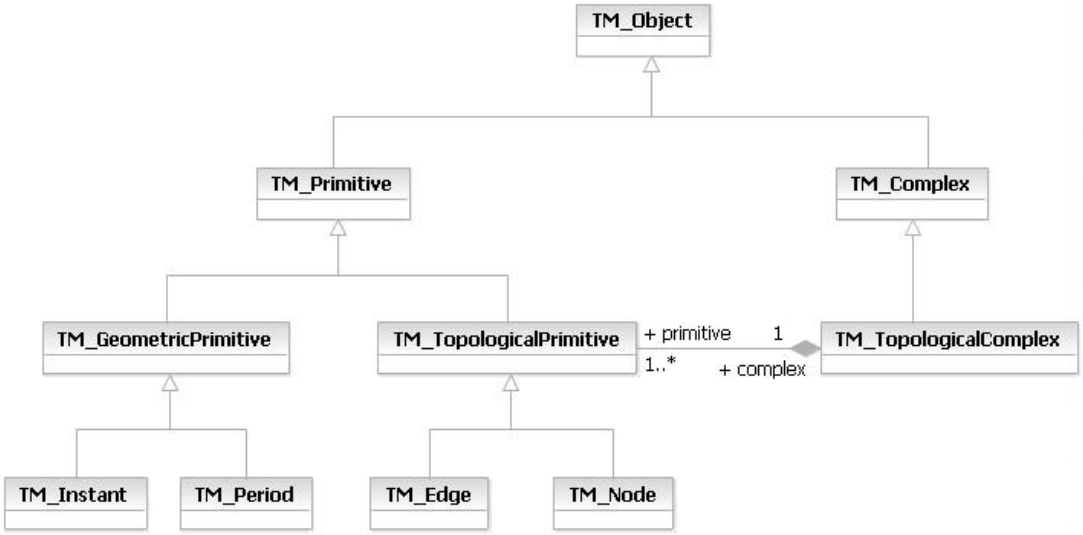
3.1.2 ISO 19108 *Temporal schema*

O documento *ISO 19108 Temporal Schema* é o padrão internacional que define os conceitos relacionados às características temporais da informação geográfica, mostrando como essas características são abstraídas do mundo real. Segundo ISO/TC211 (2002), a padronização das características temporais é fundamental para a utilização da informação geográfica em certos tipos de aplicações como, por exemplo, simulações e modelagem preditiva. O tempo é assunto de interesse para uma grande variedade de disciplinas técnicas e científicas. Assim, muitos dos conceitos descritos neste padrão internacional são aplicáveis também fora do campo da informação geográfica.

Jensem et al. (1994) conceitua dois tipos de tempo, o tempo válido e o tempo de transação. O primeiro é o tempo quando um fato é verdadeiro na realidade observada e é gerado pelo usuário. Já o segundo é o tempo quando um fato está em vigor num banco de dados e pode ser recuperado. Esse tempo é gerado pelo sistema. A modelagem completa da realidade só é alcançada se forem utilizadas em conjunto as características de tempo de transação e de tempo de validade no banco de dados.
Sampaio (2005), porém, destaca que nas aplicações de SIG, o mais importante é saber quando um determinado dado é válido na realidade modelada. Esse padrão

internacional segue essa linha, dando ênfase somente ao tempo válido em lugar do tempo de transação.

A hierarquia de classes definidas em *ISO 19108 Temporal Schema* é mostrada na Figura 3.3. São considerados os aspectos geométricos e topológicos das características temporais. Um objeto temporal está simbolizado no topo da hierarquia de classes com a classe abstrata *TM_Object*, que tem duas subclasses, também abstratas, *TM_Primitive* e *TM_Complex*. *TM_Primitive* é a classe abstrata que generaliza as primitivas temporais geométricas e topológicas. As primitivas geométricas, representadas pela classe abstrata *TM_GeometricPrimive*, consistem em instante e período, representadas, respectivamente, pelas subclasses *TM_Instant* e *TM_Primitive*. Já as primitivas topológicas, que provêem informação sobre conectividade no tempo, são representadas pela classe abstrata *TM_TopologicalPrimitive*, a qual possui duas subclasses, *TM_Edge* e *TM_Node*. Um conjunto de primitivas topológicas unidas formam um complexo topológico temporal, isto é, um *TM_TopologicalComplex*. Cada *TM_TopologicalPrimitive* será membro de um e somente um *TM_TopologicalComplex*.



Fonte: (ISO/TC211, 2002)

Figura 3.3. Hierarquia de classes temporais do padrão ISO 19108

3.1.3 ISO 19123 *Schema for Coverage Geometry and Functions*

Segundo ISO/TC211 (2005), os fenômenos geográficos podem ser divididos em duas categorias. São elas: os fenômenos discretos e os fenômenos contínuos. Os fenômenos discretos são aqueles cujos objetos são reconhecíveis e possuem limites ou extensão espacial bem definido como, por exemplo, prédios, estações de medição, entre outros. E os fenômenos contínuos são aqueles cujos objetos não possuem uma extensão espacial definida como, por exemplo, temperatura, composição do solo e outros.

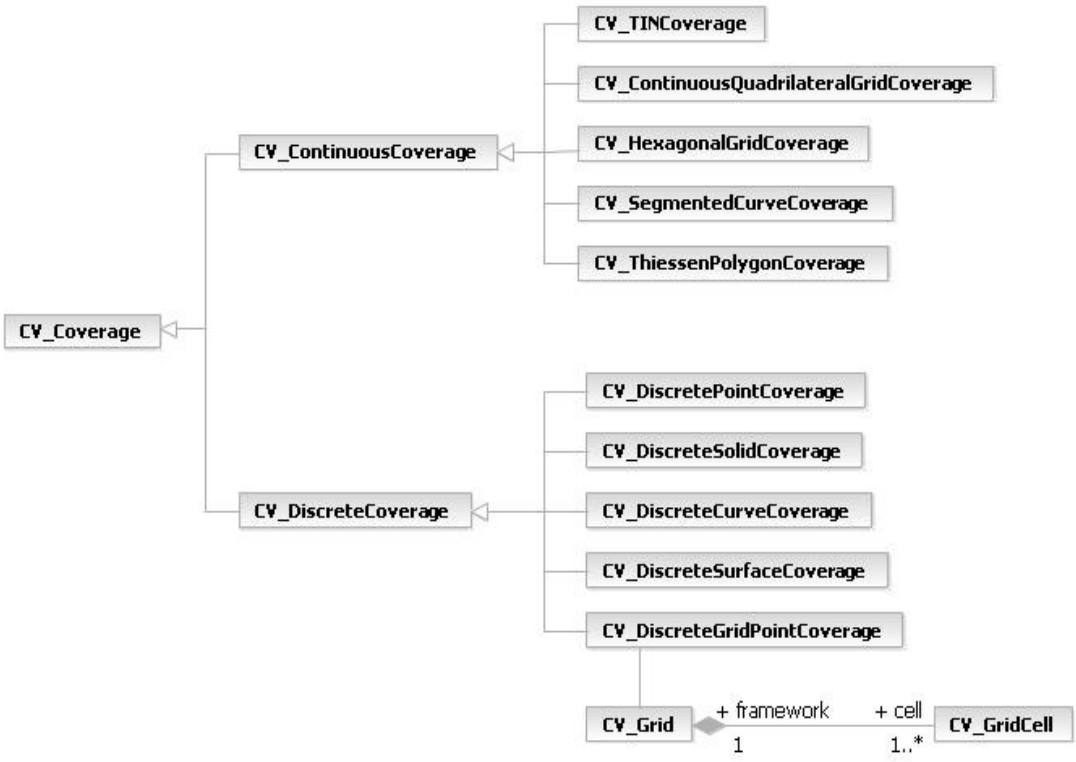
O padrão ISO 19123 *Schema for Coverage Geometry and Function* define um esquema para as características espaciais de coberturas. Uma cobertura é uma feição que tem múltiplos valores para cada tipo de atributo e pode representar uma feição simples ou um conjunto de feições. Elas integram fenômenos geográficos discretos e contínuos e são usadas em várias áreas como, por exemplo, em sensoriamento remoto, meteorologia, mapeamento de vegetação, solos, etc. (ISO/TC211, 2005).

A Figura 3.4 apresenta o diagrama de classes do padrão. Situada no topo da hierarquia está a classe *CV_Coverage*, que possui como subclasses *CV_DiscreteCoverage* e *CV_ContinuousCoverage*, as quais representam os fenômenos geográficos discretos e contínuos, respectivamente.

Para os fenômenos discretos (classe *CV_DiscreteCoverage*) são especificadas cinco subclasses, as quais são: *CV_DiscretePointCoverage*, *CV_DiscreteCurveCoverage*, *CV_DiscreteSurfaceCoverage*, *CV_DiscreteGridPointCoverage* e *CV_DiscreteSolidCoverage*.

E para os fenômenos contínuos (classe *CV_DiscreteCoverage*) também são especificadas cinco subclasses, que são: *CV_ThiessenPolygonCoverage*, *CV_ContinuousQuadrilateralGridCoverage*, *CV_HexagonalGridCoverage*, *CV_TINCoverage* e *CV_SegmentedCurveCoverage*.

Maiores detalhes sobre o padrão internacional ISO 19123 pode ser encontrado em ISO/TC211 (2005).



Fonte: (ISO/TC211, 2005)

Figura 3.4. Diagrama de classes do padrão internacional ISO 19123

3.2 OpenGIS – Simple Feature Access

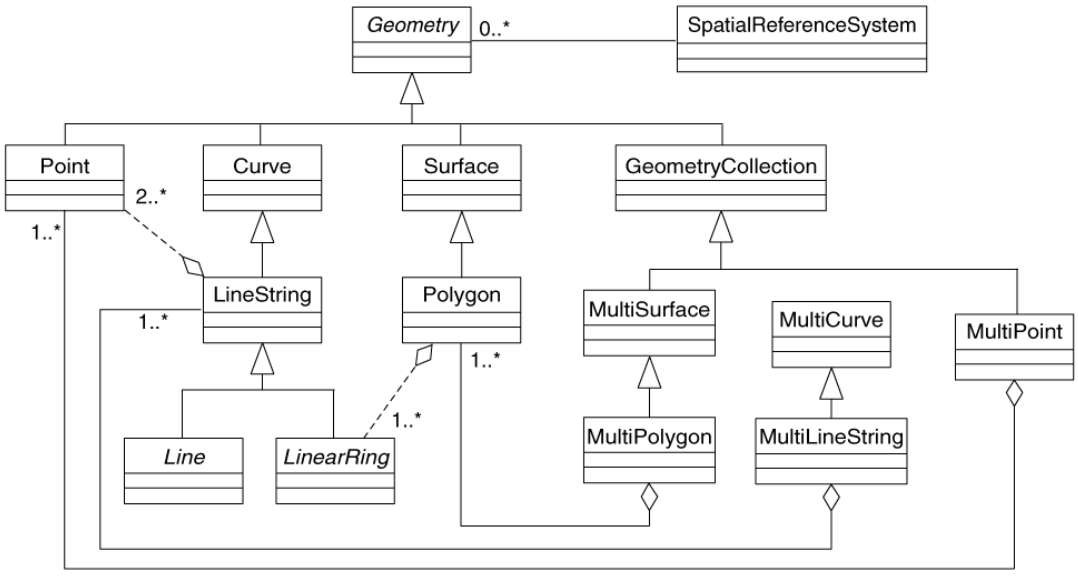
O *Open Geospatial Consortium* (OGC) é atualmente o principal consórcio responsável pelo desenvolvimento de padrões industriais para sistemas de informação geográfica. Seu processo de desenvolvimento é diferente da abordagem ISO. O OGC desenvolve principalmente especificações voltadas para implementação, enquanto a ISO desenvolve especificações mais abstratas.

Apesar dessas diferenças sutis, ambas têm desenvolvido acordos cooperativos para a harmonização de seus trabalhos e para o desenvolvimento de trabalhos futuros (BRODEUR; BÉDARD; PROULX, 2000). A prova disso é o documento *OpenGIS – Simple Feature Access*, que também é reconhecido pela ISO com a denominação ISO 19125.

O documento está dividido em duas partes. A primeira (*Common architecture*) descreve uma arquitetura comum para feições geográficas simples e a segunda (*SQL option*) descreve uma implementação em SQL do modelo descrito na primeira parte. Ambas as partes lidam com feições simples, ou seja, feições cuja geometria é restrita a duas dimensões.

3.2.1 Parte 1: *Common architecture*

Essa primeira parte do *OpenGIS Simple Features Access* (SFA), também chamada ISO 19125, descreve uma arquitetura comum para geometria de feições simples. Feições simples são aquelas em que todos os atributos geométricos são descritos por interpolação linear entre vértices. O modelo cobre feições de até duas dimensões, usa a UML e tem como base o padrão ISO 19107 (OGC, 2006a). A hierarquia de classes do modelo é mostrada na Figura 3.5.



Fonte: (OGC, 2006a)

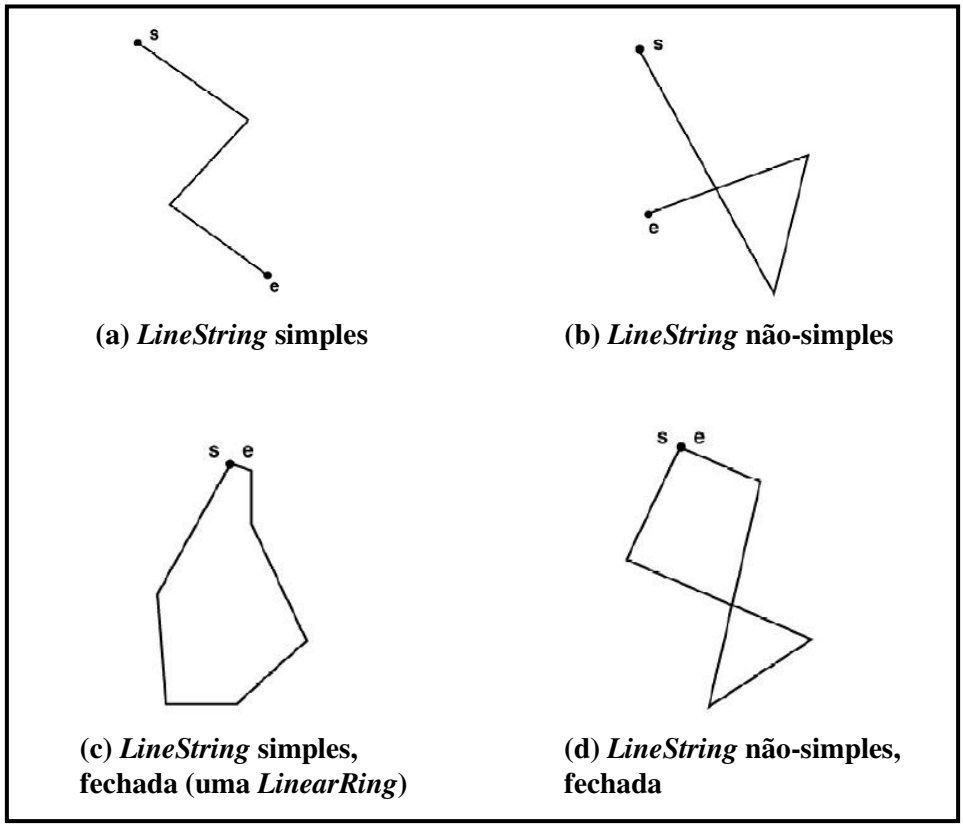
Figura 3.5. Hierarquia de classes de geometria

A classe raiz da hierarquia é a classe abstrata *Geometry*, que possui as subclasses *Point*, *Curve*, *Surface* e *GeometryCollection*. Todas elas são classes instanciáveis e estão restritas a objetos geométricos de até duas dimensões. Além disso, todas as classes de geometria descritas nesta especificação são topologicamente fechadas, isto é, todas as geometrias representadas incluem seus limites como conjunto de pontos. No entanto, isso não afeta sua representação.

Um ponto (classe *Point*) é um objeto geométrico com dimensão 0 e representa uma localização simples no espaço de coordenadas. Um ponto tem um valor de coordenada x e um valor de coordenada y. O limite ou fronteira de um ponto é o conjunto vazio.

Uma curva (classe *Curve*) é um objeto geométrico com dimensão 1 e usualmente é armazenado como uma sequência de pontos. O subtipo da curva é que

vai especificar a forma da interpolação entre os pontos. Essa especificação define apenas uma subclasse de curva, *LineString*, a qual usa interpolação linear entre os pontos. Uma linha (classe *Line*) é uma *LineString* com exatamente dois pontos. Já uma *LinearRing* é uma *LineString* fechada e simples. A Figura 3.6 mostra exemplos de *LineStrings*.



