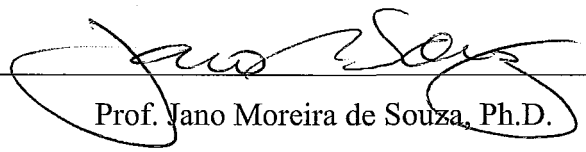
UMA ARQUITETURA EXTENSÍVEL PARA INTEGRAÇÃO E PUBLICAÇÃO DE DADOS GEORREFERENCIADOS

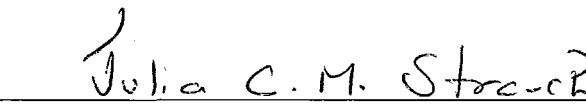
Lúcio Rogério Botêlho

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

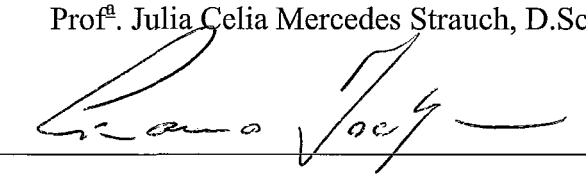
Aprovada por:



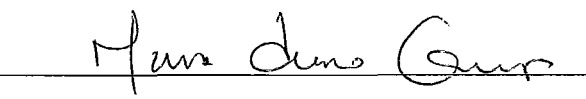
Prof. Jano Moreira de Souza, Ph.D.



Profª. Julia Celia Mercedes Strauch, D.Sc.



Prof. Cirano Iochpe, Ph.D.



Profª. Maria Luiza Machado Campos, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

AGOSTO DE 2004

BOTÊLHO, LÚCIO ROGÉRIO

Uma Arquitetura Extensível para Integração e Publicação de Dados Georreferenciados [Rio de Janeiro] 2004

XV, 172 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2004)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Integração de Bases de Dados 2. Dados Georreferenciados Heterogêneos e Distribuídos 3. Arquitetura para interoperabilidade entre SIG 4. Modelo de Dados Canônico

I. COPPE/UFRJ II. Título (série)

Aos meus pais, Lauro e Alcione

À minha irmã, Thayenne

À minha namorada, Lys

AGRADECIMENTOS

Ao Professor Jano Moreira de Souza, pela sua orientação, sua dedicação de tempo precioso, e incentivos, tanto na vida acadêmica como profissional.

À Professora Julia Celia Mercedes Strauch, pela sua co-orientação, disponibilidade mesmo em seus momentos de descanso, e ajuda constante neste trabalho.

Aos outros membros da banca deste trabalho, Prof. Cirano Iochpe e Profª. Maria Luiza Machado Campos por terem aceito participar da mesma cedendo um pouco do seu valioso tempo e conhecimento.

À Patrícia Leal e ao pessoal da secretaria do PESC, por terem me ajudado em vários tramites e pelo companheirismo.

Aos meus colegas de mestrado, pelas discussões produtivas, bons momentos e companheirismo.

Aos meus colegas da COPPETEC, principalmente Renato, Bruno Kinder, Yura, Ricardo e Leonardo, pela amizade, pelas brincadeiras e pelo apoio em diversas fases tanto do mestrado quanto do projeto.

À CAPES e ao CNPq, pelo suporte financeiro a este trabalho.

À meu primo Batista, que me recebeu em sua casa quando precisei vir morar no Rio, pela ajuda de sempre, e principalmente pela amizade.

Aos meus familiares, obrigado de diversas formas.

Aos meus pais, Lauro e Alcione, os quais tanto amo e me dedico incondicionalmente, mesmo distantes fisicamente, sempre me apoiando e ajudando em minhas dificuldades e participando da minha vida emocional.

À minha irmã Thayenne, minha jóia e amor da minha vida, que mesmo no seu silêncio contra sua vontade, me dá forças e bravura para suportar as dificuldades e tristezas, e perto do que vem passando aquelas são simplesmente nada.

À minha linda, Lys, a quem tanto amo, e que me faz feliz mesmo nos momentos mais atribulados desta passagem. Chegou na minha vida em um momento conturbado, e não tenho dedicado tanto tempo quanto gostaria, peço desculpas.

À Deus, que me ilumina e me guia pelos caminhos que devo passar. Tenho realizado tantos sonhos que por vezes nem acredito.

Agradeço também a todos aqueles que participaram desta fase da minha vida e que me ajudaram de uma maneira ou de outra. Não posso colocar seus nomes aqui por que não conseguiria enumerar todos e seria falta grave deixar de colocar nome de alguém.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UMA ARQUITETURA EXTENSÍVEL PARA INTEGRAÇÃO E PUBLICAÇÃO DE DADOS GEORREFERENCIADOS

Lúcio Rogério Botêlho

Agosto/2004

Orientadores: Jano Moreira de Souza
Julia Celia Mercedes Strauch

Programa: Engenharia de Sistemas e Computação

Este trabalho apresenta uma arquitetura de um sistema para integração e publicação de dados convencionais e georreferenciados na *Web*. O sistema chamado WISE (*Web data Integration SystEm*) utiliza como Modelo de Dados Canônico para representar dados convencionais e georreferenciados o XML Schema Semântico (SXMLS) desenvolvido neste trabalho. O SXMLS utiliza as tecnologias XML, GML e afins, bem como um conjunto de elementos informativos, para representar os esquemas de fontes de dados locais em um modelo comum que armazene informações descritivas e semânticas. Os SXMLSs são integrados pelo WISE utilizando estas informações descritivas e semânticas com o auxílio de ontologias do domínio da aplicação. O WISE possui um conjunto de componentes, distribuídos nas camadas Cliente, de Integração de Dados, de Tradutores, e de Acesso a Dados, para realizar a integração e publicação de dados, sendo que alguns deles podem ser estendidos para agregar novos formatos de dados de saída e também novos tipos de fontes de dados de entrada. Foi desenvolvido um protótipo do WISE com algumas funcionalidades, sendo mostrado um estudo de caso na área de hidrologia.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

AN EXTENSIBLE ARCHITECTURE FOR GEO-REFERENCED DATA INTEGRATION AND PUBLICATION

Lúcio Rogério Botêlho

August/2004

Advisors: Jano Moreira de Souza
Julia Celia Mercedes Strauch

Department: Systems and Computer Engineering

This work presents an architecture of a system which perform conventional and geo-referenced data integration and publication in the Web. This system called WISE (Web data Integration SystEm) uses as Canonic Data Model to represent conventional and geo-referenced data, XML Schema Semântico (SXMLS), developed in this work. The SXMLS uses XML, GML and related technologies, together with a set of informative elements, to represent local data sources schemas in a common model which store semantic information. The SXMLSs are integrated by WISE using these semantics information with domain application ontologies. The WISE has a set of components, distributed in Client, Data Integration, Wrappers, and Data Access layers, to perform data integration and publication, and some of them can be extended to aggregate new output formats and new data sources input types. In this work was developed a WISE prototype with some functionalities, being presented a case study in hydrology area.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTAGENS

Capítulo 1 – Introdução

1.1 Motivação

Organizações distribuídas geograficamente necessitam realizar operações globais sem centralização da informação ou perda da autonomia dos seus sistemas locais. Assim sendo, a tendência de compartilhamento de dados tem sido substituir os ambientes fechados, com soluções orientadas a produtos, por ambientes abertos caracterizados por diferentes plataformas, aplicativos e bases de dados interoperáveis em tempo real.

Esta descentralização da informação nos setores das organizações representou uma evolução na gerência de suas atividades, porém tornou-se evidente a necessidade de compartilhar informações entre os diversos setores, a fim de permitir uma visão global da gerência do negócio para a tomada de decisões corporativas entre outras atividades.

A descentralização das informações dentro de uma organização também induziu à proliferação de bancos de dados e outras fontes de dados que, geralmente, obedecem a diferentes conjuntos de requisitos para modelar objetos idênticos ou similares, possibilitando o surgimento de informações relacionadas, bem como, a presença de informações redundantes nas diferentes fontes de dados.

Com isto surgiu a necessidade da integração destas fontes de dados, que é mais conhecida como Integração de Dados, conforme pode ser encontrado na literatura (SOUZA, 1986; BATINI et al., 1986; BREITBART et al., 1986; SHETH et al., 1993; CHAWATHE et al., 1994; ABEL et al., 1994; BONJOUR & FALQUET, 1994; FRANKHAUSER et al., 1995; BARU et al., 1999).

Pesquisas têm levado ao desenvolvimento de estratégias para integração de dados heterogêneos e distribuídos. Estas visam capturar a semântica dos esquemas das fontes de dados, compreender suas diferenças, seus requisitos de consistência e suas restrições de integridade. Além disso, as estratégias visam aplicar técnicas desenvolvidas para resolver os conflitos semânticos e sintáticos decorrentes das diferenças entre os modelos de dados e do processo de modelagem que por ventura existirem de modo que seja possível integrá-los.

PINTO (2003) afirma que com o surgimento da *World Wide Web* (Web), a integração de dados mudou de uma arquitetura tradicional de múltiplas bases de dados para um novo *framework* capaz de manipular uma variedade de informações disponíveis em diversos formatos e estruturas. Este novo contexto da Internet gerou inovações que conduzem a novos problemas que se apresentam mais difíceis que os já conhecidos. Esta necessidade também existe no âmbito da gestão de instituições que manipulam dados georreferenciados. Um exemplo é citado por STRAUCH (1998), que mostra em seu trabalho a necessidade de compartilhamento de dados georreferenciados, os meios utilizados para a troca destes dados e os esforços dos órgãos competentes, como o Open GIS Consortium (OGC, 2003), para estabelecer uma especificação para a interoperabilidade dos dados.

1.2 Objetivo

Atualmente existem vários sistemas para publicação e integração tanto de dados convencionais como de dados georreferenciados (CHAWATHE et al., 1994; GUPTA et al., 1999; BARU et al., 1999; ZHANG et al., 2000; TANAKA et al., 2001; PINTO et al., 2001). Este trabalho propõe uma arquitetura para a construção de um sistema para integração e publicação de dados convencionais e georreferenciados na Web, chamado WISE (*Web data Integration SystEm*) (BOTÊLHO et al., 2003a; BOTÊLHO et al., 2003b), que visa a integração semântica de dados usando ontologias do domínio da aplicação (FONSECA et al., 2000a; FONSECA et al., 2000b) e um conjunto de heurísticas.

O WISE mantém um foco no tratamento de dados georreferenciados, porém o tratamento de dados convencionais, que é uma generalização daqueles também é contemplado. Desta forma o WISE pode trabalhar somente com dados convencionais, ou com dados georreferenciados, ou mesmo a combinação de ambos.

O WISE faz parte do projeto SPeCS (MEDEIROS et al., 2001) que tem por objetivo fornecer um ambiente de trabalho cooperativo comum, flexível e fácil de usar onde os membros do grupo podem estar espalhados geograficamente em diversos ambientes heterogêneos e mesmo assim serem capazes de interagir durante um processo de tomada de decisão. Trata-se de um *framework* para suportar a decisão espacial colaborativa. Neste projeto são implementadas ações de pesquisa em: Integração de Bases de Dados, Ferramentas de Trabalho Cooperativo, Ferramentas e Técnicas de

Workflow, Modelagem Matemática para apoio à decisão, Acesso e manipulação de informações via Internet.

O WISE surgiu para substituir a X-Arc que era, até então, o sistema publicador de dados georreferenciados do SPeCS. Cabe ressaltar que o WISE é desenvolvido como um superconjunto da X-Arc, e realiza a tarefa de integrar e publicar dados georreferenciados. Assim como a X-Arc o WISE é uma arquitetura baseada em mediadores.

A arquitetura do WISE é flexível, pois possui módulos que podem ser estendidos, para se adaptar a novos formatos de dados de saída, e para novos tipos de fontes de dados, visando proporcionar uma gama de formatos de dados, de modo a atingir um grande número de usuários que trabalham com dados convencionais ou georreferenciados. Trata-se de uma arquitetura diferente de outras arquiteturas de integração e publicação de dados georreferenciados existentes (SOROKINE & MERZLIAKOVA, 1998; ZASLAVSKY et al., 2000), que apresentam resultados de consultas usando arquivos de imagens raster ou applets, e geralmente não permitem manipulação (ou permitem uma manipulação restrita) apropriada às necessidades dos usuários alvo.

O WISE possui ferramentas para acessar fontes de dados heterogêneas e distribuídas de maneira uniforme, converter os esquemas das fontes de dados para um esquema comum à arquitetura, integrar estes esquemas, realizar e processar consultas, converter e publicar resultados na Web, entre outras.

Para que todas as fontes de dados heterogêneas e distribuídas sejam convertidas para um formato padrão, de modo a facilitar o processo de integração de dados, é utilizado como Modelo de Dados Canônico, o chamado XML Schema Semântico criado neste trabalho.

O XML Schema Semântico é desenvolvido utilizando a *Extensible Markup Language* (XML) (XML, 2004) e a *Geography Markup Language* (GML) (GML, 2003), que são, respectivamente, tecnologias atuais para troca de dados convencionais, e georreferenciados, e foram desenvolvidas com o intuito de realizar troca de dados entre diversos aplicativos distintos, entre outras funções.

O XML Schema Semântico utiliza recursos do esquema de dados XML Schema (XML SCHEMA, 2003) da XML, que serve para representar a estrutura (ou esquema)

dos documentos XML e GML, para representar as fontes de dados heterogêneas e distribuídas a serem integradas.

1.3 Organização do trabalho

Os estudos desenvolvidos neste trabalho foram organizados neste texto em cinco capítulos adicionais, como segue.

O Capítulo 2 apresenta uma revisão da literatura de integração de dados, dissertando sobre o Modelo de Dados Canônico, com os tipos de arquiteturas de integração de dados mais comuns, a utilização de integração de dados na Web, alguns exemplos de arquiteturas para integração de dados e, finalmente são descritos alguns conflitos que podem acontecer no processo de integração de dados convencionais e georreferenciados.

No Capítulo 3 são descritos alguns aspectos de informações georreferenciadas, conceitos de cartografia e geodésia e características de bases de dados georreferenciados.

A Arquitetura WISE é mostrada no Capítulo 4, apresentando o Modelo de Dados Canônico do WISE, XML Schema Semântico. É também apresentada a organização em camadas desta arquitetura, os papéis de usuários e funcionamento do WISE, e por último seus componentes.

O Capítulo 5 apresenta uma descrição das ferramentas e tecnologias utilizadas na implementação do WISE, bem como um estudo de caso de utilização do WISE na integração de fontes de dados da ENCE/IBGE e do Laboratório de Hidrologia da COPPE/UFRJ.

O Capítulo 6 apresenta conclusões, contribuições deste trabalho e as suas perspectivas futuras.

Capítulo 2 – Integração de Dados

A integração de dados pode ocorrer de várias formas, utilizando diversos artifícios, como alguns apresentados neste capítulo que visa revisar um pouco da literatura. A seção 2.1 trata da integração de dados, a seção 2.2 do Modelo de Dados Canônico utilizado para integrar fontes de dados heterogêneas. Os tipos de arquiteturas para integração de dados são vistas na seção 2.3 e a seção 2.4 apresenta assuntos relacionados à integração de dados na *Web*. Alguns exemplos de arquiteturas de integração de dados que estão relacionados a este trabalho são vistos na seção 2.5. Finalmente na seção 2.6 são discutidos alguns dos problemas ou conflitos decorrentes da integração de dados.

2.1 Panorama da Integração de Dados

Informações dos mais variados tipos encontram-se hoje distribuídas em diversos locais, utilizando plataforma de software e hardware diversificado, bem como estando em representações com formatos heterogêneos. Esta descentralização da informação nos setores das organizações representou uma evolução na gerência das atividades dos seus setores, porém tornou-se evidente a necessidade de compartilhar informações entre as diversas unidades, a fim de permitir uma visão global da gerência do negócio para a tomada de decisões corporativas. Além disso, as organizações, freqüentemente, possuem a necessidade de compartilhar informações com outras, para adquirirem dados pertinentes às suas atividades fim, para realizarem pesquisas e novas análises e descobertas.

A descentralização das informações dentro de uma organização também induziu à proliferação de bancos de dados e outras fontes de informação que, geralmente obedecem a diferentes conjuntos de requisitos para modelar objetos idênticos ou similares, possibilitando o surgimento de informações relacionadas, bem como a presença de informações redundantes nas diferentes fontes de informação.

Com isto surge a necessidade da integração de informações ou integração de dados, conforme pode ser encontrado na literatura (SOUZA, 1986; BATINI et al., 1986; BREITBART et al., 1986; SHETH et al., 1993; CHAWATHE et al., 1994; ABEL et al.,

1994; BONJOUR & FALQUET, 1994; FRANKHAUSER et al., 1995; BARU et al., 1999).

A integração de dados de fontes heterogêneas e distribuídas não é um processo simples e além de visar a integração sintática dos dados, também se preocupa com a sua semântica. Com isto, os diferentes modelos de dados utilizados pelas fontes de dados heterogêneas podem ser convertidos para um modelo de dados comum, também chamado Modelo de Dados Canônico (MDC).

2.2 Modelo de Dados Canônico

O processo de integração de fontes de dados pode compreender diversos passos, sendo um deles o mapeamento de modelos das fontes de dados heterogêneas e distribuídas para um modelo de dados comum.

O modelo de dados é o mecanismo da tecnologia de Banco de Dados para abstrair uma realidade e representá-la. Em geral, os esquemas das fontes de dados locais podem empregar modelos de dados heterogêneos. Todavia, no processo de integração de esquemas é importante que o responsável pela atividade de integração compreenda e raciocine sobre as semânticas dos esquemas das fontes de dados locais para encontrar as suas similaridades. Desta forma, para facilitar este processo se faz necessário adotar um modelo de dados único, denominado de Modelo de Dados Canônico (MACFARLANE et al., 1996; GARDELS, 1996; AGUIAR & MEDEIROS, 1996; STRAUCH et al., 1998).

Segundo SOUZA (1986) a adoção deste modelo no desenvolvimento de sistemas de integração de dados tem por objetivo oferecer uma compreensão normalizada do significado dos dados armazenados por cada esquema, facilitar a comparação dos esquemas no processo de integração e ser um mecanismo comum sobre o qual é realizada uma consulta.

O MDC consiste em um modelo de dados para o qual todos esquemas das fontes de dados componentes são transformados, de modo a minimizar as diferenças sintáticas conseqüentes do uso de diferentes modelos de dados. Esta transformação dos esquemas locais para o MDC não é uma simples tradução de um modelo para o outro. Isto inclui um processo de enriquecimento semântico para capturar a semântica implícita nas fontes de dados locais.

O MDC é usado como referência padrão para o qual todos os esquemas devem ser traduzidos e analisados no processo de integração. A escolha de um modelo de dados para expressar o modelo conceitual dos esquemas é um fator crítico no processo de integração de esquemas. Esta decisão influencia na qualidade da etapa de integração. Além disto, o conceito de sistema de integração de dados, no qual outras fontes de dados podem entrar no futuro, introduz a necessidade do modelo de dados a ser adotado como MDC assegurar que as novas fontes de dados tenham modelos de dados mais pobres ou iguais a ele.

Um modelo simples, ou semanticamente pobre, isto é, com poucos construtores de modelagem, é uma ferramenta fraca para descobrir as similaridades e heterogeneidades no processo de integração de dados. Apesar de um MDC com esta característica empregar operações de tradução e integração simples que envolvem poucas operações primitivas, ele tem a desvantagem de não expressar toda a semântica dos esquemas locais, apenas um subconjunto, limitando a representação do esquema local e as informações necessárias para a integração. Em geral, a sua utilização requer um processo de enriquecimento semântico dos esquemas, através de um processo de aquisição de conhecimento pelo qual todas as semânticas extraídas são explicitadas em uma representação formal para cada fonte de dados local.

Um modelo semanticamente rico é um modelo que oferece construtores para modelar a semântica capturando os aspectos estruturais e comportamentais dos esquemas locais, bem como qualquer semântica adicional obtida do processo de enriquecimento. Um modelo de dados semanticamente rico, como MDC, facilita a detecção de similaridades e heterogeneidades entre esquemas em um nível mais alto de abstração, contudo o número de heterogeneidades representadas é maior. A operação de integração de esquemas é facilitada com o uso dos construtores do modelo de dados e operações de integração.

As características que um modelo de dados deve conter para ser adotado como MDC têm sido bastante pesquisadas na literatura. Dentre estas características, destacam-se (VENTRONE & HEILER, 1991; OLIVEIRA & MAGALHÃES, 1993):

- Capacidade de expressar diferentes níveis de abstração para representar as diversas semânticas contidas em outros modelos, de forma simples e clara;
- Possibilidade de acrescentar informações relevantes, não contidas nos esquemas das fontes de dados, que possam facilitar a comparação dos

esquemas; como por exemplo, restrições de integridade implícitas ou mantidas pelas aplicações;

- Facilidades para representar interdependências e diferenças entre os esquemas e suportar mecanismos para reforçá-las;
- Habilidade para expressar a informação semântica comum a todos os usuários do ambiente integrado.

O MDC adotado pela arquitetura WISE é denominado XML Schema Semântico e será descrito no capítulo 4.

2.3 Tipos de Arquiteturas para Integração de Dados

Como a integração de dados vem sendo estudada há bastante tempo, as formas como estes dados são integrados foram variando neste período, dando origem a diversos tipos de arquiteturas de integração de dados.

Segundo ÖZSU & VALDURIEZ (1999) as arquiteturas para integração de informações diferem quanto a:

- Autonomia - refere-se à distribuição do controle, não dos dados. Ela indica o grau segundo o qual os SGBDs individuais podem operar independentemente. A *autonomia de projeto* refere-se à liberdade que um SGBD tem para utilizar um modelo de dados e as técnicas para gerenciamento de transações mais adequados às suas necessidades. A *autonomia de comunicação* refere-se à liberdade que um SGBD tem de decidir que tipo de informações disponibilizar para outros SGBDs ou para o software que controla suas execuções globais. A *autonomia de execução* refere-se à liberdade que um SGBD tem de decidir a ordem de execução das operações locais e externas;
- Distribuição – distribuição física dos dados em múltiplos lugares. ÖZSU & VALDURIEZ (1999) abstrai as diversas alternativas de distribuição em: distribuição cliente/servidor, que concentra as tarefas de gerenciamento nos servidores, enquanto os clientes fornecem os serviços para aplicações; e distribuição ponto-a-ponto (P2P), em que cada máquina tem funcionalidades de um SGBD completo e pode se comunicar com outras máquinas para realizar consultas e transações;

- Heterogeneidade - A heterogeneidade pode ocorrer em diferentes níveis, variando da heterogeneidade de *hardware* e diferenças nos protocolos de rede até variações nos gerentes de dados. As mais interessantes no enfoque de banco de dados são as heterogeneidades de modelos de dados, linguagens de consultas, e protocolos de gerenciamento de transações.

Nas seções abaixo, são apresentadas as principais arquiteturas propostas na literatura para solucionar o problema da integração de dados (WIEDERHOLD, 1992; ELMAGARMID et al., 1999).

2.3.1 Esquema Global

A arquitetura do Esquema Global foi a primeira a prover o compartilhamento de dados entre sistemas de bancos de dados heterogêneos e distribuídos. Esta arquitetura é baseada na integração total dos múltiplos esquemas dos bancos de dados para oferecer uma única visão integrada destes bancos (BATINI et al., 1986; STRAUCH, 1998).

A Figura 2.1 apresenta esta arquitetura que consiste dos seguintes componentes: a) bancos de dados locais; b) esquemas dos bancos de dados locais; c) esquema global resultante da integração de todos os esquemas locais; e d) aplicações de usuários, que consultam os bancos de dados locais através do esquema global. A principal vantagem desta arquitetura é que ela provê transparência no acesso a dados heterogêneos e distribuídos. Assim, múltiplos bancos de dados podem ser apresentados, logicamente, aos usuários como um grande banco de dados. Algumas das desvantagens desta arquitetura são: a) a complexidade na geração automática do esquema global, devido à dificuldade de identificar as correspondências entre os esquemas locais; b) a visão única e extensa oferecida pelo esquema global, representando a integração unificada de todos os bancos de dados, quando os usuários têm interesses em partes mais específicas do esquema global; e c) a dificuldade na manutenção do esquema global, visto que, para qualquer evolução dos esquemas locais, o esquema global tem que ser atualizado.

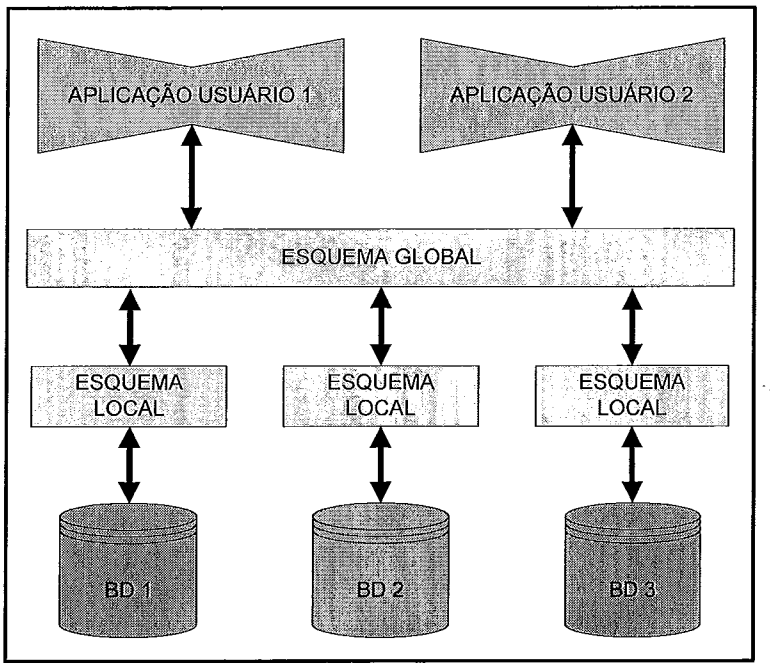


Figura 2.1 – Esquema Global

2.3.2 Sistema de Bancos de Dados Federados

Sistema de Bancos de Dados Federados é uma coleção de sistemas de bancos de dados cooperantes e autônomos que participam da federação para permitir o compartilhamento parcial e controlado de seus dados. A característica principal de uma federação é a cooperação entre sistemas independentes (SHETH & LARSON, 1990; ELMAGARMID et al., 1999; BUSSE et al., 1999). Esta arquitetura está apresentada na Figura 2.2 e consiste dos seguintes componentes: a) bancos de dados locais; b) esquemas dos bancos de dados locais; c) esquemas componentes, que correspondem à representação dos bancos de dados no modelo canônico; d) esquemas de exportação, que definem a parte dos esquemas componentes acessível para a camada da federação; e) esquema federado, que provê a visão integrada de todos os esquemas de exportação disponíveis; f) esquema externo, que representa um nível de abstração usado quando o esquema federado é muito complexo; e g) aplicações de usuários, que podem acessar os dados através do esquema federado ou do esquema externo.

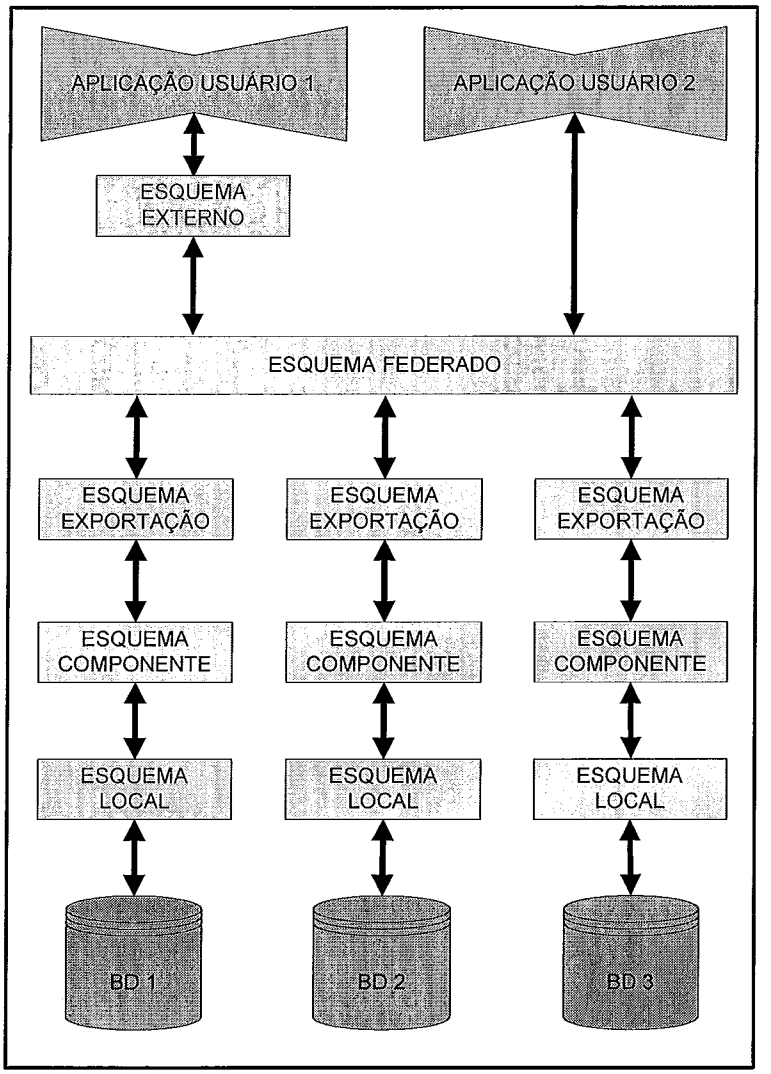


Figura 2.2 – Bancos de Dados Federados

A arquitetura do Sistema de Bancos de Dados Federado apresenta algumas vantagens similares àquelas apresentadas pela arquitetura do Esquema Global, tais como: a transparência de distribuição e heterogeneidade dos dados. Outra vantagem, é que o usuário pode definir um esquema de acordo com seus próprios requisitos (esquema externo), a partir do esquema federado.

Algumas desvantagens dessa arquitetura também estão presentes na arquitetura de Esquema Global, como por exemplo, a dificuldade para geração e manutenção do esquema federado.

2.3.3 Sistema de Bancos de Dados Múltiplos

Sistema de Bancos de Dados Múltiplos é uma coleção de múltiplos sistemas de bancos de dados locais. Nesta arquitetura não existe o conceito de esquema integrado que represente a unificação dos esquemas locais. Para permitir o acesso integrado aos bancos de dados é disponibilizada uma linguagem que oferece construtores para executar consultas envolvendo vários bancos de dados ao mesmo tempo (BREITBART, 1990; HURSON et al., 1994; RIBEIRO & OLIVEIRA, 1996; ELMAGARMID et al., 1999).

A Figura 2.3 apresenta esta arquitetura que consiste dos seguintes componentes: a) bancos de dados locais; b) esquemas locais; c) esquemas conceituais, representados em um mesmo modelo de dados, os quais definem a parte do esquema local acessível para o sistema de integração; d) esquema de dependências, que define as dependências e o conjunto de restrições de integridade globais envolvendo todos os bancos de dados; e) esquema externo, que representa uma visão externa, envolvendo dados de um ou mais esquemas conceituais; e f) aplicações de usuários, que consultam os dados distribuídos através do esquema externo.

Uma vantagem apresentada por esta arquitetura é que ela permite que os bancos de dados participantes do sistema tenham autonomia total. Por outro lado, como não existe a noção de um esquema integrado, a desvantagem desta arquitetura é que, o usuário deve conhecer a localização exata dos bancos de dados, os conceitos definidos nos esquemas locais, e o conteúdo de cada banco, para realizar uma consulta. Como conseqüência, o usuário é responsável por detectar e resolver os conflitos sintáticos e semânticos entre os esquemas conceituais.

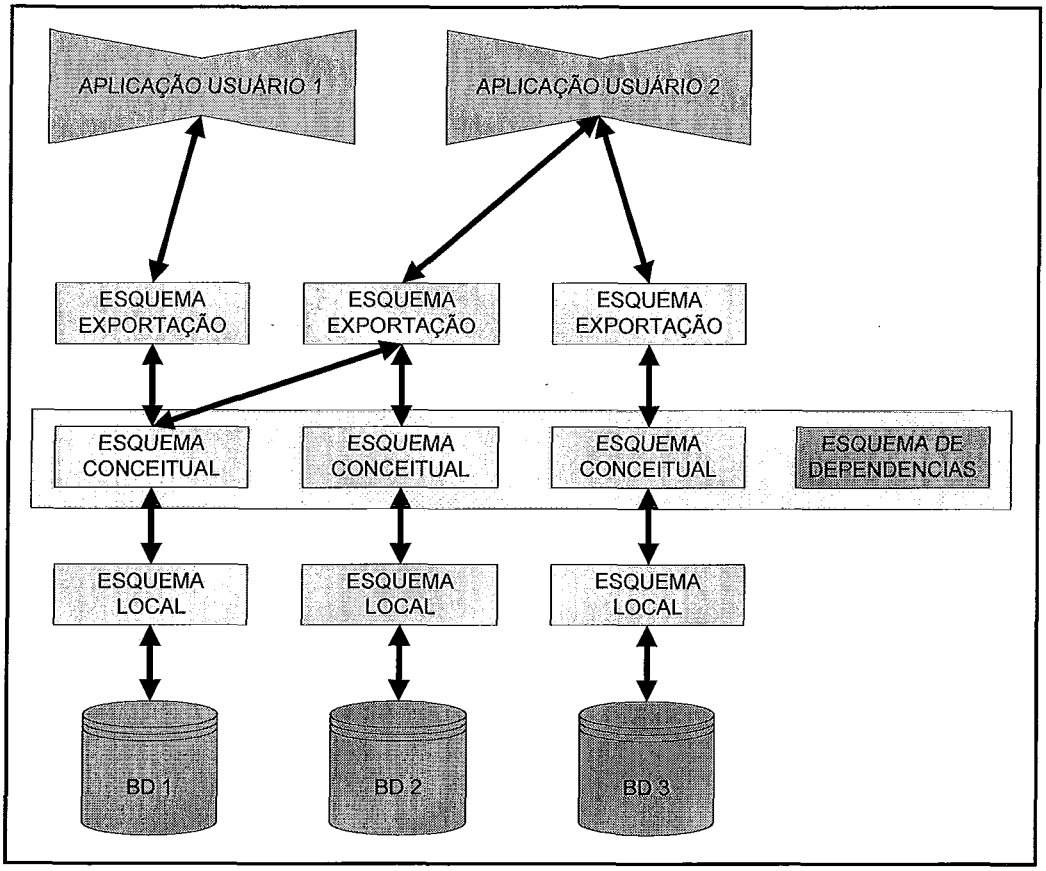


Figura 2.3 – Sistemas de Bancos de Dados Múltiplos

2.3.4 Mediadores

A arquitetura de mediadores provê acesso integrado e transparente a múltiplas fontes de informação através de mediadores. Mediadores são componentes de *software* que exploram o conhecimento representado em um conjunto ou subconjunto de dados para gerar informações para aplicações residentes em uma camada superior (GARCIA-MOLINA et al., 1997; WIEDERHOLD & GENESERTH, 1997; PINTO, 2003).

Na arquitetura de mediadores, o acesso aos dados distribuídos em múltiplas fontes de dados é efetuado através de consultas que são submetidas ao sistema através do mediador (VIDAL et al., 2001b). O mediador oferece uma visão integrada dessas fontes. Basicamente, existem dois enfoques para suportar visões integradas: virtual e materializado. No enfoque virtual, o sistema de integração utiliza um conjunto de correspondências entre o esquema da visão do mediador e os esquemas locais, para determinar como as consultas na visão do mediador serão respondidas a partir de consultas nas fontes locais. Os resultados dessas consultas são traduzidos, filtrados e integrados, e depois, o resultado final é retornado ao usuário. No enfoque materializado,

as informações relevantes são previamente extraídas das fontes de dados, e, posteriormente, traduzidas, filtradas, integradas e armazenadas em um repositório (também chamado de *Data Warehouse*), de maneira que as consultas possam ser avaliadas diretamente neste repositório centralizado, sem a necessidade de acessar as fontes de dados locais. Um ponto crítico do enfoque materializado é manter as visões materializadas consistentes com relação às atualizações nas fontes (manutenção da visão materializada).

A arquitetura de mediadores para o enfoque virtual é apresentada na Figura 2.4 e consiste de quatro componentes distribuídos em três níveis:

a) Nível dos dados - nível que representa as informações locais.
   - Fontes de Dados (FDs) - podem ser autônomas e heterogêneas, porque, na maioria das vezes, elas não foram projetadas com a intenção de compartilhamento. As Fontes de Dados podem ser bancos de dados, depósitos de objetos, bibliotecas digitais, páginas HTML e XML (XML, 2003; BOTÊLHO & SOUZA, 2003), entre outros.
   - Tradutores - convertem os dados das FDs para um modelo canônico e convertem consultas de aplicações em consultas específicas da fonte correspondente.

b) Nível de integração - nível que representa a integração das FDs que fazem parte do sistema de integração de informações.
   - Mediadores: "interfaces" através das quais os usuários consultam e atualizam múltiplas fontes de informação.

c) Nível dos usuários - nível que representa o acesso às FDs.
   - Aplicações de usuários: podem acessar as informações das fontes heterogêneas de maneira transparente e uniforme através de mediadores e tradutores.

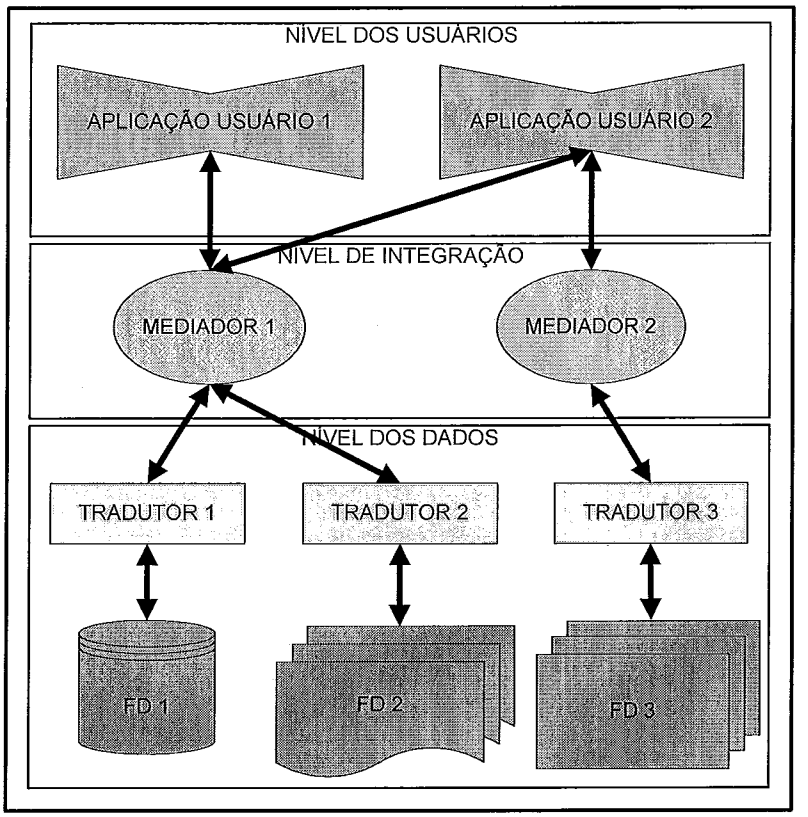


Figura 2.4 – Arquitetura de Mediadores

2.4 Integração de Dados na Web

Devido à maneira simples e universal para publicação e troca de informações na internet, a *World Wide Web* (Web) vem crescendo nos últimos anos tanto em número de usuários (individuais, empresas, governos, etc) quanto em aplicações (serviços de busca, comércio eletrônico, etc.), possibilitando desta forma que cada vez mais informações sejam disponibilizadas neste ambiente. Com isso, atualmente, um dos desafios das pesquisas em bancos de dados e Web é prover acesso integrado e transparente a essas informações (GOLDMAN et al., 1999; BARU et al., 1999; GARDARIN et al., 1999; ZASLAVSKY et al., 2000; CHRISTOPHIDES et al., 2000; ABITEBOUL et al., 2000; PINTO et al., 2001; VIDAL et al., 2001a).

Muitos dos problemas encontrados na construção dos sistemas de integração de informações na Web são similares aos problemas encontrados em sistemas de integração de bancos de dados convencionais. Porém, para integrar dados na Web, alguns problemas adicionais precisam ser solucionados:

- O número de fontes de dados pode ser muito grande, o que dificulta a construção de uma visão integrada destas fontes;

- Algumas fontes de dados são mais dinâmicas, assim a adição ou remoção destas fontes deve ser feita de maneira a minimizar o impacto na visão integrada;
- A maioria dos dados disponíveis na Web é considerado dado semi-estruturado. Este tipo de dado, apesar de ter uma estrutura, esta pode ser desconhecida, irregular, ou mesmo estar embutida no dado (ABITEBOUL, 1997).

Para realizar a integração de dados na Web, vários autores (GOLDMAN et al., 1999; CHRISTOPHIDES et al., 2000) estão utilizando XML (*eXtensible Markup Language*) (XML, 2003), que foi proposta pelo W3C (W3C, 2003) como um padrão para representação e troca de informações na *Web*, devido a sua flexibilidade em representar informações com estruturas heterogêneas (tanto dados estruturados como semi-estruturados) e a facilidade que se tem em converter qualquer dado nela.

Assim como XML pode ser usada na integração de dados convencionais, pode-se usar uma de suas extensões, GML (*Geography Markup Language*) (GML, 2003) para realizar a integração de dados georreferenciados.

Estas duas linguagens, XML e GML, são descritas brevemente nas próximas seções, pois serão utilizadas ao longo deste trabalho. Cabe ressaltar que uma discussão mais aprofundada destas linguagens está além do escopo desta dissertação, e para um entendimento maior deste assunto os interessados devem se referir às documentações de XML e GML.

2.4.1 XML

XML (*eXtensible Markup Language*) é uma metalinguagem, ou seja, uma linguagem para descrever outras linguagens, onde pode-se criar linguagens de marcação próprias. Cabe ressaltar que, assim como HTML, XML é um subconjunto da SGML (*Standard Generalized Markup Language*).

Ao contrário da HTML que tem um conjunto de marcações fixas e pré-definidas que são entendidas por todos os navegadores, mas que são limitados à exibição de dados, a XML é o resultado de uma evolução no sentido de definir um formato padrão para documentos que seja portátil e estruturado e que forneça informações tanto sobre conteúdo quanto contexto.

A dificuldade em estabelecer um formato alternativo, que tanto "leitores" quanto "publicadores" de dados entendam o mesmo conjunto de marcações sem limitar sua riqueza e flexibilidade, vem sendo resolvida com XML, pelo fato de ser autodescritiva. Dentre as principais características de XML, cabe destacar:

- Independência de conteúdo e apresentação. XML aborda apenas o conteúdo de um documento. A apresentação pode ser tratada por linguagens específicas para apresentação;
- Linguagem Extensível. XML permite que os usuários definam novas marcações de acordo com o domínio que está sendo modelado. Estas marcações servem para descrever o conteúdo de um documento;
- Validação. Um documento XML pode ser associado a um esquema XML, o qual define a estrutura do documento. Assim, aplicações podem validar seus dados de acordo com este esquema.

Os benefícios da XML incluem:

- Suporte para múltiplas visões de um mesmo conteúdo para diferentes usuários e tecnologias;
- Semântica agregada aos documentos através das marcações;
- Consultas baseadas na semântica fornecida pelos documentos;
- Uma plataforma para compartilhamento de informações através de uma sintaxe comum.

No apêndice B são descritos alguns componentes de XML e tecnologias afins que são utilizados nos capítulos 4 e 5.

2.4.2 GML

GML (*Geography Markup Language*) é uma gramática XML escrita em XML Schema (XML SCHEMA, 2003) para a modelagem, transporte, e armazenamento de informações geográficas. Os conceitos-chave usados pela GML para modelar o mundo são originados da Especificação Abstrata OGC (OGC, 2003).

A GML na sua versão 3 provê uma variedade de tipos de objetos para descrever geografia incluindo feições, sistemas de coordenadas de referência, geometria, topologia, tempo, unidades de medidas, entre outros, como pode ser visto na Figura 2.5. Dentre os tipos de objetos cabe destacar:

- *Feature* - é uma abstração de um fenômeno do mundo real que está associada com a localização relativa a Terra. Uma representação digital do mundo real pode ser tratada como um conjunto de feições. O estado de uma feição é definido por um conjunto de propriedades, onde cada propriedade pode ser representada como uma tripla {nome, tipo, valor}.

  O número de propriedades que uma *feature* pode ter, junto com seus nomes e tipos, é determinado pela sua definição de tipo. *Features* geográficas com geometria são aquelas com propriedades que podem ser valoradas à geometria. *Feature Collection* é uma coleção de feições que pode por si só se comportar como uma *feature*; como uma conseqüência, uma *feature collection* tem um tipo de *feature* e assim pode ter propriedades distintas das suas *features* conteúdo, em acréscimo às *features* que ela contém.

  Features geográficas em GML incluem *coverages* e *observations* como subtipos.
- *Coverage* - é um subtipo de *feature* que possui uma função de cobertura com um domínio espacial e um valor abrangendo tuplas de duas à n dimensões. Uma *coverage* pode representar uma *feature* ou um conjunto de *features* para modelar e tornar visíveis os relacionamentos espaciais entre os fenômenos da Terra, e a distribuição espacial destes fenômenos.
- *Observation* - modela o ato de observar, freqüentemente com uma câmera, uma pessoa ou alguma forma de instrumento (um ato de reconhecer e notar um fato ou ocorrência freqüentemente envolvendo medidas com instrumentos). Uma *observation* é considerada como uma *feature* GML com um tempo no qual a observação é realizada, e com um valor para a observação.

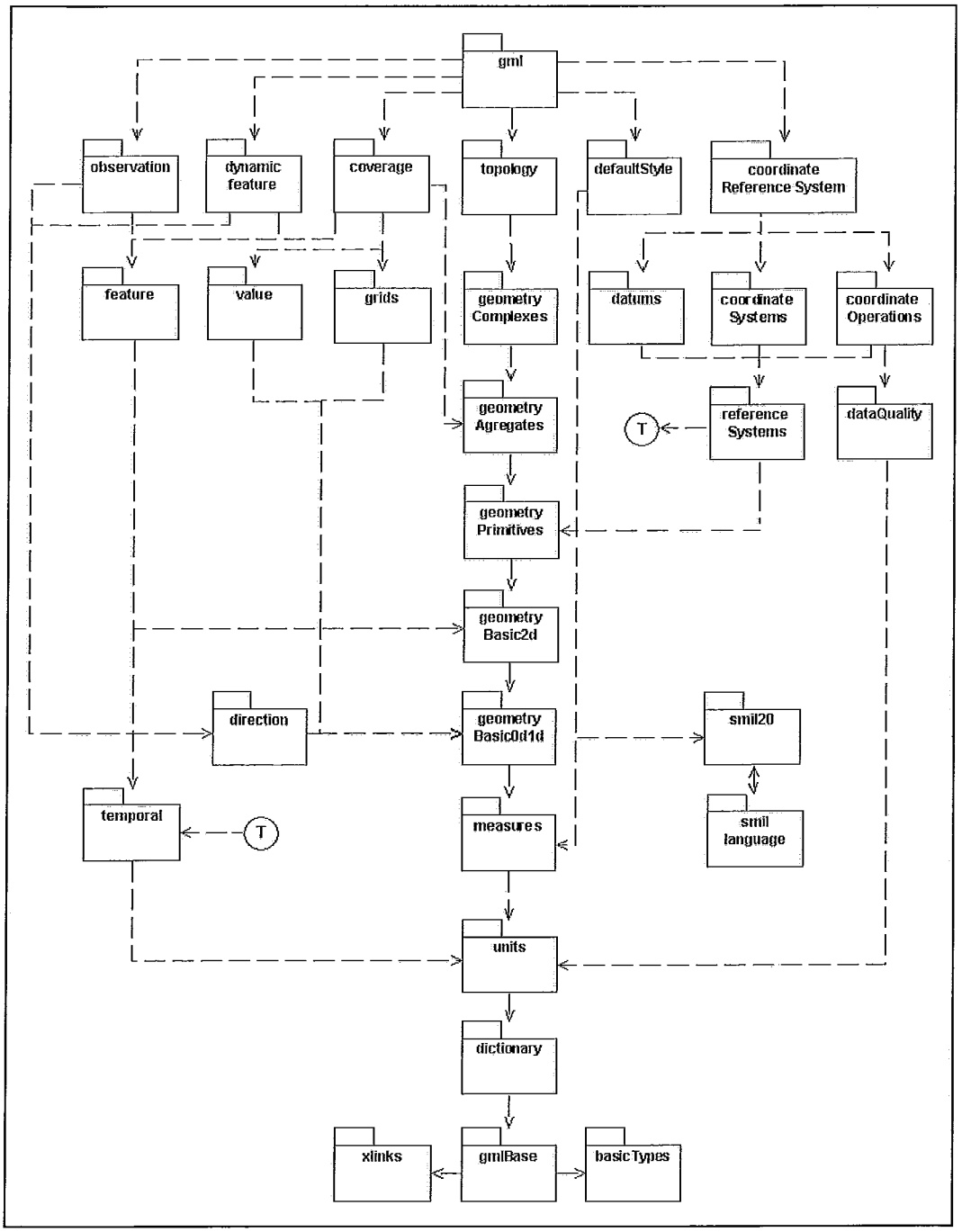


Figura 2.5 – XML Schemas de GML 3

- Sistema de coordenadas de referência – trata-se de um conjunto de sistemas de eixos coordenados que estão relacionados com a Terra através de um *datum* que define a forma, a posição e tamanho desta. As Geometrias em GML indicam o sistema de coordenadas de referência no qual suas medidas foram realizadas.

- Sistema temporal de referência - provê unidades padrão para medir tempo e descrever comprimento ou duração temporal.
- Dicionário de Unidades de Medidas (UOM, de *Units of Measure*) - provê definições de medidas numéricas de quantidades físicas, tais como comprimento, temperatura, e pressão, e de conversões entre UOMs.

Os conceitos apresentados acima estão representados em XML Schemas da GML. A Figura 2.5 mostra um diagrama UML com pacotes representando XML Schemas que definem GML 3.

2.5 Arquiteturas para Integração e Publicação de Dados

Na literatura são encontradas diversas arquiteturas para integração e publicação de dados, algumas para dados convencionais e outras para dados georreferenciados. A seguir são apresentadas algumas dessas arquiteturas usando mediadores, de forma a mostrar alguns trabalhos relacionados ao trabalho atual, e na seção 2.5.7 estas arquiteturas serão comparadas. A comparação destas arquiteturas com a arquitetura proposta neste trabalho será realizada no capítulo que a descreve, ou seja, no capítulo 4.

2.5.1 Garlic

O IBM *Almaden Research Center* desenvolveu o Garlic que é um sistema baseado no conceito de mediadores, onde é provida uma visão integrada de diversas fontes de dados legadas, sem alterar onde e como o dado está armazenado. Encapsula o modelo dos dados legados como objetos, os quais participam no planejamento da consulta. O Garlic provê interfaces padrão para a chamada de métodos e para a execução da consulta. Trata o desafio da diversidade de padrões considerando como a informação das fontes de dados é descrita e acessada (CAREY & ET AL., 1997; ROTH & SCHWARZ, 1997).

O MDC utilizado pelo sistema é o orientado a objetos que juntamente com a interface de programação são baseados no padrão estabelecido pelo *Object Database Management Group* (ODMG). O esquema unificado do Garlic é descrito por metadados globais mantidos pela arquitetura.

Devido a abordagem orientada a objetos, os métodos podem ser associados a dados. Esta capacidade é explorada pelo Garlic provendo uma maneira natural e

conveniente para modelar em fontes de dados não convencionais a busca e a manipulação de dados.

