UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO
2006.2

ESPECIFICAÇÃO DE TESTES FUNCIONAIS PARA O PROCESSADOR DE CONSULTAS DA ARQUITETURA GOLAPA

POR
DIEGO MARTINS VIEIRA BARROS

TRABALHO DE GRADUAÇÃO

Recife

Março de 2007

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO
2006.2

DIEGO MARTINS VIEIRA BARROS

ESPECIFICAÇÃO DE TESTES FUNCIONAIS PARA O PROCESSADOR DE CONSULTAS DA ARQUITETURA GOLAPA

Este trabalho foi apresentado ao programa de graduação em Engenharia da Computação da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Engenheiro da Computação.

Orientadora: Valéria Cesário Times
Co-orientador: Robson do Nascimento Fidalgo

Recife

Março de 2007

Agradecimentos

Primeiramente, gostaria de agradecer a Deus por todos os objetivos que me ajudou a alcançar.

Agradeço aos meus pais Edmilson e Rosa, e às minhas irmãs Bruna e Milena, por constituirmos uma família estruturada, o que sempre foi muito importante para a minha formação acadêmica. Gostaria de agradecer também à minha namorada Luciana, pela enorme paciência, especialmente durante este semestre, e pela ajuda no trabalho, que foram essenciais para que eu conseguisse concluí-lo. Ao meu tio Gildo e a Silvana, por terem me recebido de braços abertos durante o começo do curso.

Gostaria de agradecer aos meus orientadores Valéria Times e Robson Fidalgo, pela ajuda inestimável, seja tirando dúvidas e ou dando sugestões sempre que foi preciso, para que o trabalho fosse desenvolvido de maneira a atingir os objetivos traçados. Aos colegas do Projeto GOLAPA, Vivianne e Joel, pelas discussões produtivas sobre o meu trabalho.

Não poderia deixar de agradecer aos meus colegas de curso Junior, Diogo, Saulo, Turah e Adson, por terem contribuído bastante para que eu chegasse ao final desses longos cinco anos de curso.

Especificação de Testes Funcionais para o Processador de Consultas da Arquitetura GOLAPA

RESUMO

As organizações cada vez mais utilizam ferramentas computacionais para auxiliar na tomada de decisões estratégicas, visando incrementar o desempenho dos seus negócios. Apesar disso, geralmente são usadas ferramentas de computação diferentes para lidar com dados analíticos e geográficos. Várias pesquisas vêm sendo feitas com o objetivo de prover sistemas capazes de processar tanto informações geográficas como multidimensionais, de maneira integrada. Esta classe de sistemas é denominada *Spatial OLAP* (*SOLAP*), pois sintetizam as características presentes nas ferramentas *On-Line Analytical Processing* (*OLAP*), *Data Warehouse* (*DW*) e Sistemas de Informações Geográficas (SIG).

Atualmente, um dos sistemas *SOLAP* existentes na academia foi implementado de acordo com a arquitetura *Geographical Online Analytical Processing Architecture* (*GOLAPA*), a qual é baseada em tecnologias abertas e extensíveis, como *Java* e *XML*. *GOLAPA* foi concebida de modo a considerar alguns aspectos importantes, tais como independência de plataforma, extensibilidade, funcionalidades integradas e uso de um *DW* Geográfico (DWG).

Por outro lado, os sistemas computacionais estão sendo cada vez mais exigidos em relação ao seu nível de qualidade. Sendo assim, a atividade de testes de software é tida como uma etapa fundamental no ciclo de desenvolvimento dos sistemas, já que possibilita o diagnóstico de falhas e pode indicar o grau de qualidade do sistema. Visando contribuir para a melhoria do nível de qualidade da arquitetura *GOLAPA*, este trabalho especifica testes funcionais (caixa-preta) para o engenho de consultas *SOLAP* desta arquitetura. Ressalta-se que a especificação dos testes foi validada através de sua execução sob uma instância de *GOLAPA*.

Palavras-chave: Suporte à Decisão, *SOLAP*, *GOLAPA*, testes funcionais.

Functional Tests Specification for the GOLAPA Architecture Query Processor

ABSTRACT

The use of computer systems supporting strategic decision making activities is increasing day after day throughout the organizations, in order to achieve better business. However, distinct computer tools are usually used to manipulate analytical and geographical data. Much research has been developed with the aim to provide systems capable of processing both geographical and anaytical information by using an integrated approach. This kind of systems is called Spatial OLAP (SOLAP) as they synthesize the capabilities found in many tools, such as On-Line Analytical Processing (OLAP), Data Warehouse (DW) and Geographical Information Systems (GIS).

Nowadays, one of the SOLAP systems found in the Database literature has been implemented according to the Geographical Online Analytical Processing Architecture (GOLAPA) which is in turn based on open and extensible technlogies, including Java and XML. GOLAPA has been designed to take into account some important issues, such as platform independency, extensibility, integrated functionality and the use of a Geographical DW (GDW).

On the other hand, computer systems have become even more demanded in relation to their quality level. Thus, the software tests activity can be seen as an essential issue in the systems development cicle, because it makes possible the fails diagnosis and may indicate the system quality level. In order to contribute to the increase of GOLAPA's quality level, this work specifies some functional tests (black-box) for the SOLAP query engine of this architecture. We highlight that this tests specification has been validated by executing them together with an instance of GOLAPA.

Keywords: Decision Support, SOLAP, GOLAPA, functional tests.

Sumário

Lista de Figuras

Lista de Quadros

CAPÍTULO 1 - Introdução

Este capítulo apresenta uma contextualização do ambiente em que este trabalho está inserido, destacando os motivos que levaram ao seu desenvolvimento e os seus objetivos. Por fim, é apresentada a estrutura em que os demais capítulos deste trabalho estão organizados.

1.1 Motivação

Considerando que as organizações têm a necessidade de analisar estrategicamente os seus dados e que os sistemas transacionais não são adequados para este tipo análise, uma área da Tecnologia da Informação que tem se destacado é a de Sistemas de Suporte a Decisão (SSD) [SWC+02]. Uma característica destes tipos de sistemas é o uso de bases de dados não convencionais que são consultadas por ferramentas sofisticadas de análise. Dentre estas ferramentas, este trabalho foca o estudo em três delas: *Data Warehouse* (*DW*) [Kim96], *Online Analytical Processing* (*OLAP*) [CD97] e Sistemas de Informações Geográficas (*SIG*) [LGM99]. *DW* e *OLAP* têm como característica principal o processamento multidimensional, enquanto que *SIG* são sistemas específicos para processamento geográfico.

A integração dessas três tecnologias possibilita que os *SIG* examinem geograficamente os dados das ferramentas de *DW* e *OLAP*, e por outro lado, permite que estas cruzem e investiguem os dados geográficos sob diferentes níveis de detalhe [Fid05]. Esta integração deu origem aos sistemas *Spatial OLAP* (*SOLAP*) [RBP+05]. Um dos sistemas que se propõem a realizar essa integração está definido de acordo com a arquitetura GOLAPA (*Geographical On-Line Analytical Processing Architecture*) [FTS01] [Fid05], a qual será apresentada no Capítulo 2 deste trabalho.

A etapa de testes é considerada fundamental para o processo de engenharia de software, pois visa verificar se os resultados estão de acordo com os padrões estabelecidos, ou seja, tem o objetivo de encontrar os erros dos sistemas. Além disso, a atividade de testes tem se tornado cada vez mais um pré-requisito para que um sistema alcance padrões de qualidade. Neste sentido, como não existe nenhum trabalho que defina uma sistemática para testar as funcionalidades da arquitetura GOLAPA, isto motiva a realização de

pesquisas com o objetivo de prover uma especificação de testes para o engenho de consultas SOLAP desta arquitetura.

1.2 Objetivos

O objetivo principal deste trabalho é a especificação de testes funcionais para o processador de consultas multidimensionais e geográficas da arquitetura GOLAPA. Trata-se do engenho SOLAP desta arquitetura, o qual é denominado GOLAPE (*Geographical On-Line Analytical Processing Engine*). Visando atingir este objetivo, primeiramente, se faz necessário identificar e detalhar as funcionalidades desempenhadas por este componente e a comunicação do mesmo com os outros componentes da arquitetura. Em seguida, é realizado um estudo sobre as técnicas de especificação de testes de software, especialmente sobre os testes funcionais, para que seja definida a abordagem mais apropriada para testar este tipo de aplicação.

Após a etapa de especificação dos testes, é necessário que eles sejam executados em uma instância da aplicação GOLAPA, de forma que seja validada a especificação dos mesmos. Ressalta-se que os testes das funcionalidades realizadas pelos outros componentes da arquitetura estão fora do escopo deste trabalho.

Por fim, os resultados obtidos com a execução dos testes são analisados, para que sejam identificados os possíveis problemas na aplicação, e assim, para contribuir para a melhoria da qualidade do sistema testado.

1.3 Organização

Visando atingir os objetivos citados na seção 1.2, bem como apresentar alguns conceitos e tecnologias relacionados a este trabalho, os capítulos restantes deste documento estão organizados da seguinte maneira: o capítulo 2 descreve a arquitetura GOLAPA e as tecnologias envolvidas; o capítulo 3 discute os conceitos e as técnicas de testes de software; o capítulo 4 contém a especificação dos testes funcionais para o componente GOLAPE e os resultados obtidos com a execução dos testes, e por fim, o capítulo 5 consiste nas conclusões e propostas de trabalhos futuros.

CAPÍTULO 2 - A Arquitetura GOLAPA

O objetivo deste capítulo é descrever a arquitetura GOLAPA (*Geographical On-Line Analytical Processing Architecture*) [FTS01] [Fid05], que é uma proposta de integração das tecnologias DW, OLAP e SIG. Além dessas tecnologias, são discutidos todos os componentes da arquitetura e suas funcionalidades.

2.1 Introdução

Existem várias propostas na literatura [SLT+99] [SLT+01] [HKS97] [FCT01] [Fer02] voltadas para o fornecimento de um ambiente para a integração de serviços analíticos e geográficos. No entanto, tais propostas não abrangem completamente os aspectos desse tipo de processamento ou não são baseadas em padrões abertos [Sil04] [Fid05]. A arquitetura GOLAPA foi projetada com os objetivos de: (1) ser baseada em tecnologias abertas e extensíveis, como *Java* e *XML*, e (2) prover suporte à decisão em um ambiente de integração entre os processamentos multidimensional e geográfico.

A opção pelas tecnologias *Java* e *XML* foi feita para alcançar o primeiro objetivo, por serem amplamente difundidas e extensíveis. Com relação ao segundo objetivo, foi feita a opção de aplicar uma arquitetura baseada em um *Data Warehouse* Geográfico, por se tratar de uma abordagem apropriada para sistemas de suporte à decisão espacial.

2.2 Tecnologias Envolvidas

Esta seção descreve as tecnologias de suporte à decisão integradas na arquitetura GOLAPA.

2.2.1 Data Warehouse

Data Warehouse (DW) é uma base de dados modelada especialmente para permitir o acesso eficiente aos dados por ferramentas de consulta OLAP, já que possuem as seguintes características [Inm97]:

a. Orientado ao assunto: ênfase no armazenamento dos dados relativos aos fatos e não nos dados relativos às transações que deram origem aos fatos;
b. Devidamente integrado: reúne dados de diversas origens;
c. Variante no tempo: manutenção do histórico dos dados;

d. Não-volátil: apenas cargas e consultas devem ser realizadas, evitando atualizações dos dados.

Dada a complexidade de se implementar um DW para todas as áreas de uma corporação, uma alternativa é implementar um DW específico por área. Este DW é denominado *Data Mart* [Tii05] [Fir97] e a diferença entre este e um DW é apenas o escopo. Desta forma, os *Data Mart* são considerados *Data Warehouse* setoriais, e conseqüentemente, levam menos tempo para serem implementados, além de consumirem menos recursos.

A modelagem de dados de um DW geralmente é feita de duas maneiras [KRR+98] [IIB96] [Fid05]. A mais comum é conhecida como esquema estrela, enquanto que a outra é conhecida como esquema flocos de neve. O esquema estrela (Figura 1) contém uma ou mais tabelas de fatos (no centro do modelo), e várias tabelas de dimensões (na periferia do modelo). A tabela de fatos possui todas as chaves primárias das tabelas de dimensões e é normalizada. Por outro lado, as tabelas de dimensões possuem as descrições textuais do negócio e geralmente não são normalizadas.

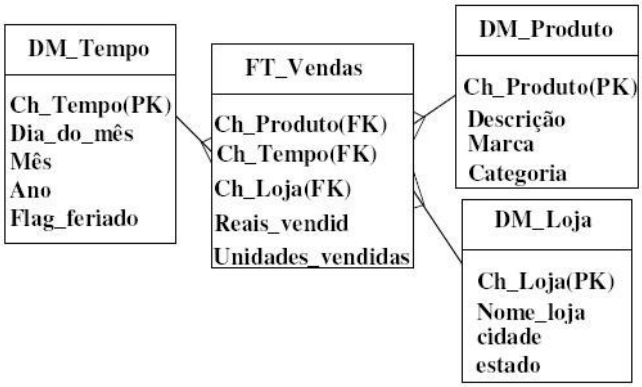

Figura 1. Exemplo do Esquema Estrela [Fid05].

A diferença entre o esquema estrela e o esquema flocos de neve é que este possui todas as dimensões normalizadas. Sua economia de armazenamento é insignificante, e portanto, seu uso é desaconselhado [Kim96]. Além disso, seu desempenho vai ser sempre

inferior ao modelo estrela, já que exige uma quantidade maior de junções para o processamento de consultas.

2.2.2 OLAP

Online Analytical Processing (OLAP) é uma tecnologia responsável pelo processamento multidimensional de dados. Isto é, possibilita realizar consultas que fazem cruzamentos sobre dados agregados em diferentes níveis de detalhe que são extraídos do DW. Existem três tipos de arquitetura em que as ferramentas OLAP disponíveis no mercado são implementadas: relacional (ROLAP), multidimensional (MOLAP) e híbrido (HOLAP). Na arquitetura relacional, a agregação e a materialização dos dados é realizada sobre uma estrutura física relacional. Já na arquitetura multidimensional, as mesmas funções da relacional são desempenhadas usando tecnologia de bancos de dados multidimensionais sobre uma estrutura física com matrizes de *n* dimensões. E por fim, a arquitetura híbrida integra as duas anteriores, deixando os dados atômicos nas estruturas relacionais e os dados agregados são replicados e materializados nas estruturas multidimensionais.

Apesar de existirem os diferentes tipos de arquitetura mencionados anteriormente, a forma como as ferramentas OLAP manipulam os dados de um DW independe deles. Os dados que são extraídos do DW são agregados e materializados em estruturas lógicas multidimensionais denominadas cubos de dados, as quais podem representar completamente ou parcialmente o DW. A estrutura dimensional de um cubo é criada com base na estrutura do esquema estrela, sendo assim definida em termos de dimensões e medidas. Cada dimensão é composta por elementos denominados membros, os quais são organizados, agregados e materializados em níveis hierárquicos. Um exemplo de cubo de dados é exibido na Figura 2.

