GUSTAVO BREDER SAMPAIO

GeoProfile – UM PERFIL UML PARA MODELAGEM CONCEITUAL DE BANCOS DE DADOS GEOGRÁFICOS

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
2009

Ficha catalográfica preparada pela Seção de Catalogação e Classificação da Biblioteca Central da UFV

T

S192g
2009

Sampaio, Gustavo Breder, 1984-
GeoProfile - um perfil UML para modelagem conceitual de bancos de dados geográficos / Gustavo Breder Sampaio. - Viçosa, MG, 2009.
ix, 65f : il. ; 29cm.

Orientador: Jugurta Lisboa Filho.
Dissertação (mestrado) - Universidade Federal de Viçosa.
Referências bibliográficas: f 62-65.

1. Sistemas de informação geográfica - Banco de dados. 2.Banco de dados. 3. UML (Computação). 4. Métodos orientados a objetos (Computação) I. Universidade Federal de Viçosa. II. Título.

CDD 22.ed. 003.3

GUSTAVO BREDER SAMPAIO

GeoProfile – UM PERFIL UML PARA MODELAGEM CONCEITUAL DE BANCOS DE DADOS GEOGRÁFICOS

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: 26 de junho de 2009

José Luís Braga
(Co-Orientador)

Karla Albuquerque de V. Borges
(Co-Orientador)

Alcione de Paiva Oliveira

Frederico Torres Fonseca

Jugurta Lisboa Filho
(Orientador)

AGRADECIMENTOS

A Deus e à minha família por ter conseguido concluir este trabalho. Aos amigos e professores pelo apoio. Agradecimento especial também ao meu orientador, pela atenção dispensada nestes últimos anos.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

BDG - Banco de Dados Geográficos

CASE - Computer-Aided Software Engineering

GeoOOA - Geographic Object-Oriented Analysis

MADS - Modeling for Application Data with Spatio-temporal features

OCL - Object Constraint Language

OMG - Object Management Group

OMT-G - Object Modeling Technique for Geographic Applications

RSM - Rational Software Modeler

SIG - Sistema de Informação Geográfica

UML - Unified Modeling Language

RESUMO

SAMPAIO, Gustavo Breder, M.Sc., Universidade Federal de Viçosa, junho de 2009. **GeoProfile – Um Perfil UML para Modelagem Conceitual de Bancos de Dados Geográficos**. Orientador: Jugurta Lisboa Filho. Co-Orientadores: José Luis Braga e Karla Albuquerque de Vasconcelos Borges.

Anos de pesquisa no campo de modelagem conceitual de bancos de dados geográficos tiveram como resultado vários modelos. Entretanto, não existe um consenso sobre qual é a melhor forma de se modelar os dados das aplicações geográficas, o que traz alguns problemas para o desenvolvimento da área. O mecanismo denominado Perfil permite uma extensão estruturada e precisa da UML, sendo uma excelente solução para padronização da modelagem de domínios específicos por aproveitar toda a infra-estrutura que a UML possui. Este trabalho apresenta o GeoProfile, Perfil UML que foi desenvolvido especificamente para modelagem conceitual de bancos de dados geográficos. A implementação completa do GeoProfile em uma ferramenta CASE também foi realizada. Isso permitiu constatar as vantagens da utilização de Perfis UML como, por exemplo, a possibilidade de verificar automaticamente a validade do esquema, tornando a modelagem livre de erros básicos. Espera-se que este trabalho seja o primeiro passo para padronização da modelagem conceitual desse domínio.

ABSTRACT

SAMPAIO, Gustavo Breder, M.Sc., Federal University of Viçosa, June of 2009. **GeoProfile – A UML Profile for Conceptual Modeling of Geographic Databases**. Adviser: Jugurta Lisboa Filho. Co-Advisers: José Luis Braga and Karla Albuquerque de Vasconcelos Borges.

Years of research in conceptual modeling of spatial databases led to several models. Nonetheless there is no agreement on what is the best way to model the applications of geographic data, what brings some problems to this field development. The mechanism called Profile allows a structured and precise extension of UML and it is an excellent resource for modeling specific domains. This paper presents the GeoProfile, UML profile that was developed specifically for conceptual modeling of geographic data bases. An implementation of GeoProfile in a CASE tool was also performed. This showed the benefits of using UML profiles, such as the ability to automatically check the scheme validity, making the model free of basic errors. It is hoped that this work is the first step towards standardization of this conceptual modeling field.

1 Introdução

A análise de dados espaciais sempre foi uma importante atividade das sociedades organizadas. Com o desenvolvimento da informática, tornou-se possível armazenar e representar estas informações em um ambiente computacional, permitindo a combinação de mapas e dados a fim de facilitar o processo de tomada de decisão. O avanço tecnológico permitiu então o surgimento dos Sistemas de Informação Geográfica (SIG). Esses sistemas são as ferramentas computacionais para o campo de Geoprocessamento (CÂMARA; DAVIS; MONTEIRO, 2003).

Um *software* de SIG, de forma geral, é um sistema composto de quatro grandes componentes: componente de captura, componente de armazenamento, componente de análise e componente de apresentação de dados. O componente de armazenamento é representado por um sistema de banco de dados geográficos que tem a responsabilidade de armazenar e recuperar de forma consistente tanto dados descritivos quanto espaciais. Em virtude da complexidade das aplicações geográficas, esses bancos de dados não podem ser modelados a partir de um modelo conceitual convencional como o Entidade-Relacionamento, pois requisitos específicos deste domínio precisam ser representados (LISBOA FILHO; IOCHPE, 1999b).

Com o objetivo de auxiliar o desenvolvimento de aplicações de SIG e solucionar os problemas relacionados ao projeto de banco de dados geográficos, muitos modelos foram propostos utilizando principalmente extensões de modelos convencionais. Por exemplo, pode-se citar OMT-G (BORGES; DAVIS; LAENDER, 2001), MADS (PARENT; SPACCAPIETRA; ZIMÁNYI, 2008), GeoOOA (KÖSTERS; PAGEL; SIX, 1997), UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008) e o modelo implementado na ferramenta CASE (Computer-Aided Software Engineering) Perceptory (PERCEPTORY, 2008), que é a união da UML (*Unified Modeling Language*) com extensões espaço-temporais propostas por Bédard (BÉDARD; LARRIVÉE, 2008).

Baseando-se neste contexto e nas atuais dificuldades encontradas para o desenvolvimento desta área de pesquisa, que tem como fonte principal a falta de um padrão de modelagem, a seguir é apresentada a motivação e os objetivos deste trabalho.

1.1 Motivação

Nos últimos 20 anos, conforme é verificado na literatura (BÉDARD et al., 2004), vários grupos de pesquisadores se dedicaram a analisar os requisitos de modelagem conceitual para aplicações de SIG. Muitos modelos conceituais específicos para este domínio são encontrados, sendo alguns citados na seção anterior.

Apesar do amadurecimento deste campo de pesquisa, até hoje não existe um consenso entre os projetistas sobre qual é o melhor modelo para se modelar um banco de dados geográficos (BDG). A falta de um padrão traz sérios problemas para o desenvolvimento da área, podendo-se citar a dificuldade de comunicação entre diferentes projetos e o aproveitamento de soluções já validadas em sistemas anteriores. Por exemplo, considerando as ferramentas CASE (*Computer-Aided Software Engineering*) que dão suporte a modelos conceituais específicos de BDG, não é possível migrar um esquema de dados entre elas, como ocorre nas ferramentas comerciais. Sendo assim, devido à grande variedade de modelos e ferramentas incompatíveis, a tarefa de se aproveitar soluções desenvolvidas para uma aplicação exige grande esforço.

Estes problemas não existiriam se houvesse um padrão para modelagem desses sistemas que reunisse as principais características dos modelos apresentados. A criação de um Perfil UML, que é um recurso que permite uma extensão estruturada e precisa dos construtores da UML para que esta possa se adequar a um domínio específico, é uma excelente opção para padronização deste tipo de modelagem. Um Perfil UML bem especificado deve ter suporte direto de ferramentas CASE. Exemplos de ferramentas CASE com suporte a Perfis UML são a Enterprise Architect (ENTERPRISE ARCHITECT, 2008) e Rational Software Modeler (RATIONAL SOFTWARE MODELER, 2008).

Logo, esta opção é adotada para o desenvolvimento da proposta de modelagem apresentada neste trabalho por aproveitar as vantagens da UML, que é uma linguagem amplamente difundida tanto no meio acadêmico quanto empresarial. Ou seja, a elaboração de um Perfil UML tem se mostrado um ótimo meio de padronização da modelagem de domínios específicos, pois aproveita a popularidade e ferramentas compatíveis com a UML 2.0, facilitando a aceitação do padrão e reduzindo o tempo de treinamento, visto que não é necessário desenvolver uma nova

linguagem ou ferramentas próprias para esta. Em alguns domínios atualmente já é adotado um Perfil UML como padrão de modelagem, como ocorre, por exemplo, para a arquitetura CORBA (OMG, 2008).

1.2 Objetivos

O principal objetivo deste trabalho é especificar um Perfil UML próprio para modelagem conceitual de bancos de dados geográficos, observando as necessidades impostas para este domínio de aplicação. Alguns modelos encontrados na literatura servirão de base para esta tarefa. Especificamente, pretende-se executar a seguinte seqüência de passos:

a) Estudo dos métodos para especificação de perfis UML;
b) Revisão dos modelos conceituais de bancos de dados geográficos a fim de definir quais características devem ser incorporadas ao Perfil UML;
c) Especificação do Perfil UML para modelagem conceitual de BDG;
d) Teste do perfil desenvolvido em uma ferramenta CASE, com suporte à perfis UML.

1.3 Estrutura da dissertação

Os próximos dois capítulos desta dissertação constituem o referencial teórico utilizado. No Capítulo 2, além de ser discutido o que é um Perfil UML, os passos para sua correta construção são apresentados. Estes são independentes do domínio abordado e podem servir de base para outras propostas. Já o Capítulo 3 contém exclusivamente uma discussão voltada para o domínio de aplicações geográficas, exibindo os requisitos e modelos próprios de um BDG.

Nos capítulos 4 e 5 estão os resultados deste trabalho, o que inclui a apresentação do Perfil UML desenvolvido, denominado GeoProfile, bem como sua implementação em uma ferramenta CASE.

Finalmente, no Capítulo 6 encontram-se as conclusões e trabalhos futuros.

2 Perfil UML

Para evitar que a UML se tornasse uma linguagem excessivamente complexa, seus criadores deixaram de implementar detalhes disponíveis em linguagens mais específicas. Entretanto, desenvolveram um recurso conhecido como Perfil para que esta pudesse ser extensível (ERIKSSON et al., 2004).

Antes de se definir o que é Perfil, é necessário citar alguns conceitos que envolvem a organização da UML. Sua especificação é definida utilizando-se uma técnica de metamodelagem, ou seja, um metamodelo é usado para especificar o modelo que consiste a UML. Tipicamente o papel de um metamodelo é definir como os elementos de um modelo devem ser instanciados. A Figura 2.1 mostra um exemplo de utilização desta técnica. As metaclasses *Association* e *Class*, que representam parte do metamodelo da UML, são instanciadas em um modelo definido pelo usuário. As classes *Person* e *Car* são instâncias da metaclasse *Class*, e a associação *Person.car* é uma instância da metaclasse *Association* (OMG, 2007).

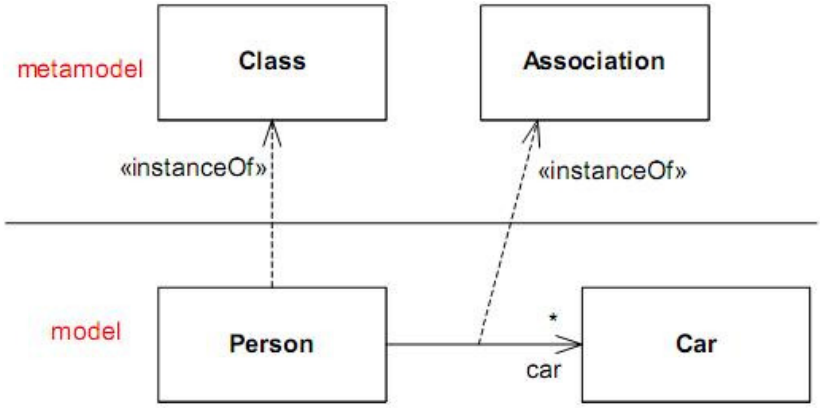


Figura 2.1 – Exemplo de metamodelagem.
Fonte: OMG (2007)

A arquitetura da UML é exibida na Figura 2.2. O pacote que agrupa os construtores mais básicos desta estrutura é denominado *Infrastructure Library* e contém dois sub-pacotes: *Core* (tem os principais elementos da linguagem); e *Profiles* (define as formas de se estender os elementos da linguagem). O metamodelo MOF (*Meta-Object Facility*) utiliza a *Infrastructure Library* e fornece as regras básicas que são usadas na construção da UML (ERIKSSON et al., 2004).

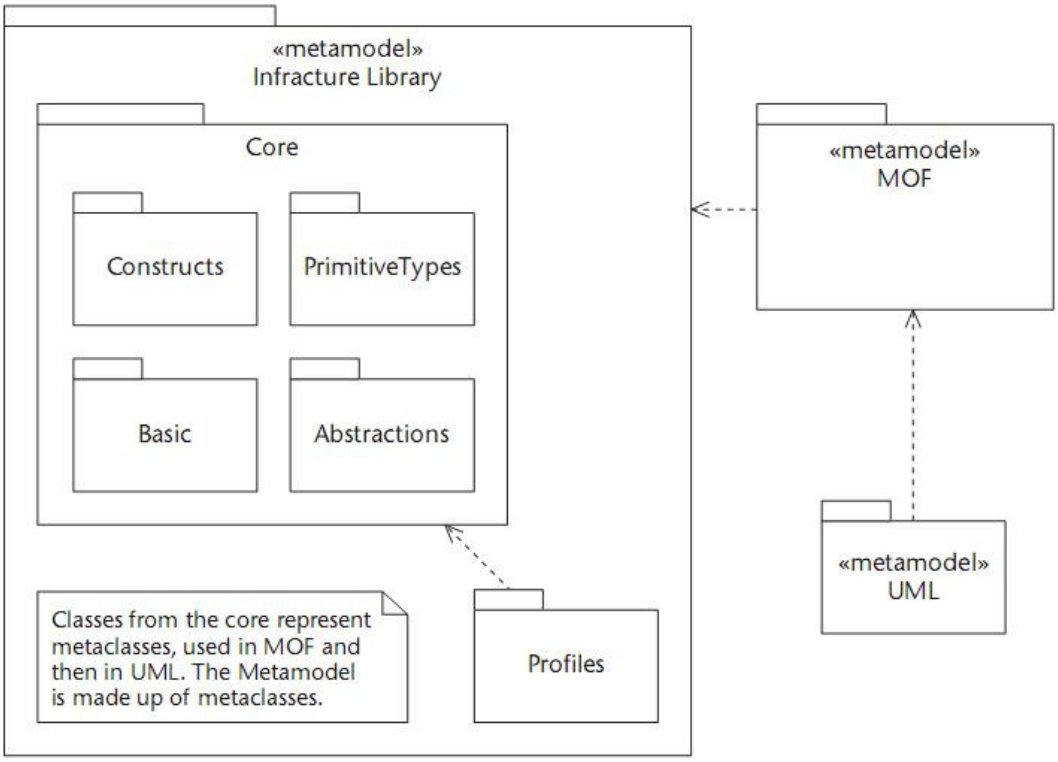


Figura 2.2 – Arquitetura da UML.
Fonte: Eriksson et al. (2004)

A OMG (*Object Management Group*) é a organização que mantém a UML. Pode-se dizer que Perfil é um recurso disponibilizado pela OMG que contém mecanismos que permitem metaclasses de metamodelos existentes serem estendidas para adaptá-las a diferentes propósitos, desde que o metamodelo seja baseado no núcleo de classes que dá origem à UML. Isso inclui a capacidade de adaptar o metamodelo para diferentes plataformas (como a J2EE ou .NET) ou domínios (como de tempo real ou processos de modelagem de negócios). Apesar de o Perfil ser capaz de estender diferentes linguagens, a UML é o caso mais convencional (OMG, 2007). Seguindo os interesses da proposta apresentada nesta dissertação, as discussões referentes a Perfis são focadas na extensão da UML.

Segundo Fuentes e Vallecillo (2004) há várias razões que levam um projetista a querer customizar o metamodelo da UML. Por exemplo, a necessidade de restringir o modo como o metamodelo e seus construtores são usados (ex.: forçar a existência de certas associações entre classes do modelo) ou a necessidade de adicionar informações que podem ser usadas durante a transformação de um modelo em outro modelo ou em código. Entretanto, é importante ressaltar que durante o processo de

extensão do metamodelo, não é permitido pelos recursos relacionados ao Perfil a adição de novos elementos ao metamodelo.

A idéia de estender a UML para propósitos específicos não é novidade. Já na UML 1.1, estereótipos e valores etiquetados (*tagged values*) podiam ser facilmente atribuídos a elementos do modelo. Entretanto, a noção de Perfil foi definida com o objetivo de prover uma forma mais estruturada e precisa de extensão (OMG, 2007).

O restante do Capítulo 2 exibe dois métodos desenvolvidos para auxiliar a criação desses perfis. Como a OCL (*Object Constraint Language*) é utilizada por esses métodos, uma breve descrição da linguagem é apresentada.

2.1 Object Constraint Language

Um diagrama UML, como o diagrama de classes, por exemplo, normalmente não é suficiente para fornecer todos os aspectos relevantes de uma especificação. Verifica-se, entre outras coisas, a necessidade de se descrever restrições adicionais sobre as classes do esquema. Essas restrições podem ser descritas em linguagem natural, entretanto, a prática tem mostrado que este procedimento sempre resulta em ambiguidades. A OCL foi desenvolvida para resolver este problema. Ela é uma linguagem formal de fácil leitura e escrita, capaz de descrever restrições sem ambiguidade (OMG, 2006).

A Figura 2.3 ilustra um esquema criado para exemplificar o uso da OCL. Neste esquema é considerado que pessoas podem trabalhar para empresas. Também é representado que o sistema deve ser capaz de informar com quem cada pessoa está casada. A seguir são discutidos alguns exemplos de restrições OCL criadas para este esquema.

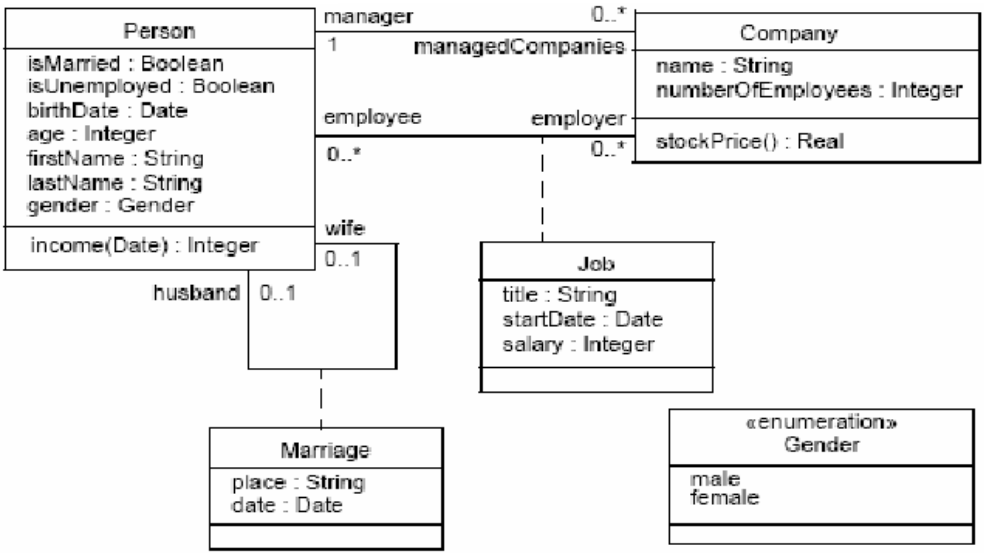


Figura 2.3 – Exemplo de modelagem.
Adaptado de OMG (2006)

Poder ser interessante, durante a modelagem de um sistema, indicar que o valor de um atributo deve estar contido em um determinado intervalo. Para restringir o valor do atributo *age* nos objetos da classe *Person*, a seguinte restrição OCL pode ser adicionada ao esquema:

```
context Person
inv: self.age > 0
```

A palavra-chave *context* define para qual elemento do esquema é definida a restrição (neste caso, a classe *Person*). Já *inv* indica que a restrição é invariável, ou seja, deve ser verdadeira durante todo o ciclo de vida da instância. Por fim, *self* é utilizada para se referir à instância do contexto.