No planejamento da consulta o tradutor e o mediador negociam dinamicamente a função que o tradutor irá desempenhar na execução da consulta, pois é o único que realmente conhece as capacidades de pesquisa e acesso do repositório que ele encapsula. O processador de consultas desenvolve planos de execução para decompor, eficientemente, as consultas que envolvem múltiplos repositórios em subconsultas que os repositórios possam manipular. O executor da consulta controla a execução dos planos de consulta, montando os resultados dos repositórios e executando qualquer processamento adicional necessário para alcançar o resultado da consulta (CAREY & ET AL., 1997; ROTH & SCHWARZ, 1997).

Para cada repositório existe um tradutor. Além disso, o Garlic provê seu próprio repositório para objetos complexos, o qual pode ser criado pelos usuários para associar objetos de outros repositórios. Este repositório pode ser visto como um repositório de visões da arquitetura que interliga os dados dos repositórios existentes.

A principal característica do Garlic é considerar no seu plano de execução a existência de repositórios de dados heterogêneos, com poder de consulta limitado ou mesmo inexistente e suprir essa limitação através do seu processador de consulta.

2.5.2 TSIMMIS

O projeto *The Stanford-IBM Manager of Multiple Information Sources* (TSIMMIS) é uma cooperação entre o IBM *Almaden Research Center* e a Universidade de Stanford que tem como objetivo fornecer ferramentas para a integração de fontes de dados heterogêneas estruturadas ou semi-estruturadas e verificação da consistência dos dados obtidos (CHAWATHE et al., 1994). No TSIMMIS, os mediadores são construídos sobre um determinado conjunto de fontes de dados, utilizando os tradutores que exportam objetos do MDC do TSIMMIS, denominado *Object Exchange Model* (OEM). Este MDC é um modelo de objetos aninhados, autodescritivo e simples, semelhante a uma estrutura de grafo direcionado com arestas rotuladas. No modelo OEM, todas as entidades são objetos que podem ser atômicos ou complexos. Estes objetos são nós do grafo, no qual os objetos complexos têm arestas rotuladas com o tipo de relacionamento para seus subobjetos e os objetos atômicos contêm valores dentre os tipos atômicos (GOLDMAN et al., 1999).

Uma fonte de dados é exportada como um conjunto de objetos OEM. Desta forma, os mediadores provêem visões OEM integradas dos dados encapsulados. Os mediadores são especificados através da linguagem *Mediator Specification Language* (MSL) que pode ser considerada uma linguagem para definição de visões, trata-se de uma linguagem orientada a objetos e baseada em lógica aplicada ao modelo OEM. A descrição do mediador é feita fornecendo regras lógicas que definem objetos OEM, os quais o mediador disponibiliza em visões.

Os tradutores de cada fonte de dados são especificados com o auxílio da linguagem *Wrapper Specification Language* (WSL), que é uma extensão da MSL para permitir a descrição do conteúdo das fontes e das capacidades de consulta. O objetivo é representar através de modelos (*templates*) as capacidades das fontes de dados.

Cada modelo é associado a uma ação que gera os comandos para a fonte encapsulada.

O tradutor do TSIMMIS é mais pesado que os tradutores de outras arquiteturas de mediadores pois muitas das tarefas geralmente desempenhadas pelo mediador foram repassadas para o tradutor, como por exemplo a decomposição das consultas e a compensação pela ausência de capacidades de consulta.

As especificações do poder de consulta são expressas na linguagem *Query Description and Translation Language* (QDTL), que é uma gramática livre de contexto para a geração de consultas.

Embora a utilização de especificações declarativas compactas para expressar o poder de consulta dos repositórios seja atraente, existem algumas desvantagens e problemas nessa abordagem. O primeiro e principal problema é a definição de uma linguagem para descrever todas as capacidades de um repositório, uma vez que é difícil capturar as restrições únicas associadas com cada repositório. Outro problema é o fato de que os modelos não são capazes de representar todas as possíveis consultas de serem executadas sobre o esquema da fonte de dados (BUSSE et al., 1999; KALINICHENKO, 2001).

A arquitetura do TSIMMIS possui ainda um componente que é responsável pelo gerenciamento das restrições sobre os dados integrados das diversas fontes de dados. Este componente fornece uma validação de restrições sobre os dados, que é menor se comparada com as restrições que um SGBD centralizado provê.

Uma desvantagem da abordagem de integração seguida pelo TSIMMIS é a necessidade de intervenção humana durante o processo, podendo em alguns casos, ocorrer da integração ser realizada manualmente pelo usuário, resultando possivelmente em erros subjetivos e de operação. Embora as dificuldades semânticas existentes nos repositórios geralmente não permitam a automatização total do processo de integração, a indispensável participação humana no processo do TSIMMIS é um fator crítico para o sucesso da integração.

O TSIMMIS não fornece um esquema do mediador, mas propaga todos os esquemas dos tradutores para o usuário, ficando sob responsabilidade do mesmo a decisão de que dado consultar e para onde enviar a consulta.

2.5.3 MIX

O projeto *Mediation of Information using XML* (MIX) é uma colaboração entre o *UCSD Database Laboratory* e o grupo *Data-Intensive Computing Environments* (DICE). O objetivo do projeto é estudar, desenvolver, aplicar e avaliar sistemas de consulta sobre fontes de dados heterogêneas que utilizam XML (BARU et al., 1999). MIX baseia-se na arquitetura de mediadores e emprega XML como seu MDC para representar de forma flexível e uniforme os dados existentes nos repositórios. A abordagem do MIX não exige a conversão dos repositórios de dados convencionais para XML, entretanto pressupõe que uma visão lógica das fontes de dados (tais como bancos de dados, coleção de páginas html, ou até mesmo um sistema de dados legados) seja fornecida em XML.

Os repositórios são consultados utilizando uma linguagem de consulta XML própria, denominada XMAS (LUDASCHER et al., 1998). XMAS é uma linguagem funcional, que é influenciada pela OQL. No MIX, o resultado de qualquer consulta é um documento XML. O mediador do MIX recebe uma consulta, a decompõe em fragmentos (subconsultas) de acordo com as capacidades de consulta dos repositórios e envia os fragmentos para os repositórios apropriados. À medida que os resultados das subconsultas são retornados pelos repositórios para o mediador, ele integra os fragmentos em um único resultado e então retorna para o usuário.

Os tradutores são responsáveis por traduzir as consultas XMAS para consultas ou comandos que os repositórios encapsulados sejam capazes de compreender. Além disso, eles convertem os resultados retornados pelos repositórios para o formato XML.

Cada repositório exporta o modelo de dados que possui na forma de um DTD XML, que é utilizado como o esquema dos dados pelos componentes da arquitetura do mediador. Os dados são exportados pelos tradutores como documentos XML que atendem ao DTD especificado.

Ao contrário de outras abordagens, que baseiam-se em modelos semi-estruturados sem esquemas, o MIX utiliza XML em seu MDC e explora a estrutura da informação fornecida pelos DTDs que descrevem os repositórios. Semelhantemente ao TSIMMIS, a avaliação da consulta é virtual.

2.5.4 Extensão do MIX

A arquitetura tradutor-mediador do MIX foi estendida (GUPTA et al., 1999; ZASLAVSKY et al., 2000) para realizar integração de dados espaciais, com novos tradutores e mediadores criados para suportar consultas espaciais.

Se o mediador da aplicação detectar que em uma avaliação a consulta requer acesso a fontes geográficas, ele delega um fragmento de consulta particular para o mediador espacial. Usando os fragmentos de consulta fornecidos pelo mediador da aplicação, e informações sobre as capacidades da fonte de dados, o mediador espacial determina um plano de avaliação de consulta ótimo. Este plano é formulado em uma linguagem comum entendida pelos tradutores de fontes de dados georreferenciados, a qual é traduzida para a linguagem de uma fonte em particular, iniciando o processo de recuperação de dados. De acordo com o plano de execução de consulta, respostas aos fragmentos das consultas são trocados entre fontes de dados, de modo que um mapa virtual é montado e enviado para o usuário da aplicação.

Os resultados das consultas, retornados como mapas e dados tabulares do mediador para a interface do usuário, usando um cliente Web, são apresentados utilizando uma linguagem de renderização de vetor 2D, chamada VML (Vector Markup Language) (VML, 1998) e um visualizador de VML de nome AXIOMap (Application of XML for Interactive Online Mapping) (AXIOMAP, 2000), onde o usuário pode interagir com os mapas.

O problema desta abordagem é que utiliza, como formato de troca de informações a linguagem XML, e não um outro formato para troca de informações georreferenciadas como GML, por exemplo. Um outro problema é na visualização que

utiliza uma interface com poucos recursos de edição e caso permita que o conteúdo seja armazenado pelo usuário, este não está em um formato georreferenciado que possa ser utilizado facilmente como ESRI Shape (SHAPE FILE, 2003) ou GML, que estão bem difundidos.

2.5.5 Arquitetura Baseada em GML da Universidade de Oklahoma

Esta arquitetura (ZHANG et al., 2000) está em desenvolvimento na Universidade de Oklahoma, e pretende ser um mecanismo de busca sobre dados georreferenciados na Internet quando estiver totalmente pronta. Ela realiza a integração de dados, originados de páginas html, bancos de dados relacionais e orientados a objeto, através da utilização de mediadores, e disponibiliza estes dados para os usuários através de uma applet Java que recebe informações no formato GML e as converte para um modo gráfico.

O sistema adota uma arquitetura de três camadas: a primeira camada é composta de Servidores Distribuídos. Um Servidor Distribuído tem três componentes: um agente de software, um banco de dados local e um tradutor (wrapper). O agente de software percorre todas páginas *web* que estão associadas a esta arquitetura, e retorna todas as informações georreferenciadas e suas informações textuais e então armazena no seu banco de dados local. O tradutor extrai os dados relevantes do banco de dados local e o transforma no formato GML quando uma requisição válida é recebida e os envia para o Mediador (na segunda camada) que os armazena no seu banco de dados de replica.

A segunda camada é composta do Mediador. Quando o Mediador recebe uma requisição de um Cliente, ele primeiro determina qual servidor, ou servidores, estão envolvidos e decompõe a consulta em subconsultas. A seguir ele encontra os Servidores Distribuídos correspondentes nos seus metadados. Se o tempo da última atualização está dentro da tolerância, o Mediador consulta os dados duplicados daquele Servidor Distribuído no seu próprio banco de dados de replica (abordagem de visão materializada); de outro modo ele atualiza a replica antes de realizar a consulta. O Mediador finalmente manda os resultados integrados da consulta no formato GML de volta para o Cliente que requisitou.

A terceira camada é o lado Cliente. Ele suporta visualização de dados geográficos baseados em consultas espaciais. O lado Cliente se comunica com o Mediador em tempo real e retorna documentos GML do Mediador. Além das

funcionalidades de visualização, a informação textual associada com as localizações geográficas são também mostradas no navegador (browser) do Cliente, e hiperlinks podem levar o usuário para as páginas correspondentes. Note que é usado GML como o protocolo de comunicação comum entre os Servidores Distribuídos e o Mediador, bem como entre o Mediador e o Cliente.

As applets neste caso são para visualização, e somente operações básicas como Zoom, e informações sobre posicionamento do cursor são mostradas, não possuindo um mecanismo para realizar modificações nestas informações, e nem mesmo salvá-las.

2.5.6 X-ARC

A X-ARC (PINTO et al., 2001; PINTO, 2003) foi desenvolvida na linha de Banco de Dados do Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ, sendo projetada para acessar dados estruturados (Bancos de Dados Relacionais e Orientados a Objetos, etc), semi-estruturados (documentos XML, arquivos texto, planilhas eletrônicas, etc) e espaciais (dados armazenados em Sistemas de Informações Geográficas). Bastando para isso que cada repositório de dados possua um tradutor que realize o mapeamento do modelo de dados do repositório para o modelo de dados comum do mediador, já que é baseada na arquitetura de mediadores. A X-ARC não realiza a integração de dados propriamente dita, porém ela agrupa dados de diversas fontes de dados e os publica através da Web.

A arquitetura X-ARC encapsula dois sistemas para a execução de seus serviços e apoio à publicação dos dados: o Le Select (XHUMARI et al., 1999), mediador do INRIA que manipula dados estruturados e semi-estruturados; e o ArcIMS (ARCIMS, 2001) que manipula dados georreferenciados. A Figura 2.6 mostra os componentes que participam da arquitetura X-ARC. Nesta figura, é possível observar o mediador que acessa os repositórios de dados disponibilizados pelos sistemas de publicação de dados espaciais (ArcIMS) e não-espaciais (Le Select). O X-SELECT e o X-MAP são os componentes tradutores da arquitetura que encapsulam o acesso aos sistemas Le Select e ArcIMS, respectivamente. A arquitetura X-ARC visando aproveitar o esforço e pesquisa envolvidos na área de banco de dados, utiliza o Le Select para prover os serviços de acesso a dados não-espaciais. Da mesma forma, o ArcIMS é utilizado para prover acesso aos dados espaciais disponíveis através de mapas digitais. A arquitetura

fornece acesso para dados armazenados nos formatos ESRI shape, texto, planilha eletrônica Microsoft Excel (MICROSOFT EXCEL, 2004) e banco de dados relacionais.

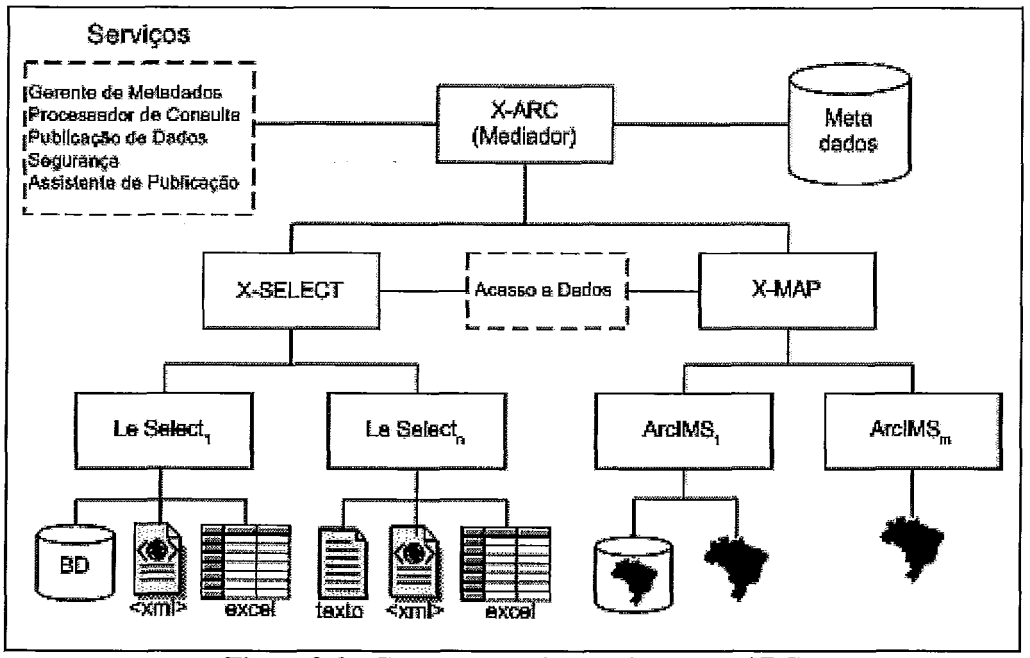


Figura 2.6 – Componentes da arquitetura X-ARC

Semelhante a outras arquiteturas que utilizam mediadores, a X-ARC é composta por um componente global (mediador) e por componentes locais (tradutores). O mediador acessa a informação compartilhada pelos componentes locais, gerenciando o acesso às fontes de dados locais e utilizando os tradutores que armazenam o mapeamento das fontes locais para o MDC do mediador.

A X-ARC utiliza o modelo relacional como seu MDC e um subconjunto da linguagem SQL como sua linguagem de consulta padrão. O modelo relacional foi adotado devido à facilidade de conversão dos dados ambientais representados em outros modelos para o relacional e ao fato do modelo relacional ser comumente utilizado pela indústria, além da representação tabular ser comum aos usuários da área ambiental que estão acostumados com a manipulação de dados neste formato.

A publicação dos resultados produzidos pela X-ARC é realizada através de *sites* publicados, onde encapsula-se os dados publicados pelos sistemas intermediários ArcIMS e Le Select. De modo que o ArcIMS disponibiliza os dados espaciais na Web como figuras raster que são geradas pelo sistema de acordo com a demanda específica, e o Le Select disponibiliza seus dados e serviços.

2.5.7 Comparação entre as Arquiteturas para Integração ou Publicação de Dados

As arquiteturas apresentadas acima são comparadas na Tabela 2.1, conforme o tipo de arquitetura, MDC, ocorrência de atualização dos dados, tipos de dados utilizados, tipos de visão e esquema global.

Esta tabela apresenta somente arquiteturas de mediadores pois o WISE foi desenvolvido utilizando este tipo de arquitetura.

Tabela 2.1 – Comparação entre as arquiteturas apresentadas

Características	Garlic	TSIMMIS	MIX	Extensão MIX	Arquitetura Universidade Oklahoma	X-ARC
Tipo de Arquitetura	Mediador	Mediador	Mediador	Mediador	Mediador	Mediador
MDC	ODMG	OEM	XML	XML	GML	Relacional
Atualização dos Dados	Não	Não	Não	Não	Não	Não
Tipos de dados	Estrut.	Estrut., Semi-Estrut.	Estrut., Semi-Estrut.	Estrut., Semi-Estrut.	Estrut., Semi-Estrut.	Estrut., Semi-Estrut.
Tipo de visão	Virtual	Virtual	Virtual	Virtual	Materializada	Virtual
Esquema Global	Sim	Não	Não	Não	Sim	Não

O WISE foi desenvolvido utilizando a idéia da arquitetura de mediadores por fornecer uma maneira mais natural de realizar o processamento de consultas de forma paralela através dos seus tradutores e fontes de dados que podem estar distribuídos geograficamente.

O MDC do WISE foi construído utilizando a tecnologia GML e XML, pois são padrões atuais para troca de dados georreferenciados e convencionais, respectivamente, e também por possuírem a capacidade de representar tipos de dados estruturados e semi-estruturados.

Os tipos de dados escolhidos para serem disponibilizados pelo WISE são os estruturados e os semi-estruturados, pois existem uma enorme quantidade destes dois tipos de dados sendo disponibilizados na Internet atualmente, e sendo necessária a sua integração.

O Tipo de Visão escolhida foi o Virtual pois foi idealizado para evitar a replicação de dados das fontes de dados e por sua vez os problemas de atualização dos mesmos.

Não foi utilizado o Esquema Global no WISE, pois este tipo de esquema apresenta todas as fontes de dados em um único esquema. Sendo assim, utilizado diversos esquemas para representar as fontes de dados, e esquemas para representar a integração de outros esquemas, possibilitando assim um particionamento de esquemas e facilitando a leitura dos mesmos pelos seus usuários.

2.6 Conflitos de Integração de Dados Convencionais e Georreferenciados

A tarefa de integrar dados de fontes heterogêneas e distribuídas é por natureza uma tarefa árdua, já que existem, nas mais diversas fontes de dados, diferentes modelos de dados, sintaxe e semântica, convenções, entre outros.

Os esquemas destas fontes de dados apresentam diferenças de representação do mesmo conceito do mundo real. Estas diferenças, denominadas de conflitos entre esquemas ou heterogeneidades semânticas, são decorrentes: a) dos projetistas modelarem a mesma perspectiva do mundo real de forma diferente; b) dos projetistas utilizarem diferentes modelos de dados ou diferentes construtores para modelar o mesmo aspecto da realidade; e c) das diferenças na especificação dos projetos das bases de dados, considerando as denominações empregadas, a especificação dos tipos e as restrições de integridade empregadas (CHATTERJEE & SEGEV, 1991; CASTELLANOS et al., 1994; KASHYAP & SHETH, 1996). Para integrar os esquemas, estas heterogeneidades precisam ser compreendidas em profundidade, a fim de detectá-las e solucioná-las.

A integração de fontes de dados georreferenciados herda todos os conflitos semânticos e estruturais das bases de dados convencionais, heterogêneas e distribuídas, e a estes são adicionados os conflitos decorrentes da componente gráfica dos dados georreferenciados. Estes conflitos tratam de aspectos da modelagem referentes à representação da semântica, as afinidades entre as aplicações e os conceitos de espaço associados a cada comunidade de informação.

Visando facilitar a identificação destas heterogeneidades, STRAUCH (1998) divide os conflitos de integração de dados convencionais e georreferenciados em quatro grandes grupos, apresentados a seguir:

2.6.1 Grupo I: Conflitos de Definição do Contexto Espacial

Uma base de dados ao ser implementada adota parâmetros cartográficos que fornecem suporte para a representação da base de dados sobre uma cartografia adequada a sua finalidade. Dentre estes parâmetros se encontram os *Data* vertical e horizontal, o sistema de coordenadas, o sistema de projeção e a escala.

Estes parâmetros são selecionados em função das necessidades de projeto, do custo da aquisição dos dados, da qualidade pretendida para a base de dados e da disponibilidade do material a ser digitalizado. Eles estão implícitos em cada esquema (metadados) de base de dados geográficos.

2.6.2 Grupo II: Conflitos Semânticos

Os conflitos semânticos ocorrem quando os conceitos, sobre uma realidade comum, são percebidos de formas diferentes, levando as bases de dados a apresentarem divergências semânticas na definição dos elementos dos esquemas. Este tipo de conflito se propaga através dos modelos de dados, da organização, das aplicações, dos procedimentos e restrições de integridade.

Estes conflitos cobrem um amplo espectro, contemplando as denominações empregadas, a abstração utilizada para modelar, as propriedades que descrevem a representação dos objetos geográficos e os conflitos geométricos destes, a saber.

- Conflitos de denominação - são devidos aos esquemas incorporarem denominações para os objetos representados. Podendo ser sinônimos ou homônimos. Os sinônimos ocorrem quando entidades ou propriedades semanticamente idênticas são denominadas diferentemente; e os homônimos ocorrem quando entidades e propriedades diferentes semanticamente compartilham a mesma denominação;
- Conflitos de abstração - são devidos às conceituações a respeito da realidade serem representadas em níveis de abstração diferente, por exemplo, um elemento de algum tipo (classe de objeto, atributo, relacionamento) corresponde a outro tipo em outro esquema;
- Conflitos de propriedades - ocorrem quando as propriedades que descrevem os objetos não estão completas, ou estão implícitas em outras propriedades;

2.6.3 Grupo III: Conflitos Estruturais

Ocorrem quando as conceitualizações, apesar de serem as mesmas e adotarem o mesmo modelo de dados, possuem diferentes especificações, restrições e domínios, fazendo com que a realidade comum seja representada de forma diferente nas bases de dados a serem integradas. Estes conflitos são classificados em:

- Conflitos de tipo - ocorrem quando as propriedades semanticamente equivalentes de dois esquemas apresentam definições de tipo incompatíveis;
- Conflitos de formatos - ocorrem quando a mesma propriedade emprega formatos diferentes nas bases de dados a serem integradas;
- Conflitos de unidades - ocorrem quando as unidades das propriedades são diferentes para a mesma conceitualização, em virtude das necessidades operacionais dos usuários;
- Conflitos de domínios - ocorrem quando os objetos do mundo real são percebidos no mesmo nível de abstração e classificados de acordo com o mesmo critério, mas a propriedade temática assume diferentes conjuntos de valores de domínio. Isto decorre do fato de que os valores de algumas propriedades não serem intrínsecos aos objetos, mas dependerem da proposta de aplicação;
- Conflitos de restrição - ocorrem quando alguma propriedade possui uma restrição lógica aos valores que pode assumir em um esquema;
- Conflitos de códigos - os códigos são usados por várias razões, tal como facilitar o acesso à informação e otimizar espaço. Eles geralmente não são uniformes entre as bases, uma vez que eles atendem às necessidades específicas dos usuários. Assim o uso de código introduz diferença de valores;
- Conflitos entre chaves - ocorrem quando as chaves primárias nos diversos esquemas são diferentes, ou diferentes chaves são usadas para o mesmo conceito;

2.6.4 Grupo IV: Conflitos entre Valores

Estes conflitos ocorrem no nível extensional das propriedades. Eles podem ser:

- Conflitos de valores *default* - os SIGs e aplicações podem proporcionar diferentes valores *default* para propriedades semanticamente iguais;
- Conflitos de valores - alguns SIGs oferecem facilidades para automaticamente calcular valores geométricos, os quais podem assumir valores diferentes decorrentes da qualidade dos dados espaciais, da precisão adotada para as coordenadas ou mesmo do algoritmo usado pelo SIG;
- Conflitos de atualização - acontecem quando as propriedades de diferentes bases de dados são atualizadas em diferentes épocas;
- Conflitos de erros de registros - podem ser erros tipográficos ou variações nos processos de medição, ou erros provenientes da aquisição de dados nos processos de digitação/digitalização.

Capítulo 3 – Aspectos de Informações Georreferenciadas

Este capítulo apresenta alguns conceitos básicos sobre informações georreferenciadas, tendo início com uma explanação sobre os Sistemas de Informação Geográfica, passando por uma discussão sobre informações georreferenciadas, e finalmente são apresentados conceitos cartográficos e geodésicos que são utilizados nos próximos capítulos.

3.1 Sistemas de Informação Geográfica

Sistemas de Informação Geográfica (SIGs) são sistemas automatizados usados para armazenar, analisar e manipular dados geográficos, ou seja, dados que representam objetos e fenômenos em que a localização geográfica é uma característica inerente à informação e indispensável para analisá-la (ARONOFF, 1989; BULL, 1994).

SIGs comportam diferentes tipos de dados e aplicações, em várias áreas do conhecimento. Exemplos são controle cadastral, gerenciamento de serviços de utilidade pública, demografia, cartografia, administração de recursos naturais, monitoramento costeiro, controle de epidemias, planejamento urbano. A utilização de SIGs facilita a integração de dados coletados de fontes heterogêneas, de forma transparente ao usuário final.

Numa visão geral, pode-se considerar que um SIG tem os seguintes componentes: a) interface com usuário; b) entrada e integração de dados; c) funções de processamento; d) visualização e plotagem; e e) armazenamento e recuperação de dados. Cada sistema, em função de seus objetivos e necessidades, implementa estes componentes de forma distinta, mas todos estão usualmente presentes num SIG.

3.2 Informações Georreferenciadas

Os SIGs armazenam dados de estudos relacionados a um espaço geográfico. Estes dados têm sido denominados, na literatura, de dados georreferenciados, de dados geográficos, de geodados (BUEHLER & MCKEE, 1996; GARDELS, 1997) ou de dados geoespaciais (CLEMENT et al., 1997). Os dados representam elementos

geográficos como fenômenos, objetos, fatos físicos ou sociais do mundo real, sejam eles discretos ou contínuos, e suas relações com o meio.

Os dados georreferenciados são descritos em um domínio espacial, já que estão associados com a localização de pontos na superfície da Terra. Estes dados são divididos em duas partes (SOUZA et al., 1993; FRIESEN & SONDHEIM, 1994; AALDERS, 1996). A primeira parte representa as propriedades gráficas dos dados georreferenciados, denominadas de atributos gráficos. Estes atributos descrevem a localização, a extensão e relacionamento espaciais com outros objetos georreferenciados sobre uma representação, a qual irá constituir o mapa digital. A segunda parte, chamada de atributos não gráficos, privilegia as características temáticas dos fenômenos do mundo real de forma a permitir sua identificação, classificação e o estabelecimento de relações relevantes.

3.3 Conceitos Cartográficos e Geodésicos

Na construção de uma base de dados georreferenciados para SIG estão embutidos alguns conceitos da ciência cartográfica para representar os atributos gráficos. Estes conceitos estão relacionados à forma utilizada para representar a superfície terrestre, a saber: *Datum*, sistema de coordenadas, sistema de projeção e escala, utilizada para representar os objetos geográficos (ARONOFF, 1989; LAURINI & THOMPSOM, 1992; FRIESEN & SONDHEIM, 1994).

3.3.1 Representação da Terra, Elipsóide e *Datum*

A Terra vista a partir do espaço assemelha-se a uma esfera com os pólos achatados. A sua forma é afetada pela gravidade, força centrífuga de rotação e variações de densidade de suas rochas e componentes minerais (ROBINSON et al., 1978). Devido à complexidade de se trabalhar com a forma real da Terra, os cartógrafos aproximam sua superfície para um modelo do globo terrestre. Neste processo de aproximação, inicialmente se constrói um geóide, resultante da medição do nível dos oceanos. Em seguida, aproxima-se o geóide por um elipsóide de revolução, mais regular. Um elipsóide de revolução é um sólido gerado pela rotação de uma elipse em torno do eixo menor dos pólos.

Estudos geodésicos apresentam valores diferentes para os elementos de um elipsóide (raio do equador, raio polar e coeficiente de achatamento). Assim, cada região deve adotar como referência o elipsóide mais indicado. No Brasil, era adotado o elipsóide de Hayford, cujas dimensões foram consideradas as mais convenientes para a América do Sul. Atualmente, no entanto, utiliza-se com mais freqüência o elipsóide da União Astronômica Internacional, homologado em 1967 pela Associação Internacional de Geodésia que passou a se chamar elipsóide de referência.

Um *datum* é um ponto onde a superfície do elipsóide de referência coincide com a superfície terrestre, sendo caracterizado a partir de uma superfície de referência (*datum* horizontal) e de uma superfície de nível (*datum* vertical). No Brasil são encontrados, como *datum* horizontal, os mapas mais antigos no *Datum* de Córrego Alegre (MG) e, os mais recentes no *Datum* SAD 69 (*Datum* Sul Americano de 1969), porém existem mapas feitos em ambos e até mesmo com *datum* locais. Já como *datum* vertical o mais usado é o Imbituba.

Apesar do globo terrestre mostrar distâncias, áreas, formas e direções reais, ele não é o mais apropriado para lidar com processamentos sistemáticos. Para permitir tais processamentos, são criadas projeções num plano, ou seja, cada ponto do elipsóide ou esfera é projetado em uma superfície plana. A Figura 3.1 apresenta uma representação cartográfica da superfície terrestre.

3.3.2 Sistemas de Coordenadas

Qualquer objeto geográfico (como uma cidade, a foz de um rio, o cume de uma montanha) somente pode ser localizado se puder ser descrito em relação a outros objetos cujas posições sejam previamente conhecidas, ou se tiver sua localização determinada em uma rede coerente de coordenadas. Quando se dispõe de um sistema de coordenadas fixas, pode-se definir a localização de qualquer ponto na superfície terrestre.

Os sistemas de coordenadas dividem-se em dois grandes grupos: sistemas de coordenadas geográficas ou terrestres e sistemas de coordenadas planas ou cartesianas.

No sistema de coordenadas geográficas ou terrestres, historicamente mais antigo, cada ponto da superfície terrestre é localizado na interseção de um meridiano com um paralelo. Meridianos são círculos máximos da esfera cujos planos contém o eixo dos pólos. O meridiano de origem (também conhecido como inicial ou

fundamental) é usualmente aquele que passa pelo antigo observatório britânico de Greenwich. Ele é escolhido convencionalmente como a origem das longitudes sobre a superfície terrestre e como base para a contagem dos fusos horários, correspondendo ao meridiano a 0°. A leste do meridiano de origem, os meridianos são medidos por valores crescentes até +180°. A oeste, suas medidas são decrescentes até o limite mínimo de -180°.

Paralelos são círculos da esfera cujos planos são perpendiculares ao eixo dos pólos. O Equador é o paralelo que divide a Terra em dois hemisférios: Norte e Sul. O paralelo a 0° corresponde ao Equador, +90° ao Pólo Norte e -90°, ao Pólo Sul. Todos os meridianos se encontram em ambos os pólos e cruzam o equador em ângulo reto. A distância entre meridianos diminui do Equador para os pólos. Os paralelos jamais se cruzam.

Representa-se um ponto na superfície terrestre por um valor de latitude e longitude. Longitude é a distância angular entre um ponto qualquer da superfície terrestre e o meridiano de origem. Latitude é a distância angular entre um ponto qualquer da superfície terrestre e a linha do Equador. Pontos que não correspondem à medição média dos oceanos podem ter também a altitude como terceiro parâmetro.

Como o sistema de coordenadas geográficas considera desvios angulares a partir do centro da Terra, não é um sistema conveniente para aplicações em que se buscam distâncias ou áreas. Para estes casos, utilizam-se outros sistemas de coordenadas, mais adequados, como, por exemplo, o sistema de coordenadas planas, descrito a seguir.

O sistema de coordenadas planas também conhecido por sistema de coordenadas cartesianas, baseia-se na escolha de dois eixos perpendiculares, usualmente denominados eixos horizontal e vertical, cuja interseção é denominada origem, estabelecida como base para a localização de qualquer ponto do plano.

Nesse sistema de coordenadas, um ponto é representado por dois números: um correspondente à projeção sobre o eixo x (horizontal), associado principalmente à longitude, e outro correspondente à projeção sobre o eixo y (vertical), associado principalmente à latitude. Estas coordenadas são relacionadas matematicamente às coordenadas geográficas, de maneira que umas podem ser convertidas nas outras.

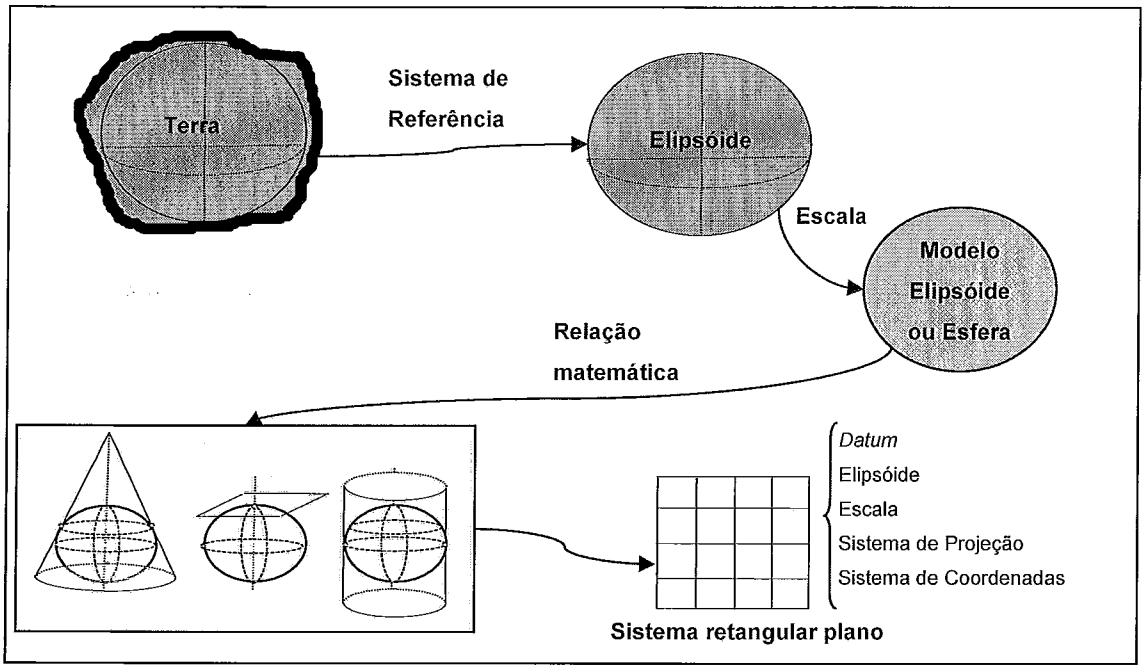


Figura 3.1 – Representação Cartográfica

3.3.3 Sistemas de Projeções

Todos os mapas são representações aproximadas da superfície terrestre, que projetam cada ponto do globo terrestre em uma superfície plana. Para se obter essa correspondência são utilizados os sistemas de projeções cartográficas.

Há um grande número de projeções cartográficas, uma vez que há uma variedade de modos de projetar em um plano os objetos geográficos que caracterizam a superfície terrestre. No entanto, é impossível se fazer uma cópia plana da superfície do globo terrestre sem desfigurá-la ou deformá-la. Esta deformação é nula nos locais onde a superfície é tangente ou secante ao elipsóide de referência. Dependendo do que se pretende analisar no mapa, cada tipo de projeção minimiza um certo tipo de deformação.

O paralelo padrão é aquele onde as deformações são nulas, isto é, a escala é verdadeira, sendo utilizado como linha de controle no cálculo de uma projeção cartográfica. A partir desse paralelo, as deformações vão aumentando progressivamente sobre os paralelos e sobre os meridianos, com valores desiguais.

A longitude de origem é representada por uma linha reta, que constitui o eixo de simetria, no sentido vertical. A definição de longitude de origem depende da projeção

adotada. A latitude de origem refere-se ao paralelo padrão mais próximo à região de interesse. Dependendo da projeção utilizada, define-se ou não a latitude de origem.

Denomina-se transformação cartográfica projetiva a operação de converter dados espaciais representados segundo um determinado sistema de projeção cartográfica para outro sistema de projeção.

Cada método de projeção da superfície terrestre preserva diferentes propriedades espaciais: área, direção, distância e forma. A preservação de uma propriedade implica normalmente na distorção das demais. Assim, quanto ao grau de deformação das superfícies representadas, as projeções são classificadas em conformes ou isogonais, equivalentes ou isométricas e eqüidistantes. As afiláticas são aquelas que não preservam nenhuma dessas três propriedades em detrimento de uma propriedade matemática.

As projeções conformes ou isogonais mantêm fidelidade aos ângulos locais observados na superfície representada (por exemplo, Mercator). As projeções equivalentes ou isométricas conservam as relações de superfície, não havendo deformação de área (por exemplo, Cônica de Albers, Azimutal de Lambert). Já as projeções equidistantes conservam a proporção entre as distâncias, em determinadas direções, na superfície representada (por exemplo, a Cilíndrica Eqüidistante).

3.3.4 Escala

Escala é a relação entre as dimensões dos elementos representados em um mapa e a grandeza correspondente, medida sobre a superfície da Terra. A escala é uma informação obrigatória para qualquer mapa e geralmente está representada de forma numérica. As escalas numéricas ou fracionárias são descritas por frações cujos denominadores representam as dimensões naturais e os numeradores, as que lhes correspondem no mapa.

A escala de 1 para 50.000 (notação 1:50.000 ou 1/50.000), por exemplo, indica que uma unidade de medida no mapa equivale a 50.000 unidades da mesma medida sobre o terreno. Assim, 1cm no mapa corresponde a 50.000cm no terreno.

Um dos problemas ao se utilizar SIGs reside no fato de que um mapa digital pode ser apresentado ou impresso em qualquer escala, independentemente da escala de origem em que os dados foram armazenados. Porém, caso a escala escolhida seja maior que a original, não há naturalmente ganho de precisão. Por outro lado, se a escala escolhida for menor, detalhes e precisão podem ser perdidos.

Capítulo 4 – Arquitetura WISE

Neste capítulo é descrita a Arquitetura do WISE, iniciando com uma caracterização, em seguida é apresentado o Modelo de Dados Canônico utilizado pelo WISE para representar as diversas fontes de dados heterogêneas, passando pela organização em camadas do WISE, sendo em seguida apresentados os papéis dos usuários e o funcionamento geral do WISE, e finalmente o detalhamento dos seus componentes é apresentado.

4.1 Caracterização da Arquitetura WISE

A arquitetura do Web data Integration SystEm (WISE) visa resolver alguns dos problemas semânticos que surgem durante a integração de dados convencionais e georreferenciados na *World Wide Web*. O objetivo principal é a integração e compartilhamento de dados georreferenciados entre diversos grupos de usuários.

A arquitetura proposta para este sistema visa abranger uma grande variedade de usuários, e para isto, foi idealizado um sistema multiplataforma, que utiliza código fonte aberto ou livre, que consiga aderir aos formatos de troca de dados mais atuais e mais genéricos utilizados nas tecnologias de Banco de Dados e de Geoprocessamento.

Diferentemente de outros sistemas de integração e publicação de dados georreferenciados (SOROKINE & MERZLIAKOVA, 1998; ZASLAVSKY et al., 2000), que apresentam resultados de consultas usando arquivos de imagens raster ou applets para visualização, e geralmente não permitem sua manipulação posterior (ou permitem uma manipulação restrita) apropriada às necessidades dos usuários-alvo, a arquitetura do WISE pretende tornar disponível os dados integrados em diversos formatos de arquivo permitindo seu processamento e manipulação pelos usuários.

O WISE faz parte do projeto SPeCS (MEDEIROS et al., 2001), Figura 4.1, que tem por objetivo fornecer um ambiente de trabalho cooperativo comum, flexível e fácil de usar onde os membros do grupo podem estar espalhados geograficamente em diversos ambientes heterogêneos e mesmo assim serem capazes de interagir durante um processo de tomada de decisão. Trata-se de um *framework* para suportar a decisão espacial colaborativa. Neste projeto são implementadas ações de pesquisa em:

Integração de Bases de Dados, Ferramentas de Trabalho Cooperativo, Ferramentas e Técnicas de Workflow, Modelagem Matemática para apoio à decisão, Acesso e manipulação de informações via Internet.

O WISE visa substituir a X-Arc que era, até então, o sistema publicador de dados georreferenciados do SpeCS (MEDEIROS et al., 2001). Cabe ressaltar que o WISE é um superconjunto da X-Arc, e realiza a tarefa de integrar e publicar dados georreferenciados. O WISE, assim como a X-Arc, é uma arquitetura baseada em mediadores.

A Figura 4.1 mostra a estrutura do SpeCS, e a sua Camada de Integração de dados onde residia a X-Arc, e onde o WISE está sendo incorporado para substituí-la.

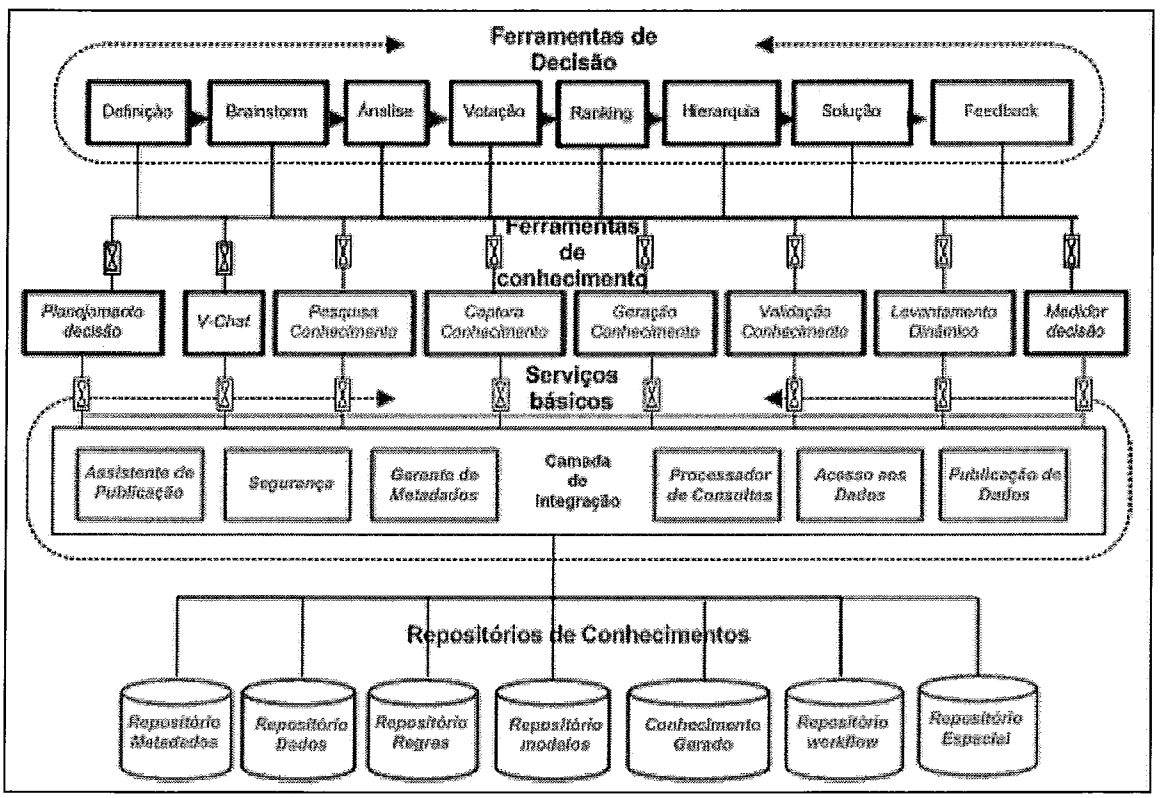


Figura 4.1 – Estrutura do SpeCS onde o WISE está sendo inserido

As principais características levadas em conta para desenvolver o WISE como o sucessor da X-Arc foram que ele deveria ser um sistema para integração semântica e publicação de dados, tanto convencionais quanto georreferenciados, enquanto que a X-Arc é um sistema publicador de dados convencionais e georreferenciados. Logo o WISE tem como principal meta integrar dados heterogêneos, enquanto que a X-Arc visa disponibilizá-los.

A proposta do WISE é fornecer uma gama de formatos de dados que sejam mais utilizados pelos usuários da área de geoprocessamento, enquanto a X-Arc somente dispõe de XML e arquivos de imagem raster.

Ele também possui a capacidade de criar novos tradutores como na X-Arc, mas além disso possui a capacidade de estender outros módulos para se adaptar a novos formatos de dados de saída.

O WISE utiliza bibliotecas de código fonte aberto na sua construção, enquanto que a X-Arc, atualmente, utiliza um sistema livre, LeSelect (LESELECT, 2002), e um proprietário, ARCIMS (ESRI, 2003).

4.2 Modelo de Dados Canônico do WISE

O processo de integração de dados ocorre com maior freqüência sobre Bancos de Dados Relacionais e Orientados a Objetos, que utilizam dados estruturados. Porém atualmente, com a crescente utilização da *Web* e a disponibilidade de um grande volume de dados semi-estruturados, através de linguagens de marcação como SGML, XML, GML e outras (W3C, 2003; BOTÊLHO & SOUZA, 2003), a demanda de integração de dados semi-estruturados tem aumentado.

Os dados semi-estruturados vêm surgindo como um importante tópico de pesquisa por várias razões: a) existem fontes de dados semi-estruturados, as quais poderiam ser tratadas como bancos de dados, através do emprego de técnicas, ferramentas e metodologias disponíveis para a manipulação de bancos de dados convencionais que são estendidas para tratar destas fontes; b) é importante ter um formato extremamente flexível para troca de informações entre fontes de dados heterogêneas, no qual todos os modelos de dados pudessem ser convertidos facilmente; e c) mesmo se tratando de dados estruturados, pode ser interessante visualizar esses dados como sendo semi-estruturados. Por exemplo, consultar as fontes de dados sem necessariamente conhecer o esquema associado a elas.

A linguagem de marcação extensível, XML (*eXtensible Markup Language*) foi proposta pelo W3C (W3C, 2003) como um padrão para representação e troca de informações na *Web*. Devido a sua flexibilidade em representar informações com estruturas heterogêneas (tanto dados estruturados como semi-estruturados) e a facilidade que se tem em converter qualquer dado nela, esta linguagem vem sendo proposta, também, como um padrão para integração de dados (GOLDMAN et al., 1999;

CHRISTOPHIDES et al., 2000). Como a XML pode ser usada para fornecer uma representação uniforme e flexível dos dados de fontes heterogêneas, pode-se utilizá-la como um MDC, capaz de representar diversos modelos de dados. Sendo assim, vários sistemas de integração de dados, atuais, tem utilizado XML como MDC (OZSU & VALDURIEZ, 1999; BARU et al., 1999; GARDARIN et al., 1999; ABITEBOUL et al., 2000; VIDAL et al., 2001a).

Do mesmo modo que XML pode ser usada para representar um MDC sobre dados convencionais, pode-se usar, uma de suas extensões, GML (*Geography Markup Language*) (GML, 2003) para representar um MDC sobre dados georreferenciados. Desta forma o WISE utiliza no seu MDC chamado XML Schema Semântico, as linguagens XML e GML, para representar dados convencionais como documentos XML, arquivos texto, planilhas do Microsoft Excel (MICROSOFT EXCEL, 2004), SGBDs convencionais, entre outros; e dados georreferenciados, como arquivos ESRI Shape (SHAPE FILE, 2003), GML, SGBDs não-convencionais, etc.

Neste trabalho são definidas algumas restrições sobre XML e GML para formar um modelo no qual fosse possível a integração de dados de maneira mais simples, abstraindo detalhes desnecessários ao domínio de aplicações que tratam de dados georreferenciados, que é o foco deste trabalho. Este MDC possui também a possibilidade de permitir que sejam adicionadas mais informações semânticas, através de ontologias do domínio da aplicação. A atual implementação do WISE usa um conjunto restrito de ontologias, que seria mais aproximado de um thesaurus, porém será utilizado o termo ontologia ao longo deste trabalho. Como o estudo de ontologias é por si só vasto, este trabalho não entrará em muitos detalhes sobre este tema, devendo os interessados procurar maiores detalhes na literatura (FONSECA et al., 2000a; FONSECA et al., 2000b).

4.2.1 Estrutura do XML Schema Semântico

O MDC do WISE, XML Schema Semântico (ou SXMLS, de *Semantic XML Schema*) é constituído de duas partes lógicas, uma para representação do modelo das fontes de dados na forma de XML Schemas, chamado Esquema Base, e a outra para representar a integração destes XML Schemas das fontes de dados, de nome Esquema Integrado Base. Estes esquemas foram concebidos utilizando características desenvolvidas neste trabalho em conjunto com alguns componentes de GML. A Figura

4.2 mostra um diagrama de classes da UML com a estrutura do XML Schema Semântico e seus esquemas componentes.

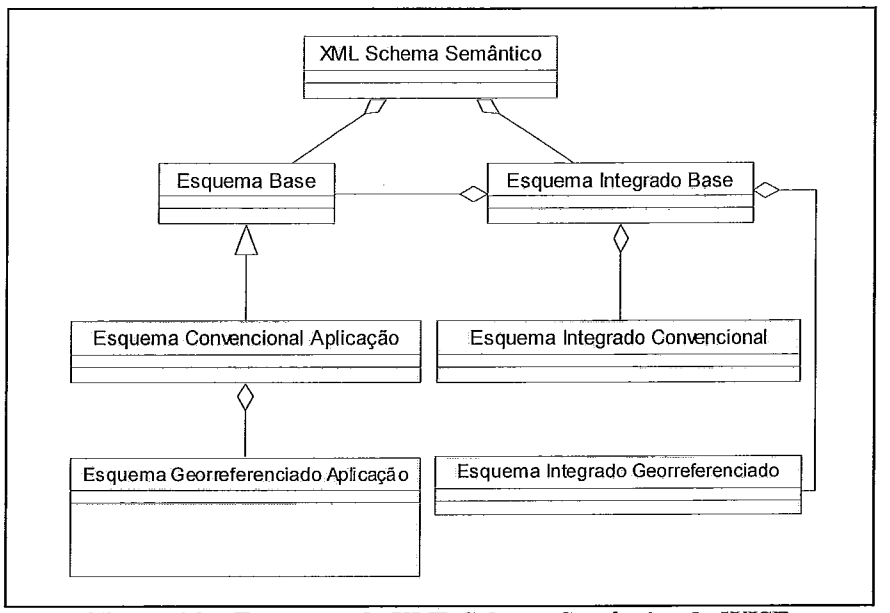


Figura 4.2 – Estrutura do XML Schema Semântico do WISE

4.2.2 Esquema Base

O MDC do WISE, SXMLS, foi definido utilizando as linguagens XML e GML, tendo como ponto de partida a utilização de um tipo de esquema XML, o XML Schema (XML SCHEMA, 2003) para definir a sua estrutura. Com XML Schema foi possível definir o Esquema Base, que é a parte fundamental do SXMLS, onde foi construído um conjunto de elementos e tipos complexos de XML que definem uma estrutura de agregação capaz de representar o esquema local de uma fonte de dados e o seu conteúdo. A estrutura do Esquema Base é uma estrutura básica que deve ser estendida por outros esquemas para representar o conteúdo de uma fonte de dados e para que sejam acrescentadas novas características, como será apresentado em breve nesta seção.