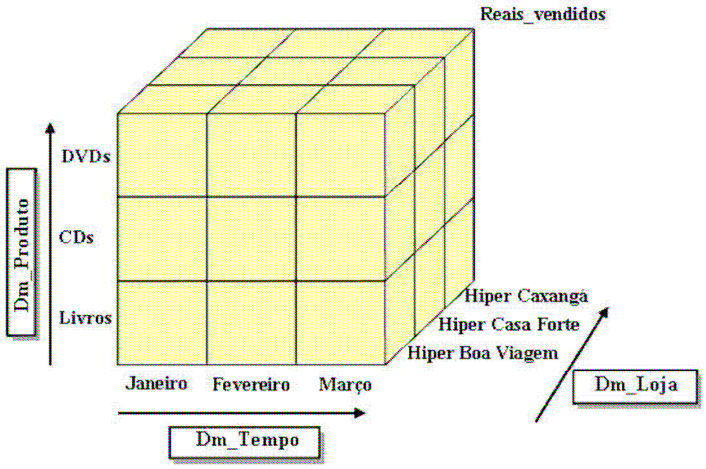

Figura 2. Exemplo de um Cubo de Dados [Med06].

Os operadores OLAP mais usuais para manipulação da estrutura multidimensional de um cubo são [Dat00]:

- **Navegação ao longo das hierarquias conceituais:**
  a. *roll-up*: agregação ou generalização dos dados;
  b. *drill-down*: desagregação ou especialização dos dados.
- **Navegação ao longo do reticulado de cubóides:**
  a. *slice*: extração de um sub-cubo das células verificando restrições em uma dimensão;
  b. *dice*: extração de um sub-cubo das células verificando restrições em várias dimensões.
- **Visualização dos resultados:**
  a. *pivoting* ou *rotate*: rotação dos eixos do cubo, visando visualizar os resultados de consultas.

A Figura 3 ilustra graficamente o resultado da aplicação dos operadores OLAP citados.

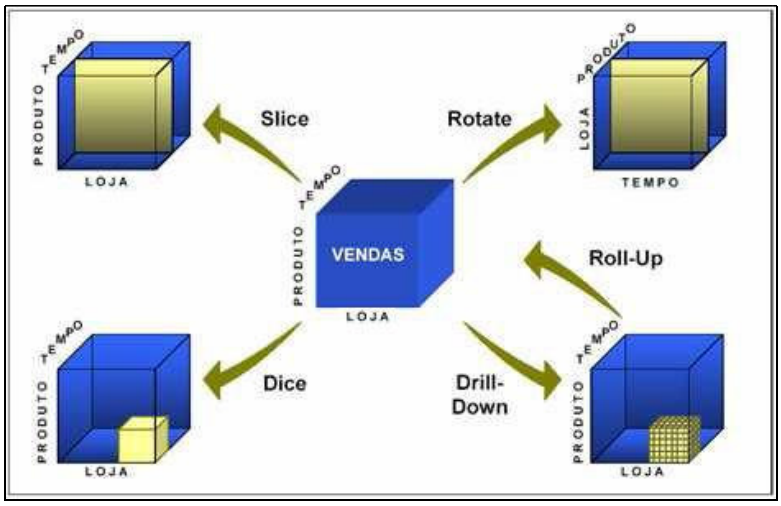

Figura 3. Operadores OLAP [Pai03].

2.2.3 Sistemas de Informações Geográficas

Os Sistemas de Informações Geográficas (SIG) [LGM99] [Dem00] são ferramentas usadas para suporte à decisão em ambientes espaciais. Entre as funcionalidades dos SIG estão: modelagem, captura, armazenamento, análise e apresentação de dados geo-referenciados. Estes tipos de dados são interpretados como feições geográficas, ou seja, abstrações de fenômenos reais, que devem possuir um domínio temporal e uma localização no globo terrestre. Desta forma, a localização das feições geográficas de um SIG é representada através de um sistema de coordenadas geográficas ou terrestres.

Em geral, o comportamento das feições geográficas é analisado visualmente com o auxílio de mapas. Mapa é a denominação dada à representação de uma seleção de características abstratas da superfície terrestre, em uma dada escala e sobre uma superfície plana. Um recurso que os SIG utilizam para visualizar diferentes tipos de informações geográficas é a de sobreposição de temas ou camadas. Com este recurso, cada espaço geográfico pode ter diversos temas de dados geo-referenciados, sendo um para cada tipo de informação espacial a ser representada. Esta abordagem torna mais simples a análise dos dados geo-referenciados, já que possibilita a combinação de dois ou mais temas para o processamento de uma determinada consulta. A Figura 4 apresenta um exemplo desta técnica.

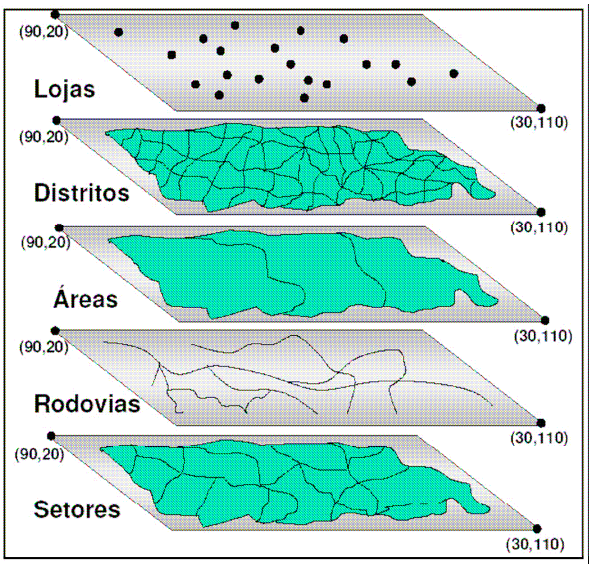

Figura 4. Exemplo de Sobreposição de Camadas [Fid05].

Existem dois modelos complementares, sobre os quais as feições geográficas dos sistemas geográficos são modeladas, a saber: campo e objeto. O modelo de campos é recomendado quando é necessário identificar os elementos ou fenômenos geográficos que não possuem uma limitação geográfica bem definida, sendo portanto, melhor especificados através da distribuição espacial de uma variável geográfica. Por sua vez, o modelo de objetos é recomendado para representar uma região geográfica formada por um conjunto de objetos possíveis de identificar e localizar, onde cada objeto possui suas próprias características e uma geometria associada que define seus limites no espaço.

A modelagem de campos ou objetos pode ser realizada de duas maneiras [CCH+96]: formato matricial (*raster*) ou formato vetorial (*vector*). Desta forma, os objetos do modelo de campos são geralmente representados na forma matricial, sendo os seus relacionamentos topológicos determinados a partir da vizinhança dos componentes da matriz, e suas coordenadas geográficas obtidas de acordo com a posição da célula na matriz. Nesta representação, podem ser realizadas operações como a sobreposição de matrizes, utilizando artifícios matemáticos para combinar os valores das células, ou a transformação de conjuntos de várias células adjacentes em uma única célula, cujo valor é calculado de acordo com os valores das células incluídas na transformação [CCH+96]. Já os objetos geográficos são representados na forma vetorial, com o uso de pontos, linhas e polígonos para descrever suas geometrias. Linhas são compostas por conjuntos de pontos, e

polígonos (abertos ou fechados), por conjuntos de linhas. Nesta representação, podem ser realizadas operações topológicas (*e.g., adjacência e inclusão*) e métricas (*e.g. distância e área*) [CCH+96]. A Figura 5 ilustra a matriz de interseções descrita em [EH91] com as principais operações topológicas. Abaixo da representação gráfica de cada operação topológica é exibida a matriz de interseção associada, obtida através de formalismos matemáticos [EH91].

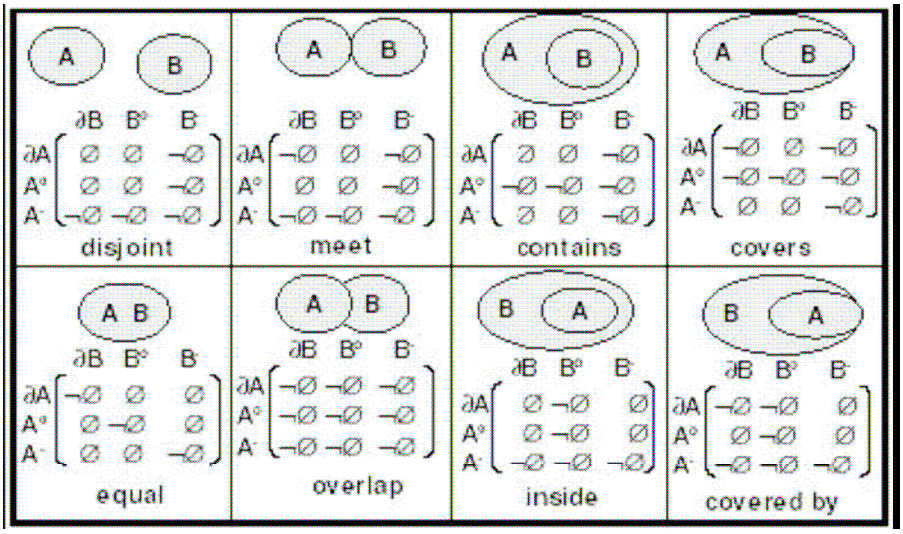

Figura 5. Matriz de Interseções [EH91].

2.3 Componentes de GOLAPA

Os componentes da arquitetura GOLAPA estão distribuídos em cinco camadas, conforme exibido na Figura 6. As camadas (**I**), (**II**) e (**III**) são essenciais para o suporte multidimensional-geográfico e tratam dos dados, serviços e interface gráfica, respectivamente. Existem ainda as camadas (A) e (B), que tratam dos dados operativos e da construção/manutenção do DWG. Por GOLAPA estar estruturada em camadas, cada uma destas camadas provê serviço para a camada imediatamente inferior e faz uso dos serviços providos pela camada imediatamente inferior. Além disso, GOLAPA foi projetada de maneira modularizada, visando facilitar a manutenção e o reuso de software [Men02].

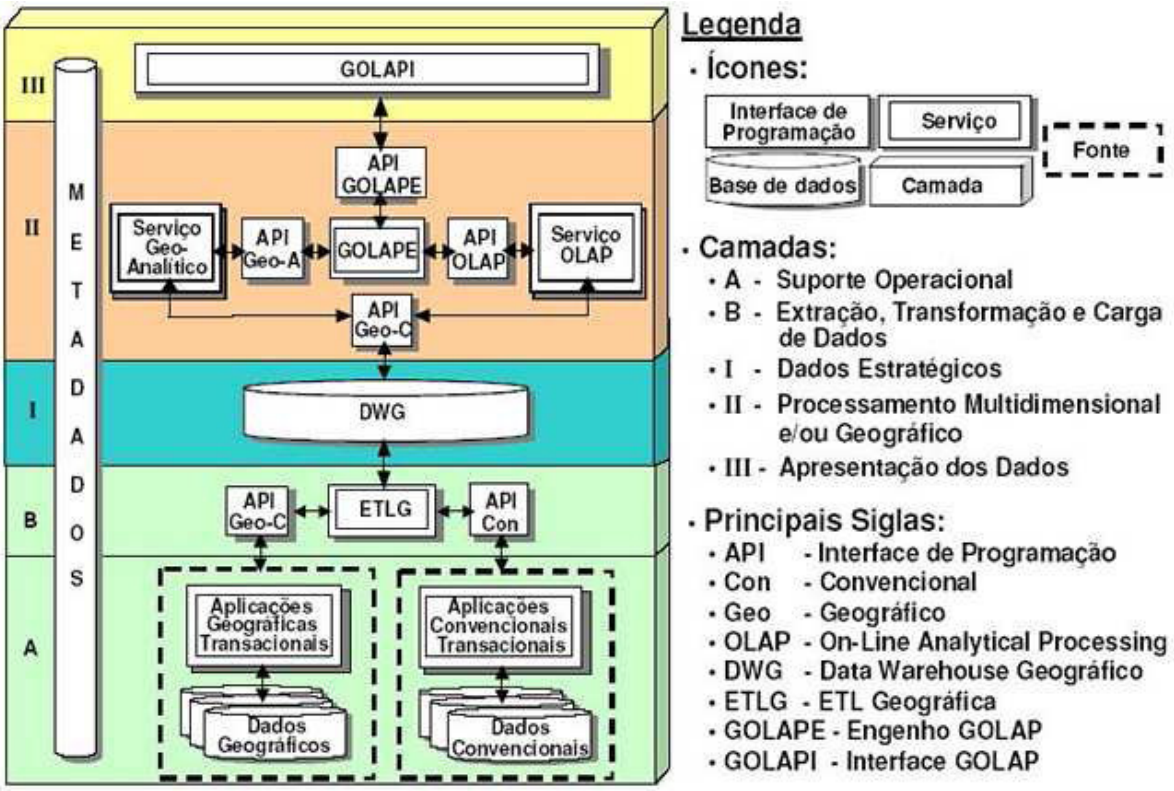

Figura 6. Arquitetura GOLAPA [Fid05].

A seguir são descritas as camadas da arquitetura GOLAPA:

Camada (A) – Dados para suporte operacional: Corresponde às fontes de dados transacionais, já que tanto o SIG como outros componentes desta camada foram projetados para processamento transacional.

Camada (B) – Acesso, extração, transformação, validação e carga dos dados: Responsável pela conversão dos dados do ambiente operativo para o de decisão.

Camada (I) – Dados para suporte à decisão: Separa o ambiente transacional do ambiente de suporte à decisão. Esta camada garante que nenhuma operação ocorrida no ambiente transacional seja refletida no ambiente de suporte à decisão, a não ser pela ocorrência de uma recarga no DWG.

Camada (II) – Suporte à decisão multidimensional-geográfico: Responsável pela manipulação das funcionalidades referentes aos processamentos analítico, geográfico e analítico-geográfico. O componente que executa esse processamento é o GOLAPE, que tem funcionalidades de processamento tanto de dados analíticos como de dados geográficos.

Camada (III) – Apresentação dos dados: Responsável pela visualização dos dados. Pode ser tanto uma interface gráfica local, quanto um cliente remoto, ou uma interface para a *web*.

A seguir são descritos os principais componentes presentes nas camadas da arquitetura GOLAPA:

ETLG (*Extraction, Load and Transformation* Geográfica): Componente que representa uma ferramenta ETL [Kim96] (extração, transformação e leitura) tradicional com suporte a dados geográficos.