Outra possível restrição é indicar que todo gerente de uma empresa deve ter idade superior a quarenta anos. Neste caso, a partir do contexto *Company* é possível acessar a instância de *Person* associada à empresa. Para isto, basta utilizar o papel da associação, que neste caso é *manager*:

```
context Company
inv: self.manager.age > 40
```

Na restrição anterior, como a multiplicidade da associação considerada é 1, e execução de "*self.manager"* sempre retorna uma instância de *Person* e o acesso à idade desta pessoa pode ser feito por "*.age"*. Já na restrição a seguir, é dito que sempre que uma pessoa possuir um marido, este deve ter mais de dezoito anos:

```
context Person
inv:self.husband->notEmpty()implies self.husband.age>=18
```

Quando a multiplicidade for maior que um, a OCL retorna uma coleção nesta navegação pelas associações do esquema. Métodos para manipular coleções e outros exemplos podem ser encontrados em (OMG, 2006) e em (WARMER e KLEPPE, 2003).

2.2 Passos para definição de um Perfil UML

A seguir são discutidas duas propostas desenvolvidas para guiar a construção de Perfis UML. Os resultados são equivalentes, sendo obtido ao final do processo um conjunto de estereótipos, que podem conter atributos (*tagged values*) e restrições definidas pelo usuário. Um estereótipo pode ser entendido como um tipo especial de classe que sempre estende uma metaclasse da UML (ex.: classe, associação, atributo), adicionando novas informações a ela. Esses elementos, que representam o Perfil UML, são agrupados em um Pacote marcado pelo estereótipo <<profile>> acima de seu nome.

2.2.1 Proposta de Fuentes e Vallecillo

De acordo com Fuentes e Vallecillo (2004), a definição de um Perfil UML deve ser orientada por cinco passos:

I. Primeiramente, é necessário definir o conjunto de elementos que irão compreender a plataforma ou sistema considerado, assim como os relacionamentos existentes entre eles. Isto pode ser realizado utilizando-se os recursos padrão da UML (ex.: classes e associações) normalmente, como se a intenção nem fosse a definição do perfil. O resultado desta tarefa é chamado de metamodelo do domínio;
II. Concluída a primeira etapa, o projetista está apto para iniciar a definição do Perfil UML. Então, para cada elemento relevante do metamodelo do domínio, é criado um estereótipo. Com o intuito de esclarecer a relação entre este metamodelo e o perfil, os estereótipos são nomeados de acordo com os elementos correspondentes do metamodelo do domínio;
III. Deve-se notar que apenas os elementos do metamodelo da UML que são estendidos devem ser representados por um estereótipo. Exemplos destes elementos são classes, associações, atributos e pacotes.

IV. Os atributos que aparecem nas classes do metamodelo do domínio dão origem a *tagged values* nos estereótipos, incluindo os tipos e valores iniciais correspondentes;

V. As restrições do perfil devem ser definidas com base nas restrições do domínio. Por exemplo, a multiplicidade das associações que aparecem no metamodelo do domínio ou as regras de negócio da aplicação.

Para ilustrar o guia acima, Fuentes e Vallecillo (2004) desenvolveram um exemplo em que se deseja modelar as conexões entre elementos de um sistema de informação com a topologia estrela. Nodos (*Node*) se conectam a nodos principais (*MainNode*), que podem se conectar a outros nodos principais. As conexões podem ser do tipo *LocalEdge*, caso liguem um nodo ao nodo principal de sua estrela, ou do tipo *Edge*, caso liguem nodos principais entre si. Cada nodo deve ser identificado pela sua localização. A Figura 2.4 exibe o metamodelo desta aplicação.

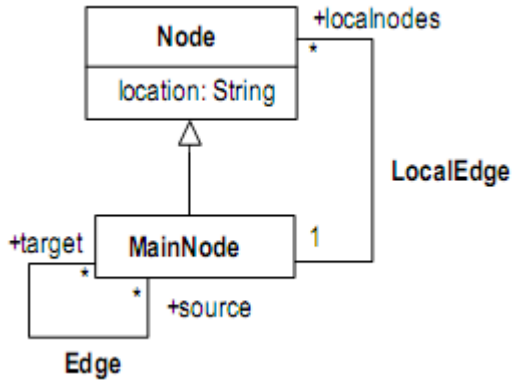


Figura 2.4 – Metamodelo da aplicação exemplo.
Fonte: Fuentes e Vallecillo (2004)

O Perfil UML relativo a este metamodelo é mostrado na Figura 2.5. Este perfil possui quatro estereótipos que correspondem às classes e associações definidas no metamodelo do domínio. Cada estereótipo indica a metaclasse da UML em que pode ser aplicado através de uma seta preenchida. Neste caso, <<Node>> e <<MainNode>> podem ser aplicados a classes enquanto as associações podem receber <<Edge>> e <<LocalEdge>>. Além disso, o estereótipo *Node* possui um *tagged value* (*location*) do tipo *String* para informar sua localização.

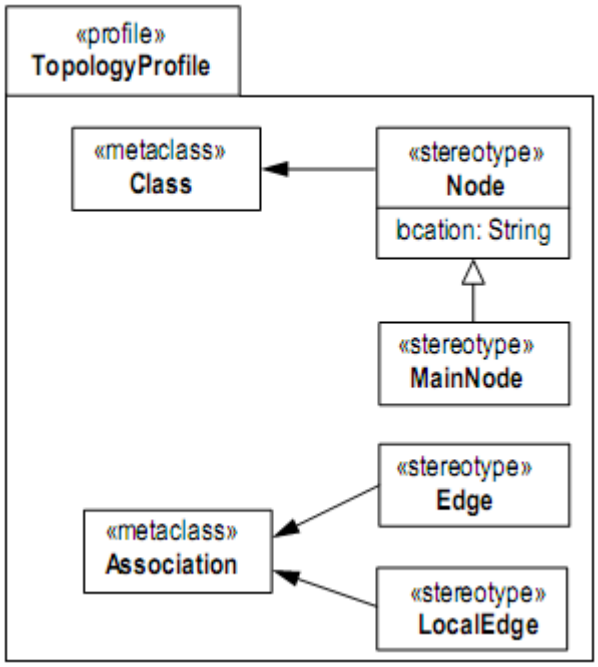


Figura 2.5 – Perfil UML da aplicação exemplo.
Fonte: Fuentes e Vallecillo (2004)

Seguindo a última etapa, devem ser definidas as restrições a serem impostas durante a utilização do perfil. Normalmente, a linguagem utilizada nesta tarefa é a OCL, pois esta, por fazer parte da UML, pode ser checada automaticamente por ferramentas compatíveis com a UML. De acordo com os autores, a possibilidade de verificar as restrições ao final da modelagem é uma das maiores vantagens de se utilizar um Perfil UML. Isto porque neste caso é garantido que o modelo desenvolvido pelo projetista está em conformidade com as regras do domínio abordado, evitando a criação de modelos mal construídos.

Duas restrições são definidas para o exemplo. A primeira é obtida através da multiplicidade da associação *LocalEdge*, que obriga todo nodo a estar conectado a um nodo principal. Dessa forma, se forem adicionadas restrições OCL adequadas ao perfil, toda vez que o estereótipo <<Node>> for utilizado, será verificado se existe uma associação ligando a classe estereotipada por <<Node>> a uma outra marcada pelo estereótipo <<MainNode>>. A outra restrição é criada pelos autores para representar uma regra do domínio que não é explicitada em seu metamodelo. É imposto que todo nodo de uma mesma estrela deve estar localizado em uma mesma região geográfica.

Assim como mostrado na Seção 2.1, as restrições OCL definidas para o perfil também utilizam papéis de associações para a navegação entre os elementos do modelo. Entretanto, isso é feito de maneira um pouco diferente. Como a elaboração de um Perfil UML se faz no nível de metamodelo, não há um modelo para basear a navegação das restrições a serem anexadas aos estereótipos. Então, para desenvolver uma restrição OCL que obriga todo nodo a se conectar com um nodo principal, por exemplo, deve-se utilizar os papéis disponíveis no metamodelo da UML.

Assumindo que o estereótipo <<Node>> é aplicado a uma classe do modelo, essa técnica permite partir dessa classe que recebeu o estereótipo e navegar por suas associações. Consequentemente é possível saber se há alguma conexão com uma classe estereotipada por <<MainNode>>, atendendo assim à restrição do domínio.

As restrições OCL encontradas no artigo de Fuentes e Vallecillo (2004) não são exibidas aqui porque alguns papéis utilizados pelos autores para navegação não puderam ser verificados na especificação mais recente da UML.

Também foi observado durante a execução do projeto apresentado nesta dissertação que não existe um método padrão para capturar os estereótipos aplicados a um elemento na especificação da UML e da OCL. Por esta razão, alguns autores criam seus próprios métodos para este fim, como o *isStereotype* (FUENTES; VALLECILLO, 2004) e o *getAppliedStereotype* (DMTF, 2007). Estas dificuldades podem prejudicar a exportação de perfis UML entre diferentes ferramentas de modelagem e possivelmente serão corrigidas nas especificações futuras pela OMG. A seguir, as restrições OCL são discutidas conforme a versão 2.1.2 da UML.

A Figura 2.6 exibe uma parte simplificada do metamodelo da UML segundo a especificação encontrada em (OMG, 2007). Uma *Property*, quando relacionada a uma classe através de *ownedAttribute*, representa um atributo e também pode representar uma ponta de associação (*association end*). Uma ponta de associação é a conexão existente entre a linha que descreve a associação e o ícone que descreve a classe. Quando uma *Property* se relaciona com uma associação através de *memberEnd*, ela representa uma ponta de associação.

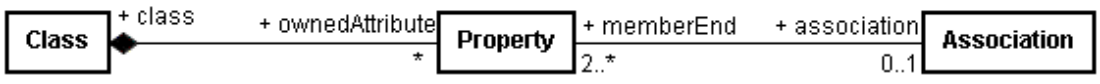


Figura 2.6 – Parte simplificada do metamodelo da UML.
Adaptado de OMG (2007)

Além disso, de acordo com a OMG, nas restrições OCL associadas a estereótipos, o termo "*self*" refere-se ao estereótipo e não à instância da metaclasse em que o estereótipo é aplicado. Para acessar esta instância, deve-se realizar a navegação *self.base_<metaclass>*, onde *<metaclass>* é substituído pela metaclasse da UML estendida pelo estereótipo (DMTF, 2007).

Logo, no caso do estereótipo <<Node>>, que estende a metaclasse *Class* da UML, pode-se fazer a navegação *self.base_Class.ownedAttribute.association* para obter as associações da classe que recebeu este estereótipo. Por outro lado, para obter as classes conectadas por uma associação que recebeu o estereótipo <<Edge>> pode-se fazer a navegação *self.base_Association.memberEnd.class*.

Na Figura 2.7 é mostrado um exemplo de aplicação desse perfil UML, chamado de *TopologyProfile*. São modeladas duas classes: *Branch* (representam nodos da rede) e *CentralOffice* (representam nodos principais). A associação entre estas duas classes recebe o estereótipo <<LocalEdge>>. Além disso, são incluídas notas para indicar o valor do *tagged value* de cada estereótipo (para as duas classes foi atribuído o código "uma.es" ao *tagged value* denominado *location*).

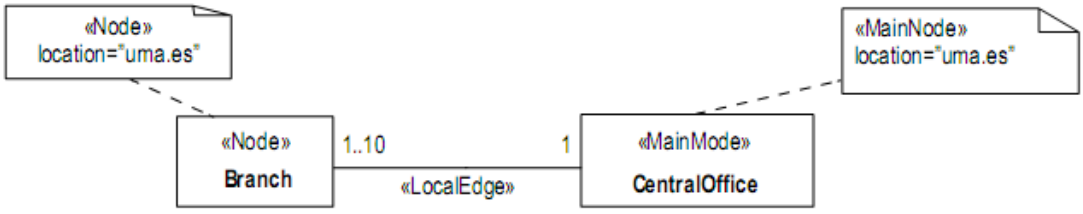


Figura 2.7 – Exemplo de utilização do *TopologyProfile*
Adaptado de Fuentes e Vallecillo (2004)

2.2.2 Proposta de Selic

Em (SELIC, 2007), outro método para construção de perfis UML é encontrado. Há várias semelhanças com a proposta exibida na seção anterior. Porém, o autor prefere separar o método em apenas duas etapas, que são relacionadas à elaboração de dois artefatos: um metamodelo, e o perfil em si.

Para Selic (2007), a experiência adquirida com a definição de perfis tem indicado que o primeiro passo deve ser focado nas características do domínio, sem qualquer consideração sobre o metamodelo da UML. Isso garante que antes do

projetista iniciar a extensão dos elementos da UML, as necessidades do campo abordado são compreendidas adequadamente.

O artefato gerado nesta tarefa é o metamodelo do domínio, que deve representar os conceitos essenciais do domínio, assim como os relacionamentos existentes entre eles e as restrições que governam o modo como estes elementos podem ser combinados para formarem um modelo válido.

Obtido o metamodelo, então a segunda etapa pode ser iniciada. É realizado um mapeamento do metamodelo do domínio para o Perfil UML. Para cada elemento do metamodelo que se deseja incluir no perfil, deve ser identificada a metaclasse da UML mais adequada para a extensão. São enumerados quatro passos que orientam este mapeamento: selecionar uma metaclasse da UML cuja semântica é a mais próxima do elemento do domínio; checar se não há conflito entre as restrições que são aplicadas à metaclasse escolhida e as restrições do estereótipo; checar se há necessidade de refinar algum atributo da metaclasse (como redefinir o valor de um atributo ou então eliminar atributos que não são relevantes para o domínio, alterando sua multiplicidade para zero); e checar se a metaclasse selecionada possui associações conflitantes com outras metaclasses.

Como resultado é obtido então o Perfil UML. A discussão exibida na seção anterior sobre restrições OCL nos perfis UML também é válida para o método de Selic (2007), visto que as soluções são equivalentes.

3 Modelagem conceitual de dados geográficos

Segundo Borges, Davis e Laender (2001), para as aplicações geográficas, o nível de representação conceitual fornece um conjunto de conceitos com os quais entidades geográficas, tais como rios, edifícios, ruas e vegetação, podem ser modeladas em um alto nível de abstração, conforme são percebidas pelo usuário. Classes a serem criadas no banco de dados são definidas neste nível, sendo estas possivelmente associadas a alguma representação espacial. Como exemplo, os autores citam uma aplicação envolvendo escolas. Em um sistema convencional, a classe *Escola* incluiria atributos de identificação (ex.: nome e número) e atributos de localização (ex.: endereço). Já em um SIG, a localização de uma escola é melhor representada através de um conjunto de coordenadas, ao invés de um atributo alfanumérico simplesmente.

Parent et al. (1998) destacam algumas vantagens de se utilizar a modelagem conceitual em aplicações que manipulam dados espaço-temporais. Primeiramente, usuários podem expressar seu conhecimento sobre o sistema utilizando conceitos que são próximos à sua realidade e independentes de conceitos computacionais. Além disso, como a modelagem conceitual é independente da implementação do sistema, o resultado da modelagem permanece válido em caso de mudanças tecnológicas. Ou seja, o esquema desenvolvido pode ser reaproveitado independentemente do SIG escolhido para implementação do sistema. Por fim, também é dito que a modelagem conceitual, devido a sua legibilidade, favorece a troca de informações referentes ao projeto.

Considerando que o principal objetivo do trabalho apresentado nesta dissertação é construir um Perfil UML próprio para modelagem conceitual de bancos de dados geográficos, neste capitulo é apresentada uma revisão bibliográfica que permitiu definir quais são os requisitos deste tipo de modelagem. Também são apresentados alguns modelos conceituais que atendem a grande parte destes requisitos. Na seção 3.3 é feita uma comparação entre os requisitos e modelos estudados. Esta tarefa possibilita a análise e aproveitamento das características essenciais de cada modelo durante a elaboração do GeoProfile. Finalmente, na seção 3.4 são discutidas as conclusões deste capítulo.

3.1 Requisitos de modelagem de dados geográficos

Em (FRIIS-CHRISTENSEN; TRYFONA; JENSEN, 2001) é realizado um levantamento dos requisitos de modelagem de dados geográficos através da análise de uma aplicação real do *Danish National Survey and Cadastre*. Estes requisitos são classificados em cinco grupos. São eles:

- **Propriedades espaço-temporais**. Incluem os requisitos espaciais (coordenadas em um sistema de referência; representação de pontos, linhas e polígonos); temporais (necessidade de guardar o período de existência e as mudanças que ocorreram em um objeto); necessidade de representação de atributos dos objetos, assim como um identificador único; e diferenciação entre *campos* (o mundo real é entendido como um conjunto de atributos que variam no espaço assim como uma função contínua. Esta abordagem é mais adequada para certos fenômenos, como relevo ou temperatura) e *objetos* (o mundo real é constituído de entidades bem definidas, como rodovias e construções);
- **Papéis**. Um mesmo objeto geográfico pode ser definido de formas diferentes dependendo do universo considerado. Ou seja, o papel de um objeto é dependente da aplicação. Deve ser possível a indicação dos papéis baseados em um mesmo tipo de objeto;
- **Associações**. Incluem relacionamentos topológicos (ex.: *overlap, touch*); métricos (envolvem a distância e dependem da posição absoluta dos objetos em um sistema de referência); semânticos (ex.: "todos os lotes devem ter acesso a estradas"); e relacionamentos para indicar que um objeto é composto de outros objetos;
- **Restrições**. Deve ser possível anexar restrições a objetos (ex.: limitar o valor de um atributo a certo intervalo) e associações (ex.: impedir que um prédio seja localizado em um lago). As restrições são relacionadas com a qualidade dos dados, pois a afetam negativamente quando não são satisfeitas.
- **Qualidade dos dados**. Esta informação é importante para se conhecer a credibilidade dos dados. Ela deve ser comparada com as especificações da aplicação para decidir se os dados são suficientemente precisos naquela ocasião.

Friis-Christensen e demais autores também fazem uma comparação entre alguns modelos e os requisitos encontrados para mostrar as vantagens e desvantagens de cada modelo. Em uma das conclusões desse trabalho de comparação é indicada a importância de balancear a facilidade de uso da notação de um modelo com o seu poder de expressão. O desafio é conseguir balancear estas duas características ou então aperfeiçoá-las simultaneamente, sendo que o desenvolvimento de um padrão de modelagem é visto pelos autores como a base para promover a tarefa de troca de dados.