O Esquema Base é representado pelo arquivo XML Schema "SXMLS-Base.xsd", mostrado na Listagem 4.1 e por um diagrama em árvore na Figura 4.3.

A estrutura do Esquema Base, é definida por um elemento para representar a raiz do XML Schema, chamado "Dados Base" (_BaseData). Após este elemento raiz são encontrados os elementos para representar uma fonte de dados e os seus componentes. O elemento "Fonte de Dados" (_DataSource) representa a única fonte de dados que o Esquema Base pode ter, e este elemento deve conter um elemento "Conjuntos de Dados" (_DataSets) para representar em seu conteúdo diversos elementos de conjuntos

de dados, podendo ser elementos "Conjunto de Dados" (_DataSet) para dados convencionais ou elementos "Coleção de Feições" (_FeatureCollection) para dados georreferenciados. Um elemento "Conjunto de Dados" deve possuir um elemento "Campos" (_Fields) para que sejam definidos campos em seu interior. Já o elemento "Coleção de Feições", que é usado neste esquema pertence à linguagem GML e está definido no XML Schema "feature.xsd" da mesma, e é do tipo AbstractFeatureCollectionType, e possui uma estrutura predefinida como apresentado na Figura 4.4.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ======================================================================
     includes and imports
     ====================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="D:\ProgsMestrado\XML_GML\Luc GML
3.0.1\base\feature.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ======================================================================
     global declarations
     ====================================================================== -->
    <element name="_BaseData" type="wisesxmls:BaseSchemaType" abstract="true"/>
    <!--================================================================== -->
    <element name="_DataSource" type="wisesxmls:AbstractDataSourceType" abstract="true"/>
    <!--================================================================== -->
    <element name="_DataSets" type="wisesxmls:AbstractDataSetsType" abstract="true"/>
    <!--================================================================== -->
    <element name="_DataSet" type="wisesxmls:AbstractDataSetType" abstract="true"/>
    <!--================================================================== -->
    <element name="_Fields" type="wisesxmls:AbstractFieldsType" abstract="true"/>
    <!-- ======================================================================
     complex types
     ====================================================================== -->
    <complexType name="BaseSchemaType">
        <sequence>
            <element ref="wisesxmls:_DataSource"/>
        </sequence>
    </complexType>
    <!--================================================================== -->
    <complexType name="AbstractDataSourceType" abstract="true">
        <sequence>
            <element ref="wisesxmls:_DataSets"/>
        </sequence>
    </complexType>
    <!--================================================================== -->
    <complexType name="AbstractDataSetsType" abstract="true">
        <choice>
            <element ref="wisesxmls:_DataSet" minOccurs="1" maxOccurs="unbounded"/>
            <element ref="gml:_FeatureCollection" minOccurs="1" maxOccurs="unbounded"/>
        </choice>
    </complexType>
    <!--================================================================== -->
    <complexType name="AbstractDataSetType" abstract="true">
        <sequence>
            <element ref="wisesxmls:_Fields"/>
        </sequence>
    </complexType>
    <!--================================================================== -->
    <complexType name="AbstractFieldsType" abstract="true"/>
    <!--================================================================== -->
</schema>
```

Listagem 4.1 – Esquema Base representado em XML Schema

O Esquema Base possui uma estrutura básica necessária para definir como um esquema local de uma fonte de dados genérica será representado. Porém para que um esquema local de uma fonte de dados específica, que possui várias características, possa ser representado em um SXMLS, o Esquema Base deve ser estendido para contemplar esta fonte de dados. A este esquema estendido, dá-se o nome de Esquema Aplicação.

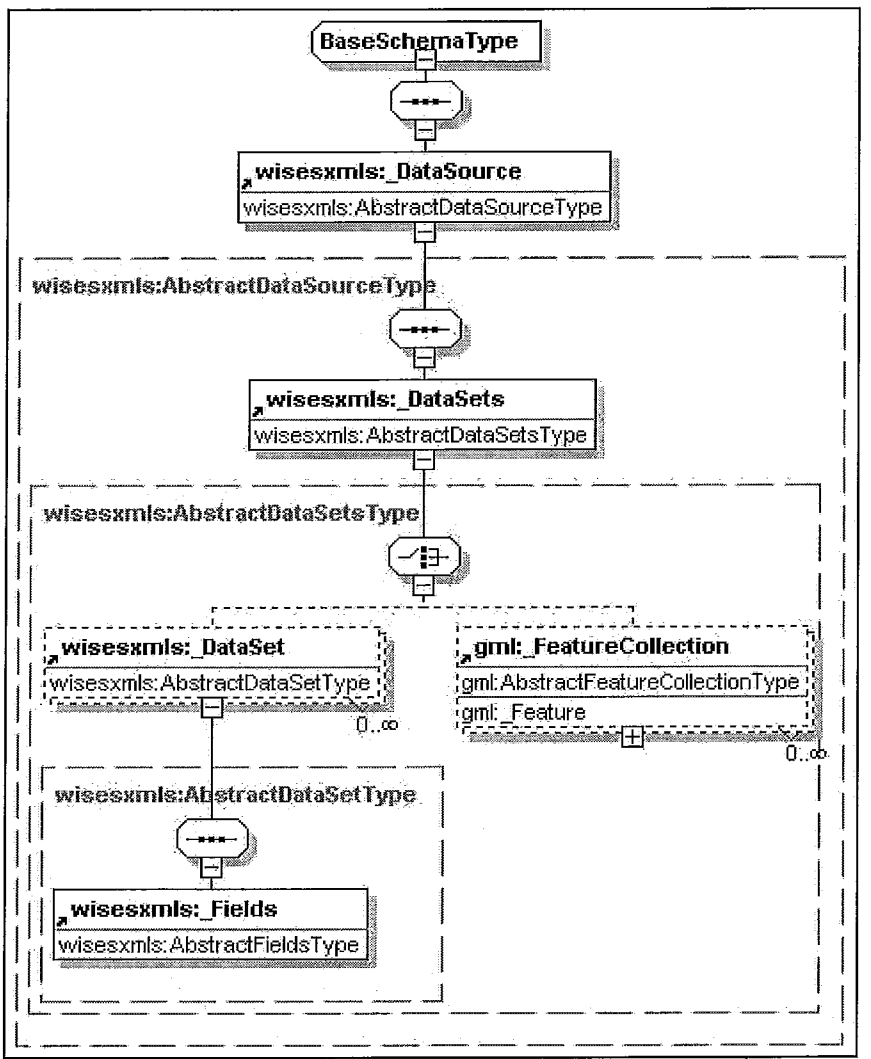


Figura 4.3 – Esquema Base representado graficamente[1]

[1] Os símbolos —•••—, ⊣⊞⊢ descrevem, respectivamente, os delimitadores de grupo *sequence* e *choice* da XML. Uma descrição sobre estes pode ser observada no Apêndice B.

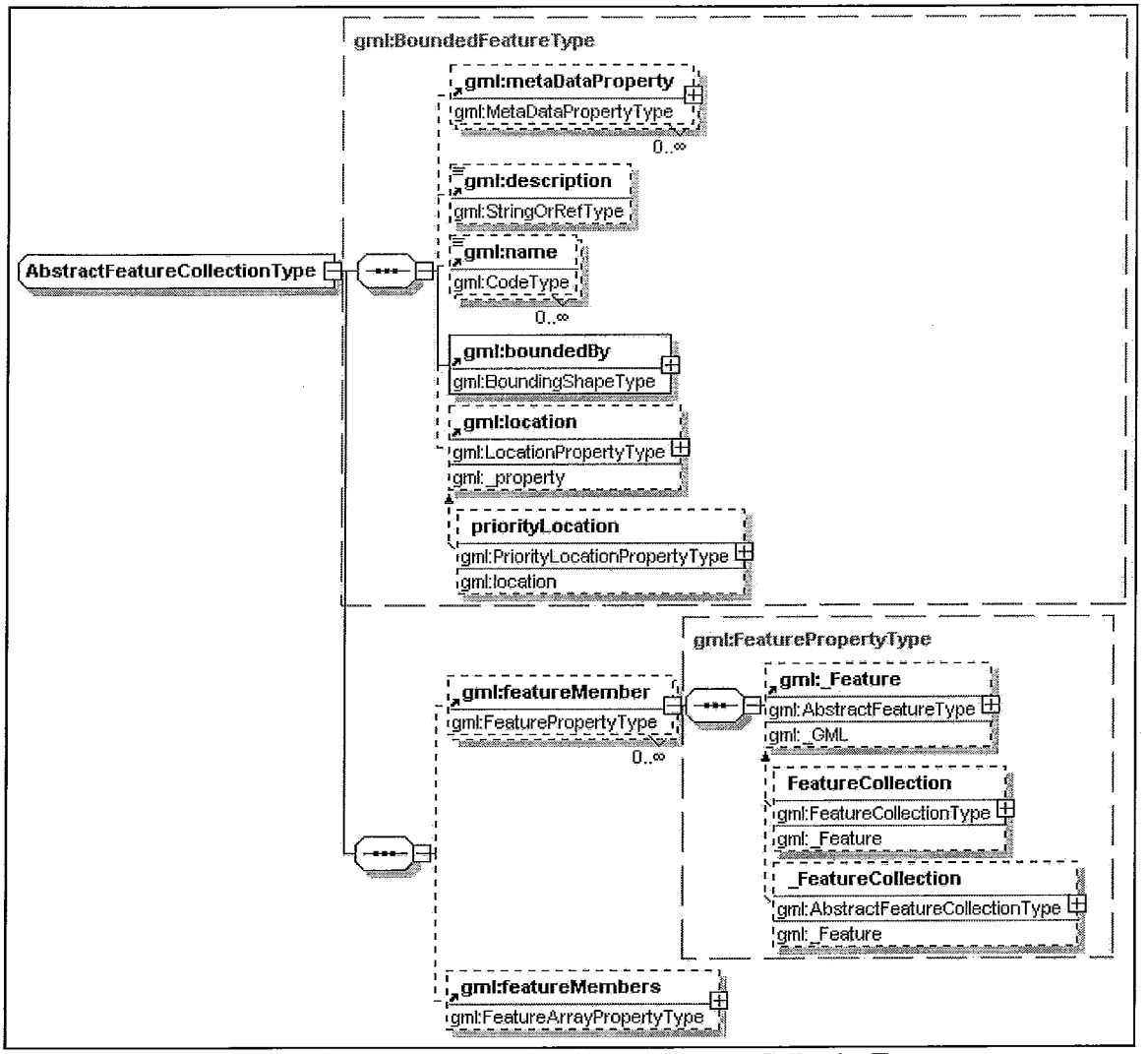


Figura 4.4 – Estrutura do Tipo AbstractFeatureCollectionType

4.2.2.1 Esquema Aplicação

O Esquema Aplicação é um tipo de esquema construído, através de especialização do Esquema Base, para representar o esquema local de uma fonte de dados particular.

O Esquema Aplicação é constituído de um XML Schema para representar dados convencionais, chamado Esquema Convencional Aplicação, que é herdado do Esquema Base. Quando o Esquema Aplicação é utilizado para representar uma fonte de dados georreferenciada é também criado, além do Esquema Convencional Aplicação, um outro tipo de XML Schema, baseado em GML (também chamado GML Schema), para representar os dados georreferenciados, de nome Esquema Georreferenciado Aplicação. Todo Esquema Georreferenciado Aplicação está agregado a um Esquema Convencional Aplicação. Cabe ressaltar que o Esquema Georreferenciado Aplicação não é criado a

partir da especialização do Esquema Base, mas de um conjunto de elementos e tipos de dados de GML e um conjunto de regras desenvolvidas neste trabalho.

Levando em consideração que os arquivos ESRI Shape que são bastante utilizados na área de geoprocessamento, e outros formatos de arquivos, representam somente um Plano de Informação por arquivo, também chamado de Coleção de Feições (FeatureCollection em GML), resolveu-se utilizar neste trabalho esta mesma concepção nos Esquemas Georreferenciados Aplicação. Assim um Esquema Georreferenciado Aplicação é capaz de representar uma e somente uma Coleção de Feições (FeatureCollection). Está escolha também se deve ao fato de que seria mais rápido converter de um outro formato de arquivo para o formato GML e vice-versa caso fosse possível utilizar uma única Coleção de Feições e não várias.

Para tratar arquivos do tipo geodatabase, que são compostos de várias Coleções de Feições, é necessário separar cada uma destas em Esquemas Georreferenciados Aplicação distintos. Em trabalhos futuros, podem existir estudos sobre a questão de mantê-los em um único Esquema Georreferenciado Aplicação.

Os Esquemas Georreferenciados Aplicação devem estar agregados a um Esquema Convencional Aplicação, que representa a sua Fonte de Dados, já que estes esquemas não a representam. Sendo assim, quando utilizados dados convencionais estes são representados por Conjuntos de Dados dentro do elemento Fonte de Dados do Esquema Convencional Aplicação, e quando os dados utilizados são georreferenciados serão utilizados tantos Esquemas Georreferenciados Aplicação quantas forem as Coleções de Feições existentes. Cada Esquema Georreferenciado Aplicação deve estar associado a um único Esquema Convencional Aplicação através de um elemento FeatureCollection deste, de mesmo nome do Esquema Georreferenciado Aplicação, dentro de um elemento Fonte de Dados.

Para que seja realizada a integração de dados, além dos metadados básicos de XML Schema que definem a estrutura dos documentos XML, precisa-se de um esquema capaz de armazenar mais informações semânticas sobre os esquemas locais das fontes de dados. Como estes tipos de elementos não podem ser herdados, eles não foram definidos no Esquema Base e serão usados somente nos Esquemas Aplicação. Logo, os Esquemas Aplicação podem ter elementos que incluirão mais restrições em relação ao Esquema Base. Nestes elementos podem ser utilizadas as funcionalidades de XML, como chaves, restrições de integridade, bem como limitar os tipos de dados

simples (ou básicos) para adicionar restrições. E também podem ser adicionados outros elementos, chamados elementos informativos, criados neste trabalho para que seja adicionada mais semântica e descrições aos esquemas.

4.2.2.2 Elementos do Esquema Aplicação

Os elementos do Esquema Aplicação são descritos abaixo, estando separados por funcionalidades.

Elementos Restritivos e de Integridade

Estes elementos visam aplicar restrições para que o modelo fique mais consistente e refletir com maior fidelidade os esquemas heterogêneos das fontes de dados do WISE.

Os elementos utilizados em um Esquema Aplicação para definirem campos não podem pertencer a qualquer tipo de dados, mas somente a tipos de dados simples de XML. Desta forma, foi criado um conjunto de tipos de dados simples, herdados dos tipos simples de XML, para formar um conjunto de tipos permitidos para definir um elemento campo. Estes tipos de dados, chamados Tipos de Dados Simples WISE, estão no arquivo "SXMLS-SimpleDataTypes.xsd", no Apêndice C, que também faz parte do XSMLS. Estes tipos de dados simples do WISE representam números inteiros e ponto-flutuante, data, tempo, texto e valores booleanos, entre outros. Estes tipos podem ainda ser estendidos e podem ser agregadas mais restrições aos valores permitidos.

XML possui um conjunto de elementos para representar integridade de chaves (key), e chaves estrangeiras (keyref), bem como um elemento para garantir a unicidade (unique) de valores, como pode ser visto no Apêndice B. Estes elementos podem ser utilizados no Esquema Convencional Aplicação para representar fontes de dados que possuam restrições de integridade.

No Esquema Convencional Aplicação os elementos key, keyref e unique podem somente ser usados como filhos diretos do elemento DataSets, que representa uma agregação de Conjuntos de Dados, como pode ser visto a seguir.

Elementos Informativos

A formalização de XML Schema impede que possam ser adicionados dados neste esquema, ao contrário de metadados. E se fossem utilizados procedimentos

artificiais para que estes dados pudessem ser inseridos em um XML Schema na forma de metadados, estes seriam replicados nos documentos XML baseados neste esquema e ficariam redundantes. Deste modo foi criado neste trabalho, através de elementos do tipo "annotation/appinfo" (tipo de anotação XML para tornar disponível informações para aplicações), um conjunto de metadados extra para o SXMLS, para que sejam representados dados e metadados a respeito das fontes de dados em um XML Schema.

Este tipo de elemento não é replicado nos documentos XML, e são úteis na integração de dados, além de poderem ser analisados por um compilador de XML de forma natural. Todos esses elementos podem ser analisados por humanos e alguns deles podem ser analisados por um compilador para auxiliar no processo de integração de esquemas e integração de dados. Os elementos criados são:

- Lineage – faz referência à qualidade dos dados, como foram adquiridos e processados. Pode ser usado como filho do elemento BaseData de um Esquema Convencional Aplicação, ou como filho do elemento FeatureCollection do Esquema Georreferenciado Aplicação;
- Source – informa qual a procedência dos dados. Também pode ser usado como filho do elemento BaseData de um Esquema Convencional Aplicação, ou como filho do elemento FeatureCollection do Esquema Georreferenciado Aplicação;
- Ontology.Name – possibilita que sejam adicionados um ou mais nomes. E estes nomes podem ser usados como ontologias do domínio da aplicação, sendo que o primeiro nome é o nome real do assunto a ser descrito;
- Description – permite que sejam adicionadas descrições;
- UnitOfMeasure – permite que seja adicionada uma unidade de medida. Este elemento só pode ser usado com um elemento campo dentro do elemento Fields.

Estes elementos podem ser utilizados uma única vez dentro de um elemento pai. Estes elementos são mutuamente independentes.

Os elementos informativos têm seu nome prefixado com "WISE=" para diferenciá-los dos demais elementos que estiverem dentro do elemento appinfo. E caso só possam ser utilizados em determinados elementos, possuem ainda o nome de tal elemento seguido do símbolo "=" após o *token* "WISE=". Um exemplo pode ser visto na Listagem 4.2:

```
<annotation>
   <appinfo>
      <element name="WISE=BaseData=Source">"ENCE/IBGE"</element>
      <element name="WISE=Ontology.Name">"Rio", "Rio", "Curso de Água"</element>
      <element name="WISE=Description">Rios do estado de Minas Gerais</element>
      <element name="WISE=Field=UnitOfMeasure">"Km"</element>
   </appinfo>
</annotation>
```

Listagem 4.2 – Alguns Elementos Informativos

Elementos Informativos Georreferenciados

Além dos Elementos Informativos vistos acima, foram criados outros elementos informativos para tratar somente dados georreferenciados, chamados Elementos Informativos Georreferenciados. Estes elementos representam conceitos da cartografia e geodésia, como apresentados no Capítulo 3, que estão presentes nas fontes de dados georreferenciadas: Datum, Ellipsoid (Elipsóide), ProjectionSystem (Sistema de Projeção), CoordinateSystem (Sistema de Coordenadas), e Scale (Escala). Estes elementos podem ser vistos na Listagem 4.3.

```
<annotation>
   <appinfo>
      <element name="WISE=FeatureCollection=Datum">DatumInfo</element>
      <element name="WISE=FeatureCollection=Ellipsoid">EllipsoidInfo</element>
      <element name="WISE=FeatureCollection=ProjectionSystem">ProjectionSystemInfo</element>
      <element name="WISE=FeatureCollection=CoordinateSystem">CoordinateSystemInfo</element>
      <element name="WISE=FeatureCollection=Scale">ScaleInfo</element>
      <element name="WISE=FeatureCollection=Source">SourceInfo</element>
      <element name="WISE=FeatureCollection=Lineage">LineageInfo</element>
      <element name="WISE=Description">DescriptionText</element>
   </appinfo>
</annotation>
```

Listagem 4.3 – Alguns Elementos Informativos Georreferenciados

4.2.2.3 Criação do Esquema Convencional Aplicação

Como já foi dito previamente, um Esquema Convencional Aplicação representa um esquema local de uma fonte de dados. Para a criação do Esquema Convencional Aplicação, deve-se estender os elementos e tipos abstratos definidos no Esquema Base, de modo a adaptá-los ao esquema local de uma fonte de dados, como pode ser visto na Listagem 4.4. São definidas algumas regras para a criação de um Esquema Convencional Aplicação:

- O elemento BaseData que é a raiz do Esquema Convencional Aplicação deve ser criado para substituir o elemento abstrato _BaseData do Esquema Base, e manter o novo conjunto de elementos não-abstratos.

- Um elemento "Fonte de Dados" (DataSource) deve ser criado com o mesmo nome da Fonte de Dados local que se deseja representar. O elemento "Fonte de Dados" é criado por extensão do tipo AbstractDataSourceType, e irá substituir o elemento abstrato _DataSource.
- Um elemento "Conjuntos de Dados" (DataSets) deve ser criado por extensão do tipo AbstractDataSetsType, que por sua vez irá substituir o elemento abstrato _DataSets. O DataSets pode incorporar elementos para definição de chaves, restrições de integridade e unicidade para os elementos "Conjunto de Dados" (DataSet) e "Coleção de Feições" (FeatureCollection) que estarão agregados a ele.
- Se o Esquema Convencional Aplicação representa uma fonte de dados convencional:
    - Devem ser criados tantos elementos "Conjunto de Dados" (DataSet), quantos forem os conjuntos de dados que a fonte de dados possuir. Cada um desses deve possuir um nome correspondente ao conjunto de dados ao qual se referem. Estes elementos realizam uma extensão por restrição do tipo AbstractDataSetType, que seria o mesmo que redefinir o tipo para este elemento. Isto é necessário, pois XML não tem o conceito de polimorfismo de sobrescrição do paradigma orientado a objetos. Sendo assim seu elemento filho (Fields) deve ser adicionado novamente, que agora deve se tornar uma referência para outro elemento global do seu mesmo tipo, observando que deve usar o mesmo nome deste último.
    - Cada "Conjunto de Dados" definido anteriormente possui "Campos" (Fields). Logo, deve ser criado um elemento global Fields, estendido a partir do tipo AbstractFieldsType, para cada "Conjunto de Dados" referenciar. Cada elemento Fields irá conter um ou mais elementos que possam representar um campo de um dado conjunto de dados. Cada campo, além de nome, possuem um Tipo de Dados Simples WISE.
- Se o Esquema Convencional Aplicação representa uma fonte de dados georreferenciada:
    - Devem ser criados tantos elementos "Coleção de Feições" (FeatureCollection), quantos forem as coleções de feição que a fonte de dados possuir. Cada um desses apontará para uma "Coleção de Feições"

no Esquema Georreferenciado Aplicação de mesmo nome. O nome do Esquema Georreferenciado Aplicação deve ser igual ao da sua "Coleção de Feições". O processo de criação deste será mostrado na seção 4.2.2.4.

- Cada um dos Esquemas Georreferenciados Aplicação deve ter o nome de seu arquivo inserido em um elemento "include" no Esquema Convencional Aplicação.
- O esquema "feature.xsd" da GML deve estar importado no Esquema Convencional Aplicação.

De acordo com as regras supracitadas foi definido um esquema modelo que representa uma fonte de dados convencional fictícia e é apresentado na Listagem 4.4. Este esquema mostra um exemplo de como alguns elementos são estendidos a partir do Esquema Base, e como alguns elementos informativos poderiam estar distribuídos. A Figura 4.5 mostra um diagrama deste esquema.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <include schemaLocation="SXMLS-Base.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="BaseData" type="wisesxmls:BaseSchemaType" substitutionGroup="wisesxmls:_BaseData">
        <annotation>
            <appinfo>
                <element name="WISE=BaseData=Source">SourceInfo</element>
                <element name="WISE=BaseData=Lineage">LineageInfo</element>
                <element name="WISE=Description">DescriptionText</element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================ -->
    <element name="DataSourceName" substitutionGroup="wisesxmls:_DataSource">
        <annotation>
            <appinfo>
                <element name="WISE=Ontology.Name">"DataSourceName", "Onto2", "Onto3"</element>
                <element name="WISE=Description">DescriptionText</element>
            </appinfo>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="wisesxmls:AbstractDataSourceType"/>
            </complexContent>
        </complexType>
    </element>
    <!--================================================================ -->
    <element name="DataSets" substitutionGroup="wisesxmls:_DataSets">
        <complexType>
            <complexContent>
                <extension base="wisesxmls:AbstractDataSetsType"/>
            </complexContent>
        </complexType>
        <!-- ============== keys constraints ============== -->
        <key name="DataSetA.key">
```

```
        <selector xpath="DataSetA"/>
        <field xpath="Fields1/F1"/>
    </key>
    <key name="DataSetB.key">
        <selector xpath="DataSetB"/>
        <field xpath="Fields2/F5"/>
        <field xpath="Fields2/F6"/>
    </key>
    <!-- ================ keyrefs constraints ================ -->
    <keyref name="DataSetB.keyref" refer="wisesxmls:DataSetA.key">
        <selector xpath="DataSetB"/>
        <field xpath="Fields2/F5"/>
    </keyref>
    <!-- ================ unique constraints ================ -->
    <unique name="DataSetA.unique.F2">
        <selector xpath="DataSetA"/>
        <field xpath="Fields1/F2"/>
    </unique>
</element>
<!--================================================================== -->
<element name="DataSetA" substitutionGroup="wisesxmls:_DataSet">
    <annotation>
        <appinfo>
            <element name="WISE=Ontology.Name">"DataSetA", "Onto4", "Onto5"</element>
            <element name="WISE=Description">DescriptionText</element>
        </appinfo>
    </annotation>
    <complexType>
        <complexContent>
            <restriction base="wisesxmls:AbstractDataSetType">
                <sequence>
                    <element ref="wisesxmls:Fields1"/>
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
</element>
<!--================================================================== -->
<element name="DataSetB" substitutionGroup="wisesxmls:_DataSet">
    <complexType>
        <complexContent>
            <restriction base="wisesxmls:AbstractDataSetType">
                <sequence>
                    <element ref="wisesxmls:Fields2"/>
                </sequence>
            </restriction>
        </complexContent>
    </complexType>
</element>
<!--================================================================== -->
<element name="Fields1">
    <complexType>
        <complexContent>
            <extension base="wisesxmls:AbstractFieldsType">
                <sequence>
                    <element name="F1" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Ontology.Name">"F1", "Onto6", "Onto7"</element>
                                <element name="WISE=Description">DescriptionText</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="F2" type="wisesxmls:wiseInt"/>
                    <element name="F3" type="wisesxmls:wiseString"/>
                    <element name="F4" type="wisesxmls:wiseDate"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<!--================================================================== -->
<element name="Fields2">
    <complexType>
        <complexContent>
```

```
            <extension base="wisesxmls:AbstractFieldsType">
               <sequence>
                  <element name="F5" type="wisesxmls:wiseInt"/>
                  <element name="F6" type="wisesxmls:wiseInt">
                     <annotation>
                        <appinfo>
                           <element name="WISE=Ontology.Name">"F6", "Onto7", "Onto8"</element>
                           <element name="WISE=Description">DescriptionText</element>
                           <element name="WISE=Field=UnitOfMeasure">UOMInfo</element>
                        </appinfo>
                     </annotation>
                  </element>
                  <element name="F7" type="wisesxmls:wiseFloat"/>
                  <element name="F8" type="wisesxmls:wiseBoolean"/>
               </sequence>
            </extension>
         </complexContent>
      </complexType>
   </element>
   <!--================================================================== -->
</schema>
```

Listagem 4.4 – Exemplo de Esquema Convencional Aplicação

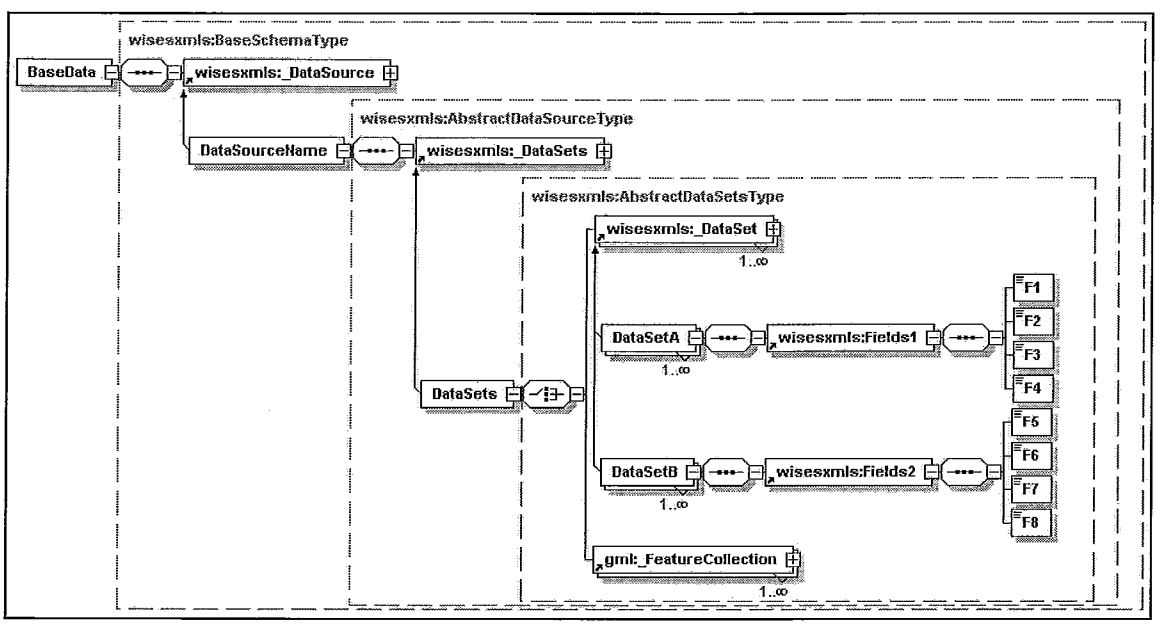


Figura 4.5 – Representação Gráfica do Esquema Convencional Aplicação

4.2.2.4 Criação do Esquema Georreferenciado Aplicação

Os Esquemas Georreferenciados Aplicação são criados para representar as coleções de feições do esquema local de uma fonte de dados georreferenciada.

O Esquema Georreferenciado Aplicação é definido através de um conjunto de tipos de dados e elementos da GML e um conjunto de regras que foram adicionadas neste trabalho. Basicamente são herdados dois tipos de dados da GML, o AbstractFeatureCollectionType, já mostrado na Figura 4.4, e o AbstractFeatureType que representa um tipo de feição, mostrado na Figura 4.6.

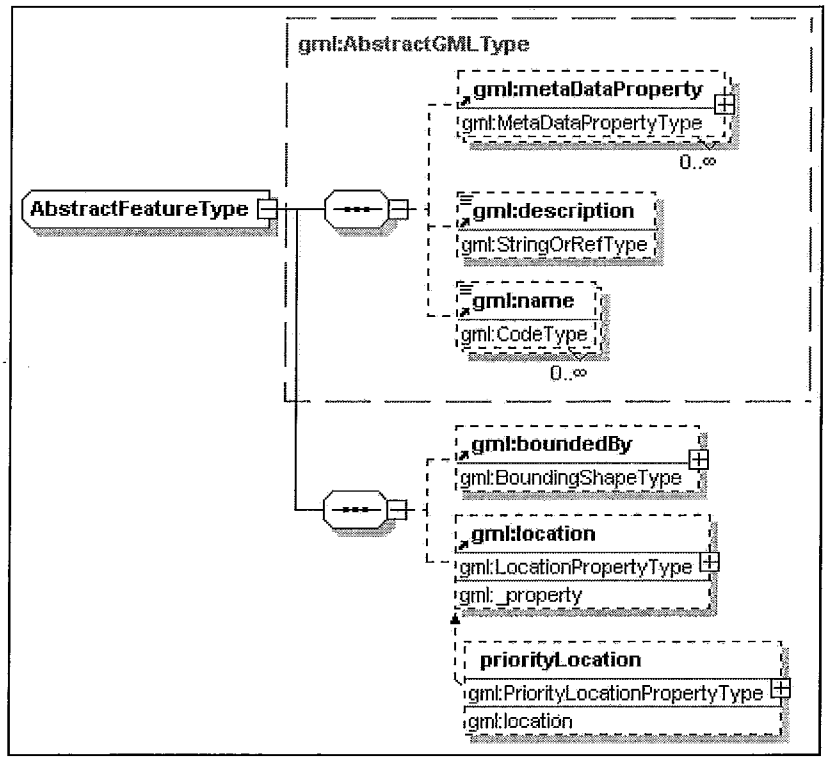


Figura 4.6 – Representação Gráfica do AbstractFeatureType

Além destes dois tipos, é usado um elemento chamado geometryMember, do XML Schema "geometryAggregates.xsd" da GML, para representar objetos geométricos, que é usado no Esquema Georreferenciado Aplicação para representar o seu único elemento (ou campo) de geometria. A Figura 4.7 mostra um diagrama simplificado do mesmo. Futuramente, podem ser realizados estudos para utilizar mais de um elemento geometryMember em uma mesma Coleção de Feições para representar fontes de dados que possuam vários objetos geométricos associados a uma Coleção de Feições.

Algumas regras definidas para a criação do Esquema Georreferenciado Aplicação são:

- Primeiramente devem ser importados os arquivos feature.xsd e geometryAggregates.xsd da GML, para que possam ser utilizados seus elementos e tipos.
- Este esquema deve representar como seu elemento raiz uma "Coleção de Feições" do tipo "FeatureCollectionType", herdada pelo WISE do tipo GML "AbstractFeatureCollectionType" da GML. Esta coleção de feições foi criada para representar feições de um esquema local de uma fonte de dados georreferenciados. Esta "Coleção de Feições" deve possuir o mesmo nome

de uma coleção de feições da fonte de dados, e também deve dar o mesmo nome ao seu Esquema Georreferenciado Aplicação. Por exemplo, se um arquivo ESRI Shape possui o nome UsoSolo.shp, o nome da "Coleção de Feições" será UsoSolo e o nome do Esquema Georreferenciado Aplicação UsoSolo.xsd.

- Este elemento "Coleção de Feições" pode possuir um conjunto de Elementos Informativos, para representar conceitos de geodésia e cartografia associados à coleção de feições.
- Deve ser criado um elemento global "Feição" para representar feições da fonte de dados. Este elemento deve ser do tipo "ApplicationFeatureType", definido neste trabalho, ou de algum tipo com estrutura semelhante, e deve ser herdado de "AbstractFeatureType" da GML.
- "ApplicationFeatureType" deve ser definido neste esquema, para conter um elemento geométrico (georreferenciado), "geometryMember" da GML, e um conjunto de elementos representando campos, de Tipos de Dados Simples WISE, referenciando este objeto geográfico.
- Cada um destes elementos pode conter Elementos Informativos ou restritivos.

A Listagem 4.5 mostra um Esquema Convencional Aplicação referenciando dois Esquemas Georreferenciados Aplicação agregados. A Listagem 4.6 mostra um dos Esquemas Georreferenciados Aplicação agregados ao esquema da Listagem 4.5. A Figura 4.8 mostra o mesmo Esquema Georreferenciado Aplicação com uma representação de diagrama em árvore.

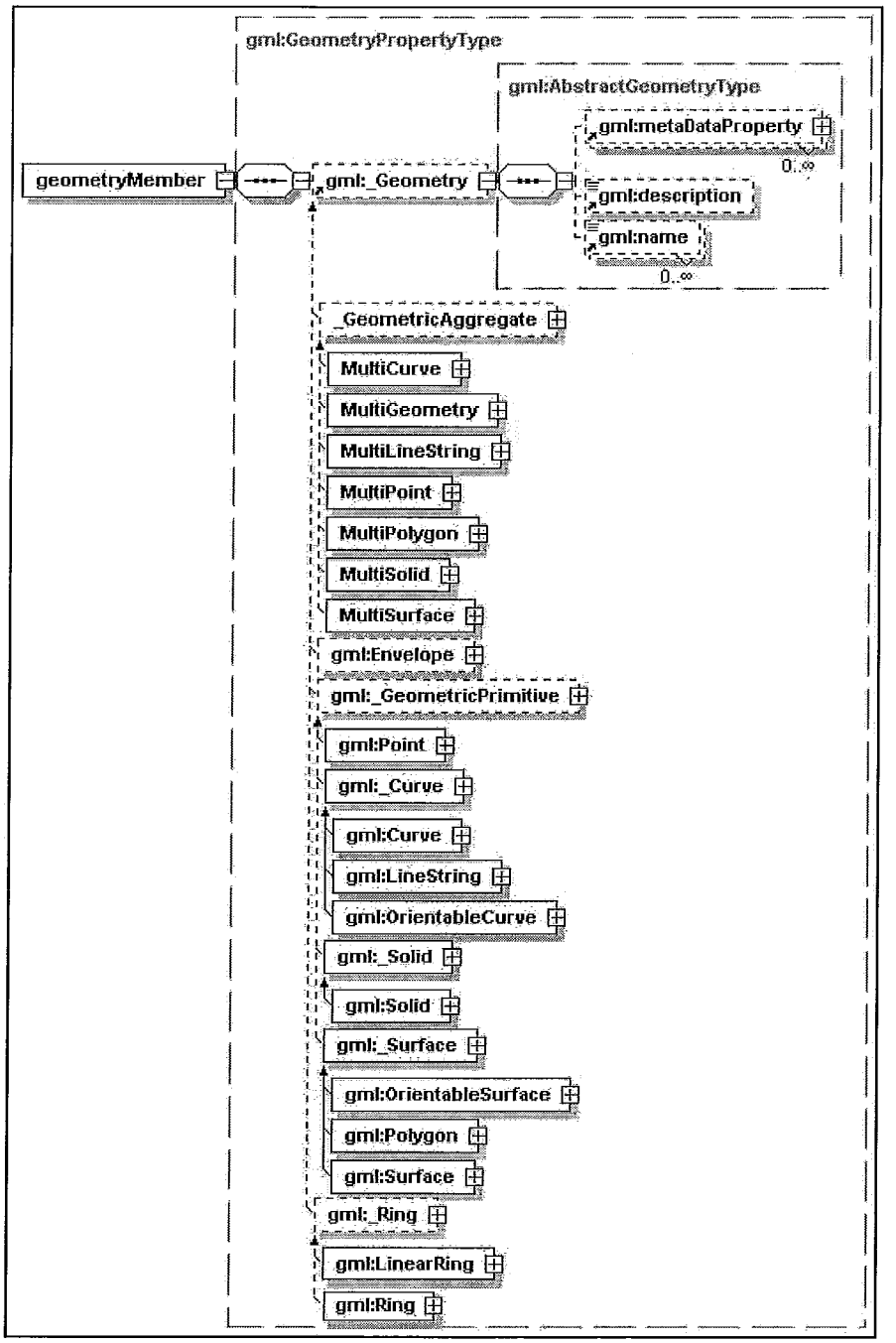


Figura 4.7 – Representação Gráfica do elemento geometryMember

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="D:\ProgsMestrado\XML_GML\Luc GML
3.0.1\base\feature.xsd"/>
    <include schemaLocation="SXMLS-Base.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <include schemaLocation="SXMLS-AppGeoRefTemplate.xsd"/>
    <include schemaLocation="SXMLS-AppGeoRefTemplate2.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="BaseData" type="wisesxmls:BaseSchemaType" substitutionGroup="wisesxmls:_BaseData">
        <annotation>
            <appinfo>
                <element name="WISE=BaseData=Source">SourceInfo</element>
                <element name="WISE=BaseData=Lineage">LineageInfo</element>
                <element name="WISE=Description">DescriptionText</element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================ -->
    <element name="DataSourceName" substitutionGroup="wisesxmls:_DataSource">
        <annotation>
            <appinfo>
                <element name="WISE=Ontology.Name">"DataSourceName", "Onto2", "Onto3"</element>
                <element name="WISE=Description">DescriptionText</element>
            </appinfo>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="wisesxmls:AbstractDataSourceType"/>
            </complexContent>
        </complexType>
    </element>
    <!--================================================================ -->
    <element name="DataSets" substitutionGroup="wisesxmls:_DataSets">
        <complexType>
            <complexContent>
                <extension base="wisesxmls:AbstractDataSetsType"/>
            </complexContent>
        </complexType>
    </element>
    <!--================================================================ -->
    <element name="SXMLS-AppGeoRefTemplate" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
    <!--================================================================ -->
    <element name="SXMLS-AppGeoRefTemplate2" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
    <!--================================================================ -->
</schema>
```

Listagem 4.5 – Exemplo de Esquema Convencional Aplicação Referenciando Dois Esquemas Georreferenciados Aplicação Agregados

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="D:\ProgsMestrado\XML_GML\Luc GML
3.0.1\base\feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="D:\ProgsMestrado\XML_GML\Luc GML
3.0.1\base\geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="SXMLS-AppGeoRefTemplate" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=FeatureCollection=Datum">DatumInfo</element>
                <element name="WISE=FeatureCollection=Ellipsoid">EllipsoidInfo</element>
                <element name="WISE=FeatureCollection=ProjectionSystem">ProjectionSystemInfo</element>
                <element name="WISE=FeatureCollection=CoordinateSystem">CoordinateSystemInfo</element>
                <element name="WISE=FeatureCollection=Scale">ScaleInfo</element>
                <element name="WISE=FeatureCollection=Source">SourceInfo</element>
                <element name="WISE=FeatureCollection=Lineage">LineageInfo</element>
                <element name="WISE=Description">DescriptionText</element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================ -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ================================================================
     complex types
     ================================================================ -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--================================================================ -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="F10" type="wisesxmls:wiseBoolean"/>
                    <element name="F11" type="wisesxmls:wiseByte">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Ontology.Name">"F11", "Onto9", "Onto10"</element>
                                <element name="WISE=Description">DescriptionText</element>
                                <element name="WISE=Field=UnitOfMeasure">UOMInfo</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="F12" type="wisesxmls:wiseDate"/>
                    <element name="F13" type="wisesxmls:wiseDateTime"/>
                    <element name="F14" type="wisesxmls:wiseDouble"/>
                    <element name="F15" type="wisesxmls:wiseFloat"/>
                    <element name="F16" type="wisesxmls:wiseInt"/>
                    <element name="F17" type="wisesxmls:wiseString"/>
                    <element name="F18" type="wisesxmls:wiseTime"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

Listagem 4.6 – Exemplo de Esquema Georreferenciado Aplicação

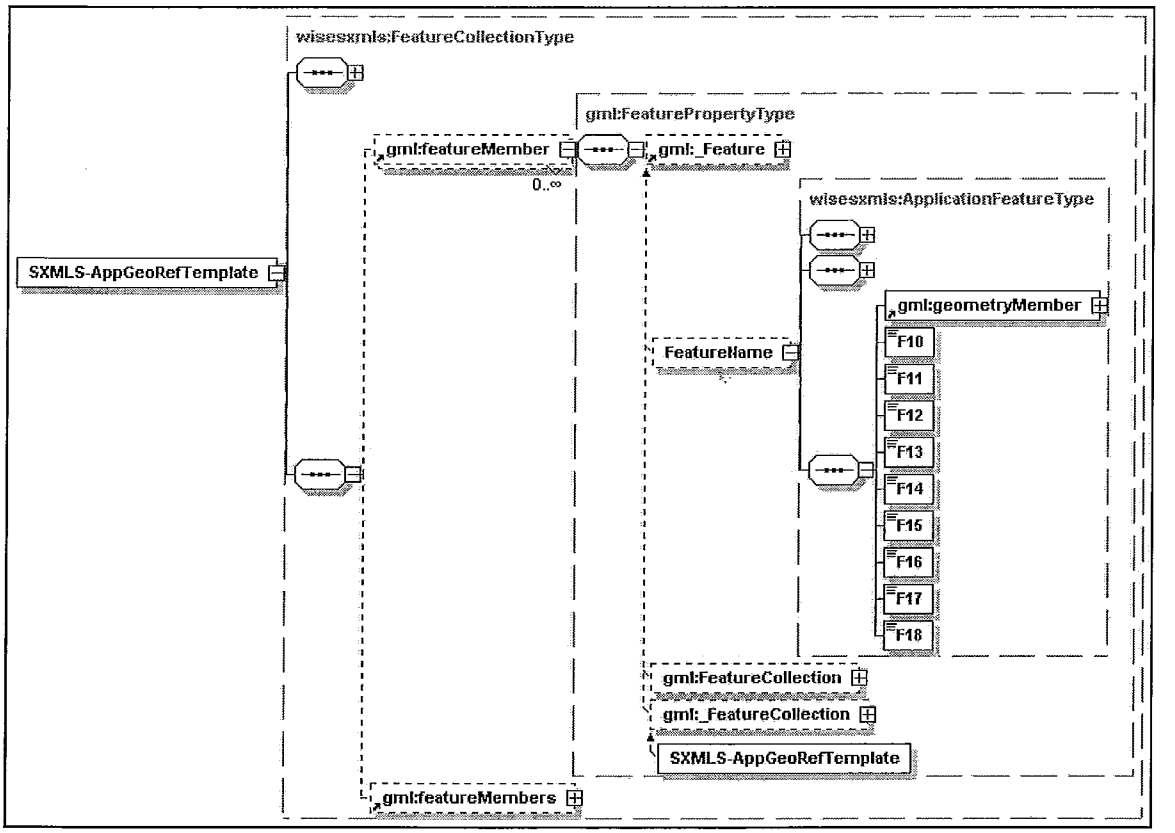


Figura 4.8 – Representação Gráfica de um Esquema Georreferenciado Aplicação

4.2.3 Esquema Integrado Base

Além dos esquemas vistos anteriormente para representar fontes de dados cadastradas no WISE, este trabalho possui ainda o Esquema Integrado Base que é um XML Schema Semântico Integrado (ou ISXMLS, de *Integrated Semantic XML Schema*) para representar a integração de mais de um SXMLS. O Esquema Integrado Base possui basicamente a mesma estrutura do Esquema Base, definido anteriormente, com uma exceção, ele não possui o elemento "Fonte de Dados". Este elemento não é necessário ao Esquema Integrado Base já que este representa a integração de "Conjuntos de Dados" ou "Coleções de Feições" de outros SXMLS com fontes de dados diferentes, bastando apenas fazer referência a estas. O Esquema Integrado Base herda características do Esquema Base como pode ser visto no esquema da Listagem 4.7 e na representação em um diagrama em árvore na Figura 4.9.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ================================================================================
     includes and imports
     ================================================================================ -->
    <include schemaLocation="SXMLS-Base.xsd"/>
    <!-- ================================================================================
     global declarations
     ================================================================================ -->
    <element name="_IntegratedData" type="wisesxmls:IntegratedSchemaType" abstract="true"/>
    <!-- ================================================================================
     complex types
     ================================================================================ -->
    <complexType name="IntegratedSchemaType">
        <sequence>
            <element ref="wisesxmls:_DataSets"/>
        </sequence>
    </complexType>
    <!--================================================================================ -->
</schema>
```

Listagem 4.7 – Esquema Integrado Base

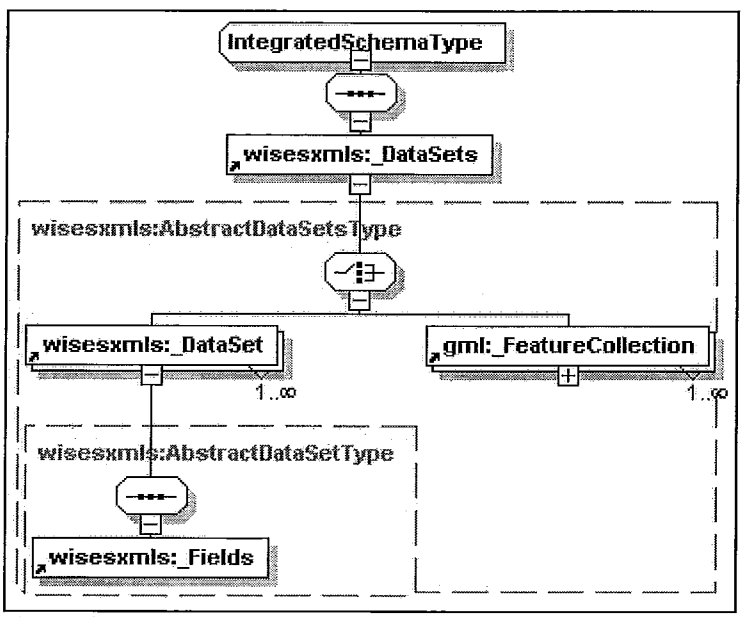


Figura 4.9 – Representação Gráfica do Esquema Integrado Base

Da mesma forma que o Esquema Base, o Esquema Integrado Base possui Esquemas Aplicação, que são o Esquema Integrado Convencional – construído a partir da integração de esquemas convencionais e georreferenciados, mas contendo somente informações convencionais, e o Esquema Integrado Georreferenciado – construído a partir da integração de esquemas convencionais e georreferenciados, porém contendo informações convencionais e georreferenciadas.

Os esquemas integrados não visam replicar as informações contidas nos seus esquemas componentes, mas necessitam fazer referências a algumas informações nestes esquemas para conseguir identificar quais metadados pertencem a cada esquema. Para

conseguir identificar que metadados provêm de cada esquema componente, bem como informar quais conversões devem ser realizadas nos dados para que estejam em um formato padrão para serem utilizados de forma integrada, foi desenvolvido um conjunto de elementos informativos, como pode ser observado em um exemplo na Listagem 4.8, para resolver tais problemas. Os elementos são:

- ComponentSchema: informa o nome de um esquema componente do esquema integrado, bem como seu apelido para ser utilizado ao longo do esquema integrado. Só pode ser usado com o elemento IntegratedData, ou seja, o elemento raiz do Esquema Integrado Base.
- SchemaRef: faz referência a um ou mais elementos de esquemas componentes. Este elemento é usado para informar que um dado elemento do esquema integrado se originou de um elemento de um esquema componente. De qual esquema veio um conjunto de dados (ou coleção de feições), ou um campo, e quais são estes, por exemplo.
- Conversion.UnitOfMeasure: é utilizado no esquema integrado para informar quais conversões de unidade de medida devem ser realizadas sobre um campo no momento da integração de dados. Este tipo de elemento só deve ser utilizado com campos.
- Conversion.Datum: Converte o Datum do esquema componente origem para o Datum do esquema integrado.
- Conversion.Ellipsoid: Converte o Elipsóide do esquema componente origem para o Elipsóide do esquema integrado.
- Conversion.ProjectionSystem: Converte o Sistema de Projeções do esquema componente origem para o Sistema de Projeções do esquema integrado.
- Conversion.CoordinateSystem: Converte o Sistema de Coordenadas do esquema componente origem para o Sistema de Coordenadas do esquema integrado.
- Conversion.Scale: Converte a Escala do esquema componente origem para o Escala do esquema integrado.

```
<annotation>
   <appinfo>
      <element name="WISE=IntegratedData=ComponentSchema">"C1", "ConventionalSchema1.xsd"</element>
      <element name="WISE=IntegratedData=ComponentSchema">"C2", "ConventionalSchema2.xsd"</element>
      <element name="WISE=SchemaRef">"C1/BaseData/DataSource/DataSets/DataSetA/Fields/F1",
"C2/BaseData/DataSource/DataSets/DataSetB/Fields/F3"</element>
      <element name="WISE=SchemaRef">"C1/BaseData/DataSource/DataSets/DataSetB"</element>
      <element name="WISE=Field=Conversion.UnitOfMeasure">
"C1/BaseData/DataSource/DataSets/DataSetA/Fields/F1", "TargetUOM"</element>
      </appinfo>
</annotation>

<annotation>
   <appinfo>
      <element name="WISE=IntegratedData=ComponentSchema">"G1", "GeoReferencedSchema1.xsd"</element>
      <element name="WISE=FeatureCollection=Conversion.Datum">
"G1/BaseData/DataSource/DataSets/FeatureCollectionA", "TargetDatum"</element>
      <element name="WISE=FeatureCollection=Conversion.Ellipsoid">
"G1/BaseData/DataSource/DataSets/FeatureCollectionA", "TargetEllipsoid"</element>
      <element name="WISE=FeatureCollection=Conversion.ProjectionSystem">
"G1/BaseData/DataSource/DataSets/FeatureCollectionA", "TargetProjectionSystem"</element>
      <element name="WISE=FeatureCollection=Conversion.CoordinateSystem">
"G1/BaseData/DataSource/DataSets/FeatureCollectionA", "TargetCoordinateSystem"</element>
      <element name="WISE=FeatureCollection=Conversion.Scale">
"G1/BaseData/DataSource/DataSets/FeatureCollectionA", "TargetScale"</element>
      </appinfo>
</annotation>
```

Listagem 4.8 – Alguns Elementos Informativos do XML Schema Semântico Integrado

4.3 Organização da Arquitetura

A arquitetura do WISE é bastante abrangente, possuindo diversos componentes. Estes componentes foram distribuídos de acordo com suas funcionalidades em quatro camadas: Camada Cliente, Camada de Integração de Dados, Camada de Tradutores, e Camada de Acesso a Dados. Estas camadas estão representadas na Figura 4.10 e serão descritas abaixo. Porém a descrição desses componentes do WISE é apresentada na seção 4.5.

