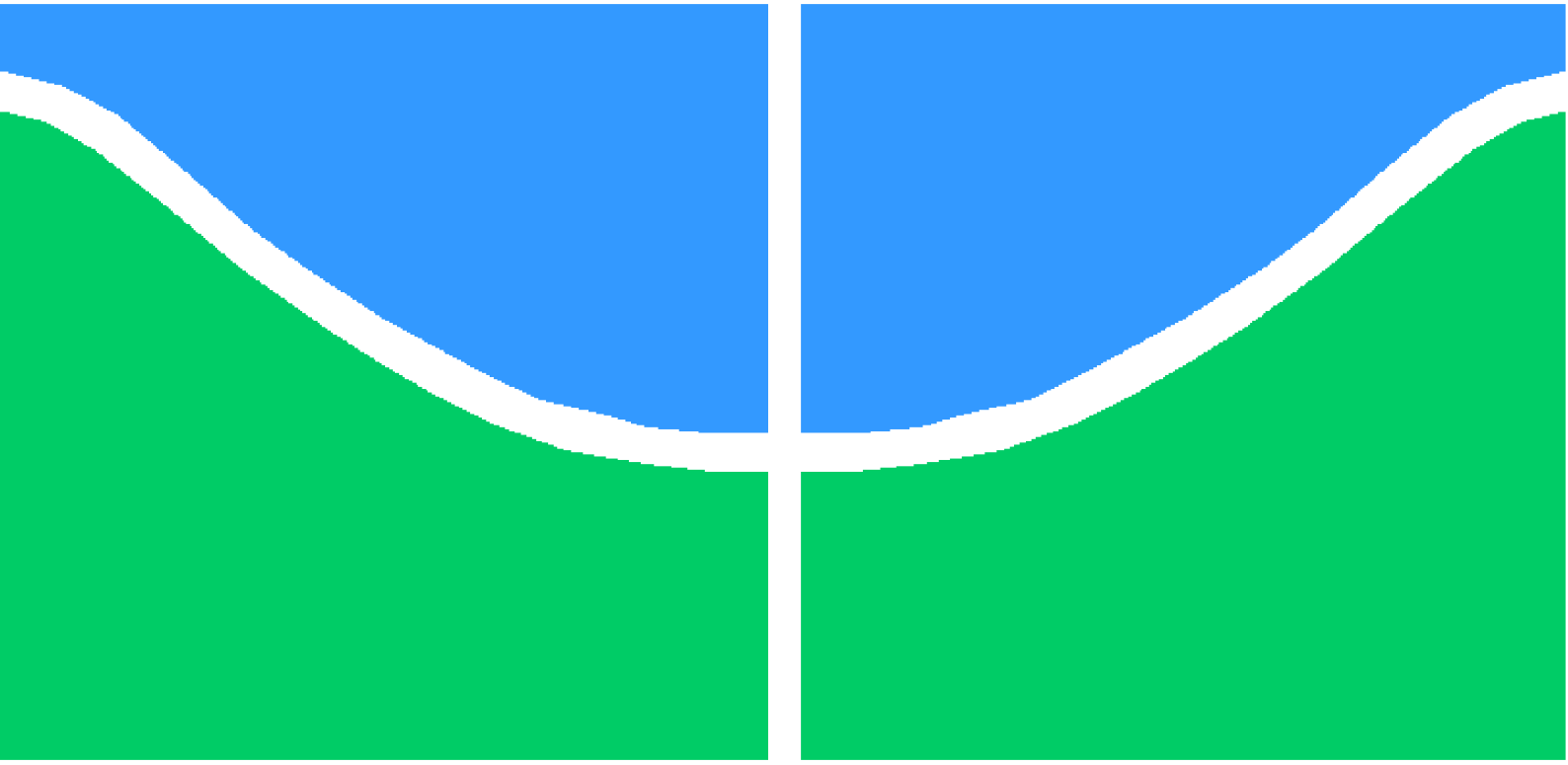

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia de Software

Estudo da Eficácia e Eficiência do Uso de um Ambiente de *Data Warehousing* para Aferição da Qualidade Interna de Software: um Estudo de Caso no TCU

Autor: Pedro da Cunha Tomioka

Orientador: Prof. Msc. Hilmer Rodrigues Neri

Brasília, DF

2014

Pedro da Cunha Tomioka

Estudo da Eficácia e Eficiência do Uso de um Ambiente de *Data Warehousing* para Aferição da Qualidade Interna de Software: um Estudo de Caso no TCU

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Msc. Hilmer Rodrigues Neri

Brasília, DF

2014

Pedro da Cunha Tomioka

Estudo da Eficácia e Eficiência do Uso de um Ambiente de *Data Warehousing* para Aferição da Qualidade Interna de Software: um Estudo de Caso no TCU/ Pedro da Cunha Tomioka. – Brasília, DF, 2014-

42 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Msc. Hilmer Rodrigues Neri

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2014.

1. Métricas de Código-Fonte. 2. *Data Warehousing*. I. Prof. Msc. Hilmer Rodrigues Neri. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Estudo da Eficácia e Eficiência do Uso de um Ambiente de *Data Warehousing* para Aferição da Qualidade Interna de Software: um Estudo de Caso no TCU

CDU 02:141:005.6

Pedro da Cunha Tomioka

Estudo da Eficácia e Eficiência do Uso de um Ambiente de *Data Warehousing* para Aferição da Qualidade Interna de Software: um Estudo de Caso no TCU

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, Ainda não se sabe:

Prof. Msc. Hilmer Rodrigues Neri
Orientador

Ainda não se sabe
Convidado 1

Ainda não se sabe
Convidado 2

Brasília, DF
2014

Este trabalho é dedicado à minha mãe, Elizabeth, e à minha irmã, Luisa.

Agradecimentos

A Universidade de Brasília pela oportunidade fazer o cruseo.

Ao professor Hilmer Rodrigues Neri pela oportunidade e apoio na elaboração deste trabalho, e também pelo ensinos e conhecimento fornecidos dentro e fora da sala de aula.

Aos meus pais pelo incentivo e preocupação por em todos estes anos.

Aos meus amigos de graduação Matheus Tristão, Nilton César Araruna e Guilherme Baufaker, pelos momentos de dedicação ao estudo e pela amizade cultivada durante o tempo do curso.

Ao meu primeiro chefe Odnalro Cruz Videira Júnior, por acreditado em mim, por ter sempre me incentivado a melhorar e por me mostrar o caminho que devo seguir para ser um bom profissional.

Por fim agradeço a meus amigos e companheiros de graduação: Pedro Henrique Potiguara, Aline Gonçalves, Hichemm Khalyd, Guilherme de Lima, Tiago Pereira, Guilherme Fay, Pedro Matias, Mayco Henrique, Carlos Filipe, Marcos Ronaldo, Rafael Ferreira, Matheus Braga, Thabata Granja, Fágner Rodrigues, Bruno Motta, Paulo Acés, Matheus Patrocínio, Greg Ouyama, Eusyar Alves, João Pedro Carvalho, Artur Potiguara, Pedro Guilherme Moreira e Pedro Inazawa.

"The Truth is Out There"
(The X-Files)

Resumo

Palavras-chaves: Métricas de Código-Fonte. *Data Warehousing*. *Data Warehouse*

Abstract

Palavras-chaves: *Source Code Metrics. Data Warehousing. Data Warehouse*

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACC	*Afferent Connections per Class*
ACCM	*Average Cyclomatic Complexity per Method*
AMLOC	*Average Method Lines of Code*
ANPM	*Average Number of Parameters per Method*
CBO	*Coupling Between Objects*
CSV	*Comma-Separated Values*
DER	Diagrama Entidade Relacionamento
DIT	*Depth of Inheritance Tree*
DW	*Data Warehouse*
ETL	*Extraction-Transformation-Load*
FTP	*File Transfer Protocol*
GQM	*Goal-Question-Metric*
IEC	*International Electrotechnical Commission*
ISO	*International Organization for Standardization*
JSON	*JavaScript Object Notation*
LCOM4	*Lack of Cohesion in Methods*
LOC	*Lines of Code*
NPA	*Number of Public Attributes*
NOC	*Number of Children*
NOM	*Number of Methods*
OLAP	*On-Line Analytical Processing*
OLTP	*Online Transaction Processing*
RFC	*Response For a Class*

SCAM	*IEEE International Working Conference on Source Code Analysis and Manipulation*
SGBD	Sistema de Gerenciamento de Bancos de Dados
XML	*Extensible Markup Language*
YAML	*YAML Ain't Markup Language*
OSS	Sistemas de Fonte de Dados Operacionais (*Operational Source Systems*)
DSA	Área de Preparação dos Dados (*Data Staging Area*)
DPA	Área de Apresentação dos dados (*Data Presentation Area*)
DAT	Ferramentas de Acesso de dados (*Data Access Tools*)

Sumário

1 Introdução

1.1 Contexto

Métricas de software fornecem uma maneira quantitativa de avaliar a qualidade de atributos internos do produto, habilitando assim o engenheiro de software a avaliar a qualidade antes de o produto ser construído (PRESSMAN, 2010).