Uma outra lista de requisitos é exibida em (LISBOA FILHO; IOCHPE, 1999b). Neste estudo, oito grupos de requisitos são mencionados, sendo cinco deles equivalentes aos apresentados por Friis-Christensen: possibilidade de modelagem dos fenômenos nas visões de campo e de objeto; aspectos espaciais; relacionamentos espaciais; aspectos temporais; e aspectos de qualidade. Os outros requisitos, que não são explicitamente citados no trabalho anterior, são: possibilidade de diferenciação entre fenômenos geográficos e objetos sem referência espacial; necessidade de organizar os fenômenos por tema; e possibilidade de modelagem de fenômenos com mais de uma representação espacial (múltiplas representações).

3.2 Modelos conceituais para representação de dados geográficos

As cinco propostas de modelagem que contribuíram para o desenvolvimento do GeoProfile são brevemente mostradas a seguir, juntamente com a pesquisa desempenhada por Clementini, Di Felice e Oosterom (1993) sobre relacionamentos topológicos, que complementou esta parte do referencial teórico. Na próxima seção, para cada requisito de modelagem, estes modelos são expostos com mais detalhes e algum exemplo de utilização.

O OMT-G (*Object Modeling Technique for Geographic Applications*) foi inicialmente baseado no diagrama de classes OMT. Posteriormente, recebeu uma adaptação de acordo com os conceitos e notação da UML (BORGES; DAVIS; LAENDER, 2001). São oferecidos três diferentes diagramas: para especificação de classes e seus relacionamentos; especificação das operações de transformação entre as classes; e para especificação dos aspectos visuais que cada classe pode assumir. Seus construtores permitem a diferenciação entre classes com representação de

objetos e campos espaciais. Com relação às associações, estas podem ser de três tipos: associações simples (representam relações estruturais entre diferentes classes, sendo estas convencionais ou georreferenciadas), relações de rede topológica (relações entre objetos que são conectados uns aos outros) e relações espaciais (representam relações topológicas, dentre outras, tais como *touch*, *in*, *cross*, *overlap* e *disjoint*).

O modelo GeoOOA (KÖSTERS; PAGEL; SIX, 1997) foi desenvolvido com base no método *Object-Oriented Analysis* (OOA). Suporta a abstração de classes espaciais, estruturas topológicas do tipo "todo-parte", estruturas de rede e classes temporais. As classes espaciais são representadas por classes do tipo ponto, linha, região e raster. As estruturas topológicas do tipo "todo-parte" modelam certos tipos de relacionamentos espaciais, podendo ser do tipo *covering*, *containment* e *partition structures*. Já as estruturas de rede são capazes de modelar relacionamentos especiais entre classes que compõem uma rede. Finalmente, as classes temporais possuem atributos padrão que armazenam as datas de "nascimento" e "morte" do objeto, assim como serviços para criar, restaurar e comparar versões destes. Também é possível representar relacionamentos temporais para indicar que uma classe é ancestral de outra.

MADS (*Modeling of Application Data with Spatio-temporal features*) é um modelo conceitual que aborda objetos e relacionamentos em seu diagrama, com estruturas bastante semelhantes ao modelo Entidade-Relacionamento (PARENT; SPACCAPIETRA; ZIMÁNYI, 1999, 2008). São oferecidos por este modelo o que se chama de *abstract data types* (ADT's) para representação gráfica das características espaciais e temporais das classes. Os ADT's espaciais incluem a representação de áreas, pontos, linhas e linhas orientadas, dentre outros, enquanto os ADT's temporais são capazes de representar instantes, intervalos e elementos temporais. Também há suporte para restrições temporais e espaciais encontradas em alguns relacionamentos. A principal característica deste modelo está na ortogonalidade, ou seja, características espaciais e temporais podem ser adicionadas tanto a objetos quanto a relacionamentos ou atributos. Isso deixa o modelo bastante poderoso, porém o trabalho de modelagem torna-se mais complexo, uma vez que erros de consistência podem ocorrer com mais facilidade.

A Perceptory é uma ferramenta CASE (PERCEPTORY, 2008) para modelagem de BDG que utiliza a UML em conjunto com pictogramas espaciais e

temporais especialmente criados para representação dos dados geográficos e que podem ser adicionados às classes e atributos do diagrama. Estes pictogramas são agrupados nas linguagens denominadas *Spatial PVL* e *Temporal PVL* (*Plug-in for Visual Languages*), que foram propostas com o objetivo de possibilitarem a adição de características espaço-temporais não só à UML, como também a outras linguagens visuais de modelagem (BÉDARD, 1999). Os pictogramas espaciais são capazes de representar pontos, linhas e polígonos, enquanto os temporais podem reproduzir instantes ou intervalos. Casos mais complexos são modelados com a combinação destes pictogramas (BÉDARD; LARRIVÉE, 2008).

O UML-GeoFrame, originalmente apresentado em (LISBOA FILHO; IOCHPE, 1999a), é baseado em uma hierarquia de classes estruturadas que compõem o framework conceitual GeoFrame. O GeoFrame foi definido de acordo com as regras do formalismo da orientação a objetos utilizando a UML e fornece os elementos básicos presentes em qualquer banco de dados geográfico. Seu diagrama de classes, apresentado na Figura 3.1, é dividido em três níveis de abstração (*Planning*, *Metamodel* e *Spatial Representation*). A combinação da UML com o GeoFrame permite o projeto de um BDG com uma linguagem de fácil compreensão (LISBOA FILHO; IOCHPE, 2008). As classes *Theme* e *GeographicRegion* são a base de todas aplicações geográficas. Cada região geográfica de interesse da aplicação pode agrupar vários temas, que reúnem informações semelhantes. A classe *ConventionalObject* representa os objetos convencionais, ou seja, aqueles que não possuem representação espacial. Já a classe *GeographicPhenomenon* generaliza qualquer fenômeno cuja localização geográfica deva ser considerada e é especializada em duas classes: *GeographicObject* (generaliza classes que possuem identidade própria e suas características são descritas através de atributos) e *GeographicField* (generaliza classes modeladas como funções sobre uma variável). As demais classes especificam as possíveis representações espaciais para os fenômenos geográficos. As principais classes do framework GeoFrame são representadas no modelo UML-GeoFrame através do uso de estereótipos.

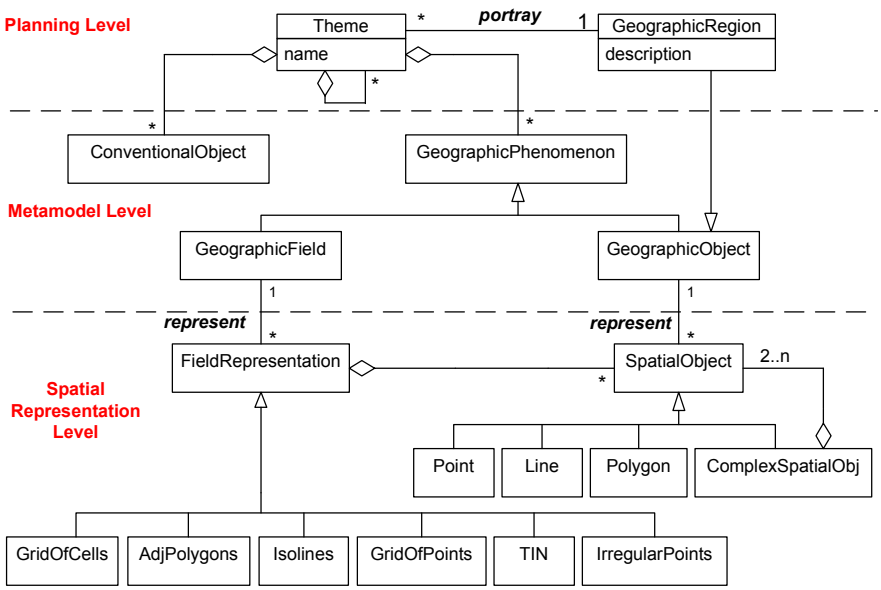


Figura 3.1 – O framework GeoFrame
Fonte: Lisboa Filho e Iochpe (2008)

Por fim, Clementini, Di Felice e Oosterom (1993) descrevem formalmente um pequeno conjunto de relacionamentos capazes de reproduzir todos os possíveis relacionamentos topológicos que podem ocorrer entre elementos espaciais com a representação de ponto, linha ou área. Esses relacionamentos são caracterizados por manterem suas propriedades sob operações topológicas (ex.: translação, rotação, escala) e como discutido na seção anterior, fazem parte de um dos requisitos de modelagem de dados geográficos. Apesar de não propor um modelo, esse trabalho adquire grande importância no escopo do projeto de construção do GeoProfile. Ao definir um conjunto mínimo de relacionamentos, elimina-se a possibilidade do uso de dois relacionamentos com nomes diferentes, mas de mesmo significado. Este conjunto inclui os seguintes relacionamentos:

- *Touch*: dois objetos geométricos se "tocam" quando se a única coisa que eles têm em comum está contida na união de suas fronteiras. Este tipo de relacionamento pode ocorrer entre área/área, linha/linha, linha/área, ponto/área e ponto/linha, mas não entre ponto/ponto;
- *In*: um objeto está "dentro" de outro se o primeiro está completamente contido no segundo. Este relacionamento aplica-se em qualquer situação envolvendo pontos, linhas ou áreas;
- *Cross*: duas linhas se "cruzam" quando se interceptam em um ponto interno (o que é diferente do relacionamento *touch*, pois neste caso a

interseção é somente nas fronteiras). De modo parecido, uma linha "cruza" uma área se ela está parcialmente dentro da área e parcialmente fora. Aplica-se a linha/linha e linha/área;

- *Overlap*: dois objetos se "sobrepõem" quando o resultado de sua interseção é um terceiro objeto de mesma dimensão, porém diferente dos dois primeiros. Aplica-se a área/área e linha/linha;
- *Disjoint*: dois objetos participam deste relacionamento quando não há interseção entre eles. Aplica-se a qualquer situação envolvendo pontos, linhas ou áreas.

Além de definirem este conjunto, os autores ainda provam que os cinco relacionamentos topológicos são mutuamente exclusivos (ou seja, é impossível que dois tipos diferentes de relacionamento estejam presentes entre dois objetos no mesmo instante) e que dados dois objetos quaisquer, o relacionamento topológico existente entre eles é um dos cinco descritos anteriormente.

3.3 Análise comparativa entre os modelos e requisitos apresentados

Todos os modelos citados anteriormente disponibilizam, de alguma forma, recursos que atendem à maioria dos requisitos de modelagem discutidos. Nesta seção esses recursos são agrupados de acordo com os requisitos, o que permite uma melhor comparação entre cada modelo.

Entretanto, dois requisitos não são analisados. O primeiro deles refere-se à *Qualidade dos Dados*. Em (FRIIS-CHRISTENSEN; TRYFONA; JENSEN, 2001) foi verificado que este requisito não era atendido pelos modelos apresentados. Ainda nas publicações mais recentes dos modelos considerados aqui, pouco ou nenhum avanço é observado em relação aos metadados. Já o segundo requisito refere-se aos *Papéis.* Mesmo não sendo explicitamente tratado pela maioria dos modelos, este requisito pode ser satisfeito de formas alternativas, como ilustra a Figura 3.2. Neste exemplo, criado na ferramenta Perceptory, a classe *Road* possui dois papéis: *Judicial Road* e *Topographic Road*. Utilizando-se apenas construtores da UML foi possível modelar este requisito. De forma semelhante, a UML é capaz de satisfazer parte do requisito *Restrições*, limitando os valores de atributos a certos intervalos com o uso da OCL.

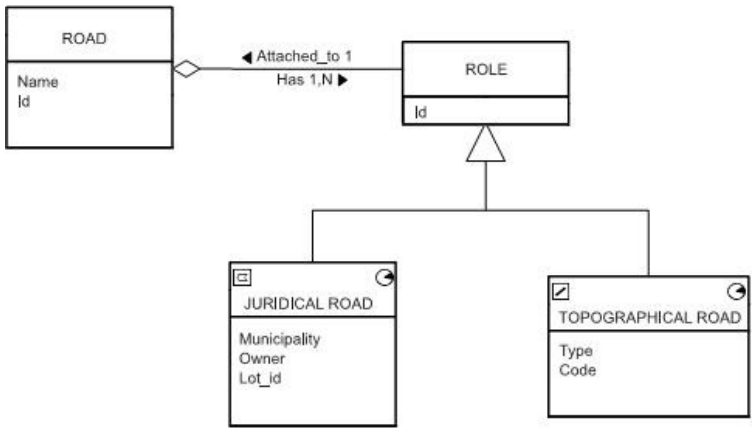


Figura 3.2 – Papéis na ferramenta Perceptory
Fonte: Friis-Christensen, Tryfona e Jensen (2001)

A análise desta seção inclui também algumas modificações ou extensões que estes modelos ganharam desde suas primeiras publicações. Outros grupos de pesquisadores ou mesmo os próprios autores realizaram estas alterações com o intuito de atender adequadamente a algum requisito não abordado na proposta original. Pode-se citar, por exemplo, a extensão temporal realizada por Meinerz (2005) e Rocha (2001) para os modelos OMT-G e UML-GeoFrame, respectivamente, a modelagem de múltiplas representações com o MADS (PARENT; SPACCAPIETRA; ZIMÁNYI, 2008) e a modelagem de redes para o UML-GeoFrame (STEMPLIUC, 2008).

3.3.1 Fenômenos geográficos e objetos convencionais

Em um banco de dados geográficos, além dos dados referentes a fenômenos georreferenciados, geralmente existem objetos convencionais, como aqueles presentes em qualquer sistema de informação. Uma fazenda, por exemplo, pode ser um fenômeno geográfico se no banco de dados estão armazenadas suas informações espaciais, como os seus limites. Entretanto, este mesmo banco de dados pode armazenar dados sobre os proprietários de cada fazenda, considerando-os objetos convencionais por não terem informação espacial associada. É importante que em um esquema conceitual esses dois tipos de classes sejam facilmente diferenciados (LISBOA FILHO; IOCHPE, 1999b).

Entre os cinco modelos apresentados na seção anterior, todos permitem a diferenciação dos fenômenos geográficos através da adição de símbolos geográficos

à suas classes. Ou seja, a ausência dessa informação indica que a classe é convencional.

O modelo UML-GeoFrame é o único a utilizar um símbolo gráfico para representação de objetos convencionais. Isto é feito através da adição do estereótipo *ConventionalObject* à classe, representado por um triângulo sem preenchimento (△). Na Figura 3.3 são ilustrados alguns exemplos de objetos convencionais modelados em cada um dos modelos estudados.

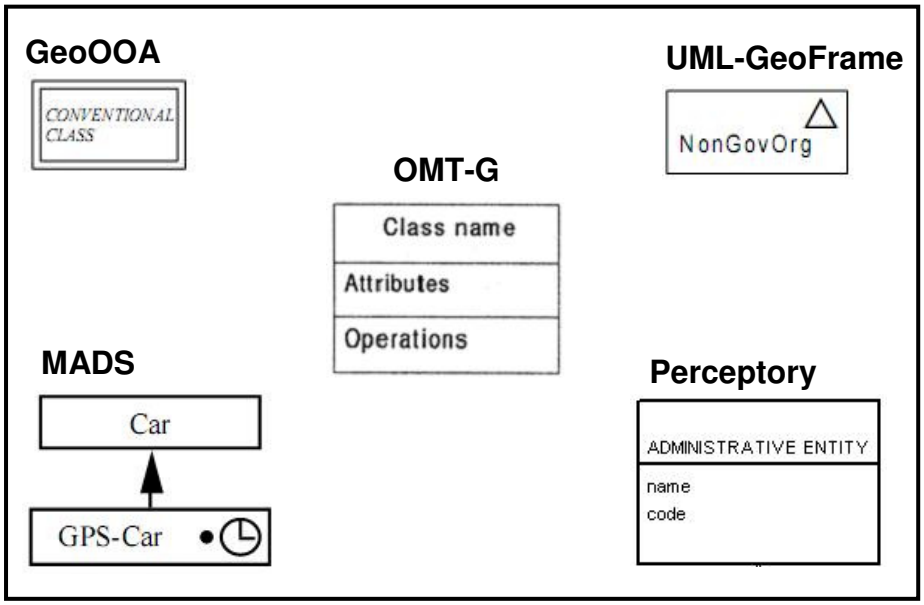


Figura 3.3 – Objetos convencionais em diferentes modelos
Fontes: (a) Kösters, Pagel e Six (1996); (b) Lisboa Filho e Iochpe (1999a); (c) Parent, Spaccapietra e Zimányi (1999); (d) Perceptory (2008); (e) Borges, Davis e Laender (2001)

3.3.2 Visões de campo e objetos

A classificação dos fenômenos geográficos nestas duas visões tem o intuito de representar de forma adequada a realidade geográfica observada. Segundo Goodchild, Yuan e Cova (2007), no início dos anos 1990 a terminologia para indicação de elementos contínuos ou discretos convergiu para *campo* e *objeto*. Enquanto na visão de *objeto* o mundo real é constituído de entidades com limites espaciais bem definidos, na visão de *campo* o mundo real é entendido como um conjunto de atributos que variam no espaço assim como uma função contínua. Conforme discutido na seção 3.1, enquanto a visão de campo é mais adequada para representação de fenômenos como relevo, temperatura ou poluição atmosférica, a visão de objeto é melhor empregada para lotes ou edificações.

Assim, a modelagem conceitual dos fenômenos geográficos necessita de construtores capazes de modelar tanto campos quanto objetos geográficos (LISBOA FILHO; IOCHPE, 1999b).

Os modelos GeoOOA, OMT-G e MADS possuem recursos para diferenciar os campos dos objetos geográficos. Entretanto, esta característica é deduzida através da representação espacial do fenômeno. O modelo UML-GeoFrame possui os estereótipos *GeographicField* (◬) e *GeographicObject* (◬), que são usados em conjunto com os estereótipos de representação espacial para diferenciação entre campos e objetos. A ferramenta Perceptory não possui suporte para modelagem de campos geográficos. Exemplos são mostrados nas seções seguintes.

3.3.3 Aspectos espaciais

Este requisito refere-se à necessidade de associar aos campos e objetos geográficos uma forma espacial abstrata para sua representação. Lisboa Filho e Iochpe (1999b) afirmam que a inclusão dos aspectos espaciais dos fenômenos geográficos no esquema conceitual é um fator fundamental na comunicação com o usuário.

Na visão de objetos os fenômenos são representados por ponto, linha, polígono ou combinações destes. Já na visão de campo, uma superfície contínua pode ser representada através de modelos numéricos, conjuntos de isolinhas ou grade de células, por exemplo. O tipo de representação a ser utilizado depende da finalidade da aplicação e das características do fenômeno.

Como dito anteriormente, nos modelos discutidos a representação dos fenômenos geográficos é feita por meio da adição de símbolos gráficos às classes. Neste aspecto, o modelo GeoOOA é o que possui a menor quantidade de opções. Os objetos geográficos são modelados como pontos, linhas ou regiões enquanto os campos podem ser modelados utilizando-se a classe *Raster*. A Figura 3.4 exibe exemplos de representação dos aspectos espaciais no modelo GeoOOA.

Figura 3.4 – Fenômenos geográficos no modelo GeoOOA
Fonte: Kösters, Pagel e Six (1996)

Os modelos OMT-G e UML-GeoFrame possuem um conjunto maior de opções para representação dos fenômenos geográficos, como pode ser observado na Figura 3.5. No OMT-G, os objetos geográficos podem ser do tipo (a) *Point*, *Line* ou *Polygon*, enquanto os campos são (b) *Triangular Irregular Network*, *Isolines*, *Planar subdivision*, *Tesselation* ou *Sampling*. As partes (c) e (d) da Figura 3.5 mostram os recursos disponíveis e dois exemplos para o modelo UML-GeoFrame. Há uma grande semelhança entre estes dois modelos, pois apresentam soluções praticamente equivalentes para este requisito. No caso da visão de objetos, o modelo UML-GeoFrame ainda dispõe do estereótipo *Complex*, que permite modelar objetos que possuem vários polígonos associados, por exemplo.