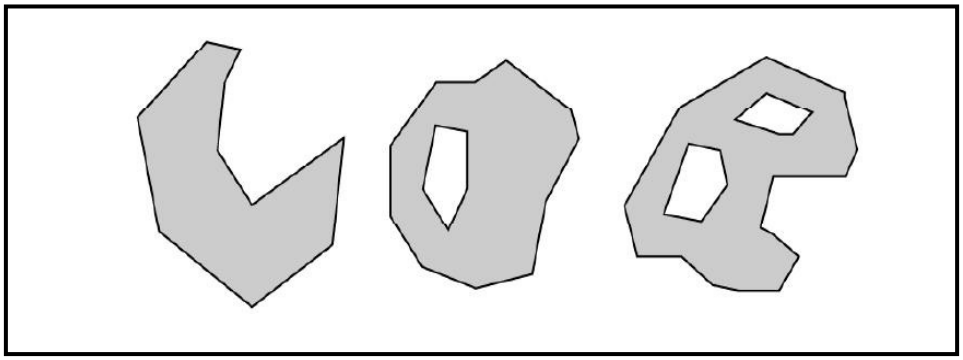
Fonte: (OGC, 2006a)

Figura 3.6. Exemplos de *LineStrings*

Uma superfície (classe *Surface*) é um objeto geométrico com dimensão 2. Como subclasse de superfície, um polígono (classe *Polygon*) é uma superfície simples e planar. Ele é definido um limite exterior e zero ou mais limites interiores. Cada limite interior define um buraco no polígono. Os limites de um polígono consistem em um conjunto de anéis lineares(classe *LinearRing*) que constroem seus limites interiores e exteriores. A Figura 3.7 mostra exemplos de objetos que são polígonos válidos, enquanto a Figura 3.8 mostra exemplos de objetos que não são representados como polígonos.

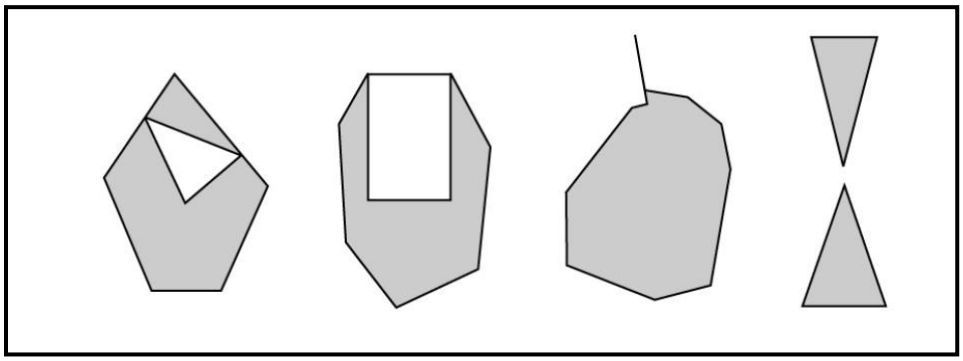
A classe *GeometryCollection* representa os objetos geométricos que são coleções de outros objetos geométricos. As sublasses *MultiPoint*, de dimensão 0,

MultiLineString, de dimensão 1 e *MultiPolygon*, de dimensão 2 representam coleções de pontos, *LineStrings* e polígonos, respectivamente. As classes *MultiCurve* e *MultiSurface* são introduzidas na hierarquia para generalizar as interfaces de coleções para manipular curvas e superfícies.



Fonte: (OGC, 2006a)

Figura 3.7. Exemplos de polígonos válidos



Fonte: (OGC, 2006a)

Figura 3.8. Exemplos de objetos que não podem ser representados como polígonos

É interessante notar que a classe *Geometry* é equivalente à classe *GM_Object*, descrita no padrão ISO 19107, assim como a classe *GeometryCollection* é equivalente à classe *GM_Aggregate*. Porém, o presente modelo difere da ISO 19107 por não incluir complexos, suporte a uma terceira dimensão e à topologia.

3.2.2 Parte 2: *SQL option*

Essa segunda parte do *OpenGIS SFA*, e também chamada ISO 19125, tem como objetivo definir um esquema na linguagem SQL que dê suporte ao armazenamento, recuperação, consulta e atualização de dados relacionados a feições geográficas. Conforme OGC (2006b), uma feição geográfica é uma abstração de um objeto do mundo real e tem atributos espaciais como também atributos não espaciais, sendo que os atributos espaciais são aqueles que possuem valores relacionados à geometria da feição.

Esse padrão depende da arquitetura definida na parte 1, a qual foi descrita para representar feições simples. Como visto, feições simples são baseadas em entidades geométricas com até duas dimensões e com interpolação linear entre vértices. No caso deste padrão, feições simples são armazenadas como uma "tabela de feições", em que as linhas representam as feições e as colunas os atributos das feições.

Os tipos de dados dos atributos não espaciais são tirados do conjunto de tipos de dados da linguagem SQL, incluindo os tipos definidos pelo usuário, da SQL3. Já os atributos espaciais são mapeados para colunas cujos tipos são baseados em tipos de dados geométricos, os quais são definidos neste padrão.

As tabelas de feições são descritas neste padrão usando dois tipos de implementação. O primeiro é baseado no modelo relacional clássico e usa os tipos de dados predefinidos na linguagem SQL com tipos adicionais para geometria. Nesse caso, o valor de geometria na tabela de feições é definido em uma coluna que faz referência a uma tabela de geometria. A tabela de geometria é implementada usando tipos numéricos padrões ou tipos binários SQL.

Já o segundo tipo de implementação utiliza uma extensão da SQL para tipos de geometria. Nesse caso, a coluna para geometria na tabela de feições tem o tipo de dados tirado desse conjunto de tipos de geometria. O mecanismo para estender o sistema de tipos da SQL é através dos Tipos Definidos pelo Usuário (TDU). A implementação do esquema é feita usando a declaração CREATE TABLE da linguagem SQL.

4 GeoProfile

Neste capítulo é descrito o GeoProfile, um perfil UML desenvolvido para a modelagem conceitual de BDGeo (SAMPAIO, 2009). A seção 4.1 apresenta a estrutura básica do GeoProfile, dando ênfase ao metamodelo do domínio e aos estereótipos do perfil. A seção 4.2 apresenta uma proposta de notação gráfica para os estereótipos do GeoProfile. A seção 4.3 apresenta a experiência de implementação do perfil em outras ferramentas CASE, já que sua versão inicial foi implementada apenas na ferramenta RSM da IBM. Na seção 4.4 são feitas algumas considerações finais sobre o capítulo.

4.1 Estrutura do GeoProfile

O GeoProfile, especificado inicialmente por Sampaio (2009), é um perfil UML para a modelagem conceitual de BDGeo que foi desenvolvido seguindo as diretrizes para especificação de perfis UML discutidas em Fuentes e Vallecillo (2004) e em Selic (2007) e teve como base os principais modelos conceituais para BDGeo existentes, que são: OMT-G (BORGES; DAVIS; LAENDER, 2001), MADS (PARENT; SPACCAPIETRA; ZIMÁNUI, 2008), GeoOOA (KÖSTERS; PAGEL; SIX, 1997), UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008) e o modelo implementado na ferramenta *Perceptory* (BÉDARD, 1999). A Figura 4.1 ilustra os modelos que contribuíram para o desenvolvimento da versão inicial do GeoProfile.

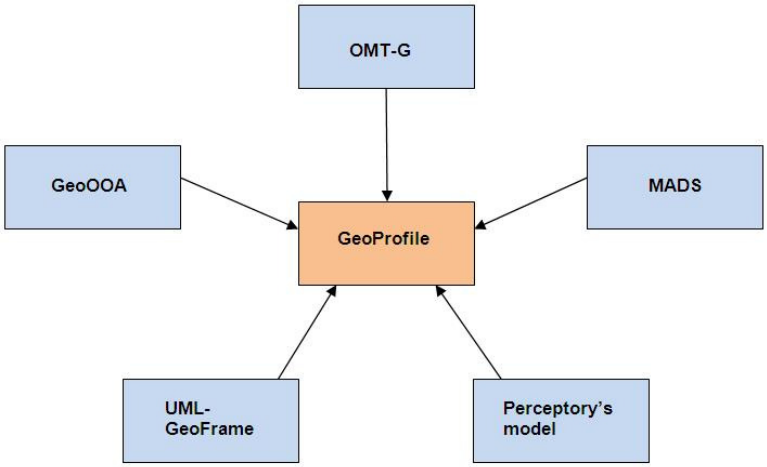
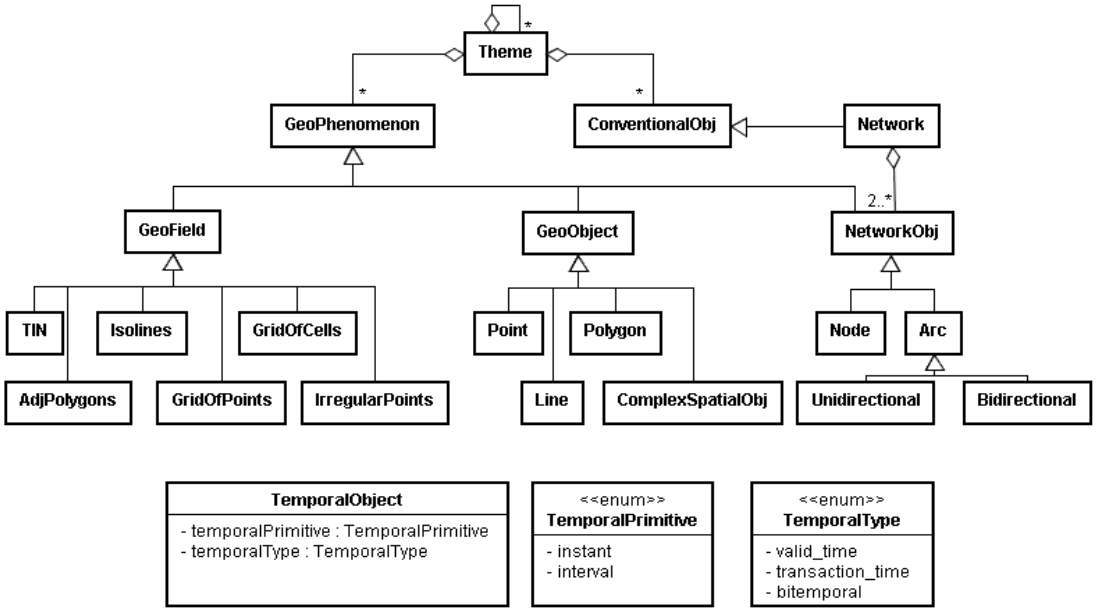


Figura 4.1. Modelos que contribuíram para a especificação do GeoProfile

O primeiro passo para a especificação do GeoProfile foi a construção do metamodelo do domínio. Nessa etapa procurou-se conhecer os principais conceitos presentes na modelagem conceitual de BDGeo e os requisitos básicos que um modelo conceitual de BDGeo deve suportar.

Para a realização dessa etapa, Sampaio (2009) fez uma análise dos principais modelos conceituais de BDGeo existentes e citados anteriormente, procurando extrair as principais contribuições de cada um, sempre buscando atender aos requisitos da modelagem conceitual de BDGeo. O resultado dessa etapa é mostrado na Figura 4.2.

A metaclasse *Theme*, situada mais no topo do metamodelo, pode ser um agregado de zero ou mais temas, como também de fenômenos geográficos ou objetos convencionais, sendo esses últimos representados pelas metaclasses *Geo-Phenomenon* e *ConventionalObj*, respectivamente. A metaclasse *GeoPhenomenon* generaliza os fenômenos cuja localização na superfície terrestre é considerada e a metaclasse *ConventionalObj* descreve os objetos sem representação espacial.



Fonte: (SAMPAIO, 2009)

Figura 4.2. Metamodelo do domínio para modelagem conceitual de BDGeo

A metaclasse *GeoPhenomenon* é especializada nas metaclasses *GeoField*, *GeogObject* e *NetworkObj*, as quais representam, respectivamente, os fenômenos geográficos percebidos na visão de campo, na visão de objetos e os elementos de rede. A metaclasse *GeoField* é uma generalização das metaclasses dos fenômenos

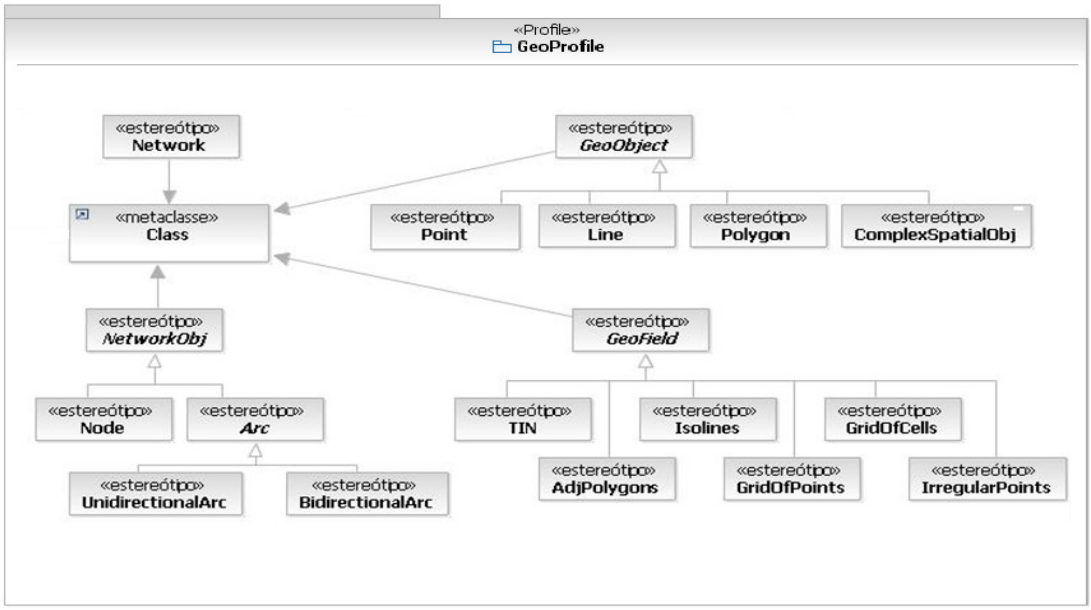
que são percebidos na visão de campo. Essas são: *AdjPolygons*, *Isolines*, *TIN*, *GridOfPoints*, *GridOfCells* ou *IrregularPoint*. A metaclasse *GeoObject* generaliza as metaclasses dos fenômenos percebidos na visão de objetos, as quais são: *Point*, *Line*, *Polygon*, *ComplexSpatialObj*. E a metaclasse *NetworkObj* generaliza os objetos formadores de uma rede, que são os nós, representados pela metaclasse *Node*, e os arcos, representados pela metaclasse *Arc*. Essa última ainda é especializada nas metaclasses *Unidirectional* e *Bidirectional*, que são os arcos uni e bidirecionais, respectivamente.

A metaclasse *Network*, incluída para tratar da representação de redes, é responsável por informar quais os elementos participantes da rede. Ela foi definida como uma especialização da metaclasse *ConventionalObj* pelo fato de não possuir informações espaciais. Dessa forma, uma rede é formada por dois ou mais objetos de rede (*NetworkObj*), que podem ser nós (*Node*), arcos unidirecionais (*Unidirectional*) ou bidirecionais (*Bidirectional*).

Em relação aos aspectos temporais, o metamodelo possui uma metaclasse denominada *TemporalObject*, a qual possui dois atributos que caracterizam a informação temporal. São eles: *temporalPrimitive* e *temporalType*. O primeiro define a primitiva temporal utilizada (instante ou intervalo), enquanto o segundo indica o tipo temporal (tempo de validade, tempo de transação ou bitemporal).

Depois de criar o metamodelo do domínio, o próximo passo foi criar o perfil em si, estendendo as metaclasses da UML. Nessa etapa, foram definidos os estereótipos, *tagged values* e as *constraints*.

Na Figura 4.3 são mostrados os estereótipos criados para os fenômenos geográficos, os quais estendem a metaclasse *Class* e cuja hierarquia é equivalente à do metamodelo do domínio. O estereótipo *Network* estende diretamente a metaclasse *Class*. Os estereótipos *GeoField*, *GeoObject*, *NetworkObj* e *Arc* foram definidos como abstratos, ou seja, eles não serão disponibilizados para serem adicionados às classes do esquema durante a utilização do GeoProfile, mas sim as suas subclasses correspondentes.



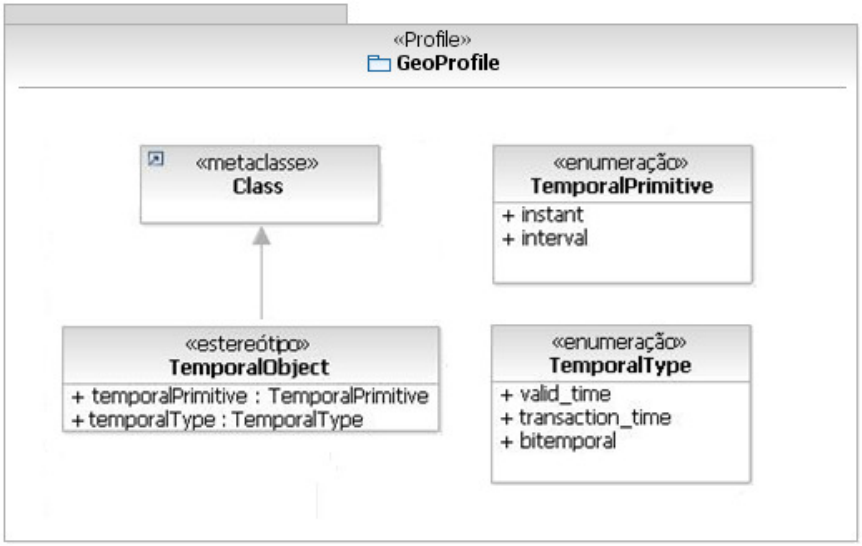
Fonte: Adaptado de (SAMPAIO, 2009)

Figura 4.3. Estereótipos para os fenômenos geográficos

Segundo Sampaio (2009), nem todas as metaclasses do metamodelo do domínio possuem um estereótipo correspondente, como ocorre para *Theme* e *ConventionalObj*. Os temas podem ser representados por pacotes UML. Já as classes de objetos convencionais são modeladas pelas classes da UML sem a adição de estereótipos. Os próprios construtores da UML são capazes de reproduzir esses dois conceitos.