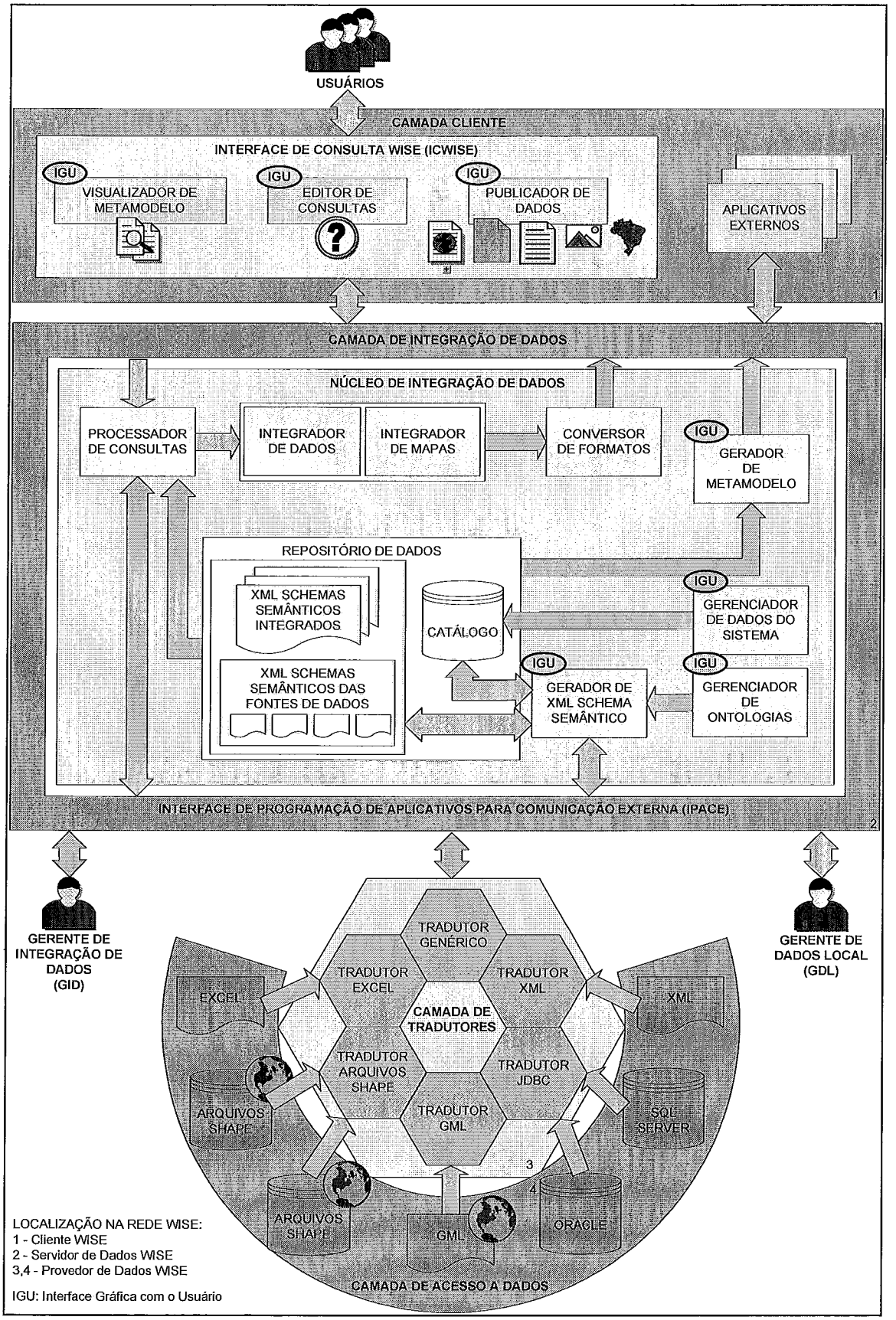


Figura 4.10 – Arquitetura do WISE

4.3.1 Camada Cliente

É a camada definida pelos aplicativos cliente do WISE, inclusive pela Interface de Consulta WISE (ICWISE). O acesso ao WISE pelos aplicativos clientes se dá através da comunicação com a Interface de Programação de Aplicativos para Comunicação Externa (IPACE) da Camada de Integração de Dados, que é uma interface que possui funcionalidades para prover acesso a aplicativos clientes.

O WISE possui a sua própria interface gráfica de consulta chamada Interface de Consulta WISE (ICWISE), que é o aplicativo cliente padrão para realizar consultas interativas pelo Usuário de Dados (UD). A ICWISE possui três componentes, o Visualizador de Metamodelo, o Editor de Consultas e o Publicador de Dados.

4.3.1.1 Visualizador de Metamodelo

A interface gráfica da ICWISE, apresenta no Visualizador de Metamodelo uma ferramenta capaz de representar o Metamodelo dos metadados armazenados no WISE acerca das suas Fontes de Dados.

A partir das informações visualizadas no Metamodelo o usuário é capaz de entender a distribuição, estrutura, semântica e qualidade dos dados, entre outras informações. A distribuição dos dados é transparente ao usuário, ou seja, ele não necessita ter conhecimento como os dados estão armazenados fisicamente.

4.3.1.2 Editor de Consultas

Com base no Metamodelo, o usuário poderá introduzir suas consultas através do Editor de Consultas da ICWISE, que fornece uma interface com componentes visuais, onde o usuário poderá selecionar as informações necessárias em cada componente e finalmente disparar a consulta para ser processada pelo WISE.

4.3.1.3 Publicador de Dados

Após as consultas terem sido processadas pelo WISE, o seu resultado é retornado e exibido no Publicador de Dados, uma interface gráfica capaz de apresentar vários formatos de dados, e permitir manipulação pelo usuário, possibilitando ainda que os dados sejam gravados localmente para utilização posterior.

4.3.2 Camada de Integração de Dados

Esta camada é composta logicamente da Interface de Programação de Aplicativos para Comunicação Externa (IPACE) e do Núcleo de Integração de Dados.

A IPACE é a Interface de Programação de Aplicativos para comunicação com o WISE. É através desta interface que qualquer aplicativo de software externo se comunica com o núcleo do WISE, incluindo os componentes da Camada Cliente que pertencem ao WISE, e da Camada de Tradutores. Muitos serviços providos pelas classes do Núcleo de Integração de Dados são acessíveis através desta interface.

O cerne da arquitetura do WISE é o Núcleo de Integração de Dados, onde são processadas as consultas recebidas de aplicativos externos, cujos resultados obtidos das Fontes de Dados cadastradas, são integrados, e a seguir convertidos para determinados formatos de arquivo e retornados para o realizador da consulta.

4.3.3 Camada de Tradutores

A Camada de Tradutores é composta basicamente de Tradutores, que são componentes de software que fornecem ao WISE serviços de acesso aos dados provenientes das Fontes de Dados cadastradas neste. É o meio pelo qual o WISE consegue acessar dados, oriundos de Fontes de Dados heterogêneas e distribuídas, de maneira uniforme.

4.3.4 Camada de Acesso a Dados

É definida por um conjunto de Provedores de Dados, que são recursos computacionais capazes de armazenar uma ou mais Fontes de Dados heterogêneas (XML, Excel, Shape, etc) entre si, e disponibilizar ao menos uma para a utilização pelo WISE. O WISE reconhece a localização de um Provedor de Dados através de sua URL (*Uniform Resource Locator*).

Uma Fonte de Dados é um repositório de dados homogêneos, armazenado em um único Provedor de Dados, que é acessada por um Tradutor utilizando uma URL e parâmetros armazenados no Catálogo do WISE.

Cada Fonte de Dados pertence a um Tipo de Fonte de Dados que especifica se os dados são georreferenciados ou convencionais, além de especificar o tipo dos dados (Relacional, XML, Objeto-Relacional, Arquivo Texto, etc.).

Um Tradutor criado para um determinado Tipo de Fonte de Dados pode ser utilizado por qualquer Fonte de Dados que esteja disponível no WISE, para aquele tipo.

4.4 Papéis e Funcionamento do WISE

A presente seção apresenta primeiramente os papéis dos usuários ao utilizarem o WISE, passando por uma descrição geral do funcionamento do WISE, chegando a descrição da interligação de vários servidores e provedores de dados através da Rede WISE.

4.4.1 Papéis de Usuários do WISE

O WISE possui usuários com funções classificadas de acordo com determinados papéis, sendo que um usuário pode realizar mais de um papel ao mesmo tempo. Cada papel que um usuário do WISE pode assumir está descrito abaixo.

4.4.1.1 Gerente de Integração de Dados (GID)

Para que o Núcleo de Integração de Dados funcione é necessário uma pessoa que seja responsável pela sua manutenção e possa realizar interações periódicas com este. A esta pessoa, dá-se o nome de Gerente de Integração de Dados (GID), cabendo a ela as atividades de integração de esquemas, criação de Tradutores, inclusão de novos formatos de arquivos de saída e controle do acesso a dados, entre outras atividades.

4.4.1.2 Gerente de Dados Local (GDL)

O Gerente de Dados Local (GDL) é uma pessoa que gerencia uma ou mais Fontes de Dados locais em um Provedor de Dados. Este escolhe quais fontes de dados deseja compartilhar, efetua o cadastro destas no WISE e é o responsável por informar a sua semântica quando necessário. Também pode auxiliar o GID em algumas de suas atividades. O GDL também é chamado de Catalogador, pois é o responsável por catalogar Fontes de Dados no WISE.

4.4.1.3 Usuário de Dados (UD)

Pessoa capaz de realizar consultas sobre uma interface de consulta, podendo interagir com a ICWISE.

4.4.1.4 Usuário de Dados Avançado (UDA)

Um Usuário de Dados Avançado é um UD que necessita de maior interação com o WISE e deseja criar seus próprios XML Schemas Semânticos Integrados.

4.4.2 Funcionamento Geral do WISE

Esta seção descreve o funcionamento geral do WISE, como alguns componentes do WISE interagem e algumas características que não são descritas na seção 4.4, Componentes da Arquitetura.

4.4.2.1 Núcleo de Integração de Dados

O Núcleo de Integração de Dados funciona dentro de um servidor de dados, o qual é chamado Servidor de Dados WISE. Para que um servidor possa disponibilizar dados para aplicações é necessário que haja Fontes de Dados cadastradas neste, e estas Fontes de Dados residem em Provedores de Dados WISE, sendo que o servidor também é um Provedor de Dados. A seção 4.4.3 descreve estes recursos computacionais em mais detalhes.

Com o Servidor WISE funcionando começa-se a cadastrar Fontes de Dados neste através da interface do Gerenciador de Dados do Sistema. Ao se cadastrar uma Fonte de Dados, são informados entre outras coisas, onde está localizada, qual Entidade que a está cadastrando e o seu responsável, Catalogador ou GDL.

Quando uma Fonte de Dados é cadastrada as suas referências são armazenadas no Catálogo do WISE, junto com o Tipo de Fonte de Dados no qual se enquadra e o nome do Tradutor - Camada de Tradutores - para este tipo. Cada Tipo de Fonte de Dados deve possuir um Tradutor específico para que o WISE consiga acessar a Fonte de Dados daquele tipo.

Uma Fonte de Dados é acessada inicialmente por uma ferramenta do WISE chamada de Gerador de XML Schema Semântico utilizada pelo GID, que nesta fase vai gerar automaticamente um XML Schema Semântico baseado no conteúdo do esquema local contido nesta Fonte de Dados, sendo desta maneira uma visão deste esquema. Desta forma vai sendo gerado o XML Schema Semântico de cada Fonte de Dados,

baseado no MDC do WISE, o que tornará possível, em uma fase posterior, cada Fonte de Dados ser visualizada da mesma forma pelo WISE.

Com o XML Schema Semântico gerado pode-se ainda adicionar mais informações a este também usando o Gerador de XML Schema Semântico, que é capaz de editar tais esquemas para serem adicionadas mais informações, entre elas ontologias do domínio da aplicação adquiridas através do Gerenciador de Ontologias. Este processo pode ser realizado em conjunto pelo GID e pelo GDL da Fonte de Dados que detém a semântica destes dados.

O Gerador de XML Schema Semântico ainda é capaz de realizar a integração semântica de XML Schemas Semânticos com interação do GID e produzir desta forma um XML Schema Semântico Integrado. Este processo é normalmente realizado pelo GID, podendo algumas vezes ser realizado por UDAs. A tarefa de integração de esquemas cria desta forma várias visões dos esquemas originais das Fontes de Dados de acordo com a necessidade de cada grupo de usuários.

Os XML Schemas Semânticos são armazenados, junto com o Catálogo, em um Repositório de Dados. Assim que existir algum XML Schema Semântico neste repositório o Gerador de Metamodelo pode ser executado, gerando um Metamodelo atualizado de todos esquemas que estão armazenados no Repositório de Dados e fica disponível para ser acessado pelos aplicativos Clientes WISE, inclusive pelo Visualizador de Metamodelos da ICWISE.

Como já foi visto na Camada Cliente, assim que um usuário verifica como está o Metamodelo este pode entrar com consultas no Editor de Consultas da ICWISE. As consultas realizadas no Editor de Consultas, e em outros aplicativos, são passadas para o Processador de Consultas que as trata e decompõe em subconsultas de acordo com informações dos XML Schemas Semânticos, dos seus Tradutores e ainda algumas informações do Catálogo, conforme apresentado na Figura 4.10.

Assim que o Processador de Consultas termina de analisar as consultas, as suas subconsultas são enviadas para os Tradutores das respectivas Fontes de Dados envolvidas na consulta, que por sua vez acessam estas e retornam os resultados das subconsultas para o Processador de Consultas, e este passa estas informações para o Integrador de Dados e Mapas.

Como as consultas solicitadas podem requerer além de dados convencionais, dados georreferenciados também, foi dividido o integrador em dois integradores

lógicos, chamados: Integrador de Dados e Integrador de Mapas, sendo a função do Integrador de Dados a de integrar dados convencionais e a do Integrador de Mapas de integrar dados georreferenciados.

O Integrador de Dados e Mapas tem o trabalho de integrar as informações a ele passadas e a seguir executar o Conversor de Formatos, que converte para o formato de arquivo de saída selecionado pelo usuário. Finalmente o Conversor de Formatos passa o resultado da consulta para o Cliente WISE que o solicitou.

4.4.2.2 Provedores de Dados, Fontes de Dados

Os Provedores de Dados disponíveis para o WISE com os seus respectivos Catalogadores (GDLs) possuem suas informações armazenadas no Catálogo do WISE. O WISE somente acessa as Fontes de Dados que estão catalogadas nestes Provedores de Dados. Uma Fonte de Dados pertence a um determinado Tipo de Fonte de Dados, que podem se dividir em dois grupos distintos, o dos dados convencionais e dos dados georreferenciados. As Fontes de Dados possuem XML Schema Semânticos, sendo que cada esquema pode representar uma Fonte de Dados inteira com os seus diversos Conjuntos de Dados, um subconjunto destes, ou somente um único Conjunto de Dados.

4.4.2.3 Tradutores

Cada Tipo de Fonte de Dados deve possuir um Tradutor para que o sistema consiga acessar os seus dados. Os Tradutores servem como uma ponte entre as Fontes de Dados e o Núcleo de Integração de Dados, assim o Catálogo do WISE também armazena a descrição do Tradutor relacionado a um dado Tipo de Fonte de Dados.

Algumas vezes é necessário passar informações, parâmetros, para os Tradutores a respeito das Fontes de Dados no momento da sua conexão, como por exemplo: no caso de um banco de dados relacional, freqüentemente é requerido que sejam informados nome, senha, serviço do banco de dados, e outros parâmetros para conexão com este. Como o WISE possui várias Fontes de Dados heterogêneas, e as informações necessárias diferem entre elas, foi criada uma tabela (Parâmetros) no Catálogo para armazenar os parâmetros que devem ser passados para um Tradutor quando for necessário o acesso a uma Fonte de Dados. Os parâmetros necessários a uma dada Fonte de Dados devem ser fornecidos pelo GDL no momento da catalogação da mesma. Uma das características mais importantes dos parâmetros é a de passar para o Tradutor a

informação do local exato onde está uma dada Fonte de Dados dentro do Provedor de Dados, independente do seu tipo.

4.4.2.4 Controle de Acesso

Os dados constantes em um Provedor de Dados que podem ser compartilhados possuem níveis de controle de acesso que podem ser aplicados a um Provedor de Dados inteiro, a uma Fonte de Dados, ou a um XML Schema Semântico. Este controle de acesso é configurado para grupos de usuários. Desta forma podemos compartilhar partes de dados e nos assegurar que um grupo de usuários só vai acessar os dados a ele destinados. Os dados podem ser compartilhados por um determinado período de tempo também.

Como este sistema poderá consultar, integrar e publicar dados, mas não poderá nem criá-los e nem alterá-los nas suas respectivas Fontes de Dados, pois são utilizadas visões não atualizáveis, os níveis de controle de acesso se limitam a três. O primeiro nível não permite acesso algum e é chamado de "Invisível", o segundo nível permite somente consultar metadados e tem nome "Somente Metadados", e o último nível, "Consulta", permite que sejam realizadas consultas onde este for aplicado. É disponibilizado o nível "Somente Metadados" para que um usuário que venha a se interessar por estes metadados possa requisitar ou negociar com o seu GDL a disponibilização dos dados dos mesmos.

Já que existe uma relação de interdependência desde um Provedor de Dados a um XML Schema Semântico, onde o XML Schema Semântico é o mais dependente e o Provedor de Dados não possui dependência, os níveis de acesso em um elemento mais dependente só podem ser de menor acessibilidade que nos elementos menos dependentes. Como exemplo, um Provedor de Dados com nível de acesso "Consulta" pode conter Fontes de Dados ou XML Schema Semânticos invisíveis, mas um Provedor de Dados com nível de Acesso "Invisível" não poderia conter, por exemplo, um XML Schema Semântico com nível de acesso "Consulta".

4.4.2.5 Entidades, Usuários e Grupos de Usuários

No WISE o acesso dados, bem como a sua disponibilização é realizada por Entidades. Uma Entidade é uma pessoa jurídica que pode conter vários usuários, ou uma pessoa física que pode conter um único usuário. A Entidade pode disponibilizar ou

acessar dados. Se esta Entidade disponibiliza dados, possuindo no mínimo um Provedor de Dados, ela tem que possuir um Catalogador (GDL), que é o usuário responsável pela sua disponibilização e manutenção.

Os usuários estão distribuídos em grupos de usuários, que podem conter além de usuários, outros grupos de usuários.

4.4.3 Rede WISE

A Rede WISE é uma forma do WISE trocar dados entre várias instituições distribuídas geograficamente. Esta rede usa as camadas, apresentadas na Figura 4.10, distribuídas em algumas regras como pode ser visto abaixo.

Na Rede WISE, um recurso computacional pode ter um ou mais de três papéis: Servidor de Dados WISE (ou simplesmente Servidor de Dados), Provedor de Dados WISE (ou simplesmente Provedor de Dados) e Cliente WISE, como mostrado na Figura 4.11. Os servidores são capazes de servir dados heterogêneos e distribuídos para seus clientes, coletados nos diversos Provedores de Dados que estão registrados em seus catálogos. O Servidor de Dados faz o trabalho principal de integração de dados e mapas assim como outras tarefas do Núcleo de Integração de Dados como mostrado na Figura 4.10 (Camada de Integração de Dados). Os Provedores de Dados armazenam as Fontes de Dados heterogêneas e seus respectivos Tradutores (Camada de Acesso a Dados e Camada de Tradutores da Figura 4.10) e fornecem dados para os Servidores de Dados. Os Clientes WISE realizam consultas sobre os Servidores de Dados usando a ICWISE, são responsáveis pela Camada Cliente da Figura 4.10. No WISE um Servidor de Dados também é um Provedor de Dados.

Além de se comunicar com Provedores de Dados WISE para acessar seus dados, os Servidores de Dados WISE podem também se comunicar com outros Servidores de Dados para compartilhar dados, formando uma rede ponto-a-ponto (P2P) híbrida. Esta rede P2P é importante para o WISE pois, entre outras coisas: a) um Cliente WISE pode acessar uma grande quantidade de dados heterogêneos diversificados distribuídos geograficamente em várias máquinas, em contraste com um único Servidor de Dados com vários Provedores de Dados, b) processamento paralelo dos dados, c) contingência de Servidores de Dados e d) escalabilidade.

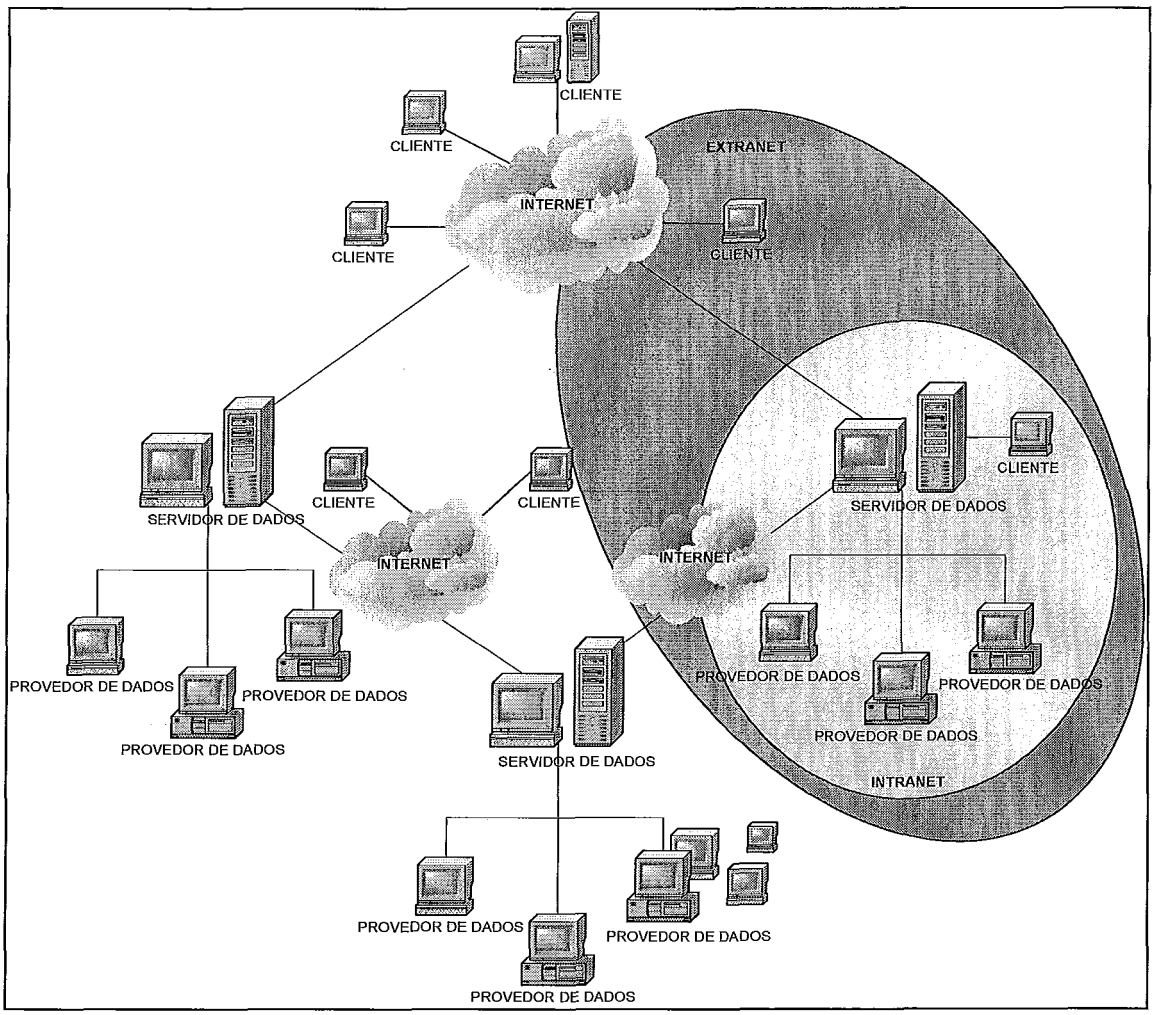


Figura 4.11 – Exemplo de Rede WISE

4.4.3.1 Registro e Troca de Dados entre Servidores de Dados WISE

O protocolo para registrar e compartilhar dados entre Servidores de Dados WISE consiste nas seguintes fases:

Fase 1 – Associação de Servidores de Dados

Uma instituição irá tornar disponível, em um momento inicial M1 (Figura 4.12a), um Servidor de Dados WISE SD1 na Internet, e este servidor pode conter, ou não, Provedores de Dados WISE registrados no seu Catálogo.

No momento M2 (Figura 4.12b), uma outra instituição emerge com um outro servidor SD2 na Internet e requisita (momento M3) compartilhar dados com SD1 (Figura 4.12c). O WISE provê interfaces que uma instituição usa para requisitar participar em um compartilhamento de dados entre servidores.

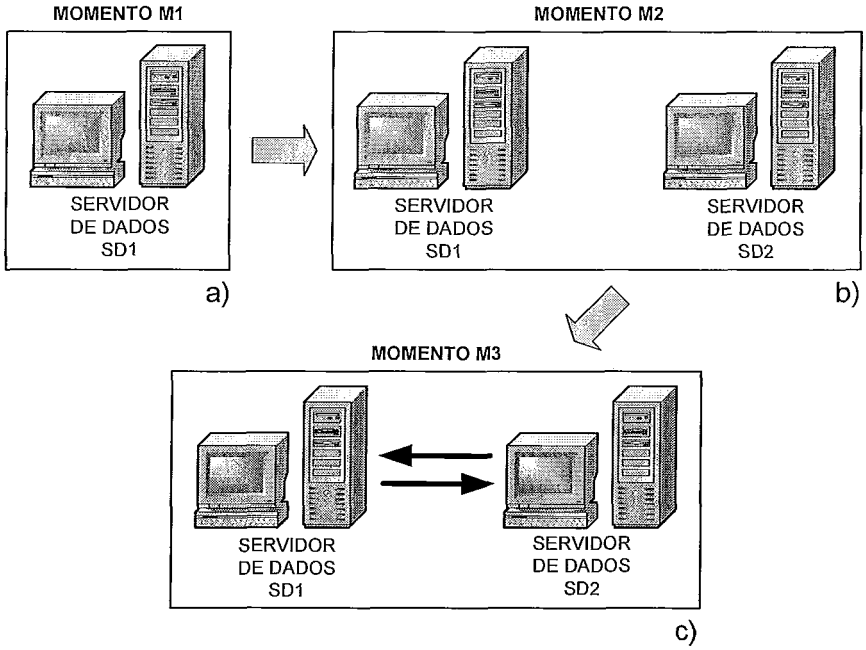


Figura 4.12 – Associação de Servidores

As instituições relacionadas com estes servidores podem concordar ou discordar em compartilhar dados com as outras, já que uma instituição pode usar o WISE sozinha, e não necessita compartilhar dados com outras instituições.

Quando um servidor SD1 aceita compartilhar dados, os servidores SD1 e SD2 trocam informações necessárias para identificar e localizar o outro, bem como informações sobre quais dados desejam compartilhar, como pode ser visto na fase 2.

Fase 2 – Transmissão de Endereço e Metamodelo

Um servidor envia um endereço (URL) para um outro, para ser identificado unicamente, e para ser localizado por este; junto com esta informação vai o nome e a descrição do servidor.

Cada servidor deve mandar um Metamodelo de dados para o outro (Figura 4.13), contendo informações sobre quais dados serão compartilhados. Quando as informações neste Metamodelo são atualizadas, um servidor precisa enviar novamente este Metamodelo para o outro.

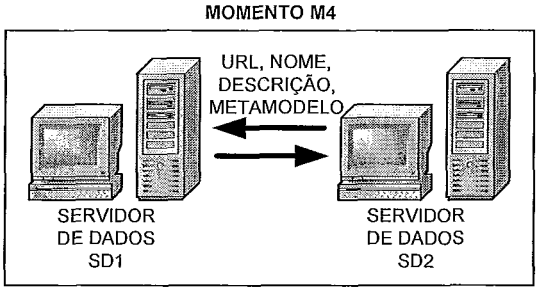


Figura 4.13 – Dois Servidores compartilhando dados

Fase 3 – Incluindo mais Servidores WISE em uma Rede

O processo de inclusão de um novo Servidor de Dados WISE em uma rede com mais de um Servidor de Dados WISE (Figura 4.14a) é similar ao processo da fase 1. A diferença é que quando o servidor SD1 aceita compartilhar dados com o novo servidor SD3, além de o primeiro passar a sua URL, nome, descrição e Metamodelo, aquele também passa a URL, nome e descrição de todos os outros servidores que estão na mesma rede, para que o novo servidor SD3 possa acessa-los e solicitar o compartilhamento de dados (Figura 4.14b).

Quando o novo servidor SD3 envia as suas informações (URL, nome, descrição e Metamodelo) para o servidor SD1, este pega estas informações e atualiza toda a rede com a URL, nome e descrição do novo membro da rede (SD3).

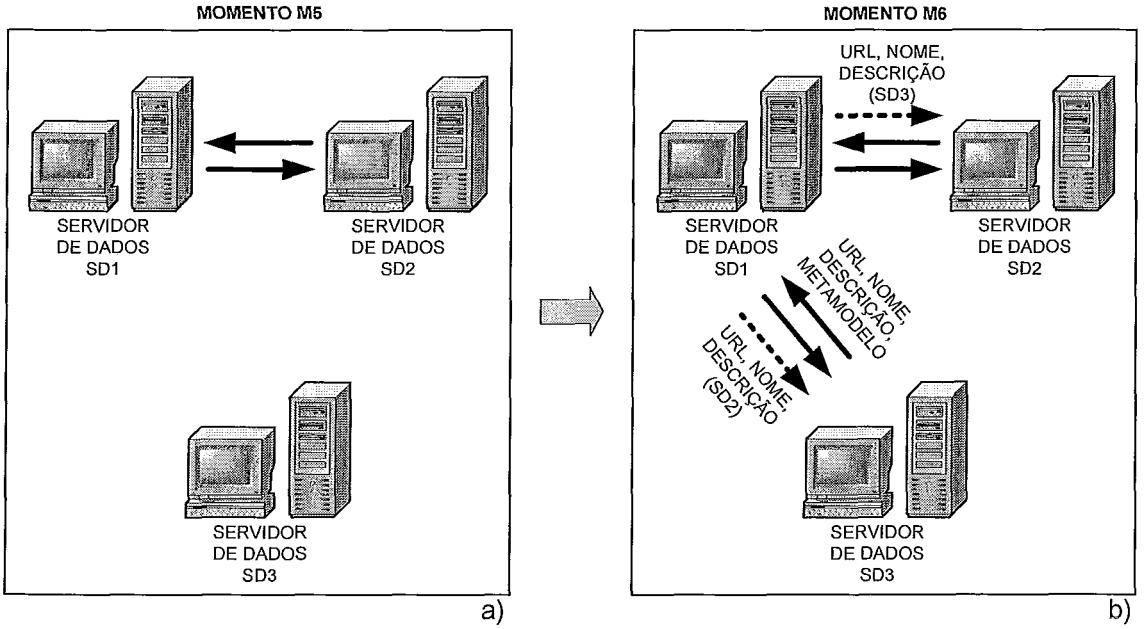


Figura 4.14 – Incluindo mais um servidor em uma rede com mais de um servidor

Fase 4 – Transmissão de Dados

Um servidor SD1 com URL e Metamodelo de um servidor SD2, poderá consultar os dados deste servidor, de forma que o servidor SD2 irá decompor a consulta e repassará para os seus Provedores de Dados, que retornarão os dados para este servidor, que por sua vez enviará o resultado para o servidor SD1, que originou a consulta.

Ao receber uma consulta o servidor SD2 sempre verifica se o servidor de origem (neste caso SD1), que está realizando a consulta, é algum servidor que está cadastrado em sua base de dados. Este procedimento mínimo de segurança é necessário, pois servidores não cadastrados podem tentar acessar dados que não foram disponibilizados para eles.

Fase 5 – Exclusão de Servidores de Dados WISE

Se um servidor de dados tiver que ser excluído, este deverá informar aos outros servidores de tal fato através de uma atualização de suas informações.

Se um servidor ficar inoperante por um determinado período de tempo, definido nos servidores nos quais está cadastrado, este também poderá ser excluído.

Finalmente, se o acordo entre as partes terminar poderá haver exclusão de servidores.

4.4.3.2 Clientes Consultando Servidores de Dados WISE

Um Cliente WISE pode se registrar em um ou mais Servidores de Dados WISE. Os Clientes estarão assim aptos a realizarem consultas diretamente sobre vários servidores em acesso P2P. O Cliente realiza a consulta nos servidores desejados, e estes servidores processam a consulta e acessam seus Provedores de Dados envolvidos na consulta, integram dados, e transmitem o resultado para o Cliente.

Um Cliente WISE pode também realizar consultas sobre outros Servidores de Dados através de um único servidor. Como o Servidor de Dados suporta o mesmo Metamodelo de dados que os outros Servidores de Dados trocando informações com o primeiro, os Clientes deste Servidor de Dados podem consultar os outros servidores através dele. Inicialmente, o Cliente realiza a consulta sobre o seu Servidor de Dados, e este último acessa seus Provedores de Dados, bem como outros Servidores de Dados que, neste caso, também estão operando como Provedores de Dados para o primeiro.

Quando um Cliente efetua uma consulta que requer integração de dados de outros Servidores de Dados, o primeiro Servidor de Dados acessa estes dados, realiza a integração e envia os resultados para o Cliente. Quando não é uma consulta para integrar dados, o primeiro servidor verifica quais servidores possuem os dados desejados no seu Metamodelo, e passa o controle para os servidores envolvidos na consulta se comunicarem diretamente com o Cliente.

4.5 Componentes da Arquitetura

Nesta seção serão apresentados diversos componentes da arquitetura do WISE, que foram vistos brevemente nas seções anteriores. Estes componentes estão ordenados no presente texto de forma que a sua leitura seja mais fácil e de certa forma como estes componentes são executados no WISE.

4.5.1 Visualizador de Metamodelo

Esta ferramenta é capaz de informar ao usuário sobre a distribuição, origem, estrutura, semântica e qualidade dos dados, entre outras informações. É o modo como o usuário é capaz de visualizar que informações o sistema tem a lhe oferecer.

Assim que um usuário consegue se conectar a ICWISE com a validação de suas informações, esta interface executa um método chamado getAccessControlInfo para adquirir as informações de Controle de Acesso para este usuário. Logo em seguida o Visualizador de Metamodelos através do método viewMetamodel da classe MetamodelViewer, Figura 4.15, executa um procedimento de requisição de um Metamodelo atualizado à classe MetamodelGenerator. Esta por sua vez retorna o Metamodelo para àquela, que ao recebê-lo aplica as permissões de controle de acesso, já adquiridas pela ICWISE, necessárias ao usuário corrente através do método applyAccessControl desta mesma classe e, cria deste modo uma visão do Metamodelo para o usuário de acordo com as suas permissões de controle de acesso e o armazena como o atributo MetamodelUserView.

O Visualizador de Metamodelos com as informações do Metamodelo podem ser visualizadas através de uma interface gráfica que torna possível a sua exibição e manipulação pelo usuário. Como a exibição destas informações é sensível ao usuário, estando de acordo com o controle de acesso aplicado a cada usuário através dos grupos

de usuários, aquele não tem acesso a informações que não sejam de seu interesse ou que não tenha permissão para visualizar.

Após o usuário ter verificado qual é a estrutura dos metadados que o interessam, este já pode realizar consultas sobre estas informações usando o Editor de Consultas da ICWISE.

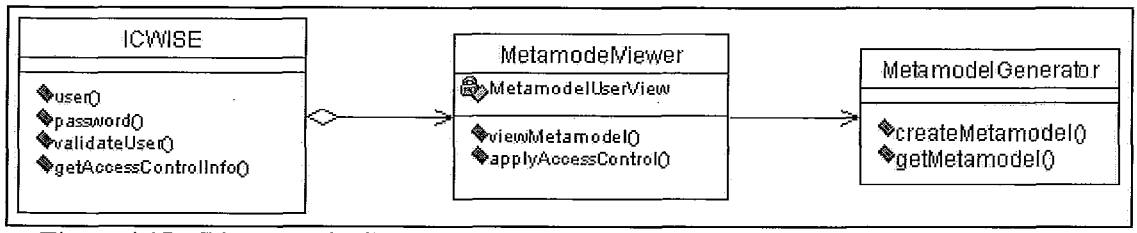


Figura 4.15 – Diagrama de Classes simplificado do MetamodelViewer e classes relacionadas

4.5.2 Editor de Consultas

O Editor de Consultas possui uma interface para que sejam introduzidas consultas através de componentes visuais, onde o usuário poderá selecionar as informações necessárias em cada componente e, finalmente, disparar a consulta para ser processada. Esta é uma interface simples que abstraí os detalhes de linguagens de consultas como SQL ou XQuery, facilitando o trabalho de usuários leigos.

Além da seleção de elementos da consulta através de componentes visuais, o usuário pode ainda selecionar itens no Visualizador de Metamodelo, facilitando o seu trabalho na seleção e busca de metadados, e adicioná-los ao Editor de Consultas.

O Editor de Consultas através da classe que o representa, QueryEditor, executa a consulta definida usando o método executeQuery, Figura 4.16. Este método passará o controle para o Processador de Consultas (classe QueryProcessor) usando o método parseQuery, onde será processada, e a seguir passará por vários componentes do Núcleo de Integração de Dados até o momento de ser retornado o resultado que será visualizado no Publicador de Dados.

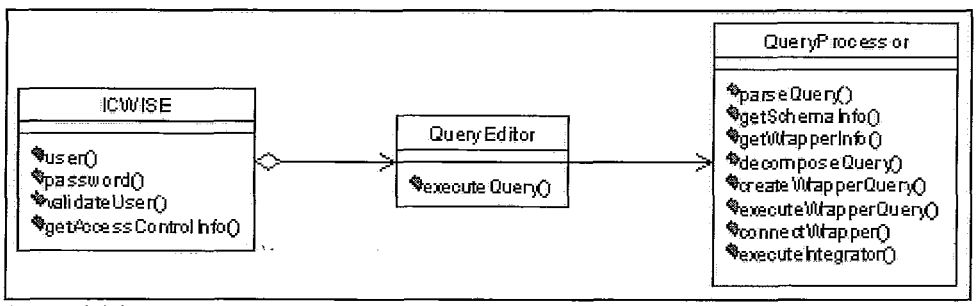


Figura 4.16 – Diagrama de Classes simplificado do QueryEditor e classes relacionadas

O formato de consulta usado internamente pelo Editor de Consultas e pelo Processador de Consultas é um subconjunto da linguagem de consulta SQL, possuindo somente elementos para realizar seleção e projeção de dados convencionais de forma simplificada. Esta consulta é armazenada em um objeto da classe WISEQuery, Figura 4.17, que armazena a consulta temporariamente. A porção que trata de objetos espaciais dos dados georreferenciados é separada em um objeto da classe GeoRefObj que é um agregado da classe WISEQuery, e em conjunto tornam possível à realização de consultas sobre dados convencionais e georreferenciados de forma simplificada. Lembrando que estas classes podem ser herdadas para que sejam agregadas novas características, tanto convencionais quanto georreferenciadas.

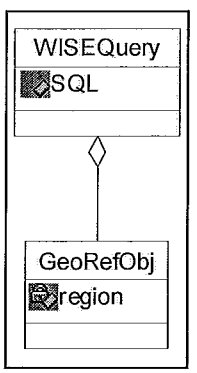


Figura 4.17 – Diagrama de Classes simplificado da WISEQuery e GeoRefObj

4.5.3 Publicador de Dados

O Publicador de Dados é a ferramenta que possibilita uma visualização dos dados, resultantes da consulta executada no Editor de Consultas, de acordo com o formato de saída que recebe do Conversor de Formatos.

O Publicador de Dados possui uma interface gráfica que pode ser usada de acordo com o tipo de formato de saída, pois utiliza uma classe, GenericForm, para o formulário gráfico genérico que pode ser herdada para suportar novos formatos de arquivos de saída. Ao possuir uma classe de formulário específica para representar um determinado formato de saída, a classe DataPublisher, Figura 4.18, pode mostrar tal formulário usando o método showForm e em seguida publicar os dados utilizando o método publish.

Além dos dados poderem ser visualizados pelo usuário de acordo com o formato de arquivo de saída, o Publicador de Dados pode ainda gravar tais dados para utilização posterior com processamento e manipulação locais, diferentemente de outras

arquiteturas (SOROKINE & MERZLIAKOVA, 1998; ZASLAVSKY et al., 2000) que permitem somente a visualização dos dados e algumas vezes só podem gravar imagens e nenhum outro formato de arquivo diferente, não proporcionando ao usuário final muita funcionalidade.

Desta forma, assim como o Conversor de Formatos pode ser estendido, o Publicador de Dados também tem classes que podem ser especializadas para que sejam manipulados os formatos de arquivos, tornando esta arquitetura bem versátil e tentando satisfazer ao máximo as necessidades de informações do usuário, bem como suportar uma diversidade de formatos de arquivos que este pudesse pretender utilizar.

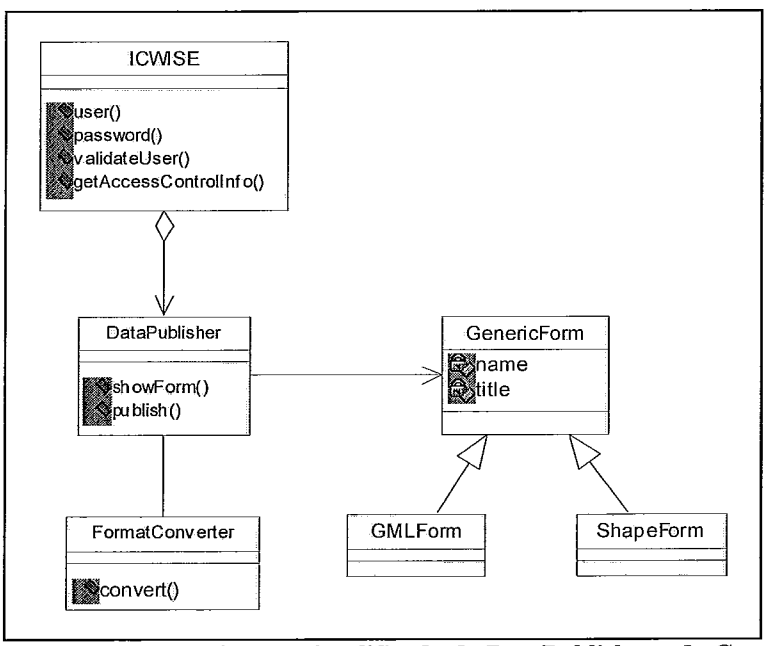


Figura 4.18 – Diagrama de Classes simplificado do DataPublisher e do GenericForm

4.5.4 Gerenciador de Dados do Sistema e Catálogo

Estes componentes da arquitetura trabalham com informações de configurações do sistema, informações dos Provedores de Dados e com armazenamento de SXMLSs em geral.

O Gerenciador de Dados do Sistema é uma interface gráfica para gerenciar as informações relacionadas ao sistema, como o cadastro de provedores de dados, fontes de dados, os tipos de Tradutores disponíveis, grupos de usuários, controle de acesso, etc. Essas informações encontram-se armazenadas no Catálogo WISE, que é o repositório de dados do WISE, onde são utilizadas por diversas partes do sistema.

O Catálogo armazena informações sobre o sistema WISE, catalogadas utilizando o Gerenciador de Dados do Sistema e também informações extraídas das fontes de

dados como os SXMLSs destas e os ISXMLSs, e agregando as funcionalidades do Repositório de Dados.

4.5.5 Tradutores

Os Tradutores (Wrappers) da arquitetura fornecem serviços para acessar Fontes de Dados heterogêneas e distribuídas de maneira uniforme. Os Tradutores são capazes de capturar os esquemas e de realizar consultas sob estas fontes de dados de tal forma que outras camadas do WISE não necessitam possuir conhecimento de como estes foram implementados. Um Tradutor deve ser construído para cada novo Tipo de Fontes de Dados registrado no WISE, pois este requer que haja um Tradutor específico para tratar cada um destes tipos (SGBD Relacional, documento XML, arquivo texto, etc), já que estes Tradutores vão deter o conhecimento de como estes novos tipos devem ser acessados e consultados. Novos Tradutores podem ser criados pelo GID ou pelos GDLs de acordo com a demanda do sistema.

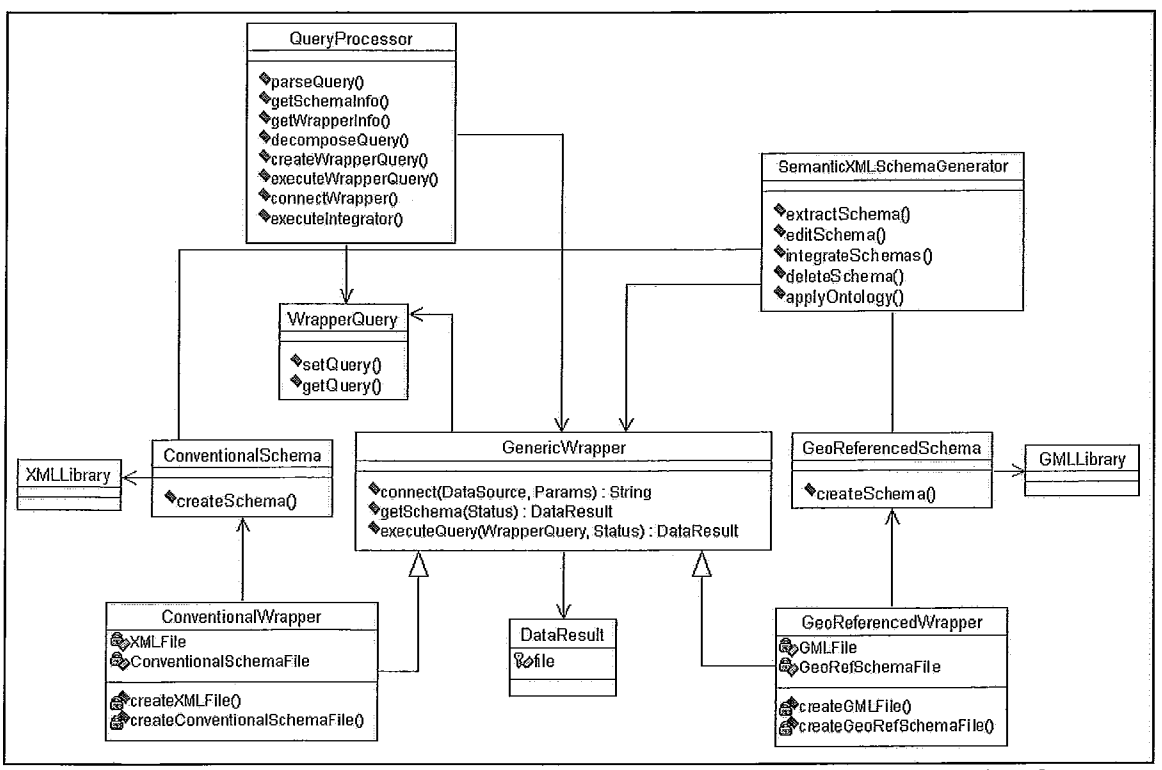


Figura 4.19 – Diagrama de Classes simplificado dos Tradutores e classes relacionadas

O Núcleo de Integração de Dados se comunica com os Tradutores usando a IPACE. O Núcleo de Integração de Dados não possui, e não precisa possuir, informações de como estão implementados os Tradutores, pois aquele acessa as fontes

de dados através dos Tradutores utilizando um conjunto de operações e atributos comuns a todos os Tradutores disponíveis através do mais genérico dos Tradutores (GenericWrapper). Como o Núcleo de Integração de Dados necessita somente conhecer as operações e atributos do Tradutor Genérico, não precisa ser modificado quando da necessidade de se agregar novos Tradutores. E com isso, possuímos também o recurso de criarmos novos Tradutores sem ter necessidade de recompilar todo o restante do código.

O Tradutor Genérico, GenericWrapper (Figura 4.19), possui vários métodos e atributos importantes para acessar as fontes de dados, dos quais cabe destacar:

- connect(DataSource, Params): String – Conecta com a fonte de dados (DataSource) e passa os parâmetros (Params) para a mesma, inclusive os parâmetros informando a localização da Fonte de Dados no Provedor de Dados. Retorna uma String contendo informações se foi bem sucedido, ou erro.
- getSchema(Status): DataResult – Retorna as informações do esquema da fonte de dados previamente conectada na forma de um objeto do tipo DataResult. Este objeto contém um arquivo, e a construção deste arquivo vai depender do Tradutor usado. Este método também retorna o Status de como foi o processamento e se ocorreu algum erro.
- executeQuery(WrapperQuery, Status): DataResult – Este método executa uma consulta em uma fonte de dados e retorna o resultado desta como um DataResult também. O Status tem as mesmas funcionalidades descritas anteriormente.

Os Tradutores irão transferir os dados das fontes de dados para o Núcleo de Integração de Dados em dois formatos, XML e GML, para passar dados convencionais e georreferenciados. Para tratar estes dados de maneira diferenciada existem dois Tradutores herdados do Tradutor genérico, um para dados convencionais (ConventionalWrapper) e outro para dados georreferenciados (GeoReferencedWrapper).

O Tradutor ConventionalWrapper é a base para todos os Tradutores que utilizam dados convencionais. Este Tradutor utiliza a classe ConventionalSchema do WISE para construir os esquemas e documentos XML - de acordo com a estrutura e regras do Esquema Base - que serão passados para o Núcleo de Integração de Dados. Ao ser

executado o método getSchema, herdado do GenericWrapper, este Tradutor acessa os metadados de uma fonte de dados convencionais, e cria um arquivo XML Schema que é o Esquema Convencional Aplicação para esta fonte de dados, e o retorna dentro de um objeto DataResult. O método executeQuery ao ser utilizado faz com que o Tradutor realize a consulta sob os dados da fonte de dados e gere o resultado como documento XML, este resultado é retornado dentro de um objeto DataResult.

Já o Tradutor GeoReferencedWrapper que é a base para todos os Tradutores que utilizam dados georreferenciados faz uso da classe GeoReferencedSchema do WISE para construir os esquemas e documentos GML que serão passados para o Núcleo de Integração de Dados. O método getSchema quando aplicado a este Tradutor acessa os metadados de uma fonte de dados georreferenciada, e cria um arquivo XML Schema que é o Esquema Convencional Aplicação e um arquivo GML Schema que é o Esquema Georreferenciado Aplicação para esta fonte de dados, e o retorna dentro de um objeto DataResult. O método executeQuery quando executado faz com que o Tradutor realize a consulta sob os dados e gere o resultado como documento GML, e este resultado é retornado dentro de um objeto DataResult.