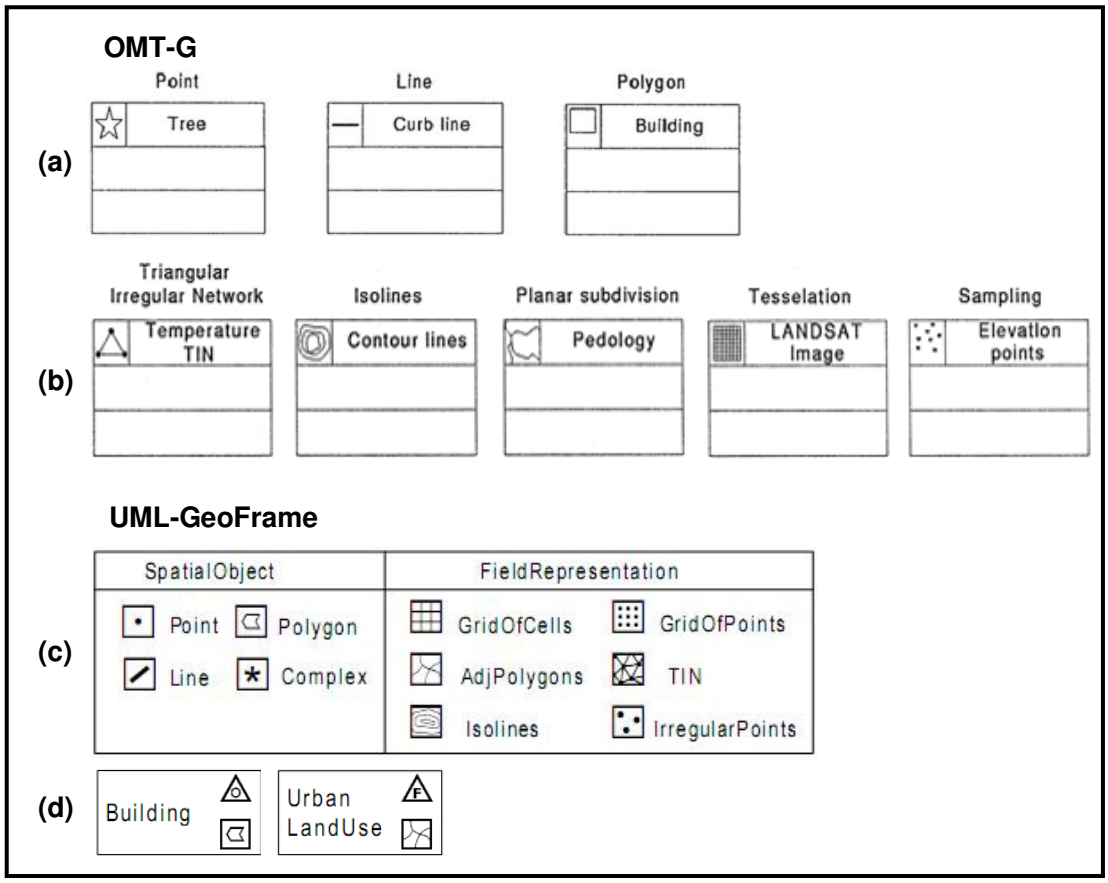


Figura 3.5 – Fenômenos geográficos nos modelos OMT-G e UML-GeoFrame
Fontes: (a,b) Borges, Davis e Laender (2001); (c,d) Lisboa Filho e Iochpe (1999a)

Na ferramenta Perceptory não é possível modelar campos geográficos. Entretanto, os objetos geográficos, representados por pontos, linhas e polígonos, podem ser considerados inclusive em um espaço tridimensional (Tabela 3.1). Estes pictogramas são combinados ou então levemente alterados para reproduzir casos

mais complexos (Tabela 3.2). Há ainda recursos disponíveis para modelagem de alguns outros casos especiais (Tabela 3.3).

Tabela 3.1 – Pictogramas para representação de objetos geográficos na Perceptory
Fonte: Bédard e Larrivée (2008)

	Espaço 2D	Espaço 3D	Exemplo de ocorrência
Geometria 0D			Hidrante
Geometria 1D			Segmento de rodovia
			Poste de eletricidade (linhas verticais)
Geometria 2D			Lago
			Muros (planos verticais)
Geometria 3D			Construções

Tabela 3.2 – Sintaxe para modelagem espacial avançada na Perceptory
Fonte: Bédard e Larrivée (2008)

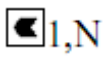
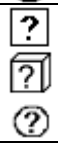
Geometria		Exemplo de sintaxe	Exemplo de ocorrência
Agregada	(complexa)		Hidrografia composta de rios e lagos (agregado de diferentes geometrias)
	(simples)		Cidades contendo várias ilhas (agregado de geometrias similares)
Alternativa			Construções tendo formato de ponto se área for menos que 1 hectare ou formato de polígono caso contrário (“ou” exclusivo)
Facultativa			Construções no banco de dados podem não ter geometria se área for menor que 0,5 hectare ou forma de ponto caso contrário
Derivada			Construções tridimensionais derivadas de construções bidimensionais com número de andares

Tabela 3.3 – Pictogramas para modelagem de casos especiais na Perceptory
Fonte: Bédard e Larrivée (2008)

Qualquer possibilidade		Significa que não há restrição quanto à geometria, pois sua forma não foi definida
Complicado		Indica que o elemento é melhor explicado textualmente ao invés de uma expressão contendo pictogramas
Ainda não definido		Durante o processo de modelagem, pode ser identificada a necessidade de representação espacial, mesmo esta não sendo definida ainda

Os pictogramas contidos nas tabelas anteriores são adicionados às classes da UML pela Perceptory. Na Figura 3.6 encontram-se dois exemplos, sendo um com geometria simples (a) e outro com geometria alternativa (b).

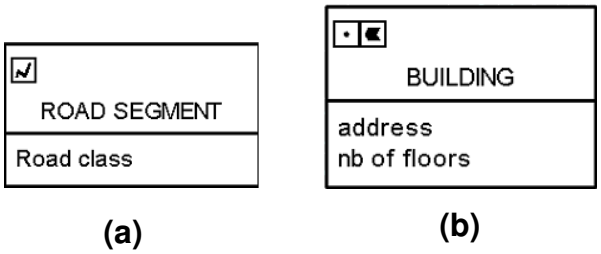


Figura 3.6 – Exemplos na ferramenta Perceptory
Fonte: Bédard e Larrivée (2008)

Finalmente, Parent, Spaccapietra e Zimányi (1999) descrevem uma hierarquia de tipos abstratos de dados espaciais que são responsáveis pela representação dos objetos geográficos no modelo MADS. Esta hierarquia consta na Figura 3.7 – parte (a). Com relação à modelagem de campos geográficos, os autores criam um novo tipo de atributo, denominado *space-varying*. Um exemplo é mostrado na Figura 3.7 (b). Neste caso, é dito que a escolha pela técnica de amostragem (ex.: TIN, grade de células) faz parte da definição do esquema lógico e, por esta razão, não há uma distinção entre os diferentes tipos de representação de campos no esquema conceitual. Recentemente, Parent, Spaccapietra e Zimányi (2008) publicaram o mesmo conceito no MADS, mas com alterações na apresentação, como é verificado em (c). O atributo *elevation* da classe *Country* é do tipo *space-varying*.

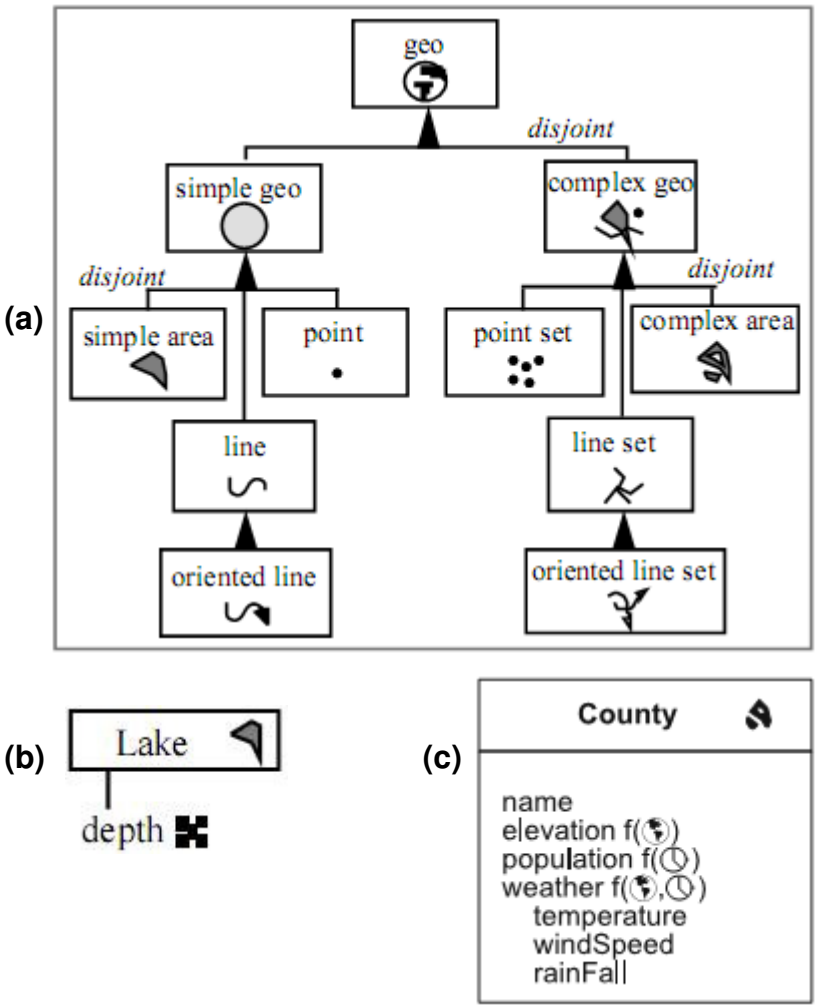


Figura 3.7 – Fenômenos geográficos no modelo MADS
Fontes: (a,b) Parent, Spaccapietra e Zimányi (1999); (c) Parent, Spaccapietra e Zimányi (2008)

3.3.4 Aspectos temáticos

Em um SIG, as entidades geográficas não são tratadas isoladamente. Elas são agrupadas conforme as características e relacionamentos que possuem em comum. No processo de modelagem descrito em (LISBOA FILHO; IOCHPE, 2008), é destacada a vantagem em se dividir um sistema por temas. Esta divisão permite a simplificação da modelagem e facilita a compreensão do domínio pelo projetista.

Apesar da existência deste requisito, os modelos geralmente não propõem um novo construtor para sua representação. No caso dos modelos baseados na UML, um tema pode ser reproduzido pelo uso do construtor denominado pacote.

A Figura 3.8 exibe um exemplo criado com o modelo UML-GeoFrame. É modelado um sistema de controle ambiental, que possui temas relacionados ao clima, hidrografia, relevo, vegetação e solo.

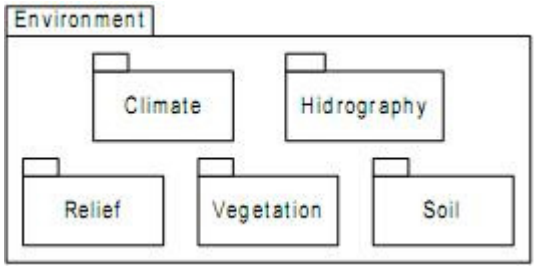


Figura 3.8 – Modelagem de temas com o UML-GeoFrame
Fonte: Lisboa Filho e Iochpe (1999a)

3.3.5 Múltiplas representações

A possibilidade de existência de múltiplas representações para um mesmo fenômeno geográfico é uma das características das aplicações geográficas. Os usuários podem ter diferentes visões de um mesmo fenômeno, que possivelmente são representados em diferentes escalas ou projeções (LISBOA FILHO; IOCHPE, 1999b).

Todos os modelos considerados satisfazem este requisito. Uma solução comum é alcançada por meio da adição de mais de um símbolo gráfico a uma mesma classe. Esta é adotada pelo UML-GeoFrame e a Perceptory, por exemplo. A Figura 3.9 (a) mostra uma situação na ferramenta Perceptory em que houve a necessidade de se modelar a classe *Building* com mais de uma representação devido às variações que ocorrem em certas mudanças de escala. Neste sistema, as construções são visualizadas como polígonos em grandes escalas e como pontos, linhas ou nada em pequenas escalas. Outra situação semelhante a anterior é demonstrada na Figura 3.9 (b), sendo utilizada a sintaxe do modelo OMT-G.

No MADS a múltipla representação também é tratada com a adição de mais de um símbolo à classe. Entretanto, ainda é permitido especificar atributos válidos apenas para determinada visão do objeto. No exemplo da Figura 3.9 (c) um segmento de estrada pode ter representação linear ou poligonal, sendo que a enumeração que indica o tipo do segmento varia para cada representação: se o segmento for visualizado em forma linear, seu tipo pode ser *European*, *National* ou *Local*, enquanto os valores para representação poligonal são *Highway* ou *National*. Além

disso, é possível indicar que um atributo pode possuir valores diferentes dependendo da representação do objeto, como ocorre no atributo *roadName*.

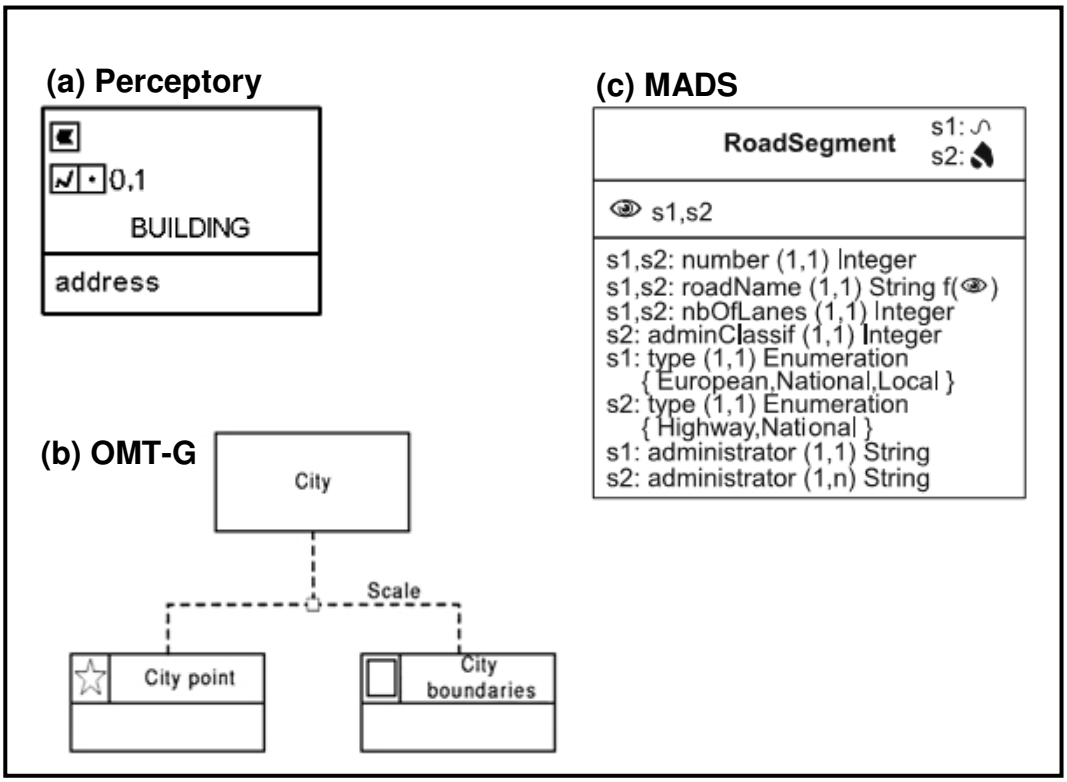


Figura 3.9 – Múltiplas representações em três diferentes modelos
Fontes: (a) Bédard e Larrivée (2008); (b) Borges, Davis e Laender (2001); (c) Parent, Spaccapietra e Zimányi (2008)

3.3.6 Relacionamentos espaciais

Segundo Lisboa Filho e Iochpe (1999b), uma das tarefas mais importantes da modelagem dos dados de um sistema é a identificação dos relacionamentos que devem ser mantidos no banco de dados. No domínio das aplicações geográficas este problema é complexo, pois o número de relacionamentos possíveis é maior, devido às interações espaciais que podem ocorrer entre os fenômenos geográficos.

Friis-Christensen, Tryfona e Jensen (2001) definem quatro tipos de relacionamentos espaciais (exemplos na seção 3.1): topológico, métrico, semântico e relacionamento para indicar que um objeto é composto de outros.

Em geral, os relacionamentos métricos não são representados no modelo, mas sim calculados durante a utilização dos dados geográficos por um SIG. Já os relacionamentos semânticos podem ser modelados através dos relacionamentos topológicos caso não haja algum recurso específico para este fim.

Todos os modelos estudados apresentam soluções, mesmo que parciais, para o requisito. Alguns utilizam os próprios construtores da linguagem em que são baseados enquanto outros disponibilizam novos construtores. Algumas características dos modelos que abordam este requisito com mais detalhes são apresentados a seguir.

O modelo GeoOOA oferece três estruturas para modelar relacionamentos entre um objeto e suas partes. Elas são chamadas de *covering strutucture* (o "todo" e suas partes pertencem ao mesmo tipo de classe geográfica e a geometria do "todo" é coberta pela união das geometrias das partes), *containment structure* (a geometria do "todo" contém a geometria de todas as partes) e *partition structure* (funciona como a estrutura anterior, porém o "todo" e suas partes pertencem à mesma classe geográfica e a geometria das partes constitui uma partição da geometria do "todo"). A Figura 3.10 (a) exibe a representação gráfica para cada uma delas.

No OMT-G, enquanto as associações convencionais são indicadas por linhas contínuas, nos relacionamentos espaciais esta linha é tracejada. O modelo considera nove diferentes relacionamentos neste caso: *touch*, *in*, *cross*, *overlap*, *disjoint*, *adjacent to*, *coincide*, *contain* e *near*. Na Figura 3.10 (b) encontra-se um exemplo.