O uso de ferramentas para a medição de código-fonte é uma realidade no contexto de desenvolvimento de software. Afinal o objetivo de tal medição é obter um produto de software de alta qualidade, algo que a comunidade de desenvolvedores de softwarem concordam mutualmente (PRESSMAN, 2010).

1.2 Problema

As ferramentas de análise estática de código-fonte, normalmente, não apresentam associação entre as medidas numéricas e a forma de interpretá-los, isto é, as métricas frequentemente mostram apenas valores numéricos isolados (MEIRELLES, 2013). Isso gera uma dificuldade em interpretar os dados disponibilizados por tais ferramentas, afetando a avaliação de projetos de software onde são um fator essencial para a tomada de decisão à nível de gerenciamento de projeto, de processo, de engenharia de software, e de outros processos de apoio (PANDIAN, 2004).

Assim Rêgo (2014) em seu trabalho propôs uma solução de DW que contribuiu na avaliação da utilização de ambientes de *Data Warehousing* em processos de medição e análise de métricas de código-fonte.

Surge então a necessidade de investigar, por meio de estudo de caso, a eficácia e eficiência da solução de *data warehousing* proposta no contexto do problema descrito. Assim foi criada a seguinte **questão geral de pesquisa**:

O uso de um ambiente de Data Warehousing para aferição e monitoramento da qualidade interna do código-fonte é eficaz e eficiente do ponto de vista da equipe de qualidade?

1.3 Objetivos

O objetivo geral do trabalho é a resolução da questão de pesquisa, sendo ele:

- **Aferir e avaliar a eficácia e eficiência da solução de data warehousing**

proposta por Rêgo (2014) quanto a qualidade interna de um produto de software.

Assim derivando-se do objetivo geral, este trabalho consiste em um conjunto de objetivos específicos divididos em dois grupos, TCC1 e TCC2.

TCC1

- Levantar a fundamentação teórica do trabalho com base no contexto do problema e adotando a solução de dw;
- Definir, projetar e caracterizar o estudo de caso.

TCC2

- Coletar os dados;
- Realizar análise dos dados coletados;
- Relatar os resultados obtidos.

1.4 Metodologia de Pesquisa

As estapas necessárias para a o desenvolvimento do projeto de estudo de caso se baseiam na diagramação de pesquisa de Gil (2008), e nos passos para elaborar um estudo de caso por Yin (2001) e no protocolo de estudo de caso de Brereton, Kitchenham e Budgen (2008). A Figura 1 apresentao os elementos do projeto com sua ordem de execucação em forma de fluxo.

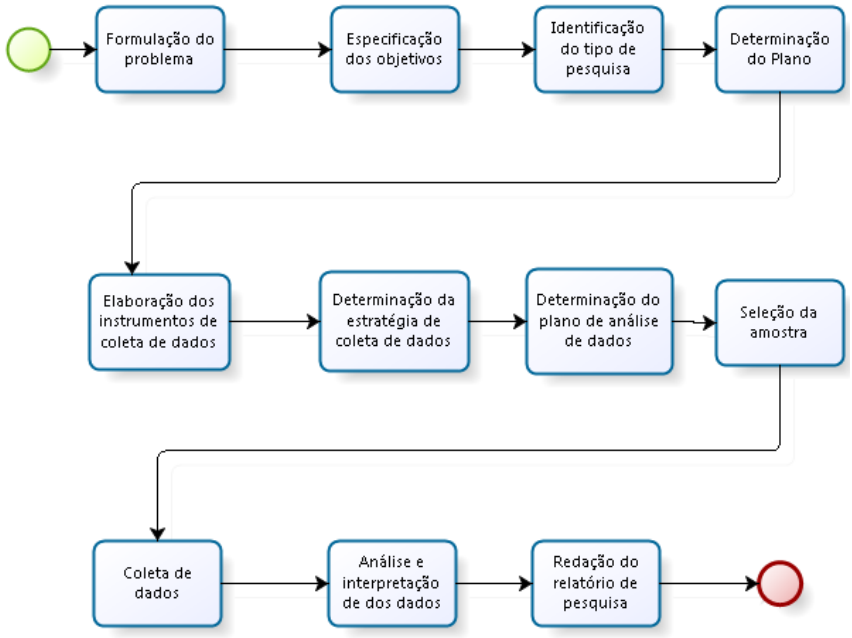

Figura 1 – Diagramação do projeto de estudo de caso

Os passos específicos do estudo de caso se apresentam após a identificação do tipo de pesquisa. Assim a partir do passo de determinação do plano até o passo de seleção da amostra, é apresentado o projeto de estudo de caso onde serão identificados elementos como o problema a ser resolvido, a questão de pesquisa, e demais tópicos definidos no protocolo de estudo de caso. Após isso a coleta de dados é feita por questionários, entrevistas informais, observações em campo, e resultados provenientes da solução de DW. Análise e interpretação dos dados se refere as atividades de categorização, exibição, verificação e conclusão de dados de natureza qualitativa e quantitava. Por fim, a redação do relatório de pesquisa apresenta a redação dos resultados de forma adequada para o leitor alvo.

1.5 Organização do Trabalho

Após a leitura desta introdução, o leitor encontrará os seguintes capítulos:

- **Capítulo 2 – Métricas**: revisão bibliográfica sobre métricas de código-fonte, definição das métricas selecionadas para o estudo e conceitos de cenários de limpeza de código-fonte.
- **Capítulo 3 – Data Warehousing**: apresentação dos conceitos de *data warehousing* e da solução de DW utilizada no estudo.
- **Capítulo 4 – Projeto de Estudo de Caso**: definição do estudo de caso e de seu protocolo.
- **Capítulo 5 – Conclusão**: as conclusões e o que será feito na segunda parte do

trabalho.

2 Métricas de Software

2.1 Classificação das métricas de software

- **Nominal:** São empregadas expressões semânticas para representar objetos para fins de identificação (PANDIAN, 2004). Na interpretação dos valores de cada atributo, a ordem não possui significado (MEIRELLES, 2013).
- **Ordinal:** os valores podem ser comparados em ordem, é possível agrupar valores em categorias também podem ser ordenadas. Porém esta escala não oferece informações sobre a maginitude da diferença entre os elementos (KAN, 2002).
- **Intervalo:** A ordem dos reultados é importante, assim como o tamanho dos intervalos que separam os pontos, mas a proporções não são necessariamente válidas (MEIRELLES, 2013). Operações matemáticas como adição e subtração podem ser aplicadas a este tipo de intervalo (KAN, 2002).
- **Racional:** possui a mesma definição da escala de intervalo, a diferença é que a proporção é preservada (MEIRELLES, 2013). Ou seja, é possível definir um unidade zero para um tamanho de unidade previamente estabelecido pela escala (KAN, 2002).

2.2 Métricas de código-fonte

2.2.1 Métrica de Tamanho e Complexidade

- **LOC** (*Lines of Code* - Número de Linhas de Código): São contadas linhas de programa que não são um comentário ou uma linha em branco (KAN, 2002).
- **ACCM** (*Average Cyclomatic Complexity per Method* - Média da Complexidade): Mede a complexidade do programa, podendo ser representado através de um grafo de fluxo de controle (MCCABE, 1976).
- **AMLOC** (*Average Method Lines of Code* - Média do número de linhas de código por método): Indica a distribuição das linhas de código entre os métodos. Quanto maior o valor, menor a distribuição. É preferível ter muitas operações pequenas e de fácil entendimento que poucas operações grandes e complexas (MEIRELLES, 2013).

2.2.2 Métricas de orientação a objeto

Rêgo (2014) selecionou o seguinte conjunto de métricas de orientação a objeto para a sua solução de DW:

- **ACC** (*Afferent Connections per Class* - Conexões Aferentes por Classe): Mede a conectividade de uma classe. A manuntenção de uma classe que apresente um grande valor para esta métrica, será mais difícil e possuirá um potencial mais alto de causar efeitos colaterais nela e em outras classes (MEIRELLES, 2013).

- **ANPM** (*Average Number of Parameters per Method* - Média do Número de Parâmetros por Método): calcula a média do número de parâmetros dos métodos de uma classe.

- **CBO** (*Coupling Between Objects* - Acoplamento entre Objetos): Calcula para uma classe o número outras classes que estão acopladas a ela (CHIDAMBER; KEMERER, 1994). Segundo Pressman (2010) é provavél que a reusabilidade de uma classe diminua à medida que o CBO aumenta, pois valores altos complicam as modificações e os testes.