DWG (*Data Warehouse* Geográfico): Componente que representa a base de dados da arquitetura GOLAPA. Esta base deve ser modelada de acordo com o esquema estrela, estendendo-o para permitir a inserção de propriedades geográficas, as quais são definidas em uma dimensão geográfica [Sil04]. Esta modelagem deve seguir os padrões definidos no arcabouço GeoDWFrame (*Geographical Data Warehouse Framework*) [Fid05] o qual tem o objetivo de definir um conjunto de conceitos e princípios de projeto para a construção de um DWG.

Metadados: Componente que representa o repositório de metadados, que contém as informações necessárias para a integração entre os serviços analítico e geográfico. Esta base armazena informações sobre as correspondências/relações existentes entre os dados analíticos e geográficos, o que se torna possível com a implementação dos metamodelos GAM (*Geographical Analytical Metamodel*) e GeoMDM (*Geographical Multidimensional Metamodel*) [Fid05].

Serviço Geo-Analítico: Componente responsável pelo processamento dos dados geográficos para o suporte à decisão não-transacional, sendo baseado em sistemas que implementam o conceito de serviços de informações geográficas analíticas (*Analytical Geographical Information Service – AGIS*) [FTS+04].

Serviço OLAP: Componente responsável pelo processamento dos dados analíticos, que é equivalente ao serviço tradicional OLAP. A integração deste componente com o componente *Serviço Geo-Analítico* permite que o componente GOLAPE realize o processamento analítico-geográfico.

GOLAPE (*Geographical On-Line Analytical Processing Engine*): Componente que representa o engenho de consultas multidimensionais e/ou geográficas, que pode ser considerado o principal componente utilizado para prover a integração entre o processamento analítico e o geográfico. Este componente, através de sua interface de programação (componente *API GOLAPE*), recebe e responde uma consulta enviada pela interface gráfica (componente *GOLAPI*). As consultas podem ser de três tipos: (1) multidimensional – MD (*e.g. total de vendas por loja e distribuidor do produto nos trimestres de 2006*), (2) geográfica - GEO (*e.g. lojas localizadas dentro de uma área ad-hoc*) ou (3) multidimensional e geográfica - GEOMD. Neste último tipo, a consulta pode ser de mapeamento (*e.g. total de vendas por endereço da loja e distribuidor do produto nos trimestres de 2006*) ou de integração (*e.g. total de vendas por endereço da loja e distribuidor do produto que foram realizadas nos trimestres de 2006 por lojas que são localizadas dentro de uma área ad-hoc*). Na consulta de mapeamento não há operação espacial a ser realizada, mas o resultado é visualizado em mapas e tabelas. Já na consulta de integração, também é dado o suporte para operações espaciais. Para as consultas multidimensionais ou geográficas, este componente as encaminha para serem executadas pelo *serviço OLAP* ou *Geo-Analítico*, respectivamente. Se a consulta for de mapeamento, GOLAPE a encaminha para ser executada pelo serviço OLAP, consulta os metadados identificando quais destes possuem correspondências

geográficas e gera o resultado final, que é exportado para *GOLAPI* através da *API GOLAPE*. Por fim, se a consulta for de integração, duas estratégias podem ser usadas: (1) encaminhar primeiro a consulta multidimensional para o *serviço OLAP*, ou (2) encaminhar primeiro a consulta geográfica para o *serviço Geo-Analítico*. Em ambos os casos, os metadados são consultados para realizar a *equi-junção* entre os dados analíticos e geográficos, e possibilitar a geração do resultado final, que é exportado para a interface gráfica. Uma implementação de uma instância do componente GOLAPE está descrita em [Sil04] [STF+04] [STF+05] [Med06]. Ressalta-se que o objetivo principal deste trabalho é especificar testes funcionais para este componente.

API GOLAPE: Componente que representa a interface de programação para o componente *GOLAPE*. Possibilita que o componente GOLAPI envie e receba dados e metadados multidimensionais e/ou geográficos. Um esquema XML para requisição dos dados e metadados foi implementado e encontra-se descrito em [Sil04] [STF+04] [STF+05]. Além disso, um metaesquema XML para integração e exportação desses dados está descrito em [Fid05] e é denominado GMLA (*Geography Markup Language for Analysis*). Da forma como foi definido, GMLA importa, estende e integra os esquemas GML [GML07] e XMLA [XML07], permitindo que dados que eram descritos por dois esquemas distintos passem a ser descritos e integrados semanticamente por um único esquema.

GOLAPI (*Geographical On-Line Analytical Processing Interface*): Componente que representa a interface gráfica para realizar os diferentes tipos de consultas disponíveis em GOLAPE e para permitir a visualização dos seus resultados. Um protótipo deste componente encontra-se descrito em [Sil04] [STF+04] [STF+05].

API Con, Geo-C, Geo-A e OLAP: Componentes que representam as interfaces de programação utilizadas para acessar os serviços convencional, geo-convencional, geo-analítico e multidimensional, respectivamente.

2.4 Considerações Finais

Este capítulo apresentou a arquitetura GOLAPA, os motivos que impulsionaram a sua criação, juntamente com a descrição de suas camadas e de seus principais componentes. A importância de GOLAPA para o presente trabalho se deve ao fato do foco desta pesquisa consistir em especificar testes funcionais para o seu componente GOLAPE. Com relação a este componente, foram descritos os diferentes tipos de consultas que ele é capaz de processar.

No próximo capítulo, são listados os conceitos relacionados ao teste de software, bem como algumas técnicas para realização de testes, com ênfase para a metodologia que é usada para especificar os testes para o componente GOLAPE da arquitetura GOLAPA.

CAPÍTULO 3 - Teste de Software

Este capítulo contém uma visão geral da área de teste de software, suas diferentes técnicas e critérios, mecanismos de especificação de testes e ferramentas para automatizar este processo. Além disso, como se objetiva obter a especificação de testes que permitam a análise das funcionalidades presentes na arquitetura GOLAPA, conceitos relativos aos testes funcionais e áreas afins são aprofundados. Estão listados também neste capítulo alguns trabalhos encontrados na literatura, que de alguma forma estão relacionados a este.

3.1 Visão Geral

Teste de software é uma etapa fundamental do processo de desenvolvimento, sendo assim uma importante premissa para alcançar padrões de qualidade no produto criado. O principal objetivo dos testes é revelar a presença de erros no programa, através de uma execução controlada, verificando se o comportamento do programa está de acordo com o especificado, e mostrando se os resultados estão ou não de acordo com os padrões estabelecidos [RSM+05].

A atividade de testes é dividida em quatro etapas [RSM+05] que geralmente devem ser executadas durante o processo de desenvolvimento de software. São elas:

a. Planejamento de testes;
b. Projeto de casos de teste;
c. Execução de testes e coleta dos resultados;
d. Avaliação dos resultados.

Normalmente, as técnicas de teste de software são classificadas em dois tipos: a funcional (caixa-preta) [BG04] [RSM+05] e a estrutural (caixa-branca) [BMV+00]. Cada uma delas oferece uma perspectiva diferente e aborda uma classe de erro diferente. Portanto, estas devem ser usadas de maneira complementar. As técnicas existentes diferenciam-se pela origem das informações utilizadas para estabelecer os requisitos e critérios de teste [Cop04]. Desta forma, a técnica estrutural necessita da estrutura do software para derivar os requisitos de teste, enquanto que a técnica funcional utiliza apenas a especificação do software [Mod06] [RSM+05].

As técnicas de aplicação de testes podem ser executadas em quatro níveis diferentes, os quais são brevemente descritos a seguir [Cop04]:

1. **Testes unitários**: trata-se de testes realizados na menor unidade do software, como, por exemplo, um método na linguagem *Java* ou uma função na linguagem *C*;
2. **Testes de integração**: duas unidades do programa podem funcionar corretamente de maneira isolada e podem não funcionar da maneira esperada quando uma das unidades faz uso de algum recurso presente na outra. Este nível de testes é realizado para verificar se as unidades estão integradas de maneira correta formando os subsistemas, e estes formando o sistema;
3. **Testes de sistema**: este nível de testes tem o foco voltado para defeitos que possam se originar no último nível de integração, ou seja, no nível de sistema. Inclui desde testes de funcionalidade até testes de usabilidade e de segurança;
4. **Testes de aceitação**: são realizados diretamente com o cliente, para garantir a aceitação do software por ele. Do ponto de vista do cliente, ele gostaria de realizar o maior número de testes de aceitação possíveis, enquanto que do ponto de vista da empresa que vende o software, esta visa minimizar a quantidade de testes de aceitação, para acelerar a venda do produto.

As metodologias de desenvolvimento de software têm uma disciplina dedicada aos testes. No entanto, em muitos casos, a especificação e a implementação dos testes é feita sem um planejamento adequado. Além disso, os testes podem ser selecionados aleatoriamente e especificados ou implementados de forma não estruturada e não sistemática. Essa realidade acontece no desenvolvimento de muitos softwares comerciais e é decorrente de recursos limitados e tempo escasso para os testes [BMV+00]. A seguir, são listados alguns problemas encontrados na atividade de testes [Sil01] [BG04]:

a. **Falta de planejamento do tempo e do custo**: geralmente, o planejamento dos testes é feito tardiamente no ciclo de desenvolvimento do software, ou seja, quando ele está próximo do fim. Desta forma, caso a execução do projeto esteja atrasada,

então normalmente, os testes são realizados de maneira superficial, comprometendo a qualidade do software;

b. **Falta de documentação**: os testes não são devidamente preparados, não havendo um plano de testes e estes não são documentados;
c. **O teste é a última etapa do processo de desenvolvimento**: o desenvolvimento dos testes é feito apenas quando o programa está praticamente pronto. No entanto, a fase de testes deveria começar imediatamente após a especificação ter sido elaborada. Isto permitiria que muitos erros, omissões e inconsistências fossem encontrados nas fases de análise e projeto do ciclo de desenvolvimento;
d. **Não existe sistemática na geração dos testes**: a escolha dos testes é feita de maneira aleatória e eles geralmente são gerados com base no conhecimento do testador e sem um procedimento definido.

A execução de testes não é suficiente para provar a corretude de um programa. Isto só pode ser feito através de provas formais. Apesar disso, se os testes forem planejados e executados de maneira criteriosa e sistemática, eles podem contribuir para o aumento da confiança de que o software realiza as funções especificadas, incrementando assim, seu nível de qualidade [Mod06].

Com a crescente complexidade dos programas de computador, torna-se fundamental a utilização de ferramentas de teste, visando a automatização desta atividade. O teste de software é, em geral, custoso e propenso a erros, tornando essencial a existência de ferramentas que apóiem essa atividade [RSM+05]. Atualmente encontram-se disponíveis diversas ferramentas que buscam atender a esse propósito. Entre estas, pode-se citar: *SPACES* [BAM+04], *JUnit* [JUN07] e *IBM Rational Robot* [ROB07]. Porém, a maioria desse tipo de ferramenta oferece apenas suporte aos testes estruturais. Aquelas que implementam a técnica funcional desconsideram seus principais critérios e não permitem análise da cobertura necessária [RSM+05].

3.2 Especificação de Testes

A especificação de testes pode ser feita através do uso de casos de teste. O termo *caso de teste* [Mod06] diz respeito ao conjunto de entradas de dados do programa,

juntamente com o conjunto de saídas esperadas e o de saídas obtidas após a execução do programa. A qualidade de um caso de teste está diretamente relacionada com a probabilidade de ele revelar um erro no software, ou seja, quanto maior a chance de um caso de teste encontrar um problema no programa, melhor ele é considerado.

O uso da *Unified Modeling Language* (UML) é comum na modelagem de sistemas. Desta forma, alguns trabalhos [BG04] [Mod06] da área de testes já propuseram o uso desta linguagem inclusive para modelar os casos de teste. A UML possui vários tipos de diagramas, por exemplo: casos de uso, classes, seqüência e estados. O diagrama de casos de uso é o que mais se adequa à especificação de testes funcionais, porque possibilita a captura das funcionalidades do sistema, na visão do usuário [Roc05]. Esta adequação se deve ao fato de que os casos de uso estão diretamente ligados aos requisitos do sistema, e os testes funcionais visam justamente identificar erros de funções que não estejam de acordo com os requisitos. Já para os testes estruturais, o diagrama que mais se adequa à especificação é o diagrama de estados, porque possibilita a modelagem do passo a passo contido no código-fonte do programa [BMV+00].

A especificação dos testes, sendo feita com qualidade, gera um conjunto de casos de teste que abrangem o maior número de possíveis entradas [BMV+00], ou seja, objetiva-se a utilização de casos de teste que tenham alta probabilidade de encontrar a maioria dos erros, minimizando tempo e esforço. Além disso, a própria atividade de especificação de testes é capaz de encontrar defeitos no programa [Mod06]. Antes de descrever os testes funcionais com um nível maior de detalhe, são descritos os conceitos de *descrição de caso de uso* e *cenários de caso de uso*, úteis para a especificação dos testes.

3.2.1 Especificação de Casos de Uso

A especificação de casos de uso é uma atividade da engenharia de software que constitui uma etapa do ciclo de desenvolvimento de software, assim como a atividade de testes. Uma das técnicas de testes listadas neste trabalho faz uso explícito dos conceitos de *descrição de caso de uso* e *cenários de caso de uso*, e por isso, estes conceitos são brevemente descritos nas seções 3.2.1.1 e 3.2.1.2.

3.2.1.1 Descrição de Caso de Uso

A descrição de um caso de uso deve conter informações como o seu nome, uma breve descrição sobre os seus objetivos, uma descrição textual dos fluxos de eventos que podem acontecer durante a execução do caso de uso, uma descrição das restrições que precisam ser satisfeitas para que o caso de uso tenha início, e uma descrição das restrições satisfeitas quando a execução do caso de uso é finalizada [Som04] [Bra04] [Coc01].

Dentre as informações descritas acima, a parte mais importante para a geração de casos de teste é a dos fluxos de eventos. A descrição de um caso de uso é composta por um fluxo principal e por alguns fluxos alternativos. O fluxo principal descreve as ações que acontecem em uma execução normal do caso de uso. Já os fluxos alternativos descrevem tanto ações opcionais que podem ocorrer durante a execução do caso de uso, como ações que impliquem na geração de falhas durante a sua execução. Isto faz com que os fluxos alternativos também sejam denominados de fluxos secundários e de fluxos de exceção, dependendo do tipo de ação que ele descreva.

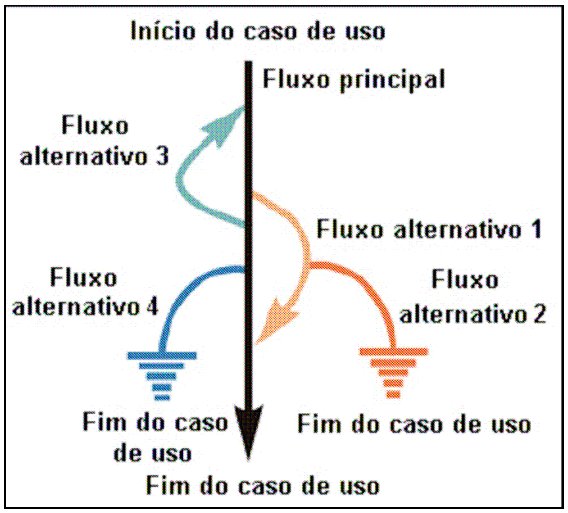

Figura 7. Fluxos de eventos de um caso de uso [Heu01]

A Figura 7 ilustra uma possível combinação de fluxos de eventos de um caso de uso. Note que os fluxos alternativos 1 e 3 são fluxos secundários, porque mudam a execução normal do caso de uso, enquanto que os fluxos alternativos 2 e 4 são fluxos de exceção, já que a execução deles faz com que o caso de uso seja encerrado de maneira indesejada.