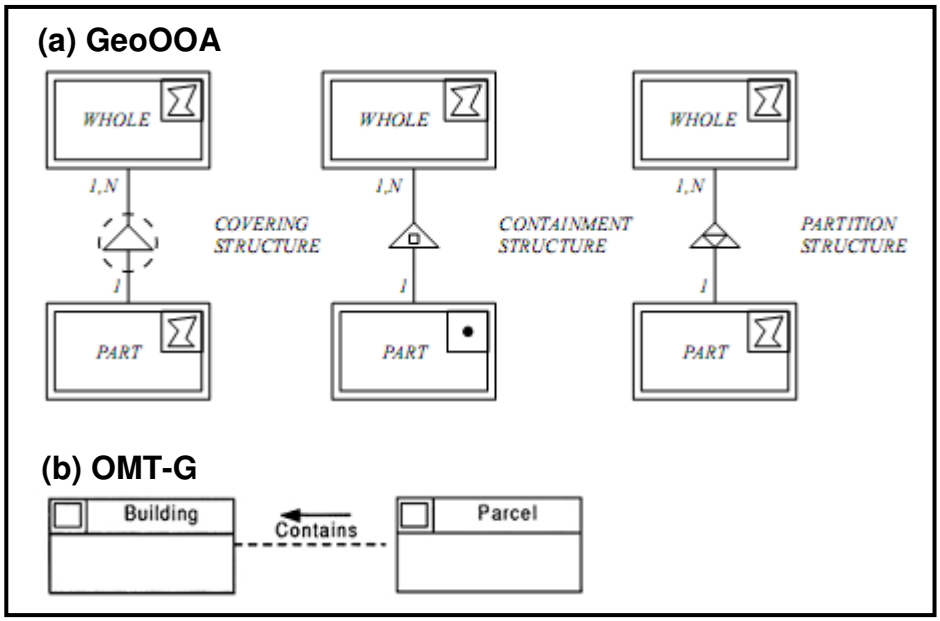


Figura 3.10 – Relacionamentos espaciais no GeoOOA e OMT-G
Fontes: (a) Kösters, Pagel e Six (1996); (b) Borges, Davis e Laender (2001)

O modelo MADS disponibiliza um conjunto de opções semelhante ao OMT-G neste requisito. A Tabela 3.4 mostra os nomes e ícones que representam estas

opções. Para indicar que um relacionamento é espacial no MADS, um destes ícones deve ser anexado ao relacionamento. Há ainda um ícone diferenciado para representação de relacionamentos de composição.

Tabela 3.4 – Tipos de relacionamentos espaciais no MADS
Fonte: Parent, Spaccapietra e Zimányi (1998)

Tipo espacial	Ícone
disjunction	
adjacency	
crossing	
overlapping	
inclusion	
equality	

Um esquema conceitual também deve ser capaz de representar relacionamentos espaciais que formam uma rede. O modelo GeoOOA permite esta modelagem adotando um símbolo (*Network*) para indicar que uma classe agrupa os elementos de uma rede e dois outros símbolos (*Link* e *Node*) que, utilizados em conjunto com a associação existente entre as classes da rede, configuram arcos e nós (Figura 3.11 – a).

O modelo OMT-G possui classes específicas para representação de nós, arcos unidirecionais e bidirecionais. Os relacionamentos de uma rede são indicados por duas linhas paralelas tracejadas. O nome dado à rede é anotado entre as duas linhas (Figura 3.11 – b).

O modelo UML-GeoFrame não possuía recursos para modelagem adequada de redes em sua proposta original. Porém, recentemente o modelo recebeu uma extensão realizada por Stempliuc (2008) que supriu esta ausência. Foi incluído um estereótipo para reprodução dos objetos que compõem a rede (N). Estes objetos podem significar nós, arcos unidirecionais ou bidirecionais, dependendo do estereótipo de representação de rede escolhido. A representação geográfica de um objeto da rede se faz com a adição de estereótipos relacionados à geometria da classe (ponto, linha ou polígono), como mostra a Figura 3.11 (c). O exemplo mostra parte da modelagem de um sistema de distribuição de energia elétrica. A rede de alta voltagem possui torres, que representam nós e têm informação geográfica do tipo

ponto, e linhas de transmissão, que representam arcos bidirecionais e contêm informação geográfica do tipo linha.

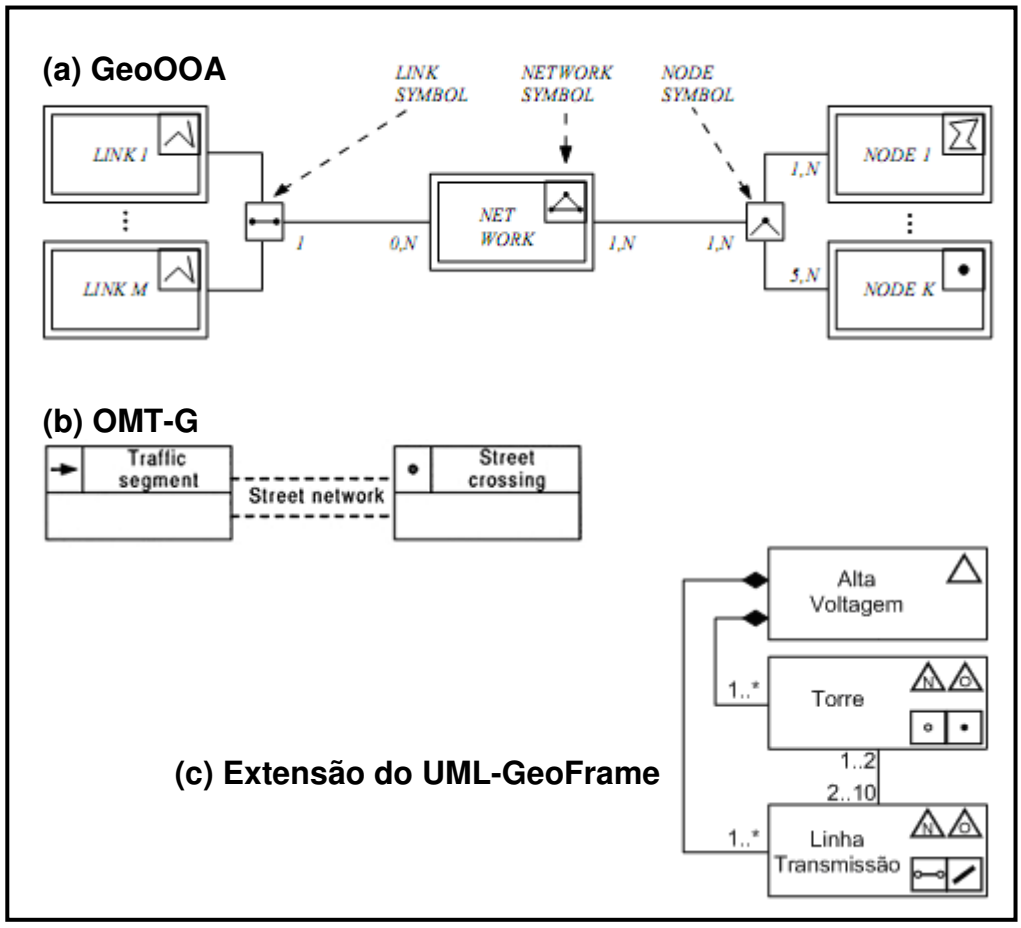


Figura 3.11 – Redes sendo representadas em três diferentes modelos
Fontes: (a) Kösters, Pagel e Six (1996); (b) Borges, Davis e Laender (2001); (c) Stempliuc (2008)

3.3.7 Aspectos temporais

O armazenamento das mudanças que ocorrem nos elementos geográficos é importante para uma melhor compreensão dos fenômenos e realização de previsões. De acordo com os requisitos temporais citados por Friis-Christensen, Tryfona e Jensen (2001), um modelo de dados geográficos deve ser capaz de representar o tempo de validade, de transação e de existência. O primeiro aplica-se a atributos e relacionamentos e indica quando o elemento é válido na realidade modelada. O segundo, além dos atributos e relacionamentos, aplica-se também a objetos e representa o tempo em que um elemento faz parte do estado atual do banco de dados. Já o último aplica-se somente a objetos e refere-se ao tempo em que o determinado

objeto existe, ou seja, o tempo de validade dos atributos e relacionamentos de um objeto deve estar contido no seu tempo de existência.

Em estudos focados nos aspectos temporais da informação geográfica, como o de Rocha (2001), uma lista maior de requisitos é encontrada. Entretanto, diante da análise realizada neste mesmo trabalho de Rocha, pode-se perceber que geralmente os modelos conceituais buscam atender em maiores detalhes aos requisitos encontrados por Friis-Christensen. Provavelmente, isso ocorre porque esses requisitos são capazes de caracterizar a grande maioria das necessidades de um SIG.

Certos autores não fazem distinção entre tempo de validade e de existência e consideram ambos sendo tempo de validade como, por exemplo, (LISBOA FILHO; IOCHPE, 2008) e (PARENT; SPACCAPIETRA; ZIMÁNYI, 2008). O trabalho apresentado nesta dissertação também agrupa estes dois conceitos sempre que o tempo de validade for mencionado.

GeoOOA, MADS e Perceptory possuem suporte ao tempo de validade, mas não consideram o tempo de transação. UML-GeoFrame e OMT-G não tinham suporte aos aspectos temporais em suas versões originais, porém posteriormente ganharam extensões que abordavam o requisito. Algumas características específicas de cada modelo são descritas a seguir.

O modelo GeoOOA utiliza um símbolo de um relógio para diferenciar classes temporais (Figura 3.12 - a). Estas possuem atributos que armazenam as datas de existência do objeto, assim como serviços para criar, restaurar e comparar versões destes. Também há possibilidade de representar relacionamentos temporais para indicar que uma classe é ancestral de outra.

O modelo MADS, conforme (PARENT; SPACCAPIETRA; ZIMÁNYI, 2008), disponibiliza três símbolos para representação temporal: *Instant* ( ), *TimeInterval* ( ) e *IntervalBag* ( ). Respectivamente, eles são utilizados para eventos que ocorrem em um ponto no tempo; eventos que possuem uma duração; e eventos de duração não contínua. Dois exemplos são mostrados na Figura 3.12 (b). Além da possibilidade de adicionar um desses três símbolos às classes, o MADS ainda permite que o símbolo f( ) seja associado a atributos ou geometrias para indicar que as mudanças que acontecem nessas propriedades devem ser armazenadas em um histórico. Finalmente, restrições temporais em relacionamentos, que são representadas por ícones específicos (ex.: *overlap* , *during*), são úteis para explicitar a ordem em que os eventos que participam do relacionamento ocorrem.

A ferramenta Perceptory utiliza pictogramas para representação de instantes (⏲) e intervalos (◔). Dependendo da localização desses pictogramas temporais no diagrama da classe, o conceito temporal caracterizado por eles pode variar entre existência ou evolução. O conceito de existência é definido para um objeto e suas fronteiras são o "nascimento" e "morte" do objeto. Se uma determinada classe não possui esta característica temporal, então quando um objeto "morre", ele é removido do banco de dados. Já o conceito de evolução é aplicável quando o estado de um objeto muda. Existem dois tipos de evolução: descritiva (o valor de um atributo é alterado) e espacial (a localização ou formato de um objeto varia). O conceito de evolução só deve ser aplicado em atributos ou geometrias em que o usuário deseja gerenciar o histórico de mudanças. Caso contrário, o antigo valor de um atributo ou geometria deve ser removido a cada atualização. A Figura 3.12 (c) mostra um exemplo na ferramenta Perceptory. O conceito de existência é definido pelo pictograma do canto superior direito. Os conceitos de evolução descritiva e evolução espacial são representados pelo pictograma temporal associado ao atributo e ao pictograma de representação geográfica, respectivamente.

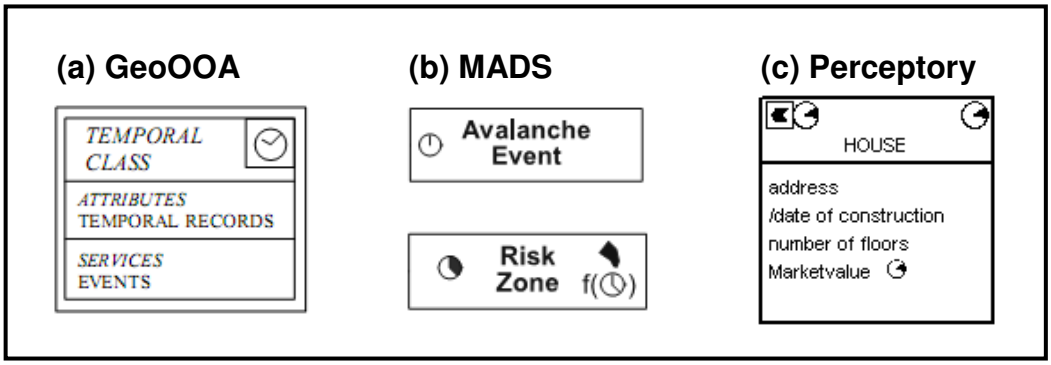


Figura 3.12 – Aspectos temporais no GeoOOA, MADS e Perceptory
Fontes: (a) Kösters, Pagel e Six (1996); (b) Parent, Spaccapietra e Zimányi (2008); (c) Perceptory (2008)

Em (ROCHA, 2001) é apresentada uma extensão temporal para o *framework* em que o modelo UML-GeoFrame é baseado, chamada GeoFrame-T. Foram criados estereótipos para representar a informação temporal através da combinação entre uma primitiva temporal (instante; intervalo; e elemento, que é a união de intervalos e instantes) e um tipo temporal (validade, transação e bitemporal). Além disso, é considerado o tempo ramificado. Neste conceito, é permitido que se tenha múltiplos futuros ou mais de uma estória passada, sendo um recurso importante para a

elaboração de previsões, segundo a autora. Há ainda um estereótipo (*Tempo Genérico*) para indicar que um elemento é temporal, mas sem detalhar a informação naquele momento. A Figura 3.13 exibe os estereótipos propostos, que podem ser adicionados às classes, atributos ou relacionamentos para caracterizar a informação geográfica.

Figura 3.13 – Estereótipos temporais do GeoFrame-T
Fonte: Rocha (2001)

De forma análoga ao trabalho de Rocha (2001), Meinerz (2005) realiza uma extensão temporal para o modelo OMT-G. São desenvolvidos estereótipos para representação de instantes e intervalos, além de serem considerados os tempos de transação, validade e ramificado.

3.4 Considerações finais

Após a realização dos estudos em torno do Perfil UML (Capítulo 2) e dos requisitos e modelos conceituais para bancos de dados geográficos (Capítulo 3), o desenvolvimento do GeoProfile pode ser iniciado.

Como discutido no início deste capítulo, um grande desafio deste campo de modelagem é conseguir balancear o poder de expressão do modelo com a facilidade de uso da notação disponibilizada por este. Dessa forma, com base nos modelos apresentados neste capítulo, procurou-se adicionar ao GeoProfile recursos capazes de atender aos requisitos encontrados, sem que o resultado final deste trabalho gerasse uma quantidade excessiva de construtores.

A tabela apresentada a seguir, além de resumir os resultados obtidos na análise comparativa entre os requisitos e modelos conceituais de bancos de dados geográficos, também exibe em sua última coluna os modelos que mais influenciaram a construção do GeoProfile em cada requisito, como é percebido no próximo capítulo.

Tabela 3.5 – Resumo da comparação entre requisitos e modelos apresentados; e principais contribuições para o GeoProfile

Modelos X Requisitos	GeoOOA	MADS	OMT-G	Perceptory	UML-GeoFrame	Contribuição para GeoProfile
Fenômenos geográficos e objetos convencionais	Sim	Sim	Sim	Sim	Sim	Perceptory
Visões de campo e objetos	Parcial	Parcial	Sim	Não	Sim	OMT-G
Aspectos espaciais	Parcial	Sim	Sim	Sim	Sim	OMT-G, UML-GeoFrame
Aspectos temáticos	Não	Não	Sim	Sim	Sim	UML-GeoFrame
Múltiplas representações	Parcial	Sim	Sim	Sim	Sim	UML-GeoFrame
Relacionamentos espaciais	Parcial	Sim	Sim	Parcial	Parcial	MADS, OMT-G
Aspectos temporais	Parcial	Sim	Não	Sim	Parcial	MADS, Perceptory

4 GeoProfile

A seguir é apresentado o GeoProfile, um Perfil UML desenvolvido para a modelagem conceitual de bancos de dados geográficos. Conforme os métodos propostos para guiar a construção de um Perfil UML, discutidos no Capítulo 2, dois artefatos são gerados durante o desenvolvimento do perfil: o metamodelo do domínio; e o perfil em si. Enquanto o primeiro é útil para compreender o problema abordado, o segundo apresenta as extensões recebidas pelas metaclasses da UML.

O metamodelo proposto para o domínio de banco de dados geográficos é descrito na Seção 4.1. Na Seção 4.2 é mostrado o mapeamento das classes do metamodelo do domínio em estereótipos. Por fim, as restrições OCL que foram anexadas aos estereótipos do GeoProfile são exibidas na Seção 4.3, concluindo assim a descrição do perfil.

4.1 Uma proposta de metamodelo para o domínio geográfico

No primeiro passo do desenvolvimento do GeoProfile, foi construído um metamodelo capaz de descrever o domínio considerado nesta dissertação. O objetivo desta tarefa é encontrar os elementos presentes em um esquema conceitual de BDG e as relações existentes entre eles, observando os requisitos deste tipo de modelagem conceitual.

A forma como cada modelo conceitual considerado nesta pesquisa (GeoOOA (KÖSTERS; PAGEL; SIX, 1997), MADS (PARENT; SPACCAPIETRA; ZIMÁNYI, 2008), UML-GeoFrame (LISBOA FILHO; IOCHPE, 2008), OMT-G (BORGES; DAVIS; LAENDER, 2001) e o modelo implementado na Perceptory (PERCEPTORY, 2008)) atende aos requisitos encontrados foi analisada, como descrito no Capítulo 3. A inclusão dos principais mecanismos presentes em cada um desses modelos no GeoProfile o torna capaz de modelar a maior parte das necessidades de um BDG. Isso porque a não ser que seja preciso descrever um requisito em maiores detalhes, como os aspectos temporais, por exemplo, o GeoProfile disponibiliza recursos suficientes para a modelagem de um BDG.

Dentre os modelos conceituais, o UML-GeoFrame é o que apresenta uma organização mais próxima de um metamodelo. Este modelo foi baseado em um

framework chamado GeoFrame. No GeoFrame é definida uma hierarquia de classes que representa os elementos presentes em um BDG. Assim, optou-se por iniciar a elaboração do metamodelo a partir de uma adaptação do GeoFrame. O resultado é exibido na Figura 4.1.

As regiões geográficas são caracterizadas por temas, o que é abstraído pela metaclasse *Theme*. Os temas podem ser formados por agregação de outros temas ou por objetos com (*GeoPhenomenon*) ou sem representação espacial (*ConventionalObj*). Quando se opta por associar uma representação espacial aos objetos de uma classe, é possível que o fenômeno geográfico seja percebido nas visões de campo (*GeoField*) ou objeto (*GeogObject*). Dependendo da técnica utilizada na aquisição da informação relativa ao campo geográfico, sua representação pode ser escolhida entre seis opções, conforme observação feita por Goodchild, Yuan e Cova (2007): *AdjPolygons*, *Isolines*, *TIN*, *GridOfPoints*, *GridOfCells* ou *IrregularPoints*. Para os objetos geográficos, sua representação pode ser do tipo ponto, linha, polígono ou complexa (a geometria do objeto é composta por outras geometrias simples).

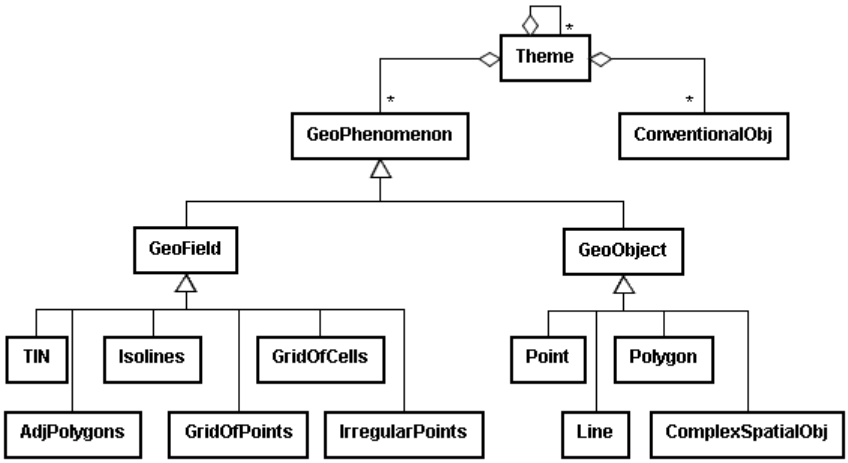


Figura 4.1 – Proposta inicial de metamodelo do domínio de BDG

Esta solução já é suficiente para atender a quatro requisitos dentre os sete listados na Seção 3.3. São eles: diferenciação entre fenômenos geográficos e objetos sem representação espacial; possibilidade de modelar os fenômenos nas visões de campo e de objetos; modelagem dos aspetos espaciais; e modelagem dos aspectos temáticos.