Para tratar dos aspectos temporais, foi incluído o estereótipo *TemporalObject*, como mostrado na Figura 4.4, e que também estende a metaclasse *Class*. As duas enumerações incluídas (*TemporalPrimitive* e *TemporalType*) servem para listar os possíveis valores que os meta-atributos (*tagged values*) *temporalPrimitive* e *temporalType* podem assumir.

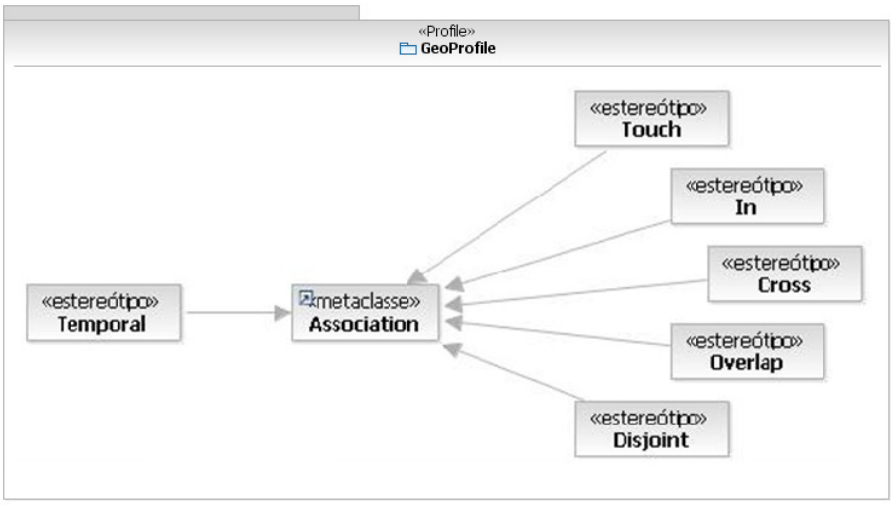
Além das extensões à metaclasse *Class*, foi incluída também extensões à metaclasse *Association*. A Figura 4.5 mostra essas extensões, as quais tiveram o propósito de criar estereótipos para atender aos relacionamentos topológicos, os quais são: *Touch*, *In*, *Cross*, *Overlap* e *Disjoint*. Foi incluído também o estereótipo *Temporal*, o qual permite aos projetistas indicar que uma associação entre dois objetos só é válida por um período e que este histórico deve ser mantido no banco de dados.



Fonte: Adaptado de (SAMPAIO, 2009)

Figura 4.4. Estereótipos para os aspectos temporais

Além dos estereótipos, foram também definidas algumas *constraints*, que servem, entre outras coisas, para validação do esquema conceitual gerado pelo projetista. As *constraints* foram definidas utilizando a linguagem OCL e, segundo Sampaio (2009), evitam a ocorrência de três tipos de erros: adição de estereótipos incompatíveis a um mesmo elemento, má construção de redes e adição de relacionamentos topológicos impossíveis de acontecer entre dois elementos. Informações mais detalhadas a respeito do GeoProfile, assim como as *constraints* definidas, estão disponíveis em Sampaio (2009).



Fonte: Adaptado de (SAMPAIO, 2009)

Figura 4.5. Estereótipos para as associações

4.2 Notação gráfica para estereótipos

Um dos requisitos para projeto de perfis UML descritos em OMG (2007) diz respeito à definição de uma notação gráfica para os estereótipos de um perfil. Na versão inicial do GeoProfile não foi introduzida nenhuma notação gráfica para os estereótipos, porém na modelagem de BDGeo, a notação gráfica para representar as características espaciais de objetos geográficos é empregada em vários modelos. Alguns modelos utilizam outras denominações como, por exemplo, os "pictogramas" desenvolvidos por Bédard e Paquette (1989).

E foram justamente Bédard e Paquette (1989) os pioneiros no uso desse tipo de recurso visual na modelagem de BDGeo. Os "pictogramas" desenvolvidos por eles tinham como objetivo facilitar a modelagem de BDGeo. Segundo Bédard (1999), a modelagem visual de banco de dados ajuda a entender e a descrever mais precisamente o conteúdo pretendido de um banco de dados, como também a controlar a complexidade do problema.

O desenvolvimento dos "pictogramas" foi baseado em ciências cognitivas, a qual mostra que combinar ambas as linguagens gráficas e textuais é fundamental para alcançar um entendimento mais claro de um determinado assunto (BÉDARD, 1999).

Manter a clareza e evitar a sobrecarga visual foi um dos desafios no desenvolvimento dos "pictogramas". Segundo Pooley e Stevens (1999) *apud* Bédard (1999), há um limite de quantidade de detalhes que uma pessoa pode entender ao mesmo tempo. Isso foi levado em consideração na especificação dos "pictogramas", e dessa forma, foi proposto um pequeno conjunto de construtores básicos, com um número mínimo de variações.

Inicialmente, esses "pictogramas" foram projetados para simplificar o modelo Entidade-Relacionamento (ER), de forma que ele pudesse descrever a geometria de feições cartográficas. A primeira solução foi testada em vários projetos e influenciou vários pesquisadores posteriormente em suas pesquisas, que a utilizaram de alguma forma no projeto dos modelos porvindouros como, por exemplo, os modelos UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008), OMT-G (BORGES; DAVIS; LAENDER, 2001), GeoOOA (KÖSTERS; PAGEL; SIX, 1997) e MADS (PARENT; SPACCAPIETRA; ZIMÁNUI, 2008).

Foi desenvolvida, posteriormente, a ferramenta CASE *Perceptory* (BÉDARD, 1999), que estende o diagrama de classes UML para modelar conceitualmente um BDGeo. Segundo Bédard (1999), o uso de "pictogramas" simplifica a modelagem visual de BDGeo na ordem de 40% para modelos orientados a objetos (OO) e 65% para modelo ER.

Neste trabalho foi adicionado ao GeoProfile alguns ícones, os quais são relacionados aos estereótipos do perfil. Esses ícones são semelhantes aos "pictogramas" e foram escolhidos com base nos principais modelos conceituais de dados para aplicações geográficas, os quais foram analisados para a especificação do perfil e já foram citados anteriormente.

Para os estereótipos relacionados aos objetos geográficos percebidos na visão de objetos, os quais são subclasses de *GeoObject*, foram adicionados quatro ícones, relativos aos estereótipos *Point*, *Line*, *Polygon* e *ComplexSpatialObj*. Para os objetos geográficos percebidos na visão de campo, os quais são subclasses de *GeoField*, foram adicionados ícones relativos aos estereótipos *TIN*, *AdjPolygons*, *Isolines*, *GridOfPoints*, *GridOfCell* e *IrregularPoints*. Não foi adicionado ícones aos estereótipos que foram definidos como *abstract*. Todos esses ícones adicionados ao GeoProfile e relativos às subclasses de *GeoObject* e *GeoField*, são mostrados na Figura 4.6 e tiveram a influência principal do modelo proposto por Bédard (1999), sendo que essas figuras também são utilizadas em outros modelos, como no modelo UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008) e no modelo OMT-G (BORGES; DAVIS; LAENDER, 2001).

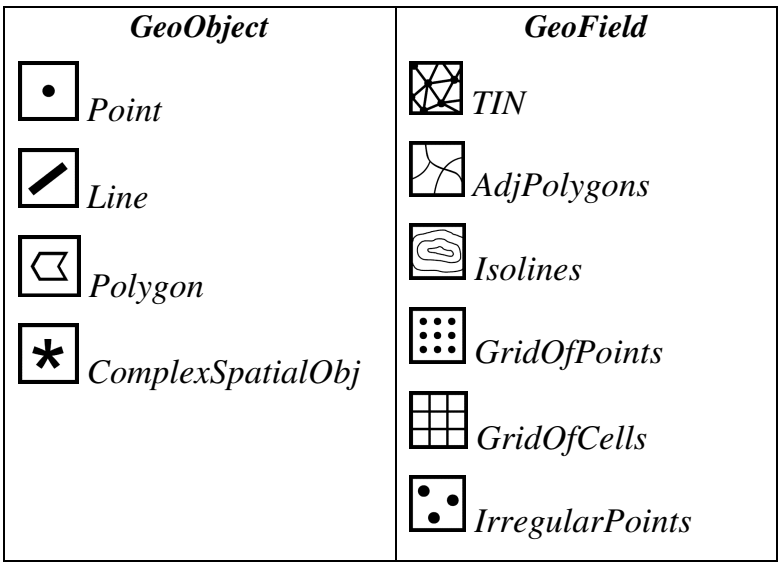


Figura 4.6. Ícones para os estereótipos de *GeoObjects* e *GeoFields*

Com relação aos relacionamentos espaciais, os quais no GeoProfile são representados por estereótipos que estendem a metaclasse *Association*, também foram incluídos ícones, os quais estão relacionados aos seguintes estereótipos: *Touch*, *In*, *Cross*, *Overlap* e *Disjoint*.

Estes estereótipos foram especificados com base no estudo feito por Clementini et al. (1993), que descrevem um conjunto mínimo de relacionamentos capazes de reproduzir os possíveis relacionamentos topológicos que podem ocorrer entre elementos espaciais com a representação de ponto, linha ou área. As *constraints* incluídas no GeoProfile por Sampaio (2009) também avaliam em um esquema conceitual a adição de relacionamentos topológicos impossíveis de acontecer entre dois elementos como, por exemplo, o relacionamento *Touch* entre dois objetos geográficos com representação espacial de ponto. Os ícones incluídos para os relacionamentos topológicos foram influenciados pelo modelo MADS (PARENT; SPACCAPIETRA; ZIMÁNUI, 2008), que os utiliza para indicar que um relacionamento é espacial. Os ícones são mostrados na Figura 4.7.

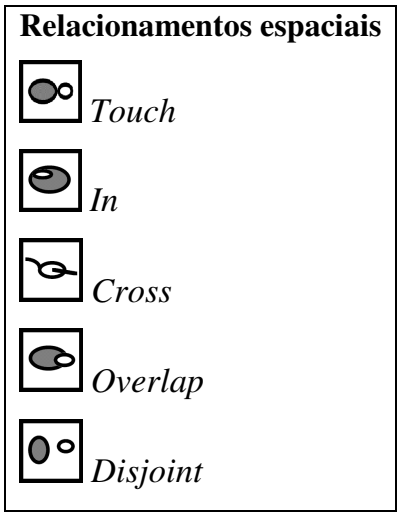


Figura 4.7. Ícones dos estereótipos para os relacionamentos espaciais

Os objetos de rede também receberam ícones para os estereótipos correspondentes, sendo que os mesmos foram influenciados pelo modelo OMT-G (BORGES; DAVIS; LAENDER, 2001), o qual utiliza representação gráfica para representar as classes com geometria e topologia. Lisboa Filho e Stempliuc (2009) também propuseram uma extensão ao modelo UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008) para dar suporte à modelagem de redes, porém os ícones utilizados possuem uma pequena variação em relação aos do modelo OMT-G. Os estereótipos

do GeoProfile que receberam os ícones relacionados aos elementos de rede são mostrados na Figura 4.8. São eles: *Node*, *UnidirectionalArc* e *BidirectionalArc*.

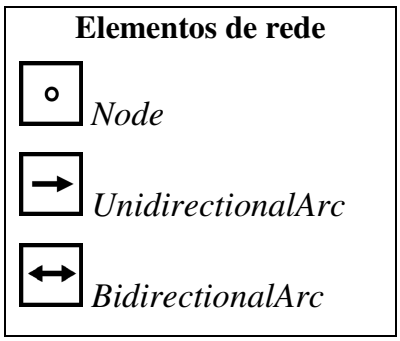


Figura 4.8. Ícones para os estereótipos dos elementos de rede

Para tratar dos aspectos temporais, o GeoProfile possui o estereótipo *TemporalObject*. Para esse foi incluído um ícone, o qual foi baseado no modelo GeoOOA (KÖSTERS; PAGEL; SIX, 1997), sendo que o mesmo é mostrado na Figura 4.9.

Figura 4.9. Ícones para objetos temporais

Como a versão inicial do GeoProfile foi implementada na ferramenta CASE RSM, optou-se também por testar a inclusão e o uso da notação gráfica para os estereótipos nessa mesma ferramenta. O ícone incluído no estereótipo é mostrado no lado esquerdo, em relação ao nome da classe. Para visualização do estereótipo, a ferramenta tem ainda como opções mostrar o estereótipo apenas na forma textual, combinar a forma textual e gráfica ou apenas mostrar a forma gráfica. Os formatos de arquivo aceitos para o ícone a ser incluído no perfil são: .BMP, .GIF, .JPG e .PNG.

Para ilustrar o uso de desses ícones na modelagem de BDGeo utilizando o GeoProfile, alguns exemplos são mostrados abaixo. Na Figura 4.10 é mostrado um exemplo de modelagem em que são utilizados os estereótipos *Polygon*, para representar as classes *District* e *City*, e *Point*, para representar a classe *School*. O esquema ainda contém a classe *Professor*, a qual não tem nenhuma representação espacial. Esse exemplo de esquema conceitual pode também ser encontrado em

Lisboa Filho e Iochpe (2008), onde ele é modelado usando o modelo UML-GeoFrame.

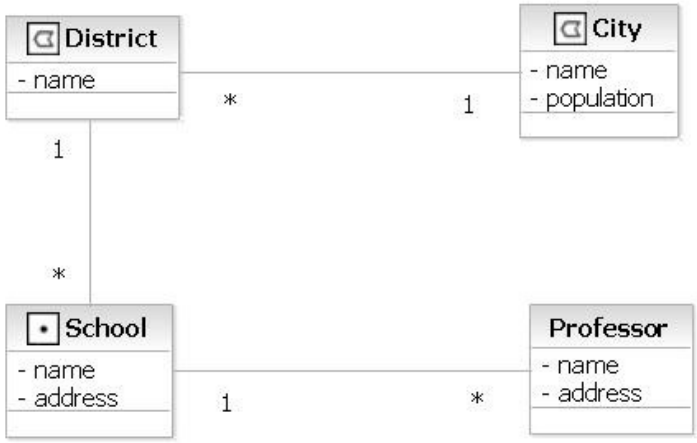


Figura 4.10. Modelagem usando estereótipos gráficos na ferramenta RSM

A Figura 4.11 ilustra um exemplo de modelagem em que é mostrado o relacionamento espacial entre duas classes com representações espaciais do tipo polígono, representadas pelo estereótipo *Polygon*. No esquema, o estereótipo *In* foi utilizado para mostrar que uma ou mais construções (classe *Building*) devem estar contidas dentro de um lote (classe *LandPlot*). A direção de aplicação do estereótipo *In* é mostrada através da seta de indicação de direção da UML, como mostrado na figura. Esse esquema também pode ser encontrado em Parent et al. (2008), que utiliza o modelo MADS para modelá-lo. Uma comparação do mesmo com o GeoProfile também é realizada em Sampaio (2009).

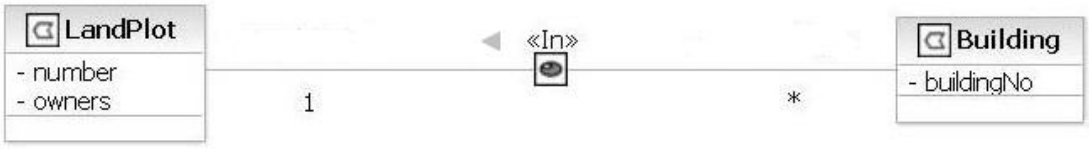


Figura 4.11. Modelagem usando estereótipos gráficos na ferramenta RSM

A Figura 4.12 também apresenta um exemplo de modelagem que faz uso dos relacionamentos espaciais, porém agora utilizando o estereótipo *Overlap*. As duas classes também possuem representações espaciais do tipo polígono. Nesse caso não é necessária a indicação da direção de aplicação do estereótipo, como foi feito na utilização do estereótipo *In*, mostrado na Figura 4.11.

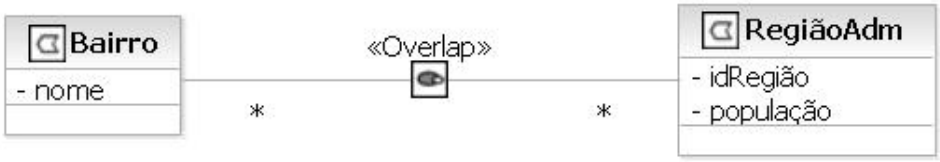


Figura 4.12. Modelagem usando estereótipos gráficos na ferramenta RSM

A Figura 4.13 ilustra alguns exemplos de classes utilizando representações de objetos geográficos na visão de campo. A classe *ImagemSat* utiliza o estereótipo *GridOfCells*, a classe *Umidade* utiliza o estereótipo *IrregularPoints* e a classe *Relevo* utiliza o estereótipo *GridOfPoints*.

Figura 4.13. Exemplos de classes utilizando estereótipos para visão de campo

Em relação aos aspectos temporais, foi utilizado o ícone baseado no modelo GeoOOA (KÖSTERS; PAGEL; SIX, 1997). O exemplo da Figura 4.14 ilustra uma classe que utiliza o estereótipo *TemporalObject*, mostrado na forma gráfica.