3.2.1.2 Cenários de Caso de Uso

Um cenário de caso de uso pode ser descrito como uma instância do caso de uso, ou seja, um caminho completo pelo caso de uso, com início e fim. A execução do caso de uso pode acontecer por vários caminhos. Cada caminho possível corresponde a um cenário. Visando agrupar as informações e facilitar o entendimento, os cenários são inseridos em um único quadro, como ilustra o Quadro 1, referente ao caso de uso da Figura 7.

Quadro 1. Cenários para o caso de uso da Figura 7 [Heu01].

Cenário	Fluxo Inicial	Alterações		
Cenário 1	Fluxo principal			
Cenário 2	Fluxo principal	Fluxo alternativo 1		
Cenário 3	Fluxo principal	Fluxo alternativo 1	Fluxo alternativo 2	
Cenário 4	Fluxo principal	Fluxo alternativo 3		
Cenário 5	Fluxo principal	Fluxo alternativo 3	Fluxo alternativo 1	
Cenário 6	Fluxo principal	Fluxo alternativo 3	Fluxo alternativo 1	Fluxo alternativo 2
Cenário 7	Fluxo principal	Fluxo alternativo 4		
Cenário 8	Fluxo principal	Fluxo alternativo 3	Fluxo alternativo 4	

3.2.2 Especificação de Testes Funcionais

Na técnica funcional, também denominada caixa-preta, os requisitos de teste são estabelecidos a partir da especificação do software, e sua estrutura não é necessariamente considerada [Mod06]. O objetivo do teste funcional é encontrar discrepâncias entre o comportamento atual do sistema e o descrito durante a especificação. Desta forma, são consideradas apenas as entradas e saídas e o testador não precisa ter acesso ao código-fonte do software [BMV+00].

Em se tratando de testes funcionais, Heumann propôs uma metodologia de geração de casos de teste a partir dos casos de uso [Heu01]. Esta metodologia define um processo composto por três passos: geração de cenários, identificação dos casos de teste e identificação dos valores que serão usados nos casos de teste. Além disso, serão discutidas, também neste capítulo, as técnicas de testes *particionamento de equivalência* e *análise de valor-limite*.

3.2.2.1 Geração de Casos de Teste pelo Método de Heumann

A metodologia proposta por Heumann [Heu01] para geração de casos de teste a partir dos casos de uso é descrita a seguir. Os passos deste processo são baseados em um caso de uso hipotético *Cadastrar Venda*, cujo diagrama UML é exibido na Figura 8.

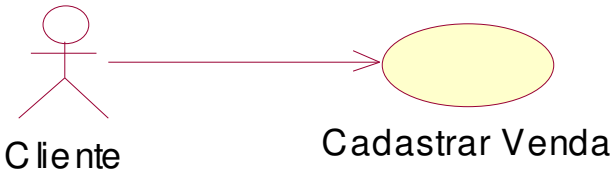

Figura 8. Diagrama UML do caso de uso *Cadastrar Venda*.

No primeiro passo, a partir da leitura da descrição do caso de uso, deve ser criada uma tabela com todos os cenários possíveis para a execução do caso de uso. Por exemplo, o caso de uso *Cadastrar Venda* pode ter as ações de *Validar Crédito Cliente* e *Validar Estoque Produto*. O fluxo principal é representado por uma situação na qual o cliente possui crédito e há estoque para o produto especificado. Pode-se ter dois fluxos alternativos. O fluxo alternativo 1 representa as situações nas quais o cliente não possui crédito e o fluxo alternativo 2, as situações nas quais não há estoque para o produto especificado. Os cenários para este caso de uso estão descritos no Quadro 2.

Quadro 2. Cenários para o caso de uso *Cadastrar Venda*.

Cenário	Fluxo Inicial	Alteração
Cenário 1	Fluxo principal	
Cenário 2	Fluxo principal	Fluxo alternativo 1
Cenário 3	Fluxo principal	Fluxo alternativo 2

No segundo passo, devem ser identificados os casos de teste referentes a cada cenário descrito. Cada cenário deve dar origem a pelo menos um caso de teste. Os casos de teste são agrupados em uma tabela para facilitar a visualização e aumentar a organização do processo. Os casos de teste referentes aos cenários do caso de uso *Cadastrar Venda* estão descritos no Quadro 3. Note que este quadro não contém dados, ou seja, as entradas para os

casos de teste não foram escolhidas. Os valores *V*, *I* e *N/A* significam *válido*, *inválido* e *não se aplica*, respectivamente. Trata-se de uma etapa intermediária, que é importante para mostrar claramente quais condições estão sendo testadas para cada caso de teste.

Quadro 3. Casos de teste para o caso de uso *Cadastrar Venda*.

ID Caso de Teste	Cenário	Crédito Cliente	Estoque Produto	Resultado Esperado
CT01	Cenário 1	V	V	Venda cadastrada
CT02	Cenário 2	I	N/A	Mensagem de erro
CT03	Cenário 3	V	I	Mensagem de erro

No terceiro passo, devem ser identificados os dados que efetivamente serão usados para implementar e executar os testes. Isto deve ser feito substituindo os *V*s e *I*s da tabela gerada no segundo passo deste processo, por entradas para os casos de teste. O Quadro 4 mostra os casos de teste com as suas respectivas entradas. A escolha dessas entradas não faz parte da metodologia de Heumann. Porém, as seções 3.2.2.2 e 3.2.2.3 apresentam uma breve discussão sobre o uso de duas técnicas para guiar este processo de escolha.

Quadro 4. Casos de teste com entradas para o caso de uso *Cadastrar Venda*.

ID Caso de Teste	Cenário	Crédito Cliente	Estoque Produto	Resultado Esperado
CT01	Cenário 1	R$ 50,00	5 unidades	Venda cadastrada
CT02	Cenário 2	R$ 0,00	N/A	Mensagem de erro
CT03	Cenário 3	R$ 30,00	0 unidades	Mensagem de erro

Note que para execução dos casos de teste especificados no Quadro 4, é necessário supor que a compra a ser cadastrada seja referente a 1 (uma) unidade de um produto com valor menor ou igual a R$ 30,00.

Após a finalização das três etapas do processo de geração dos casos de teste, pode-se redesenhar o diagrama de casos de uso original, inserindo os casos de teste a fim de permitir uma visualização gráfica do resultado do processo. O novo diagrama referente ao caso de uso *Cadastrar Venda* pode ser visto na Figura 9, na qual são vistos os casos de teste *Validar Crédito Cliente* e *Validar Estoque Produto*.

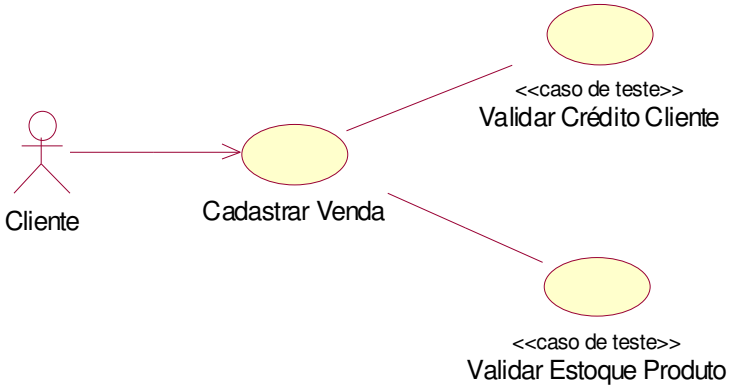

Figura 9. Novo diagrama UML para o caso de uso *Cadastrar Venda* [BG04].

3.2.2.2 Particionamento de Equivalência

Neste critério, divide-se a entrada do programa em classes de dados, dos quais os casos de teste são derivados. Para exemplificar este critério, será utilizado o programa *Identifier* [BMV+00], que consiste na implementação de uma classe Java com um método *verify*, cuja assinatura é mostrada a seguir:

```
public static boolean verify(String)
```

Este método é responsável por verificar se um identificador recebido como parâmetro é válido ou não, em uma linguagem hipotética, retornando verdadeiro ou falso, respectivamente. Um identificador é considerado válido se começar com uma letra e contiver apenas letras ou dígitos. Além disso, deve ter o seu tamanho entre 1 e 6 caracteres.

Analisando estas restrições, é possível observar seis classes de equivalência para o método *verify*, a partir de três condições de entrada, como é mostrado no Quadro 5. Os casos de teste são especificados e implementados com base nestas classes de equivalência, como por exemplo, o seguinte conjunto de quatro casos de teste: *T = {(a1, válido), (2B3, inválido), (Z-12, inválido), (A1b2C3d, inválido)}*. Esses casos de teste exercitam todas as seis classes de equivalência, porque o particionamento de equivalência exige que sejam criados casos de teste para exercitar cada classe inválida e o mínimo de casos de teste para exercitar as classes válidas. Assim, a divisão é feita em classes de equivalência válidas e

inválidas, buscando-se produzir casos de teste que cubram diversas classes de erro e que ao mesmo tempo reduzam o total de casos necessários.

Quadro 5. Classes de equivalência para o método *verify*.

	Classes de Equivalência	
Condição de Entrada	**Válida**	**Inválida**
Tamanho do identificador	$1 \leq tam \leq 6$	$Tam \geq 6$
Começa com uma letra	sim	Não
Contém somente letras ou dígitos	sim	Não

O particionamento do domínio de entrada deve ser feito de forma que para cada classe de equivalência, qualquer um dos seus elementos seja representativo de toda a classe, e dessa forma, o programa em teste se comporte da mesma maneira, independente do elemento escolhido na classe. Por exemplo, o comportamento do programa na execução do método *verify* descrito anteriormente deve ser o mesmo para identificadores de tamanho 3 e 5. Portanto, se um caso de teste de uma determinada classe de equivalência revela um erro, qualquer outro caso de teste nessa classe deve revelar o mesmo erro [RSM+05].

3.2.2.3 Análise de Valor-Limite

A partir da aplicação do particionamento de equivalência, nota-se que grande parte dos erros tende a ocorrer nas fronteiras dos intervalos de entrada de dados [RSM+05]. Este critério complementa o primeiro por meio de casos de teste que exercitam valores-limite, os quais se referem a situações que ocorrem dentro de uma classe de equivalência, diretamente acima ou abaixo dela. Ou seja, enquanto no critério anterior qualquer valor de uma classe de equivalência poderia ser selecionado, neste critério, apenas casos de teste nas extremidades das classes devem ser selecionados.

Por exemplo, o seguinte conjunto de casos de teste exercita os valores-limite das classes de equivalência especificadas no Quadro 5: *T = {(a1b2c3, válido), (A1b2C3d, inválido)}*. Note que os identificadores *a1b2c3* e *A1b2C3d* possuem tamanho 6 e 7, respectivamente. Eles foram escolhidos porque exercitam os valores-limite da classe de equivalência definida no Quadro 5, a qual descreve que apenas identificadores que possuam no máximo 6 caracteres são válidos na linguagem hipotética.

Ainda neste critério, a saída do programa também é particionada e auxilia na derivação dos casos de teste, que por sua vez devem ser escolhidos em cada um dos limites das classes. Ou seja, os casos de teste devem ser projetados de forma que produzam um número máximo (e mínimo) de saídas esperadas, em relação às classes de equivalência.

3.2.2.4 Análise comparativa entre as técnicas baseadas em casos de teste

Com o uso do método de Heumann para geração de testes a partir dos casos de uso, todas as possibilidades de execução de cada caso de uso são cobertas através dos cenários. Além disso, nesta técnica todos os tipos de entradas podem ser usados na execução dos testes. Esses foram os fatores que levaram à escolha deste método para especificar os testes para o engenho de consultas da arquitetura GOLAPA (ver Capítulo 4).

Por outro lado, as técnicas *particionamento de equivalência* e *análise de valor-limite* requerem que classes de equivalência sejam criadas, a partir dos tipos de entradas. A criação das classes de equivalência também cobre todas as possibilidades de execução da funcionalidade, mas tem a desvantagem de não ser tão clara quanto os casos de uso, que descrevem o passo a passo de execução da funcionalidade. Além disso, os tipos de entradas são restritos a valores booleanos, intervalos de valores inteiros ou seqüência de caracteres. Como se deseja clareza e praticidade na especificação dos testes e as entradas para os testes são consultas complexas, estes dois métodos não foram escolhidos para especificação dos testes no Capítulo 4.

3.3 Ferramentas de Testes Funcionais

A utilização de ferramentas de teste desempenha um papel importante na aplicação de critérios de teste. Sem o auxílio de uma ferramenta, a atividade de teste pode se tornar trabalhosa, propensa a erros e limitada a programas simples.

Outro aspecto importante é que a automatização permite a verificação rápida e eficiente das correções de defeitos, agiliza o processo de depuração, permite a captura e análise dos resultados de teste de forma consistente, além de facilitar a execução de testes de regressão, nos quais os casos de teste podem ser reutilizados para revalidação do software após modificação [RSM+05].

Em geral, as ferramentas utilizam o conceito de cobertura para medir o quanto um critério foi satisfeito em relação aos seus requisitos. A análise de cobertura permite

identificar áreas do programa que não são exercitadas por um conjunto de casos de teste, e dessa forma, avaliar a qualidade desse conjunto. Além disso, por meio da análise de cobertura é possível medir o progresso do teste e decidir quando finalizar essa atividade.

Embora a maioria das ferramentas existentes ofereça apoio apenas ao teste estrutural, algumas ferramentas específicas para teste funcional encontram-se disponíveis, como *TestComplete* [TES07], *SPACES* [BAM+04], *Jtest* [JTE07], *JUnit* [JUN07] e *Rational Functional Tester* [FUN07]. A ferramenta *JUnit* é a única gratuita entre as citadas.

3.4 Trabalhos Relacionados

Os seguintes trabalhos encontrados na literatura utilizam o método de Heumann [Heu01] para geração de casos de teste a partir dos casos de uso:

- **TestCen: Ferramenta de Suporte ao Planejamento de Teste Funcional de Software a partir de Diagramas de Caso de Uso [BG04]**: Neste trabalho foi especificada uma ferramenta que possibilita a documentação do projeto de teste, seus casos de teste e os procedimentos para execução de cada caso de teste. Todos os três passos do método de Heumann foram implementados nesta ferramenta.