No caso das múltiplas representações, a solução adotada no GeoProfile define que este requisito deve ser caracterizado pela adição de mais de um estereótipo à classe do esquema conceitual. Logo, não houve necessidade de alterar o metamodelo.

Os requisitos relacionados aos papéis e metadados não são considerados nesta proposta inicial do GeoProfile. Apesar de representarem informações importantes relativas aos dados geográficos, acredita-se que elas não precisam ser necessariamente demonstradas durante a modelagem conceitual de um BDG.

Entretanto, essa solução inicial do metamodelo ainda é incompleta por não disponibilizar recursos capazes de modelar os relacionamentos espaciais e os aspectos temporais.

De acordo com a discussão realizada na Seção 3.3.6, os principais tipos de relacionamentos espaciais que devem ser representados em um esquema conceitual são os relacionamentos topológicos e os de composição. Para caracterizar o segundo tipo não houve necessidade de adicionar novos construtores ao GeoProfile, visto que a UML permite indicar que uma associação é do tipo composição ou agregação (graficamente, a associação adquire a forma de um losango preenchido ou não em uma de suas pontas). Porém, era necessário adicionar novos construtores para modelagem dos relacionamentos topológicos, incluindo a capacidade de representar redes.

Baseando-se nos modelos GeoOOA e OMT-G, que apresentam soluções mais detalhadas para representação de redes, Stempliuc et al. (2009) propôs uma extensão do GeoFrame para tratar do requisito. Esta extensão foi incluída no metamodelo após sofrer algumas adaptações. O resultado é exibido na Figura 4.2, sendo a primeira parte da solução final do metamodelo.

As classes responsáveis por armazenar os dados alfanuméricos e as informações sobre quais elementos participam da rede são representadas pela metaclasse *Network*. Por não possuir informações espaciais, esta metaclasse foi definida como uma especialização de *ConventionalObj*. As redes são formadas pelos objetos de rede (*NetworkObj*), que podem ser nodos (*Node*), arcos unidirecionais (*Unidirectional*) ou bidirecionais (*Bidirectional*). Embora na teoria de grafos uma rede mínima possa ser composta somente por um nó, no GeoProfile é restringido que toda rede deve possuir pelo menos um nó e um arco. Essa consideração é admitida por acreditar-se que em uma aplicação geográfica real não faz sentido manter uma

rede com um único nó. Além disso, uma outra restrição é o fato de todo nó ter que estar associado a um arco, e vice-versa.

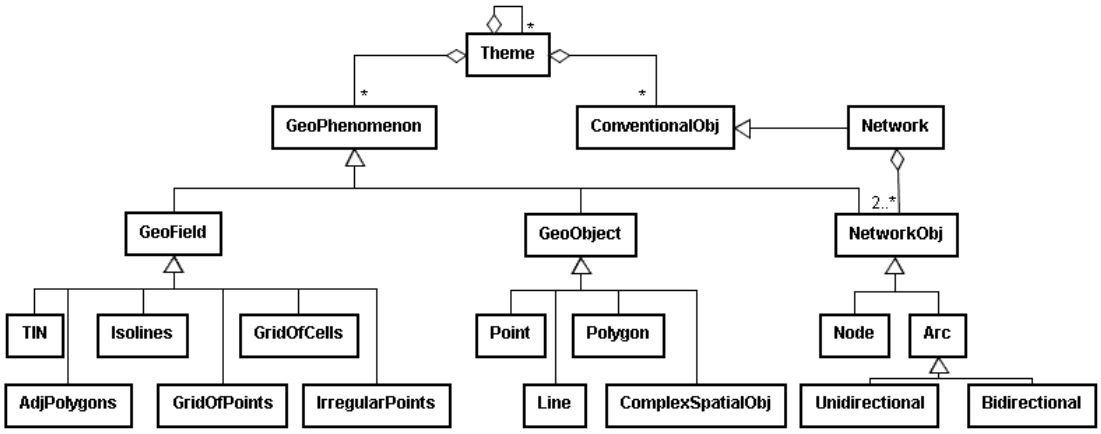


Figura 4.2 – Proposta final do metamodelo para o domínio de BDG (Parte I)

Os demais tipos de relacionamentos topológicos são definidos diretamente na criação dos estereótipos e das restrições OCL. Isto porque a grande quantidade de relacionamentos possíveis entre os objetos geográficos do tipo ponto, linha e polígono, sobrecarregaria excessivamente o metamodelo.

No tocante aos aspectos temporais, as abordagens MADS e Perceptory se destacam. Embora não considerem o tempo de transação, ícones adicionados em diferentes posições do diagrama da classe são capazes de indicar que deve ser mantido no banco de dados o tempo de existência do objeto, sua evolução espacial ou então a evolução dos valores de certos atributos daquela classe. Apesar de ser uma solução interessante, isso pode acabar sobrecarregando visualmente o esquema e conseqüentemente dificultando seu entendimento.

Outra solução, que é a adotada pelo GeoProfile, é indicar apenas se uma classe é considerada temporal ou não, como ocorre no modelo GeoOOA. Neste caso, fica implícito que tanto os atributos descritivos como os dados espaciais de um objeto podem variar, e que estas mudanças devem ser mantidas no banco de dados. Uma implementação para banco de dados relacional que utiliza esta técnica para tratar o tempo de validade pode ser encontrada em (LISBOA FILHO et al., 2007). Nesta solução, uma estrutura é gerada para armazenar classes temporais de forma que a cada variação em seu atributo ou geometria, uma nova versão do objeto é criada. Assim, o tempo de existência de um objeto pode ser obtido através da união dos intervalos de validade das suas versões.

Dessa forma, foi adicionada ao metamodelo a metaclasse *TemporalObject*. Ela possui dois atributos, que caracterizam a informação temporal. Um desses atributos indica o tipo temporal (tempo de validade, tempo de transação ou bitemporal), enquanto o outro define a primitiva temporal utilizada (instante ou intervalo). Foram criadas duas enumerações (*TemporalType* e *TemporalPrimitive*) para listar os possíveis valores que estes atributos podem assumir.

Preferiu-se não tratar a questão do tempo alterando a hierarquia principal de classes do metamodelo. Com isso, além desta estrutura ter se tornado mais simples, o que facilita a sua compreensão, é possível adicionar características temporais a qualquer classe do esquema livremente. Portanto, a Figura 4.3 mostra o restante do metamodelo que serviu de base para elaboração do GeoProfile.

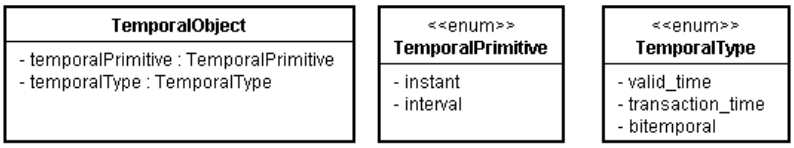


Figura 4.3 – Proposta final de metamodelo do domínio de BDG (Parte II)

4.2 Estereótipos para o GeoProfile

A partir do metamodelo do domínio é possível realizar as extensões das metaclasses da UML. Isso consiste em definir estereótipos e associar restrições OCL a eles.

Nota-se que 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. Já as classes de objetos convencionais são modeladas pelas classes da UML sem a adição de estereótipos. Assim, os próprios construtores da UML são capazes de reproduzir esses dois conceitos.

Outra observação importante é que alguns estereótipos são definidos como abstratos (*GeoObject*, *GeoField*, *NetworkObj* e *Arc*). Durante a utilização do GeoProfile, estes estereótipos não são disponibilizados para serem adicionados às classes do esquema. Entretanto, são úteis para organização dos elementos do perfil, além de permitir a adição de restrições comuns a todos os outros estereótipos criados como especialização desses, como é visto na próxima seção. Por exemplo, uma restrição que é comum aos estereótipos *UnidirectionalArc* e *BidirectionalArc* pode ser adicionada a *Arc*.

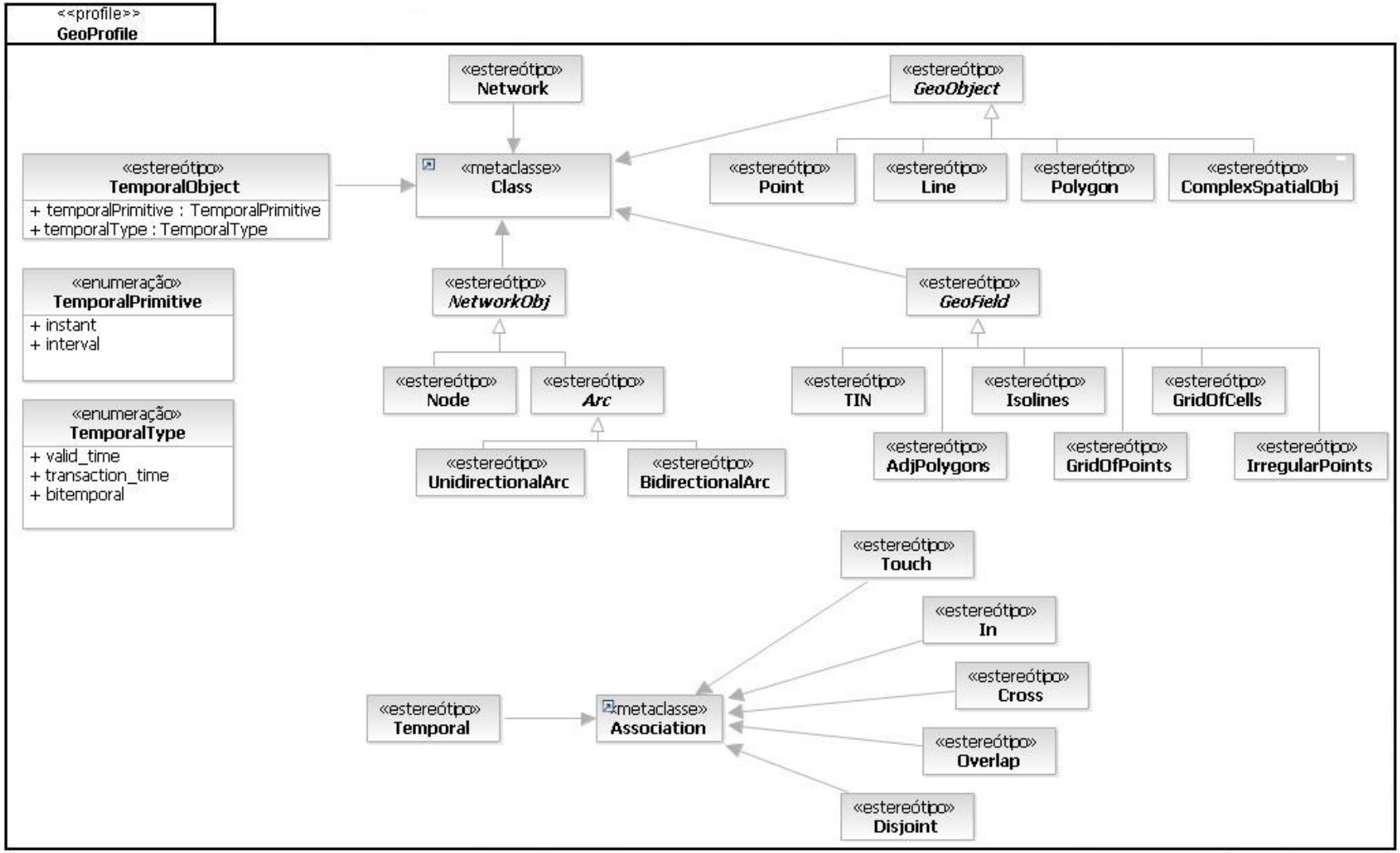


Figura 4.4 – Estereótipos do GeoProfile

A Figura 4.4 ilustra os estereótipos do GeoProfile. Os fenômenos geográficos, que estendem a metaclasse *Class*, são definidos em uma hierarquia equivalente a encontrada no metamodelo do domínio. O estereótipo *Network* estende diretamente a metaclasse *Class*, pois não há estereotipo definido para representação de objetos convencionais.

Para tratar os aspectos temporais, foi adicionado ao GeoProfile o estereótipo *TemporalObject*, assim como duas enumerações (*TemporalPrimitive* e *TemporalType*). Além disso, é permitido 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. Basta atribuir o estereótipo *Temporal*, que estende a metaclasse *Association*, a uma associação.

Por fim, foram criados estereótipos para representação do restante dos relacionamentos topológicos que não foram considerados durante a elaboração do metamodelo. Optou-se por utilizar o conjunto de cinco relacionamentos proposto por Clementini, Di Felice e Oosterom (1993), pois estes são capazes de representar qualquer relacionamento topológico existente entre objetos do tipo ponto, linha ou polígono. Logo, foram adicionados os estereótipos *Touch*, *In*, *Cross*, *Overlap* e *Disjoint*, sendo que todos estendem a metaclasse *Association*.

4.3 Restrições OCL

As restrições incluídas no GeoProfile têm como foco a validação do esquema conceitual gerado pelo projetista. Não faz parte do escopo deste trabalho definir restrições para verificar a consistência dos objetos de um sistema implementado a partir do esquema conceitual.

Dessa forma, as restrições OCL descritas nesta seção sempre possuem como contexto um estereótipo do GeoProfile, além de serem invariantes. A navegação entre os estereótipos e as classes do esquema conceitual se faz conforme a discussão realizada no Capítulo 2. Entretanto, o acesso à classe do esquema que recebe o estereótipo, que é feito através do comando "*base_<metaclasse>*", é omitido para simplificação do código. Algumas ferramentas, como a Rational Software Modeler (RATIONAL SOFTWARE MODELER, 2008), permitem que isso seja feito sem causar problemas de referência.

Outra consideração a ser feita refere-se ao método *getAppliedStereotypes()*, que é capaz de retornar todos os estereótipos aplicados a um elemento. Ele não está presente na especificação da OCL 2.0 (OMG, 2006). Durante a implementação do GeoProfile em alguma ferramenta que tenha suporte a perfis UML, esse método deve ser substituído por outro recurso semelhante disponibilizado pela ferramenta.

As restrições especificadas para o GeoProfile basicamente 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 (ex.: relacionamento *Cross* entre dois objetos geográficos com representação de ponto). Estes três grupos de restrições são analisados a seguir. Com relação aos aspectos temporais, como já mencionado, não existe limitação para a inclusão do estereótipo <<TemporalObject>> em classes do esquema ou de <<Temporal>> em suas associações.

4.3.1 Verificação da existência de estereótipos incompatíveis em um mesmo elemento

Este tipo de restrição é o que impõe a verificação mais simples. São capturados os estereótipos aplicados a cada classe e associação do esquema. Então é testado se naquele elemento foram adicionados estereótipos que não devem ser aceitos simultaneamente.

Estereótipos incompatíveis para as classes são aqueles que combinam características de objeto geográfico/campo geográfico; rede (que é um objeto convencional)/fenômenos geográficos; objeto de rede/campo geográfico; nodo/arco; arco unidirecional/arco bidirecional.

A Tabela 4.1 contém as restrições adicionadas ao GeoProfile para este tipo de verificação. Por exemplo, na restrição (a) é definido que para toda classe que receber um estereótipo de campo geográfico (`context GeoField`) devem ser capturados todos os estereótipos aplicados a ela (`getAppliedStereotypes`). No resultado obtido são selecionados os estereótipos do tipo objeto geográfico através do método `select`, sendo que o conjunto retornado deve ser vazio (`isEmpty`).

As demais restrições OCL adicionadas para impor restrições sobre a criação de esquemas livres deste tipo de erro apresentado nesta seção seguem o mesmo raciocínio.

Tabela 4.1 – Restrições OCL para verificação da existência de estereótipos incompatíveis em uma mesma classe do esquema

A	**context** GeoField **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Point' or s.name = 'Line' or s.name = 'Polygon' or s.name = 'ComplexSpatialObj') -> isEmpty()
B	**context** Network **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Point' or s.name = 'Line' or s.name = 'Polygon' or s.name = 'ComplexSpatialObj' or s.name = 'TIN' or s.name = 'Isolines' or s.name = 'GridOfCells' or s.name = 'AdjPolygons' or s.name = 'GridOfPoints' or s.name = 'IrregularPoints' or s.name = 'Node' or s.name = 'UnidirectionalArc' or s.name = 'BidirectionalArc') -> isEmpty()
C	**context** NetworkObj **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'TIN' or s.name = 'Isolines' or s.name = 'GridOfCells' or s.name = 'AdjPolygons' or s.name = 'GridOfPoints' or s.name = 'IrregularPoints') -> isEmpty()
D	**context** Node **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'UnidirectionalArc' or s.name = 'BidirectionalArc') -> isEmpty()
E	**context** UnidirectionalArc **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'BidirectionalArc') -> isEmpty()

No caso das associações, é impossível que uma mesma associação possua mais de um estereótipo referente aos relacionamentos topológicos, visto que eles são mutuamente exclusivos. Por exemplo, é impossível que um relacionamento seja do tipo *Cross* e *Disjoint* ao mesmo tempo. A Tabela 4.2 exibe as restrições OCL adicionadas ao GeoProfile para este tipo de verificação.

Tabela 4.2 - Restrições OCL para verificação da existência de estereótipos incompatíveis em uma mesma associação do esquema

F	**context** Cross **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Disjoint' or s.name = 'In' or s.name = 'Overlap' or s.name = 'Touch') -> isEmpty()
G	**context** Disjoint **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Cross' or s.name = 'In' or s.name = 'Overlap' or s.name = 'Touch') -> isEmpty()
H	**context** In **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Disjoint' or s.name = 'Cross' or s.name = 'Overlap' or s.name = 'Touch') -> isEmpty()
I	**context** Overlap **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Disjoint' or s.name = 'Cross' or s.name = 'In' or s.name = 'Touch') -> isEmpty()
J	**context** Touch **inv**: self.getAppliedStereotypes() -> select(s | s.name = 'Disjoint' or s.name = 'Cross' or s.name = 'In' or s.name = 'Overlap') -> isEmpty()

4.3.2 Validação das redes definidas no esquema

Como foi discutido durante a elaboração do metamodelo do domínio, as regras para definição de redes no GeoProfile determinam que toda rede deve possuir pelo menos um nodo e um arco. Ou seja, toda classe estereotipada por <<Network>> precisa estar associada à pelo menos um nodo (<<Node>>) e um arco (<<BidirectionalArc>> ou <<UnidirectionalArc>>).

Por exemplo, a restrição (k) define que toda classe que recebe o estereótipo <<Network>>, o que a faz ser responsável por armazenar os dados alfanuméricos e as informações sobre quais elementos participam da rede, deve estar conectada por uma associação a pelo menos uma classe que defina nodos de uma rede. Para isso, é feita a navegação partindo-se da classe estereotipada por <<Network>> até às classes conectadas a ela por alguma associação (`self.ownedAttribute.association.memberEnd.class`). Então é testado se no conjunto obtido existe alguma classe que possua o estereótipo <<Node>> através dos métodos `getAppliedStereotypes`, `select` e `isEmpty`.

De maneira semelhante, a restrição (l) garante que toda classe estereotipada por <<Network>> estará conectada a um arco para que o esquema seja considerado correto. Além disso, também é definido que todo nó deve estar conectado a pelo menos um arco, e vice-versa, o que dá origem às restrições (m) e (n).