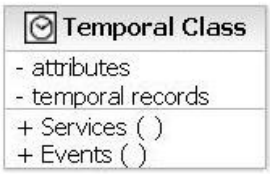


Figura 4.14. Exemplos de classes utilizando estereótipos para aspectos temporais

Uma das limitações da ferramenta CASE RSM é o fato dela não mostrar mais de um ícone, caso seja aplicado à classe mais de um estereótipo. Essas situações são possíveis na modelagem de BDGeo para múltiplas representações de um mesmo objeto geográfico. Se mais de um estereótipo for aplicado a uma determinada classe, por exemplo, a ferramenta mostrará apenas o ícone do primeiro estereótipo aplicado. Para contornar essa situação, é necessário usar a opção de visualização que combina a forma textual e gráfica ou somente a forma textual. Assim, é possível ver que a classe possui mais de um estereótipo aplicado a ela. A Figura 4.15 ilustra esta situação para as classes *City* e *School*, as quais foram modeladas usando os estereótipos *Point* e *Polygon*, respectivamente. Na figura, esses estereótipos são

mostrados combinando as formas textuais e gráficas para a classe *City* e somente na forma textual para a classe *School*.

Figura 4.15. Exemplos de classes com mais de um estereótipo

4.3 Análise da implementação do GeoProfile em ferramentas CASE

Uma das vantagens de se utilizar perfis UML para representar domínios específicos é a possibilidade de se aproveitar toda a infra-estrutura da UML já consolidada como, por exemplo, as ferramentas CASE. Os modelos conceituais de BDGeo estudados têm, cada um, suas formas particulares de implementação. Alguns criaram suas próprias ferramentas CASE, enquanto outros estenderam ferramentas já existentes. O problema é a falta de interoperabilidade entre as soluções criadas, o que faz com que especialistas em um modelo tenham dificuldades de trabalhar com os outros modelos e também prejudica o reuso de soluções em outros projetos, por exemplo, na forma de padrões de análise (LISBOA FILHO; IOCHPE; BORGES, 2002).

A versão inicial do GeoProfile foi implementada na ferramenta CASE RSM da IBM. A ferramenta apresentou-se como uma ótima alternativa para a especificação de perfis, oferecendo recursos como o suporte à linguagem OCL para definição de *constraints* e a inclusão de ícones nos estereótipos. Neste trabalho foi testada a implementação do GeoProfile em outras ferramentas, mais especificamente nas ferramentas *Papyrus UML2 Modeler* (PAPYRUS UML, 2010), *Visual Paradigm* (VISUAL PARADIGM, 2010) e *Star UML* (STAR UML, 2010). As subseções seguintes apresentam uma análise da utilização dessas ferramentas CASE para implementação do GeoProfile.

4.3.1 *Papyrus UML2 Modeler*

A ferramenta CASE *Papyrus UML2 Modeler* (PAPYRUS UML, 2010) mostrou-se uma opção interessante pelo fato de ser uma ferramenta de código aberto (*opensource*). Ela é baseada no ambiente *Eclipse*, assim como a ferramenta RSM (RATIONAL SOFTWARE MODELER, 2010), e está sob a licença EPL (*Eclipse Public License*). A Figura 4.16 ilustra a interface principal da ferramenta, onde é possível notar a semelhança entre ela e a ferramenta RSM.

Entre outros recursos interessantes para modelagem de sistemas utilizando o padrão UML2, a ferramenta dá suporte à criação de perfis UML, que é um dos objetivos de estudo desse trabalho. O perfil é criado selecionando-se os estereótipos a serem utilizados e as metaclasses que serão estendidas por esses estereótipos. Na Figura 4.16 é possível notar parte dos estereótipos do GeoProfile sendo especificados na ferramenta.

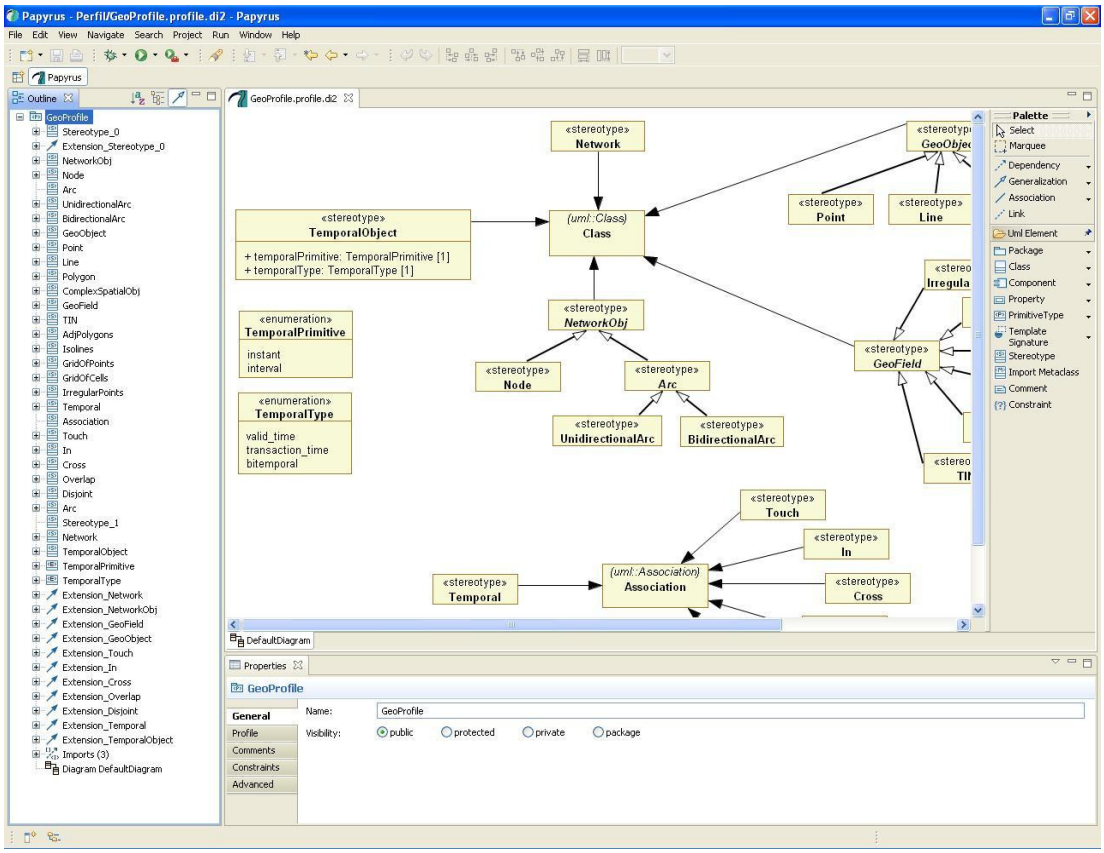


Figura 4.16. Interface da ferramenta CASE *Papyrus UML2 Modeler*

As opções para inclusão de ícones nos estereótipos são as mesmas da ferramenta RSM. São possíveis os seguintes tipos de visualização dos estereótipos:

apenas texto, apenas ícone ou uma combinação de texto e ícone. Um exemplo é mostrado na Figura 4.17, em que o estereótipo *Point*, aplicado à classe *City*, é mostrado como texto, ícone e combinação de texto e ícone, respectivamente.

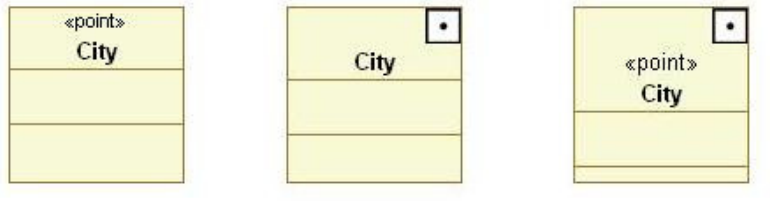


Figura 4.17. Formas de visualização de estereótipos na ferramenta CASE *Papyrus UML2 Modeler*

A ferramenta oferece suporte à linguagem OCL para definição de *constraints***,** sendo as mesmas utilizadas para validar o esquema conceitual gerado. Entretanto, não há opção para importação/exportação de modelos usando o formato XMI e também há a ocorrência do problema da visualização de mais de um estereótipo por classe. Assim como na ferramenta RSM, caso seja aplicado mais de um estereótipo a uma classe, só será visualizado o ícone do primeiro estereótipo aplicado. Para contornar esse problema, deve-se optar pela visualização dos estereótipos na forma textual.

4.3.2 *Visual Paradigm for UML*

Visual Paradigm for UML (VISUAL PARADIGM, 2010) é uma ferramenta CASE com várias opções de modelagem com os diagramas da UML2 e que também oferece suporte a diagramas de requisitos *SysML* e a diagramas ER. A ferramenta possui um bom ambiente de trabalho, o que facilita a visualização e manipulação do projeto de modelagem. É uma ferramenta comercial e também oferece suporte a transformações específicas para códigos-fonte de algumas linguagens de programação como, por exemplo, C++ e Java.

O suporte a perfis UML é oferecido, sendo também permitida a utilização de notação gráfica para os estereótipos. Na implementação de um perfil, ao adicionar os estereótipos, já se escolhe a metaclasse que ele vai estender. Essa extensão não é mostrada explicitamente, como nas ferramentas *Papyrus UML2 Modeler* e RSM. É

possível, também, efetuar importação/exportação de modelos usando o formato padrão de intercâmbio de modelos XMI.

Para implementar o GeoProfile foi utilizada a versão 7.2 da ferramenta. A Figura 4.18 ilustra uma classe com características espaciais que está usando estereótipos do GeoProfile. Diferente das demais, essa ferramenta oferece a possibilidade de visualizar mais de um ícone por classe, caso seja necessário, como ilustra a Figura 4.18.

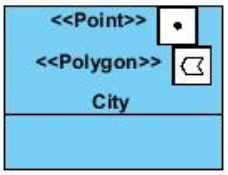


Figura 4.18. Exemplo de classe usando o GeoProfile e modelada na ferramenta CASE *Visual Paradigm for UML*

Apesar do bom suporte à inclusão de estereótipos e da boa usabilidade, a ferramenta não oferece suporte à linguagem OCL para definição de *constraints*. Isso configura uma desvantagem, pois impede que as *constraints* incluídas no GeoProfile sejam utilizadas para validar o esquema conceitual.

4.3.3 *Star UML*

Star UML (STAR UML, 2010) é uma ferramenta CASE de código aberto (*opensource*) e está sob a licença GPL (*General Public License*). Ela dá suporte à modelagem de sistemas utilizando os diagramas da UML2 e também à MDA, com definições de transformações para algumas plataformas específicas. É permitida também a importação/exportação de modelos utilizando o formato XMI.

A especificação de perfis UML na ferramenta é feita de forma diferente das outras ferramentas analisadas. Não há uma forma visual de se implementar perfis. É necessário escrever o código do perfil em um documento XML, salvar o arquivo com a extensão .PRF e colocá-lo em um dos diretórios de instalação da ferramenta. A Figura 4.19 ilustra uma pequena parte do código XML escrito para o GeoProfile.

```
1<?xml version="1.0" encoding="UTF-8" ?>
2<PROFILE version="1.0">
3   <HEADER>
4     <NAME>GeoProfile</NAME>
5     <DISPLAYNAME>GeoProfile</DISPLAYNAME>
6     <DESCRIPTION>GeoDB conceptual modeling</DESCRIPTION>
7   </HEADER>
8   <BODY>
9     <STEREOTYPELIST>
10          <STEREOTYPE>
11                <NAME>GeoObject</NAME>
12                <DESCRIPTION>Object view.</DESCRIPTION>
13                <BASECLASSES>
14                      <BASECLASS>UMLClass</BASECLASS>
15                </BASECLASSES>
16          </STEREOTYPE>
17          <STEREOTYPE>
18                <NAME>Point</NAME>
19                <DESCRIPTION>Indicate a point.</DESCRIPTION>
20                <BASECLASSES>
21                      <BASECLASS>UMLClass</BASECLASS>
22                </BASECLASSES>
23                <PARENT>GeoObject</PARENT>
24                <ICON>Point.bmp</ICON>
25          </STEREOTYPE>
26    </STEREOTYPELIST>
27   </BODY>
28</PROFILE>
```

Figura 4.19. Parte do código do GeoProfile na ferramenta CASE *Star UML*

Ao inicializar a ferramenta, o arquivo em que foi escrito o código do perfil é lido e, caso seja carregado corretamente, ele fica à disposição para uso na modelagem de aplicações. A Figura 4.20 ilustra a interface da ferramenta responsável por incluir os perfis UML disponíveis para uso.

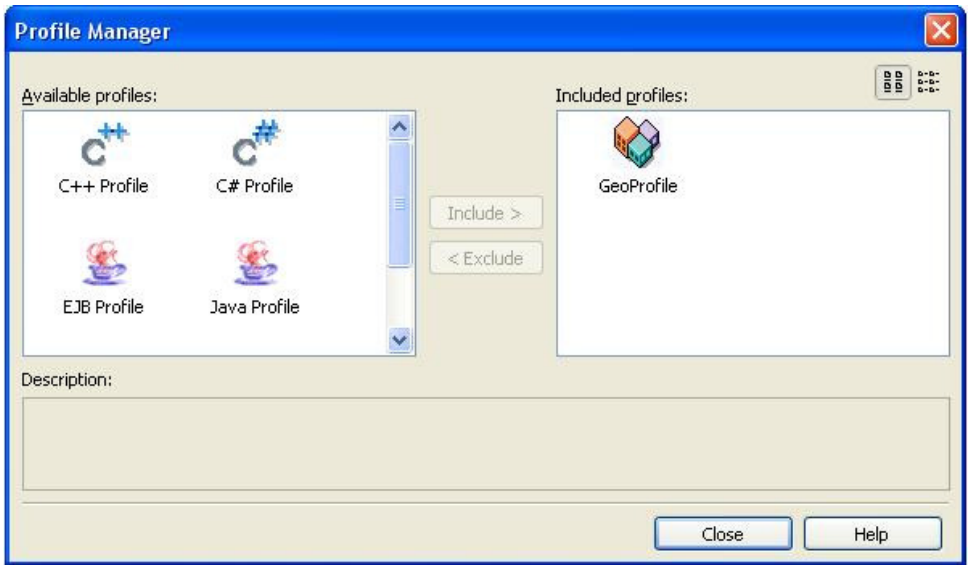


Figura 4.20. Gerenciamento de perfis UML na ferramenta CASE *Star UML*

Para a implementação do GeoProfile foi utilizada a versão 5.0 da ferramenta. Além da baixa usabilidade para implementação de perfis, também não é permitido associar mais de um estereótipo por classe. A ferramenta também não dá suporte à definição de *constraints* na linguagem OCL. Apesar disso, há opção para usar notação gráfica para os estereótipos. Para usar esse recurso, é necessário declarar no código XML o ícone a ser utilizado. Na Figura 4.19, isso é mostrado na linha 24. O arquivo do ícone a ser adicionado precisa estar no mesmo diretório do arquivo do perfil e são aceitos arquivos nos seguintes formatos: .WMF, .EMF e .BMP.

4.4 Considerações finais

A inclusão da notação gráfica para o GeoProfile proporcionou uma forma mais intuitiva de modelar um BDGeo. Assim como na especificação dos estereótipos, a definição dessa notação gráfica levou em consideração os principais modelos conceituais da área.

O teste de implementação do GeoProfile nas ferramentas CASE com suporte a perfis mostrou que é possível aproveitar toda a infra-estrutura da UML para modelar um BDGeo, não necessitando, portanto, de implementar ferramentas próprias para o modelo. Isso diminui a curva de aprendizado do projetista para se trabalhar com o modelo. No apêndice A é apresentado um breve guia prático sobre como criar e utilizar o GeoProfile na ferramenta CASE RSM.

A Tabela 4.1 resume as principais características que foram analisadas, comparando-as com cada uma das ferramentas CASE usadas. Com relação à coluna "Suporte à notação gráfica para estereótipos", todas as ferramentas testadas atenderam a essa característica, porém somente na ferramenta *Visual Paradigm* foi possível utilizar mais de um ícone por classe.

Tabela 4.1 – Comparação entre as ferramentas CASE com suporte a perfis UML

Ferramenta CASE	Licença	Suporte à notação gráfica para estereótipos	Suporte à OCL	Suporte a XMI
Rational Software Modeler	Comercial	Sim	Sim	Sim
Papyrus UML2 Modeler	Livre	Sim	Sim	Não
Visual Paradigm	Comercial	Sim	Não	Sim
Star UML	Livre	Sim	Não	Sim

5 Integração do GeoProfile com os padrões internacionais utilizando a abordagem MDA

O desenvolvimento do GeoProfile teve como principal motivação o fato de poder utilizar a linguagem de modelagem UML, juntamente com todos os seus recursos disponíveis como, por exemplo, as ferramentas CASE para modelar conceitualmente um BDGeo.

Apesar de terem sido levadas em consideração na sua especificação inicial as características dos principais modelos conceituais existentes para aplicações geográficas, não foi considerada a utilização dos padrões internacionais da área. A utilização desses padrões é muito importante para viabilizar a aceitação do GeoProfile pela comunidade científica e pelos projetistas de BDGeo, já que um dos objetivos da elaboração do GeoProfile é torná-lo uma referência na área.

Para suprir essa necessidade, esse capítulo mostra a integração do GeoProfile com os padrões internacionais publicados pelas organizações ISO e OGC. Essa integração é realizada utilizando os diferentes níveis de abstração de modelos da abordagem MDA. É visto também um exemplo de aplicação com transformação de modelos sendo executada de forma automatizada, utilizando a linguagem de transformação de modelos ATL.