- **DIT** (*Depth of Inheritance Tree* - Profundidade da Árvore de Herança): Calcula quantas ancestrais da hierarquia de uma classe pode potencialmente afeta-la. Quanto mais profunda for a árvore, maior será a complexidade de seu design, pois estarão envolvidos um número grande de métodos e classes. Observa-se também que uma classe mais abaixo da hierarquia terá um maior potencia de reuso dos métodos herdados (CHIDAMBER; KEMERER, 1994).

- **LCOM4** (*Lack of Cohesion in Methods* - Falta de Coesão entre Métodos): Proposta inicialmente por Chidamber e Kemerer (1994), LCOM calcula o número de métodos que têm acesso a um ou mais atributos de uma dada classe (PRESSMAN, 2010). Por ter recebido diversas críticas, várias alternativas foram criadas. Hitz e Montazeri (1995) definiu uma revisão desta métrica e assim elaborou uma outra versão conhecida como LCOM4. Para calcular LCOM4 de um módulo, é necessário construir um gráfico não-orientado em que os nós são os métodos e atribuos de uma classe. Para cada método, deve haver uma areste entre ele e um outro método ou variável que ele usa. O valor da LCOM4 é o número de componentes fracamente conectados nesse gráfico (MEIRELLES, 2013).

- **NOC** (*Number of Children* - Número de Filhos): Filhos são subclasses imediatamente subordinadas a uma classes. À medida que o número de filhos cresce, o reuso também cresce, porém à medida que NOC aumenta, a abstração representada pela classe pai pode ser diluída, fazendo com que alguns dos filhos não necessariamente sejam membros adequados da classe pai (PRESSMAN, 2010).

- **NOM** (*Number of Methods* - Número de Métodos): mede a quantidade de métodos presentes em uma classe. Segundo Meirelles (2013) muitos métodos em uma classe podem significar um baixo nível de reuso para ela devido a propensão de casos assim apresentarem uma menor coesão.

- **NPA** (*Number of Public Attributes* - Número de Atributos Públicos): mede o encapsulamento. O número ideal para essa métrica é zero (MEIRELLES, 2013).

- **RFC** (*Response For a Class* - Respostas para uma Classe): mede o número de métodos que podem ser executadas em resposta a uma mensagem recebida por um objeto da classe medida (KAN, 2002).

2.3 Configurações de Qualidade para Métricas de Código-Fonte

Meirelles (2013) em seu estudo avaliou a distribuição e correlação dos valores das métricas de 38 projetos de software livre, um dos critérios da escola foi o número de contribuidores ativos em seus repositórios.

Software	**Mín**	**1%**	**5%**	**10%**	**25%**	**50%**	**75%**	**90%**	**95%**	**99%**	**Máx**
Eclipse	1,0	1,0	1,0	1,0	2,0	5,0	9,0	18,0	26,0	49,7	606,0
Open JDK8	1,0	1,0	1,0	1,0	2,0	3,0	8,0	17,0	27,0	60,0	596,0
Ant	1	1	1	1	2	5	11	20	29	54	126
Checkstyle	1	1	1	1	1	3	6	11	15	26	55
Eclipse Metrics	1	1	1	1	2	4	6	12	15	22	27
Findbugs	1	1	1	1	2	3	7	15	22	48	198
GWT	1	1	1	1	2	4	9	20	29	53	385
Hudson	1	1	1	1	2	3	6	12	19	43	149
JBoss	1	1	1	1	2	3	6	11	17	40	382
Kalibro	1	1	1	1	3	5	8	12	16	26	33
Log4J	1	1	1	1	2	4	8	15	23	53	123
Netbeans	1	1	1	1	2	4	9	16	23	46	1002
Spring	1	1	1	1	2	3	7	14	21	46	122
Tomcat	1	1	1	1	2	4	10	21	35	80	300

Tabela 1 – Percentis para métrica NOM para projetos em java extraídos de Meirelles (2013)

3 *Data Warehousing*

Este capítulo apresenta algumas definições necessárias para que se obtenha uma visão geral de data warehousing e OLAP. Primeiramente é definido o que é um data warehouse e quais são as terminologias relacionadas ao assunto. Em seguida é mostrado as características específicas de um data warehouse, procurando compara-lo aos banco de dados relacionais. Os componentes de um DW são definidos e apresentados e subsequentemente a modelagem que se aplica a um dw é explicitada. Ao final é apresentado a solução de data warehousing para métricas de código-fonte e como ela foi pensada segundo os fundamentos explicados nos tópicos anteriores.

3.1 Definições e terminologia

Inmon (1992) define um data warehouse como uma coleção de dados orientada a assunto, integrada, não volátil, variável no tempo para o suporte às decisões da gerência.

Hobbs et al. (2011) afirma que data warehouse é um banco de dados que contêm dados de múltiplos sistemas que foram consolidados, integrados, agregados e estruturados de modo que que possam ser usados para apoiar o processo de análise e tomada de decisão de um negócio.

Os Data Warehouses oferecem armazenamento, funcionalidade e responsividade às consultas além das capacidades dos bancos de dados orientados à transação (ELMASRI; NAVATHE, 2011). São vários os tipos de aplicação aceitos para um data warehouse. Algumas delas serão definidas a seguir.

OLAP (online analytical processing — processamento analítico on-line): é um conjunto de princípios que fornecem uma estrutura dimensional de apoio à decisão (KIMBALL; ROSS, 2002). As ferramentas OLAP empregam as capacidades de computação distribuída para análises que requerem mais armazenamento e poder de processamento do que pode estar localizado econômica e eficientemente em um desktop individual (ELMASRI; NAVATHE, 2011). Nestas aplicações dados sumariados e históricos são mais importantes que dados atômicos (NERI, 2002).

DSS (decision-support systems — sistemas de apoio à decisão): são sistemas que ajudam os principais tomadores de decisões de uma organização com dados de nível mais alto em decisões complexas e importantes (ELMASRI; NAVATHE, 2011). Segundo Kimball (2008) os DSSs têm como objetivo tornar a informação de uma organização acessível e consistente, prover uma fonte adaptável e resiliente de informações, e garantir a segurança aos dados para assim ser um sistema base para a tomada de decisão.

OLTP (online transaction processing) — processamento on-line de transações: bancos de dados tradicionais têm suporte para o OLTP, com suas operações de inserção, atualização e exclusão e também à consulta de dados. Sharma (2011) explica que sistemas OLTP usam tabelas simples para armazenar dados, estes que são normalizados, para se reduzir a redundância ou até mesmo eliminá-los, buscando sempre a garantia de sua consistência.

3.2 Características dos data warehouses

Elmasri e Navathe (2011) diferencia um Data warehouse de uma estratégia de multibancos de dados afirmando que o primeiro possui um armazenanento em um modelo multidimensional, assim dados de múltiplas fontes são integrados e processados para tal e sendo assim contrário ao segundo, que oferece acesso a bancos de dados disjuntos e normalmente heterogêneos. A vantagem aqui se nota pela adoção de um padrão de design mais conciso, que pode levar a uma menor complexidade de implantação.

Diferentemente da maioria dos banco de dados transacionais, data warehouses costumam apoiar a análise de série temporal e tendência, ambas exigindo mais dados históricos do que geralmente é mantido nos bancos de dados transacionais (ELMASRI; NAVATHE, 2011).

Em comparação com os bancos de dados transacionais, os data warehouses são não-voláteis (ELMASRI; NAVATHE, 2011). Sendo assim, suas informações possuem a caracterísca de mudarem de uma forma muito menos frequente, portanto dificilmente seria enquadrada como uma informação em tempo real, e sim como uma de atualização periódica.

Segundo Elmasri e Navathe (2011) data warehouses são otimizados para recuperação de dados, e não para processamento de rotina, uma característica intrísica à sistemas OLTP.

Elmasri e Navathe (2011) ordena as seguintes características diferenciadoras de data warehouses, proveniente da lista original de Codd, Codd e Salley (1993) sobre OLAP:

- Visão conceitual multidimensional.
- Dimensionalidade genérica.
- Dimensões e níveis de agregação ilimitados.
- Operações irrestritas entre dimensões.
- Tratamento dinâmico de matriz esparsa.
- Arquitetura de cliente-servidor.

- Suporte para múltiplos usuários.
- Acessibilidade.
- Transparência.
- Manipulação de dados intuitiva.
- Desempenho de relatório consistente.
- Recurso de relatório flexível.

3.2.1 Componentes Básicos de um Data Warehouse

A figura 2 representa a estrutura básica de um DW segundo Kimball e Ross (2002), os componentes serão definidos a seguir.