- **A UML-Based Approach for Testing Web Applications [Nil03]**: Neste trabalho foi estendido o método de Heumann com a criação de um passo extra, no qual os cenários são priorizados. Além disso, este estudo foi direcionado para o teste de aplicações *web*.

- **A UML-based approach to system testing [HVF+05]**: Neste trabalho foram descritas abordagens para automatizar a geração de testes através dos casos de uso e a execução de casos de teste, para sistemas genéricos. O objetivo desta pesquisa é mostrar os benefícios da automatização dos testes e de baseá-los em casos de uso.

No entanto, esses trabalhos especificam testes para outros tipos de aplicações, e não para um engenho de consultas SOLAP. A ausência de um trabalho deste tipo na literatura foi um dos fatores que motivou o desenvolvimento deste trabalho.

3.5 Considerações Finais

Neste capítulo, foram mostrados conceitos relacionados aos testes de software, como por exemplo, as fases do processo de testes, os níveis sob os quais os testes podem ser aplicados e algumas metodologias que podem ser usadas na especificação deles. Uma visão geral da área de testes com características e problemas enfrentados na atividade de testes também fez parte deste capítulo. Foi dado um enfoque maior aos testes funcionais, e à especificação de testes segundo o método de Heumann, pois eles foram usados para especificar os testes para a arquitetura GOLAPA, que é o tema do Capítulo 4. Também foram mostradas neste capítulo, as justificativas para a escolha do método de Heumann, bem como os trabalhos encontrados na literatura que estão relacionados com este trabalho.

Além disso, foram citadas diversas ferramentas de automatização de testes e foram brevemente discutidos os benefícios por elas trazidos. No entanto, não foi possível utilizar a ferramenta de automatização gratuita *JUnit* para a realização dos testes sob uma instância da arquitetura GOLAPA, por motivo de incompatibilidade com aplicações *web*. E portanto, os testes foram realizados manualmente.

CAPÍTULO 4 - Especificação de Testes Funcionais para GOLAPE

Este capítulo descreve a especificação de testes funcionais para o componente GOLAPE da arquitetura GOLAPA. Trata-se do engenho multidimensional-espacial desta arquitetura (ver Capítulo 2).

Para realizar esta especificação, foi escolhido o processo de geração de casos de teste a partir dos casos de uso, descrito por Heumann [Heu01]. No entanto, como não existe nenhum caso de uso especificado para este componente da arquitetura GOLAPA, se fez necessária a especificação dos mesmos, através da análise de suas funcionalidades. Os casos de uso são apresentados na seção 4.1 e constituem uma contribuição importante para o processo de documentação da arquitetura GOLAPA.

O capítulo contém, além de todos os casos de teste especificados para os casos de uso, o esquema do *Data Warehouse* Geográfico utilizado para possibilitar a execução dos testes, bem como as entradas usadas e os resultados obtidos após esta execução.

4.1 Casos de uso para o componente GOLAPE

Foram identificados cinco casos de uso para o componente GOLAPE, com base nas suas funcionalidades (ver Capítulo 2): (1) *Processar Consulta Multidimensional*, (2) *Processar Consulta Geográfica*, (3) *Processar Consulta de Mapeamento*, (4) *Processar Consulta de Integração MD-GEO* e (5) *Processar Consulta de Integração GEO-MD*. Com relação a estas duas últimas funcionalidades, as denominações MD-GEO e GEO-MD indicam a ordem de execução das consultas multidimensional (MD) e geográfica (GEO). As descrições destes casos de uso são encontradas nos quadros de 6, 7, 8, 9 e 10, exibidos a seguir. Além disso, a Figura 10 ilustra o diagrama de casos de uso para o componente GOLAPE.

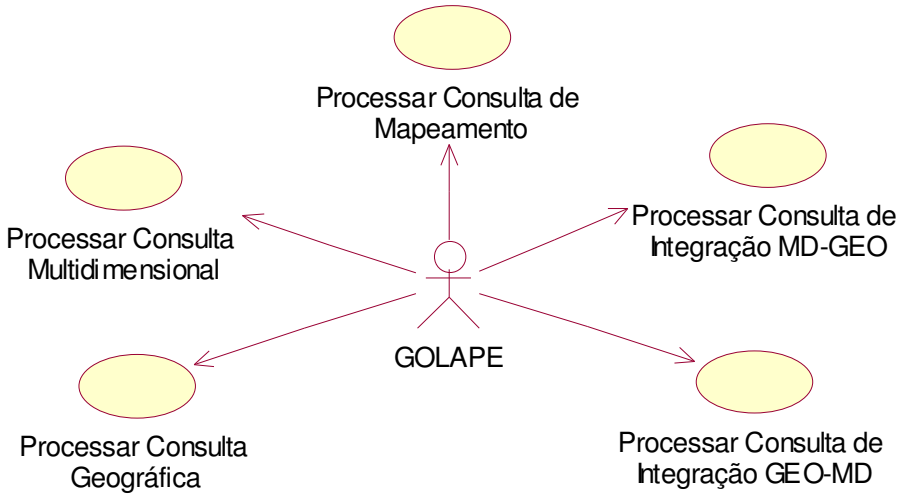

Figura 10. Diagrama de Casos de Uso para o GOLAPE.

Quadro 6. Descrição do caso de uso *Processar Consulta Multidimensional*.

[UC01] Processar Consulta Multimensional	
Ator	GOLAPE
Pré-condições	A consulta multidimensional deve estar com a sintaxe correta.
Fluxo Principal	1. Submeter a consulta multidimensional para processamento pelo servidor OLAP; 2. Montar o resultado final com os registros da consulta multidimensional.
Pós-condições	GOLAPE deve estar apto a enviar o resultado da consulta MD para a GOLAPI.

Quadro 7. Descrição do caso de uso *Processar Consulta Geográfica*.

[UC02] Processar Consulta Geográfica	
Ator	GOLAPE
Pré-condições	A consulta geográfica deve estar com a sintaxe correta.
Fluxo Principal	1. Submeter a consulta geográfica para processamento pelo SGBD com extensão espacial; 2. Montar o resultado final com as geometrias retornadas da consulta geográfica.
Pós-condições	GOLAPE deve estar apto a enviar o resultado da consulta GEO para a GOLAPI.

Quadro 8. Descrição do caso de uso *Processar Consulta de Mapeamento*.

[UC03] Processar Consulta de Mapeamento	
Ator	GOLAPE
Pré-condições	As consultas multidimensional e geográfica devem estar com a sintaxe correta.
Fluxo Principal	1. Consultar os metadados a fim de avaliar se a declaração da consulta multidimensional possui campos que referenciam objetos geográficos; 2. Submeter a consulta multidimensional para processamento pelo servidor OLAP; 3. Consultar os metadados a fim de capturar os *UniqueName* dos registros retornados na consulta multidimensional; 4. Escrever a consulta geográfica de forma que esta retorne apenas os identificadores e as geometrias dos objetos geográficos referentes aos *UniqueName* do passo 2; 5. Submeter a consulta geográfica para processamento pelo SGBD com extensão espacial; 6. Montar o resultado final com os registros da consulta multidimensional e as geometrias da consulta geográfica.
Fluxo Alternativo 1	1. No passo 1 do fluxo principal, caso a declaração da consulta multidimensional não possua campos que referenciem objetos geográficos, uma exceção deve ser lançada; 2. Encerra-se este caso de uso.
Pós-condições	GOLAPE deve estar apto a enviar o resultado da consulta GEOMD de mapeamento para a GOLAPI.

Quadro 9. Descrição do caso de uso *Processar Consulta de Integração MD-GEO*.

[UC04] Processar Consulta de Integração MD-GEO	
Ator	GOLAPE
Pré-condições	1. As consultas multidimensional e geográfica devem estar com a sintaxe correta; 2. A análise dos operadores presentes nas declarações das consultas multidimensional e geográfica define que a ordem de execução da consulta de integração deve ser MD-GEO.
Fluxo Principal	1. Consultar os metadados a fim de avaliar se as consultas multidimensional e geográfica foram corretamente escritas. Isto é, se a declaração destas consultas possui campos que referenciam objetos em comum; 2. Submeter a consulta multidimensional para processamento pelo servidor OLAP; 3. Submeter a consulta geográfica para processamento pelo SGBD com extensão espacial; 4. Consultar os metadados a fim de capturar os *UniqueName* dos registros retornados nas duas consultas; 5. Comparar os registros retornados das duas consultas através de seus identificadores *UniqueName*; 6. Montar o resultado final com os registros da consulta multidimensional e as geometrias da consulta geográfica que satisfaçam a interseção das duas consultas.
Fluxo Alternativo 1	1. No passo 1 do fluxo principal, caso não existam atributos em comum, uma exceção deve ser lançada; 2. Encerra-se este caso de uso.
Pós-condições	GOLAPE deve estar apto a enviar o resultado da consulta GEOMD para a GOLAPI.

Quadro 10. Descrição do caso de uso *Processar Consulta de Integração GEO-MD*.

[UC05] Processar Consulta de Integração GEO-MD	
Ator	GOLAPE
Pré-condições	1. As consultas multidimensional e geográfica devem estar com a sintaxe correta; 2. A análise dos operadores presentes nas declarações das consultas multidimensional e geográfica define que a ordem de execução da consulta de integração deve ser GEO-MD.
Fluxo Principal	1. Consultar os metadados a fim de avaliar se as consultas multidimensional e geográfica foram corretamente escritas. Isto é, se a declaração destas consultas possui campos que referenciam objetos em comum; 2. Submeter a consulta geográfica para processamento pelo SGBD com extensão espacial; 3. Consultar os metadados a fim de capturar os *UniqueName* dos registros retornados na consulta espacial; 4. Reescrever a consulta multidimensional substituindo os campos que fazem referência a objetos geográficos pelos *UniqueName* resultantes do passo 3; 5. Submeter a consulta multidimensional para processamento pelo servidor OLAP; 6. Montar o resultado final com os registros da consulta multidimensional e as geometrias da consulta geográfica.
Fluxo Alternativo 1	1. No passo 1 do fluxo principal, caso não existam atributos em comum, uma exceção deve ser lançada; 2. Encerra-se este caso de uso.
Pós-condições	GOLAPE deve estar apto a enviar o resultado da consulta GEOMD para a GOLAPI.

4.2 Geração dos Casos de Teste

Como foi dito no início deste capítulo, o processo de geração dos casos de teste adotado para este trabalho é o especificado por Heumann [Heu01], que está descrito na seção 3.2.2.1. Para cada caso de uso são executados os três passos do processo: geração de cenários, identificação dos casos de teste e identificação das entradas para os casos de teste.

4.2.1 Geração de Cenários

Os cenários para os cinco casos de uso especificados para o componente GOLAPE estão descritos nos quadros 11, 12 e 13 exibidos a seguir. Cada cenário corresponde a um passo a passo de execução do caso de uso e, para cada caso de uso, são gerados todos os cenários possíveis. Com isso, todas as possibilidades de execução de cada caso de uso são

cobertas pela geração dos casos de teste, ao final da terceira etapa deste processo. Note que o Quadro 11 agrupa os cenários referentes aos casos de uso [UC01] e [UC02], pois suas descrições resultam na mesma combinação de cenários. Da mesma forma, o

Quadro 13 agrupa os cenários referentes aos casos de uso [UC04] e [UC05], já que suas descrições também resultam na mesma combinação de cenários.

Quadro 11. Cenários para os casos de uso *Processar Consulta Multidimensional e Processar Consulta Geográfica*.

[UC01] Processar Consulta Multidimensional [UC02] Processar Consulta Geográfica		
Nome do Cenário	**Fluxo Inicial**	**Alteração**
Cenário 1 – Consulta executada com sucesso.	Fluxo Principal	Nenhuma

Quadro 12. Cenários para o caso de uso *Processar Consulta de Mapeamento*.

[UC03] Processar Consulta de Mapeamento		
Nome do Cenário	**Fluxo Inicial**	**Alteração**
Cenário 1 – Consulta executada com sucesso.	Fluxo Principal	Nenhuma
Cenário 2 – Consulta não referencia objeto geográfico.	Fluxo Principal	Fluxo Alternativo 1

Quadro 13. Cenários para os casos de uso *Processar Consulta de Integração MD-GEO e GEO-MD*.

[UC04] Processar Consulta de Integração MD-GEO [UC05] Processar Consulta de Integração GEO-MD		
Nome do Cenário	**Fluxo Inicial**	**Alteração**
Cenário 1 – Consulta de integração executada com sucesso.	Fluxo Principal	Nenhuma
Cenário 2 – Consultas não referenciam objetos em comum.	Fluxo Principal	Fluxo Alternativo 1

4.2.2 Identificação dos Casos de Teste

A partir dos cenários, são criados os respectivos casos de teste, de acordo com a segunda etapa da metodologia adotada neste trabalho para geração dos casos de teste. Os quadros 14, 15, 16 e 17 exibem os casos de teste para os casos de uso do engenho de consultas GOLAPE. Note que os casos de teste referentes aos casos de uso [UC04] e

[UC05] foram agrupados no Quadro 17, assim como os seus cenários na seção 4.2.1. Isto acontece porque, apesar de os casos de uso diferirem nas descrições dos seus passos, eles possuem a mesma condição para que ocorra o fluxo alternativo, que também é semelhante.

Quadro 14. Casos de teste para o caso de uso *Processar Consulta Multidimensional*.

[UC01] Processar Consulta Multidimensional			
ID Caso de Teste	**Cenário**	**Consulta Multidimensional**	**Resultado Esperado**
CT01	Cenário 1	Válida	Consulta executada com sucesso.

Quadro 15. Casos de teste para o caso de uso *Processar Consulta Geográfica*.

[UC02] Processar Consulta Geográfica			
ID Caso de Teste	**Cenário**	**Consulta Geográfica**	**Resultado Esperado**
CT01	Cenário 1	Válida	Consulta executada com sucesso.

Quadro 16. Casos de teste para o caso de uso *Processar Consulta de Mapeamento*.

[UC03] Processar Consulta de Mapeamento				
ID Caso de Teste	**Cenário**	**Consulta Multidimensional**	**Consulta referencia objeto geográfico?**	**Resultado Esperado**
CT01	Cenário 1	Válida	Sim	Consulta executada com sucesso.
CT02	Cenário 2	Válida	Não	Exceção lançada.

Quadro 17. Casos de teste para os casos de uso *Processar Consulta de Integração MD-GEO e GEO-MD*.

[UC04] Processar Consulta de Integração MD-GEO [UC05] Processar Consulta de Integração GEO-MD					
ID Caso de Teste	**Cenário**	**Consulta Multidimensional**	**Consulta Geográfica**	**Consultas referenciam objetos em comum?**	**Resultado Esperado**
CT01	Cenário 1	Válida	Válida	Sim	Consulta de integração executada com sucesso.
CT02	Cenário 2	Válida	Válida	Não	Exceção lançada.