5.1 Comparação entre o GeoProfile e os padrões ISO/OGC

Para adequar o perfil aos padrões internacionais, os elementos do GeoProfile são mapeados para elementos dos padrões internacionais, utilizando a abordagem MDA para realizar as transformações. Como visto no capítulo 2, a abordagem MDA separa o desenvolvimento de sistemas em modelos com três níveis de abstração distintos, a saber, o CIM, o PIM e o PSM.

O GeoProfile foi projetado para ajudar os projetistas a trabalharem em um nível de abstração mais alto, auxiliando-os nos primeiros passos de um projeto de BDGeo. Esse nível de abstração, na abordagem clássica de projeto de banco de dados é chamado de nível conceitual, no qual apenas aspectos relacionados ao domínio do problema são tratados, sem lidar com detalhes de implementação. Na abordagem MDA, esse nível mais abstrato é o CIM. Conforme OMG (2003), tal modelo usa um vocabulário familiar aos especialistas do domínio em questão. Um CIM não mostra

detalhes da estrutura dos sistemas, mas o ambiente em que o sistema vai operar, sendo útil para entender o problema.

Esses modelos de níveis de abstração mais altos devem ser transformados em modelos de níveis mais baixos, enriquecidos com elementos de ordem mais técnica, até atingir detalhes de implementação. Na abordagem clássica, essa transformação é chamada de mapeamento conceitual-lógico. É o que ocorre, por exemplo, na transformação de um esquema feito no Modelo ER para o Modelo Relacional. Já na abordagem MDA, um CIM é transformado em um PIM, que é um modelo menos abstrato e apresenta alguns detalhes técnicos, porém ainda independente de alguma plataforma específica.

Um dos principais benefícios da abordagem MDA é o ganho de produtividade no desenvolvimento de sistemas de *software* através da ênfase dada à modelagem e à transformação de modelos de níveis de abstração mais altos para modelos de níveis mais baixos de forma automatizada (KLEPPE; WARMER; BAST, 2003). O projeto de BDGeo pode seguir esses passos. Por exemplo, utilizando ferramentas que dêem suporte às transformações, será possível gerar, a partir do GeoProfile, modelos de níveis mais baixos e, posteriormente, modelos para plataformas específicas.

O GeoProfile atua como um CIM por representar o BDGeo de forma mais abstrata, sem levar em consideração detalhes de implementação como, por exemplo, os detalhes da implementação das características espaciais de um objeto geográfico. Como os construtores básicos da UML não dão suporte à representação do domínio de aplicações geográficas, o GeoProfile utiliza os mecanismos de extensão da UML para representar "quais" são as características espaciais de determinado objeto geográfico e não "como" essas características serão implementadas.

Os padrões internacionais ISO/OGC, analisados no capítulo 3, por apresentarem alguns detalhes técnicos, atuam em um nível de abstração mais baixo, como um PIM. Apesar de ainda estarem em nível conceitual e não apresentarem detalhes de implementação, esses padrões não estão no mesmo nível de abstração do GeoProfile.

Nesta seção é feita, então, a correspondência entre os elementos do GeoProfile e os elementos dos padrões ISO/OGC. Os padrões da série ISO 19100 usados são: o padrão ISO 19107 *Spatial Schema* (ISO/TC211, 2003), que define as características geométricas e topológicas necessárias para descrever as feições geográficas, o padrão ISO 19108 *Temporal Schema* (ISO/TC211, 2002), que trata

das características temporais e o padrão ISO 19123 *Schema for Coverage Geometry and Functions* (ISO/TC211, 2005), que trata das características espaciais de coberturas. Do consórcio internacional OGC será usado o padrão *OpenGIS Simple Feature Access Part1: Common Architecture*, que também é reconhecido pela ISO com a numeração ISO 19125.

Os padrões internacionais usados para fazer as correspondências com os elementos do GeoProfile são os que mais se aproximam dos requisitos para modelagem conceitual de BDGeo. O padrão ISO 19107, por exemplo, foi usado para fazer a correspondência com os estereótipos do GeoProfile que representam os objetos geográficos percebidos na visão de objetos e também com os elementos de redes. Os fenômenos geográficos percebidos na visão de objetos são representados no GeoProfile como especialização da classe *GeoObject*, que são as subclasses *Point*, *Line*, *Polygon* e *ComplexSpatialObj*. A Tabela 5.1 mostra a correspondência desses elementos feita com o padrão ISO 19107.

O padrão ISO 19107 é dividido em duas partes, sendo que na primeira são tratados os aspectos geométricos da informação geográfica e daí a correspondência com os elementos percebidos na visão de objetos. Na segunda parte são tratados os aspectos topológicos, os quais no GeoProfile são representados pelos elementos de rede, que são os estereótipos: *Node*, *Arc*, *UnidirectionalArc* e *BidirectionalArc*. O padrão não faz a distinção entre arcos uni e bidirecionais; ambos podem ser representados pela classe *TP_DirectedEdge*.

Tabela 5.1 – Correspondência entre o GeoProfile e o padrão internacional ISO 19107

Requisitos	GeoProfile	Classes no padrão ISO 19107
Objetos geográficos na visão de objetos	*Point*	*GM_Point*
	Line	*GM_Curve*
	Polygon	*GM_Surface*
	ComplexSpatialObj	*GM_Complex*
Elementos de rede	*Node*	*TP_Node*
	Arc	*TP_Edge*
	UnidirectionalArc	*TP_DirectedEdge*
	BidirectionalArc	*TP_DirectedEdge*

O padrão *OpenGIS Simple Feature Access Part1: Common Architecture*, do consórcio internacional OGC, pode também ser usado para fazer a correspondência com os estereótipos do GeoProfile que representam os objetos geográficos

percebidos na visão de objetos. Esse padrão é uma simplificação do padrão ISO 19107 e cobre apenas geometrias com até duas dimensões. O padrão também não dá suporte a topologias e, por isso, não foi feita correspondência com o GeoProfile para esse requisito. A Tabela 5.2 apresenta a correspondência realizada. No padrão, a classe *Polygon* é uma subclasse de *Surface*. Foi utilizada a subclasse *Polygon* na correspondência pelo fato do conceito estar mais próximo do estereótipo *Polygon* do GeoProfile.

Tabela 5.2 – Correspondência entre o GeoProfile e o *OpenGIS* SFA Parte 1

Requisitos	GeoProfile	*OpenGIS* SFA Parte 1
Objetos geográficos na visão de objetos	*Point*	*Point*
	Line	*LineString*
	Polygon	*Polygon*
	ComplexSpatialObj	*GeometryCollection*

Os objetos geográficos percebidos na visão de campo, os quais no GeoProfile são representados pelos estereótipos *TIN*, *Isolines*, *AdjPolygons*, *GridOfPoints* e *GridOfCells*, nos padrões internacionais foi encontrada correspondência no padrão ISO 19123, o qual trata das características de coberturas, e cujo conceito é semelhante à visão de campo representada pelo GeoProfile. A Tabela 5.3 mostra a correspondência realizada com esse padrão internacional.

Tabela 5.3 – Correspondência entre o GeoProfile e o padrão internacional ISO 19123

Requisitos	GeoProfile	Classes no padrão ISO 19123
Objetos geográficos na visão de campo	*TIN*	*CV_TINCoverage*
	Isolines	*CV_SegmentedCurveCoverage*
	AdjPolygons	*CV_DiscreteSurfaceCoverage*
	GridOfPoints	*CV_DiscreteGridPointCoverage*
	GridOfCells	*CV_GridCell*
	IrregularPoints	*CV_DiscretePointCoverage*

Já os objetos temporais são identificados no GeoProfile pelo estereótipo *TemporalObject*. Os elementos correspondentes se encontram no padrão ISO 19108, que trata dos aspectos temporais da informação geográfica. Com base no diagrama de classes temporais contida no padrão, o tipo *TM_Object* será o elemento correspondente ao estereótipo *TemporalObject*. Esse estereótipo tem ainda um

tagged value chamado *temporalPrimitive*, que pode assumir os valores *Instant* ou *Interval*. Para esses valores existem também os tipos correspondente no padrão ISO 19108 chamados *TM_Instant* e *TM_Interval*, respectivamente. A Tabela 5.4 mostra essa correspondência entre os aspectos temporais do GeoProfile e do padrão ISO 19108.

Tabela 5.4 – Correspondência entre o GeoProfile e o padrão internacional ISO 19108

Requisitos	GeoProfile	Classes no padrão ISO 19108
Objetos temporais	*TemporalObject*	*TM_Object*
	Instant	*TM_Instant*
	Interval	*TM_Period*

Em relação aos relacionamentos espaciais, os quais no GeoProfile são representados pelos estereótipos *Touch*, *In*, *Cross*, *Overlap* e *Disjoint* e estendem a metaclasse *Association*, nos padrões internacionais eles são tratados como métodos de classes. O padrão *OpenGIS* SFA Parte 1 especifica esses métodos, os quais estão relacionados à classe *Geometry*, que é uma classe abstrata e raiz do diagrama de classes apresentado no padrão. Sendo assim, esses métodos são herdados por todas as subclasses instanciáveis de *Geometry*. Todos esses métodos têm como argumentos um atributo do tipo *Geometry* e retorna um valor booleano. A correspondência feita com os estereótipos do GeoProfile é mostrada na Tabela 5.5. Por exemplo, o método *Within* retorna verdadeiro se um objeto geométrico está espacialmente dentro de outro objeto geométrico. No GeoProfile esse tipo de relacionamento espacial é representado pelo estereótipo *In*.

Tabela 5.5 – Correspondência entre o GeoProfile e o padrão *OpenGIS* SFA Parte 1 com relação aos relacionamentos espaciais

Requisitos	GeoProfile	Métodos no padrão *OpenGIS* SFA Parte 1
Relacionamentos espaciais	*Touch*	*Touches (another: Geometry): Boolean*
	In	*Within (another: Geometry): Boolean*
	Cross	*Crosses (another: Geometry): Boolean*
	Overlap	*Overlaps (another: Geometry): Boolean*
	Disjoint	*Disjoint (another: Geometry): Boolean*

5.2 Integração do GeoProfile com a abordagem MDA

As correspondências realizadas na seção anterior podem ser executadas como uma transformação entre um CIM, que é um esquema utilizando o GeoProfile, e um PIM, que é um esquema enriquecido com os elementos dos padrões internacionais. Com a transformação de um esquema modelado com o GeoProfile para um esquema enriquecido com os detalhes técnicos dos padrões internacionais, a geração de um esquema customizado para uma plataforma específica (*Platform Specific Model* ou PSM) e a posterior geração de código-fonte será facilitada. A Figura 5.1 mostra a diferença de níveis de abstração do GeoProfile em relação aos padrões internacionais ISO/OGC, conforme a abordagem MDA. É importante ressaltar que, a partir do PIM, podem ser gerados vários PSMs, sendo que, nesse trabalho será mostrado um exemplo de transformação para o Modelo Objeto-Relacional, conforme ilustrado na Figura 5.1.

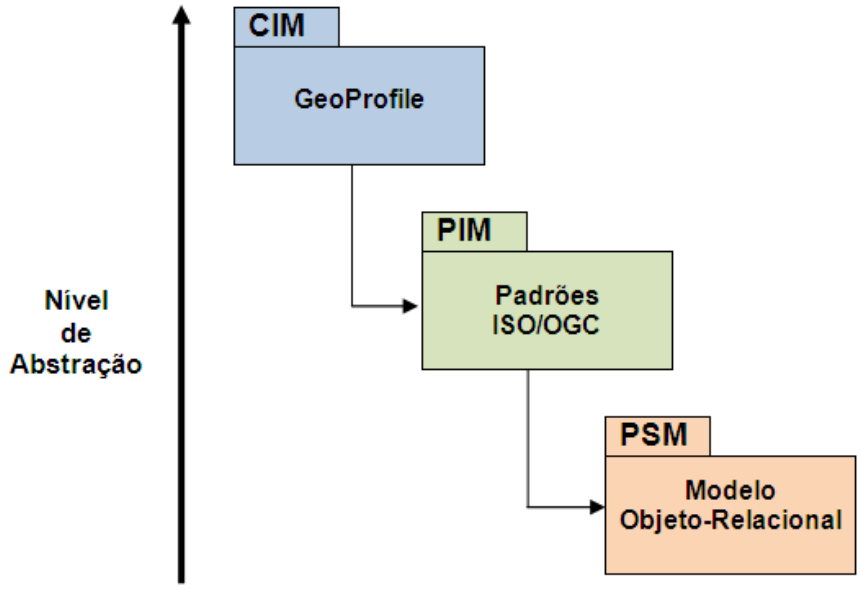


Figura 5.1. Modelos em diferentes níveis de abstração, conforme MDA

Para ilustrar a transformação de modelos, é mostrado um exemplo de esquema modelado com o GeoProfile (nível CIM), o qual é transformado, por meio de regras de transformação executadas com a linguagem ATL, para um esquema enriquecido com os tipos providos pelos padrões da série ISO 19100 (nível PIM), de acordo com as correspondências realizadas na seção anterior. Posteriormente, esse esquema é transformado em um esquema customizado para o Modelo Objeto-

Relacional (nível PSM), o qual pode ser mapeado para um *script* de BDGeo utilizando, por exemplo, o *Oracle Spatial*, o que é mostrado na Figura 5.5.

Por exemplo, os fenômenos percebidos na visão de objetos modelados com o GeoProfile são mapeados para um PIM enriquecido com os padrões ISO da seguinte forma: as classes estereotipadas como *Point* serão mapeadas para uma classe que terá um atributo chamado *geometria* do tipo *GM_Point*. No padrão ISO 19107, *GM_Point* é um tipo de dados básico para objetos com dimensão zero. O mesmo pode ser feito com as outras três classes, *Line*, *Polygon* e *ComplexSpatialObj*, que são mapeadas para uma classe com o atributo *geometria* do tipo *GM_Curve*, *GM_Surface* e *GM_Complex*.

A Figura 5.2 mostra um esquema conceitual modelado com o GeoProfile. O esquema utiliza a notação gráfica para os estereótipos do GeoProfile, a qual foi apresentada no capítulo 4. Neste esquema está sendo utilizada a notação gráfica para os estereótipos `<<Polygon>>` e `<<Point>>`.

O esquema mostra quatro classes, sendo três delas com características espaciais. Nesse nível de abstração foram consideradas apenas "quais" as representações geográficas e não "como" elas serão implementadas, assim como alguns atributos básicos. Portanto, o esquema é um CIM, o qual usa conceitos que estão mais próximos aos usuários finais.

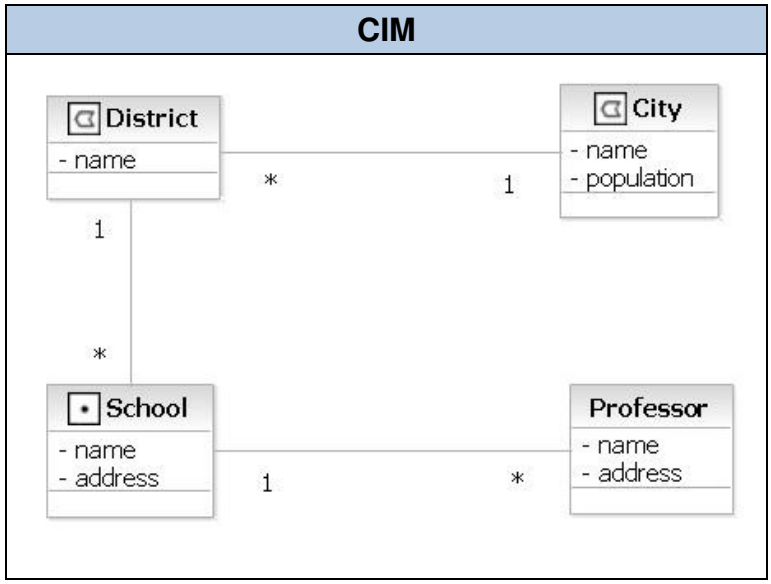


Figura 5.2. Esquema conceitual utilizando o GeoProfile (nível CIM)

Após a construção do CIM usando o GeoProfile, ele deve ser transformado para um PIM, o qual levará em consideração alguns detalhes técnicos que não seriam

desejáveis em um modelo de nível de abstração mais alto e mais próximo do usuário. A Figura 5.3 mostra o PIM resultante dessa transformação. As características espaciais foram transformadas em atributos com os tipos de acordo com a correspondência com os padrões da série ISO 19100. Por exemplo, a classe *City*, que foi modelada com o estereótipo `<<Polygon>>`, passa a ter um atributo geometria, denominado *geometry*, do tipo *GM_Polygon*. O mesmo foi feito com as demais classes que possuem características espaciais.