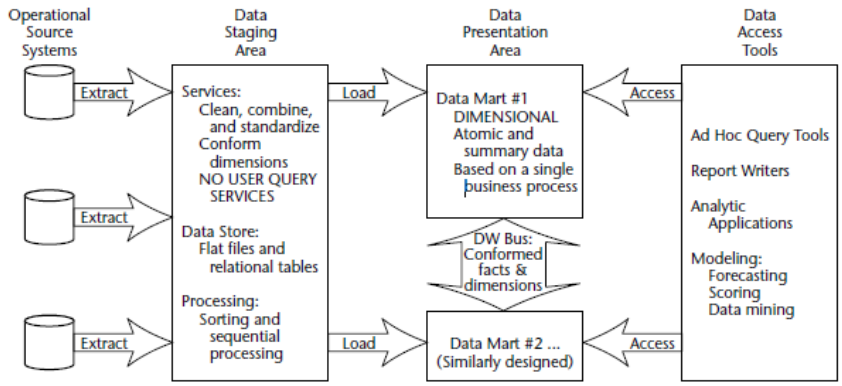

Figura 2 – Elementos básicos de um data warehouse

Sistemas de Fonte de Dados Operacionais (OSS - *Operational Source Systems*): Representa os sistemas que irão prover informações para o DW. Seus dados podem ser provenientes de outros sistemas OLTPs, planilhas e etc, que compoêm o negócio a ser tratado. Devem ser pensados como fora do data warehouse porque, temos pouco ou nenhum controle sobre o conteúdo e formato dos dados presentes nestes tipos de sistemas legados (KIMBALL; ROSS, 2002).

Área de Preparação dos Dados (DSA - *Data Staging Area*): É onde irá ocorrer a possível limpeza e reformatação dos dados antes que sejam carregados no *data warehouse* (ELMASRI; NAVATHE, 2011), processo esse conhecido como extração transformação e carga (ETL).

O passo de extração consiste em ler os dados das fontes operacionais e copiar o que for necessário para o DW na área de preparação dos dados.

Transformação é o passo que irá tratar os dados agora presente na área de preparação dos dados. Segundo Kimball e Ross (2002) existem diversos tipos de transformações

que podem ocorrer, como a limpeza dos dados (correção de erros ortográficos, resolução de dados conflitantes com o domínio, tratamento de dados que estão em falto, ou o *parse* de dados para um certo padrão), e também ações como combinar dados de várias fontes, e por fim pode-se atribuir chaves para os dados que irão para o DW.

O terceiro passo do processo de ETL, a carga, é onde os dados que foram tratados, limpo, transformados e padronizados dentro da área de preparação dos dados serão carregados nas tabelas do data warehouse.

Área de Apresentação dos dados (DPA - *Data Presentation Area*): Representa a área onde os dados são organizados, armazenados e disponibilizados para consulta direta pelos usuários, autores de relatórios, e outras aplicações. Os dados da área de apresentação devem ser dimensionais e atômicos e devem estar de acordo com a arquitetura do DW (KIMBALL; ROSS, 2002).

Ferramentas de Acesso de dados (DAT - *Data Access Tools*):

Metadados: Definido como toda a informação no ambiente de data warehouse que não são os dados em si (KIMBALL; ROSS, 2002). Os metadados num ambiente DW podem apontar para dados sobre tabelas do sistema, índices, relacionamentos, e etc. Kimball (1998) recomenda que a arquitetura de um DW seja orientada a metadados, devido a seu papel crítico de prover informações e parâmetros que permitem que as aplicações executarem suas tarefas com um controle maior sobre os dados provenientes das fontes de dados e outros elementos fundamentais para sua execução.

3.3 Modelagem de dados para data warehouses

Modelos multidimensionais tiram proveito dos relacionamentos inerentes nos dados para preencher os dados em matrizes multidimensionais, chamadas cubos de dados (ELMASRI; NAVATHE, 2011).

Uma planilha simples contendo uma matriz bidimensional poderia representar as vendas regionais por produto para um determindo período (Figura 3). Poderia então ser acrescentado a dimensão de tempo para a relação, assim teríamos o trimestre fiscal para os produtos da região. Isto produziria uma matriz tridimensional, representada por um cubo (Figura 4).

Produto	Região			
	Reg 1	Reg 2	Reg 3	...
Prod 1				
Prod 2				
Prod 3				
...				

Figura 3 – Uma matriz bidimensional

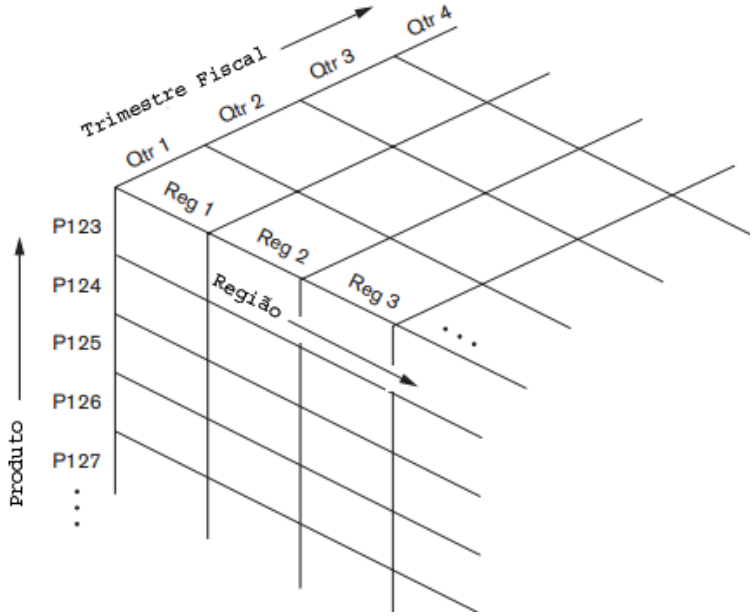

Figura 4 – Um cubo de dados tridimensiona adaptado de Elmasri e Navathe (2011)

O cubo da Figura 4 organiza os dados de produto por trimestres fiscais e por região. Cada célula representa a venda de um certo produto, num certo trimestre fiscal em uma certa região. Assim a modelagem permite adicionar mais dimensões, criando-se um hipercubo que combinaria todas as outras anteriores, o único problema é que não seria facilmente visualizada a sua forma gráfica. Porém diversas ferramentas de acesso de dados são capazes de apresentar tais dimensões de acordo com a escolha de combinação do usuário.

As operações possíveis de serem aplicadas a um cubo OLAP são:

- **Pivoting**: Também conhecido como *rotation* (rotação), nessa técnica, o cubo de dados pode ser imaginado girando para mostrar uma orientação diferente dos eixos (ELMASRI; NAVATHE, 2011).

- **Roll-up**: *Drilling* em modelagem multidimensional significa ir de um nível hierárquico a outro (BALLARD, 2006). Portanto *roll-up* ou *drill-up* consiste em subir na hierarquia, agrupando em unidades maiores ao longo de uma dimensão (ELMASRI; NAVATHE, 2011).

- **Drill-Down**: É o inverso de *roll-up*, busca alcançar a informação em um nível maior de detalhamento quanto a sua dimensão.
- **Slice**: Termo usado para definir um membro ou um grupo de membros que será separado, *Slice*, (de todas as outras dimensões) e em seguida, avaliado através de todas as outras dimensões (BALLARD, 2006).
- **Dice**: Similar ao *slice*, Ballard (2006) caracteriza que esta operação busca combinar membros de diferentes dimensões a partir da escolha de um ou mais membros de uma mesma dimensão, para que assim a interrelação entre elas possa ser analizada.

3.3.1 Tipos de tabelas do modelo multidimensional

O modelo multidimensional apresenta dois tipos de tabelas: a tabela de dimensão e fato. Uma tabela de dimensão contém os descritores textuais do negócio (KIMBALL; ROSS, 2002), ou seja, consiste em tuplas de atributos da dimensão (ELMASRI; NAVATHE, 2011). Uma tabela de fato possui tuplas para cada fato registrado. Um fato é uma medida de desempenho do negócio, tipicamente numérico e aditivo (KIMBALL; ROSS, 2002). Assim a tabela de fatos contém também as dimensões que são indicadas pelos fatos, isto é representado por ponteiros para tabelas de dimensão (um para cada variável de fato observado).

3.3.2 Esquemas multidimensionais

Dois esquemas comuns são o esquema estrela e o esquema floco de neve. O esquema estrela consiste em uma tabela de fatos com uma única tabela para cada dimensão (Figura 5) (ELMASRI; NAVATHE, 2011). O esquema floco de neve é resultado da normalização e expansão das tabelas de dimensão do esquema estrela (BALLARD, 2006) (Figura 6). Segundo Kimball e Ross (2002) o floco de neve não é recomendado em uma ambiente de data warehouse, pois quase sempre faz com que a apresentação dos dados ao usuário seja mais complexa, além do impacto negativo que causa sobre o desempenho da navegação.