O método executeQuery produz a sua saída em formato de documento XML ou GML, dependendo do XML Schema Semântico atual da Fonte de Dados usada na consulta pelo Tradutor. E este documento obedece à estrutura imposta neste esquema e a ordem passada pelos elementos da consulta WrapperQuery neste método.

As consultas utilizadas nos Tradutores (WrapperQuery) são padronizadas e estruturadas na forma de objetos. As consultas WrapperQuery são independentes de uma linguagem de consulta específica, de modo que se for necessário alterar a linguagem de consulta ou o modo como as consultas são realizadas no Núcleo de Integração de Dados, estas modificações não terão efeito sobre os Tradutores e o modo como realizam consultas. A independência de linguagem de consulta nos Tradutores também se deve ao fato de que estão sendo consultadas fontes de dados em diferentes modelos de dados e estes podem ter diferentes linguagens de consulta, e em alguns casos como o de fontes de dados com arquivos texto, planilhas Excel, estas nem mesmo possuem linguagem de consulta. Alguns Tradutores podem converter internamente uma consulta no formato WrapperQuery para uma consulta em uma linguagem de consulta específica, como SQL ou XQuery por exemplo.

Atualmente, o WISE tem os Tradutores para ESRI Shape, JDBC e Excel construídos, já que são mais utilizados nos casos de testes com as instituições nas quais este trabalho mantém vinculo. Os outros Tradutores vão ser construídos de acordo com a demanda do sistema por novos Tipos de Fontes de Dados.

4.5.5.1 Consultas Utilizadas nos Tradutores

Todas as consultas passadas para um Tradutor convencional ou georreferenciado devem estar no formato WrapperQuery (Figura 4.20), ou melhor, devem ser um objeto desta classe. Esta classe possui um objeto DataSourceQuery, que representa uma Fonte de Dados na consulta, que por sua vez possui um vetor de objetos DataSetQuery, que são Conjuntos de Dados da consulta. Um DataSetQuery pode ter nenhum, ou vários, objetos FieldQuery, que representam Campos na consulta. Quando o DataSetQuery passado na consulta não possuir nenhum objeto FieldQuery, o interpretador do Tradutor entenderá como sendo para retornar todos os Campos deste Conjunto de Dados no DataSetQuery. Cada Campo pode possuir restrições quanto aos valores que podem ser retornados na consulta. Essas restrições são representadas por objetos ConstraintQuery. Um exemplo de restrição seria retornar somente as cidades com população acima de 100000 habitantes (ConstraintQuery: populacao > 100000).

A classe DataSetQuery foi herdada, e possui características adicionais para suportar consultas georreferenciadas para serem utilizadas por Tradutores georreferenciados (GeoReferencedWrappers). A classe descendente se chama GeoDataSetQuery e possui um objeto da classe GeoRefInfo para suportar passar informações como o atributo *region*, que define que região do mapa deve ser retornada na consulta, definida por dois pares de coordenadas (x_1,y_1 e x_2,y_2).

Estes tipos de consultas usados nos Tradutores somente realizam a seleção e projeção dos dados, mas não realizam junções nestes. As junções vão ficar a cargo do Núcleo de Integração de Dados.

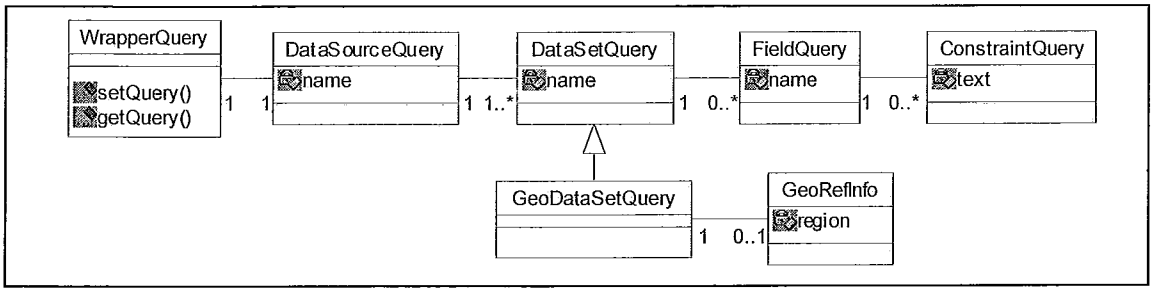


Figura 4.20 – Diagrama de Classes simplificado do WrapperQuery e classes afins

4.5.6 Gerador de XML Schema Semântico

É uma ferramenta capaz de extrair os esquemas das fontes de dados na forma de XML Schemas Semânticos, usando os Tradutores. Além disso, esta ferramenta é capaz de editar um XML Schema Semântico e o compor com mais elementos informativos, inclusive com a utilização de ontologias do domínio da aplicação adquiridas no Gerenciador de Ontologias. A integração de esquemas (SXMLS) também é realizada com esta ferramenta. A Figura 4.21 apresenta este componente.

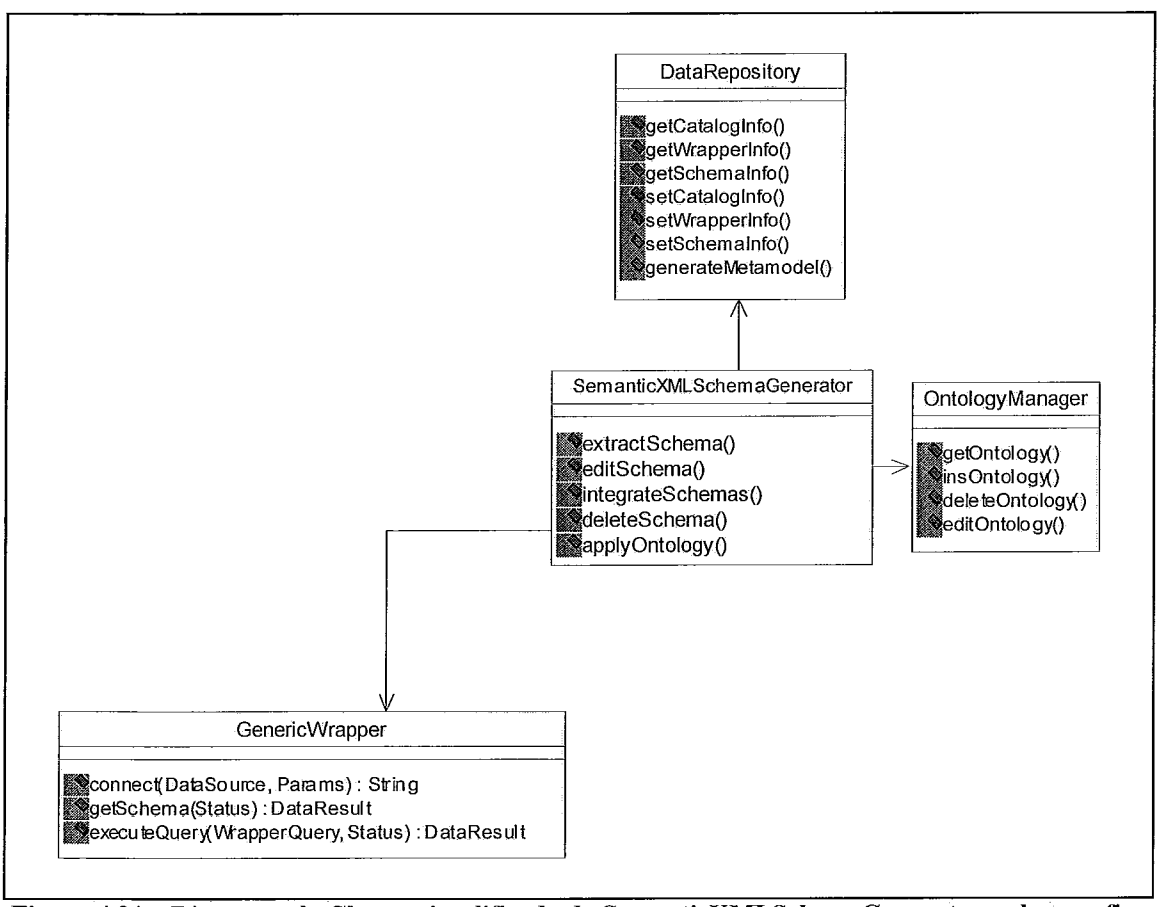


Figura 4.21 – Diagrama de Classes simplificado do SemanticXMLSchemaGenerator e classes afins

4.5.6.1 Extração dos esquemas das Fontes de Dados

O Gerador de XML Schema Semântico extrai os esquemas, tanto convencionais como georreferenciados das Fontes de Dados usando os Tradutores. Para tanto é necessário que o GID ou um GDL execute o Gerador de XML Schema Semântico, selecione um Provedor de Dados e uma Fonte de Dados para gerar o seu SXMLS.

Feito isto o WISE pode realizar a extração dos esquemas das fontes de dados ao chamar o método extractSchema da classe SemanticXMLSchemaGenerator, Figura 4.21, que é o próprio Gerador de XML Schema Semântico. Este método por sua vez executa o método connect do Tradutor associado à Fonte de Dados no Provedor de Dados, e após estar conectado executa o método getSchema do Tradutor para adquirir o esquema desejado. Este esquema é finalmente trazido para o Gerador de XML Schema Semântico que o armazena no Repositório de Dados e fica disponível para o sistema.

Este esquema pode trazer além da Fonte de Dados, Conjunto de Dados (ou Coleção de Feições), Campos e Chaves do esquema, todos os elementos restritivos e informativos admissíveis.

Este processo é realizado após serem adicionadas novas Fontes de Dados ao WISE, ou ainda quando essas Fontes de Dados têm seus esquemas atualizados localmente.

4.5.6.2 Edição de XML Schemas Semânticos

Após um SXMLS ter sido criado, pode-se editá-lo para que sejam adicionadas mais informações descritivas e semânticas, e também ontologias do domínio da aplicação para que este esquema fique ainda mais rico, de modo a facilitar o processo de integração de esquemas, e assim o processo de integração de dados.

O processo de edição de um SXMLS é iniciado com a interface de edição do Gerador de XML Schema Semântico ao executar o método editSchema, que é utilizado para a edição de SXMLSs. Este método permite que um esquema deste tipo seja editado utilizando os recursos para editar esquemas convencionais e georreferenciados disponíveis, respectivamente, nas classes ConventionalSchema e GeoReferencedSchema. Estas classes utilizam ainda informações das classes InformativeElement e GeoRefInformativeElement, para tratar alguns elementos informativos convencionais e georreferenciados.

A semântica do SXMLS é preferencialmente acrescida a este por quem tem conhecimento dos dados de uma determinada Fonte de Dados, ou seja, pelo seu GDL. Com o Gerador de XML Schema Semântico, o GDL pode transmitir mais informações semânticas a este esquema utilizando ontologias do domínio da aplicação, e também heurísticas pré-definidas no WISE, como conversão de unidades de medidas.

Com o SXMLS definido, além do GDL, é também viável que terceiros (GID, e Usuários de Dados Avançados) consigam integrar vários destes esquemas em outros esquemas, os chamados XML Schemas Semânticos Integrados.

4.5.6.3 Integração de XML Schemas Semânticos

Utilizando a parte de Integração de SXMLS do Gerador de XML Schema Semântico, estes esquemas podem ser integrados em outros esquemas, chamados de XML Schemas Semânticos Integrados. Criando desta forma várias visões dos esquemas originais de acordo com a necessidade de cada grupo de usuários.

Com esta ferramenta pode-se integrar qualquer combinação de tipos de esquemas, ou seja, dois esquemas convencionais, um esquema convencional com um georreferenciado ou dois esquemas georreferenciados. Para tanto, deve ser escolhido que tipo de esquema integrado deverá ser gerado, Esquema Integrado Convencional ou Esquema Integrado Georreferenciado. O Esquema Integrado Convencional pode ser criado através da integração de esquemas convencionais ou georreferenciados, ou a combinação dos dois, porém só manterá em seu conteúdo as características não georreferenciadas. Já o Esquema Integrado Georreferenciado pode ser criado através da integração de esquemas convencionais ou georreferenciados, ou a combinação dos dois, mantendo as características georreferenciadas.

Este processo de integração de esquemas é um processo semi-automático e deve ser realizado com a interação de uma pessoa, como o GID, o GDL, e algumas vezes por Usuários de Dados Avançados, para realizar a verificação de consistência das informações que estão sendo integradas, para que sejam realizadas escolhas quando ocorrerem disparidades, de modo também a verificar se as integrações de esquemas estão sendo realizadas de acordo com o esperado ou se necessitam de alguma alteração.

Para iniciar o processo de integração de esquemas deve-se executar o método integrateSchemas da classe SemanticXMLSchemaGenerator, que solicita dois esquemas a serem integrados, e quando estiver processando realizará interrupções para que o

usuário desta classe possa interagir no processo de integração de esquemas, conforme pode ser visto no Algoritmo de Integração de SXMLS listado no Apêndice C.

Como é utilizado o processo de integração binária, um interpretador analisa cada um dos dois esquemas a serem integrados e tenta identificar as características, que estes esquemas têm em comum e suas disparidades, que poderiam ser integradas, respeitando um conjunto de restrições sintáticas e semânticas baseado em alguns conflitos de integração de dados existentes na literatura (CHATTERJEE & SEGEV, 1991; CASTELLANOS et al., 1994; KASHYAP & SHETH, 1996), como os apresentados no capítulo 2.

Como existem diversos conflitos de integração de dados na literatura, tanto conflitos sobre dados convencionais como georreferenciados, foi escolhido um subconjunto destes para serem tratados neste trabalho, já que o tratamento de todos vai além do escopo deste trabalho e que podem ser abordados em trabalhos futuros. Os conflitos escolhidos foram: Conflitos de Sistema de Projeções e Coordenadas, Conflitos de Nomes, e Conflitos de Unidades de Medidas. Estes conflitos foram selecionados por se tratar de conflitos fundamentais na resolução da integração de dados georreferenciados ou convencionais.

Resolução de Conflitos de Sistemas de Projeções e Sistemas de Coordenadas

Como existem diversas maneiras de se representarem objetos georreferenciados no espaço, e quando existe a necessidade de integração de dados usando objetos que possuem representações diferentes no espaço, deve-se converter estes objetos para um mesmo Sistema de Projeções e Coordenadas de modo que não sejam gerados conflitos ou aberrações ao se integrarem estes objetos.

Alguns dos Elementos Informativos Georreferenciados armazenam as informações relativas aos Sistemas de Projeções e Coordenadas dos XML Schemas Semânticos, como visto no Esquema Base. No processo de integração de esquemas, ao gerar o XML Schema Semântico Integrado, estas informações são adicionadas a este, e também informações de conversão de Sistemas de Projeções e Coordenadas caso seja necessário converter alguma informação deste tipo, de um esquema componente para o esquema integrado. Cabe ressaltar que nesta fase de integração dos esquemas são adicionadas informações ao ISXMLS que serão utilizadas pelo Integrados de Dados e

Mapas para converter os dados das Fontes de Dados no momento da integração de dados.

Resolução de Conflitos de Denominação

Como já mencionado no capítulo 2, conflitos de denominação ocorrem devido aos esquemas incorporarem denominações para os objetos representados. Podendo ser sinônimos ou homônimos. Os sinônimos ocorrem quando entidades ou propriedades semanticamente idênticas são denominadas diferentemente; e os homônimos ocorrem quando entidades e propriedades diferentes semanticamente compartilham a mesma denominação.

Nos processos de extração e edição de SXMLSs, os nomes existentes nos esquemas das Fontes de Dados originais podem ser complementados com um conjunto de outros termos definidos usando o elemento Ontology.Name definido neste trabalho, que serve para auxiliar no processo de integração de dados ao comparar os termos pertencentes a um esquema com os termos do outro esquema. Se os termos comparados forem iguais ou parecidos, estes são exibidos ao usuário da interface de integração de esquemas do Gerador de XML Schema Semântico para que este possa decidir, baseado nas descrições dos elementos, se estes realmente são idênticos ou não, ou ainda verificar se são sinônimos ou homônimos. Caso não haja descrição dos elementos acerca das informações a serem analisadas, o usuário terá que tentar por seus próprios meios descobrir do que se trata cada assunto, ou consultar o Gerente de Dados Local dos esquemas envolvidos na integração.

Resolução de Conflitos de Unidades de Medida

Estes conflitos ocorrem devido ao fato de que unidades de medidas diferentes podem estar atribuídas aos esquemas a serem integrados. Desta forma é necessário converter estas diversas unidades de medidas para uma unidade de medida em comum no esquema integrado.

Como as unidades de medidas estão associadas aos campos, devemos converter cada unidade de medida de acordo com os campos que as utilizam, para isto este trabalho possui o elemento UnitOfMeasure para representar a unidade de medida de cada campo de um esquema. As informações dos elementos UnitOfMeasure devem ser preenchidas nos processos de extração ou edição de SXMLS, para que, depois, no

processo de integração de esquemas venham a ser comparadas e caso necessário realizar a conversão destas unidades de medida entre os esquemas.

Um dicionário de unidades de medidas e suas respectivas conversões é mantido pelo WISE para que possa ser usado no processo de extração, edição e integração de esquemas, bem como no processo de integração de dados que irá ocorrer posteriormente, dependendo deste processo inicial.

As informações a respeito dos campos que necessitarem de conversão de unidades de medidas, bem como as conversões que devem ser realizadas são armazenadas no ISXMLS, na forma do elemento Conversion.UnitOfMeasure, para cada campo.

4.5.6.4 Exclusão de XML Schemas Semânticos

Os XML Schemas Semânticos podem ser excluídos utilizando o método deleteSchema do Gerador de XML Schema Semântico. Ao executar este método esta ferramenta exclui o esquema selecionado do Repositório de Dados onde reside, e atualiza o Metamodelo através do método generateMetamodel da classe DataRepository que por sua vez chama o método createMetamodel da classe MetamodelGenerator, que é o Gerador de Metamodelos.

4.5.6.5 Classes para Tratar XML Schemas Semânticos

O Gerador de XML Schema Semântico possui algumas classes para a criação de SXMLS, ConventionalSchema e GeoReferencedSchema (Figura 4.22) para tratar respectivamente de esquemas convencionais e georreferenciados. Estas classes possuem ainda outras classes relacionadas para ajudarem nesta tarefa. Estas classes são InformativeElement e GeoRefInformativeElement para tratar dos elementos informativos definidos na seção 4.2 e outras classes para representar informações de Fonte de Dados, Conjunto de Dados, Campos, e Chaves.

Todas estas classes são utilizadas pelos Tradutores para criarem os SXMLSs das suas Fontes de Dados, bem como pelo Gerador de XML Schema Semântico para criar o ISXMLS e para editar qualquer tipo de SXMLS. Estas classes formam a estrutura básica para a construção dos esquemas, e devem ser estendidas ou alteradas para suportar um novo formato de esquema, caso seja necessário a construção de um.

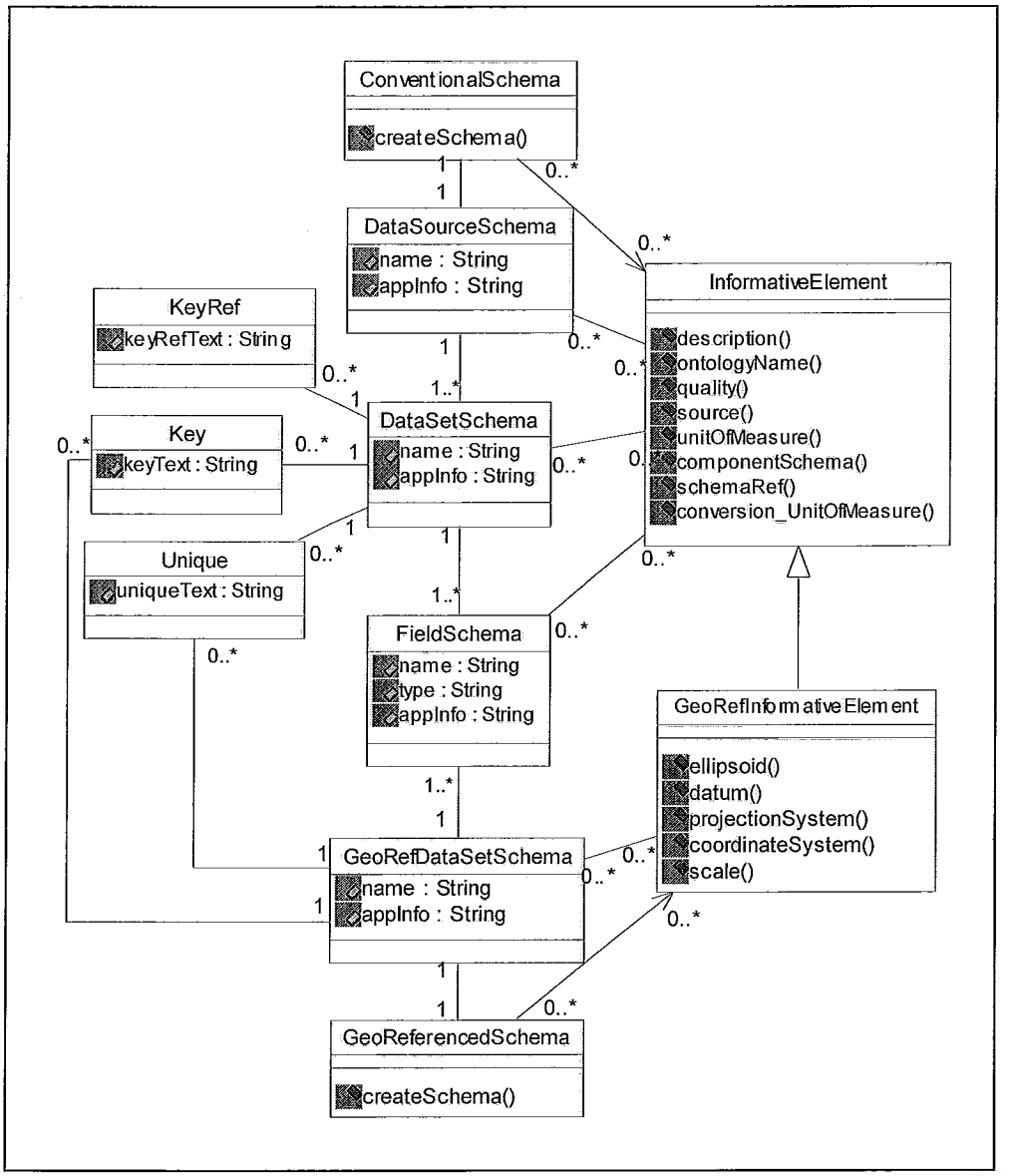


Figura 4.22 – Diagrama de Classes simplificado do ConventionalSchema e GeoReferencedSchema e outras classes relacionadas

4.5.7 Gerenciador de Ontologias

O Gerenciador de Ontologias é uma ferramenta para acessar um sistema, ou biblioteca onde são cadastradas, e mantidas ontologias. O sistema que está sendo utilizado é o Protégé (PROTÉGÉ, 2004), onde é mantida uma interface para que sejam extraídas as ontologias e utilizadas na geração dos SXMLS e na Integração de Dados e Mapas. Ontologias do domínio da aplicação podem ser adicionadas, editas e excluídas na sua interface gráfica.

O Gerador de XML Schema Semântico através do método getOntology da classe OntologyManager, Figura 4.23, que é o próprio Gerenciador de Ontologias, torna disponível estas ontologias tanto para as classes ConventionalSchema, GeoReferencedSchema quanto para um usuário de sua interface. Ele permite que as ontologias possam ser aplicadas nos XML Schemas Semânticos através do elemento "Ontology.Name", que posteriormente será utilizado no processo de integração semântica de esquemas.

O elemento "Ontology.Name" é o responsável por manter as ontologias nos SXMLSs. Quando é realizado o processo de integração de esquemas SXMLSs estas ontologias são utilizadas para encontrar correspondência entre elementos semelhantes de fontes de dados diferentes. Essas ontologias são as únicas utilizadas no processo de integração de esquemas, e não é realizada nenhuma busca no Gerenciador de Ontologias por informações de ontologias que estejam associadas àquelas.

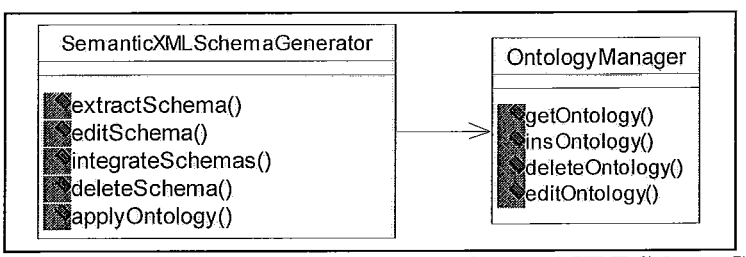


Figura 4.23 – Diagrama de Classes simplificado do SemanticXMLSchemaGenerator e OntologyManager

4.5.8 Gerador de Metamodelo

Este gerador provê serviços para construir e disponibilizar um Metamodelo de Dados construído a partir dos SXMLSs armazenados no Repositório de Dados (Catálogo).

Este Metamodelo de Dados é definido em XML, onde é construída uma listagem de todos os SXMLS que estão no Repositório de Dados, separados por esquemas simples e esquemas integrados.

O Gerador de Metamodelo, representado pela classe MetamodelGenerator (Figura 4.24) possui um método chamado getMetamodel que serve para acessar o Metamodelo construído que fica armazenado no Repositório de Dados. Caso não exista

nenhum modelo construído ele executa o método createMetamodel para criar o Metamodelo.

O método createMetamodel também pode ser chamado pelo método generateMetamodel da classe DataRepository, sempre que houver a necessidade deste Metamodelo ser gerado novamente para refletir as alterações realizadas nos SXMLSs armazenados no Repositório de Dados.

As informações deste Metamodelo podem ser visualizadas pelos usuários no Visualizador de Metamodelos da ICWISE, onde aqueles podem visualizar os metadados necessários aos seus interesses, para que possam realizar consultas no Editor de Consultas.

A Listagem 4.9 apresenta o XML Schema usado como base para criar o documento XML que representa o Metamodelo de Dados.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="Metamodel" type="wisesxmls:MetamodelType"/>
    <!--================================================================ -->
    <element name="SimpleSchema" type="wisesxmls:SimpleSchemaType"/>
    <!--================================================================ -->
    <element name="IntegratedSchema" type="wisesxmls:IntegratedSchemaType"/>
    <!-- ================================================================
     complex types
     ================================================================ -->
    <complexType name="MetamodelType">
        <sequence>
            <element ref="wisesxmls:SimpleSchema" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="wisesxmls:IntegratedSchema" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <!--================================================================ -->
    <complexType name="SimpleSchemaType">
        <sequence>
            <element name="DataProvider">
                <complexType>
                    <sequence>
                        <element name="Name" type="wiseString"/>
                        <element name="Cod" type="wiseInt"/>
                        <element name="DataSource" maxOccurs="unbounded">
                            <complexType>
                                <sequence>
                                    <element name="Name" type="wiseString"/>
                                    <element name="Cod" type="wiseInt"/>
                                    <element name="Schema" maxOccurs="unbounded">
                                        <complexType>
                                            <sequence>
                                                <element name="Name" type="wiseString"/>
                                                <element name="SchemaContent" type="wiseString"/>
                                            </sequence>
                                        </complexType>
                                    </element>
```

```
                    </sequence>
                  </complexType>
                </element>
              </sequence>
            </complexType>
          </element>
        </sequence>
        <attribute name="id" type="wisesxmls:wiseID" use="required"/>
    </complexType>
    <!--================================================================== -->
    <complexType name="IntegratedSchemaType">
        <sequence>
            <element name="Name" type="wiseString"/>
            <element name="Cod" type="wiseInt"/>
            <element name="SchemaContent" type="wiseString"/>
            <element name="ComponentSchema" minOccurs="2" maxOccurs="unbounded">
                <complexType>
                    <sequence>
                        <element name="schema" type="wiseIDREF" use="required"/>
                    </sequence>
                </complexType>
            </element>
        </sequence>
        <attribute name="id" type="wisesxmls:wiseID" use="required"/>
    </complexType>
    <!--================================================================== -->
</schema>
```

Listagem 4.9 – XML Schema usado como base para criar o documento XML do Metamodelo

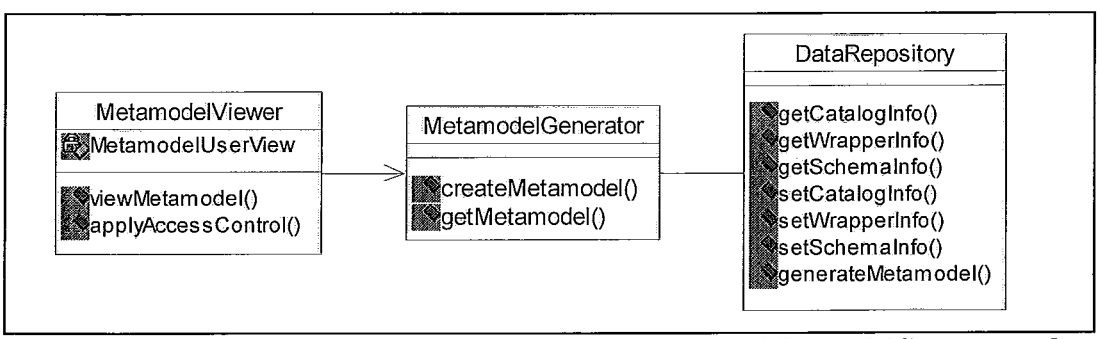


Figura 4.24 – Diagrama de Classes simplificado mostrando a classe MetamodelGenerator e classes afins

4.5.9 Processador de Consultas

Esta ferramenta fornece um conjunto de serviços para processar as consultas realizadas no WISE.

O Processador de Consultas recebe através do método parseQuery da classe QueryProcessor (Figura 4.25) que é o próprio Processador de Consultas, uma consulta externa no formato WISEQuery. Quando aquele a analisa, usando o método getSchemaInfo, e verifica se esta é sobre algum dos ISXMLSs ou sobre um dos SXMLSs individuais, a partir de informações adquiridas no Repositório de Dados sobre como estes dados estão distribuídos e qual sua estrutura. Se a consulta é sobre um ISXMLS, ele divide a consulta original em subconsultas de acordo com as Fontes de Dados que estão neste, usando o método decomposeQuery. Então é executado o método

getWrapperInfo para acessar informações dos Tradutores relacionados às Fontes de Dados, para que em seguida sejam criadas consultas no formato WrapperQuery de acordo com o Tradutor e sua Fonte de Dados, usando o método createWrapperQuery. O Processador de Consultas executa seu método connectWrapper para se conectar com cada Tradutor, e este método chama o método connect de cada um destes. Finalmente, cada consulta no formato WrapperQuery é enviada para o Tradutor responsável pela Fonte de Dados informada naquela utilizando o método executeWrapperQuery.

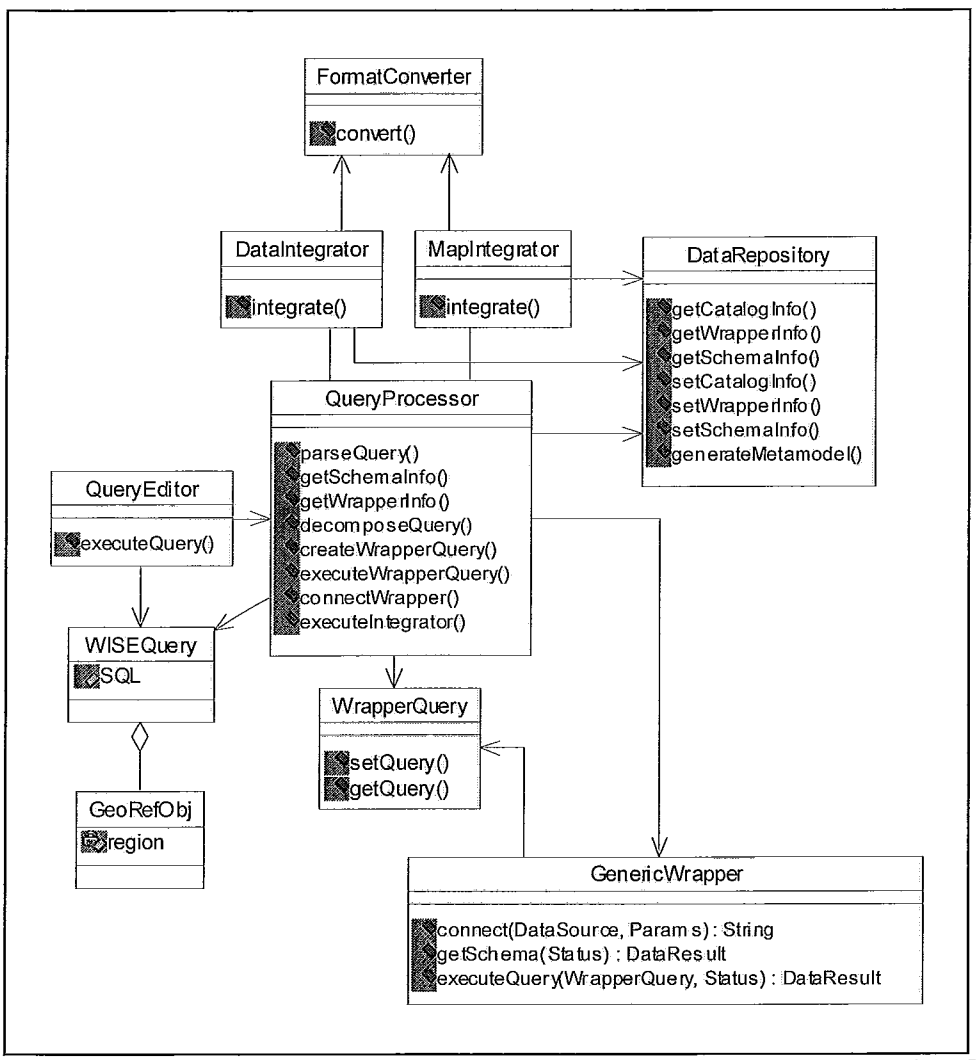


Figura 4.25 – Diagrama de Classes simplificado do QueryProcessor e classes relacionadas

Os Tradutores recebem consultas no formato da estrutura WrapperQuery e as interpreta de acordo com os Tipos de Fontes de Dados a que estão relacionados. Após ter interpretado uma consulta, o Tradutor consulta a Fonte de Dados destino, que por sua vez retorna um resultado, ou um erro, para o Tradutor que a consultou, este retorna o resultado através do método executeWrapperQuery para o Processador de Consultas, que então o entregará ao Integrador de Dados e Mapas ao executar o método integrate

deste. Cabe ressaltar que o Processador de Consultas não integra os resultados, sendo esta tarefa destinada ao Integrador de Dados e Mapas.

4.5.10 Integrador de Dados e Mapas

O Integrador de Dados e Mapas é uma ferramenta que provê um conjunto de serviços para a realização da integração semântica de dados e mapas, ou seja, é capaz de integrar dados convencionais e dados georreferenciados. O Integrador de Dados e Mapas, é composto de duas classes, DataIntegrator e MapIntegrator, para tratar dados convencionais e georreferenciados separadamente.

Como já foi dito, as funcionalidades de integração de esquemas não estão nesta ferramenta e são realizadas pelo Gerador de XML Schema Semântico, onde dois ou mais esquemas são integrados em um XML Schema Semântico Integrado. Esta ferramenta só realiza integração de dados (e mapas).

O Processador de Consultas após ter obtido o resultado das consultas dos Tradutores, chama o Integrador de Dados e Mapas executando o método executeIntegrator, que serve para chamar o método integrate da classe DataIntegrator ou MapIntegrator (Figura 4.26), de acordo com o tipo de integração de dados que se pretende realizar. São passados neste método as consultas WISEQuery, os XML Schemas Semânticos das Fontes de Dados envolvidas na consulta, bem como seus resultados das consultas, onde o Integrador de Dados e Mapas age para realizar as integrações necessárias nestes dados.

Quando o Integrador de Dados e Mapas termina a sua execução é produzido um arquivo, XML ao usar a classe DataIntegrator ou GML ao utilizar a classe MapIntegrator, que é enviado para o Conversor de Formatos caso seja necessário realizar alguma conversão para outro tipo de formato de arquivo. E, finalmente, é enviado para o Publicador de Dados onde é exibido o resultado para o usuário.

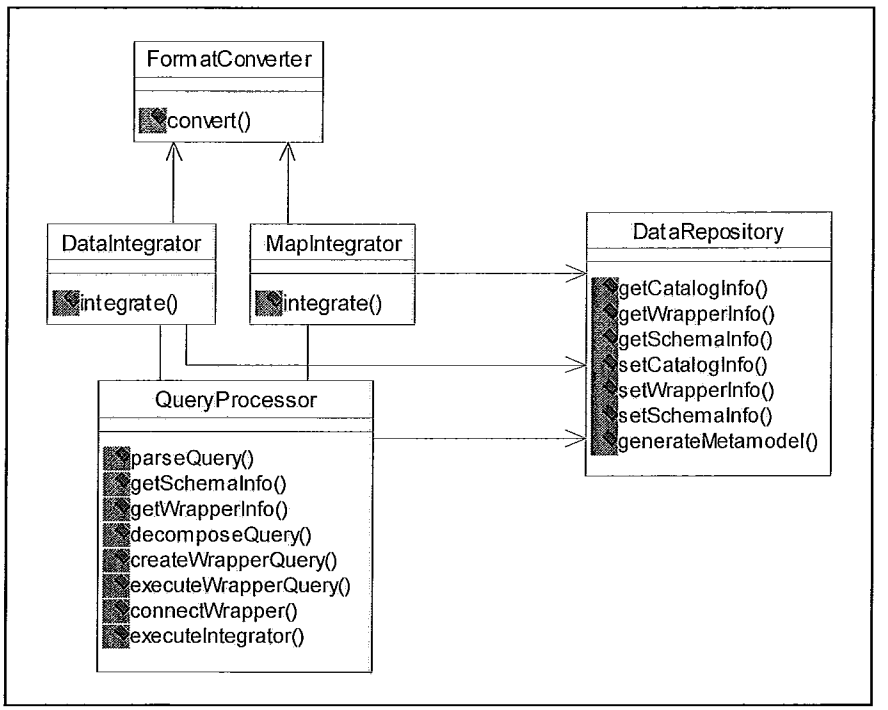


Figura 4.26 – Diagrama de Classes simplificado do DataIntegrator e MapIntegrator e classes relacionadas

4.5.11 Conversor de Formatos

O Conversor de Formatos é uma ferramenta capaz de converter o resultado de uma consulta que está no formato XML (no caso de dados convencionais) e GML (dados georreferenciados) para um outro formato de arquivo de saída.

O usuário pode escolher um formado de arquivo dentre os disponíveis na arquitetura, no momento de realizar a consulta, ou utilizar os formatos padrão XML e GML, para dados convencionais e georreferenciados, respectivamente.

Como é intenção deste trabalho disponibilizar uma variedade de formatos de arquivos de saída, de modo que o usuário possa selecionar aquele que melhor se adapte a suas necessidades, foram criadas algumas classes para tratar da conversão entre formatos. Uma delas é a classe FormatConverter (Figura 4.27), que possui um método chamado convert que recebe como parâmetros o nome do arquivo de origem e o nome do arquivo destino e faz a conversão entre eles, usando a informação da extensão dos arquivos para decidir qual conversor de formato deve executar. Uma outra classe notória é a GenericConverter, que é a classe de conversão genérica que pode ser estendida para suportar novos tipos de conversores de formatos específicos. Esta última também possui um método chamado convert que é executado pelo método de mesmo nome da classe FormatConverter.

Pretende-se que o WISE ao estar totalmente funcional disponibilize dados nos formatos: XML, Excel, Texto, para dados convencionais e, para dados georreferenciados: GML, ESRI Shape, SVG (*Scalable Vector Graphics*) (SVG, 2003) e tipos de imagens raster, além de alguns formatos de arquivos de banco de dados.

Como o Conversor de Formatos pode ser estendido, pois possui classes que podem ser especializadas para que sejam manipulados novos formatos de arquivos, flexibiliza-se bastante a arquitetura para se adaptar aos padrões mais atuais de formatos de troca de dados do mercado e do meio científico.

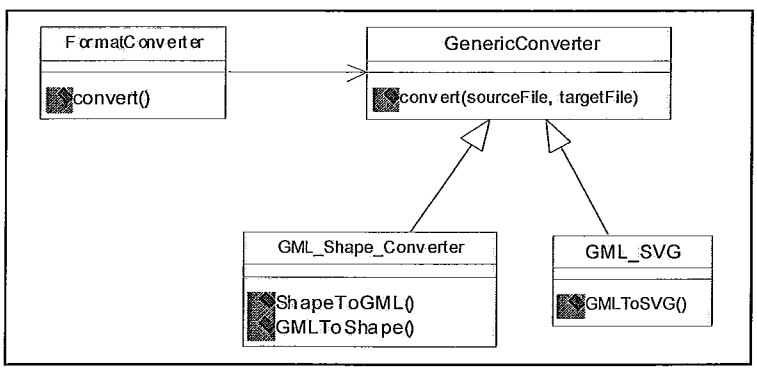


Figura 4.27 – Diagrama de Classes simplificado do FormatConverter e GenericConverter

Capítulo 5 – Questões de Implementação e Estudo de Caso

O presente capítulo apresenta uma explicação de como o WISE foi implementado até o momento atual. São apresentadas as tecnologias e ferramentas que foram utilizadas na sua concepção, sendo também apresentadas algumas partes do protótipo e sua utilização em um estudo de caso para a geração de SXMLSs e consulta.

5.1 Concepção do WISE

O sistema WISE foi desenvolvido até o estágio atual baseando-se na concepção de ser um sistema multiplataforma, que utiliza ferramentas e bibliotecas de software de código-fonte aberto ou livre.

Como linguagem de programação foi escolhida a linguagem Java (JAVA, 2004) na sua versão 1.4.2, por se tratar de uma linguagem que gera código executável para diversas plataformas de software/hardware, além de ser distribuída como uma ferramenta de código-fonte aberto e ser livre. Java também possui recursos, por ser interpretada e por possuir classes *Factory*, que permite agregar novos módulos sem que o todo necessite ser reconstruído. Isto ajuda a tornar o WISE um sistema extensível, capaz de agregar novos Tradutores, formatos de saída, e outros módulos, com uma quantidade insignificante de configuração e sem precisar compilá-lo novamente.

O SGBDR utilizado foi o Firebird (FIREBIRD, 2004), versão 1.5, que foi originado do Borland Interbase (INTERBASE, 2004), versão 6.0, sendo uma ferramenta de código-fonte aberto e livre, que possui diversos recursos que não serão apresentados por estar além do escopo deste trabalho, devendo os interessados procurar a referência supracitada. O banco de dados Firebird é acessado pelo WISE através de seu *driver* JDBC tipo 4, ou seja, é totalmente independente de plataforma de software/hardware.

Foram utilizadas as linguagens XML e GML e tecnologias afins, como já citadas nos capítulos 2 e 4, para servirem de MDC e transporte de dados do WISE. Para tratar estas linguagens foi utilizado um conjunto de *parsers*, processadores e interfaces de programação de aplicativos de XML, Xerces (XERCES, 2004), Xalan (XALAN, 2004) e JAXP (JAXP, 2004).

Para tratar dados georreferenciados e realizar diversos tipos de operações como apresentação, conversão de formato, conversão de sistemas de projeções entre outros, foram utilizadas algumas ferramentas e bibliotecas, que algumas vezes foram utilizadas como estão, outras foram alteradas para se adaptarem as necessidades deste trabalho, e ainda integradas. Estas ferramentas e bibliotecas são: Deegree (DEEGREE, 2004), GeoTools (GEOTOOLS, 2004) e OpenMap (OPENMAP, 2004), sendo todas de código-fonte aberto. Cabe destacar que estas ferramentas quando não possuem documentação possuem uma documentação precária, por não serem produtos comerciais, dificultando ainda mais o processo de se trabalhar com elas e seus inúmeros módulos.

Foi utilizado para tratar ontologias o software Protégé (PROTÉGÉ, 2004) de código-fonte aberto da Universidade de Stanford.

Alguns Tradutores do WISE foram construídos usando algumas bibliotecas não contidas no pacote da linguagem Java. O Tradutor para arquivo ESRI Shape, usou a biblioteca Deegree citada acima. O Tradutor para arquivos Microsoft Excel, usou uma biblioteca chamada JExcelAPI (JEXCELAPI, 2004) para conseguir tratar arquivos deste tipo. Novos Tradutores podem vir a utilizar outra diversidade de bibliotecas, sempre procurando utilizar bibliotecas de código-fonte aberto ou livre.

Como o WISE é um sistema bastante amplo e complexo ficaria inviável implementá-lo por completo durante este trabalho. Portanto foram implementados alguns dos seus módulos para permitir que este sistema fosse utilizado em Estudos de Caso de forma suficientemente abrangente. Sendo assim, foram implementados os módulos: Gerador de XML Schema Semântico com mais detalhes; Tradutores para arquivos ESRI Shape, Excel, e JDBC; Integrador de Dados e Mapas, Editor de Consultas, Publicador de Dados, Gerenciador de Dados do Sistema e Processador de Consultas implementados superficialmente.

O Estudo de Caso apresentado abaixo foi construído para exemplificar uma visão bem geral da arquitetura, apresentando algumas das diversas funcionalidades implementadas neste trabalho.

5.2 Estudo de Caso

O presente estudo de caso visa mostrar um exemplo de utilização do WISE, particularmente sobre a geração e integração de SXMLSs e a realização de uma

consulta. Um exemplo de todas as funcionalidades implementadas da arquitetura seria demasiadamente longo, logo não será apresentado neste texto.

Este estudo de caso parte do pressuposto que já existem fontes de dados cadastradas no WISE e algumas destas serão utilizadas na integração de dados.

Seja a necessidade de se integrar fontes de dados contendo informações de bacias hidrográficas e municípios do Rio de Janeiro. Existem no sistema duas fontes de dados que tratam destes assuntos. Será utilizado no processo de integração de esquemas estas duas fontes de dados, que tratam da bacia do rio Paraíba do Sul com suas sub-bacias e também informações de municípios. As fontes de dados utilizadas são da Escola Nacional de Ciências Estatísticas (ENCE) do IBGE, e do Laboratório de Hidrologia da COPPE/UFRJ.

A fonte de dados da ENCE é chamada BaciaParaibaSulRioJaneiro e possui como conteúdo dois arquivos ESRI Shape de nomes BaciaParSulEstRioJaneiro e MunicipiosEstRioJaneiro, que possuem informações de partes da bacia do Rio Paraíba do Sul que cortam o Estado do Rio de Janeiro e informações dos municípios deste estado, respectivamente. Já a fonte de dados do Laboratório de Hidrologia é chamada BaciaRioParaibaSul e também possui arquivos ESRI Shape chamados BacParSul, MunBacParSul, DistritBacParSul, SolosBacParSul, que tratam respectivamente da bacia e sub-bacias do Rio Paraíba do Sul, municípios em que esta bacia e sub-bacias estão localizadas, distritos destes municípios e solos da região abrangida pela bacia. Cabe ressaltar que estas duas fontes de dados possuem dados georreferenciados. A Tabela 5.1 apresenta algumas informações sobre os dados dos arquivos Shape.

Tabela 5.1 – Informações das fontes de dados do estudo de caso

FONTE DE DADOS	ARQUIVO SHAPE	DATUM	UOMs
ENCE BaciaParaibaSulRioJaneiro	BaciaParSulEstRioJaneiro	SAD-69	m, m^2
	MunicipiosEstRioJaneiro	SAD-69	Km, Km^2
Lab. Hidrologia BaciaRioParaibaSul	BacParSul	Córrego Alegre	m, m^2
	MunBacParSul	Córrego Alegre	m, m^2
	DistritBacParSul	Córrego Alegre	m, m^2
	SolosBacParSul	SAD-69	m, m^2

A representação gráfica dos arquivos ESRI Shape destas fontes de dados podem ser visualizados na Figura 5.1.

Este estudo de caso está dividido em algumas fases: a) extração dos esquemas das fontes de dados e geração de SXMLSs; b) edição de SXMLSs; c) integração de SXMLSs e finalmente d) consulta a um SXMLS. Como os SXMLS utilizados são extensos, alguns deles serão colocados no Apêndice C.

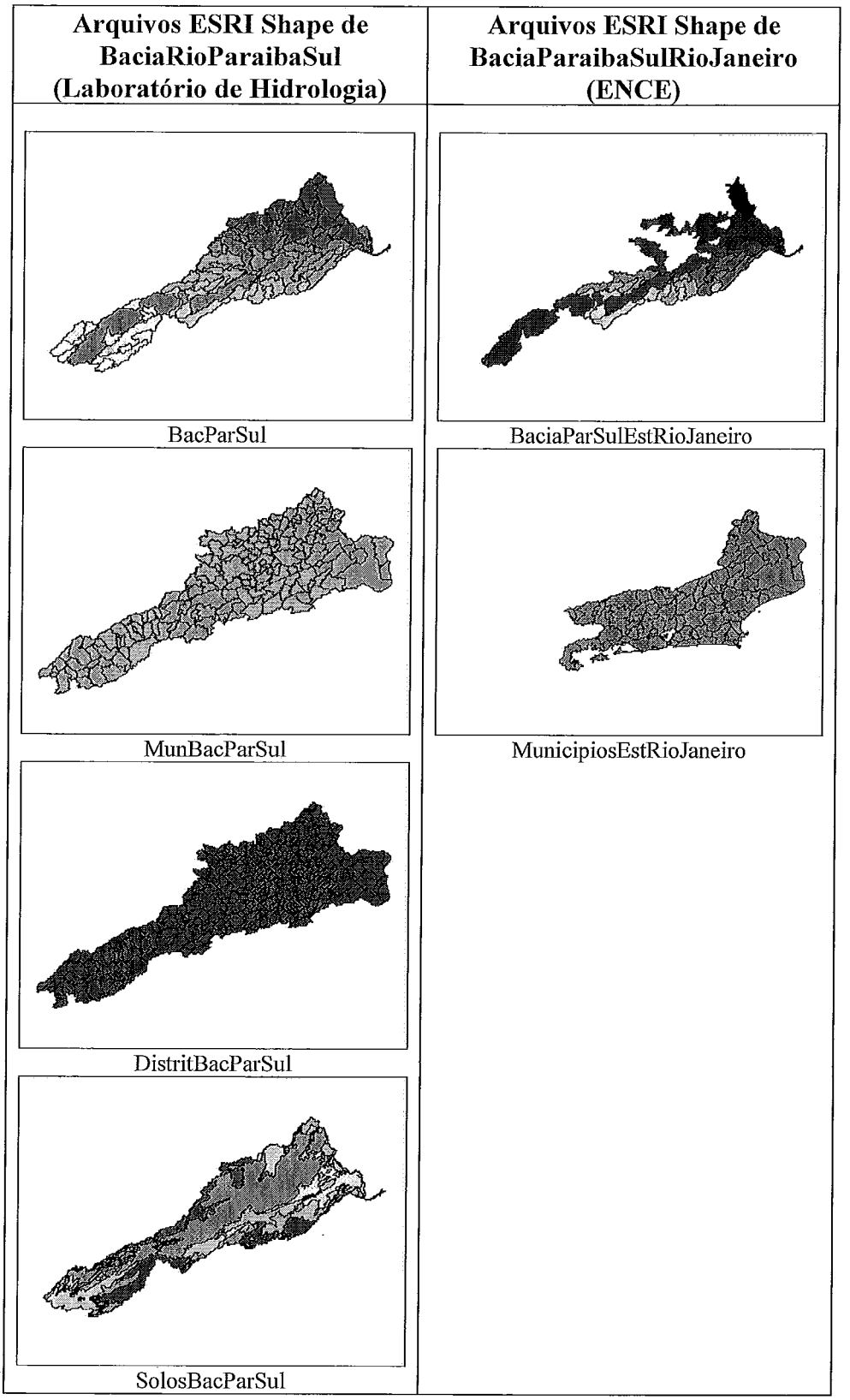


Figura 5.1 – Representação gráfica dos arquivos ESRI Shape das fontes de dados da ENCE e do Laboratório de Hidrologia

5.2.1 Extração e Geração de SXMLSs

Considerando que as fontes de dados da ENCE e do Laboratório de Hidrologia já estão cadastradas no WISE, será utilizado o Gerador de XML Schema Semântico (*Semantic XML Schema Generator*) para extrair seus esquemas locais e gerar o respectivo SXMLS. A Figura 5.2 mostra a interface da ferramenta de extração de esquemas do Gerador de XML Schema Semântico, que armazena os SXMLSs no Catálogo do WISE após serem gerados.

Esta ferramenta é executada uma única vez por fonte de dados, e realiza a extração de todos os seus esquemas locais. Feito isto é gerado o Esquema Convencional Aplicação e seus Esquemas Georreferenciados Aplicação de cada fonte de dados.

A Listagem 5.1 mostra o Esquema Georreferenciado Aplicação da fonte de dados da ENCE que é gerado com o mesmo nome desta, BaciaParaibaSulRioJaneiro.xsd. A Listagem 5.2 mostra o Esquema Georreferenciado Aplicação BaciaParSulEstRioJaneiro.xsd que está agregado ao primeiro, e foi criado utilizando o nome do arquivo ESRI Shape correspondente (um arquivo Shape corresponde a uma coleção de feições ou Feature Collection). O outro Esquema Georreferenciado Aplicação, MunicipiosEstRioJaneiro.xsd, da fonte de dados da ENCE, bem como o Esquema Convencional Aplicação do Laboratório de Hidrologia, BaciaRioParaibaSul.xsd, e seus Esquemas Georreferenciados Aplicação, BacParSul.xsd, MunBacParSul.xsd, DistritBacParSul.xsd, SolosBacParSul.xsd, que foram construídos da mesma forma que os da fonte de dados da ENCE, são encontrados no Apêndice C.

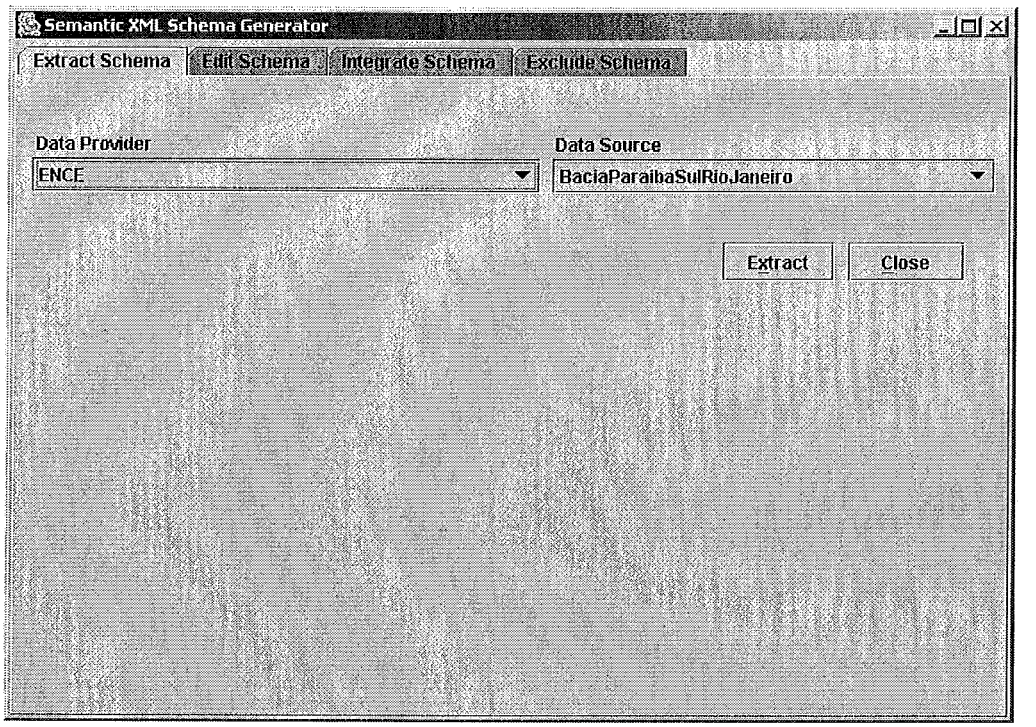


Figura 5.2 – Interface de extração de esquemas do Gerador de XML Schema Semântico

O Esquema Convencional Aplicação BaciaParaibaSulRioJaneiro.xsd, Listagem 5.1, representa seus Esquemas Georreferenciados Aplicação agregados através da palavra-chave *include*, e os dois elementos ao final deste arquivo representam as coleções de feições destes. Alguns elementos informativos já foram passados pelo tradutor no momento da geração deste, Source e Description.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <include schemaLocation="SXMLS-Base.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <include schemaLocation="BaciaParSulEstRioJaneiro.xsd"/>
    <include schemaLocation="MunicipiosEstRioJaneiro.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="BaseData" type="wisesxmls:BaseSchemaType" substitutionGroup="wisesxmls:_BaseData">
        <annotation>
            <appinfo>
                <element name="WISE=BaseData=Source">"ENCE/IBGE"</element>
                <element name="WISE=Description">"Municípios do estado do Rio de Janeiro com as sub-bacias da
                  Bacia do Rio Paraíba do Sul"
                </element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================ -->
    <element name="BaciaParaibaSulRioJaneiro" substitutionGroup="wisesxmls:_DataSource">
        <annotation>
```

```
            <appinfo>
               <element name="WISE=Description">"Fonte de dados das sub-bacias da Bacia do Rio Paraíba do Sul
                  que passam pelo estado do Rio de Janeiro. Possui informações das sub-bacias da Bacia do Rio
                  Paraiba do Sul, e dos municípios do estado do Rio de Janeiro"
               </element>
            </appinfo>
         </annotation>
         <complexType>
            <complexContent>
               <extension base="wisesxmls:AbstractDataSourceType"/>
            </complexContent>
         </complexType>
      </element>
      <!-- ================================================================== -->
      <element name="DataSets" substitutionGroup="wisesxmls:_DataSets">
         <complexType>
            <complexContent>
               <extension base="wisesxmls:AbstractDataSetsType"/>
            </complexContent>
         </complexType>
      </element>
      <!-- ================================================================== -->
      <element name="BaciaParSulEstRioJaneiro" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
      <!-- ================================================================== -->
      <element name="MunicipiosEstRioJaneiro" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
      <!-- ================================================================== -->
</schema>
```