Tabela 4.3 – Restrições OCL para validação das redes definidas no esquema

K	**context** Network **inv:** not self.ownedAttribute.association.memberEnd.class .getAppliedStereotypes() -> select(s | s.name = 'Node') -> isEmpty()
L	**context** Network **inv:** not self.ownedAttribute.association.memberEnd.class .getAppliedStereotypes() -> select(s | s.name = 'UnidirectionalArc' or s.name = 'BidirectionalArc') -> isEmpty()
M	**context** Arc **inv:** not self.ownedAttribute.association.memberEnd.class .getAppliedStereotypes() -> select(s | s.name = 'Node') -> isEmpty()
N	**context** Node **inv:** not self.ownedAttribute.association.memberEnd.class .getAppliedStereotypes() -> select(s | s.name = 'UnidirectionalArc' or s.name = 'BidirectionalArc') -> isEmpty()

4.3.3 Verificação da compatibilidade dos elementos envolvidos em um relacionamento topológico

Este último grupo de restrições limita o uso dos relacionamentos topológicos de acordo com os resultados obtidos por Clementini, Di Felice e Oosterom (1993). Conforme mencionado na Seção 3.2, estes autores definem cinco tipos de relacionamentos topológicos que podem ocorrer entre objetos com geometria de ponto, linha ou polígono, assim como as regras para sua utilização. Nos dois casos mais simples (*In* e *Disjoint*), basta verificar se as classes conectadas pela associação estereotipada possuem alguma representação geográfica do tipo ponto, linha ou polígono.

Por exemplo, na restrição (o) são capturadas as classes conectadas por uma associação estereotipada por <<In>> (`self.memberEnd.class`). Então é testado se cada uma dessas classes possui algum estereótipo de representação geográfica do tipo ponto, linha ou polígono, através dos métodos `forAll`, `getAppliedStereotypes`, `select` e `notEmpty`.

Tabela 4.4 – Restrições OCL para validação de relacionamentos topológicos do tipo In e Disjoint

O	**context** In **inv**: self.memberEnd.class -> forAll (c | c.getAppliedStereotypes() -> select(s | s.name = 'Point' or s.name = 'Line' or s.name = 'Polygon') -> notEmpty())
P	**context** Disjoint **inv**: self.memberEnd.class -> forAll (c | c.getAppliedStereotypes() -> select(s | s.name = 'Point' or s.name = 'Line' or s.name = 'Polygon') -> notEmpty())

O uso dos outros três relacionamentos topológicos disponíveis no GeoProfile é mais restrito. Por exemplo, *Cross* só é possível ocorrer entre elementos do tipo linha/linha e linha/área. Na restrição (q) é definido que as duas classes conectadas pela associação estereotipada por <<Cross>> são capturadas (`self.memberEnd.class`) e o resultado é armazenado em uma variável denominada *classes* (`let classes`). Então é testado separadamente para cada classe (`classes->at(i)`) se elas possuem representação geográfica compatível com o relacionamento topológico escolhido através de métodos já discutidos em outras restrições (`getAppliedStereotypes`, `select` e `notEmpty`).

Tabela 4.5 – Restrições OCL para validação de relacionamentos topológicos do tipo Cross, Overlap e Touch

Q	**context** Cross **inv**: let classes : OrderedSet(Class) = self.memberEnd.class -> asOrderedSet() in ((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name = 'Line') -> notEmpty() and (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name = 'Line') -> notEmpty()) or ((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name = 'Line') -> notEmpty() and (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name = 'Polygon') -> notEmpty()) or ((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name = 'Polygon') -> notEmpty() and (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name = 'Line') -> notEmpty())

R

```
context Overlap
      inv: let classes : OrderedSet(Class) = self.memberEnd.class
-> asOrderedSet()
      in

((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() )
```

S

```
context Touch
      inv: let classes : OrderedSet(Class) = self.memberEnd.class
-> asOrderedSet()
      in

((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Point') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Polygon') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Point') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Point') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() )
      or
((classes -> at(1)).getAppliedStereotypes() -> select(s | s.name
= 'Line') -> notEmpty() and
 (classes -> at(2)).getAppliedStereotypes() -> select(s | s.name
= 'Point') -> notEmpty() )
```

Lembrando que as restrições OCL do GeoProfile validam apenas o esquema conceitual, é válido dizer que um relacionamento topológico modelado entre classes que possuem múltipla representação está conceitualmente correto se o tipo deste relacionamento topológico puder existir para pelo menos uma representação geográfica das classes envolvidas. Neste caso, cabe à aplicação restringir qual representação geográfica da classe estará envolvida no relacionamento escolhido.

Com este último grupo de restrições é finalizada a especificação do GeoProfile. A utilização dos estereótipos em conjunto com as restrições OCL propostas para o GeoProfile em uma ferramenta compatível com Perfis UML é discutida no próximo capítulo.

5 Implementação e análise do GeoProfile

Uma das grandes vantagens em se utilizar um Perfil UML, como base para modelagem de um domínio específico, é aproveitar toda a infra-estrutura que a UML possui. Além da popularidade conquistada por esta linguagem, tanto no meio acadêmico quanto no empresarial, nas atualizações de algumas ferramentas comerciais com suporte às versões 2.x da UML já são encontrados recursos para sua extensão. Dessa forma, não é necessário desenvolver uma nova linguagem ou ferramenta para modelar determinado domínio. Além disso, o tempo de aprendizado é reduzido, visto que é aproveitada toda experiência que o projetista já possui na modelagem de sistemas convencionais projetados na UML.

Este capítulo apresenta uma ferramenta capaz de auxiliar na construção de perfis UML. É descrita a implementação completa do GeoProfile na Rational Software Modeler (RSM, 2008). Foram testados com sucesso os mecanismos para criação de estereótipos, assim como a validação automática de esquemas a partir da verificação das restrições OCL. Em (UML FORUM, 2008) é encontrada uma comparação entre algumas ferramentas comercias, sendo a lista atualizada periodicamente.

A Seção 5.2 exibe alguns exemplos de modelagem desenvolvidos com o auxílio da RSM tendo o GeoProfile implementado. Nessa mesma seção é feita ainda uma comparação entre a modelagem realizada no GeoProfile e em outros modelos. Por fim, na Seção 5.3 estão as conclusões deste capítulo.

5.1 Implementação na ferramenta CASE Rational Software Modeler

A RSM é produzida pela IBM e suporta a UML 2.1. Sua versão 7.0.5, que foi utilizada neste trabalho, pôde ser obtida na página da empresa na Internet e usada gratuitamente por 30 dias.

É possível modificar a interface da ferramenta conforme as preferências do usuário. A Figura 5.1 ilustra um exemplo, dividido em quatro partes: (1) Explorador do Projeto, que contém uma lista de todos os elementos pertencentes ao projeto; (2) compartimento para visualização dos diagramas; (3) propriedades de um elemento; (4) Paleta indicando os elementos disponíveis para serem adicionados.

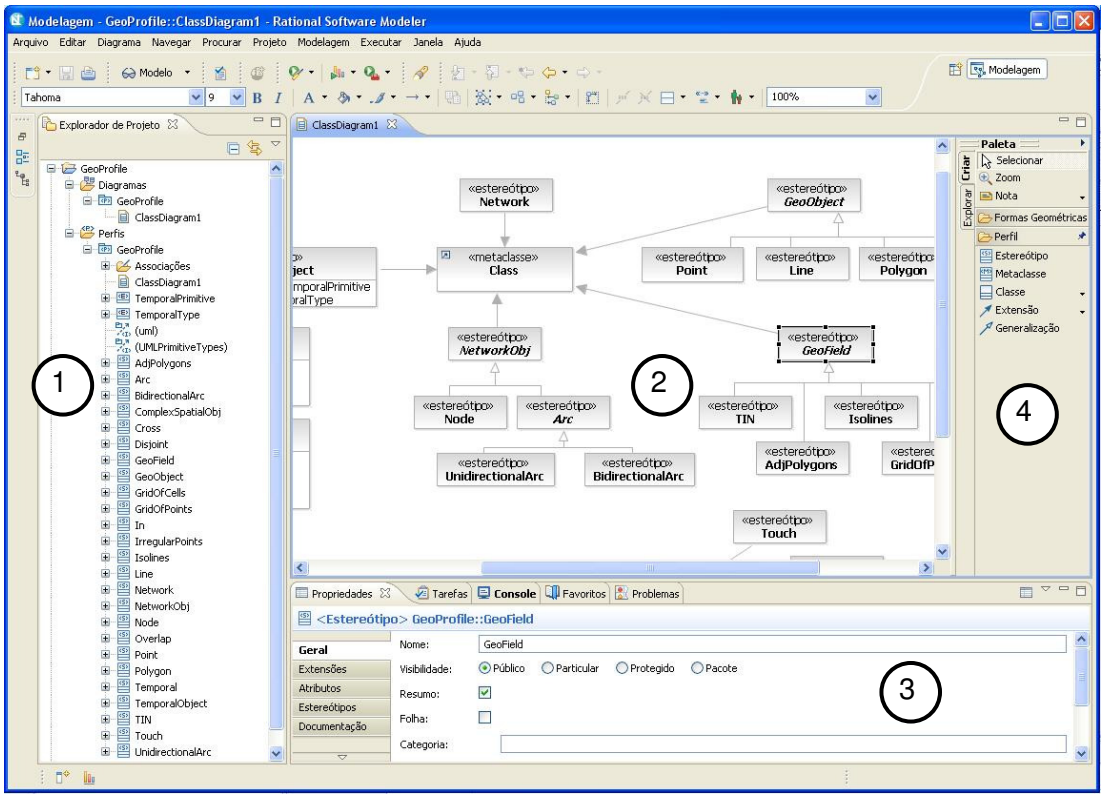


Figura 5.1 – Interface da ferramenta RSM

Em (MISIC, 2008) é encontrado um guia completo para desenvolvimento de Perfis UML na RSM. A seguir são discutidos as principais ações executadas na implementação do GeoProfile.

Inicialmente foi solicitada a criação de um novo projeto pelo menu principal (*Arquivo → Novo → Projeto*). Na janela que se abriu, bastou selecionar o assistente para criação de um *Projeto de Perfil UML* (*Modelagem → Extensibilidade do UML*) e continuar o processo informando alguns dados do projeto. A partir deste ponto foi possível adicionar ao perfil os elementos (ex.: estereótipos, enumerações, extensões) que constituem o GeoProfile, clicando nos itens da Paleta (Figura 5.1 – Parte 4).

Feito isso, era necessário então incluir nos estereótipos as restrições OCL adequadas. A RSM possui implementado o método *getAppliedStereotyes()*, que foi usado na descrição do GeoProfile exibida no Capítulo 4. Logo, as restrições OCL contidas na Seção 4.3 foram aproveitadas sem alterações. No painel Propriedades (Figura 5.2), as restrições puderam ser configuradas. Dentre as opções, é possível determinar o modo de validação da restrição: *Batch*, quando se deseja que a verificação e a informação sobre possíveis erros de modelagem sejam realizadas somente quando solicitado; ou *Ativo*, caso o projetista deseje que a verificação seja

realizada também quando ocorrem mudanças no esquema. Após o usuário digitar a expressão em OCL, a ferramenta informa se existe algum erro de sintaxe. Outro recurso bastante útil durante a elaboração das restrições é o auto-completar, que exibe as possíveis operações que podem ser utilizadas em um determinado ponto. Deste modo, pode-se dizer que a RSM dá um ótimo suporte à edição de comandos OCL, principalmente aos usuários menos experientes.

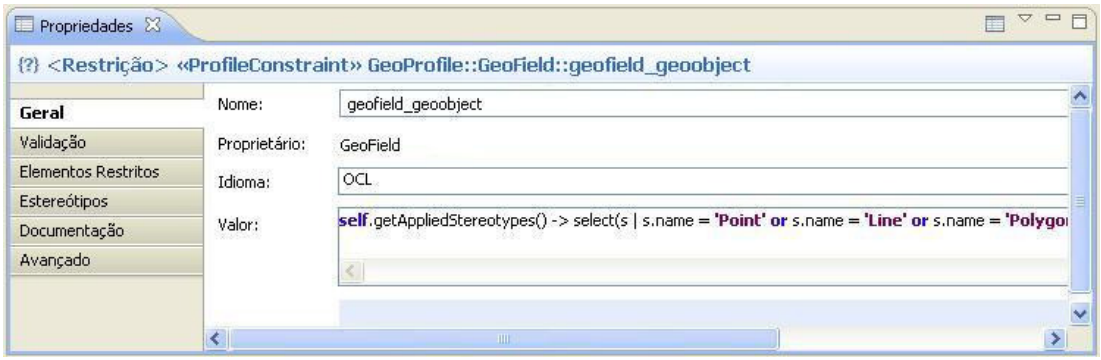


Figura 5.2 – Painel para edição de restrições OCL

Completada a definição do GeoProfile na RSM, foi criado um novo projeto para testar o perfil. No painel com as propriedades deste projeto, foi informado que se desejava aplicar o GeoProfile. Isso tornou possível a adição de seus estereótipos nos elementos do esquema construído.

Após alguns testes serem realizados foi solicitada a validação automática do esquema, o que evidenciou a utilidade deste recurso disponível na ferramenta. Supondo que um projetista tenha criado uma classe *Poste* e atribuído o estereótipo <<Node>> a ela, um erro é mostrado após a validação ser executada enquanto esta classe não estiver associada a outra classe que represente um arco da rede. Seguindo outra restrição do GeoProfile, uma classe denominada *Rodovia* também causará erro se for adicionada a ela um estereótipo com representação de campo geográfico e outro de objeto geográfico. Na Figura 5.3 são mostrados estes casos.

Em geral os mecanismos utilizados pela ferramenta na abordagem de perfis UML funcionaram corretamente. O único problema observado refere-se ao uso da generalização de classes. A especificação da UML (OMG, 2007) define que cada instância de uma subclasse é também uma instância da superclasse relacionada a ela. Por esta razão, todas as características especificadas para a superclasse são implicitamente especificadas para suas subclasses. Essa definição também inclui a herança das restrições aplicadas na superclasse.

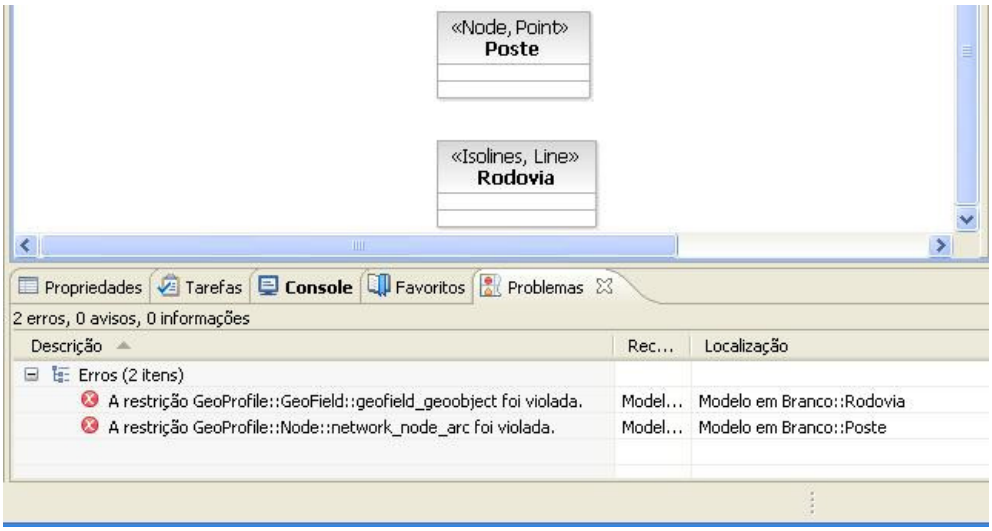


Figura 5.3 – Validação de um esquema: dois erros foram encontrados

Logo, pode-se dizer que os dois exemplos mostrados na Figura 5.4 não são tratados adequadamente pela RSM. No primeiro deles, é adicionado à subclasse (*Classe2*) um estereótipo de campo geográfico. Como esta classe possui implicitamente um estereótipo de objeto geográfico herdado da superclasse (*Class1*), um erro deveria ser informado, mas não é o que ocorre. Em outra situação, que foi modelada corretamente, um erro é encontrado pela ferramenta. Apesar da subclasse (*Classe4*) não conter nenhum estereótipo em seu diagrama, implicitamente ela herda <<Polygon>> de sua superclasse (*Class3*). Deveria ser permitida a criação de uma associação marcada por <<In>> entre a *Classe4* e *Classe5*.

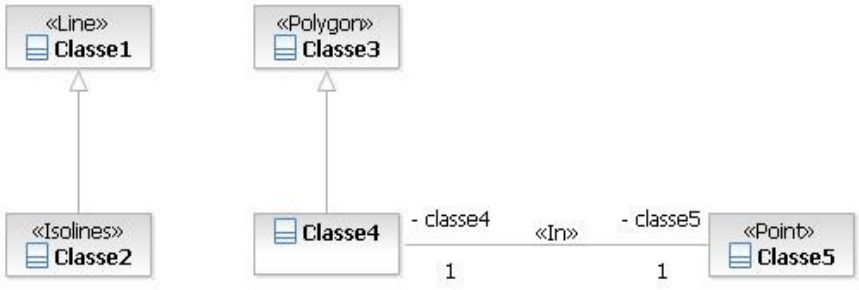


Figura 5.4 – Exemplo de problemas de modelagem com o uso de generalização

Este problema pode ser resolvido adicionando-se uma nova restrição OCL, para cada estereótipo do GeoProfile, que obrigue uma subclasse a possuir pelo menos os mesmos estereótipos de sua superclasse. Dessa forma, com a propagação dos estereótipos da superclasse para suas subclasses, a ferramenta RSM torna-se capaz de validar o esquema corretamente.

Por exemplo, para o estereótipo <<Point>> é definida a seguinte restrição:

```
context Point
inv: self.getModel().allOwnedElements() ->
select(g | g.oclAsType (Generalization)) ->
forAll(g | g.oclAsType(Generalization).general.
getAppliedStereotypes() -> select(s | s.name = 'Point') ->
notEmpty()
     implies
g.oclAsType(Generalization).specific.getAppliedStereotypes()
-> select(s | s.name = 'Point') -> notEmpty() )
```

Neste caso, primeiramente são selecionadas todas as generalizações do esquema (`self.getModel().allOwnedElements() -> select(g | g. oclAsType (Generalization))`). Então, para cada generalização, é testado se a superclasse possui o estereótipo <<Point>> (`forAll(g | g.oclAsType (Generalization).general.getAppliedStereotypes() -> select(s | s.name = 'Point') -> notEmpty()`). Se o resultado for verdadeiro, então é imposto que as subclasses também devem possuir o estereótipo <<Point>> (`implies g.oclAsType(Generalization).specific.getApplied Stereotypes() -> select(s | s.name = 'Point') -> notEmpty()`). Para os outros estereótipos do GeoProfile, uma restrição equivalente a esta deve ser definida.

5.2 Comparação entre o GeoProfile e outros modelos

Considerando os resultados obtidos após a definição do GeoProfile e sua implementação na RSM, nesta seção são exibidos exemplos de modelagem conceitual utilizando o GeoProfile. Com o objetivo de realizar uma comparação entre o GeoProfile e alguns modelos conceituais que serviram de base para a sua definição, para cada exemplo é mostrado também o esquema conceitual correspondente em outro modelo.