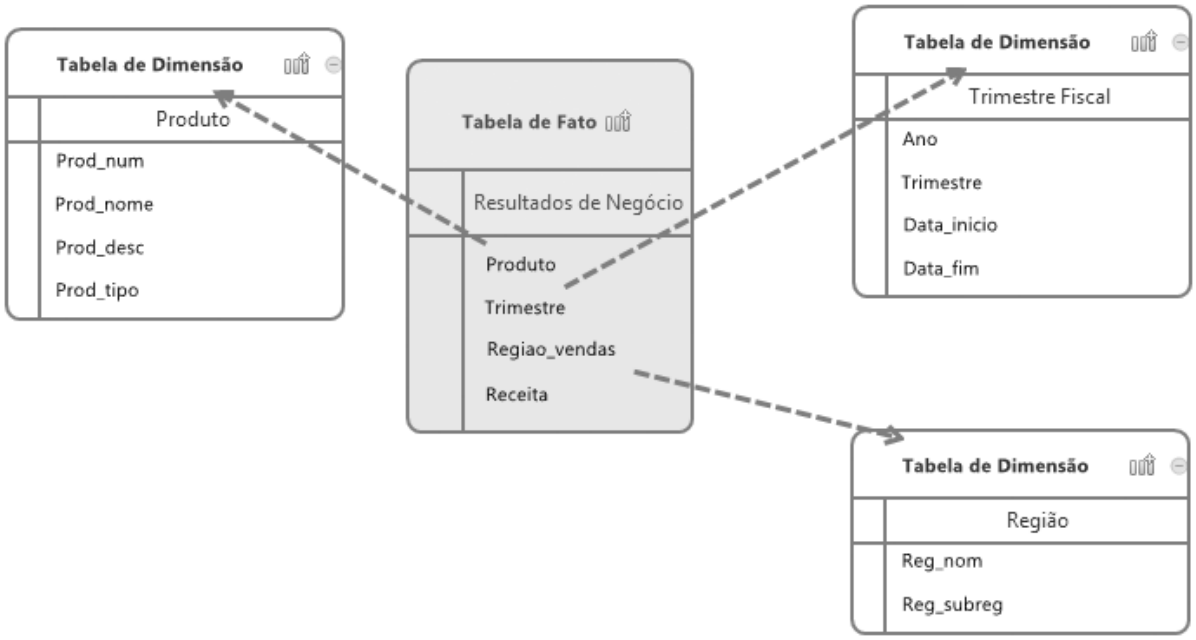

Figura 5 – Um esquema estrela

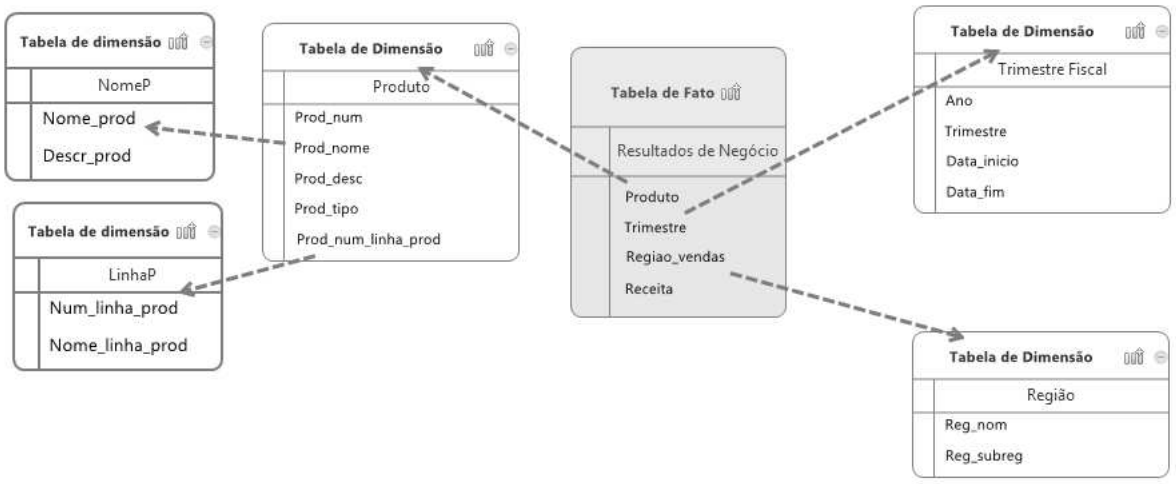

Figura 6 – Um esquema floco de neve

3.4 Solução de *Data Warehousing* para Métricas de Código-Fonte

Seguindo a metodologia de Kimball e Ross (2002) para projetar um data warehouse, Rêgo (2014) projetou uma solução de data warehousing para métricas de código-fonte seguindo quatro passos da metodologia, definidos em cada sub-seção a seguir.

3.4.1 Seleção do processo de negócio

Rêgo (2014) buscou indentificar os processos de negócio e seus respectivos requisitos. Assim foram levantados 15 requisitos de negócio, sendo que os 8 primeiros se referem a avaliação dos valores percentis das métricas de código-fonte e o restante a avaliação de cenários de limpeza e taxa de aproveitamento de oportunidades de melhoria de código-fonte. Os requisitos levantados por Rêgo (2014) foram os seguintes:

RN1: Visualizar o intervalo qualitativo obtido para cada métrica de código-fonte em uma determinada release do projeto para a configuração *Open JDK8 Metrics*.

RN2: Comparar o intervalo qualitativo obtido para cada métrica de código-fonte ao longo de todas as releases de um projeto para a configuração *Open JDK8 Metrics*.

RN3: Visualizar o o valor pecentil obtida para cada métrica de código-fonte em uma determinada release do projeto para a configuração *Open JDK8 Metrics*.

RN4: Comparar o o valor pecentil a para cada métrica de código-fonte ao longo de todas as releases para a configuração *Open JDK8 Metrics*.

RN5: Visualizar o intervalo qualitativo obtido para cada métrica de código-fonte em uma determinada release do projeto para a configuração *Tomcat Metrics*.

RN6: Comparar o intervalo qualitativo obtido para cada métrica de código-fonte ao longo de todas as releases de um projeto para a configuração *Tomcat Metrics*.

RN7: Visualizar a medida obtida para cada métrica de código-fonte em uma determinada release do projeto para a configuração *Tomcat Metrics*.

RN8: Comparar o valor percentil obtido para cada métrica de código-fonte ao longo de todas as releases para a configuração *Tomcat Metrics*.

RN9: Visualizar a quantidade de cenários de limpeza identificados por tipo de cenários de limpeza de código-fonte em cada classe ao longo de cada *release* de um projeto.

RN10: Comparar a quantidade de cenários de limpeza por tipo de cenários de limpeza de código-fonte em uma release de um projeto.

RN11: Visualizar o total de cenários de limpeza em uma determinada release de um projeto.

RN12: Visualizar cada uma das classes com um determinado cenário de limpeza de código-fonte ao longo das *releases* do projeto.

RN13: Visualizar as 10 classes de um projeto com menor número de cenários de limpeza identificados.

RN14: Visualizar as 10 classes de um projeto com maior número de cenários de limpeza identificados.

RN15: Acompanhar a Taxa de Aproveitamento de Oportunidades de Melhoria de Código-Fonte que é a divisão do total de cenários de limpeza identificados em uma release e o o número total de classes da mesmarelease de um projeto.

3.4.2 Verificação da periodicidade de coleta de dados

A identificação da periodicidade de coleta dos dados é essencial para que esta seja realizada de maneira correta, além de viabilizar a agregação dos dados em níveis ou hierarquias (RêGO, 2014). Assim a peridicidade foi identificada como as *releases* do software.

3.4.3 Identificação dos fatos e das dimensões

Rêgo (2014) identificou os fatos e dimensões a partir dos termos mais relevantes dos requisitos de negócios levantados. O relacionamento entre fato, natureza do fato e dimensão se mostra na Tabela 2.