Listagem 5.1 – Esquema Convencional Aplicação BaciaParaibaSulRioJaneiro.xsd Extraído

O Esquema Georreferenciado Aplicação MunicipiosEstRioJaneiro.xsd extraído, Listagem 5.2, possui informação de Datum e Sistemas de Coordenadas, como exemplos de Elementos Informativos. Ele está usando o datum SAD-69, ao contrário de alguns SXMLSs que estão no Apêndice C que usam o Córrego Alegre. O tipo complexo ApplicationFeatureType possui os campos que foram extraídos do arquivo Shape. A descrição destes campos será adicionada na próxima fase, edição de SXMLSs.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
   <!-- ================================================================
    includes and imports
    ================================================================ -->
   <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
   <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
   <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
   <!-- ================================================================
    global declarations
    ================================================================ -->
   <element name="MunicipiosEstRioJaneiro" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
      <annotation>
         <appinfo>
            <element name="WISE=Description">"Municípios do Estado do Rio de Janeiro"</element>
            <element name="WISE=FeatureCollection=Datum"><![CDATA[D_South_American_1969]]></element>
            <element name="WISE=FeatureCollection=Ellipsoid">
               <![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]>
            </element>
            <element name="WISE=FeatureCollection=CoordinateSystem">
               <![CDATA[GEOGCS["GCS_South_American_1969", DATUM, PRIMEM["Greenwich",0],
               UNIT["Degree",0.0174532925199433]]]]>
            </element>
         </appinfo>
```

```
    </annotation>
  </element>
  <!--================================================================== -->
  <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
  <!-- ====================================================================
   complex types
   ==================================================================== -->
  <complexType name="FeatureCollectionType">
    <complexContent>
      <extension base="gml:AbstractFeatureCollectionType"/>
    </complexContent>
  </complexType>
  <!--================================================================== -->
  <complexType name="ApplicationFeatureType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element ref="gml:geometryMember"/>
          <element name="Area_1" type="wisesxmls:wiseFloat"/>
          <element name="Perimetro_" type="wisesxmls:wiseFloat"/>
          <element name="Geocodigo" type="wisesxmls:wiseInt"/>
          <element name="Nome" type="wisesxmls:wiseString"/>
          <element name="Latitudese" type="wisesxmls:wiseFloat"/>
          <element name="Longitudes" type="wisesxmls:wiseFloat"/>
          <element name="Area_tot_g" type="wisesxmls:wiseFloat"/>
          <element name="Codmeso" type="wisesxmls:wiseInt"/>
          <element name="Nomemeso" type="wisesxmls:wiseString"/>
          <element name="Codmicro" type="wisesxmls:wiseInt"/>
          <element name="Nomemicro" type="wisesxmls:wiseString"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
</schema>
```

Listagem 5.2 – Esquema Georreferenciado Aplicação MunicipiosEstRioJaneiro.xsd Extraído

5.2.2 Edição de SXMLSs

Após terem sido extraídos os esquemas das fontes de dados, pode-se editá-los adicionando mais descrições e semântica a estes. A ferramenta de edição de SXMLS do Gerador de XML Schema Semântico é usada para este fim, como mostrado na Figura 5.3, ao editar o SXMLS MunicipiosEstRioJaneiro.xsd da Listagem 5.2. Desta forma as informações descritivas, através dos Elementos Informativos definidos neste trabalho, são adicionadas aos esquemas SXMLS. Neste estudo de caso somente serão adicionadas informações descritivas e semânticas aos Esquemas Georreferenciados Aplicação como pode ser visto na Listagem 5.3 e nos SXMLSs do Apêndice C.

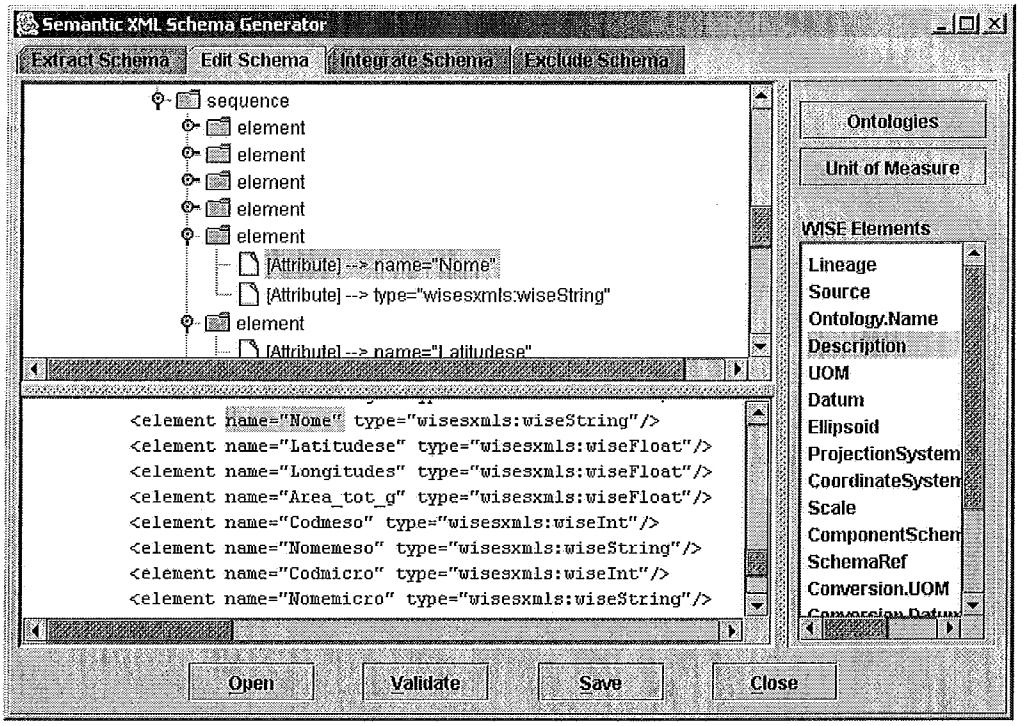


Figura 5.3 – Interface de edição de SXMLS do Gerador de XML Schema Semântico

A Listagem 5.3 mostra o SXMLS MunicipiosEstRioJaneiro.xsd com a adição de Elementos Informativos, Ontology.Name, Description, UnitOfMeasure. Estes elementos poderão ser usados no momento da integração de esquemas.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="MunicipiosEstRioJaneiro" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Municípios do Estado do Rio de Janeiro"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_South_American_1969]]></element>
                <element name="WISE=FeatureCollection=Ellipsoid">
                    <![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]>
                </element>
                <element name="WISE=FeatureCollection=CoordinateSystem">
                    <![CDATA[GEOGCS["GCS_South_American_1969", DATUM, PRIMEM["Greenwich",0],
                    UNIT["Degree",0.0174532925199433]]]]>
                </element>
                <element name="WISE=Ontology.Name">"MunicipiosEstRioJaneiro", "Municípios", "Cidades", "Estado
                    Rio de Janeiro"
                </element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================ -->
```

```
<element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
<!-- ==========================================================================
 complex types
 ========================================================================== -->
<complexType name="FeatureCollectionType">
   <complexContent>
      <extension base="gml:AbstractFeatureCollectionType"/>
   </complexContent>
</complexType>
<!-- ========================================================================== -->
<complexType name="ApplicationFeatureType">
   <complexContent>
      <extension base="gml:AbstractFeatureType">
         <sequence>
            <element ref="gml:geometryMember"/>
            <element name="Area_1" type="wisesxmls:wiseFloat">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Área do município"</element>
                     <element name="WISE=Ontology.Name">"Area_1", "Área", "Área Município",
                        "Extensão"
                     </element>
                     <element name="WISE=Field=UnitOfMeasure">"Km2"</element>
                  </appinfo>
               </annotation>
            </element>
            <element name="Perimetro_" type="wisesxmls:wiseFloat">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Perímetro do município"</element>
                     <element name="WISE=Ontology.Name">"Perimetro_", "Perímetro"</element>
                     <element name="WISE=Field=UnitOfMeasure">"Km"</element>
                  </appinfo>
               </annotation>
            </element>
            <element name="Geocodigo" type="wisesxmls:wiseInt">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Código que associa a representação gemétrica de
                        um objeto a um registro na tabela"
                     </element>
                     <element name="WISE=Ontology.Name">"Geocodigo", "Código"</element>
                  </appinfo>
               </annotation>
            </element>
            <element name="Nome" type="wisesxmls:wiseString">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Nome do município"</element>
                     <element name="WISE=Ontology.Name">"Nome", "Município", "Cidade"</element>
                  </appinfo>
               </annotation>
            </element>
            <element name="Latitudese" type="wisesxmls:wiseFloat">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Latitude"</element>
                     <element name="WISE=Ontology.Name">"Latitudese", "Latitude"</element>
                  </appinfo>
               </annotation>
            </element>
            <element name="Longitudes" type="wisesxmls:wiseFloat">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Longitude"</element>
                     <element name="WISE=Ontology.Name">"Longitudes", "Longitude"</element>
                  </appinfo>
               </annotation>
            </element>
            <element name="Area_tot_g" type="wisesxmls:wiseFloat">
               <annotation>
                  <appinfo>
                     <element name="WISE=Description">"Área do Município, adquirida através de medições do
                        terreno"
                     </element>
                     <element name="WISE=Ontology.Name">"Area_tot_g", "Área", "Área Município", "Área
```

```
                    Real"
                </element>
                <element name="WISE=Field=UnitOfMeasure">"Km2"</element>
            </appinfo>
        </annotation>
    </element>
    <element name="Codmeso" type="wisesxmls:wiseInt">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Código da mesorregião"</element>
                <element name="WISE=Ontology.Name">"Codmeso", "Código"</element>
            </appinfo>
        </annotation>
    </element>
    <element name="Nomemeso" type="wisesxmls:wiseString">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Nome da mesorregião"</element>
                <element name="WISE=Ontology.Name">"Nomemeso", "Mesorregião"</element>
            </appinfo>
        </annotation>
    </element>
    <element name="Codmicro" type="wisesxmls:wiseInt">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Código da microrregião"</element>
                <element name="WISE=Ontology.Name">"Codmicro", "Código"</element>
            </appinfo>
        </annotation>
    </element>
    <element name="Nomemicro" type="wisesxmls:wiseString">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Nome da microrregião"</element>
                <element name="WISE=Ontology.Name">"Nomemicro", "Microrregião"</element>
            </appinfo>
        </annotation>
    </element>
  </sequence>
 </extension>
</complexContent>
</complexType>
</schema>
```

Listagem 5.3 – Esquema Georreferenciado Aplicação MunicipiosEstRioJaneiro.xsd com Elementos Informativos

5.2.3 Integração de SXMLSs

A integração de SXMLSs se baseia nos Elementos Informativos para ter um bom resultado. Como os Elementos Informativos já foram adicionados aos SXMLS, estes serão agora integrados em um esquema chamado ISXMLS.

A interface de integração de SXMLSs do Gerador de XML Schema é mostrada na Figura 5.4, onde aparecem os dois Esquemas Convencionais Aplicação a serem integrados, BaciaParaibaSulRioJaneiro e BaciaRioParaibaSul, em conjunto com o nome do ISXMLS a ser gerado, IntegradoBaciaRioParaibaSul. Após esta interface aparecem outras interfaces de dialogo, à medida que os SXMLSs vão sendo integrados, que

ajudam o usuário a resolver conflitos como: nomes, unidades de medida e outros, e quais conversões devem ser realizadas.

O ISXMLS também fica armazenado no Catálogo, estando associado aos seus esquemas componentes e às suas fontes de dados.

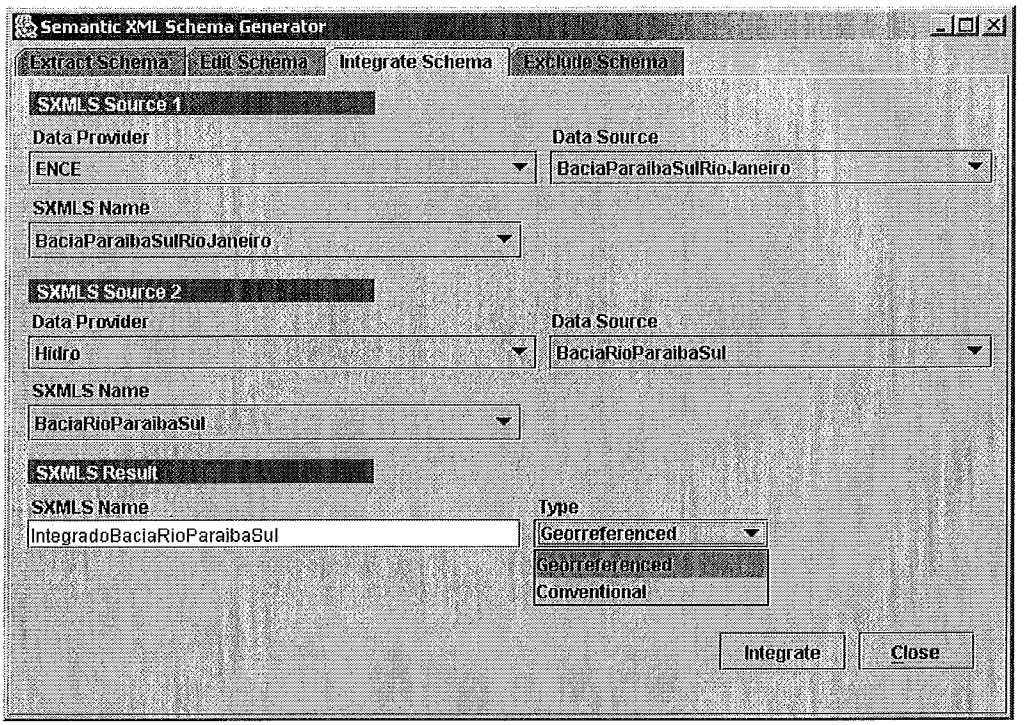


Figura 5.4 – Interface de integração de SXMLS do Gerador de XML Schema Semântico

Terminado o processo de integração de esquemas, que se iniciou como mostrado na Figura 5.4, o ISXMLS que foi gerado, neste caso, o Esquema Integrado Georreferenciado é apresentado na Listagem 5.4, e um dos seus Esquemas Georreferenciados Aplicação, Municipios_Integ.xsd, é mostrado na Listagem 5.5, estando os outros três (BacParSul_Integ.xsd, DistritBacParSul_Integ.xsd, SolosBacParSul_Integ.xsd) no Apêndice C.

Os SXMLSs resultantes sofreram algumas conversões de Sistema de Projeção e Coordenadas, e de Unidades de Medidas, realizados nesta ferramenta. Estas informações estão representadas nos Elementos Informativos de conversão, como Conversion.UnitOfMeasure, Conversion.Datum entre outros.

Também ocorreram casos de resolução de conflitos de nomes, como podem ser vistos na Listagem 5.5, SXMLS Municipios_Integ.xsd, e no Apêndice C.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ======================================================================
     includes and imports
     ====================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <include schemaLocation="SXMLS-BaseIntegrated.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <include schemaLocation="BacParSul_Integ.xsd"/>
    <include schemaLocation="Municipios_Integ.xsd"/>
    <include schemaLocation="DistritBacParSul_Integ.xsd"/>
    <include schemaLocation="SolosBacParSul_Integ.xsd"/>
    <!-- ======================================================================
     global declarations
     ====================================================================== -->
    <element name="IntegratedData" type="wisesxmls:IntegratedSchemaType"
substitutionGroup="wisesxmls:_IntegratedData">
        <annotation>
            <appinfo>
                <element name="WISE=BaseData=Source">"Laboratório de Hidrologia - COPPE/UFRJ",
                    "ENCE/IBGE"
                </element>
                <element name="WISE=Description">"Integração dos dados da Bacia do Rio Paraíba do Sul e
                    municípios do estado do Rio de Janeiro"
                </element>
                <element name="WISE=IntegratedData=ComponentSchema">"C1", "BaciaParaibaSulRioJaneiro.xsd"
                </element>
                <element name="WISE=IntegratedData=ComponentSchema">"C2", "BaciaRioParaibaSul.xsd"
                </element>
            </appinfo>
        </annotation>
    </element>
    <!-- ====================================================================== -->
    <element name="DataSets" substitutionGroup="wisesxmls:_DataSets">
        <complexType>
            <complexContent>
                <extension base="wisesxmls:AbstractDataSetsType"/>
            </complexContent>
        </complexType>
    </element>
    <!-- ====================================================================== -->
    <element name="BacParSul_Integ" type="wisesxmls:FeatureCollectionType"
      substitutionGroup="gml:_FeatureCollection"/>
    <!-- ====================================================================== -->
    <element name="Municipios_Integ" type="wisesxmls:FeatureCollectionType"
      substitutionGroup="gml:_FeatureCollection"/>
    <!-- ====================================================================== -->
    <element name="DistritBacParSul_Integ" type="wisesxmls:FeatureCollectionType"
      substitutionGroup="gml:_FeatureCollection"/>
    <!-- ====================================================================== -->
    <element name="SolosBacParSul_Integ" type="wisesxmls:FeatureCollectionType"
      substitutionGroup="gml:_FeatureCollection"/>
    <!-- ====================================================================== -->
</schema>
```

Listagem 5.4 –Esquema Integrado Georreferenciado IntegradoBaciaRioParaibaSul.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <!-- ======================================================================
     includes and imports
     ====================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ======================================================================
```

```
 global declarations
 ========================================================================== -->
<element name="Municipios_Integ" type="wisesxmls:FeatureCollectionType"
  substitutionGroup="gml:_FeatureCollection">
    <annotation>
      <appinfo>
        <element name="WISE=Description">"Integração dos municípios do Estado do Rio de Janeiro com os
          municípios da Bacia do Rio Paraíba do Sul"
        </element>
        <element name="WISE=SchemaRef">
          "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro",
          "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul"
        </element>
        <element name="WISE=FeatureCollection=Conversion.Datum">
          "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul", <![CDATA[D_South_American_1969]]>
        </element>
        <element name="WISE=FeatureCollection=Conversion.Ellipsoid">
          "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul",
          <![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]>
        </element>
        <element name="WISE=FeatureCollection=Conversion.CoordinateSystem">
          "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul",
          <![CDATA[GEOGCS["GCS_South_American_1969", DATUM, PRIMEM["Greenwich",0],
          UNIT["Degree",0.0174532925199433]]]]>
        </element>
        <element name="WISE=Ontology.Name">"Municipios_Integ", "Municípios", "Cidades"</element>
      </appinfo>
    </annotation>
</element>
<!--========================================================================== -->
<element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
<!-- ========================================================================
 complex types
 ========================================================================== -->
<complexType name="FeatureCollectionType">
    <complexContent>
      <extension base="gml:AbstractFeatureCollectionType"/>
    </complexContent>
</complexType>
<!--========================================================================== -->
<complexType name="ApplicationFeatureType">
    <complexContent>
      <extension base="gml:AbstractFeatureType">
        <sequence>
          <element ref="gml:geometryMember">
            <annotation>
              <appinfo>
                <element name="WISE=SchemaRef">
              "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/geometryMember",
              "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/geometryMember"
                </element>
              </appinfo>
            </annotation>
          </element>
          <element name="Area" type="wisesxmls:wiseFloat">
            <annotation>
              <appinfo>
                <element name="WISE=Description">"Área do município"</element>
                <element name="WISE=Ontology.Name">"Area", "Área", "Área Município", "Extensão"
                </element>
                <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                <element name="WISE=SchemaRef">
                  "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Area_1",
                  "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/Area"
                </element>
                <element name="WISE=Field=Conversion.UnitOfMeasure">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Area_1", "m2"
                </element>
              </appinfo>
            </annotation>
          </element>
          <element name="Perimetro" type="wisesxmls:wiseFloat">
            <annotation>
              <appinfo>
                <element name="WISE=Description">"Perímetro do município"</element>
                <element name="WISE=Ontology.Name">"Perimetro", "Perímetro"</element>
```

```
            <element name="WISE=Field=UnitOfMeasure">"m"</element>
            <element name="WISE=SchemaRef">
              "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Perimetro_",
              "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/Perimeter"
            </element>
            <element name="WISE=Field=Conversion.UnitOfMeasure">
          "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Perimetro_", "m"
            </element>
          </appinfo>
      </annotation>
  </element>
  <element name="Nome" type="wisesxmls:wiseString">
      <annotation>
          <appinfo>
              <element name="WISE=Description">"Nome do município"</element>
              <element name="WISE=Ontology.Name">"Nome", "Município", "Cidade"</element>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Nome",
                "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/Nome_muni"</element>
          </appinfo>
      </annotation>
  </element>
  <element name="Geocodigo" type="wisesxmls:wiseInt">
      <annotation>
          <appinfo>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Geocodigo"
              </element>
          </appinfo>
      </annotation>
  </element>
  <element name="Latitudese" type="wisesxmls:wiseFloat">
      <annotation>
          <appinfo>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Latitudese"
              </element>
          </appinfo>
      </annotation>
  </element>
  <element name="Longitudes" type="wisesxmls:wiseFloat">
      <annotation>
          <appinfo>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Longitudes"
              </element>
          </appinfo>
      </annotation>
  </element>
  <element name="Area_tot_g" type="wisesxmls:wiseFloat">
      <annotation>
          <appinfo>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Area_tot_g"
              </element>
          </appinfo>
      </annotation>
  </element>
  <element name="Codmeso" type="wisesxmls:wiseInt">
      <annotation>
          <appinfo>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Codmeso"
              </element>
          </appinfo>
      </annotation>
  </element>
  <element name="Nomemeso" type="wisesxmls:wiseString">
      <annotation>
          <appinfo>
              <element name="WISE=SchemaRef">
                "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Nomemeso"
              </element>
          </appinfo>
      </annotation>
  </element>
```

```
            <element name="Codmicro" type="wisesxmls:wiseInt">
              <annotation>
                <appinfo>
                  <element name="WISE=SchemaRef">
                    "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Codmicro"
                  </element>
                </appinfo>
              </annotation>
            </element>
            <element name="Nomemicro" type="wisesxmls:wiseString">
              <annotation>
                <appinfo>
                  <element name="WISE=SchemaRef">
                    "C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/MunicipiosEstRioJaneiro/Nomemicro"
                  </element>
                </appinfo>
              </annotation>
            </element>
            <element name="Munbacparsul_" type="wisesxmls:wiseInt">
              <annotation>
                <appinfo>
                  <element name="WISE=SchemaRef">
                    "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/Munbacparsul_"
                  </element>
                </appinfo>
              </annotation>
            </element>
            <element name="Munbacparsul_id" type="wisesxmls:wiseInt">
              <annotation>
                <appinfo>
                  <element name="WISE=SchemaRef">
                    "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/Munbacparsul_id"
                  </element>
                </appinfo>
              </annotation>
            </element>
            <element name="Nome_est" type="wisesxmls:wiseString">
              <annotation>
                <appinfo>
                  <element name="WISE=SchemaRef">
                    "C2/BaseData/BaciaRioParaibaSul/DataSets/MunBacParSul/Nome_est"
                  </element>
                </appinfo>
              </annotation>
            </element>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
</schema>
```

Listagem 5.5 –Esquema Georreferenciado Aplicação Municipios_Integ.xsd do Esquema Integrado Georreferenciado

Para verificar o resultado da integração de esquemas, foram gerados alguns arquivos ESRI Shape de exemplo de acordo com a estrutura dos SXMLS integrados, ou seja, um exemplo mostrando os Shapes de uma integração de dados à partir da integração de esquemas que foi realizada anteriormente.

A Figura 5.5 mostra a representação gráfica destes arquivos para visualizar o resultado da integração dos Esquemas Georreferenciados Aplicação do ISXMLS IntegradoBaciaRioParaibaSul.xsd.

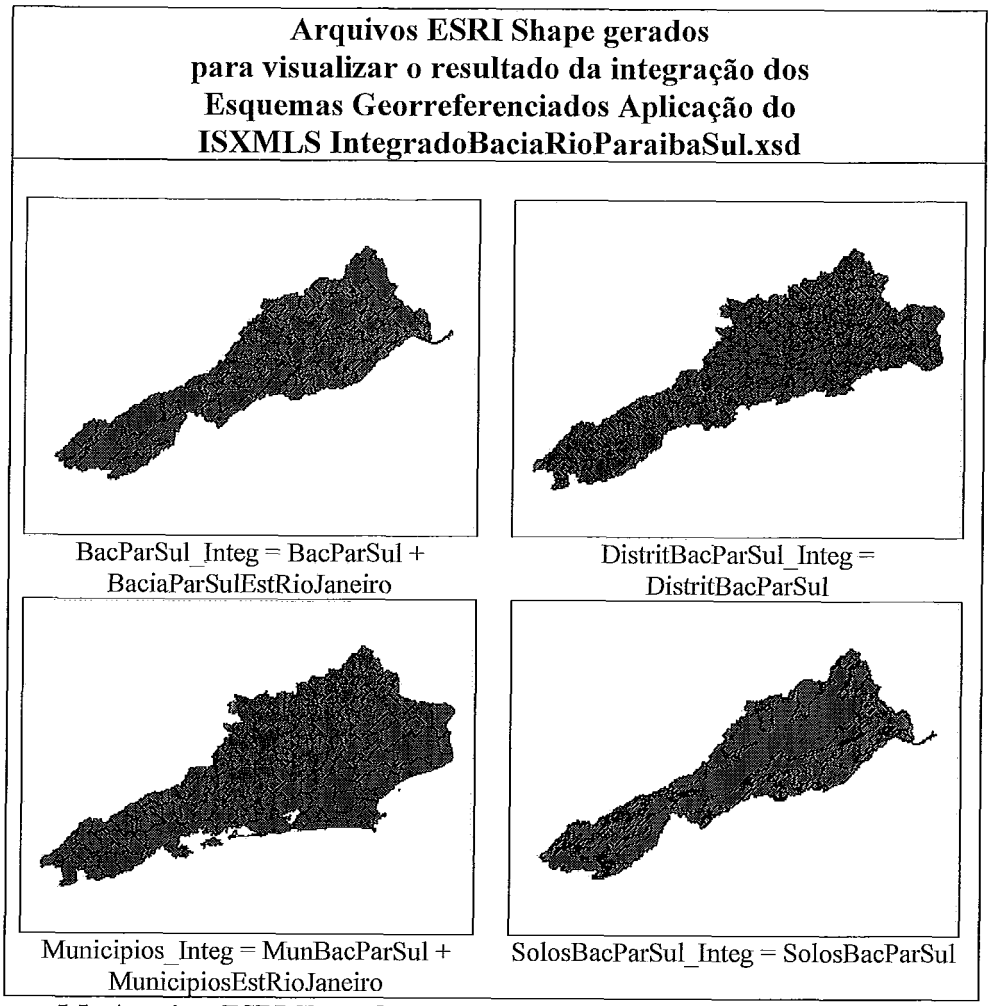


Figura 5.5 –Arquivos ESRI Shape de exemplo para visualizar o resultado da integração de esquemas

5.2.4 Consulta a um SXMLS

Até esta fase foram tratados assuntos relacionados à geração de SXMLS. Com os SXMLS gerados podem ser realizadas consultas sobre as fontes de dados através da Interface de Consulta WISE (ICWISE). Os SXMLSs gerados podem ser visualizados através de um Metamodelo que pode ser acessado no Visualizador de Metamodelos da ICWISE. Com este Metamodelo pode-se ver como estão distribuídas as fontes de dados e analisadas informações descritivas e semânticas, que depois podem servir para realizar consultas através do Editor de Consultas da ICWISE. O Editor de Consulta também possui uma interface que possibilita a visualização da estrutura dos dados, de forma mais restrita. Como o Visualizador de Metamodelos ainda não está implementado, será utilizado simplesmente o Editor de Consultas para realizar consultas em um SXMLS. E os resultados destas consultas são apresentados no Publicador de Dados da ICWISE.

A Figura 5.6 mostra a interface do Editor de Consultas com uma consulta, informando que devem ser visualizados alguns campos a partir de um critério de seleção. A mesma consulta é representada de forma textual em SQL. Esta consulta pede que sejam visualizados os campos Nome, Latitudese, Longitudes, Nomemeso, Nomemicro, Area e Perimetro da coleção de feições que é representada por Municípios_Integ (que foi integrado e é mostrado na Listagem 5.5), e que somente sejam selecionados os registros que possuam Nomemeso (Nome da mesorregião) igual a "Sul Fluminense". O formato de saída utilizado nesta consulta será GML como pode ser observado na Figura 5.6.

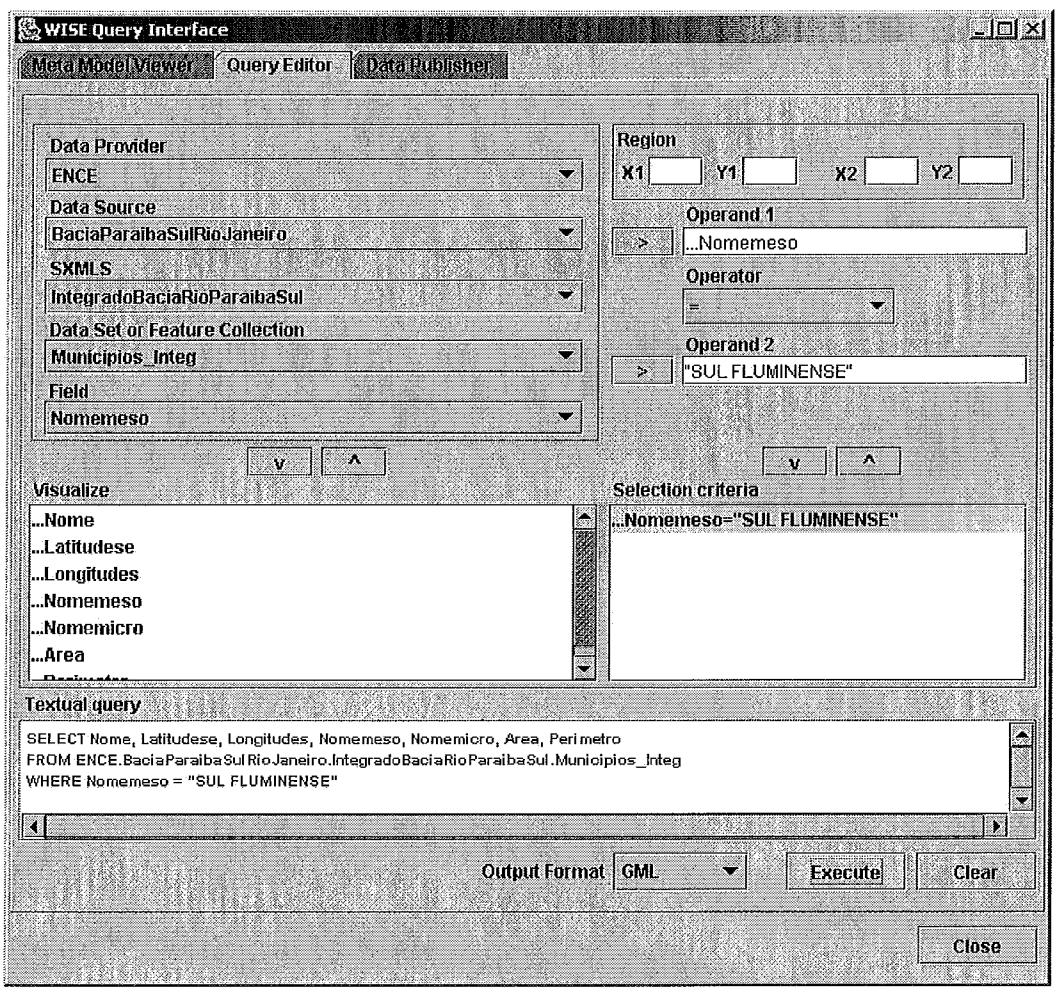


Figura 5.6 – Interface do Editor de Consultas da ICWISE

O resultado da consulta da Figura 5.6 é exibido no Publicador de Dados, Figura 5.7. O Publicador de Dados possui algumas funcionalidades básicas para visualização de arquivos Shape e GML, e também possibilita que estes sejam armazenados ou

abertos localmente. O resultado apresentado no Publicador de Dados foi armazenado localmente no arquivo sulfluminense.gml, como pode ser visto na Listagem 5.6.

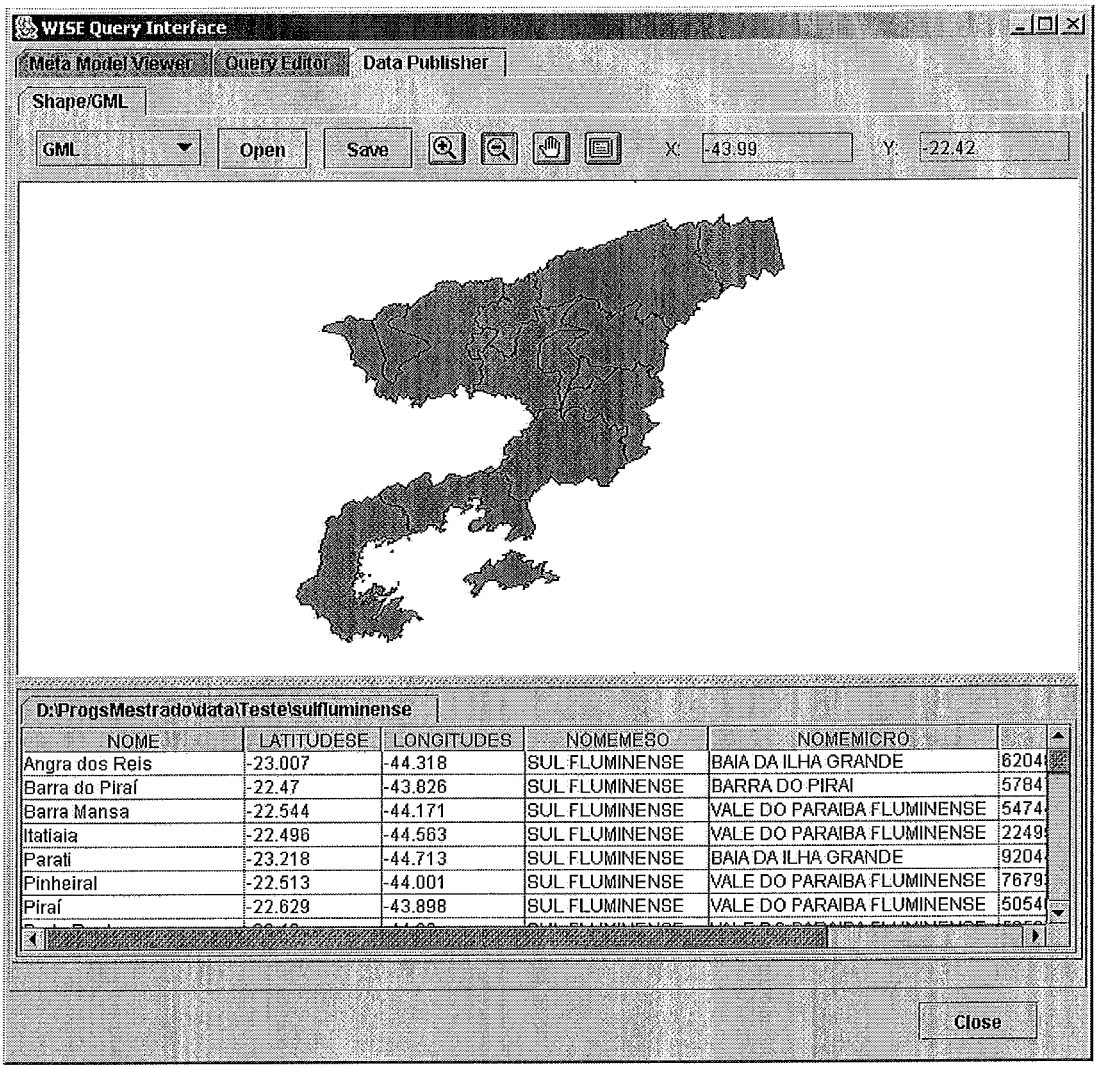


Figura 5.7 – Interface do Publicador de Dados da ICWISE

A Listagem 5.6 apresenta o resultado da consulta armazenado em um arquivo GML, que também poderia ter sido armazenado como arquivo ESRI Shape. Como o arquivo é muito extenso, somente foi mostrado a sua parte inicial (três primeiros municípios) e final. Cabe ressaltar que este arquivo foi gerado seguindo a ordem e os critérios de seleção dos elementos da consulta e não a ordem que estava definida no SXMLS Municípios_Integ.xsd.

Este arquivo, Listagem 5.6, apresenta diversos elementos de GML. O elemento *Envelope* define o menor retângulo envolvente (MBR), com dois pares de coordenadas, desta coleção de feições. Cada elemento *featureMember* define uma feição da coleção de feições, possuindo como conteúdo um elemento *geometryMember* e os campos que

estavam inicialmente nos arquivos shape. O elemento *geometryMember* representa um objeto geométrico (ou geográfico), definido por um conjunto de pontos (coordenadas x,y) em seqüência separados por espaço.


```
<?xml version="1.0" encoding="UTF-8"?>
<Municipios_Integ xmlns="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance" xmlns:xlink="http://www.w3.org/1999/xlink"
xsi:schemaLocation="http://www.cos.ufrj.br/bd/XMLSchemaSemantico Municipios_Integ.xsd">
    <gml:boundedBy>
        <gml:Envelope>
            <gml:coordinates>-44.88887993017323,-23.36816602988057 -43.4030221826001,-
22.05202516130729</gml:coordinates>
        </gml:Envelope>
    </gml:boundedBy>
    <gml:featureMember>
        <Feature fid="0">
            <gml:geometryMember>
                <gml:Polygon srsName="null">
                    <gml:outerBoundaryIs>
                        <gml:LinearRing>
                            <gml:coordinates cs="," decimal="." ts=" ">-44.639,-22.914 -44.635,-22.907 -44.629,-22.905 -
44.607,-22.886 -44.601,-22.889 -44.576,-22.879 -44.555,-22.889 -44.544,-22.885 -44.546,-22.881 -44.533,-22.869 -
44.519,-22.865 -44.506,-22.852 -44.495,-22.846 -44.484,-22.846 -44.48,-22.851 -44.476,-22.856 -44.481,-22.866 -
44.476,-22.868 -44.477,-22.874 -44.472,-22.876 -44.467,-22.885 -44.437,-22.871 -44.427,-22.856 -44.422,-22.851 -
44.413,-22.848 -44.403,-22.849 -44.402,-22.855 -44.392,-22.853 -44.387,-22.858 -44.374,-22.858 -44.327,-22.838 -
44.32,-22.84 -44.322,-22.845 -44.311,-22.844 -44.302,-22.854 -44.296,-22.845 -44.282,-22.842 -44.209,-22.846 -
44.261,-22.848 -44.248,-22.857 -44.245,-22.868 -44.239,-22.872 -44.231,-22.871 -44.238,-22.886 -44.237,-22.897 -
44.242,-22.902 -44.237,-22.905 -44.242,-22.913 -44.223,-22.929 -44.215,-22.932 -44.124,-22.928 -44.188,-22.94 -
44.189,-22.952 -44.163,-23.009 -44.166,-23.015 -44.167,-23.033 -44.177,-23.038 -44.176,-23.042 -44.172,-23.043 -
44.173,-23.049 -44.179,-23.05 -44.183,-23.045 -44.185,-23.053 -44.194,-23.055 -44.195,-23.051 -44.193,-23.042 -
44.2,-23.039 -44.238,-23.057 -44.244,-23.054 -44.246,-23.05 -44.25,-23.052 -44.251,-23.049 -44.244,-23.045 -44.236,-
23.026 -44.222,-23.017 -44.222,-23.012 -44.231,-23.014 -44.243,-23 -44.249,-23.001 -44.248,-22.996 -44.261,-23.008 -
44.267,-23.003 -44.274,-23.007 -44.279,-23.014 -44.278,-23.018 -44.285,-23.015 -44.289,-23.017 -44.291,-23.024 -
44.299,-23.026 -44.306,-23.024 -44.298,-23.007 -44.304,-23.003 -44.308,-23.007 -44.317,-23.008 -44.316,-23.014 -
44.32,-23.01 -44.323,-23.016 -44.328,-23.015 -44.327,-23.021 -44.332,-23.021 -44.336,-23.026 -44.341,-23.023 -
44.351,-23.031 -44.351,-23.023 -44.358,-23.023 -44.358,-23.02 -44.363,-23.018 -44.36,-23.011 -44.363,-23.01 -
44.357,-23.008 -44.355,-22.999 -44.345,-22.991 -44.328,-22.994 -44.329,-22.983 -44.324,-22.983 -44.321,-22.986 -
44.31,-22.98 -44.312,-22.973 -44.306,-22.964 -44.302,-22.963 -44.3,-22.959 -44.302,-22.957 -44.313,-22.96 -44.319,-
22.957 -44.33,-22.961 -44.333,-22.967 -44.336,-22.964 -44.337,-22.957 -44.329,-22.954 -44.333,-22.954 -44.333,-
22.945 -44.323,-22.942 -44.32,-22.937 -44.326,-22.937 -44.323,-22.929 -44.328,-22.926 -44.327,-22.924 -44.332,-
22.928 -44.34,-22.922 -44.348,-22.922 -44.353,-22.931 -44.359,-22.923 -44.364,-22.924 -44.361,-22.926 -44.366,-
22.931 -44.36,-22.932 -44.371,-22.938 -44.37,-22.945 -44.367,-22.948 -44.364,-22.94 -44.35,-22.946 -44.359,-22.957 -
44.363,-22.955 -44.367,-22.96 -44.373,-22.959 -44.374,-22.967 -44.378,-22.97 -44.381,-22.967 -44.384,-22.972 -
44.387,-22.969 -44.378,-22.96 -44.392,-22.95 -44.407,-22.953 -44.412,-22.942 -44.414,-22.946 -44.423,-22.951 -
44.422,-22.96 -44.426,-22.957 -44.426,-22.963 -44.434,-22.963 -44.436,-22.973 -44.431,-22.978 -44.435,-22.98 -
44.435,-22.988 -44.441,-22.992 -44.439,-22.996 -44.43,-22.993 -44.425,-22.998 -44.439,-23.003 -44.446,-23.012 -
44.44,-23.021 -44.431,-23.021 -44.445,-23.028 -44.453,-23.017 -44.46,-23.019 -44.459,-23.011 -44.463,-23.006 -
44.468,-23.006 -44.469,-23.011 -44.477,-23.006 -44.484,-23.009 -44.489,-23.021 -44.506,-23.031 -44.511,-23.027 -
44.519,-23.028 -44.531,-23.021 -44.537,-23.025 -44.547,-23.021 -44.544,-23.009 -44.549,-23.01 -44.549,-23.006 -
44.546,-23.006 -44.541,-22.999 -44.545,-22.992 -44.543,-22.986 -44.55,-22.973 -44.555,-22.974 -44.56,-22.972 -
44.557,-22.963 -44.562,-22.955 -44.577,-22.947 -44.595,-22.948 -44.606,-22.944 -44.61,-22.939 -44.621,-22.939 -
44.63,-22.918 -44.639,-22.914</gml:coordinates>
                        </gml:LinearRing>
                    </gml:outerBoundaryIs>
                </gml:Polygon>
            </gml:geometryMember>
            <NOME><![CDATA[Angra dos Reis]]></NOME>
            <LATITUDESE><![CDATA[-23.007]]></LATITUDESE>
            <LONGITUDES><![CDATA[-44.318]]></LONGITUDES>
            <NOMEMESO><![CDATA[SUL FLUMINENSE]]></NOMEMESO>
            <NOMEMICRO><![CDATA[BAIA DA ILHA GRANDE]]></NOMEMICRO>
            <AREA><![CDATA[620485.0]]></AREA>
            <PERIMETRO><![CDATA[225742.0]]></PERIMETRO>
        </Feature>
    </gml:featureMember>
    <gml:featureMember>
        <Feature fid="1">
            <gml:geometryMember>
                <gml:Polygon srsName="null">
                    <gml:outerBoundaryIs>
                        <gml:LinearRing>
```

<gml:coordinates cs="," decimal="." ts=" ">-44.031,-22.497 -44.054,-22.48 -44.056,-22.471 -44.062,-22.463 -44.071,-22.46 -44.05,-22.445 -44.037,-22.43 -44.026,-22.425 -44.003,-22.409 -44.005,-22.407 -44.085,-22.383 -44.085,-22.376 -44.076,-22.357 -44.069,-22.353 -44.064,-22.336 -44.023,-22.304 -44.025,-22.296 -44.023,-22.295 -44.017,-22.306 -44.009,-22.31 -43.988,-22.309 -43.975,-22.312 -43.967,-22.344 -43.959,-22.339 -43.952,-22.344 -43.951,-22.35 -43.941,-22.349 -43.937,-22.345 -43.93,-22.352 -43.933,-22.359 -43.938,-22.357 -43.948,-22.367 -43.941,-22.372 -43.941,-22.38 -43.915,-22.351 -43.894,-22.345 -43.885,-22.33 -43.87,-22.326 -43.856,-22.315 -43.849,-22.316 -43.842,-22.328 -43.85,-22.337 -43.858,-22.373 -43.848,-22.369 -43.845,-22.384 -43.837,-22.386 -43.837,-22.375 -43.827,-22.373 -43.81,-22.353 -43.803,-22.357 -43.796,-22.357 -43.797,-22.364 -43.784,-22.362 -43.78,-22.368 -43.776,-22.368 -43.775,-22.376 -43.766,-22.39 -43.749,-22.404 -43.755,-22.408 -43.762,-22.424 -43.772,-22.437 -43.779,-22.438 -43.778,-22.441 -43.767,-22.444 -43.765,-22.449 -43.759,-22.45 -43.761,-22.457 -43.755,-22.46 -43.766,-22.459 -43.77,-22.518 -43.778,-22.532 -43.792,-22.535 -43.798,-22.552 -43.809,-22.554 -43.812,-22.548 -43.818,-22.548 -43.817,-22.536 -43.82,-22.534 -43.805,-22.521 -43.812,-22.512 -43.824,-22.517 -43.84,-22.51 -43.845,-22.504 -43.855,-22.507 -43.854,-22.517 -43.865,-22.524 -43.865,-22.529 -43.913,-22.521 -43.944,-22.515 -43.952,-22.519 -43.981,-22.516 -43.992,-22.517 -43.999,-22.514 -44.003,-22.509 -44.004,-22.5 -44.021,-22.484 -44.027,-22.486 -44.031,-22.497</gml:coordinates>

```
                    </gml:LinearRing>
                </gml:outerBoundaryIs>
            </gml:Polygon>
        </gml:geometryMember>
        <NOME><![CDATA[Barra do Piraí]]></NOME>
        <LATITUDESE><![CDATA[-22.47]]></LATITUDESE>
        <LONGITUDES><![CDATA[-43.826]]></LONGITUDES>
        <NOMEMESO><![CDATA[SUL FLUMINENSE]]></NOMEMESO>
        <NOMEMICRO><![CDATA[BARRA DO PIRAI]]></NOMEMICRO>
        <AREA><![CDATA[578471.0]]></AREA>
        <PERIMETRO><![CDATA[152458.0]]></PERIMETRO>
    </Feature>
</gml:featureMember>
<gml:featureMember>
    <Feature fid="2">
        <gml:geometryMember>
            <gml:Polygon srsName="null">
                <gml:outerBoundaryIs>
                    <gml:LinearRing>
```

<gml:coordinates cs="," decimal="." ts=" ">-44.344,-22.588 -44.347,-22.574 -44.329,-22.573 -44.33,-22.569 -44.325,-22.565 -44.326,-22.554 -44.321,-22.551 -44.328,-22.52 -44.325,-22.519 -44.325,-22.514 -44.321,-22.513 -44.32,-22.508 -44.323,-22.499 -44.331,-22.494 -44.33,-22.492 -44.337,-22.47 -44.333,-22.467 -44.32,-22.469 -44.308,-22.453 -44.302,-22.448 -44.287,-22.459 -44.259,-22.475 -44.257,-22.483 -44.242,-22.487 -44.234,-22.484 -44.231,-22.477 -44.222,-22.48 -44.225,-22.474 -44.222,-22.473 -44.218,-22.464 -44.222,-22.464 -44.226,-22.459 -44.226,-22.455 -44.22,-22.453 -44.222,-22.435 -44.218,-22.433 -44.217,-22.423 -44.211,-22.419 -44.214,-22.413 -44.212,-22.404 -44.205,-22.393 -44.208,-22.387 -44.203,-22.384 -44.187,-22.381 -44.172,-22.375 -44.157,-22.376 -44.155,-22.373 -44.169,-22.333 -44.147,-22.331 -44.141,-22.338 -44.124,-22.339 -44.117,-22.33 -44.096,-22.33 -44.068,-22.316 -44.053,-22.314 -44.047,-22.307 -44.025,-22.296 -44.023,-22.304 -44.064,-22.336 -44.069,-22.353 -44.076,-22.357 -44.085,-22.376 -44.085,-22.383 -44.005,-22.407 -44.09,-22.406 -44.096,-22.408 -44.106,-22.419 -44.138,-22.432 -44.152,-22.453 -44.156,-22.495 -44.153,-22.523 -44.142,-22.527 -44.139,-22.532 -44.139,-22.536 -44.115,-22.534 -44.09,-22.53 -44.089,-22.542 -44.086,-22.545 -44.091,-22.56 -44.09,-22.564 -44.098,-22.577 -44.091,-22.579 -44.097,-22.597 -44.088,-22.604 -44.091,-22.607 -44.086,-22.642 -44.089,-22.649 -44.117,-22.644 -44.148,-22.649 -44.161,-22.678 -44.174,-22.658 -44.19,-22.648 -44.184,-22.637 -44.191,-22.63 -44.198,-22.63 -44.204,-22.627 -44.204,-22.615 -44.208,-22.615 -44.219,-22.618 -44.227,-22.605 -44.233,-22.609 -44.248,-22.607 -44.253,-22.612 -44.271,-22.601 -44.283,-22.604 -44.286,-22.61 -44.295,-22.602 -44.296,-22.598 -44.306,-22.599 -44.313,-22.595 -44.318,-22.597 -44.326,-22.589 -44.34,-22.591 -44.344,-22.588</gml:coordinates>

```
                    </gml:LinearRing>
                </gml:outerBoundaryIs>
            </gml:Polygon>
        </gml:geometryMember>
        <NOME><![CDATA[Barra Mansa]]></NOME>
        <LATITUDESE><![CDATA[-22.544]]></LATITUDESE>
        <LONGITUDES><![CDATA[-44.171]]></LONGITUDES>
        <NOMEMESO><![CDATA[SUL FLUMINENSE]]></NOMEMESO>
        <NOMEMICRO><![CDATA[VALE DO PARAIBA FLUMINENSE]]></NOMEMICRO>
        <AREA><![CDATA[547441.0]]></AREA>
        <PERIMETRO><![CDATA[177580.0]]></PERIMETRO>
    </Feature>
</gml:featureMember>

... PEDAÇO REMOVIDO DO ARQUIVO...

</Municipios_Integ>
```

Listagem 5.6 – Resultado da consulta armazenado no arquivo sulfluminense.gml

Capítulo 6 – Conclusões e Perspectivas Futuras

6.1 Conclusões

Foi apresentado neste trabalho uma arquitetura para um sistema de integração e publicação de dados georreferenciados na Web, chamado WISE.