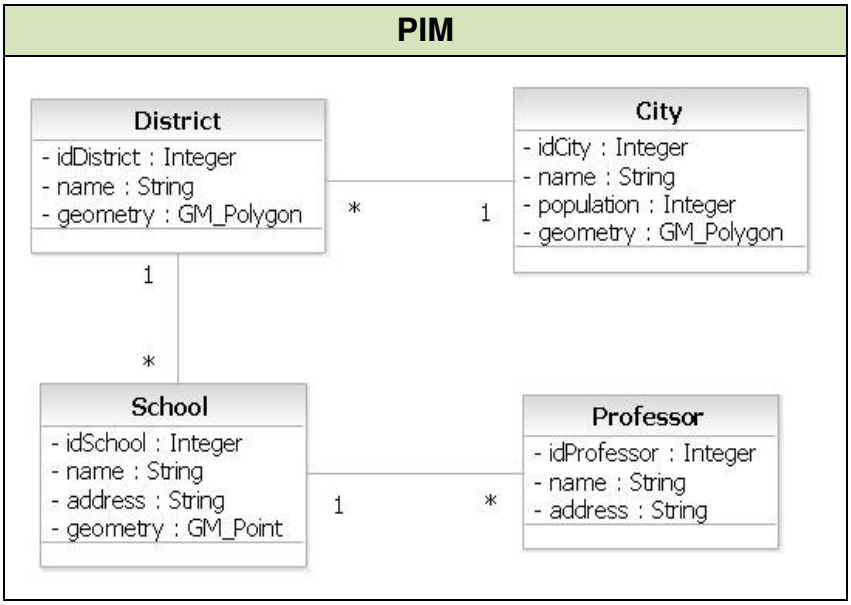


Figura 5.3. Esquema utilizando os padrões da série ISO 19100 (nível PIM)

A próxima etapa é transformar o PIM em um ou mais PSMs. Nesse trabalho é mostrado apenas um exemplo de mapeamento para um modelo de banco de dados objeto-relacional.

O PSM resultante dessa transformação é mostrado na Figura 5.4. Como o esquema de BDGeo gerado com o uso do perfil usa a UML, parte das regras a serem produzidas são semelhantes às usadas no mapeamento conceitual-lógico de bancos de dados convencionais. Nesse caso, as classes foram marcadas com o estereótipo `<<Table>>` para mostrar que elas serão tabelas no modelo objeto-relacional. Alguns atributos também foram marcados com os estereótipos `<<PK>>` e `<<FK>>`, os quais representam as chaves primárias e estrangeiras, respectivamente. O objetivo dessa etapa é deixar o modelo o mais próximo possível da plataforma escolhida para que a geração do script de banco de dados seja facilitada.

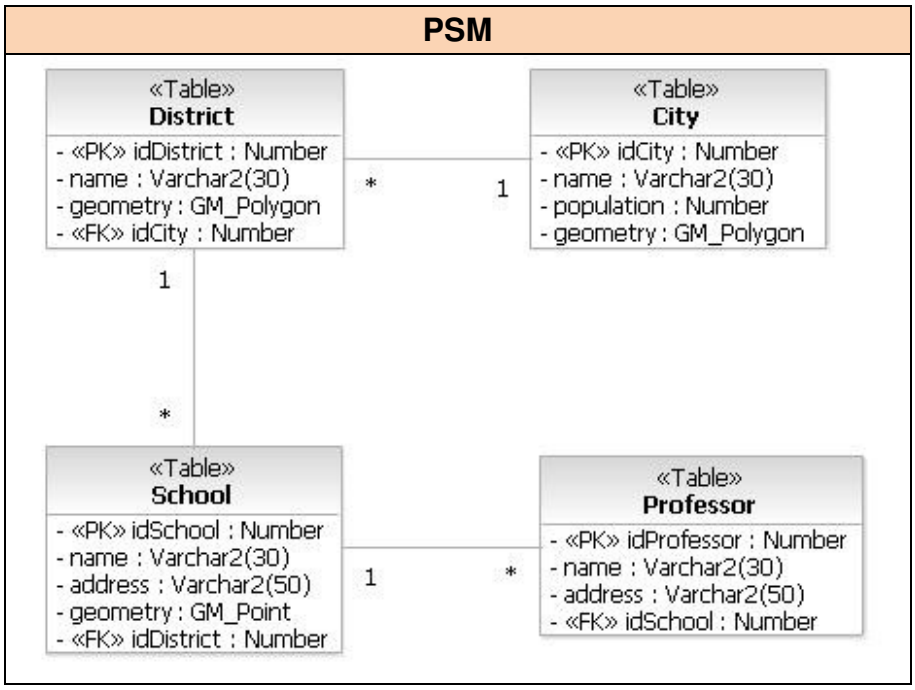


Figura 5.4. Esquema customizado para o Modelo Objeto-Relacional (nível PSM)

Um exemplo do script de BDGeo gerado a partir do PSM da Figura 5.4 e usando o SGBD Oracle Spatial é mostrado na Figura 5.5.

```
CREATE TABLE DISTRICT(
  IDDISTRICT    NUMBER,
  NAME          VARCHAR2(30),
  GEOMETRY      SDO_GEOMETRY,
  IDCITY        NUMBER,
  CONSTRAINT pk_District PRIMARY KEY (IDDISTRICT),
  CONSTRAINT fkDistrict FOREIGN KEY (IDCITY) REFERENCES CITY (IDCITY));

CREATE TABLE CITY(
  IDCITY        NUMBER,
  NAME          VARCHAR2(30),
  POPULATION    NUMBER,
  GEOMETRY      SDO_GEOMETRY,
  CONSTRAINT pk_City PRIMARY KEY (IDCITY));

CREATE TABLE SCHOOL(
  IDSCHOOL      NUMBER,
  NAME          VARCHAR2(30),
  ADDRESS       VARCHAR2(30),
  GEOMETRY      SDO_GEOMETRY,
  IDDISTRICT    NUMBER,
  CONSTRAINT pk_School PRIMARY KEY (IDSCHOOL),
  CONSTRAINT fk_School FOREIGN KEY (IDDISTRICT) REFERENCES DISTRICT (IDDISTRICT));

CREATE TABLE PROFESSOR(
  IDPROFESSOR   NUMBER,
  NAME          VARCHAR2(30),
  ADDRESS       VARCHAR2(30),
  IDSCHOOL      NUMBER,
  CONSTRAINT pk_Professor PRIMARY KEY (IDPROFESSOR),
  CONSTRAINT fk_Professor FOREIGN KEY (IDSCHOOL) REFERENCES SCHOOL (IDSCHOOL));
```

Figura 5.5. *Script* de BDGeo usando o SGBD *Oracle Spatial*

5.3 Definição das regras de transformação de modelos utilizando a linguagem ATL

A realização das transformações de modelos, conforme visto no capítulo 2, pode ser feita por meio de ferramentas que dêem suporte a linguagens de transformação de modelos.

Uma das linguagens de transformação de modelos é a ATL. Essa linguagem permite a definição de regras de transformação de modelos, em que, dado um modelo de entrada, juntamente com regras de transformação, seja gerado um modelo de saída, de acordo com essas regras de transformação. Essa linguagem é utilizada nessa seção para exemplificar a transformação do CIM, apresentado na Figura 5.2, para o PIM, apresentado na Figura 5.3. O exemplo de código mostrado é apenas uma pequena parte do que pode ser feito em relação a essas transformações. Posteriormente, o PIM pode ser transformado no PSM, apresentado na Figura 5.4, o qual é um modelo objeto-relacional de BDGeo, e que pode ser facilmente transformado em um script de definição do BDGeo, como mostrado na Figura 5.5, em que foi utilizada a tecnologia *Oracle Spatial*.

A definição de transformações utilizando a ATL começa com a declaração do módulo de transformação e dos modelos fonte e alvo. O módulo é definido usando a palavra-chave *module* seguido pelo nome do módulo. A palavra-chave *create* indica os modelos fonte e alvo. No código abaixo é mostrado a criação do módulo *GeoProfile2ISO* e a especificação dos modelos de entrada e de saída. É importante destacar que os metamodelos usados nos dois casos é o metamodelo da UML.

```
module GeoProfile2ISO;
create OUT : ISO from IN : GeoProfile;
```

Após essa etapa são definidas as regras de transformação, as quais são construídas para especificar a funcionalidade da transformação. Um módulo ATL é basicamente constituído de um conjunto de regras ATL. Cada regra define a maneira como um elemento de entrada é transformado em um elemento alvo.

Essas regras são escritas utilizando a sintaxe da linguagem, são salvas em arquivos com a extensão *.atl* e podem usar o estilo declarativo ou imperativo. O código apresentado na Figura 5.6 mostra uma das regras de transformação do CIM

para o PIM, apresentados anteriormente. Essa regra é responsável por criar as classes que possuem informação geográfica, que nesse caso são as representadas pelos estereótipos do GeoProfile e por criar os elementos que não continham no CIM como, por exemplo, o atributo *geometry*, cujo tipo será o correspondente no padrão ISO. O apêndice C apresenta o restante do código das transformações efetuadas neste exemplo, usando a linguagem ATL. Neste trabalho é mostrada apenas uma parte das transformações para ilustrar o exemplo mostrado anteriormente. Por exemplo, na Figura 5.6 estão sendo considerados apenas os dois estereótipos contidos no exemplo, a saber, o estereótipo *Point* e *Polygon*. No entanto, essas regras de transformação podem ser estendidas para todos os outros estereótipos do perfil.

Com a aplicação da transformação, o modelo de saída é gerado no formato XMI (XML *Metadata Interchange*), que é um formato padrão para intercâmbio de modelos e que pode ser importado pela maioria das ferramentas CASE com suporte à UML2.

```
1 rule stereotypedClass{
2   from
3       input : GeoProfile!Class(
4           not thisModule.emptyGeometry(input.stereotype))
5   to
6       output : ISO!Class(
7          name <- input.name,
8                reference <- input.reference ->
9          collect( e | thisModule.getReferences(e) ).asSet(),
10                attribute <- input.attribute ->
11         collect( e | thisModule.getAttributes(e) ).asSet(),
12                attribute <- id,
13                attribute <- geometry
14           ),
15     id : ISO!Attribute(
16        name <- 'id' + input.name,
17        type <- thisModule.integerDataType()
18     ),
19     geometry : ISO!Attribute(
20        name <- input.name + 'Geometry',
21        type <- if( thisModule.isPolygon( input.stereotype )) then
22                   thisModule.polygonDataType()
23                else
24                   thisModule.pointDataType()
25                endif
26     )
27 }
```

Figura 5.6. Parte do código da transformação do CIM para o PIM

6 Conclusões e trabalhos futuros

Neste capítulo são apresentadas as conclusões desta dissertação e também são propostos alguns trabalhos a serem realizados futuramente.

6.1 Contribuições deste trabalho

O desenvolvimento do GeoProfile teve como principal motivação o fato de poder utilizar a linguagem de modelagem UML, juntamente com todos os seus recursos disponíveis como, por exemplo, as ferramentas CASE, para modelar conceitualmente um BDGeo. O GeoProfile reúne na sua definição os principais requisitos para aplicações geográficas e possui características dos principais modelos existentes.

O primeiro objetivo deste trabalho foi dar continuidade à especificação do GeoProfile, inicialmente proposto por Sampaio (2009), criando uma notação gráfica para os estereótipos, as quais tornam a modelagem de BDGeo mais intuitiva e de fácil entendimento para o usuário final. O capítulo 4 mostrou a proposta de notação gráfica para os estereótipos do GeoProfile. Assim como para a especificação da proposta inicial do perfil, essa notação gráfica teve como base os principais modelos conceituais da área, os quais em sua maioria também fazem uso de ícones gráficos para modelar um BDGeo. A notação gráfica para o GeoProfile foi testada na ferramenta CASE RSM, onde foram modelados alguns exemplos de esquemas conceituais.

O segundo objetivo foi testar a implementação do perfil em outras ferramentas CASE que dão suporte a perfis, já que sua versão inicial foi implementada apenas na ferramenta RSM (SAMPAIO, 2009). O capítulo 4 também mostrou uma análise de algumas ferramentas CASE que dão suporte a perfis. Contando com a ferramenta RSM, o GeoProfile foi implementado em duas ferramentas comerciais e duas com licença livre. A tendência é que as ferramentas CASE, de uma forma geral, passem a dar suporte a esse mecanismo de extensão da UM, proporcionando, assim, um maior número de opções para o projetista.

O terceiro objetivo foi investigar padrões internacionais voltados para informação geográfica e que podem auxiliar na modelagem de BDGeo. Foram pesquisados os padrões internacionais publicados pelas organizações ISO e OGC. O

Comitê Técnico ISO/TC 211 é o responsável pela série ISO 19100, cujos padrões estão relacionados à informação geográfica. O capítulo 3 apresentou alguns desses padrões que podem ser usados na modelagem de BDGeo.

O quarto objetivo deste trabalho foi adequar o GeoProfile aos padrões internacionais. A utilização desses padrões é muito importante para viabilizar a aceitação do GeoProfile pela comunidade científica e pelos projetistas de BDGeo. O capítulo 5 mostrou a correspondência entre o GeoProfile e os padrões internacionais ISO/OGC. Utilizando a abordagem MDA, foi possível mostrar a diferença de níveis de abstração entre o GeoProfile e os padrões internacionais.

O último objetivo deste trabalho foi criar regras de transformação de esquemas conceituais de dados elaborados com base no GeoProfile, para esquemas compatíveis com as especificações ISO/TC 211 e OGC utilizando a abordagem MDA. Em relação a esse objetivo, foi estudada a linguagem de transformação de modelos ATL, na qual foi implementado um exemplo de transformação. Essa automatização das transformações constitui um dos principais benefícios da abordagem MDA. O capítulo 5 apresentou esse exemplo e o restante do código da transformação encontra-se no apêndice C.

Assim, é possível concluir que os objetivos propostos para essa dissertação foram alcançados e, dessa forma, mais um passo foi dado na tarefa de padronização da modelagem de BDGeo.

6.2 Trabalhos futuros

Como trabalhos futuros, pode-se citar a definição de transformações para modelos de plataformas específicas, os quais não foram tratados nessa dissertação e a geração de código como, por exemplo, a estrutura do BDGeo utilizando a linguagem SQL.

Outro trabalho a ser explorado consiste em estender o perfil GeoProfile para dar suporte a bases de dados tridimensionais e a investigação de outros padrões internacionais que possam ser utilizados para enriquecer a modelagem de aplicações geográficas.

APÊNDICE A

A.1 Implementação e uso do GeoProfile na ferramenta CASE *Rational Software Modeler*

Este apêndice apresenta um breve guia prático sobre a implementação e uso do GeoProfile na ferramenta CASE *Rational Software Modeler* (RSM), utilizando, para isso, a versão 7.5.5 da ferramenta. Informações mais detalhadas sobre a criação e uso perfis UML com a ferramenta RSM podem ser encontradas em Misic (2010).

A criação de perfis UML na ferramenta RSM é realizada de forma bastante intuitiva. Para criar um novo projeto de perfil, basta clicar em: *Arquivo* → *Novo* → *Projeto* → *Modelagem* → *Extensibilidade da UML* → *Projeto de Perfil UML*. Uma janela é então aberta para definir o nome do projeto. Ao clicar em *Avançar* uma nova janela é aberta, como ilustrado na Figura A.1, para se adicionar o nome do perfil e outras informações sobre o mesmo.

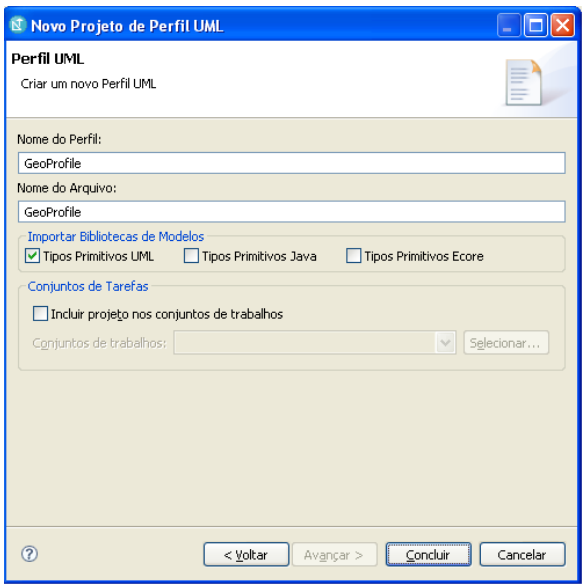


Figura A.1. Criando um novo perfil

Feito isso, basta clicar em *Concluir* e o projeto com o perfil será criado e estará disponível para manipulação no "*Explorador de Projeto*" da ferramenta (Figura A.2).

A seguir, devem ser adicionados os elementos da UML ao perfil. Para isso, basta selecionar o perfil, como mostrado na Figura A.2 e clicar com o botão direito do mouse nele e depois em: *Incluir Diagrama* → *Diagrama de Classes*.

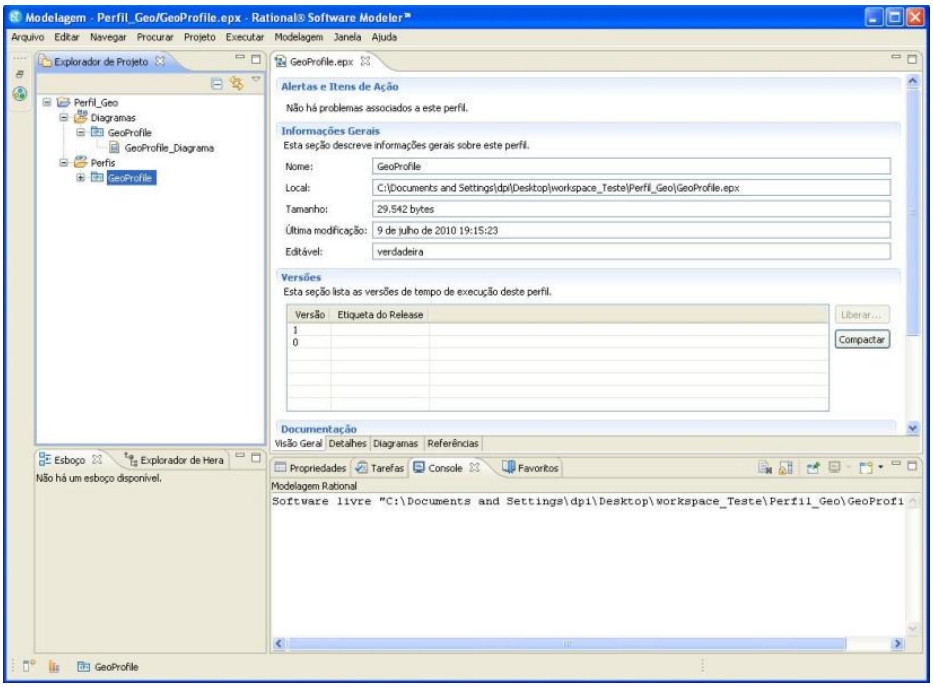


Figura A.2. Visualização do perfil criado

Com o diagrama de classes criado, é possível adicionar, de forma visual, os elementos de extensão da UML ao perfil como, por exemplo, os estereótipos, metaclasses e *tagged values*. A Figura A.3 ilustra alguns estereótipos do GeoProfile sendo adicionados ao diagrama de classes.