Fato	Natureza do Fato	Dimensões
Valor Percentil	Não Aditivo	• Projeto • Métrica • Configuração • Qualidade • *Release* • Tempo
Quantidade de Cenários de Limpeza	Aditivo	• Projeto • Cenário de Limpeza • Classe • *Release*
Taxa de Aproveitamento de Oportunidade de Melhoria de Código-Fonte	Não Aditivo	• Projeto • *Release*

Tabela 2 – Fatos e Dimensões do *Projeto de Data Warehouse* por Rêgo (2014)

Assim Rêgo (2014) elaborou o modelo físico para o data warehouse:

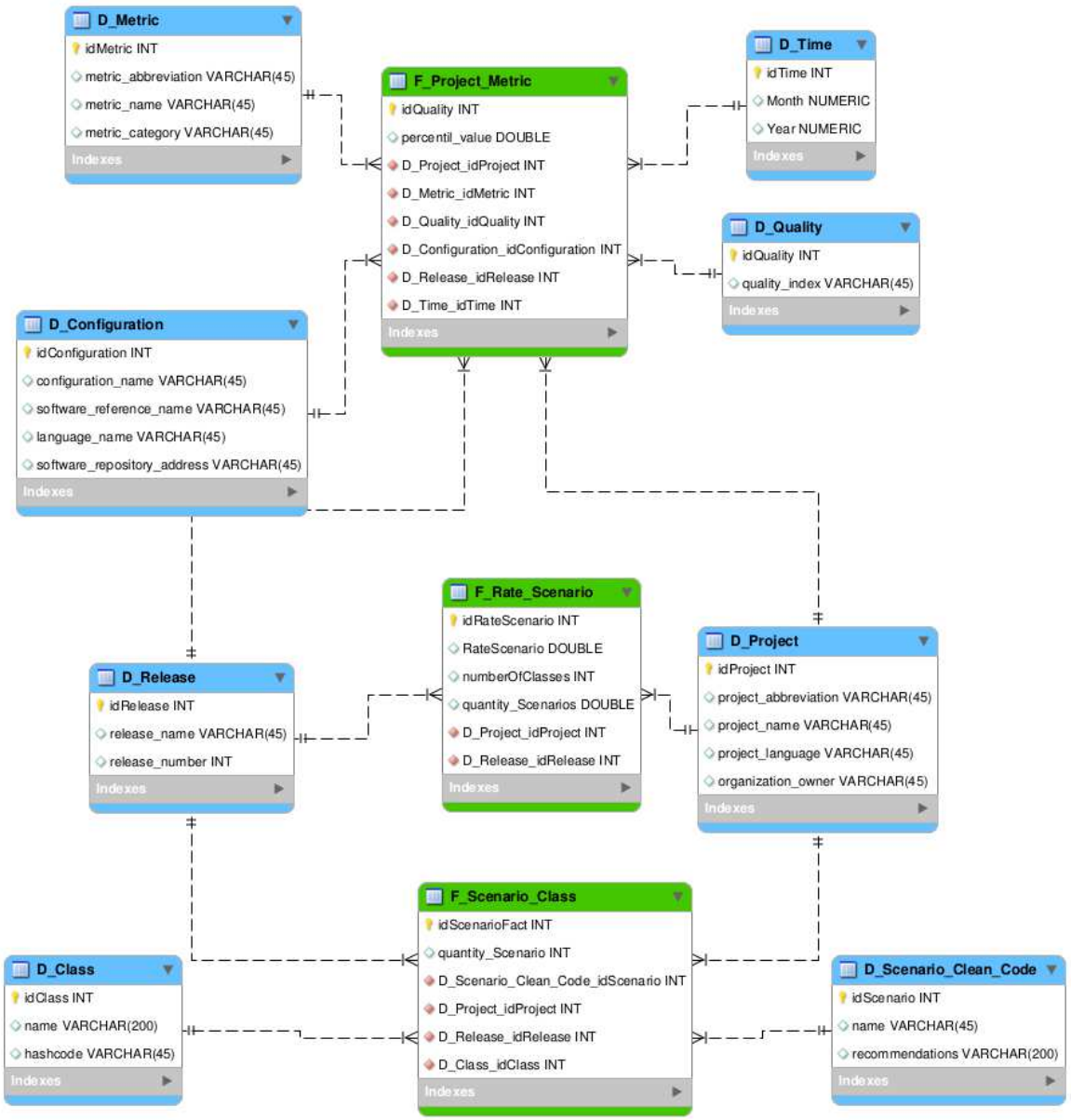

Figura 7 – Projeto Físico do *Data Warehouse* por Rêgo (2014)

Rêgo (2014) criou uma área de metadados para facilitar o processo de ETL. A Figura 8 mostra a modelagem dos metadados com suas chaves, atributos e relacionamentos. A tabela "Meta_Metric_Ranges" contém os intervalos qualitativos para cada um das métricas de código-fonte. A tabela "Meta_Scenario" contém os cenários de limpeza do código-fonte e também as recomendações de limpeza para cada cenário de limpeza previsto.

A tabela "Meta_Metric_Ranges_Meta_Scenario" possui duas chaves da tabela "Meta_Metric_Ranges", pois Rêgo (2014) observou a que cada cenário de limpeza identificado deve conter dois intervalos qualitativos de métricas de código-fonte.

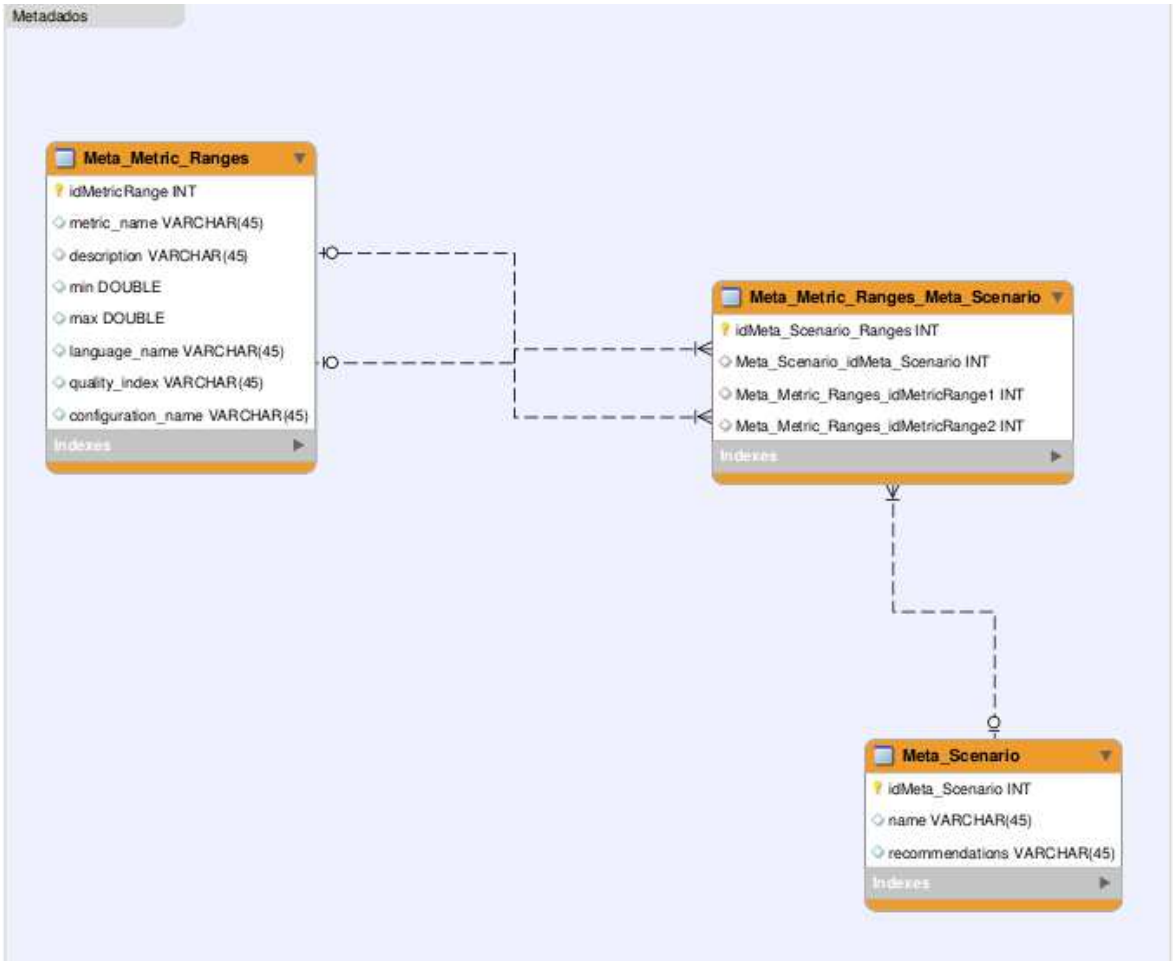

Figura 8 – Metadados do Data Warehouse por Rêgo (2014)

3.5 Considerações finais do Capítulo

Nesse capítulo foi apresentado os conceitos de *data warehousing* que foram a base teórica para a elaboração da solução de DW de Rêgo (2014). No próximo capítulo será apresentado o projeto de estudo de caso com seu protocolo, apresentando a questão de pesquisa motivadora do estudo.

4 Projeto de Estudo de Caso

Este capítulo apresenta o protocolo do estudo de caso a ser realizado. A sua definição e os requisitos necessários para sua execução são identificados e explicados.