Esta arquitetura utiliza um MDC chamado XML Schema Semântico que foi desenvolvido neste trabalho usando as tecnologias XML e GML, e um conjunto de Elementos Informativos. O XML Schema Semântico mapeia o conteúdo dos esquemas das fontes de dados locais que são heterogêneas para um modelo de dados que é comum ao WISE facilitando o processo de integração de dados. A utilização de GML como MDC para dados georreferenciados é recente, e são pouquíssimos trabalhos que a utilizam atualmente para este fim, devido talvez a sua complexidade, grande abrangência e pouca documentação. Porém devido ao seu poder de representação de dados georreferenciados, a sua utilização como MDC deve aumentar, assim como está ocorrendo para a representação e transferência de dados georreferenciados.

A arquitetura WISE possui diversos componentes que atuam nas suas quatro camadas: Camada Cliente, Camada de Integração, Camada de Tradutores e Camada de Acesso a Dados. Estes componentes foram projetados com o intuito de serem bastante flexíveis para suportar diversos formatos de dados de saída, bem como suportar diversos tipos de fontes de dados heterogêneas, tornando desta forma o WISE um sistema extensível, não necessitando ser recompilado para agregar novos módulos de formatos de dados ou fontes de dados, somente necessitando a codificação dos novos módulos. Desta forma o WISE acessa fontes de dados convencionais e georreferenciados, em diversos provedores de dados distribuídos geograficamente, através de Tradutores sem a necessidade de ser alterado para que sejam agregados novos Tradutores.

Como o WISE possui um formato próprio de linguagem de consulta para os seus Tradutores (WrapperQuery), se houver a necessidade de ser alterada a linguagem de consulta externa, os tradutores não precisam ser alterados para suportarem esta modificação. Este formato de consulta também pode ser estendido para agregar novas características.

Com o Gerador de XML Schema Semântico, o WISE possui uma ferramenta automática para extrair o conteúdo de um esquema de uma fonte de dados local, bem como ferramentas semi-automáticas para a edição e integração de esquemas SXMLS criados a partir dos esquemas locais. O Gerador de XML Schema Semântico é capaz de realizar a integração de SXMLSs tanto convencionais como georreferenciados ou a combinação dos dois, com o auxílio de ontologias do domínio da aplicação inseridas através de Elementos Informativos.

O WISE também pode ter servidores distribuídos geograficamente formando a rede WISE, que aumenta o seu poder de compartilhamento e integração de dados convencionais e georreferenciados entre diversas instituições.

6.2 Contribuições

Uma das contribuições importantes deste trabalho foi o desenvolvimento do XML Schema Semântico como MDC do WISE e a representação de dados georreferenciados no formato GML.

Outra contribuição importante foi o desenvolvimento do protótipo do WISE com alguns de seus componentes, para integração e publicação de dados convencionais e georreferenciados na Web, que é uma ferramenta extensível em vários aspectos, inclusive no Conversor de Formatos, Publicador de Dados e nos Tradutores, possibilitando utilizar diversos formatos de arquivos e fontes de dados.

Os módulos implementados no protótipo foram: Gerador de XML Schema Semântico com mais detalhes; Tradutores para arquivos ESRI Shape, Excel, e JDBC; Integrador de Dados e Mapas, Editor de Consultas, Publicador de Dados, Gerenciador de Dados do Sistema e Processador de Consultas implementados com um nível menor de detalhes.

O protótipo foi utilizado em um Estudo de Caso real construído para exemplificar uma visão geral da arquitetura, apresentando algumas das diversas funcionalidades implementadas neste trabalho.

Finalmente, o projeto SpeCS recebeu uma ferramenta para integração e publicação de dados convencionais e georreferenciados na web, para auxiliar nas suas tarefas de tomadas de decisão.

6.3 Perspectivas Futuras

O primeiro protótipo foi desenvolvido e testado como foi apresentado no capítulo 5. Foi observado, contudo, que o WISE necessita de complementação, com a implementação dos seguintes quesitos em trabalhos futuros.

Primeiramente deve ser criada uma política de atualização de esquemas SXMLS. Atualmente somente são criados novos esquemas substituindo os anteriores, acarretando com isto a perda das informações descritivas e semânticas (Elementos Informativos) adicionadas manualmente pelos usuários. Com uma política de atualização, estas informações tem grande possibilidade de serem mantidas no novo esquema, bastando ao usuário contemplar somente as modificações ou acréscimos.

Os conflitos de integração de dados resolvidos pelo WISE atualmente são poucos (item 4.5.6.3 deste texto), devendo ser adicionadas novas funções para resolução de outros conflitos como os descritos no capítulo 2. Talvez seja interessante realizar um estudo sobre a possibilidade de modularizar o Gerador de XML Schema Semântico no que tange o processo de integração de esquemas, de modo que quando forem adicionados novos métodos de resolução de conflitos não seja necessário recompilar todo o código, mas sim agregar um novo módulo.

Devem ser adicionados mais operadores espaciais sobre os dados georreferenciados no momento da consulta e da integração de dados, já que os únicos que estão disponíveis são o *region*, para definir o Menor Retângulo Envolvente nas consultas e um operador de união para tratar a integração de mais de uma Coleção de Feições no integração de mapas.

Atualmente as comparações de ontologias na integração de esquemas são realizadas somente entre as ontologias que estão nos esquemas SXMLS, que foram adquiridas no Gerenciador de Ontologias. Como objeto de trabalhos futuros, além destas comparações, devem ser realizadas buscas no Gerenciador de Ontologias por novas ontologias que podem estar associadas às ontologias dos SXMLSs, e que sejam úteis na comparação de nomes no momento da integração de esquemas.

Podem ser implementados novos formatos de saída e tradutores para satisfazer às demandas de usuários de geoprocessamento.

Deve ser estudada a possibilidade de trocar o banco de dados convencional por um banco de dados especial para XML.

Uma outra interface para introduzir consultas espaciais mais complexas através de uma linguagem de consulta, também poderá ser objeto de trabalhos futuros. Como motivação sabe-se que extensões espaciais para SQL têm sido estudadas extensivamente (EGENHOFER, 1994). Spatial SQL (EGENHOFER, 1994) apresenta uma extensão da SQL para consulta e visualização espacial, e GeoSQL (FENG et al., 2000) é uma outra extensão para SIGs orientados a objetos. Tem-se ainda que as consultas espaciais podem ser classificadas em três grandes grupos (BARRERA & BUCHMANN, 1981):

- Consultas espaciais exclusivas: consultas envolvendo somente atributos e relacionamentos espaciais;
- Consultas não-espaciais exclusivas: consultas sobre atributos não-espaciais;
- Consultas mistas: consultas envolvendo atributos não-espaciais, atributos e relacionamentos espaciais.

É um requisito primário que as três categorias de consultas espaciais sejam suportadas em sistemas de informações espaciais. A questão é se deve estender uma linguagem existente ou criar uma nova. Se analisadas detalhadamente as três categorias, a maioria das consultas envolve atributos não-espaciais. Como SQL é um padrão, bem como a linguagem de consulta primária em bancos de dados relacionais, então parece lógico estender SQL ao invés de desenvolver uma nova linguagem. A mesma analogia pode ser aplicada à tecnologia XML. XQuery é uma linguagem de consulta poderosa projetada para consultar e transformar dados XML que são semi-estruturados e aninhados. Assim sendo, XQuery não provê suporte para consultar elementos espaciais. Ela trata elementos espaciais como valores numéricos e nem todos os critérios de pesquisa podem ser expressos como predicados de consulta nestes elementos espaciais individuais.

XQuery é a linguagem de consulta padrão para documentos XML e oferece um rico conjunto de características para trabalhar com consultas envolvendo atributos não-espaciais. Seria útil estender XQuery para suportar consultas espaciais, ao invés de definir uma nova linguagem.

Como bancos de dados relacionais tradicionais somente suportam tipos de dados simples, o Open GIS Consortium definiu uma extensão do modelo de dados da SQL para representar e consultar dados geométricos (OGC, 1999). Esta extensão inclui um conjunto de tipos de dados espaciais e funções de predicados espaciais para simplificar

consultas em campos espaciais. Desta forma, pode ser trazida esta idéia para o projeto de uma linguagem de consulta, que é uma extensão da XQuery, e poderia ser chamada Geo-XQuery.

Para realizar consultas em elementos espaciais, que podem ser tão simples como um ponto ou tão complexos quanto um conjunto de polígonos, a sintaxe da Geo-XQuery proveria um conjunto de predicados espaciais para facilitar consultas envolvendo tais elementos.

Com esta nova linguagem de consultas o Editor de Consultas possuiria uma linguagem poderosa para consultar dados georreferenciados e produzir resultados mais versáteis e complexos em GML.

Referências Bibliográficas

AALDERS, H., 1996, "Quality metrics for GIS". In: *XVII International Symposium on Spatial Data Handling: Advances in GIS Research II*, pp. 277-286.

ABEL, D. J., KILBY, P. J., DBOUK, M., 1994, "The systems integration problem", *International Journal of Geographical Information Systems*, v. 8, pp. 1-12

ABITEBOUL, S., 1997, "Query Semi-Structured Data". In: *International Conference on Database Theory*, pp. 1-18, Delphi, Greece.

ABITEBOUL, S., BUNEMAN, P., SUCIU, D., 2000, *Data on the Web: from Relations to Semistructured Data and XML*. 1 ed. San Francisco, California, USA, Morgan Kaufmann Publishers.

AGUIAR, C. D.,MEDEIROS, C. B., 1996, "Construção de um modelo básico unificado a partir de sistemas stand-alone". In: *Anais do GIS 96*, pp. 503-515, Curitiba.

ARCIMS, 2001, *The ArcIMS 3 Architecture*.In: J-8488. ESRI.

ARONOFF, S., 1989, *Geographic Information Systems: A management perspective*. 1 ed. Ottawa, WDL Publications.

AXIOMAP, 2000, "AXIOMap: Application of XML for Interactive Online Mapping". In: http://www.elzaresearch.com/, Acessado em 24/02/2000.

BARRERA, R.,BUCHMANN, A., 1981, "Schema Definition and Query Language for a Geographical Database System", *IEEE Computer Architecture for Pattern Analysis and Image Database Management*, v. November, pp. 250-256.Hot Springs.

BARU, C., GUPTA, A., LUDASCHER, B., et al, 1999, "XML-based information mediation with MIX". In: *Procedings of ACM SIGMOD Conf. on Management of Data*, pp. 597-599, Philadelphia, PA, USA, June.

BATINI, C., LENZERINI, M., NAVATHE, S. B., 1986, "A comparative analysis of methodologies for database schema integration", *ACM Computing Survey*, v. 18 (Dec.), pp. 322-364

BONJOUR, M.,FALQUET, G., 1994, "Concept bases: a support to information system integration". In: *Lecture Notes in Computer Science: Advances Information Systems*, pp. 242-245, Utrecht: Holanda.

BOTÊLHO, L. R.,SOUZA, J. M., 2003, "Pequeno Glossário de Termos de Linguagens de Marcação", *Developers' Magazine*, v. Ano VII, número 77, pp. 22-25.Axcel Books do Brasil Editora.

BOTÊLHO, L. R., STRAUCH, J. C. M., SOUZA, J. M., 2003a, *An XML-based architecture for geo-referenced data integration*.In: ES-614/03. COPPE/UFRJ - Programa de Engenharia de Sistemas e Computação. Rio de Janeiro, RJ.

BOTÊLHO, L. R., STRAUCH, J. C. M., SOUZA, J. M., 2003b, "WISE: An architecture for geo-referenced data integration ". In: *NG2I 2003 - Next Generation Geospatial Information - Extended Abstract*, Cambridge, MA, USA, October.

BREITBART, Y., 1990, "Multidatabase Interoperability", *SIGMOD Record*, v. 19 (Sept.), pp. 53-60

BREITBART, Y., OLSON, P. L., THOMPSON, G. R., 1986, "Database integration in a distributed database system". In: *Proc.of International Conference on DataEngineering*, pp. 301-310, California, USA.

BUEHLER, K.,MCKEE, L., 1996, *The OpenGIS guide: Introduction to interoperable Geoprocessing*.In: OGIS TC 96-001. http://www.ogis.org/guide/guide.html.

BULL, G., 1994, "Ecosystem Modelling with GIS", *Environmental Management*, v. 18, n. 3, pp. 345-349.

BUSSE, S., KUTSCHE, R., LESER, U., et al, 1999, *Federated Information Systems: Concepts, Terminology and Architectures*.In: 99-9. http://cis.cs.tu-berlin.de/~sbusse/publications/, TU Berlin.

CAREY, M. J.,ET AL., 1997, *Towards Heterogeneous Multimedia Information Systems: The Garlic Approach*.In: RJ9911.

CASTELLANOS, M., KUDRASS, T., SALTOR, F., et al, 1994, "Interdatabase existence dependencies: a metaclass approach", pp. 213-216, Austin.

CHATTERJEE, A.,SEGEV, A., 1991, "Data Manipulation in Heterogeneous Databases", *SIGMOD Record*, v. 20 (Dec.), pp. 64-68

CHAWATHE, S., GARCIA-MOLINA, H., HAMMER, J., et al, 1994, "The TSIMMIS Project: Integration of Heterogeneous Information Sources". In: *Proceedings of the 10th Anniversary Meeting*, pp. 7-18, Tokyo, Japan.

CHRISTOPHIDES, V., CLUET, S., SIMÉON, J., 2000, "On Wrapping Query Languages, Efficient XML Integration". In: *Proc. of ACM SIGMOD Conf. on Management of Data*, pp. 141-152, Dallas, Texas, USA, May.

CLEMENT, G., LAROUCHE, C., GOUIN, D., et al, 1997, "OGDI: Toward Interoperability among Geospatial Databases", *SIGMOD Record*, v. 26 (Sept.), pp. 18-23.

DEEGREE, 2004, "Deegree". In: deegree.sourceforge.net, Acessado em 10/07/2004.

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

ELMAGARMID, A., RUSINKIEWICZ, M., SHETH, A., 1999, *Management of Heterogeneuos and Autonomous Database Systems*. San Francisco, CA, Morgan Kaufmann Publishers.

ESRI, 2003, "ArcIMS". In: www.esri.com, Acessado em 19/06/2003.

FENG, W., JICHANG, S., HUOWANG, C., et al, 2000, "GeoSQL: A Spatial Query Language of Object-Oriented GIS". In: *Procedings of the 2nd International Workshop on Computer Science and Information Technologies*, Ufa, Russia.

FIREBIRD, 2004, "Firebird". In: firebird.sourceforge.net, Acessado em 28/07/2004.

FONSECA, F., EGENHOFER, M., BORGES, K. A. V., 2000a, "Ontologias e Interoperabilidade Semântica entre SIGs". In: *Anais do II Workshop Brasileiro de GeoInformática - GeoInfo 2000*, pp. 45-52, São Paulo - São Paulo, Brasil, June.

FONSECA, F. T., EGENHOFER, M. J., DAVIS, C. A., et al, 2000b, "Ontologies and Knowledge Sharing in Urban GIS", *CEUS - Computer, Environment and Urban Systems*, v. 24, n. 3, pp. 232-251.

FRANKHAUSER, P., MOTZ, R., HUCK, G., 1995, *SIM Schema integration methodology*.In: IRO/SPEC/D4-4.1-V1.0/PF950210.

FRIESEN, P.,SONDHEIM, M., 1994, "Mixing data from diverse sources". In: *GIS'94 Symposium*, pp. 151-156, Vancouver, February.

GARCIA-MOLINA, H., PAPAKONSTANTINOU, Y., QUASS, D., et al, 1997, "The TSIMMIS Approach to Mediation: Data Models and Languages", *Journal of Intelligent Information Systems*, v. 8, n. 2, pp. 117-132.

GARDARIN, G., SHA, F., NGOC, T., 1999, "XML-based Components for Federating Multiple Heterogeneous Data Sources.". In: *Proc. of International Conference on Conceptual Modeling*, pp. 506-519, Paris, France.

GARDELS, K., 1996, "Comprehensive data model for distributed, heterogeneous geographic information". In: http://www.regis.berkley.edu/gardels/geomodel_def.html.

GARDELS, K., 1997, "Open GIS and on-line environmental libraries", *SIGMODRecord*, v. 26, n. 1 (Mar.), pp. 32-38

GEOTOOLS, 2004, "GeoTools". In: www.geotools.org, Acessado em 29/07/2004.

GML, 2003, "Geography Markup Language". In: www.opengis.net/gml, Acessado em 14/10/2003.

GOLDMAN, R., MCHUGH, J., WIDOM, J., 1999, "From Semistructured Data to XML:Migrating the Lore Data Model and Query Language". In: *2nd International Workshop on the Web and Databases (WebDB '99)*, pp. 25-30, Philadelphia, Pennsylvania, USA.

GUPTA, A., MARCIANO, R., ZASLAVSKY, I., et al, 1999, "Integrating GIS and Imagery through XML-based Information Mediation", *Lecture Notes in Computer Science*, v. 1737.

HURSON, A. R., BRIGHT, M. W., PAKZAD, S. H., 1994, *Multidatabase Systems: An Advanced Solution for Global Information Sharing*. Los Alamitos, CA, IEEE Computer Society Press.

INTERBASE, 2004, "Interbase". In: www.borland.com/interbase, Acessado em 29/07/2004.

JAVA, 2004, "Java". In: www.java.sun.com, Acessado em 06/07/2004.

JAXP, 2004, "JAXP (Java API for XML Parsing)". In: java.sun.com/xml/jaxp/index.jsp, Acessado em 29/07/2004.

JEXCELAPI, 2004, "JExcelAPI". In: www.andykhan.com/jexcelapi/index.html, Acessado em 10/07/2004.

KALINICHENKO, L. A., 2001, "Subject Mediation for Integrated Access to Heterogeneous Information Sources". In: *ADBIS 2001*.

KASHYAP, V.,SHETH, A., 1996, "Semantic and Schematic similarities between database objects: a context-based approach". In: Apers, P., Scheck, H., Gray, J., Su, S., *VLDB Journal*.

LAURINI, R.,THOMPSOM, D., 1992, *Fundamentals of Spatial Information Systems*. 1 ed. Londres, Academic Press.

LESELECT, 2002, "Le Select- A Mediator System Developed in the Caravel Project". In: http://caravel.inria.fr/Fprototype_LeSelect. html, Acessado em 15/03/2000.

LUDASCHER, B., PAPAKONSTANTINOU, Y., VELIKHOV, P., 1998, "A Brief Introduction to XMAS", *Information Systems*, v. 23, n. 8.

MACFARLANE, A., MCCANN, LIDDELL, H., 1996, "A common data model for meta-data interoperable environments". In: *Proceedings of II International Baltic Workshop onDatabase and Information Systems*.

MEDEIROS, S. P. J., STRAUCH, J. C. M., SOUZA, J. M., et al, 2001, "SPeCS - a Spatial Decision Support Collaborative System for Environment Design". In: *Procedings of SAC2001*, Las Vegas, USA.

MICROSOFT EXCEL, 2004, "MicroSoft Excel". In: www.microsoft.com/office/excel/default.asp, Acessado em 10/07/2004.

OGC, 1999, "Open GIS Simple Features Specification for SQL. Revision 1.1". In: http://www.opengis.org/techno/specs/99-049.pdf, Acessado em 10/08/2001.

OGC, 2003, "Open GIS Consortium". In: www.opengis.org, Acessado em 28/07/2003.

OLIVEIRA, R. L.,MAGALHÃES, G. C., 1993, *Metodologias para conversão de esquemas em sistemas de Banco de Dados heterogêneos*.In: Relatório Técnico DCC-17/93. UNICAMP, Campinas.

OPENMAP, 2004, "OpenMap". In: openmap.bbn.com, Acessado em 29/07/2004.

OZSU, M.,VALDURIEZ, P., 1999, *Principles of Distributed Database Systems*. 2ª ed ed.

ÖZSU, M. T.,VALDURIEZ, P., 1999, *Principles of Distributed Database Systems*. 2 ed. Upper Saddle River, New Jersey, Prentice-Hall.

PINTO, G. R. B., 2003, *Publicação de Dados Ambientais*. Tese de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

PINTO, G. R. B., MEDEIROS, S. P. J., SOUZA, J. M., et al, 2001, "X-Arc Spatial Data Integration in the SPeCS Collaborative Design Framework". In: *Proceedings of the Sixth International Conference on CSCW in Design*, pp. 56-60, London, Ontario, Canada.

PROTÉGÉ, 2004, "Protégé". In: protege.stanford.edu, Acessado em 29/07/2004.

RIBEIRO, C. H. F. P.,OLIVEIRA, J. P. M., 1996, "Multidatabase interoperability through conceptual schema mapping in an object oriented model", *Proceedings.of the International Conference Parallel and Distributed Computing Systems*, v. 2 (Sept.), pp. 647-652

ROBINSON, A., SALE, R., MORRISON, J., 1978, *Elements of Cartography*, John Wiley & Sons.

ROTH, M. T.,SCHWARZ, P., 1997, "A Wrapper Architecture for Legacy Data Sources". In: *Proceedings of VLDB 97*.

SHAPE FILE, 2003, "ESRI Shape". In: www.esri.com/library/whitepapers/pdfs/shapefile.pdf, Acessado em 19/06/2003.

SHETH, A., GALA, S. K., NAVATHE, S. B., 1993, "On automatic reasoning for schema integration", *International Journal Of Intelligent and Cooperative Information Systems*, v. 2, n. 1, pp. 23-50.

SHETH, A.,LARSON, J. A., 1990, "Federated Database Systems for Managing Distributed, Heterogeneuos and Autonomous Databases", *ACM Computing Surveys*, v. 22, n. 3, pp. 183-236.

SOROKINE, A.,MERZLIAKOVA, I., 1998, "Interactive map applet for illustrative purpose", *AGM GIS'98*, pp. 46-51.

SOUZA, J. M., 1986, *Software tools for Conceptual Schema integration*. Tese de Ph. D., School of Information Systems, University of East Anglia, England.

SOUZA, J. M., FERRARI, R., DUARTE, M., et al, 1993, "Uma Arquitetura Organizacional para Sistemas de Informação Geográfica Orientados a Objetos".

In: *Anais da IV Conferência Latino Americana sobre Sistemas deInformação Geográfica*, São Paulo.

STRAUCH, J. C. M., 1998, *Integração de Bases de Dados Geográficas Heterogêneas e Distribuídas*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

STRAUCH, J. C. M., SOUZA, J. M., MATTOSO, M. L. Q., 1998, *Estudo do metamodelo Spatial Archive and Interchange Format (SAIF)*.In: Relatório Técnico da COPPE/UFRJ.

SVG, 2003, "Scalable Vector Graphics (SVG)". In: www.w3.org/Graphics/SVG, Acessado em 19/06/2003.

TANAKA, A., VALDURIEZ, P., SIMON, E., et al, 2001, "The Ecobase Project: Database and Web Technologies for Environmental Information Systems", *ACM SIGMOD Record*, v. 30, n. 3, pp. 70-75.

VENTRONE, V.,HEILER, S., 1991, "Semantic heterogeneity as a result of domain evolution", *SIGMOD Record*, v. 20, n. 4 (Dec.), pp. 16-20

VIDAL, V. M. P., BRAYNER, A., OLIVEIRA, A., 2001a, "Self-Maintenance of Materialized Views in XMLBased Mediator". In: *Informal Proc. of 1th Int'l Workshop on Data Integration Over the Web (DIWeb)*, pp. 32-46, Interlaken, Switzerland, Junea.

VIDAL, V. M. P., LÓSCIO, B. F., SALGADO, A. C., 2001b, "Using Correspondence Assertions for Specifying the Semantics of XML-Based Mediators". In: *Proceedings of the International Workshop on Information Integration on the Web*, pp. 3-11, Rio de Janeiro, RJ, Brasil.

VML, 1998, "VML (Vector Markup Language)". In: www.w3c.org/TR/1998/NOTE-VML-19980513, Acessado em 13/05/1998.

W3C, 2003, "World Wide Web Consortium". In: http://www.w3.org, Acessado em 14/05/2003.

WIEDERHOLD, G., 1992, "Mediators in the Architecture of Future Information Systems", *IEEE Computer*, v. 25, n. 3, pp. 38-49.

WIEDERHOLD, G.,GENESERTH, M., 1997, "The Conceptual Basis for Mediation Services", *IEEE Expert*, v. 12, n. 5, pp. 38-47.

XALAN, 2004, "Xalan". In: xml.apache.org/xalan-j/index.html, Acessado em 29/07/2004.

XERCES, 2004, "Xerces". In: xml.apache.org/xerces2-j/index.html, Acessado em 29/07/2004.

XHUMARI, F., AMZAL, M., SIMON, E., 1999, "Le Select: a Middleware System for Publishing Autonomous and Heterogeneous Information Sources". In: http://caravel.inria.fr/Fprototype_LeSelect.html, Acessado em 15/06/2000.

XML, 2003, "Extensible Markup Language (XML)". In: www.w3.org/xml, Acessado em 19/06/2003.

XML, 2004, "Extensible Markup Language (XML)". In: www.w3.org/xml, Acessado em 19/07/2004.

XML SCHEMA, 2003, "XML Schema". In: www.w3c.org/XML/Schema, Acessado em 19/06/2003.

ZASLAVSKY, I., MARCIANO, R., GUPTA, A., et al, 2000, "XML-based Spatial Data Mediation Infrastructure for Global Interoperability", Cape Town, South Africa, March.

ZHANG, J., JAVED, M. S., SHAHEEN, A., et al, 2000, "A Prototype for Wrapping and Visualizing Geo-Referenced Data in Distributed Environments Using the XML Technology". In: *ACMGIS 2000*, pp. 27-32, VA, USA.

Apêndice A – Glossário

É apresentado neste apêndice um pequeno glossário de alguns termos utilizados ao longo do texto desta dissertação.

A.1 Termos de XML e Afins

CSS (*Cascading Stylesheets*): fornece um mecanismo simples de controlar o estilo de páginas HTML sem comprometer a sua estrutura, permitindo que sejam controladas suas características visuais. Como exemplo, podemos citar: fontes, margens, bordas, cores, entre outros.

DOM (*Document Object Model*): possibilita que *scripts* manipulem HTML usando um conjunto de métodos e tipos de dados, definidos independentemente de linguagens de programação ou de plataformas de computadores. Forma a base para efeitos dinâmicos em páginas *Web*.

DTD (*Document Type Definition*): é um arquivo em anexo ao documento XML que define o conjunto de regras que este documento deve seguir, as quais devem ser processadas antes do documento XML ter sido lido. Essas regras geralmente mostram o nome e conteúdo de cada elemento e atributo do documento. Sem o DTD, os programas que acessam o documento podem não saber como processar *links*, imagens ou entidades.

GML (*Geography Markup Language*): é uma codificação XML para o transporte e armazenamento de informações geográficas, incluindo propriedades espaciais e não-espaciais das características geográficas.

HTML (*HyperText Markup Language*): é a linguagem para a publicação de hipertextos na *World Wide Web*. É um formato não-proprietário baseado em SGML (subconjunto de SGML), e pode ser criado e processado por uma grande variedade de

ferramentas. HTML usa *tags* como "<h1>" e "</h1>" para estruturar o texto em cabeçalhos, parágrafos, listas, *links* de hipertexto, etc.

Quando HTML foi criada visava-se construir uma linguagem simples a partir da SGML, que é muito complexa. Entretanto, o esforço de simplificar a SGML para criar a HTML foi exagerado, criando somente marcações fixas e pré-definidas. HTML visava apenas mostrar dados formatados, não se importando com a sua estrutura ou semântica, limitando bastante o potencial da *Web*. Desta forma a HTML é boa para apresentação de dados. Essas e ou tras deficiências estão sendo sanadas com o uso da XML (*Extensible Markup Language*), que oferece os benefícios da SGML sem sua complexidade.

Namespaces XML: um padrão que permite especificar identificadores para o conjunto de nomes de elementos definidos por um DTD. Um documento usando um DTD pode ser incluído em qualquer outro documento sem que exista conflito com outros nomes de elementos. Os elementos definidos em um DTD são identificados unicamente, de modo que, por exemplo, um interpretador possa dizer quando um elemento chamado "<disciplina>" deve ser interpretado de acordo com este DTD, ao invés de usar a definição de um elemento "<disciplina>" de um outro DTD.

RDF (*Resource Description Framework*): define um modelo simples para descrever relacionamentos entre recursos em termos de propriedades (nomeadas) e valores. As propriedades RDF podem ser pensadas como atributos de recursos e neste senso correspondem a pares atributo-valor tradicionais. As propriedades RDF também representam relacionamentos entre recursos. Como tal, o modelo de dados RDF pode portanto parecer com um diagrama de entidades e relacionamentos.

O modelo de dados RDF por si só, contudo, não fornece mecanismos para descrever estas propriedades, nem provê qualquer mecanismo para descrever os relacionamentos entre estas propriedades e outros recursos, este é o objetivo do *RDF Schema*.

RDF *Schema*: linguagem de descrição de vocabulário RDF. Define classes e propriedades que podem ser usadas para descrever outras classes e propriedades.

O RDF *Schema* é especificado em termos do modelo de informação RDF básico – uma estrutura de grafos descrevendo recursos e propriedades. Todos vocabulários

RDF compartilham alguma estrutura comum básica: eles descrevem classes de recursos e tipos de relacionamentos entre recursos. Esta conformidade permite uma mistura de vocabulários processáveis por máquina, e endereça a necessidade de criar metadados cujas declarações possam criar múltiplos vocabulários que são gerenciáveis de uma maneira descentralizada por comunidades independentes.

SGML (*Standard Generalized Markup Language*): é um padrão internacional para definir descrições de estrutura e conteúdo de diversos tipos de documentos. Permite criar um documento a partir de diversas fontes, definir a estrutura do documento usando uma gramática especial e adicionar marcas a um documento para mostrar sua estrutura.

SMIL (*Synchronized Multimedia Integration Language*): possibilita a autoria de apresentações audiovisuais interativas. SMIL é usada para "mídia rica"/apresentações multimídia, as quais integram fluxo (streaming) de áudio e vídeo com imagens, texto ou qualquer outro tipo de mídia. SMIL é uma linguagem fácil de aprender como HTML, e muitas de suas apresentações são escritas usando um editor de texto comum.

SVG (*Scalable Vector Graphics*): é uma linguagem para descrição de gráficos bidimensionais em XML. SVG admite três tipos de objetos gráficos: formas gráficas vetoriais (e.g., caminhos (*paths*) consistindo de linhas retas e, curvas), imagens e texto. Os objetos gráficos podem ser agrupados, estilizados, transformados, e compostos em objetos previamente reenderizados. O texto pode estar em qualquer *namespace* XML adequado a aplicação, o qual reforça a capacidade de busca e acessibilidade dos gráficos SVG. O conjunto de características inclui transformações aninhadas, filtro de efeitos, objetos modelo e extensibilidade, entre outros.

Os desenhos SVG podem ser dinâmicos e interativos. O DOM para SVG, o qual inclui o DOM XML completo, possibilita animação de gráficos vetoriais através de *scripts*.

URI (*Uniform Resource Identifiers*): Conjunto genérico de todos nomes/endereços, que são pequenas *strings*, que identificam recursos na *Web*: documentos, imagens, serviços, e outros recursos.

URL (*Uniform Resource Locator*): Um termo informal (não mais usado em relatórios técnicos) associado com esquemas URI populares: http, ftp, mailto, etc.

XHTML (*Extensible HyperText Markup Language*): é uma reformulação de HTML em XML. Combina a intensidade da HTML com o poder da XML.

XLink (*XML Linking Language*): permite que elementos sejam inseridos em documentos XML para criar e descrever *links* entre recursos. Ela usa sintaxe XML para criar estruturas que podem ser descritas com *hiperlinks* unidirecionais simples da HTML atual, bem como *links* mais sofisticados.

Ele permite que documentos XML:

- Declarem relacionamentos de *links* entre mais de dois recursos;
- Associe metadados com um *link*;
- Expresse *links* que residam em um local separado dos recursos relacionados.

XML (*Extensible Markup Language*): é uma metalinguagem, ou seja, uma linguagem para descrever outras linguagens, onde podemos criar nossa própria linguagem de marcação. Cabe ressaltar que, assim como HTML, XML é um subconjunto da SGML.

Seu objetivo é disponibilizar SGML para ser servida, recebida e processada na *Web*, do mesmo modo como é possível com HTML. Ela foi projetada para interagir tanto com SGML como com HTML.

Ao contrário da HTML que tem um conjunto de marcações fixas e pré-definidas que são entendidas por todos os navegadores, mas que são limitados à exibição de dados, a XML é o resultado de uma evolução no sentido de definir um formato padrão para documentos que seja portátil e estruturado e que forneça informações tanto sobre conteúdo quanto contexto.

A dificuldade em estabelecer um formato alternativo, que tanto "leitores" quanto "publicadores" de dados entendam o mesmo conjunto de marcações sem limitar sua

riqueza e flexibilidade, vem sendo resolvido com XML, pelo fato de ser auto-descritiva. A descrição do documento (atualmente) é dada por um DTD (*Document Type Definition*) que é anexado a cada documento XML. Essas marcações são extensíveis, possibilitando um mecanismo para que sejam criadas ontologias, ou seja, um sistema comum para codificação de conceitos que são significativos em um dado contexto.

XML *Schema*: enquanto o DTD é usado para a declaração de restrições no uso da marcação, o processamento automatizado de documentos XML requer facilidades mais rigorosas e compreensíveis nesta área. As necessidades são por restrições em como as partes componentes de uma aplicação se ajustam juntas, a estrutura do documento, atributos, tipos de dados, e assim por diante. O *XML Schema* é responsável pelo modo da definição da estrutura, conteúdo e semântica de documentos XML.

XPath (*XML Path Language*): é o resultado de um esforço para prover uma sintaxe e semântica comuns para compartilhar funcionalidades entre XSLT (*XSL Transformations*) e Xpointer (*XML Pointer Language*). A sua proposta inicial é a de endereçar partes de um documento XML. No suporte desta proposta inicial, ela também fornece facilidades básicas para manipulação de *strings*, números e valores boleanos. XPath usa uma sintaxe compacta e não-xml para facilitar o seu uso dentro de URIs e em valores de atributos XML. XPath opera na estrutura abstrata e lógica de um documento XML, ao invés da sintaxe superficial deste. O nome XPath veio do uso de uma notação de caminho como em URLs para navegação através da estrutura hierárquica de um documento XML.

XPointer (*XML Pointer Language*): faz parte do padrão Xlink, especifica como declarar endereços dentro de expressões XLink.

XPointer, a qual é baseada na XPath (*XML Path Language*), suporta endereçamento nas estruturas internas dos documentos XML e em entidades externas. Ela permite o exame de uma estrutura de documentos hierárquicos e a escolha de suas partes internas baseada em várias propriedades, tais como: tipos de elementos, valores de atributos, característica de conteúdo, e posição relativa.

XQuery (XML *Query Language*): uma linguagem de consulta para XML. É projetada para ser uma linguagem pequena e de fácil implementação, cujas consultas são concisas e facilmente inteligíveis. Ela também é flexível o suficiente para consultar uma grande quantidade de fontes de informações XML, incluindo documentos estruturados e semi-estruturados, bancos de dados relacionais, e repositórios de objetos.

XSL (*Extensible Stylesheet Language*): é uma linguagem para expressar *stylesheets*. Ela consiste de três partes: XSLT (*XSL Transformations*), que é uma linguagem para transformação de documentos XML, XPath (*XML Path Language*), que é uma linguagem de expressão usada pela XSLT para acessar ou referir a partes de um documento XML. A terceira parte *XSL Formatting Objects*, que é um vocabulário XML para especificação de formatação semântica. Um *stylesheet* XSL especifica a apresentação de uma classe de documentos XML pela descrição de como uma instância da classe é transformada em um documento XML que usa o vocabulário de formatação.

XSLT (*XSL Transformations*): é uma linguagem para a transformação de documentos XML em outros documentos XML.

XSLT é projetada para uso como parte da XSL. Em acréscimo à XSLT, XSL inclui um vocabulário XML para especificação de formatação. XSL especifica o estilo de um documento XML usando XSLT para descrever como o documento é formatado em outro documento que usa o vocabulário de formatação.

XSLT é projetada para ser usada independentemente da XSL. Contudo, não é a promessa da XSLT ser completamente uma linguagem de transformação XML de proposta geral. Ao invés, ela é projetada primariamente para os tipos de transformações que são necessárias quando é usada como parte da XSL.

Apêndice B – XML

Uma pequena revisão de XML é apresentada neste apêndice com a finalidade de esclarecer sobre alguns dos termos e conceitos utilizados ao longo dos capítulos 4 e 5.

B.1 Principais componentes de XML

Alguns dos principais componentes da XML são descritos nesta seção.

B.1.1 Elemento

O elemento é descrito por uma marca inicial (i.é, <nome_elemento>) e uma marca final (i.é, </nome_elemento>). As marcas inicial e final são também chamadas de marcação. O conteúdo de um elemento é delimitado pela marcação. Cada elemento pode conter texto, conter outros elementos aninhados (subelementos), ter conteúdo misto ou ser vazio.

Exemplo:

```
<pessoa>
      <nome>Thayenne Rogério Botêlho</nome>
      <idade>20</idade>
</pessoa>
```

As marcas <pessoa> e </pessoa> descrevem a estrutura do elemento pessoa. O elemento pessoa possui subelementos, como: nome e idade, todos contendo suas marcações. Como podemos observar, as marcas de um documento são balanceadas, ou seja, são fechadas na ordem inversa da que foram abertas.

B.1.2 Atributo

XML permite associar atributos aos seus elementos. XML usa o termo "atributo" para especificar propriedades dos elementos. O valor de um atributo é sempre um texto e deve aparecer entre aspas.

Exemplo:

```
<pessoa cpf="12345678911">
      <nome>Thayenne Rogério Botêlho</nome>
      <idade>20</idade>
</pessoa>
```

No exemplo acima o cpf é um atributo do elemento pessoa.

Um documento XML pode ser considerado bem-formado e válido. Para ser bem-formado, o documento deve obedecer algumas regras: a) conter pelo menos um elemento; b) conter uma única marca inicial e marca final descrevendo um elemento que contenha todo o documento (chamado elemento raiz); e c) todas as outras marcas devem estar aninhadas apropriadamente. Para ser válido, além do documento ser bem-formado, este deve obedecer a uma estrutura especificada por um esquema XML.

A Listagem B.1 mostra um exemplo de documento XML sobre uma biblioteca e suas publicações, artigos, livros, e seus respectivos autores e instituições.

```
<exemplo:Biblioteca xmlns:exemplo="http://www.cos.ufrj.br/bd/Exemplo"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.cos.ufrj.br/bd/Exemplo
D:\ProgsMestrado\XML_GML\EsqBiblioteca.XSD">
   <Livro>
        <ano>2004</ano>
        <isbn>01234567890</isbn>
        <título>A arte do simples saber</título>
        <autor_ref>1</autor_ref>
        <autor_ref>2</autor_ref>
        <editora>Editora Nova Ilusão</editora>
   </Livro>
   <Autor>
        <id_autor>1</id_autor>
        <nome>Alcione Maria Rogério Botêlho</nome>
        <e_mail>alcione@starmedia.com</e_mail>
        <instituicao_ref>13</instituicao_ref>
        <instituicao_ref>5</instituicao_ref>
   </Autor>
   <Autor>
        <id_autor>2</id_autor>
        <nome>Eugênia Antunes Vallory</nome>
        <e_mail>vo.geny@starmedia.com</e_mail>
        <instituicao_ref>5</instituicao_ref>
   </Autor>
   <Instituicao>
        <id_instituicao>5</id_instituicao>
        <nome>Instituição Visão Singular</nome>
        <telefone>32-1221-3112</telefone>
        <endereco>
             <cidade>Patrocínio do Muriaé</cidade>
             <estado>MG</estado>
             <pais>Brasil</pais>
        </endereco>
   </Instituicao>
   <Instituicao>
        <id_instituicao>13</id_instituicao>
        <nome>Escola Superior de Artes</nome>
        <telefone>32-1222-2223</telefone>
        <endereco>
             <cidade>Patrocínio do Muriaé</cidade>
             <estado>MG</estado>
             <pais>Brasil</pais>
        </endereco>
   </Instituicao>
</exemplo:Biblioteca>
```

Listagem B.1 – Exemplo de documento XML de uma Biblioteca

B.2 Esquemas para Documentos XML

Um esquema descreve a estrutura lógica de uma fonte de informação, incluindo os tipos dos dados, os relacionamentos entre os dados e as restrições envolvendo os dados. Para XML, um esquema é usado para descrever os elementos, o conteúdo dos elementos, a lista de atributos de um determinado elemento e as restrições sobre os elementos e os atributos.

A primeira proposta para definição de esquema XML foi o DTD (*Document Type Definition*)(W3C, 2003). Entretanto, o DTD possui algumas limitações. Para sobrepor as limitações do DTD, foi padronizado outro tipo de esquema, chamado XML Schema (XML SCHEMA, 2003), possuindo mais semântica e oferecendo recursos adicionais para definição de esquemas.

B.3 XML Schema

O XML Schema introduz novos construtores que fazem com que esta linguagem seja mais expressiva do que o DTD. Com isso, o XML Schema pode ser utilizado em uma grande variedade de aplicações. Algumas das principais características do XML Schema são:

- Um XML Schema é escrito em XML. Desta forma, o usuário não precisa aprender uma nova sintaxe proprietária. Além disso, o esquema pode ser armazenado em um sistema de armazenamento XML, junto com os documentos XML;
- Permite que o usuário defina novos tipos de dados;
- Permite que o usuário defina o relacionamento de tipo e subtipo;
- Possui formas de herança;
- Possui restrições de integridade.

A seguir, são descritos alguns componentes de XML Schema. Para exemplificar o uso destes componentes, é utilizado o esquema XML Schema, na Listagem B.2, que representa a estrutura do documento Biblioteca, apresentado na Listagem B.1.

B.3.1 Declaração de Elemento

Elementos são declarados utilizando o construtor element. Os principais atributos deste construtor são: name, que associa o elemento declarado a um nome; type, que atribui um tipo ao elemento; e minOccurs e maxOccurs, que especificam a restrição mínima e máxima de ocorrência do elemento, respectivamente: se a restrição mínima for igual a zero, o elemento é opcional, se for igual a um é obrigatório; e se a restrição máxima for igual a um, o elemento é mono-ocorrência, e se for unbounded é multi-ocorrência. Sendo que, por padrão, as ocorrências mínima e máxima são iguais a um, e neste caso as ocorrências não precisam aparecer declaradas explicitamente.

Considere, por exemplo, o elemento Autor, mostrado na Listagem B.2. A declaração <element name="Autor" type="AutorType" minOccurs="0" maxOccurs="unbounded"/> especifica que este elemento é do tipo AutorType e a restrição de ocorrência é opcional e multi-ocorrência.

B.3.2 Declaração de Atributo

Atributos são declarados utilizando o construtor attribute. As principais propriedades deste construtor são: name, que associa o atributo declarado a um nome; type, que atribui um tipo ao atributo.

```
<schema targetNamespace="http://www.cos.ufrj.br/bd/Exemplo" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:exemplo="http://www.cos.ufrj.br/bd/Exemplo" elementFormDefault="unqualified"
attributeFormDefault="unqualified">
   <element name="Biblioteca">
      <complexType>
         <sequence>
            <element name="Artigo" type="exemplo:ArtigoType" minOccurs="0" maxOccurs="unbounded"/>
            <element name="Livro" type="exemplo:LivroType" minOccurs="0" maxOccurs="unbounded"/>
            <element name="Autor" type="exemplo:AutorType" minOccurs="0" maxOccurs="unbounded"/>
            <element name="Instituicao" type="exemplo:InstituicaoType" minOccurs="0" maxOccurs="unbounded"/>
         </sequence>
      </complexType>
      <key name="Artigo.key">
         <selector xpath="Artigo"/>
         <field xpath="isbn"/>
      </key>
      <key name="Livro.key">
         <selector xpath="Livro"/>
         <field xpath="isbn"/>
      </key>
      <key name="Autor.key">
         <selector xpath="Autor"/>
         <field xpath="id_autor"/>
      </key>
      <key name="Instituicao.key">
         <selector xpath="Instituicao"/>
         <field xpath="id_instituicao"/>
      </key>
      <keyref name="Artigo.keyref" refer="Autor.key">
         <selector xpath="Artigo"/>
```

```
            <field xpath="autor_ref"/>
        </keyref>
        <keyref name="Livro.keyref" refer="Autor.key">
            <selector xpath="Livro"/>
            <field xpath="autor_ref"/>
        </keyref>
        <keyref name="Autor.keyref" refer="Instituicao.key">
            <selector xpath="Autor"/>
            <field xpath="instituicao_ref"/>
        </keyref>
        <unique name="Autor.unique.e_mail">
            <selector xpath="Autor"/>
            <field xpath="e_mail"/>
        </unique>
    </element>
    <complexType name="PublicacaoType">
        <sequence>
            <element name="ano" type="string"/>
            <element name="isbn" type="integer"/>
            <element name="titulo" type="string"/>
            <element name="autor_ref" type="integer" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <complexType name="ArtigoType">
        <complexContent>
            <extension base="exemplo:PublicacaoType">
                <sequence>
                    <element name="local_publicacao" type="string"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="LivroType">
        <complexContent>
            <extension base="exemplo:PublicacaoType">
                <sequence>
                    <element name="editora" type="string"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="AutorType">
        <sequence>
            <element name="id_autor" type="integer"/>
            <element name="nome" type="string"/>
            <element name="e_mail" type="string"/>
            <element name="instituicao_ref" type="integer" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <complexType name="InstituicaoType">
        <sequence>
            <element name="id_instituicao" type="integer"/>
            <element name="nome" type="string"/>
            <element name="telefone" minOccurs="0"/>
            <element name="endereco" type="exemplo:EnderecoType"/>
        </sequence>
    </complexType>
    <complexType name="EnderecoType">
        <sequence>
            <element name="cidade" type="string"/>
            <element name="estado" type="string"/>
            <element name="pais" type="string"/>
        </sequence>
    </complexType>
</schema>
```

Listagem B.2 – XML Schema de uma Biblioteca

B.3.3 Tipos Simples

São tipos de dados atômicos ou primitivos definidos em XML Schema. O esquema Biblioteca atribui tipos simples a vários atributos e elementos, tais como, *integer* e *string*. O valor de um tipo atômico é indivisível de acordo com a perspectiva da XML Schema.