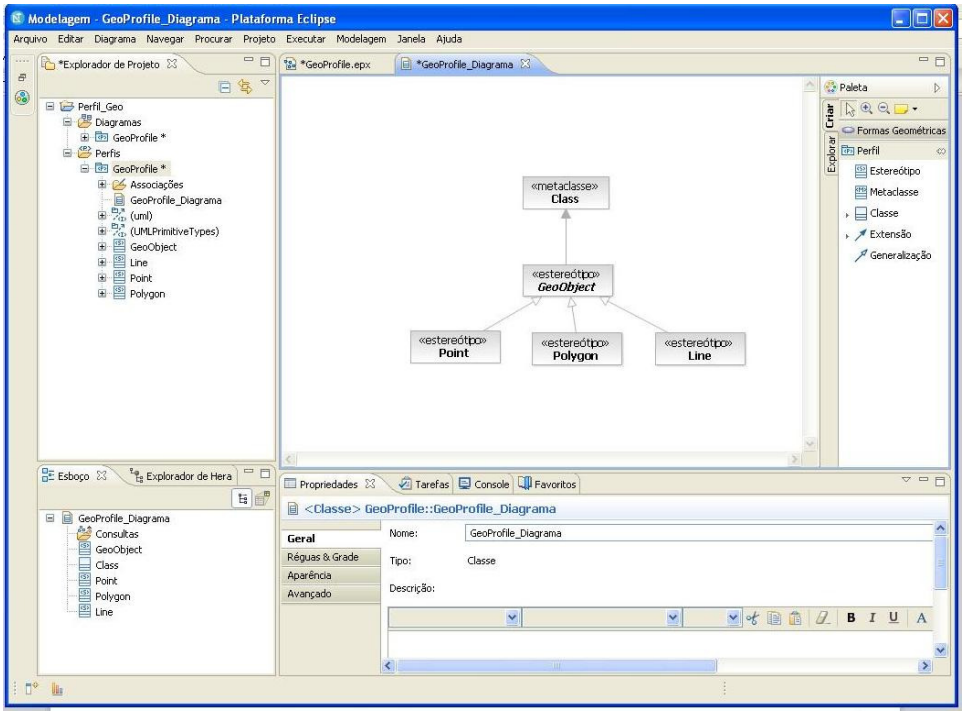


Figura A.3. Elementos sendo adicionados ao perfil

As *constraints*, definidas na linguagem OCL, também podem ser facilmente adicionadas ao GeoProfile. Para isso, basta selecionar o estereótipo no "*Explorador de Projeto*", clicar com o botão direito nele e depois em*: Incluir UML* → *Restrição*. Esse passo é ilustrado na Figura A.4.

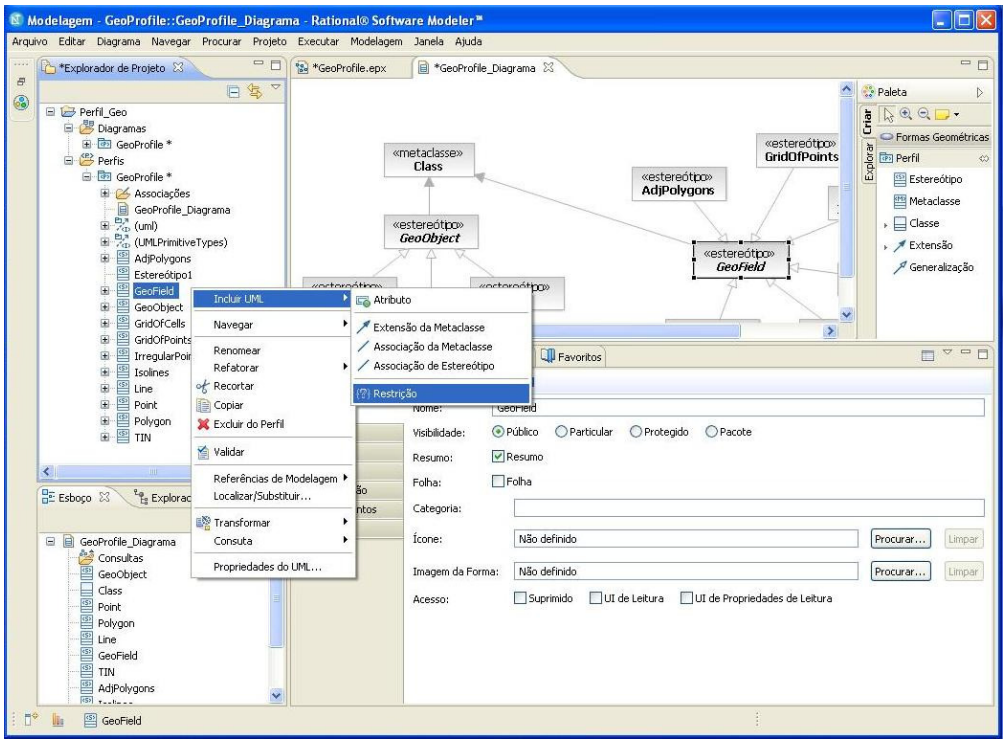


Figura A.4. Criação de *constraints*

Com a restrição criada, é possível alterar suas propriedades na guia *Geral* como, por exemplo, a definição do nome, a escolha da linguagem (nesse caso será a OCL) e o seu conteúdo propriamente dito. Esse passo é ilustrado na Figura A.5. Depois da especificação de todos os elementos do GeoProfile, ele poderá, então, ser utilizado na modelagem de banco de dados geográficos. Os arquivos relacionados a perfis na ferramenta serão salvos utilizando a extensão .EPX.

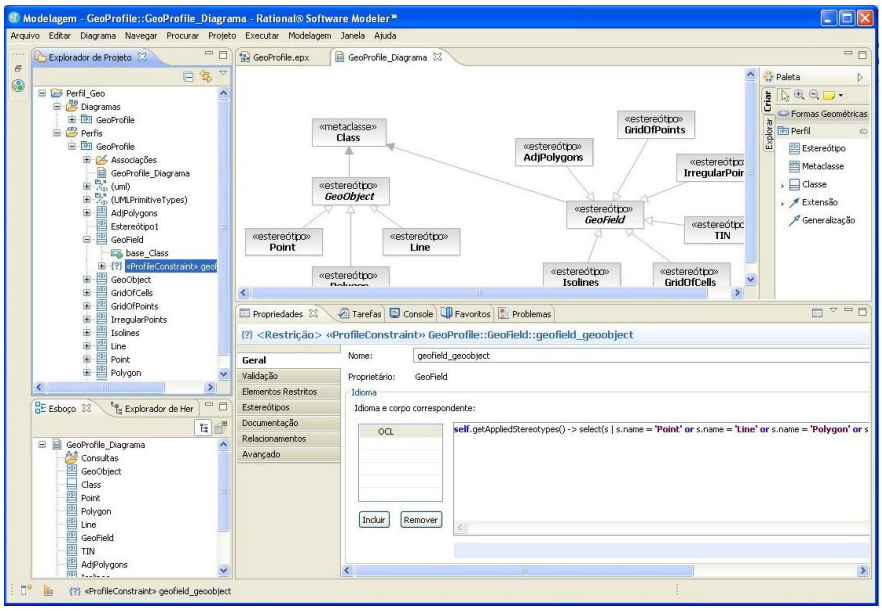


Figura A.5. Definição das *constraints*

Para aplicar o GeoProfile a um modelo, basta selecionar o modelo ir até a guia *Propriedades*, clicar em "*Adicionar Perfil...*" e depois, selecionar o perfil no espaço de trabalho. Feito isso, os elementos adicionados ao modelo poderão fazer uso dos estereótipos e *constraints* do GeoProfile. Ao incluir uma classe no modelo, por exemplo, basta ir à guia *Propriedades*, clicar em *Estereótipos* e depois em "*Aplicar Estereótipos...*". Será aberta uma janela com todos os estereótipos disponíveis. Basta selecionar os desejados e clicar em *OK*. O estereótipo será, então, aplicado à classe correspondente. Esse passo é ilustrado na Figura A.6.

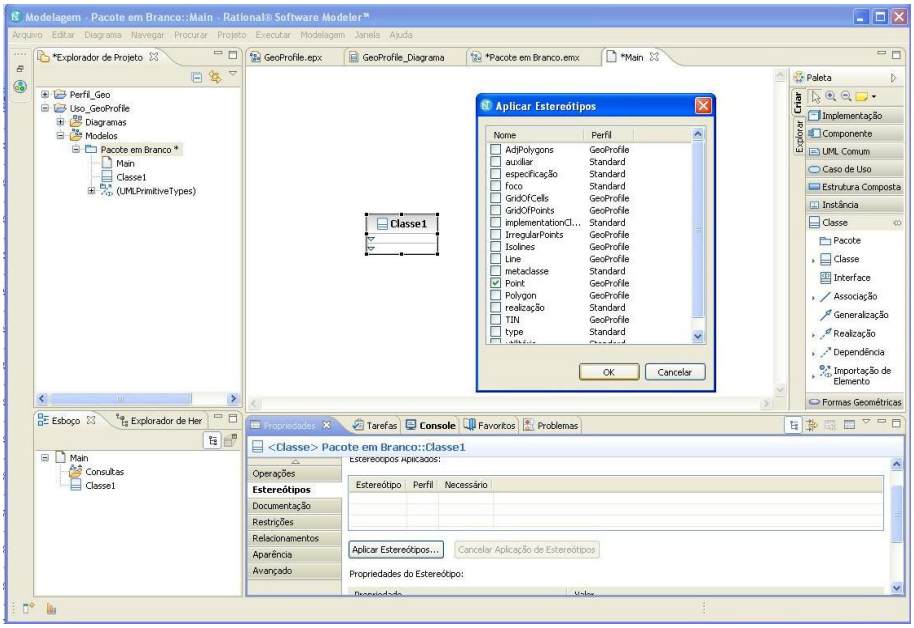


Figura A.6. Selecionando os estereótipos a serem usados no modelo

Com o estereótipo aplicado à classe, é possível alterar algumas propriedades como, por exemplo, a forma de visualização do mesmo. A ferramenta disponibiliza algumas formas de visualização de estereótipos, a saber: Decoração e Texto, Decoração, Imagem da Forma, Texto e Nenhum. Para realizar isso, basta selecionar a classe, clicar com o botão direito do mouse nela e depois em*: Filtros → Estilo de Estereótipo e Visibilidade* e depois selecionar a forma desejada. Esse passo é ilustrado na Figura A.7.

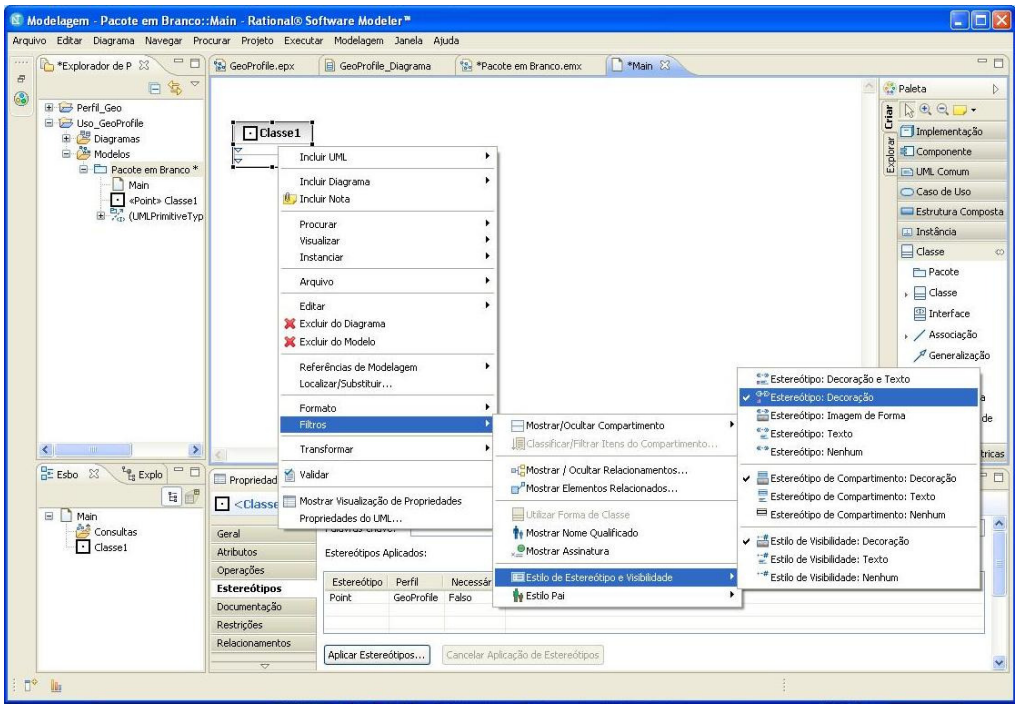


Figura A.7. Escolhendo a forma de visualização de estereótipos

Dessa forma, então, o GeoProfile pode ser usado na ferramenta CASE RSM para modelar um BDGeo. Maiores detalhes sobre a sobre a criação e uso perfis UML com a ferramenta RSM podem ser encontradas em Misic (2010).

APÊNDICE B

B.1 Padrões publicados pelo Comitê Técnico ISO/TC 211

Este apêndice lista os padrões da série ISO 19100 publicados pelo Comitê Técnico ISO/TC 211. Os padrões foram separados em grupos específicos e são mostrados de acordo com estes grupos. Maiores detalhes sobre estes padrões podem ser encontrados em ISO/TC211 (2009).

INFRASTRUCTURE STANDARDS
ISO 19101:2002 Reference Model
ISO/TS 19103:2005 Conceptual Schema Language
ISO/TS 19104:2008 Terminology
ISO 19105:2000 Conformance and testing
ISO 19106:2004 Profiles

DATA MODEL STANDARDS
ISO 19109:2005 Rules for application schema
ISO 19107:2003 Spatial schema
ISO 19123:2005 Schema for coverage geometry and functions
ISO 19108:2002 Temporal schema
ISO 19141:2008 Schema for moving features
ISO 19137:2007 Core profile of the spatial schema

GEOGRAPHIC INFORMATION MANAGEMENT STANDARDS
ISO 19110:2005 Methodology for feature cataloguing
ISO 19111:2007 Spatial referencing by coordinates
ISO 19112:2003 Spatial referencing by geographic identifiers
ISO 19113:2002 Quality principles
ISO 19114:2003 Quality evaluation procedures
ISO 19115:2003 Metadata
ISO 19131:2007 Data product specifications
ISO 19135:2005 Procedures for item registration
ISO/TS 19127:2005 Geodetic codes and Parameters
ISO/TS 19138:2006 Data quality measures

GEOGRAPHIC INFORMATION SERVICES STANDARDS
ISO 19119:2005 Services
ISO 19116:2004 Positioning services
ISO 19117:2005 Portrayal
ISO 19125-1:2004 Simple feature access — Part 1: Common architecture
ISO 19125:2004 Simple feature access — Part 2: SQL option
ISO 19128:2005 Web map server interface
ISO 19132:2007 Location based services — Reference model
ISO 19133:2005 Location based services — Tracking and navigation
ISO 19134:2007 Location based services — Multimodal routing and navigation

GEOGRAPHIC INFORMATION ENCODING STANDARDS
ISO 19118:2005 Encoding
ISO 19136:2007 Geography Markup Language (GML)
ISO/TS 19139:2007 Metadata — XML schema implementation

STANDARDS FOR SPECIFIC THEMATIC AREAS
ISO/TS 19101-2:2008 Reference model — Part 2: Imagery
ISO/TS 19115-2:2008 Metadata — Part 2: Extensions for imagery and gridded data

APÊNDICE C

C.1 Regras de transformação CIM para PIM usando ATL

Esse apêndice apresenta o código criado para realizar as transformações do exemplo de aplicação mostrado no capítulo 5. As regras de transformação foram criadas usando a linguagem ATL. Está sendo mostrada apenas a parte que concerne ao exemplo. No entanto, essas regras podem ser estendidas para todo o perfil.

```
module GeoProfile2ISO;
create OUT : ISO from IN : GeoProfile;

helper def : emptyGeometry( str : GeoProfile!Stereotype ): Boolean =
      str -> collect( e | e.name ) -> isEmpty();

helper def : isPolygon( str : GeoProfile!Stereotype ) : Boolean =
      str -> collect( e | e.name ) -> includes( 'Polygon' );

rule stereotypedClass{
   from
      input : GeoProfile!Class(
         not thisModule.emptyGeometry( input.stereotype ) )
   to
      output : ISO!Class(
         name <- input.name,
         reference <- input.reference ->
            collect( e | thisModule.getReferences(e) ).asSet(),
         attribute <- input.attribute ->
            collect( e | thisModule.getAttributes(e) ).asSet(),
         attribute <- id,
         attribute <- geometry
       ),

      id : ISO!Attribute(
         name <- 'id' + input.name,
         type <- thisModule.integerDataType()
      ),

      geometry : ISO!Attribute(
         name <- input.name + 'Geometry',
         type <- if( thisModule.isPolygon( input.stereotype ) ) then
                     thisModule.polygonDataType()
                 else
                     thisModule.pointDataType()
                 endif
      )
}

rule nonStereotypedClass{
   from
      input : GeoProfile!Class( thisModule.emptyGeometry(
input.stereotype ) )
   to
      output : ISO!Class(
         name <- input.name,
```

```
            reference <- input.reference ->
                collect( e | thisModule.getReferences(e) ).asSet(),
            attribute <- input.attribute ->
                collect( e | thisModule.getAttributes(e) ).asSet(),
            attribute <- id
        ),

        id : ISO!Attribute(
            name <- 'id' + input.name,
            type <- thisModule.integerDataType()
        )
}

unique lazy rule getAttributes{
    from
        input : GeoProfile!Attribute
    to
        output : ISO!Attribute(
            name <- input.name,
            type <- if( input.name = 'population' ) then
                        thisModule.integerDataType()
                    else
                        thisModule.stringDataType()
                    endif
        )
}

unique lazy rule getReferences{
    from
        input : GeoProfile!Reference
    to
        output : ISO!Reference(
            name <- input.name,
            lowerBound <- input.lowerBound,
            upperBound <- input.upperBound,
            type <- input.type
        )
}

unique lazy rule polygonDataType{
    from
        input : GeoProfile!Class
    to
        output: ISO!DataType(
            name <- 'GM_Surface'
        )
}

unique lazy rule pointDataType{
    from
        input : GeoProfile!Class
    to
        output: ISO!DataType(
            name <- 'GM_Point'
        )
}


unique lazy rule integerDataType{
    from
        input : GeoProfile!Class
```

```
   to
      output: ISO!DataType(
         name <- 'Integer'
      )
}

unique lazy rule stringDataType{
   from
      input : GeoProfile!Class
   to
      output: ISO!DataType(
         name <- 'String'
      )
}
```

A seguir é mostrado o modelo de entrada e o modelo de saída gerado com a aplicação das regras acima. Esses modelos são incluídos no formato XMI e se referem aos esquemas apresentados nas Figuras 5.2 e 5.3, respectivamente.