4.1 Definição

Um estudo de caso consiste no estudo profundo e exaustivo de um ou poucos objetos, de maneira que permita seu amplo e detalhado conhecimento (GIL, 2008), assim representa uma maneira de se investigar um tópico empírico seguindo-se um conjunto de procedimentos pré-especificados (YIN, 2001). Uma vantagem dos estudos de caso é que eles são mais fáceis de planejar e são mais realistas, mas as desvantagens são que os resultados são difíceis de generalizar e mais difícil de interpretar , ou seja, é possível mostrar os efeitos de uma situação típica, mas requer mais análise para que se possa generalizar a outras situações (WOHLIN, 2012).

A estrutura do protocolo de estudo de caso prosposta se baseia em Brereton, Kitchenham e Budgen (2008) e se apresenta dividido nos seguintes tópicos, cada um com seu objetivo específico:

- **Background - Seção 4.2**: Identificar outras estudos acerca do tópico, define a questão de pesquisa principal e suas proposições derivadas que serão abordado por este estudo.
- **Design - Seção 4.3**: Identificar se é um caso único ou múltiplo e seu propósito geral.
- **Seleção - Seção 4.4**: Critérios para a seleção do caso e descriçao do objeto de estudo.
- **Fonte e Método de Coleta de Dados - Seção 4.5**: Identificar os dados que serão coletados, definindo um plano para a coleta e como a informação será armazenada.
- **Análise - Seção 4.6**: Identificar os critérios para interpretação dos resultados do estudo de caso, relacionar os dados com a questão de pesquisa e elaborar a explicação do encontrado.
- **Validade - Seção 4.7**: Se baseia nos tipos de validades aplicáveis a um estudo de caso por Yin (2001), sendo elas: constructo, interna, externa e confiabiliade.

4.2 Background

4.2.1 Outros Trabalhos

O uso do monitoramento de métricas de código-fonte num ambiente de data warehousing foi proposto e desenvolvido no trabalho de Rêgo (2014).

4.2.2 Questão de Pesquisa

A questão de pesquisa deriva do problema identificado, e busca definir o propósito da medição, o objeto que será medido, a questão a se tratar, e o ponto de vista que se delimita a questao (BASILI; CALDIERA; ROMBACH, 1996).

Para responder a questão de pesquisa foi construído um GQM (*Goal-Question-Metric*) (BASILI; CALDIERA; ROMBACH, 1996) representado na Figura 9. O GQM possui objetivos específicos para a questão de pesquisa, cada um tem suas questão específicas que busca responder com a coleta de métricas a questão maior de pesquisa e assim atender o problema do estudo de caso.

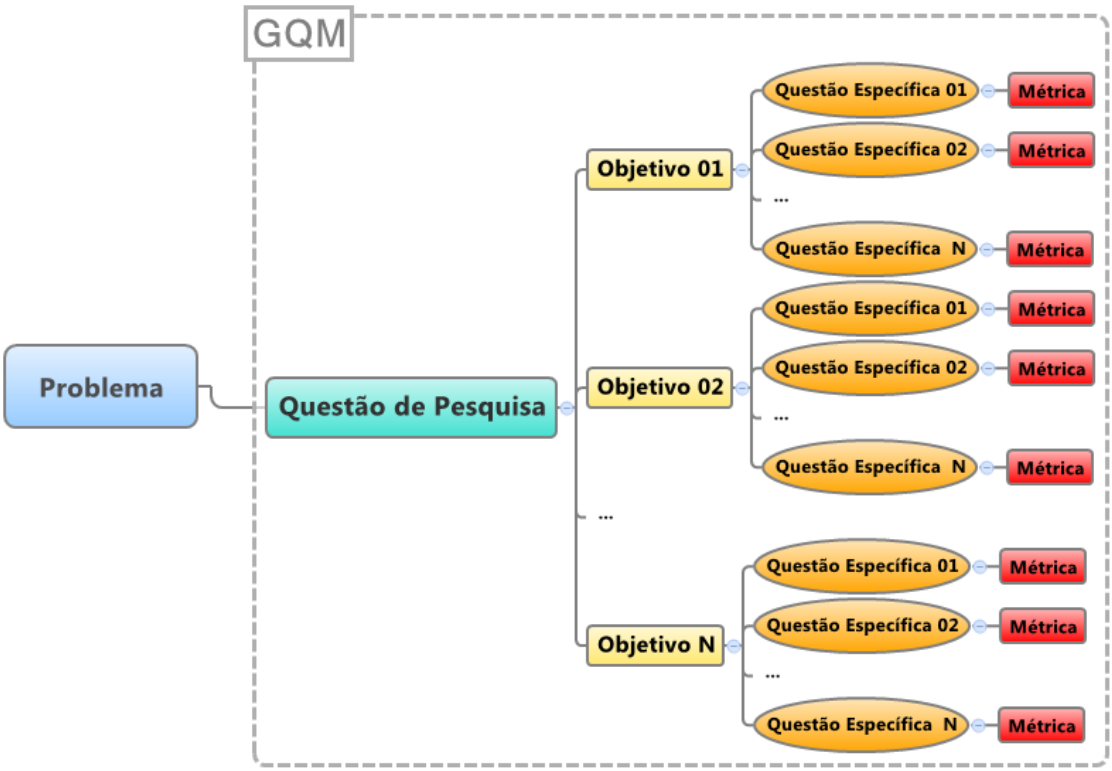

Figura 9 – Estrutura do Estudo de Caso

O problema se refere ao problema de pesquisa identificado. A questão de pesquisa se refere a questão geral de pesquisa. Assim foi definido um objetivo para a questão de pesquisa, e para ele foram levantadas nove questões. A caracterização de cada item está a seguir.

Questão de Pesquisa: ***O uso de um ambiente de Data Warehousing para aferição e monitoramento da qualidade interna do código-fonte é eficaz e eficiente do ponto de vista da equipe de qualidade?***

Objetivo 01: Avaliar a eficácia e eficiência da solução de *Data Warehousing* do ponto de vista da <equipe de qualidede> no contexto de desenvolvimento de software.

Questão 01: Quantas tomadas de decisão foram realizadas pela equipe baseado no uso da solução?

Fonte: Observação em campo.

Métrica: número de decisões tomadas.

Questão 02: Quantas tomadas de decisão foram realizadas pela equipe baseado no uso da solução em um <período de tempo>?

Fonte: Observação em campo (áudio/vídeo).

Métrica: número de decisões tomadas/tempo.

Questão 03: Qual a avaliação da equipe de qualidade quanto a detecção de cenarios de limpeza de codigo?

Fonte: Questionário com a equipe de qualidade.

Métrica: muito bom, bom, regular, ruim, muito ruim.

Questão 04: Com que frequência a equipe de qualidade encontra falhas relacionados à utilização da ferramenta?

Fonte: Observação em campo (áudio/vídeo).

Métrica: quantidade de falhas.

Questão 05: Com que frequência a equipe de qualidade encontra falhas relacionados à utilização da ferramenta por tempo?

Fonte: Registro de Observação em campo (áudio/vídeo).

Métrica: quantidade de falhas / tempo.

Interpretação do valor da métrica: Quanto mais próximo de zero melhor. X >= 0.

Questão 06: Qual a proporção de uso da ferramenta para a tomada de decisões?

Fonte: Questionário com a equipe de qualidade.

Métrica: número de decisões tomadas / número de vezes que a solução foi usada

Questão 07: Qual a quantidade de cenários que foram corrigidos após utilização da solução?

Fonte: Código fonte.

Métrica: números de cenários corrigidos por release / número de cenários encontrados por release.

Questão 08: Qual o nível de satisfação do uso da solucão em comparação à solução anterior?

Fonte: Equipe de Qualidade.

Métrica: muito satisfeito, satisfeito, neutro, insatisfeito, muito insatisfeito.

Questão 09: Qual a taxa de oportunidade de melhoria de código da solução em uma <intervalo de tempo (sprint, release)>?

Fonte: Código-fonte.

Métrica: A Taxa de Aproveitamento de Oportunidades de Melhoria de Código, em uma release, é calculada como:

$$T_r = \frac{\sum_{i=1}^{n} Ce_i}{\sum_{i=1}^{n} Cl_i}$$

onde Ce é o total de cenários de limpezas identificados e Cl é total de classes em uma release.

Interpretação do valor da métrica:

- **Número de classes crescente e constante, número de oportunidade de melhoria estável:** Projeto cresceu mais dos que o cenários, cenários podem ou não estar sendo tratados.
- **Número de classes crescente e constante, número de oportunidade de melhoria crescendo:** Projeto cresce junto com cenários que não são tratados com eficiência ou não são tratados
- **Número de classes crescente ou não, mas constante, número de oportunidade de melhoria diminuindo:** Projeto cresce e cenários são tratados com eficiência.