A Figura 5.5 ilustra parte da modelagem conceitual de um sistema para controle de poluição em lotes de terra. Nesta figura, (a) e (b) exibem a modelagem desenvolvida utilizando-se o modelo GeoOOA e o GeoProfile, respectivamente. Os lotes possuem uma representação poligonal. Para cada lote, deve ser armazenado um dado não geográfico capaz de informar quem são os donos de cada lote. Além disso, cada lote pode conter vários controles de poluição, que são representados geograficamente por pontos. Nesta associação entre lotes e pontos de controle de

poluição é representado no esquema conceitual a restrição de que cada ponto de controle deve estar contido na área do lote associado a ele.

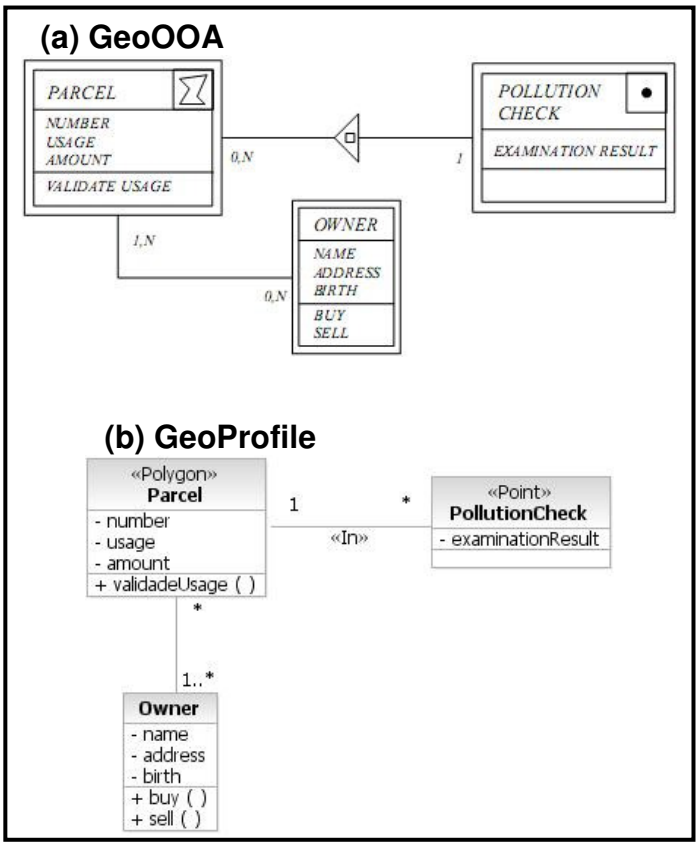


Figura 5.5 – Comparação entre GeoOOA e GeoProfile
Fonte: (a) Kösters, Pagel e Six (1996)

Outro exemplo semelhante de relacionamento topológico envolvendo lotes de terra é exibido na Figura 5.6 utilizando-se (a) o modelo MADS e o (b) GeoProfile. No esquema conceitual é informado que cada lote pode conter várias construções, sendo que ambas as classes possuem representação poligonal. Além disso, é restringido que as construções pertencentes a um determinado lote devem ter sua área geográfica contida na área do lote. No caso do relacionamento topológico *In*, pode ser importante, para uma correta interpretação do esquema, explicitar qual dos objetos envolvidos em uma determinada associação deve possuir sua geometria contida na geometria do outro objeto que participa da associação. No exemplo da Figura 5.6 (b), foram utilizados os papéis da associação para este fim. Uma outra opção é indicar a navegabilidade da associação. Ambas as soluções utilizam recursos presentes na especificação da UML.

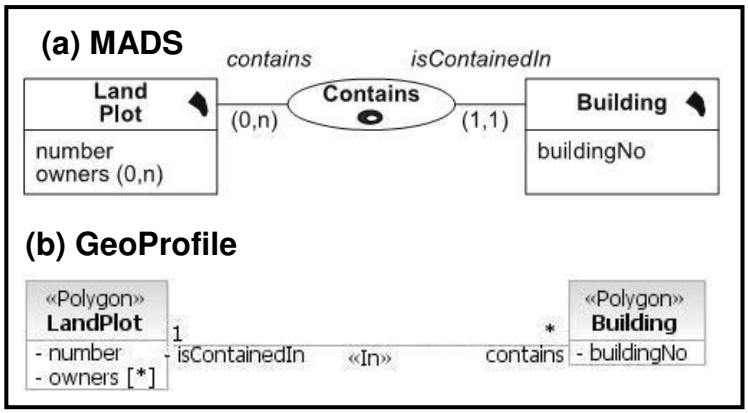


Figura 5.6 – Comparação entre MADS e GeoProfile.
(a) Adaptado de Parent, Spaccapietra e Zimányi (2008)

O próximo exemplo aborda a construção de redes. A Figura 5.7 ilustra um mesmo esquema desenvolvido por meio do modelo OMT-G e do GeoProfile. Uma rede de ruas é composta por segmentos de tráfego (arcos da rede) e por cruzamentos (nodos da rede). Neste caso, no modelo OMT-G o nome da rede (*Street network*) aparece entre duas linhas tracejadas que representam uma associação de rede. Já no GeoProfile, uma classe sem representação espacial é criada para agrupar os atributos pertencentes à rede, além de ser responsável por indicar quais arcos e nodos participam de cada rede.

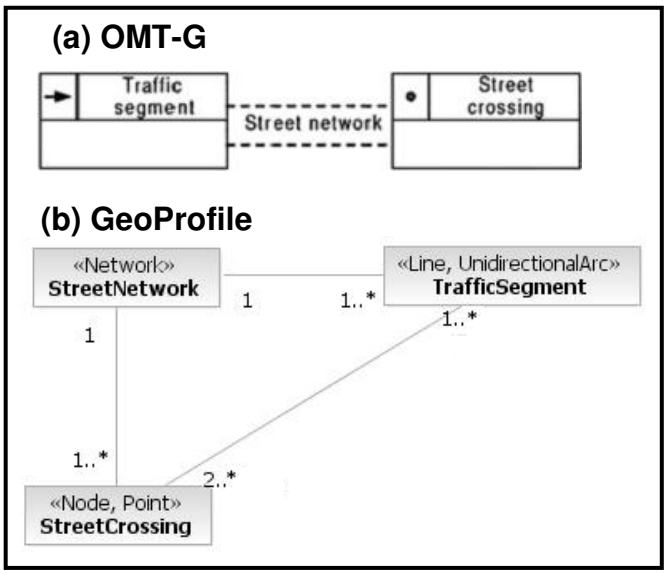


Figura 5.7 – Comparação entre OMT-G e GeoProfile
Fonte: (a) Borges, Davis e Laender (2001)

Por fim, no último exemplo são considerados os aspectos temporais. Na Figura 5.8 (a) é modelado uma classe *House* utilizando-se a ferramenta Perceptory. O pictograma (◔) localizado no canto superior direito do diagrama desta classe

indica que o período em que a casa existir na realidade modelada (ex.: data de construção até a data de demolição) deve ser armazenado no banco de dados. Já o mesmo pictograma, quando adicionado ao lado do pictograma de representação espacial ou ao lado de um atributo, indica que o histórico da evolução espacial do objeto ou a evolução dos valores de um atributo, respectivamente, deve ser mantido no banco de dados. Como já discutido anteriormente, no GeoProfile estes conceitos são agrupados em apenas um estereótipo, denominado <<TemporalObject>>. Neste caso, fica implícito que tanto o período de existência quanto o histórico da evolução dos atributos ou geometria de um objeto devem ser mantidos no banco de dados.

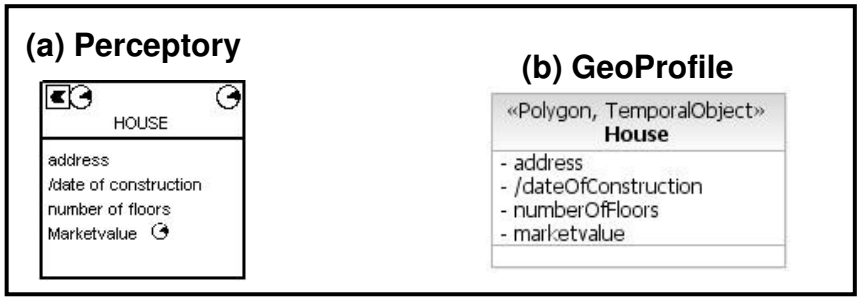


Figura 5.8 – Comparação entre Perceptory e GeoProfile
Fonte: (a) Perceptory (2008)

5.3 Considerações finais

Baseando-se nas principais características de modelos bem aceitos e difundidos no campo da modelagem conceitual de bancos de dados geográficos, o GeoProfile tornou-se capaz de representar todos os requisitos necessários nesta modelagem.

Além disso, a Seção 5.1 apresentou uma grande vantagem do GeoProfile frente aos outros modelos considerados. Ao ser construído conforme os padrões especificados na documentação da UML, este Perfil UML pode ser facilmente utilizado em ferramentas que seguem a mesma especificação UML.

O GeoProfile, além de atender aos requisitos indispensáveis do tipo de modelagem que aborda, é também facilmente manipulado em um ambiente computacional, o que agiliza a construção, validação e transferência de esquemas. Todas estas características são fundamentais para o desenvolvimento de um padrão de modelagem, pois ampliam o número de usuários e a sua própria aceitação nesta comunidade.

A tabela a seguir é semelhante à Tabela 3.5. Contudo, além de mostrar os requisitos atendidos por cada modelo, também exibe a mesma comparação para o GeoProfile.

Tabela 5.1 – Comparação entre requisitos e modelos, incluindo o GeoProfile

Modelos X Requisitos	GeoOOA	MADS	OMT-G	Perceptory	UML-GeoFrame	GeoProfile
Fenômenos geográficos e objetos convencionais	Sim	Sim	Sim	Sim	Sim	Sim
Visões de campo e objetos	Parcial	Parcial	Sim	Não	Sim	Sim
Aspectos espaciais	Parcial	Sim	Sim	Sim	Sim	Sim
Aspectos temáticos	Não	Não	Sim	Sim	Sim	Sim
Múltiplas representações	Parcial	Sim	Sim	Sim	Sim	Sim
Relacionamentos espaciais	Parcial	Sim	Sim	Parcial	Parcial	Sim
Aspectos temporais	Parcial	Sim	Não	Sim	Parcial	Sim

6 Conclusões e trabalhos futuros

Neste capítulo são discutidos os resultados obtidos pelo trabalho apresentado nesta dissertação, assim como sugestões de tarefas a serem realizadas futuramente.

6.1 Contribuições deste trabalho

A existência de vários modelos conceituais traz prejuízo para os usuários e projetistas de SIG, os quais ficam impedidos de migrarem seus projetos de uma ferramenta CASE para outra. Outro grande problema com esta falta de padronização é a dificuldade de treinamento e capacitação dos projetistas, uma vez que os modelos, embora tenham sido elaborados com o mesmo propósito, cada um tem suas particularidades. O usuário acostumado com um modelo (e respectiva ferramenta CASE) cria grande resistência para conhecer e utilizar outro modelo.

A principal contribuição deste trabalho foi mostrar que atualmente é possível desenvolver um padrão para modelagem conceitual de BDG. Tendo como base modelos conceituais encontrados na literatura e os recursos de extensão da UML, foi desenvolvido um Perfil UML próprio para a modelagem conceitual deste domínio, denominado GeoProfile.

Durante a elaboração do GeoProfile, procurou-se adicionar ao perfil os principais recursos que se destacavam em cada modelo estudado. Dessa forma, com o GeoProfile é possível modelar a maior parte dos fenômenos geográficos que podem ser caracterizados nestes outros modelos e isto consequentemente é um fator importante para a aceitação deste perfil como padrão de modelagem.

A adoção de um perfil UML vem resolver a maioria dos problemas citados acima. Além da UML já ter grande aceitação por parte da comunidade desenvolvedora de software, a disponibilidade de ferramentas CASE com suporte para perfis descarta a necessidade de implementação de ferramentas específicas.

Para evidenciar as vantagens de se utilizar um Perfil UML foi realizada a implementação do GeoProfile em uma ferramenta CASE. Com facilidade a ferramenta tornou-se capaz de disponibilizar os estereótipos do perfil. Além disso, visto que a RSM (ferramenta escolhida nesta implementação) possui suporte para a inclusão de restrições OCL nos estereótipos, foi possível também observar que a

validação automática de esquemas conceituais é um importante recurso para eliminar erros básicos de modelagem, aumentando a produtividade e a qualidade dos bancos de dados.

Portanto, dados os resultados obtidos, é possível concluir que todos os objetivos enumerados na Seção 1.2 foram alcançados com sucesso por este trabalho.

6.2 Trabalhos futuros

A proposta do GeoProfile contida nesta dissertação pode ser vista como um primeiro passo para padronização da modelagem conceitual de BDG. Espera-se que este Perfil UML receba contribuições da comunidade de pesquisadores e desenvolvedores de SIG. Então, futuramente deve-se submeter o GeoProfile à OMG para que este seja disponibilizado no site da organização.

Também deve ser investigada a possibilidade de incluir no GeoProfile a opção de substituir a representação textual do estereótipo por uma representação gráfica, como ocorre na maioria dos modelos conceituais.

Além disso, é importante disponibilizar uma implementação do GeoProfile em outras ferramentas CASE a fim de ampliar o uso deste Perfil UML. A utilização do GeoProfile em diferentes projetos de modelagem conceitual de BDG com certeza seria uma grande contribuição para verificar se há necessidade de inclusão de novos recursos neste perfil UML.

À medida que o GeoProfile torne-se um padrão de modelagem, um próximo passo é trabalhar na transformação conceitual-lógico dos esquemas produzidos com base no GeoProfile. Uma característica interessante é possibilitar a transformação do esquema conceitual de acordo com os padrões lógicos definidos pelo OGC e ISO. Além disso, outros aspectos que não faziam parte da proposta inicial deste Perfil UML podem ser incorporados futuramente. Por exemplo, em (KANG et al., 2004) são definidas restrições OCL para verificação da consistência de relacionamentos existentes entre objetos geográficos presentes em um banco de dados. Na versão atual do GeoProfile, as restrições validam apenas a construção do esquema conceitual, sem considerar os objetos criados com base neste esquema.

Referências bibliográficas

BÉDARD, Y. Visual modeling of spatial databases: towards spatial PVL and UML. **Geomatica**, vol. 53, n°2, p. 169-186. 1999.

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

BÉDARD, Y.; LARRIVÉE, S.; PROULX, M.; NADEAU, M. Modeling Geospatial Databases with Plug-Ins for Visual Languages: A Pragmatic Approach and the Impacts of 16 Years of Research and Experimentations on Perceptory. In: ER Workshops 2004 CoMoGIS, 2004, China. **Proceedings…** Berlin: Springer-Verlag, 2004. p. 17-30.

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.

CÂMARA, G; DAVIS, C; MONTEIRO, A. M. V. **Introdução à Ciência da Geoinformação.** São José dos Campos: Inpe: Editora, 2003. Disponível em: <www.dpi.inpe.br/gilberto/livro/introd/>. Acesso em: 14 mar. 2008.

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. 530 p. 277-295.

DMTF - Distributed Management Task Force. UML Profile for CIM. Versão 1.0.0b, 2007. Disponível em: <www.dmtf.org/standards/published_documents/DSP0219.pdf >. Acesso em: 25 ago. 2008.

ENTERPRISE ARCHITECT. Disponível em: <www.sparxsystems.com/products/ea/index.html>. Acesso em: 25 ago. 2008.

ERIKSSON, H.; PENKER, M.; LYONS, B.; FADO, D. **UML 2 Toolkit**. Indianapolis: Wiley, 2004.

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. 2, p. 6-13. 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. 2007.

KANG, M.; PINET, F.; SCHNEIDER, M.; CHANET, J. and VIGIER, F. How to Design Geographic Databases? Specific UML Profile and Spatial OCL Applied to Wireless Ad Hoc Networks. In: Agile Conference on Geographic Information Science, 7, 2004, Heraklion, Geece. **Proceedings...** . p. 289 - 299.

KÖSTERS, G.; PAGEL, B.; SIX, H. GeoOOA: object-oriented analysis for geographic information systems. In: International Conference on Requirements Engineering, 2, 1996, Colorado Springs, CO, USA. **Proceedings...** IEEE, 1996. p. 245 - 253.

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

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

LISBOA FILHO, J.; IOCHPE, C. Specifying analysis patterns for geographic databases on the basis of a conceptual framework. In: International Symposium on Advances in Geographic Information Systems, 7, 1999, Kansas City. **Proceedings…** New York: ACM, 1999a. 166 p. 7-13.

LISBOA FILHO, J.; IOCHPE, C. Um Estudo sobre Modelos conceituais de dados para projeto de bancos de dados geográficos. **IP. Informação pública**, Belo Horizonte, v. 1, n. 2, p. 67-90, 1999b.

LISBOA FILHO, J.; SAMPAIO, G. B.; SILVA, E. O.; GAZOLA, A. Design and inplementarion of the valid time for spatio-temporal databases. In: International Conference on Enterprise Information Systems (ICEIS), 2007, Funchal-Madeira. **Proceedings…** Miami : Florida International University, 2007. p. 569-573.

MEINERZ, G. V. **OMT-G Temporal: Uma Técnica de Extensão do Modelo OMT-G para Representar os Aspectos Temporais de Dados Geográficos**. 2005. Tese de Mestrado – Instituto Tecnológico de Aeronáutica, São José dos Campos.

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: 25 ago. 2008.

OBJECT MANAGEMENT GROUP. **Object Constraint Language**. Versão 2.0, 2006.

OBJECT MANAGEMENT GROUP. Profile Catalog. Disponível em: < http://www.omg.org/technology/documents/profile_catalog.htm>. Acesso em: 25 ago. 2008.

OBJECT MANAGEMENT GROUP. **Unified Modeling Language: Infrastructure**. Versão 2.1.2, 2007.

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

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.

PARENT, C.; SPACCAPIETRA, S.; ZIMÁNYI, E.; et al. Modeling Spatial Data in the MADS Conceptual Model. In: International Symposium on Spatial Data Handling, 8., 1998, Vancouver, Canada. **Proceedings...** . p. 138-150.

PERCEPTORY. Disponível em: <http://sirs.scg.ulaval.ca/perceptory/>. Acesso em: 14 mar. 2008.

RATIONAL SOFTWARE MODELER. Disponível em: <www-01.ibm.com /software/awdtools/modeler/swmodeler/>. Acesso em: 25 ago. 2008.

ROCHA, L. V. **GeoFrame-T: um Framework Conceitual Temporal para Aplicações de Sistemas de Informação Geográfica.** 2001. Dissertação (Mestrado) - Universidade Federal do Rio Grande do Sul, Porto Alegre.

SELIC, B. A Systematic Approach to Domain-Specific Language Design Using UML. In: 10TH IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC'07), 2007. **Proceedings...** Washington: IEEE, 2007. p. 2-9.

STEMPLIUC, S. M. **Modelagem de restrições de integridade espaciais em aplicações de rede através do modelo UML-GeoFrame**. 2008. Dissertação (Mestrado) - Universidade Federal de Viçosa, Viçosa-MG.

STEMPLIUC, S. M.; LISBOA F., J.; ANDRADE, M. V. A.; BORGES, K. V. A. Extending the UML-GeoFrame data model for conceptual modeling of network applications. In: International Conference on Enterprise Information Systems (ICEIS), 11., 2009, Milão, Itália. **Proceedings...** Milão: INSTICC/ACM-SIGMIS, 2009. p. 164-170.

UML FORUM. Disponível em: <www.uml-forum.com/tools.htm>. Acesso em: 25 ago. 2008.

WARMER, J.; KLEPPE, A. **The Object Constraint Language:** Getting Your Models Ready for MDA. 2. ed. Boston: Addison Wesley, 2003.