4.2.3 Identificação das Entradas para os Casos de Teste

A partir dos casos de teste gerados, é necessário identificar as entradas para possibilitar a execução dos testes. Este passo representa a última etapa do processo de geração dos casos de teste a partir dos casos de uso. As subseções 4.2.3.1 e 4.2.3.2 trazem as consultas multidimensionais e geográficas que serão usadas como entradas para os casos de teste. Estas consultas foram elaboradas para serem executadas sobre o DWG apresentado em [Med06], cujo esquema multidimensional é exibido na Figura 11.

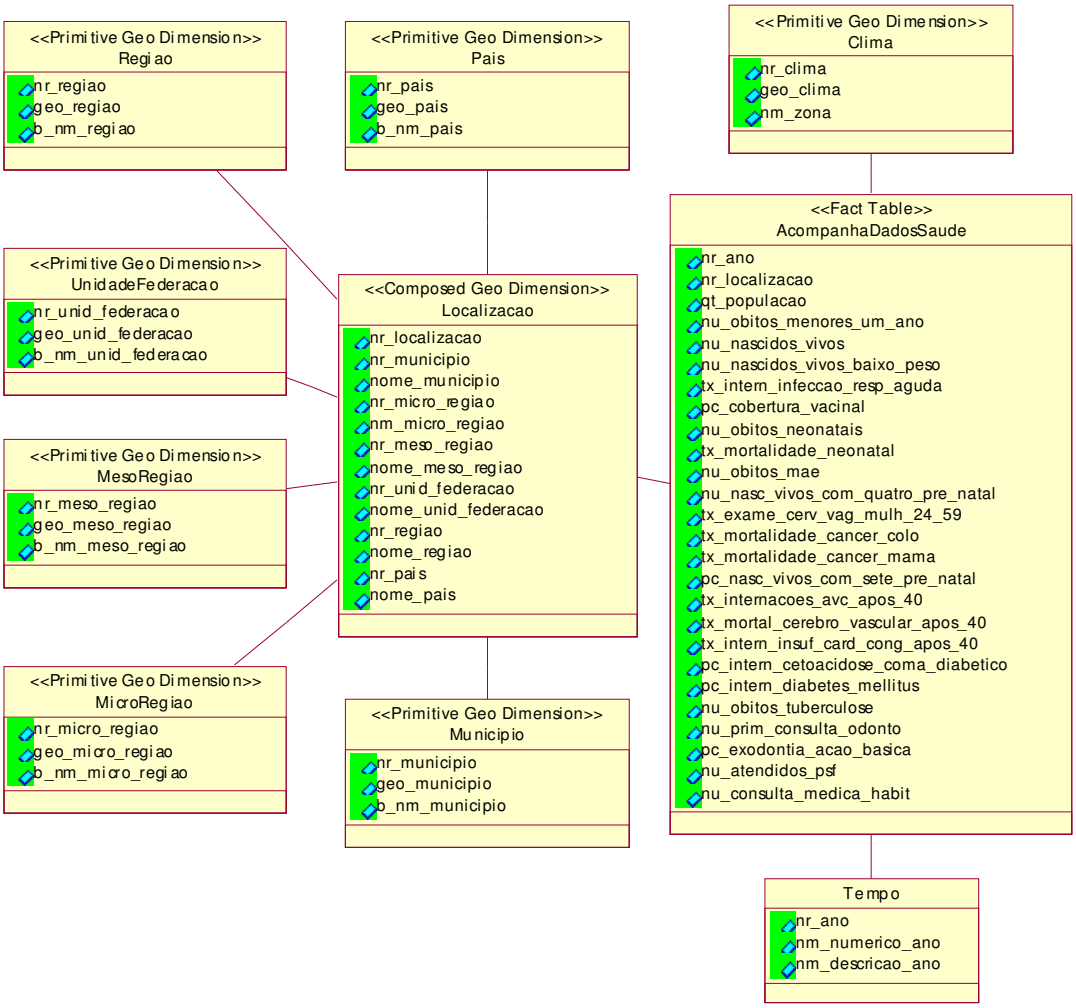

Figura 11. Esquema DWG DataSUS [Med06].

O esquema multidimensional do DWG exibido na Figura 11 segue as orientações gerais do GeoDWFrame [FTS+04], o qual define conceitos e princípios de projeto para a modelagem de um DWG. Este esquema apresenta a dimensão geográfica composta *Localizacao* associada às seguintes dimensões geográficas primitivas: *Pais*, *Regiao*, *Unidade-Federacao*, *MesoRegiao*, *MicroRegiao* e *Municipio*. O modelo apresenta também a dimensão convencional *Tempo* e a dimensão geográfica primitiva *Clima*. A tabela de fatos *AcompanhaDadosSaude* armazena medidas que permitem análises focadas no setor de assistência à saúde como, por exemplo: número de nascidos vivos, número de óbitos menores de um ano, número de óbitos neonatais, número de pessoas atendidas pelo Programa de Saúde da Família (PSF), taxa de mortalidade por câncer de colo e de mama, número de óbitos por tuberculose, entre outros.

4.2.3.1 Consultas Multidimensionais

O Quadro18 apresenta as descrições das consultas multidimensionais usadas como entradas para os testes. No Apêndice A, encontram-se as consultas detalhadas referentes a estas descrições, especificadas na linguagem MDX (*Multi-dimensional Expressions*) compatível com o *Mondrian* [MON07], que é um servidor OLAP de código aberto. Este servidor foi projetado conforme a arquitetura ROLAP e foi estendido em [Med06], visando a implementação de uma instância da arquitetura GOLAPA. Visto que o universo de funções MDX não é pequeno (mais de 100 funções [MDX07]), optou-se por usar as funções que representam as operações OLAP mais encontradas na literatura [SAS07] [HYP04]. Isto é, as funções *DrilldownLevel, DrilldownMember, order, crossjoin, NonEmptyCrossjoin, hierarchize* e *non empty*. Maiores detalhes sobre a sintaxe e a funcionalidade de cada função podem ser encontrados em [MDX07].

Quadro 18. Descrições das consultas multidimensionais usadas como entradas para os testes.

ID Consulta	Descrição da Consulta
MD01	Selecionar a população do ano de 2002 de todas a micro-regiões e municípios, separada por clima.
MD02	Selecionar a população do ano de 2002 de todas micro-regiões e dos municípios da micro-região ILHEUS-ITABUNA, separada por clima.
MD03	Selecionar a população do ano de 2002 de todas as micro-regiões, ordenadas pela população e separadas por clima.
MD04	Selecionar a população do ano de 2002 de todas as micro-regiões, cruzando-as com os climas.
MD05	Selecionar a população do ano de 2002 de todas as micro-regiões, cruzando-as com os climas e eliminando os registros vazios.
MD06	Selecionar a população do ano de 2002 de todas as micro-regiões e dos municípios da micro-região ILHEUS-ITABUNA, separada por clima e ordenada em uma hierarquia.
MD07	Selecionar a população do ano de 2002 de todas as micro-regiões, separada por clima e eliminando os registros vazios.
MD08	Selecionar a população do ano de 2002 de todas as micro-regiões, separada por clima.
MD09	Selecionar a população por ano.
MD10	Selecionar a população para o ano de 2002 de todas as meso-regiões, separada por clima.
MD11	Selecionar a população de todos os estados, para todos os anos e separada por clima.
MD12	Selecionar os nascidos vivos por ano.
MD13	Selecionar a população do ano de 2002 das regiões Norte e Sul, separada por clima.

4.2.3.2 Consultas Geográficas

O Quadro19 apresenta as descrições das consultas geográficas usadas como entradas para os testes. No Apêndice A encontram-se as consultas detalhadas referentes a estas descrições, especificadas em linguagem SQL compatível com a extensão espacial *PostGIS* [POS07] do SGBD de código aberto *PostgreSQL* [POS07b]. Visto que a quantidade de operações geométricas suportadas pelo *PostGIS* não é pequena (mais de 150 operações [POS07c]), optou-se por usar as operações topológicas. Isto é, as operações: *contains*, *touches*, *crosses*, *within*, *overlaps*, *intersects* e *disjoint*. Maiores detalhes sobre a sintaxe e a funcionalidade de cada operação podem ser encontrados em [POS07c].

Quadro 19. Descrições das consultas geográficas usadas como entradas para os testes.

ID Consulta	Descrição da Consulta
GEO01	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que estão contidas nele.
GEO02	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que tocam ele.
GEO03	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que cruzam ele.
GEO04	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que contém ele.
GEO05	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que se sobrepõem a ele.
GEO06	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que intersectam ele.
GEO07	Selecionar as geometrias do clima Tropical Brasil Central e das micro-regiões que são disjuntas a ele.
GEO08	Selecionar as geometrias do clima Tropical Brasil Central e das meso-regiões que estão contidas nele.
GEO09	Selecionar as geometrias do clima Tropical Brasil Central e das unidades da federação que se sobrepõem a ele.

4.2.3.3 Casos de Teste

Os quadros 20 e 21 representam os casos de teste referentes aos casos de uso [UC03], [UC04] e [UC05], que foram especificados na seção 4.2.2, com suas respectivas entradas. Os casos de teste relativos aos casos de uso [UC01] e [UC02] foram omitidos para evitar repetição, já que possuem apenas um cenário, no qual o resultado esperado para todas as entradas é a execução da consulta com sucesso. É importante observar que em cada quadro está contido o ID único de cada caso de teste. Este ID corresponde ao identificador do caso de teste mais um número seqüencial que define uma entrada possível (*i.e.* multidimensional, geográfica ou multidimensional-geográfica). Por exemplo, o ID CT01-02 define que o caso de teste CT01 está sendo executado com a sua segunda entrada. Esta etapa caracteriza-se como a última parte da especificação dos testes a partir dos casos de uso.

Quadro 20. Casos de teste com entradas para o caso de uso *Processar Consulta de Mapeamento*.

[UC03] Processar Consulta de Mapeamento				
ID Caso de Teste	**Cenário**	**Consulta Multidimensional**	**Consulta retorna geometrias?**	**Resultado Esperado**
CT01-01	Cenário 1	MD01	Sim	Consulta executada com sucesso.
CT01-02	Cenário 1	MD02	Sim	Consulta executada com sucesso.
CT01-03	Cenário 1	MD03	Sim	Consulta executada com sucesso.
CT01-04	Cenário 1	MD04	Sim	Consulta executada com sucesso.
CT01-05	Cenário 1	MD05	Sim	Consulta executada com sucesso.
CT01-06	Cenário 1	MD06	Sim	Consulta executada com sucesso.
CT01-07	Cenário 1	MD07	Sim	Consulta executada com sucesso.
CT01-08	Cenário 1	MD08	Sim	Consulta executada com sucesso.
CT01-09	Cenário 1	MD10	Sim	Consulta executada com sucesso.
CT01-10	Cenário 1	MD11	Sim	Consulta executada com sucesso.
CT01-11	Cenário 1	MD13	Sim	Consulta executada com sucesso.
CT02-01	Cenário 2	MD09	Não	Exceção lançada.
CT02-02	Cenário 2	MD12	Não	Exceção lançada.

Quadro 21. Casos de teste com entradas para os casos de uso *Processar Consulta de Integração MD-GEO e GEO-MD*.

[UC04] Processar Consulta de Integração MD-GEO [UC05] Processar Consulta de Integração GEO-MD					
ID Caso de Teste	**Cenário**	**Consulta Multidimensional**	**Consulta Geográfica**	**Consultas referenciam objetos em comum?**	**Resultado Esperado**
CT01-01	Cenário 1	MD03	GEO01	Sim	Consulta de integração executada com sucesso
CT01-02	Cenário 1	MD03	GEO02	Sim	Consulta de integração executada com sucesso
CT01-03	Cenário 1	MD03	GEO03	Sim	Consulta de integração executada com sucesso
CT01-04	Cenário 1	MD03	GEO04	Sim	Consulta de integração executada com sucesso
CT01-05	Cenário 1	MD03	GEO05	Sim	Consulta de integração executada com sucesso
CT01-06	Cenário 1	MD03	GEO06	Sim	Consulta de integração executada com sucesso
CT01-07	Cenário 1	MD03	GEO07	Sim	Consulta de integração executada com sucesso
CT01-08	Cenário 1	MD04	GEO01	Sim	Consulta de integração executada com sucesso
CT01-09	Cenário 1	MD04	GEO02	Sim	Consulta de integração executada com sucesso
CT01-10	Cenário 1	MD04	GEO03	Sim	Consulta de integração executada com sucesso
CT01-11	Cenário 1	MD04	GEO04	Sim	Consulta de integração executada com sucesso
CT01-12	Cenário 1	MD04	GEO05	Sim	Consulta de integração executada com sucesso
CT01-13	Cenário 1	MD04	GEO06	Sim	Consulta de integração executada com sucesso
CT01-14	Cenário 1	MD04	GEO07	Sim	Consulta de integração executada com sucesso
CT01-15	Cenário 1	MD05	GEO01	Sim	Consulta de integração executada com sucesso
CT01-16	Cenário 1	MD05	GEO02	Sim	Consulta de integração executada com sucesso
CT01-17	Cenário 1	MD05	GEO03	Sim	Consulta de integração executada com sucesso
CT01-18	Cenário 1	MD05	GEO04	Sim	Consulta de integração executada com sucesso
CT01-19	Cenário 1	MD05	GEO05	Sim	Consulta de integração executada com sucesso
CT01-20	Cenário 1	MD05	GEO06	Sim	Consulta de integração executada com sucesso
CT01-21	Cenário 1	MD05	GEO07	Sim	Consulta de integração executada com sucesso
CT01-22	Cenário 1	MD07	GEO01	Sim	Consulta de integração executada com sucesso
CT01-23	Cenário 1	MD07	GEO02	Sim	Consulta de integração executada com sucesso
CT01-24	Cenário 1	MD07	GEO03	Sim	Consulta de integração executada com sucesso
CT01-25	Cenário 1	MD07	GEO04	Sim	Consulta de integração executada com sucesso