B.3.4 Definição de Tipos Complexos

Tipos complexos podem ser definidos utilizando o construtor complexType. Tais definições geralmente contêm um nome, declarações de elementos e declarações de atributos.

Além das restrições de ocorrência declaradas para elementos individuais, o XML Schema provê restrições a serem aplicadas em grupos de elementos. Estas restrições também podem ser chamadas de delimitadores de grupos. Existem três tipos de delimitador de grupo:

- sequence - indica que os elementos do grupo devem aparecer no documento na mesma ordem em que foram declarados no esquema;
- all - estabelece que todos os elementos do grupo podem aparecer uma ou nenhuma vez, e que eles podem aparecer em qualquer ordem; e
- choice - indica que somente um dos elementos declarados no grupo pode aparecer no documento.

Considerando o tipo InstituicaoType, a sua definição especifica que este tipo complexo contém os elementos id_instituicao, nome, telefone e endereço, que devem aparecer nesta ordem no documento.

Esquemas podem ser construídos a partir de definições de tipos complexos nomeados, tais como AutorType e InstituicaoType, e declarações de elementos, tais como Autor e Instituicao, que referenciam os tipos através do atributo type. Porém, um tipo complexo também pode ser definido de forma anônima, inserido na declaração de um elemento. No esquema Biblioteca, a declaração do elemento Biblioteca contém uma definição de tipo anônima, especificando que este elemento contém os elementos Artigo, Livro, Autor e Instituicao.

B.3.5 Derivação de Tipos por Extensão

Um tipo complexo também pode ser definido a partir da extensão de um outro tipo complexo ou de um tipo simples.

No esquema Biblioteca, a definição do tipo ArtigoType especifica que este tipo complexo é definido a partir do tipo complexo PublicacaoType (atributo base do elemento extension). Assim, um elemento do tipo complexo ArtigoType contém os elementos do tipo complexo PublicacaoType, além do elemento editora. Neste caso, podemos dizer que ArtigoType é um subtipo (especialização) de PublicacaoType.

B.3.6 Restrições de Integridade

XML Schema oferece mecanismos de restrição de integridade, tais como: key e keyref e unique. Estas restrições de integridade possuem as características: a) podem ser aplicadas tanto a elementos como a atributos; b) podem ser aplicadas a mais de um elemento ou atributo; c) permitem limitar o escopo, no qual o valor do elemento ou do atributo deve ser único, ou deve referenciar; e d) a restrição keyref permite capturar o tipo dos elementos ou dos atributos que estão sendo referenciados através do escopo da chave referenciada.

Para exemplificar a especificação de restrições de integridade no XML Schema pode-se referir ao XML Schema Biblioteca na Listagem B.2.

B.3.7 Restrição de Integridade de Chave

XML Schema permite especificar, através do construtor key, que o valor de um conjunto de elementos ou atributos deve ser único dentro de um determinado escopo e, além de único, pode ser referenciado.

Considere a restrição de chave Autor.key na Listagem B.2. Esta restrição especifica que o valor do elemento id_autor é único dentro do escopo do conjunto de elementos Autor.

A Listagem B.3 mostra como uma key é declarada. Como podemos observar, o construtor key contém:

- um atributo name (Chave), que identifica unicamente a restrição.

- um elemento selector (Seletor1), que especifica o escopo, através de uma expressão de caminho, para o qual a restrição é declarada. Define o elemento que vai possuir a chave.
- um ou mais elementos field (F1_1,...,F1_n), que vai determinar, através de expressões de caminho, o conjunto de elementos ou atributos que deve ser único dentro do escopo especificado pelo selector.

```
<key name="Chave">
   <selector xpath="Seletor1"/>
   <field xpath="F1_1"/>
   ....
   <field xpath="F1_n"/>
</key>
```

Listagem B.3 – Declaração de uma chave

B.3.8 Restrição de Integridade Referencial

XML Schema permite declarar, através do construtor keyref, que o valor de um conjunto de elementos ou atributos é uma referência ao valor de um outro conjunto de elementos ou atributos.

Considere a restrição de integridade referencial Artigo.keyref na Listagem B.2. Esta restrição especifica que, dentro do escopo do conjunto de elementos Artigo, o elemento autor_ref deste elemento referencia a chave Autor.key. Ou seja, para qualquer autor_ref em Artigo, existe um id_autor em Autor tal que id_autor = autor_ref.

A Listagem B.4 mostra como uma keyref é declarada. O construtor keyref contém:

- um atributo name (Referencia), que identifica unicamente a restrição.
- um atributo refer (Chave), que indica a qual chave está associada a referência.
- um elemento selector (Seletor2), que especifica o escopo, através de uma expressão de caminho, para o qual a restrição é declarada.
- um ou mais elementos field (F2_1,..., F2_n), que vai determinar, através de expressões de caminho, o conjunto de elementos ou atributos que deve corresponder ao conjunto de elementos ou atributos especificado pela restrição de chave correspondente.

```
<keyref name="Referencia" refer="Chave">
   <selector xpath="Seletor2"/>
   <field xpath="F2_1"/>
   ...
   <field xpath="F2_n"/>
</keyref>
```

Listagem B.4 – Declaração de restrição de integridade referencial

B.3.9 Restrição de Unicidade

A restrição de identidade unique assegura que cada seqüência de elementos – o conjunto de valores especificados por expressões de caminho dos elementos field aninhados – é única. Uma seqüência de elementos é única se não possui dois valores iguais para todos os elementos da seqüência. A restrição de identidade unique é apropriada, por exemplo, para elementos opcionais ou atributos. Um exemplo é mostrado na Listagem B.2 onde a restrição de unicidade Autor.unique.e_mail define que todo e-mail de um autor deve ser único dentre outros autores.

A Listagem B.5 mostra como um construtor unique é declarado. O construtor unique contém:

- um atributo name (Unicidade), que identifica unicamente a restrição de identidade.
- um elemento selector (Seletor3), que especifica o escopo, através de uma expressão de caminho, para o qual a restrição de unicidade é declarada.
- um ou mais elementos field (F3_1,..., F3_n), que vai determinar, através de expressões de caminho, o conjunto de elementos ou atributos que deve corresponder à restrição de unicidade.

```
<unique name="Unicidade">
   <selector xpath="Seletor3"/>
   <field xpath="F3_1"/>
      ...
   <field xpath="F3_n"/>
</unique>
```

Listagem B.5 – Declaração de restrição de unicidade

Apêndice C – Algoritmos e Códigos-Fonte

Este apêndice apresenta algoritmos e os códigos-fonte de SXMLSs utilizados durante este trabalho.

C.1 Algoritmo de Integração de SXMLS do WISE

Este algoritmo é utilizado durante o processo de integração de esquemas SXMLSs.

Normalmente para representar dados georreferencidados são utilizados os Esquemas Georreferenciados Aplicação, que necessitam possuir como seu agregador um Esquema Convenvional Aplicação. No caso de dados convencionais só é necessário o Esquema Convencional Aplicação. Porém para facilitar o entendimento deste algoritmo foi considerado que quando usando dados georreferenciados, os Esquemas Georreferenciados Aplicação e o seu Esquema Convencional Aplicação, constituam um único esquema. Desta forma, quando for citado um esquema convencional, será usado o termo Esquema Convencional, e esquema georreferenciado o termo Esquema Georreferenciado. O esquema integrado resultante também obedece a esta regra.

```
1- Seja E1 e E2 dois SXMLSs a serem integrados, e E3 o ISXMLS
   resultante da integração de E1 e E2;
2- Se destino de E3 é "Esquema Integrado Convencional" então
  2.1- Se E1="Esquema Convencional" e E2="Esquema Convencional" então
     2.1.1- Para cada Conjunto de Dados CD1, em E1 faça
                Para cada Conjunto de Dados CD2, em E2 faça
                   Se CD1[i] = CD2[j], segundo informações contidas no
                   elemento informativo Ontology.Name de cada Conjunto
                   de Dados então
                      ConjDadosSemelhantes ← informação de semelhança
                      entre CD1[i] e CD2[j]
     2.1.2- Apresentar ConjDadosSemelhantes, contendo todos os
            relacionamentos entre Conjuntos de Dados semelhantes,
            para o usuário que está integrando os esquemas validar e,
            se necessário selecionar novos relacionamentos entre os
            Conjuntos de Dados. Também apresentar a este usuário os
            outros Conjuntos de Dados (ConjDadosNãoSemelhantes) que
            não foram contemplados em ConjDadosSemelhantes. Permitir
            ao usuário renomear os Conjuntos de Dados destino.
            Atualizar ConjDadosSemelhantes e ConjDadosNãoSemelhantes.
     2.1.3- Para cada relacionamento R em ConjDadosSemelhantes entre
            um Conjunto de Dados CD1, em E1 e um Conjunto de Dados
            CD2, em E2 faça
```

Se Nome do Conjunto de Dados CD1[i] <> Nome do Conjunto de Dados CD2[i] então
Se usuário não renomeou Conjunto de Dados CD1[i] ou CD2[i] então
NomesConjDadosDestino ● CD1[i].Nome
Para cada Campo C1, em CD1[i] faça
Para cada Campo C2, em CD2[i] faça
Se C1[g] = C2[h], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
CamposSemelhantes ● informação de semelhança entre CD1[i].C1[g] e CD2[j].C2[h]

2.1.4- Apresentar CamposSemelhantes, contendo todos os relacionamentos entre Campos semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre os Campos. Também apresentar a este usuário os outros Campos (CamposNãoSemelhantes) que não foram contemplados em CamposSemelhantes. Permitir ao usuário renomear os Campos destino. Permitir ao usuário definir a(s) UnitOfMeasure destino. Finalmente atualizar CamposSemelhantes e CamposNãoSemelhantes.

2.1.5- Para cada relacionamento R em CamposSemelhantes entre um Campo C1, em Conjunto de Dados CD1, em E1 e um Campo C2, em Conjunto de Dados CD2, em E2 faça
Se Nome do Campo C1[g] <> Nome do Campo C2[g], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
Se usuário não renomeou Campos C1[g] ou C2[g] então
NomesCamposDestino ● CD1[i].C1[g].Nome
Para cada Campo, C1, em CD1[i] faça
Para cada Campo, C2, em CD2[i] faça
Se elemento UnitOfMeasure (UOM) de C1[g] <> UnitOfMeasure de C2[g] então
Se usuário não definiu UOM destino de C1[g] ou C2[g] então
UOMDiferentes ← informação de desigualdade entre CD1[i].C1[g].UOM e CD2[i].C2[g].UOM, usar CD1[i].C1[g].UOM como unidade de medida padrão para o campo destino

2.2- Se E1="Esquema Convencional" e E2="Esquema Georreferenciado" então

2.2.1- Para cada Conjunto de Dados CD, em E1 faça
Para cada Coleção de Feições CF, em E2 faça
Se CD[i] = CF[j], segundo informações contidas no elemento informativo Ontology.Name de cada Conjunto de Dados ou Coleção de Feições então
ConjDadosColFeicoesSemelhantes ← informação de semelhança entre CD[i] e CF[j]

2.2.2- Apresentar ConjDadosColFeicoesSemelhantes, contendo todos os relacionamentos entre Conjuntos de Dados e Coleções de Feições semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre os Conjuntos de Dados e Coleções de Feições. Também apresentar a este usuário os outros Conjuntos de Dados e Coleção de Feições (ConjDadosColFeicoesNãoSemelhantes) que não foram contemplados em ConjDadosColFeicoesSemelhantes. Permitir ao usuário renomear os Conjuntos de Dados e Coleções de Feições destino. Atualizar ConjDadosColFeicoesSemelhantes e ConjDadosColFeicoesNãoSemelhantes.

2.2.3- Para cada relacionamento R em ConjDadosColFeicoesSemelhantes entre um Conjunto de Dados CD em E1 e uma Coleção de Feições CF, em E2 faça
Se Nome do Conjunto de Dados CD[i] <> Nome da Coleção de Feições CF[i] então
Se usuário não renomeou Conjunto de Dados CD[i] ou Coleção de Feições CF[i] então
NomesConjDadosColFeicoesDestino ← CD[i].Nome
Para cada Campo C1, em CD[i] faça
Para cada Campo C2, em CF[i] faça
Se C1[g] = C2[h], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
CamposSemelhantes ← informação de semelhança entre CD[i].C1[g] e CF[j].C2[h]

2.2.4- Apresentar CamposSemelhantes, contendo todos os relacionamentos entre Campos semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre os Campos. Também apresentar a este usuário os outros Campos (CamposNãoSemelhantes) que não foram contemplados em CamposSemelhantes. Permitir ao usuário renomear os Campos destino. Permitir ao usuário definir a(s) UnitOfMeasure destino. Finalmente atualizar CamposSemelhantes e CamposNãoSemelhantes.

2.2.5- Para cada relacionamento R em CamposSemelhantes entre um Campo C1, em Conjunto de Dados CD em E1 e um Campo C2, em Coleção de Feições CF, em E2 faça
Se Nome do Campo C1[g] <> Nome do Campo C2[g], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
Se usuário não renomeou Campos C1[g] ou C2[g] então
NomesCamposDestino ← CD[i].C1[g].Nome
Para cada Campo, C1, em CD[i] faça
Para cada Campo, C2, em CF[i] faça
Se elemento UnitOfMeasure (UOM) de C1[g] <> UnitOfMeasure de C2[g] então
Se usuário não definiu UOM destino de C1[g] ou C2[g] então
UOMDiferentes ● informação de desigualdade entre CD[i].C1[g].UOM e CF[i].C2[g].UOM, usar CD[i].C1[g].UOM como unidade de medida padrão para o campo destino

2.3- Gerar ISXMLS Esquema Integrado Convencional, E3, com informações adquiridas até este passo.

3- Se destino de E3 é Esquema Georreferenciado então

3.1- Se E1="Esquema Georreferenciado" e E2="Esquema Georreferenciado" então

3.1.1- Para cada Coleção de Feições CF1, em E1 faça
Para cada Coleção de Feições CF2, em E2 faça
Se CF1[i] = CF2[j], segundo informações contidas no elemento informativo Ontology.Name de cada Coleção de Feições então
ColFeicoesSemelhantes ● informação de semelhança entre CF1[i] e CF2[j]

3.1.2- Apresentar ColFeicoesSemelhantes, contendo todos os relacionamentos entre Coleções de Feições semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre as Coleções de Feições. Também apresentar a este usuário as outras Coleções de Feições (ColFeicoesNãoSemelhantes) que

não foram contempladas em ColFeicoesSemelhantes. Permitir ao usuário renomear as Coleções de Feições destino. Permitir ao usuário selecionar Sistema de Projeção/Coordenada destino. Atualizar ColFeicoesSemelhantes e ColFeicoesNãoSemelhantes.

3.1.3- Para cada relacionamento R em ColFeicoesSemelhantes entre uma Coleção de Feições CF1, em E1 e uma Coleção de Feições CF2, em E2 faça
Se Nome da Coleção de Feições CF1[i] <> da Coleção de Feições CF2[i] então
Se usuário não renomeou Coleções de Feições CF1[i] ou CF2[i] então
NomesColFeicoesDestino ● CF1[i].Nome
Se Sistema de Projeção/Coordenada em CF1[i] <> Sistema de Projeção/Coordenada em CF2[i] então
Se usuário definiu Sistema de Projeção/Coordenada destino então
SistProjCoordDest ← SistProjCoordDestinoUsuario
Senão
SistProjCoordDest ● CF1[i].SistProjCoord
Para cada Campo C1, em CF1[i] faça
Para cada Campo C2, em CF2[i] faça
Se C1[g] = C2[h], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
CamposSemelhantes ← informação de semelhança entre CF1[i].C1[g] e CF2[j].C2[h]

3.1.4- Apresentar CamposSemelhantes, contendo todos os relacionamentos entre Campos semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre os Campos. Também apresentar a este usuário os outros Campos (CamposNãoSemelhantes) que não foram contemplados em CamposSemelhantes. Permitir ao usuário renomear os Campos destino. Permitir ao usuário definir a(s) UnitOfMeasure destino. Finalmente atualizar CamposSemelhantes e CamposNãoSemelhantes.

3.1.5- Para cada relacionamento R em CamposSemelhantes entre um Campo C1, em Coleção de Feições CF1, em E1 e um Campo C2, em Coleção de Feições CF2, em E2 faça
Se Nome do Campo C1[g] <> Nome do Campo C2[g], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
Se usuário não renomeou Campos C1[g] ou C2[g] então
NomesCamposDestino ← CD1[i].C1[g].Nome
Para cada Campo, C1, em CF1[i] faça
Para cada Campo, C2, em CF2[i] faça
Se elemento UnitOfMeasure (UOM) de C1[g] <> UnitOfMeasure de C2[g] então
Se usuário não definiu UOM destino de C1[g] ou C2[g] então
UOMDiferentes ← informação de desigualdade entre CF1[i].C1[g].UOM e CF2[i].C2[g].UOM, usar CF1[i].C1[g].UOM como unidade de medida padrão para o campo destino

3.2- Se E1="Esquema Convencional" e E2="Esquema Georreferenciado" então

3.2.1- Para cada Conjunto de Dados CD, em E1 faça
Para cada Coleção de Feições CF, em E2 faça

Se CD[i] = CF[j], segundo informações contidas no elemento informativo Ontology.Name de cada Conjunto de Dados ou Coleção de Feições então
ConjDadosColFeicoesSemelhantes ← informação de semelhança entre CD[i] e CF[j]

3.2.2- Apresentar ConjDadosColFeicoesSemelhantes, contendo todos os relacionamentos entre Conjuntos de Dados e Coleções de Feições semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre os Conjuntos de Dados e Coleções de Feições. Também apresentar a este usuário os outros Conjuntos de Dados e Coleção de Feições (ConjDadosColFeicoesNãoSemelhantes) que não foram contemplados em ConjDadosColFeicoesSemelhantes. Permitir ao usuário renomear os Conjuntos de Dados e Coleções de Feições destino. Permitir ao usuário selecionar Sistema de Projeção/Coordenada destino. Atualizar ConjDadosColFeicoesSemelhantes e ConjDadosColFeicoesNão-Semelhantes.

3.2.3- Para cada relacionamento R em ConjDadosColFeicoesSemelhantes entre um Conjunto de Dados CD em E1 e uma Coleção de Feições CF, em E2 faça
Se Nome do Conjunto de Dados CD[i] <> Nome da Coleção de Feições CF[i] então
Se usuário não renomeou Conjunto de Dados CD[i] ou Coleção de Feições CF[i] então
NomesConjDadosColFeicoesDestino ● CD[i].Nome
Se usuário definiu Sistema de Projeção/Coordenada destino de CF[i] então
SistProjCoordDest ← SistProjCoordDestinoUsuario
Senão
SistProjCoordDest ● CF1[i].SistProjCoord
Para cada Campo C1, em CD[i] faça
Para cada Campo C2, em CF[i] faça
Se C1[g] = C2[h], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
CamposSemelhantes ● informação de semelhança entre CD[i].C1[g] e CF[j].C2[h]

3.2.4- Apresentar CamposSemelhantes, contendo todos os relacionamentos entre Campos semelhantes, para o usuário que está integrando os esquemas validar e, se necessário selecionar novos relacionamentos entre os Campos. Também apresentar a este usuário os outros Campos (CamposNãoSemelhantes) que não foram contemplados em CamposSemelhantes. Permitir ao usuário renomear os Campos destino. Permitir ao usuário definir a(s) UnitOfMeasure destino. Finalmente atualizar CamposSemelhantes e CamposNãoSemelhantes.

3.2.5- Para cada relacionamento R em CamposSemelhantes entre um Campo C1, em Conjunto de Dados CD em E1 e um Campo C2, em Coleção de Feições CF, em E2 faça
Se Nome do Campo C1[g] <> Nome do Campo C2[g], segundo informações contidas no elemento informativo Ontology.Name de cada Campo então
Se usuário não renomeou Campos C1[g] ou C2[g] então
NomesCamposDestino ← CD[i].C1[g].Nome
Para cada Campo, C1, em CD[i] faça
Para cada Campo, C2, em CF[i] faça
Se elemento UnitOfMeasure (UOM) de C1[g] <> UnitOfMeasure de C2[g] então

```
                    Se usuário não definiu UOM destino de C1[g] ou
                C2[g] então
                    UOMDiferentes ← informação de desigualdade
                entre CD[i].C1[g].UOM e CF[i].C2[g].UOM, usar
                CD[i].C1[g].UOM como unidade de medida padrão
                para o campo destino
3.3- Gerar ISXMLS Esquema Integrado Georreferenciado, E3, com
    informações adquiridas até este passo.
```

C.2 SXMLSs

Esta seção apresenta alguns dos SXMLSs usados durante os capítulos 4 e 5.

C.2.1 Esquema para representar Tipos de Dados Simples WISE

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ================================================================
     simple  types
     ================================================================ -->
    <simpleType name="wiseBoolean">
        <restriction base="boolean"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseByte">
        <restriction base="byte"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseDate">
        <restriction base="date"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseDateTime">
        <restriction base="dateTime"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseDouble">
        <restriction base="double"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseFloat">
        <restriction base="float"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseID">
        <restriction base="ID"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseIDREF">
        <restriction base="IDREF"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseIDREFS">
        <restriction base="IDREFS"/>
    </simpleType>
    <!--================================================================ -->
    <simpleType name="wiseInt">
        <restriction base="int"/>
    </simpleType>
    <!--================================================================ -->
```

```
    <simpleType name="wiseLong">
        <restriction base="long"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseShort">
        <restriction base="short"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseString">
        <restriction base="string"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseTime">
        <restriction base="time"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseUnsignedByte">
        <restriction base="unsignedByte"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseUnsignedInt">
        <restriction base="unsignedInt"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseUnsignedLong">
        <restriction base="unsignedLong"/>
    </simpleType>
    <!-- ================================================================== -->
    <simpleType name="wiseUnsignedShort">
        <restriction base="unsignedShort"/>
    </simpleType>
    <!-- ================================================================== -->
</schema>
```

C.2.2 SXMLSs do Caso de Teste

Os SXMLSs usados no caso de teste do Capítulo 5 que não estão no mesmo, foram colocados nesta seção.

C.2.2.1 Esquema Georreferenciado Aplicação BaciaParSulEstRioJaneiro.xsd Extraído

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
    <element name="BaciaParSulEstRioJaneiro" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Sub-bacias da Bacia do Rio Paraíba do Sul que passam pelo
estado do Rio de Janeiro"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_South_American_1969]]></element>
                <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]></element
>
```

```
            <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_South_American_1969", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
          </appinfo>
        </annotation>
    </element>
    <!--================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ==================================================================
     complex types
     ================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Area" type="wisesxmls:wiseFloat"/>
                    <element name="Perimeter" type="wisesxmls:wiseFloat"/>
                    <element name="Baciaparsulestriojaneiro_" type="wisesxmls:wiseInt"/>
                    <element name="Baciaparsulestriojaneiro_id" type="wisesxmls:wiseInt"/>
                    <element name="Nome" type="wisesxmls:wiseString"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

C.2.2.2 Esquema Convencional Aplicação BaciaRioParaibaSul.xsd Extraído

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:gml="http://www.opengis.net/gml" elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- ==================================================================
     includes and imports
     ================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <include schemaLocation="SXMLS-Base.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <include schemaLocation="BacParSul.xsd"/>
    <include schemaLocation="MunBacParSul.xsd"/>
    <include schemaLocation="DistritBacParSul.xsd"/>
    <include schemaLocation="SolosBacParSul.xsd"/>
    <!-- ==================================================================
     global declarations
     ================================================================== -->
    <element name="BaseData" type="wisesxmls:BaseSchemaType" substitutionGroup="wisesxmls:_BaseData">
        <annotation>
            <appinfo>
                <element name="WISE=BaseData=Source">"Laboratório de Hidrologia - COPPE/UFRJ"</element>
                <element name="WISE=Description">"Dados extraídos da Bacia do Rio Paraíba do Sul"</element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================== -->
    <element name="BaciaRioParaibaSul" substitutionGroup="wisesxmls:_DataSource">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Fonte de dados da Bacia do Rio Paraíba do Sul. Contém
informações das bacias, solos da região, municípios e distritos abrangidos pela bacia do Rio Paraíba do
Sul"</element>
            </appinfo>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="wisesxmls:AbstractDataSourceType"/>
```

```
        </complexContent>
      </complexType>
    </element>
    <!--================================================================== -->
    <element name="DataSets" substitutionGroup="wisesxmls:_DataSets">
      <complexType>
        <complexContent>
          <extension base="wisesxmls:AbstractDataSetsType"/>
        </complexContent>
      </complexType>
    </element>
    <!--================================================================== -->
    <element name="BacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
    <!--================================================================== -->
    <element name="MunBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
    <!--================================================================== -->
    <element name="DistritBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
    <!--================================================================== -->
    <element name="SolosBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection"/>
    <!--================================================================== -->
</schema>
```

C.2.2.3 Esquema Georreferenciado Aplicação BacParSul.xsd Extraído

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ===================================================================
     includes and imports
     ================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ===================================================================
     global declarations
     ================================================================== -->
    <element name="BacParSul" type="wisesxmls:FeatureCollectionType" substitutionGroup="gml:_FeatureCollection">
      <annotation>
        <appinfo>
          <element name="WISE=Description">"Bacia do Rio Paraíba do Sul"</element>
          <element name="WISE=FeatureCollection=Datum"><![CDATA[D_Corrego_Alegre]]></element>
          <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["International_1924",6378388,297]]]></element>
          <element
name="WISE=FeatureCollection=ProjectionSystem"><![CDATA[PROJCS["Corrego_Alegre_UTM_Zone_23S",
COORDINATESYSTEM, PROJECTION["Transverse_Mercator"], PARAMETER["False_Easting",500000],
PARAMETER["False_Northing",10000000], PARAMETER["Central_Meridian",-45],
PARAMETER["Scale_Factor",0.9996], PARAMETER["Latitude_Of_Origin",0], UNIT["Meter",1]]]]></element>
          <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_Corrego_Alegre", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
        </appinfo>
      </annotation>
    </element>
    <!--================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ===================================================================
     complex types
     ================================================================== -->
    <complexType name="FeatureCollectionType">
      <complexContent>
        <extension base="gml:AbstractFeatureCollectionType"/>
      </complexContent>
    </complexType>
    <!--================================================================== -->
    <complexType name="ApplicationFeatureType">
      <complexContent>
```

```
        <extension base="gml:AbstractFeatureType">
          <sequence>
            <element ref="gml:geometryMember"/>
            <element name="Area" type="wisesxmls:wiseFloat"/>
            <element name="Perimeter" type="wisesxmls:wiseFloat"/>
            <element name="Bacparsul_" type="wisesxmls:wiseInt"/>
            <element name="Bacparsul_id" type="wisesxmls:wiseInt"/>
            <element name="Nome" type="wisesxmls:wiseString"/>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
</schema>
```

C.2.2.4 Esquema Georreferenciado Aplicação MunBacParSul.xsd Extraído

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ============================================================
     includes and imports
     ============================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ============================================================
     global declarations
     ============================================================ -->
    <element name="MunBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
          <appinfo>
            <element name="WISE=Description">"Municípios da Bacia do Rio Paraíba do Sul"</element>
            <element name="WISE=FeatureCollection=Datum"><![CDATA[D_Corrego_Alegre]]></element>
            <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["International_1924",6378388,297]]]></element>
            <element
name="WISE=FeatureCollection=ProjectionSystem"><![CDATA[PROJCS["Corrego_Alegre_UTM_Zone_23S",
COORDINATESYSTEM, PROJECTION["Transverse_Mercator"], PARAMETER["False_Easting",500000],
PARAMETER["False_Northing",10000000], PARAMETER["Central_Meridian",-45],
PARAMETER["Scale_Factor",0.9996], PARAMETER["Latitude_Of_Origin",0], UNIT["Meter",1]]]]></element>
            <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_Corrego_Alegre", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
          </appinfo>
        </annotation>
    </element>
    <!--============================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ============================================================
     complex types
     ============================================================ -->
    <complexType name="FeatureCollectionType">
        <complexContent>
          <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--============================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
          <extension base="gml:AbstractFeatureType">
            <sequence>
              <element ref="gml:geometryMember"/>
              <element name="Area" type="wisesxmls:wiseFloat"/>
              <element name="Perimeter" type="wisesxmls:wiseFloat"/>
              <element name="Munbacparsul_" type="wisesxmls:wiseInt"/>
              <element name="Munbacparsul_id" type="wisesxmls:wiseInt"/>
              <element name="Nome_muni" type="wisesxmls:wiseString"/>
              <element name="Nome_est" type="wisesxmls:wiseString"/>
```

```
            </sequence>
          </extension>
        </complexContent>
    </complexType>
</schema>
```

C.2.2.5 Esquema Georreferenciado Aplicação DistritBacParSul.xsd Extraído

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ======================================================================
     includes and imports
     ====================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ======================================================================
     global declarations
     ====================================================================== -->
    <element name="DistritBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Distritos de municípios da Bacia do Rio Paraíba do Sul"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_Corrego_Alegre]]></element>
                <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["International_1924",6378388,297]]]></element>
                <element
name="WISE=FeatureCollection=ProjectionSystem"><![CDATA[PROJCS["Corrego_Alegre_UTM_Zone_23S",
COORDINATESYSTEM, PROJECTION["Transverse_Mercator"], PARAMETER["False_Easting",500000],
PARAMETER["False_Northing",10000000], PARAMETER["Central_Meridian",-45],
PARAMETER["Scale_Factor",0.9996], PARAMETER["Latitude_Of_Origin",0], UNIT["Meter",1]]]]></element>
                <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_Corrego_Alegre", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
            </appinfo>
        </annotation>
    </element>
    <!--======================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ======================================================================
     complex types
     ====================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--======================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Area" type="wisesxmls:wiseFloat"/>
                    <element name="Perimeter" type="wisesxmls:wiseFloat"/>
                    <element name="Distritbacparsul_" type="wisesxmls:wiseInt"/>
                    <element name="Distritbacparsul_id" type="wisesxmls:wiseInt"/>
                    <element name="Nome_dist" type="wisesxmls:wiseString"/>
                    <element name="Nome_muni" type="wisesxmls:wiseString"/>
                    <element name="Nome_est" type="wisesxmls:wiseString"/>
                    <element name="Pop_urb" type="wisesxmls:wiseInt"/>
                    <element name="Pop_rur" type="wisesxmls:wiseInt"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

C.2.2.6 Esquema Georreferenciado Aplicação SolosBacParSul.xsd Extraído

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <!-- ======================================================================
     includes and imports
     ====================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ======================================================================
     global declarations
     ====================================================================== -->
    <element name="SolosBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Solos da região da Bacia do Rio Paraíba do Sul"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_South_American_1969]]></element>
                <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]></element
>
                <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_South_American_1969", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
            </appinfo>
        </annotation>
    </element>
    <!--====================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ======================================================================
     complex types
     ====================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--====================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Area" type="wisesxmls:wiseFloat"/>
                    <element name="Perimeter" type="wisesxmls:wiseFloat"/>
                    <element name="Solos_" type="wisesxmls:wiseInt"/>
                    <element name="Tiposolo" type="wisesxmls:wiseString"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

C.2.2.7 Esquema Georreferenciado Aplicação BaciaParSulEstRioJaneiro.xsd com Elementos Informativos

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ======================================================================
     includes and imports
     ====================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ======================================================================
     global declarations
```

```
    ========================================================================== -->
    <element name="BaciaParSulEstRioJaneiro" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Sub-bacias da Bacia do Rio Paraíba do Sul que passam pelo
estado do Rio de Janeiro"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_South_American_1969]]></element>
                <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]></element
>
                <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_South_American_1969", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
                <element name="WISE=Ontology.Name">"BaciaParSulEstRioJaneiro", "Bacia", "Paraíba do Sul", "Sub-
bacias", "Bacias Rio de Janeiro"</element>
            </appinfo>
        </annotation>
    </element>
    <!--========================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ========================================================================
     complex types
     ========================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--========================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Area" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Área da bacia"</element>
                                <element name="WISE=Ontology.Name">"Area", "Área", "Área Bacia", "Extensão"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Perimeter" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Perímetro da bacia"</element>
                                <element name="WISE=Ontology.Name">"Perimeter", "Perímetro"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Baciaparsulestriojaneiro_" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Número da bacia"</element>
                                <element name="WISE=Ontology.Name">"Baciaparsulestriojaneiro_", "Número", "Número
Bacia"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Baciaparsulestriojaneiro_id" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Identificador da bacia"</element>
                                <element name="WISE=Ontology.Name">"Baciaparsulestriojaneiro_id", "Identificador",
"Id"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Nome" type="wisesxmls:wiseString">
                        <annotation>
```

```
                    <appinfo>
                        <element name="WISE=Description">"Nome da bacia"</element>
                        <element name="WISE=Ontology.Name">"Nome", "Bacia"</element>
                    </appinfo>
                </annotation>
            </element>
        </sequence>
    </extension>
</complexContent>
</complexType>
</schema>
```

C.2.2.8 Esquema Georreferenciado Aplicação BacParSul.xsd com Elementos Informativos

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ==========================================================
     includes and imports
     ========================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ==========================================================
     global declarations
     ========================================================== -->
    <element name="BacParSul" type="wisesxmls:FeatureCollectionType" substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Bacia do Rio Paraíba do Sul"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_Corrego_Alegre]]></element>
                <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["International_1924",6378388,297]]]></element>
                <element
name="WISE=FeatureCollection=ProjectionSystem"><![CDATA[PROJCS["Corrego_Alegre_UTM_Zone_23S",
COORDINATESYSTEM, PROJECTION["Transverse_Mercator"], PARAMETER["False_Easting",500000],
PARAMETER["False_Northing",10000000], PARAMETER["Central_Meridian",-45],
PARAMETER["Scale_Factor",0.9996], PARAMETER["Latitude_Of_Origin",0], UNIT["Meter",1]]]]></element>
                <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_Corrego_Alegre", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
                <element name="WISE=Ontology.Name">"BacParSul", "Bacia", "Rio Paraíba do Sul"</element>
            </appinfo>
        </annotation>
    </element>
    <!--========================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ==========================================================
     complex types
     ========================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--========================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Shape" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Tipo do shape, define a sua semiologia"</element>
                                <element name="WISE=Ontology.Name">"Shape", "Tipo shape"</element>
                            </appinfo>
                        </annotation>
                    </element>
```

```
                <element name="Area" type="wisesxmls:wiseFloat">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Área da bacia"</element>
                            <element name="WISE=Ontology.Name">"Area", "Área", "Área Bacia"</element>
                            <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Perimeter" type="wisesxmls:wiseFloat">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Perímetro da Bacia"</element>
                            <element name="WISE=Ontology.Name">"Perimeter", "Perímetro", "Perímetro
 Bacia"</element>
                            <element name="WISE=Field=UnitOfMeasure">"m"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Bacparsul_" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Número da bacia"</element>
                            <element name="WISE=Ontology.Name">"Bacparsul_", "Número", "Número
 Bacia"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Bacparsul_id" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Identificador da bacia"</element>
                            <element name="WISE=Ontology.Name">"Bacparsul_id", "Identificador", "Id"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Nome" type="wisesxmls:wiseString">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Nome da Bacia"</element>
                            <element name="WISE=Ontology.Name">"Nome", "Bacia", "Nome Bacia", "Denominação",
 "Rio"</element>
                        </appinfo>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
  </complexType>
</schema>
```

C.2.2.9 Esquema Georreferenciado Aplicação MunBacParSul.xsd com Elementos Informativos

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ================================================================
     global declarations
     ================================================================ -->
   <element name="MunBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
```

```
            <element name="WISE=Description">"Municípios da Bacia do Rio Paraíba do Sul"</element>
            <element name="WISE=FeatureCollection=Datum"><![CDATA[D_Corrego_Alegre]]></element>
            <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["International_1924",6378388,297]]]></element>
            <element
name="WISE=FeatureCollection=ProjectionSystem"><![CDATA[PROJCS["Corrego_Alegre_UTM_Zone_23S",
COORDINATESYSTEM, PROJECTION["Transverse_Mercator"], PARAMETER["False_Easting",500000],
PARAMETER["False_Northing",10000000], PARAMETER["Central_Meridian",-45],
PARAMETER["Scale_Factor",0.9996], PARAMETER["Latitude_Of_Origin",0], UNIT["Meter",1]]]]></element>
            <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_Corrego_Alegre", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
            <element name="WISE=Ontology.Name">"MunBacParSul", "Municípios", "Municípios Bacia Rio Paraíba
do Sul"</element>
        </appinfo>
    </annotation>
</element>
<!--========================================================================== -->
<element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
<!-- ==========================================================================
 complex types
 ========================================================================== -->
<complexType name="FeatureCollectionType">
    <complexContent>
        <extension base="gml:AbstractFeatureCollectionType"/>
    </complexContent>
</complexType>
<!--========================================================================== -->
<complexType name="ApplicationFeatureType">
    <complexContent>
        <extension base="gml:AbstractFeatureType">
            <sequence>
                <element ref="gml:geometryMember"/>
                <element name="Shape" type="wisesxmls:wiseString">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Tipo do shape, define a sua semiologia"</element>
                            <element name="WISE=Ontology.Name">"Shape", "Tipo shape"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Area" type="wisesxmls:wiseFloat">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Área do município"</element>
                            <element name="WISE=Ontology.Name">"Area", "Área", "Área Município"</element>
                            <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Perimeter" type="wisesxmls:wiseFloat">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Perímetro do município"</element>
                            <element name="WISE=Ontology.Name">"Perimeter", "Perímetro", "Perímetro
Município"</element>
                            <element name="WISE=Field=UnitOfMeasure">"m"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Munbacparsul_" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Número do município"</element>
                            <element name="WISE=Ontology.Name">"Munbacparsul_", "Número", "Número
Município"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Munbacparsul_id" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"Identificador do município"</element>
```

```
                        <element name="WISE=Ontology.Name">"Munbacparsul_id", "Identificador", "Id"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Nome_muni" type="wisesxmls:wiseString">
                <annotation>
                    <appinfo>
                        <element name="WISE=Description">"Nome do município"</element>
                        <element name="WISE=Ontology.Name">"Nome_muni", "Nome", "Nome Município",
"Município"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Nome_est" type="wisesxmls:wiseString">
                <annotation>
                    <appinfo>
                        <element name="WISE=Description">"Nome do estado"</element>
                        <element name="WISE=Ontology.Name">"Nome_est", "Nome", "Nome Estado",
"Estado"</element>
                    </appinfo>
                </annotation>
            </element>
        </sequence>
    </extension>
  </complexContent>
</complexType>
</schema>
```

C.2.2.10 Esquema Georreferenciado Aplicação DistritBacParSul.xsd com Elementos Informativos

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
  <!-- ======================================================================
   includes and imports
   ====================================================================== -->
  <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
  <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
  <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
  <!-- ======================================================================
   global declarations
   ====================================================================== -->
  <element name="DistritBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
    <annotation>
      <appinfo>
        <element name="WISE=Description">"Distritos de municípios da Bacia do Rio Paraíba do Sul"</element>
        <element name="WISE=FeatureCollection=Datum"><![CDATA[D_Corrego_Alegre]]></element>
        <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["International_1924",6378388,297]]]></element>
        <element
name="WISE=FeatureCollection=ProjectionSystem"><![CDATA[PROJCS["Corrego_Alegre_UTM_Zone_23S",
COORDINATESYSTEM, PROJECTION["Transverse_Mercator"], PARAMETER["False_Easting",500000],
PARAMETER["False_Northing",10000000], PARAMETER["Central_Meridian",-45],
PARAMETER["Scale_Factor",0.9996], PARAMETER["Latitude_Of_Origin",0], UNIT["Meter",1]]]]></element>
        <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_Corrego_Alegre", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
        <element name="WISE=Ontology.Name">"DistritBacParSul", "Distritos", "Distritos Bacia Rio Paraíba do
Sul"</element>
      </appinfo>
    </annotation>
  </element>
  <!--====================================================================== -->
  <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
  <!-- ======================================================================
   complex types
   ====================================================================== -->
  <complexType name="FeatureCollectionType">
    <complexContent>
```

```
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--============================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Shape" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Tipo do shape, define a sua semiologia"</element>
                                <element name="WISE=Ontology.Name">"Shape", "Tipo shape"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Area" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Área do distrito"</element>
                                <element name="WISE=Ontology.Name">"Area", "Área", "Área Distrito"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Perimeter" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Perímetro do distrito"</element>
                                <element name="WISE=Ontology.Name">"Perimeter", "Perímetro", "Perímetro
Distrito"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Distritbacparsul_" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Número do distrito"</element>
                                <element name="WISE=Ontology.Name">"Distritbacparsul_", "Número", "Número
Distrito"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Distritbacparsul_id" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Identificador do distrito"</element>
                                <element name="WISE=Ontology.Name">"Distritbacparsul_id", "Identificador", "Id"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Nome_dist" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Nome do distrito"</element>
                                <element name="WISE=Ontology.Name">"Nome_dist", "Nome", "Nome Distrito",
"Distrito"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Nome_muni" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Nome do município"</element>
                                <element name="WISE=Ontology.Name">"Nome_muni", "Nome", "Nome Município",
"Município"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Nome_est" type="wisesxmls:wiseString">
                        <annotation>
```

```
                        <appinfo>
                            <element name="WISE=Description">"Nome do estado"</element>
                            <element name="WISE=Ontology.Name">"Nome_est", "Nome", "Nome Estado",
"Estado"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Pop_urb" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"População Urbana"</element>
                            <element name="WISE=Ontology.Name">"Pop_urb", "População", "População
Urbana"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Pop_rur" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element name="WISE=Description">"População Rural"</element>
                            <element name="WISE=Ontology.Name">"Pop_rur", "População", "População
Rural"</element>
                        </appinfo>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</schema>
```

C.2.2.11 Esquema Georreferenciado Aplicação SolosParSul.xsd com Elementos Informativos

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!-- ============================================================
     includes and imports
     ============================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ============================================================
     global declarations
     ============================================================ -->
    <element name="SolosBacParSul" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Solos da região da Bacia do Rio Paraíba do Sul"</element>
                <element name="WISE=FeatureCollection=Datum"><![CDATA[D_South_American_1969]]></element>
                <element
name="WISE=FeatureCollection=Ellipsoid"><![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]></element
>
                <element
name="WISE=FeatureCollection=CoordinateSystem"><![CDATA[GEOGCS["GCS_South_American_1969", DATUM,
PRIMEM["Greenwich",0], UNIT["Degree",0.0174532925199433]]]]></element>
                <element name="WISE=Ontology.Name">"SolosBacParSul", "Solos", "Solos Bacia Rio Paraíba do
Sul"</element>
            </appinfo>
        </annotation>
    </element>
    <!--============================================================ -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ============================================================
     complex types
     ============================================================ -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
```

```
        </complexContent>
    </complexType>
    <!--============================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember"/>
                    <element name="Shape" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Tipo do shape, define a sua semiologia"</element>
                                <element name="WISE=Ontology.Name">"Shape", "Tipo shape"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Area" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Área da região"</element>
                                <element name="WISE=Ontology.Name">"Area", "Área", "Área Região"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Perimeter" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Perímetro da região"</element>
                                <element name="WISE=Ontology.Name">"Perimeter", "Perímetro", "Perímetro
Região"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Solos_" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Número do solo ou código do solo"</element>
                                <element name="WISE=Ontology.Name">"Solos_", "Número", "Código", "Número
Solo"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Tiposolo" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Tipo do solo da Região"</element>
                                <element name="WISE=Ontology.Name">"Tiposolo", "Tipo", "Tipo Solo", "Nome", "Nome
Solo"</element>
                            </appinfo>
                        </annotation>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```

C.2.2.12 Esquema Georreferenciado Aplicação BacParSul_Integ.xsd do Esquema Integrado Georreferenciado

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:gml="http://www.opengis.net/gml"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" elementFormDefault="qualified">
    <!--==============================================================
     includes and imports
     ============================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
```

```
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ========================================================================
     global declarations
     ======================================================================== -->
    <element name="BacParSul_Integ" type="wisesxmls:FeatureCollectionType"
substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element name="WISE=Description">"Integração de sub-bacias da Bacia do Rio Paraíba do
Sul"</element>
                <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro",
"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul"</element>
                <element
name="WISE=FeatureCollection=Conversion.Datum">"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul",
<![CDATA[D_South_American_1969]]></element>
                <element
name="WISE=FeatureCollection=Conversion.Ellipsoid">"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul",
<![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]></element>
                <element
name="WISE=FeatureCollection=Conversion.CoordinateSystem">"C2/BaseData/BaciaRioParaibaSul/DataSets/BacPar
Sul", <![CDATA[GEOGCS["GCS_South_American_1969", DATUM, PRIMEM["Greenwich",0],
UNIT["Degree",0.0174532925199433]]]]></element>
                <element name="WISE=Ontology.Name">"BacParSul_Integ", "Bacia", "Paraíba do Sul", "Sub-bacias",
"Bacias Rio de Janeiro"</element>
            </appinfo>
        </annotation>
    </element>
    <!--========================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ========================================================================
     complex types
     ======================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--========================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember">
                        <annotation>
                            <appinfo>
                                <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro/geometryM
ember", "C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul/geometryMember"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Area" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Área da bacia"</element>
                                <element name="WISE=Ontology.Name">"Area", "Área", "Área Bacia", "Extensão"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m2"</element>
                                <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro/Area",
"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul/Area"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Perimetro" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element name="WISE=Description">"Perímetro da bacia"</element>
                                <element name="WISE=Ontology.Name">"Perimetro", "Perímetro"</element>
                                <element name="WISE=Field=UnitOfMeasure">"m"</element>
```

```
                        <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro/Perimeter",
"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul/Perimeter"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Nome" type="wisesxmls:wiseString">
                <annotation>
                    <appinfo>
                        <element name="WISE=Description">"Nome da bacia"</element>
                        <element name="WISE=Ontology.Name">"Nome", "Bacia", "Nome Bacia"</element>
                        <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro/Nome",
"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul/Nome"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Baciaparsulestriojaneiro_" type="wisesxmls:wiseInt">
                <annotation>
                    <appinfo>
                        <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro/Baciaparsu
lestriojaneiro_"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Baciaparsulestriojaneiro_id" type="wisesxmls:wiseInt">
                <annotation>
                    <appinfo>
                        <element
name="WISE=SchemaRef">"C1/BaseData/BaciaParaibaSulRioJaneiro/DataSets/BaciaParSulEstRioJaneiro/Baciaparsu
lestriojaneiro_id"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Bacparsul_" type="wisesxmls:wiseInt">
                <annotation>
                    <appinfo>
                        <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul/Bacparsul_"</element>
                    </appinfo>
                </annotation>
            </element>
            <element name="Bacparsul_id" type="wisesxmls:wiseInt">
                <annotation>
                    <appinfo>
                        <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/BacParSul/Bacparsul_id"</element>
                    </appinfo>
                </annotation>
            </element>
        </sequence>
    </extension>
</complexContent>
</complexType>
</schema>
```

C.2.2.13 Esquema Georreferenciado Aplicação DistritBacParSul_Integ.xsd do Esquema Integrado Georreferenciado

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <!-- ================================================================
     includes and imports
     ================================================================ -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
```

```
    <!-- ========================================================================
     global declarations
     ======================================================================== -->
    <element name="DistritBacParSul_Integ" type="wisesxmls:FeatureCollectionType"
 substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul"</element>
                <element
 name="WISE=FeatureCollection=Conversion.Datum">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul",
 <![CDATA[D_South_American_1969]]></element>
                <element
 name="WISE=FeatureCollection=Conversion.Ellipsoid">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul",
 <![CDATA[SPHEROID["GRS_1967",6378160,298.247167427]]]></element>
                <element
 name="WISE=FeatureCollection=Conversion.CoordinateSystem">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritB
 acParSul", <![CDATA[GEOGCS["GCS_South_American_1969", DATUM, PRIMEM["Greenwich",0],
 UNIT["Degree",0.0174532925199433]]]]></element>
            </appinfo>
        </annotation>
    </element>
    <!-- ======================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ========================================================================
     complex types
     ======================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!-- ======================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/geometryMember"</element
 >
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Area" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Area"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Perimeter" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Perimeter"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Distritbacparsul_" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Distritbacparsul_"</element
 >
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Distritbacparsul_id" type="wisesxmls:wiseInt">
                        <annotation>
```

```
                        <appinfo>
                            <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Distritbacparsul_id"</elemen
t>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Nome_dist" type="wisesxmls:wiseString">
                    <annotation>
                        <appinfo>
                            <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Nome_dist"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Nome_muni" type="wisesxmls:wiseString">
                    <annotation>
                        <appinfo>
                            <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Nome_muni"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Nome_est" type="wisesxmls:wiseString">
                    <annotation>
                        <appinfo>
                            <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Nome_est"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Pop_urb" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Pop_urb"</element>
                        </appinfo>
                    </annotation>
                </element>
                <element name="Pop_rur" type="wisesxmls:wiseInt">
                    <annotation>
                        <appinfo>
                            <element
name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/DistritBacParSul/Pop_rur"</element>
                        </appinfo>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</schema>
```

C.2.2.14 Esquema Georreferenciado Aplicação SolosBacParSul_Integ.xsd do Esquema Integrado Georreferenciado

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.cos.ufrj.br/bd/XMLSchemaSemantico"
xmlns:wisesxmls="http://www.cos.ufrj.br/bd/XMLSchemaSemantico" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <!-- ========================================================================
     includes and imports
     ======================================================================== -->
    <import namespace="http://www.opengis.net/gml" schemaLocation="feature.xsd"/>
    <import namespace="http://www.opengis.net/gml" schemaLocation="geometryAggregates.xsd"/>
    <include schemaLocation="SXMLS-SimpleDataTypes.xsd"/>
    <!-- ========================================================================
     global declarations
     ======================================================================== -->
```

```
    <element name="SolosBacParSul_Integ" type="wisesxmls:FeatureCollectionType"
 substitutionGroup="gml:_FeatureCollection">
        <annotation>
            <appinfo>
                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/SolosBacParSul"</element>
            </appinfo>
        </annotation>
    </element>
    <!--================================================================== -->
    <element name="Feature" type="wisesxmls:ApplicationFeatureType" substitutionGroup="gml:_Feature"/>
    <!-- ====================================================================
     complex types
     ==================================================================== -->
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType"/>
        </complexContent>
    </complexType>
    <!--================================================================== -->
    <complexType name="ApplicationFeatureType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element ref="gml:geometryMember">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/SolosBacParSul/geometryMember"</element
 >
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Area" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/SolosBacParSul/Area"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Perimeter" type="wisesxmls:wiseFloat">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/SolosBacParSul/Perimeter"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Solos_" type="wisesxmls:wiseInt">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/SolosBacParSul/Solos_"</element>
                            </appinfo>
                        </annotation>
                    </element>
                    <element name="Tiposolo" type="wisesxmls:wiseString">
                        <annotation>
                            <appinfo>
                                <element
 name="WISE=SchemaRef">"C2/BaseData/BaciaRioParaibaSul/DataSets/SolosBacParSul/Tiposolo"</element>
                            </appinfo>
                        </annotation>
                    </element>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```