Modelo de entrada

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0"
xmlns:xmi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="GeoProfile">
      <Class name="City">
            <attribute name="cityName"/>
            <attribute name="population"/>
            <reference name="linkTodistrict" lowerBound="0"
upperBound="-1" type="/1"/>
            <stereotype name="Polygon"/>
      </Class>
      <Class name="District">
            <attribute name="districtName"/>
            <reference name="city" lowerBound="1" upperBound="1"
type="/0"/>
            <reference name="linkToSchool" lowerBound="0"
upperBound="-1" type="/2"/>
            <stereotype name="Polygon"/>
      </Class>
      <Class name="School">
            <attribute name="schoolName"/>
            <attribute name="schoolAddress"/>
            <reference name="district" lowerBound="1" upperBound="1"
type="/1"/>
            <reference name="linkToProfessor" lowerBound="0"
upperBound="-1" type="/3"/>
            <stereotype name="Point"/>
      </Class>
      <Class name="Professor">
            <attribute name="professorName"/>
            <attribute name="professorAddress"/>
            <reference name="school" lowerBound="1" upperBound="1"
type="/2"/>
      </Class>
</xmi:XMI>
```

Modelo de saída

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns="ISO">
  <Class name="City" attribute="/12 /14 /1 /2" reference="/11"/>
  <Attribute name="idCity" owner="/0" type="/15"/>
  <Attribute name="CityGeometry" owner="/0" type="/16"/>
  <Class name="District" attribute="/19 /4 /5" reference="/18 /17"/>
  <Attribute name="idDistrict" owner="/3" type="/15"/>
  <Attribute name="DistrictGeometry" owner="/3" type="/16"/>
  <Class name="School" attribute="/23 /22 /24 /7 /8" reference="/20
/21"/>
  <Attribute name="idSchool" owner="/6" type="/15"/>
  <Attribute name="SchoolGeometry" owner="/6" type="/25"/>
  <Class name="Student" attribute="/27 /28 /29 /10"
reference="/26"/>
  <Attribute name="idProfessor" owner="/9" type="/15"/>
  <Reference name="linkTodistrict" lowerBound="0" upperBound="-1"
type="/3"/>
  <Attribute name="cityName" owner="/0" type="/13"/>
  <DataType name="String"/>
  <Attribute name="population" owner="/0" type="/15"/>
  <DataType name="Integer"/>
  <DataType name="GM_Surface"/>
  <Reference name="city" lowerBound="1" upperBound="1" type="/0"/>
  <Reference name="linkToSchool" lowerBound="0" upperBound="-1"
type="/6"/>
  <Attribute name="districtName" owner="/3" type="/13"/>
  <Reference name="district" lowerBound="1" upperBound="1"
type="/3"/>
  <Reference name="linkToProfessor" lowerBound="0" upperBound="-1"
type="/9"/>
  <Attribute name="schoolName" owner="/6" type="/13"/>
  <Attribute name="contact" owner="/6" type="/13"/>
  <Attribute name="schoolAddress" owner="/6" type="/13"/>
  <DataType name="GM_Point"/>
  <Reference name="school" lowerBound="1" upperBound="1" type="/6"/>
  <Attribute name="professorName" owner="/9" type="/13"/>
  <Attribute name="professorAddress" owner="/9" type="/13"/>
</xmi:XMI>
```

Referências bibliográficas

ALHIR, S.S. **Learning UML**. Sebastopol: O'Reilly&Associates, Inc., 2003. 252p.

ATLAS TRANFORMATION LANGUAGE. **ATL Project**. Disponível em: < http://www.eclipse. org/atl/ >. Acesso em: 01 mar. 2010.

BÉDARD, Y. Visual modeling of spatial databases: towards spatial PVL and UML. **Geomatica**, v. 53, n. 2, p. 169-185, 1999.

BÉDARD, Y.; LARRIVÉE, S. Modeling with Pictogrammic Languages. In: Shekhar, S.; Xiong, H. (Eds.). **Encyclopedia of GIS**. Berlin: Springer-Verlag, 2008. p. 716-725.

BÉDARD, Y.; PAQUETTE, F. Extending Entity/Relationship Formalism for Spatial Information Systems. In: INTERNATIONAL SYMPOSIUM ON COMPUTER-ASSISTED CARTOGRAPHY (AUTO-CARTO), 9, 1989, Baltimore, United States. **Proceedings...** Baltimore: ACSM/ASPRS, 1989. p. 818-827.

BEZERRA, E. **Princípios de análise e projeto de sistemas com UML**. Rio de Janeiro: Elsevier, 2002. 320p.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML: guia do usuário**. 2. ed. Rio de Janeiro: Elsevier, 2005. 474p.

BOOCH, G. **The booch method: process and pragmatics**. Santa Clara: Calif.: Rational, 1992.

BORGES, K. A. V.; DAVIS Jr., C. A.; LAENDER, A. H. F. OMT-G: An Object-Oriented Data Model for Geographic Applications. **GeoInformatica**, v.5, n.3, p. 221-260, set. 2001.

BRODEUR, J.; BADARD, T. Modeling with ISO 191xx Standards. In: Shekhar, S.; Xiong, H. (Eds.). **Encyclopedia of GIS**. Berlin: Springer-Verlag, 2008. p. 705-716.

BRODEUR, J.; BÉDARD, Y; PROULX, M.J. Modeling geospatial application databases using UML-based repositories aligned with international Standards in geomatics. In: ACM INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, 8, 2000, Washington, United State. **Proceedings…** New York: ACM, 2000. p. 39-46.

CASANOVA, M. et al. **Banco de Dados Geográficos**. Curitiba: MundoGeo, 2005. 506p.

CHRISMAN, N. **Exploring Geographical Information Systems**. 2. ed. New York: John Wiley & Sons, 2001. 320p.

CLEMENTINI, E.; DI FELICE, P.; OOSTEROM, P. A Small Set of Formal Topological Relationships Suitable for End-User Interaction. In: INTERNATIONAL SYMPOSIUM ON ADVANCES IN SPATIAL DATABASES, 3, 1993, Singapore. **Proceedings…** London: Springer-Verlag, 1993. p. 277-295.

ELMASRI, R.; NAVATHE, S. B. **Fundamentals of database systems**. 4. ed. Boston: Addison-Wesley, 2003. 1009p.

ERIKSSON, H. et al. **UML 2 Toolkit**. Indianapolis: Wiley Publishing, 2004. 552p.

FRANKEL, D. S. **Model Driven Architecture: Applying MDA to Enterprise Computing**. Indianapolis: Wiley Publishing, 2003. 352p.

FRIIS-CHRISTENSEN, A.; TRYFONA, N.; JENSEN, C.S. Requirements and Research Issues in Geographic Data Modeling. In: ACM INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, 9, 2001, Atlanta, Georgia, USA. **Proceedings...** New York: ACM, 2001. p. 2-8.

FUENTES, L.; VALLECILLO, A. An Introduction to UML Profiles. **UPGRADE, The European Journal for the Informatics Professional**, v. 5, n. 4, p. 5-13, abr. 2004.

GOODCHILD, M. F.; YUAN, M.; COVA, T. J. Towards a general theory of geographic representation in GIS. **International Journal of Geographic Information Science**, v. 21, n. 3, p. 239-260, jan. 2007.

ISO/TC211. **Standards guide**. Geneva, Switzerland, 2009. Disponível em: <http://www.isotc211.org/>. Acesso em: 01 nov. 2009.

ISO/TC211. **ISO 19123:2005 Geographic Information – Schema for Coverage Geometry and Functions**. Geneva, Switzerland, 2005.

ISO/TC211. **ISO 19107:2003 Geographic Information – Spatial Schema**. Geneva, Switzerland, 2003.

ISO/TC211. **ISO 19108:2002 Geographic Information – Temporal Schema**. Geneva, Switzerland, 2002.

JACOBSON, I. et al. **Object-oriented software engineering: a use case driven approach**. Boston: Addison-Wesley, 1992. 552p.

JENSEN, C.S et al. A consensus glossary of temporal database concepts. **ACM SIGMOD Record**, v. 23, n. 1, p. 52-64, mar. 1994.

JOUAULT, F. AND KURTEV, I. Transforming models with ATL. In: MODEL TRANSFORMATIONS IN PRACTICE WORKSHOP (MoDELS'05), Montego Bay, Jamaica. **Proceedings...** Berlin: Springer-Verlag, 2005. p. 128-138.

KLEPPE, A.; WARMER, J.; BAST, W. **MDA Explained: The Model Driven Architecture: Practice and Promise**. 2 ed. Boston: Addison-Wesley, 2003. 192p.

KÖSTERS, G.; PAGEL, B.; SIX, H. GIS-Application Development with GeoOOA. **International Journal of Geographical Information Science**, v. 11, n. 4, p. 307-335, jan. 1997.

KRESSE, W.; FADAIE, K. **ISO Standards for Geographic Information**. Berlin: Springer, 2004. 322p.

LAUDON, K. C.; LAUDON, J. P. **Sistemas de informação gerenciais**. 7.ed. São Paulo: Pearson Prentice-Hall, 2007. 452p.

LISBOA FILHO, J.; STEMPLIUC, S.M. Modeling spatial constraints in conceptual database design of network applications. In: URBAN AND REGIONAL DATA MANAGEMENT (UDMS), 27, 2009, Ljubljana, Slovenia. **Proceedings...** London : Taylor & Francis, 2009, p. 185-193.

LISBOA FILHO, J.; IOCHPE, C. Modeling with a UML Profile. In: Shekhar, S.; Xiong, H. (Eds.). **Encyclopedia of GIS**. Berlin: Springer-Verlag, 2008. p. 691-700.

LISBOA FILHO, J.; IOCHPE, C.; BORGES, K.A.V. Analysis patterns for GIS Data Schema reuse on urban management applications. **CLEI Electronic Journal**, Merida, Venezuela, v. 5, n. 2, p. 1-15, jan. 2002.

LISBOA FILHO, J.; IOCHPE, C.; HASENACK, H.; WEBER, E.J. Modelagem conceitual de banco de dados geográficos: o estudo de caso do projeto PADCT/CIAMB. In: UFRGS/CENTRO DE ECOLOGIA. **Energia e Meio Ambiente**. Porto Alegre: Editora da Universidade, 2000.

LISBOA FILHO, J.; IOCHPE, C. Um estudo sobre modelos conceituais de dados para projeto de bancos de dados geográficos. **Revista IP-Informática Pública**, Belo Horizonte, v. 1, n. 2, p. 67-90, dez. 1999.

MISIC, D. **Authoring UML Profiles: Part 1. Using Rational Software Architect, Rational Systems Developer, and Rational Software Modeler to create and deploy UML Profiles**. Disponível em: <www.ibm.com/developerworks/rational/library/08/0429_misic1/index.html>. Acesso em: 01 fev. 2010.

OBJECT MANAGEMENT GROUP. **Unified Modeling Language: Superstructure**, v. 2.1.2, OMG Document formal/2007-11-02 edition. Needham, MA, USA, 2007.

OBJECT MANAGEMENT GROUP. **MDA Guide**, v.1.0.1, OMG Document formal/2003-06-01 edition. Needham, MA, USA, 2003.

OPEN GEOSPATIAL CONSORTIUM INC. **OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture**, v. 1.2.0, OpenGIS® Implementation Specification/2006-10-05. Wayland, Massachusetts, USA, 2006a.

OPEN GEOSPATIAL CONSORTIUM INC. **OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option**, v.1.2.0, OpenGIS® Implementation Specification/2006-10-05. Wayland, Massachusetts, USA, 2006b.

PARENT, C.; SPACCAPIETRA, S.; ZIMÁNYI, E. Modeling and Multiple Perceptions. In: Shekhar, S.; Xiong, H. (Eds.). **Encyclopedia of GIS**. Berlin: Springer-Verlag, 2008. p. 682-690.

PAPYRUS UML. Disponível em: <www.papyrusuml.org/>. Acesso em: 01 fev. 2010.

PARENT, C.; SPACCAPIETRA, S.; ZIMÁNYI, E. Spatio-temporal conceptual models: data structures + space + time. In: ACM INTERNATIONAL SYMPOSIUM ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, 7, 1999, Kansas City, Missouri, United States. **Proceedings...** New York: ACM, 1999. p. 26-33.

PEUQUET, D. J.; DUAN, N. An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographical data. **International Journal of Geographical Information Systems**, London, v. 9, n. 1, p. 7-24, 1995.

POOLEY, R.; STEVENS, P. **Using UML: Software engineering with objects and components**. Boston: Addison Wesley, 1999. 256p.

RATIONAL SOFTWARE MODELER. Disponível em: <www-01.ibm.com / software/awdtools/modeler/swmodeler/features/index.html/>. Acesso em: 01 fev. 2010.

RUMBAUGH, J. et al. **Object-oriented modeling and design**. Englewood Cliffs: Prentice Hall, 1990. 512p.

SAMPAIO, G. B. **GeoProfile – Um Perfil UML para Modelagem Conceitual de Bancos de Dados Geográficos**. 2009. 65f. Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Viçosa, Viçosa, 2009.

SAMPAIO, G.B.; GAZOLA, A.; LISBOA FILHO, J. Modelagem e projeto de bancos de dados geográficos com características temporais. In: SIMPÓSIO MINEIRO DE SISTEMAS DE INFORMAÇÃO, 2, 2005, Belo Horizonte. **Anais...** Belo Horizonte: SBC, 2005.

SELIC, B. A systematic approach to domain-specific language design using UML. In: IEEE INTERNATIONAL SYMPOSIUM ON OBJECT AND COMPONENT-ORIENTED REAL-TIME DISTRIBUTED COMPUTING (ISORC'07), 10, 2007, Santorini, Island. **Proceedings...** Washington: IEEE Computer Society, 2007. p. 2-9.

SENDALL, S.; KOZACZYNSKI, W. 2003. Model transformation: the heart and soul of model-driven software development. **IEEE Software**, v. 20, n. 5, p. 42-45, set./out. 2003.

STAR UML. Disponível em: <www.staruml.sourceforge.net/>. Acesso em: 01 fev. 2010.

THOMAS, D; BARRY, B. Model Driven Development – The Case for Domain Oriented Programming. In: ANNUAL ACM SIGPLAN CONFERENCE ON OBJECT-ORIENTED PROGRAMMING, SYSTEMS, LANGUAGES AND APPLICATIONS (OOPSLA'03), 18, 2003, Anaheim, Estados Unidos. **Proceedings…** New York: ACM, 2003, p. 2-7.

UML FORUM. Disponível em: <www.uml-forum.com/tools.htm>. Acesso em: 01 dez. 2009.

VISUAL PARADIGM. Disponível em: <www.visual-paradigm.com/>. Acesso em: 01 fev. 2010.

WORBOYS, M.; DUCKHAN, M. **GIS: A Computing Perspective**. 2. ed. Boca Raton: CRC Press, 2004. 426p.

ZUBCOFF, J.; PARDILLO, J.; TRUJILLO, J. A UML profile for the conceptual modelling of data-mining with time-series in data warehouses**. Information and Software Technology**, Newton, MA, EUA, v. 51, n. 6, p. 977-992, jun. 2009.