CT01-26	Cenário 1	MD07	GEO05	Sim	Consulta de integração executada com sucesso
CT01-27	Cenário 1	MD07	GEO06	Sim	Consulta de integração executada com sucesso
CT01-28	Cenário 1	MD07	GEO07	Sim	Consulta de integração executada com sucesso
CT01-29	Cenário 1	MD08	GEO01	Sim	Consulta de integração executada com sucesso
CT01-30	Cenário 1	MD08	GEO02	Sim	Consulta de integração executada com sucesso
CT01-31	Cenário 1	MD08	GEO03	Sim	Consulta de integração executada com sucesso
CT01-32	Cenário 1	MD08	GEO04	Sim	Consulta de integração executada com sucesso
CT01-33	Cenário 1	MD08	GEO05	Sim	Consulta de integração executada com sucesso
CT01-34	Cenário 1	MD08	GEO06	Sim	Consulta de integração executada com sucesso
CT01-35	Cenário 1	MD08	GEO07	Sim	Consulta de integração executada com sucesso
CT01-36	Cenário 1	MD10	GEO08	Sim	Consulta de integração executada com sucesso
CT01-37	Cenário 1	MD11	GEO09	Sim	Consulta de integração executada com sucesso
CT02-01	Cenário 2	MD01	GEO01	Não	Exceção lançada
CT02-02	Cenário 2	MD01	GEO02	Não	Exceção lançada
CT02-03	Cenário 2	MD01	GEO03	Não	Exceção lançada
CT02-04	Cenário 2	MD01	GEO04	Não	Exceção lançada
CT02-05	Cenário 2	MD01	GEO05	Não	Exceção lançada
CT02-06	Cenário 2	MD01	GEO06	Não	Exceção lançada
CT02-07	Cenário 2	MD01	GEO07	Não	Exceção lançada
CT02-08	Cenário 2	MD01	GEO08	Não	Exceção lançada
CT02-09	Cenário 2	MD01	GEO09	Não	Exceção lançada
CT02-10	Cenário 2	MD02	GEO01	Não	Exceção lançada
CT02-11	Cenário 2	MD02	GEO02	Não	Exceção lançada
CT02-12	Cenário 2	MD02	GEO03	Não	Exceção lançada
CT02-13	Cenário 2	MD02	GEO04	Não	Exceção lançada
CT02-14	Cenário 2	MD02	GEO05	Não	Exceção lançada
CT02-15	Cenário 2	MD02	GEO06	Não	Exceção lançada
CT02-16	Cenário 2	MD02	GEO07	Não	Exceção lançada
CT02-17	Cenário 2	MD02	GEO08	Não	Exceção lançada
CT02-18	Cenário 2	MD02	GEO09	Não	Exceção lançada
CT02-19	Cenário 2	MD03	GEO08	Não	Exceção lançada
CT02-20	Cenário 2	MD03	GEO09	Não	Exceção lançada
CT02-21	Cenário 2	MD04	GEO08	Não	Exceção lançada
CT02-22	Cenário 2	MD04	GEO09	Não	Exceção lançada
CT02-23	Cenário 2	MD05	GEO08	Não	Exceção lançada
CT02-24	Cenário 2	MD05	GEO09	Não	Exceção lançada
CT02-25	Cenário 2	MD06	GEO01	Não	Exceção lançada
CT02-26	Cenário 2	MD06	GEO02	Não	Exceção lançada
CT02-27	Cenário 2	MD06	GEO03	Não	Exceção lançada
CT02-28	Cenário 2	MD06	GEO04	Não	Exceção lançada
CT02-29	Cenário 2	MD06	GEO05	Não	Exceção lançada
CT02-30	Cenário 2	MD06	GEO06	Não	Exceção lançada
CT02-31	Cenário 2	MD06	GEO07	Não	Exceção lançada
CT02-32	Cenário 2	MD06	GEO08	Não	Exceção lançada
CT02-33	Cenário 2	MD06	GEO09	Não	Exceção lançada
CT02-34	Cenário 2	MD07	GEO08	Não	Exceção lançada
CT02-35	Cenário 2	MD07	GEO09	Não	Exceção lançada

CT02-36	Cenário 2	MD08	GEO08	Não	Exceção lançada
CT02-37	Cenário 2	MD08	GEO09	Não	Exceção lançada
CT02-38	Cenário 2	MD09	GEO01	Não	Exceção lançada
CT02-39	Cenário 2	MD09	GEO02	Não	Exceção lançada
CT02-40	Cenário 2	MD09	GEO03	Não	Exceção lançada
CT02-41	Cenário 2	MD09	GEO04	Não	Exceção lançada
CT02-42	Cenário 2	MD09	GEO05	Não	Exceção lançada
CT02-43	Cenário 2	MD09	GEO06	Não	Exceção lançada
CT02-44	Cenário 2	MD09	GEO07	Não	Exceção lançada
CT02-45	Cenário 2	MD09	GEO08	Não	Exceção lançada
CT02-46	Cenário 2	MD09	GEO09	Não	Exceção lançada
CT02-47	Cenário 2	MD10	GEO01	Não	Exceção lançada
CT02-48	Cenário 2	MD10	GEO02	Não	Exceção lançada
CT02-49	Cenário 2	MD10	GEO03	Não	Exceção lançada
CT02-50	Cenário 2	MD10	GEO04	Não	Exceção lançada
CT02-51	Cenário 2	MD10	GEO05	Não	Exceção lançada
CT02-52	Cenário 2	MD10	GEO06	Não	Exceção lançada
CT02-53	Cenário 2	MD10	GEO07	Não	Exceção lançada
CT02-54	Cenário 2	MD10	GEO09	Não	Exceção lançada
CT02-55	Cenário 2	MD11	GEO01	Não	Exceção lançada
CT02-56	Cenário 2	MD11	GEO02	Não	Exceção lançada
CT02-57	Cenário 2	MD11	GEO03	Não	Exceção lançada
CT02-58	Cenário 2	MD11	GEO04	Não	Exceção lançada
CT02-59	Cenário 2	MD11	GEO05	Não	Exceção lançada
CT02-60	Cenário 2	MD11	GEO06	Não	Exceção lançada
CT02-61	Cenário 2	MD11	GEO07	Não	Exceção lançada
CT02-62	Cenário 2	MD11	GEO08	Não	Exceção lançada
CT02-63	Cenário 2	MD12	GEO01	Não	Exceção lançada
CT02-64	Cenário 2	MD12	GEO02	Não	Exceção lançada
CT02-65	Cenário 2	MD12	GEO03	Não	Exceção lançada
CT02-66	Cenário 2	MD12	GEO04	Não	Exceção lançada
CT02-67	Cenário 2	MD12	GEO05	Não	Exceção lançada
CT02-68	Cenário 2	MD12	GEO06	Não	Exceção lançada
CT02-69	Cenário 2	MD12	GEO07	Não	Exceção lançada
CT02-70	Cenário 2	MD12	GEO08	Não	Exceção lançada
CT02-71	Cenário 2	MD12	GEO09	Não	Exceção lançada
CT02-72	Cenário 2	MD13	GEO01	Não	Exceção lançada
CT02-73	Cenário 2	MD13	GEO02	Não	Exceção lançada
CT02-74	Cenário 2	MD13	GEO03	Não	Exceção lançada
CT02-75	Cenário 2	MD13	GEO04	Não	Exceção lançada
CT02-76	Cenário 2	MD13	GEO05	Não	Exceção lançada
CT02-77	Cenário 2	MD13	GEO06	Não	Exceção lançada
CT02-78	Cenário 2	MD13	GEO07	Não	Exceção lançada
CT02-79	Cenário 2	MD13	GEO08	Não	Exceção lançada
CT02-80	Cenário 2	MD13	GEO09	Não	Exceção lançada

4.3 Resultados da Execução dos Testes

Nesta seção, são apresentados os resultados obtidos com a execução dos testes sobre uma instância do componente GOLAPE descrita em [Med06]. Cada caso de teste, relativo a cada caso de uso, foi alimentado com as respectivas entradas, conforme especificado na seção 4.2.3.3. Para facilitar a visualização e evitar repetições, apenas os casos de teste que não foram executados com sucesso são exibidos nos quadros 22, 23 e 24, separados por caso de uso. Caso exista algum comentário pertinente sobre a execução de algum caso de teste, esta informação também está disponível nestes quadros.

Quadro 22. Resultado da execução dos testes para o caso de uso *Processar Consulta de Mapeamento*.

[UC03] Processar Consulta de Mapeamento	
ID Caso de Teste	**Comentário**
CT01-01	Foi lançada uma exceção inesperada.
CT01-02	Foi lançada uma exceção inesperada.
CT01-03	Foi lançada uma exceção inesperada.
CT01-04	Foi lançada uma exceção inesperada.
CT01-05	Foi lançada uma exceção inesperada.
CT01-06	Foi lançada uma exceção inesperada.
CT01-07	Foi lançada uma exceção inesperada.
CT01-08	Foi lançada uma exceção inesperada.
CT02-01	Resultado final da consulta vazio. Exceção não lançada.
CT02-02	Resultado final da consulta vazio. Exceção não lançada.

Quadro 23. Resultado da execução dos testes para o caso de uso *Processar Consulta de Integração MD-GEO*.

[UC04] Processar Consulta de Integração MD-GEO	
ID Caso de Teste	**Comentário**
CT02-21	Resultado final da consulta não vazio. Exceção não lançada.
CT02-22	Resultado final da consulta não vazio. Exceção não lançada.
CT02-23	Resultado final da consulta não vazio. Exceção não lançada.
CT02-24	Resultado final da consulta não vazio. Exceção não lançada.
CT02-34	Resultado final da consulta não vazio. Exceção não lançada.

Quadro 24. Resultado da execução dos testes para o caso de uso *Processar Consulta de Integração GEO-MD*.

[UC05] Processar Consulta de Integração GEO-MD	
ID Caso de Teste	**Comentário**
CT02-20	Resultado final da consulta não vazio. Exceção não lançada.
CT02-21	Resultado final da consulta não vazio. Exceção não lançada.
CT02-22	Resultado final da consulta não vazio. Exceção não lançada.
CT02-23	Resultado final da consulta não vazio. Exceção não lançada.
CT02-24	Resultado final da consulta não vazio. Exceção não lançada.
CT02-37	Resultado final da consulta não vazio. Exceção não lançada.

É importante observar que a execução de todos os casos de teste foi realizada com sucesso apenas para os casos de uso [UC01] e [UC02], visto que não foi relatado nenhum insucesso nesta seção. Ou seja, para o conjunto de entradas usadas, o conjunto de saídas obtidas foi igual ao conjunto de saídas esperadas. Desta forma, estes resultados podem servir como indicador positivo em relação a corretude da implementação das funcionalidades *Processar Consulta Multidimensional* e *Processar Consulta Geográfica* do engenho de consultas GOLAPE.

Por outro lado, a execução dos casos de teste relativos aos casos de uso [UC03], [UC04] e [UC05] encontrou alguns problemas, exibidos nos quadros 22, 23 e 24, respectivamente. No primeiro caso, foram encontrados problemas em 10 dos 13 casos de teste especificados, nos quais o programa não se comportou como se esperava. No segundo caso, a execução encontrou problemas em 5 dos 117 casos de teste, nos quais era esperado o lançamento de uma exceção, que não ocorreu. Já no terceiro caso, 6 dos 117 casos de teste executados indicaram a presença de falhas, pelo mesmo motivo do segundo caso. Por fim, pode-se concluir que as funcionalidades *Processar Consulta de Mapeamento*, *Processar Consultar de Integração MD-GEO* e *Processar Consulta de Integração GEO-MD* não estão corretamente implementadas pela instância da arquitetura GOLAPA em questão.

4.4 Considerações Finais

Este capítulo apresentou todos os passos do processo de especificação dos testes funcionais para o componente GOLAPE, que resultou em um conjunto de casos de teste para cada caso de uso. Visando possibilitar esta especificação, foi elaborada a descrição para os casos de uso referentes às funcionalidades do engenho de consultas SOLAP. E

neste sentido, foram também elaboradas as consultas multidimensionais e geográficas usadas como entradas para os casos de teste, bem como foi apresentado o DWG que serviu de base para esta elaboração.

Também é possível observar neste capítulo, que se encontram relatados os resultados da execução dos testes e os comentários feitos sobre da execução de cada caso de teste. Com isto, foi possível validar a especificação dos testes, além de ter possibilitado a identificação dos problemas existentes em algumas das funcionalidades da instância do componente GOLAPE testado.

CAPÍTULO 5 - Conclusões

Este capítulo tem como objetivo apresentar as considerações finais acerca dos principais tópicos e conceitos presentes neste trabalho, incluindo as contribuições alcançadas e indicações para trabalhos futuros.

5.1 Considerações Finais

Neste trabalho, foram descritos a arquitetura GOLAPA e os seus principais componentes, assim como as tecnologias DW, OLAP e SIG, que apóiam a tomada de decisão e são integradas nesta arquitetura. Foram também descritos conceitos relacionados à área de testes de software, principalmente sobre os testes funcionais, e algumas técnicas de especificação de testes.

Ao longo do desenvolvimento desta pesquisa foram encontradas tais dificuldades: (1) falta de uma sistemática consolidada para especificação de testes funcionais, (2) não existência de casos de uso para o componente GOLAPE, (3) problemas para instalação da instância da arquitetura GOLAPA sob a qual foram realizados os testes e (4) inadequação da ferramenta gratuita para automatização dos testes.

O principal objetivo deste trabalho foi alcançado com a especificação dos testes funcionais para o processador de consultas da arquitetura GOLAPA. Esta especificação foi realizada seguindo os passos da metodologia de geração de casos de teste a partir dos casos de uso, conforme descrito no Capítulo 4.

Além disso, foram executados todos os casos de teste elaborados na especificação, validando o estudo desenvolvido neste trabalho. Os resultados obtidos com a execução dos testes possibilitaram a identificação de falhas na implementação de algumas funcionalidades na instância da arquitetura GOLAPA tomada como base para a realização dos testes.

5.2 Contribuições

São destacadas como principais contribuições deste trabalho:

- Especificação dos casos de uso para o componente GOLAPE, na qual foram detalhados os passos executados em cada funcionalidade deste componente.

Esta especificação não existia e por isso, contribui para o processo de documentação da arquitetura GOLAPA;

- Especificação dos casos de teste para o componente GOLAPE, referentes a todos os casos de uso especificados para este componente. Os casos de teste foram elaborados de forma a exercitar todas as possibilidades de execução de cada caso de uso;

- Validação da especificação dos testes, através da execução dos mesmos em uma instância da arquitetura GOLAPA. Esta execução também possibilitou a identificação de falhas na implementação, contribuindo para a melhoria do nível de qualidade do sistema.

5.3 Trabalhos Futuros

Embora os objetivos deste trabalho tenham sido alcançados, ainda existem temas de pesquisas e implementações que podem ser realizados:

- Especificação de testes estruturais para o processador de consultas da arquitetura GOLAPA, uma vez que este trabalho visa apenas os testes funcionais. Visto que esses dois tipos de teste são complementares, sair do universo restrito aos testes funcionais pode vir a aumentar a qualidade do sistema;

- Especificação de testes para o componente GOLAPI. Existe a necessidade de se definir uma sistemática para testar a interface gráfica da arquitetura GOLAPA, já que todas as consultas multidimensionais e/ou geográficas precisam ser validadas e o componente GOLAPE deve receber apenas consultas que estiverem sintática e semanticamente corretas. Além disso, o resultado das informações retornadas pelas consultas deve ser exibido de

maneira satisfatória na interface. Portanto, será necessário também que os testes possuam um foco voltado para a usabilidade deste componente.

Referências Bibliográficas

[BAM+04] BARBOSA, D. L.; ANDRADE, W. L.; MACHADO, P. D. L.; FIGUEREDO, J. C. A. ***SPACES – Uma Ferramenta para Teste Funcional de Componentes***, XVIII Simpósio Brasileiro de Engenharia de Software, 2004.

[BG04] BIANCHINI, J.; GRAHL, E. A. ***TestCen: Ferramenta de Suporte ao Planejamento de Teste Funcional de Software a partir de Diagramas de Caso de Uso***, XIII SEMINCO – Universidade Regional de Blumenau, 2004.

[BMV+00] BARBOSA, E. F.; MALDONADO, J. C.; VINCENZI, A. M. R.; DELAMARO, M. E.; SOUZA, S. R. S.; JINO, M. ***Introdução ao Teste de Software***, EIN'2000 - II Escola de Informática Norte, 2000.

[Bra04] BRAUDE, E. ***Software Design: from programming to architecture***, John Wiley & Sons, Inc., 2004.

[CCH+96] CÂMARA, G.; CASANOVA, M. A.; HEMERLY, A. S.; MAGALHÃES, G. C.; MEDEIROS, C. M. B. ***Anatomia de Sistemas de Informação Geográfica,*** SBC X Escola de Computação, 1996.

[CD97] CHAUDHURI, S.; DAYAL, U. ***An overview of Data Warehousing and OLAP Technology*****,** SIGMOD Record, 26(1): 65--74. 1997.

[Coc01] COCKBURN, A. ***Writing Effective Use Cases***, Addison-Wesley, 2001.

[Cop04] COPELAND, L. ***A Practitioner's Guide to Software Test Design***, Artech House, 2004.

[Dat00] DATE, C. J. ***An Introduction to Database Systems***, 7th Edition. Addison-Wesley,2000.

[Dem00] DEMERS, M. N. ***Fundamentals of Geographic Information Systems***. 2nd ed. John Wiley & Sons, 2000.

[EH91] EGENHOFER, M.; HERRING, J. ***Categorizing Binary Topological Relationships Between Regions, Lines and Points in Geographic Databases***, Department of Surveying Engineering, University of Maine, Orono, ME, 1991.

[FCT01] FERREIRA, A. C.; CAMPOS, M. L.; TANAKA, A. ***An Architecture for Spatial and Dimensional Analysis Integration***. Proceedings of the World Multiconference on Systemics, Cibernetics and Informatics (SCI 2001), Volume XIV Computer Science and Engineering, 2001.

[Fer02] FERREIRA, A. C. ***Um Modelo para Suporte à Integração de Análises Multidimensionais e Espaciais***. Dissertação de Mestrado - UFRJ, 2002.

[Fid05] FIDALGO, R. N. ***Uma Infra-estrutura para Integração de Modelos, Esquemas e Serviços Multidimensionais e Geográficos***, Teste de Doutorado – UFPE, 2005.

[Fir97] FIRESTONE, J. M. ***A Systems Approach to Dimensional Modeling in Data Marts***, 1997.

[FTS01] FIDALGO, R. N.; TIMES, V. C.; SOUZA, F. F. ***GOLAPA: Uma Arquitetura Aberta e Extensível para Integração entre SIG e OLAP***, Simpósio Brasileiro de Geoinformática - GEOINFO 2001.

[FTS+04] FIDALGO, R. N.; TIMES, V. C.; SILVA, J.; SOUZA, F. F. ***GeoDWFrame: A framework for guiding the design of geographical dimensional schemas***, In Proceedings of the 6th International Conference on Data Warehousing and Knowledge Discovery (DaWaK), 2004.

[FUN07] ***IBM Rational Functional Tester***. Dìsponível em <http://www-306.ibm.com/software/awdtools/tester/functional>. Último acesso em março de 2007.

[GML07] ***GML – the Geography Markup Language***. Disponível em: <http://www.opengis.net/gml>. Último acesso em março de 2007.

[Heu01] HEUMANN, J. ***Generating Test Cases From Use Cases***. Disponível em: <http://www.therationaledge.com/content/jun_01/m_cases_jh.html>. Último acesso em março de. 2007.

[HKS97] HAN, J.; KOPERSKI, K.; STEFANOVIC, N. ***GeoMiner: A System Prototype for Spatial Data Mining***. Proceedings of the 1997 ACM-SIGMOD Conference, 1997.

[HVF+05] HARTMANN, J.; VIEIRA, M.; FOSTER, H.; RUDER, A. ***A UML-based approach to system testing***, Innovations in Systems and Software Engineering, 2005.

[HYP04] ***MDX Allows Complex, Multidimensional Queries***, Hyperion Solutions Corporation, 2004.

[IIB96] INMON, W. H.; IMHOFF, C.; BATTAS, G. ***Building the Operational Data Store***. John Wiley & Sons, Inc, 1996.

[Inm97] INMON, W. H. ***Como construir um Data Warehouse***, Editora Campus, 1997.

[Kim96] KIMBALL, R. ***The Data warehouse Lifecycle Toolkit***, John Wiley & Sons, Inc., 1996.

[JTE07] ***Parasoft Jtest***. Disponível em <http://www.parasoft.com/jsp/products/home.jsp?product=Jtest>. Último acesso em março de 2007.

[JUN07] ***JUnit, Testing Resources for Extreme Programming***. Disponível em <http://www.junit.org>. Último acesso em março de 2007.

[KRR+98] KIMBALL, R.; REEVES, L.; ROSS, M.; THORNTHWAITE, W. ***The Data Warehouse Lifecycle Toolkit***, John Wiley & Sons, Inc, 1998.

[LGM99] LONGLEY, P. A.; GOODCHILD, M. F.; MAGUIRE, D. J. ***Geographical Information Systems: Principles, Techniques, Applications and Management,*** 2nd Edition, John Wiley and Sons, 1999.

[MDX07] ***MDX Specification***, Pentaho Analysis Services: Mondrian Project. Disponível em <http://mondrian.pentaho.org/documentation/mdx.php>. Último acesso em março de 2007.

[Med06] MEDEIROS, V. N. ***Uma Infra-Estrutura Eficiente de Data Warehouse Geográfico Apoiada em Software Livre***, Dissertação de Mestrado – UFPE, 2006.

[Men02] MENDES, A. ***Arquitetura de Software: Desenvolvimento Orientado para a Arquitetura***, Editora Campus, 2002.

[Mod06] MODESTO, L. R. ***Teste Funcional Baseado em Diagramas UML***, Dissertação de Mestrado – UNIVEM, 2006.

[MON07] ***Pentaho Analysis Services: Mondrian Project***. Disponível em <http://mondrian.pentaho.org>. Último acesso em março de 2007.

[Nil03] NILAWAR, M. ***A UML-Based Approach for Testing Web Applications***, Dissertação de Mestrado, University of Nevada – Reno, 2003.

[Pai03] PAIM, F. R. S. ***Uma metodologia para análise de requisitos em sistemas Data Warehouse***, Dissertação de Mestrado - UFPE, 2003.

[POS07] ***PostGIS***. Disponível em <http://postgis.refractions.net/>. Último acesso em março de 2007.

[POS07b] ***PostgreSQL: The world's most advanced open source database***. Disponível em <http://www.postgresql.org/>. Último acesso em março de 2007.

[POS07c] ***PostGIS Manual***. Disponível em <http://postgis.refractions.net/docs/>. Último acesso em março de 2007.

[RBP+05] RIVEST, S.; BÉDARD, Y.; PROULX, M.; NADEAU, M.; HUBERT, F.; PASTOR, J. ***SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysis of data***, ISPRS Journal of Photogrammetry & Remote Sensing 60 (2005) 17 – 33, 2005.

[ROB07] ***IBM Rational Robot***. Disponível em <http://www-306.ibm.com/software/awdtools/tester/robot>. Último acesso em março de 2007.

[Roc05] ROCHA, C. M. ***Explorando o Relacionamento entre Métricas Baseadas em Caso de Uso e o Número de Casos de Teste***, Dissertação de Mestrado – UFPR, 2005.

[RSM+05] ROCHA, A. D.; SIMÃO, A. S.; MALDONADO, J. C.; MASIERO, P. C. ***Uma ferramenta baseada em aspectos para o teste funcional de programas Java***, XIX Simpósio Brasileiro de Engenharia de Software, 2005.

[SAS07] ***SAS 9.1 OLAP Server***. Disponível em <http://support.sas.com/documentation/onlinedoc/91pdf/sasdoc_91/olap_mdx_7002.pdf>. Último acesso em março de 2007.

[Sil01] SILVEIRA, F. F. ***Ferramenta de Apoio ao Teste de Aplicações Java Baseada em Reflexo Computacional***, Dissertação de Mestrado – UFRGS, 2001.

[Sil04] SILVA, J. ***Integrando Serviços Analíticos e Geográficos para Suporte à Decisão na Web***, Dissertação de Mestrado – UFPE, 2004.

[SLT+99] SHEKHAR, S.; LU, C.T.; TAN, X.; CHAWLA, S. ***Map Cube: A Visualization Tool for Spatial Data Warehouses - A Summary of Result***, Proceedings of NSF workshop on Data Mining in GIS, 1999.

[SLT+01] SHEKHAR, S.; LU, C.T.; TAN, X.; CHAWLA, ***Map Cube: A Visualization Tool for Spatial Data Warehouses***, as Chapter of Geographic Data Mining and Knowledge Discovery. Harvey J. Miller and Jiawei Han (eds.), Taylor and Francis, 2001.

[Som04] SOMMERVILLE, I. ***Engenharia de Software***, Addison-Wesley, 2004.

[STF+04] SILVA, J.; TIMES, V.; FIDALGO, R.; BARROS, R. ***Towards a Web Service for Geographic and Multidimensional Processing***, Proceedings of the Brazilian Symposium on Geoinformatics (GeoInfo), 1--17, 2004.

[STF+05] SILVA, J.; TIMES, V. C.; FIDALGO, R. N.; BARROS, R. S. M. ***Providing Geographic-Multidimensional Decision Support over the Web***, Proceedings of the Seventh Asia Pacific Web Conference (APWeb), 477--488, 2005.

[SWC+02] SHIM, J. P.; WARKENTIN, M.; COURTNEY, J. F.; POWER, D. J.; SHARDA, R.; CARLSSON, C. ***Past, present, and future of decision support technology***, Decision Support Systems, 33(2):111--126, 2002.

[TES07] ***TestComplete – Automate your tests***. Disponível em <http://www.automatedqa.com/products/testcomplete>. Último acesso em março de 2007.

[Tii05] TIIDUMAA, A. ***A Data Mart approach to analysing exam data***, Proceedings of the 11th International Conference of European University Information Systems, 2005.

[XML07] ***XMLA – XML for Analysis***. Disponível em <http://www.xmlforanalysis.com>. Último acesso em março de 2007.

Apêndice A – Consultas multidimensionais e geográficas

Este apêndice contém as declarações das consultas usadas como entradas para os testes realizados neste trabalho.

Consultas Multidimensionais

As consultas multidimensionais, que são submetidas ao servidor OLAP, foram especificadas na linguagem MDX e estão listadas no Quadro 25.

Quadro 25. Consultas multidimensionais usadas como entradas para os testes.

ID Consulta	Consulta
MD01	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, DrilldownLevel({[Localizacao].[Micro Regiao].MEMBERS}) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002], [Measures].[Populacao])
MD02	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, DrilldownMember({[Localizacao].[Micro Regiao].MEMBERS}, {[Localizacao].[Brasil].[Nordeste].[BAHIA].[SUL BAIANO].[ILHEUS-ITABUNA]}) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002], [Measures].[Populacao])
MD03	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, order({[Localizacao].[Micro Regiao].MEMBERS}, [Measures].[Populacao]) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002])
MD04	SELECT {[Measures].[Populacao]} ON COLUMNS, crossjoin({[Localizacao].[Micro Regiao].MEMBERS}, {[Clima].[Zona].members}) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002])
MD05	SELECT {[Measures].[Populacao]} ON COLUMNS, NonEmptyCrossjoin({[Localizacao].[Micro Regiao].MEMBERS}, {[Clima].[Zona].members}) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002])
MD06	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, hierarchize(DrilldownMember({[Localizacao].[Micro Regiao].MEMBERS}, {[Localizacao].[Brasil].[Nordeste].[BAHIA].[SUL BAIANO].[ILHEUS-ITABUNA]})) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002], [Measures].[Populacao])
MD07	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, non empty {[Localizacao].[Micro Regiao].MEMBERS} ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002], [Measures].[Populacao])

MD08	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, {[Localizacao].[Micro Regiao].MEMBERS} ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002], [Measures].[Populacao])
MD09	SELECT {[Tempo].[Ano].MEMBERS} ON COLUMNS FROM [BRSaude] WHERE ([Measures].[Populacao])
MD10	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, {[Localizacao].[Meso Regiao].MEMBERS} ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002], [Measures].[Populacao])
MD11	SELECT {[Clima].[Zona].MEMBERS} ON COLUMNS, {[Localizacao].[Estado].MEMBERS} ON ROWS FROM [BRSaude] WHERE ([Measures].[Populacao])
MD12	SELECT {[Tempo].[Ano].MEMBERS} ON COLUMNS FROM [BRSaude] WHERE ([Measures].[Nascidos Vivos])
MD13	SELECT {[Measures].[Populacao]} ON COLUMNS, crossjoin({[Localizacao].[Brasil].[Norte], [Localizacao].[Brasil].[Sul]}, {[Clima].[Zona].MEMBERS}) ON ROWS FROM [BRSaude] WHERE ([Tempo].[Ano].[2002])

Consultas Geográficas

As consultas geográficas, que são submetidas ao SGBD *PostgreSQL* com a extensão espacial *PostGIS*, foram especificadas em linguagem SQL compatível com o *PostGIS* e estão listadas no Quadro 26.

Quadro 26. Consultas geográficas usadas como entradas para os testes.

ID Consulta	Consulta
GEO01	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND contains(geo_clima, geo_micro_regiao)
GEO02	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND touches(geo_clima, geo_micro_regiao)
GEO03	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND crosses(geo_clima, geo_micro_regiao)
GEO04	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima

	WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND within(geo_clima, geo_micro_regiao)
GEO05	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND overlaps(geo_clima, geo_micro_regiao)
GEO06	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND intersects(geo_clima, geo_micro_regiao)
GEO07	SELECT asText(geo_micro_regiao), asText(geo_clima) FROM pgd_micro_regiao, pgd_clima WHERE geo_clima && geo_micro_regiao AND nm_zona = 'Tropical Brasil Central' AND disjoint(geo_clima, geo_micro_regiao)
GEO08	SELECT asText(geo_meso_regiao), asText(geo_clima) FROM pgd_meso_regiao, pgd_clima WHERE geo_clima && geo_meso_regiao AND nm_zona = 'Tropical Brasil Central' AND contains(geo_clima, geo_meso_regiao)
GEO09	SELECT asText(geo_unid_federacao), asText(geo_clima) FROM pgd_unidade_federacao, pgd_clima WHERE geo_clima && geo_unid_federacao AND nm_zona = 'Tropical Brasil Central' AND overlaps(geo_clima, geo_unid_federacao)