4.3 Design

Stake (1995) Identifica três modalidades de estudo de caso: intríseco, instrumental e coletivo. Dentre essas definições, este estudo de caso se caracteriza com instrumental, pois se busca um entendimento geral sobre o problema de pesquisa a partir do estudo de um caso particular. Tal característica se assemelha ao tipo de estudo de caso definido como exploratório por Yin (2001), onde a visão acurada de um caso particular poderá fornecer uma visão geral do problema considerado.

A justificativa para este estudo de um caso particular se dá pelo motivo de que o tema abordado foi pouco explorado até o momento. Assim espera-se que este caso único componha um estudo de múltiplos casos, pois segundo Gil (2008) esta abordagem possa inserir as evidências coletadas aqui em diferentes contextos, com o intuito de elaborar uma pesquisa de melhor qualidade.

4.4 Seleção

A organização selecionada para este estudo de caso foi o TCU. O software que terá seu código-fonte analisado foi desenvolvido segundo a metodologia , com uma equipe de .

4.5 Fonte e Método de Coleta de Dados

Os dados serão coletados no órgão por meio de entrevistas informais, questionários, análise de documentos da organização, observação em campo e do código-fonte do software disponibilizado. Toda a informação coletada será categorizada e armazenada em meios eletrônicos.

Os questionários serão construidos a partir das questões específicas do processo levantado neste estudo de caso.

4.6 Análise

O processo de análise é composto pelas seguintes etapas:

- **Categorização:** Organização dos dados em duas categorias: Qualitativos e quantitativos. Os dados qualitativos referem-se aos questionários realizados. Os dados quantitativos, por sua vez, referem-se aos valores numéricos da solução de DW para monitoramento de métricas

- **Exibição:** Consiste na organização dos dados coletados para serem exibidos através de gráficos, tabelas e texto para poderem ser analisados.

- **Verificação:** Atestar padrões, tendências e aspectos específicos dos significados dos dados. Procurando assim gerar uma discussão e interpretação de cada dado exibido.

- **Conclusão:** Agrupamento dos resultados mais relevantes das discussões e interpretações dos dados anteriormente apresentados.

4.7 Validade

Para se julgar a qualidade de um projeto de pesquisa, Yin (2001) propoem quatro testes que devem ser considerados ao longo das fases de um estudo de caso, sendo eles:

- **Validade do Constructo:** Está presente na fase de coleta de dados onde deve ser evidenciado as múltiplas fontes de evidência e a coleta de um conjunto de métricas para que se possa saber exatamente o que medir e quais dados são relevantes para o estudo, de forma a responder as questões de pesquisa (YIN, 2001) buscou garantir a validade de construção ao se definir objetivos com evidências diferentes. Estas por sua vez estão diretamente relacionadas com os objetivos do estudo de caso e os objetivos do trabalho

- **Validade interna:** Para Yin (2001), o uso de várias fontes de dados e métodos de coleta permite a triangulação, uma técnica para confirmar se os resultados de diversas fontes e de diversos métodos convergem. Dessa forma é possível aumentar a validade interna do estudo e aumentar a força das conclusões. A triangulação de dados se deu pelo resultado da solução de DW que utiliza o código-fonte e foi explicada no capítulo 3, de base de documentos, de questionários e de entrevistas para coleta de dados. A triangulação de métodos ocorreu pelo uso de métodos de coleta quantitativos e qualitativos

- **Validade externa:** Por este ser um caso único a generalização do estudo de caso se dá de maneira pobre (YIN, 2001), assim é necessário a utilização do estudo em múltiplos casos para que se comprove a generalidade dos resultados. O fato deste trabalho ser o primeiro a utilizar a solução para o estudo de caso no órgão, faz com que não seja possível correlacionar os resultados obtidos a nenhum outro estudo.

- **Confiabilidade:** Sobre isso, Yin (2001) associa à repetibilidade, desde que seja usada a mesma fonte de dados. Nesse trabalho o protocolo de estudo de caso apresentado nessa seção garante a repetibilidade desse trabalho e consequentemente a validade relacionada a confiabilidade

4.8 Considerações finais do Capítulo

Nesse capítulo foi apresentado o projeto de estudo de caso com a definição de todos os itens de seu protocolo. A questão de pesquisa procurou responder as questões de eficácia e eficiência da solução de DW. No próximo capítulo será apresentado as conclusão do trabalho, assim como suas contribuições e a proposta de trabalhos futuros.

5 Conclusão

Referências

BALLARD, C. (Ed.). *Dimensional modeling: in a business intelligence environment*. 1st ed. ed. San Jose, Calif.?: IBM International Technical Support Organization, 2006. (IBM redbooks). ISBN 0738496448. Citado 2 vezes nas páginas 26 e 27.

BASILI, V. R.; CALDIERA, G.; ROMBACH, H. D. *The Goal Question Metric Approach*. [S.l.]: Encyclopedia of Software Engineering, 1996. Citado na página 34.

BRERETON, P.; KITCHENHAM, B.; BUDGEN, D. Using a protocol template for case study planning. In: *Proceedings of EASE 2008*. [S.l.]: BCS-eWiC, 2008. Citado 2 vezes nas páginas 16 e 33.

CHIDAMBER, S. R.; KEMERER, C. F. A metrics suite for object oriented design. *IEEE Trans. Softw. Eng.*, IEEE Press, Piscataway, NJ, USA, v. 20, n. 6, p. 476–493, jun. 1994. ISSN 0098-5589. Disponível em: <http://dx.doi.org/10.1109/32.295895>. Citado na página 20.

CODD, E.; CODD, S.; SALLEY, C. *Providing OLAP (On-line Analytical Processing) to User-analysts: An IT Mandate*. [S.l.]: Codd & Associates, 1993. Citado na página 23.

ELMASRI, R.; NAVATHE, S. B. *Sistemas de banco de dados*. Sao Paulo (SP): Pearson Addison Wesley, 2011. Citado 7 vezes nas páginas 9, 22, 23, 24, 25, 26 e 27.

GIL, A. C. *Como elaborar projetos de pesquisa*. Sao Paulo: Atlas, 2008. Citado 3 vezes nas páginas 16, 33 e 37.

HITZ, M.; MONTAZERI, B. Measuring Coupling and Cohesion in Object-Oriented Systems. In: *Proceedings of International Symposium on Applied Corporate Computing*. [S.l.: s.n.], 1995. Citado na página 20.

HOBBS, L. et al. *Oracle 10g Data Warehousing*. Elsevier Science, 2011. (Oracle9iR2 Data Warehousing Series). Disponível em: <http://books.google.com.br/books?id=EzpC7cuOGqUC>. Citado na página 22.

INMON, W. H. *Building the Data Warehouse*. New York, NY, USA: John Wiley & Sons, Inc., 1992. Citado na página 22.

KAN, S. H. *Metrics and models in software quality engineering*. [S.l.]: Addison Wesley, 2002. Citado 2 vezes nas páginas 19 e 21.

KIMBALL, R. *The data warehouse lifecycle toolkit: expert methods for designing, developing, and deploying data warehouses*. [S.l.]: Wiley. com, 1998. Citado na página 25.

KIMBALL, R. (Ed.). *The data warehouse lifecycle toolkit*. 2nd ed. ed. Indianapolis, IN: Wiley Pub, 2008. Citado na página 22.

KIMBALL, R.; ROSS, M. *The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling*. 2nd. ed. New York, NY, USA: John Wiley & Sons, Inc., 2002. Citado 5 vezes nas páginas 22, 24, 25, 27 e 28.

MCCABE, T. J. A Complexity Measure. *IEEE Transactions Software Engineering*, v. 2, n. 4, p. 308–320, December 1976. Citado na página 19.

MEIRELLES, P. R. M. *Monitoramento de métricas de código-fonte em projetos de software livre*. Tese (Doutorado) — Instituto de Matemática e Estátistica – Universidade de São Paulo (IME/USP), 2013. Citado 5 vezes nas páginas 10, 15, 19, 20 e 21.

NERI, H. R. *Análise, Projeto e Implementação de um Esquema MOLAP de Data Warehouse utilizando SGBD-OR Oracle 8.1*. Universidade Federal da Paraíba - UFPB: [s.n.], 2002. Citado na página 22.

PANDIAN, C. R. *Software metrics: a guide to planning, analysis, and application*. Boca Raton, Fla: Auerbach Publications, 2004. ISBN 0849316618. Citado 2 vezes nas páginas 15 e 19.

PRESSMAN, R. S. *Engenharia de software*. Porto Alegre (RS): AMGH, 2010. Citado 2 vezes nas páginas 15 e 20.

RêGO, G. B. Monitoramento dea métricas de código-fonte com suporte de um ambiente de data warehousing: um estudo de caso em uma autarquia da administração pública federal. 2014. Disponível em: <http://bdm.unb.br/handle/10483/8069>. Citado 11 vezes nas páginas 9, 10, 15, 16, 20, 28, 29, 30, 31, 32 e 34.

SHARMA, N. *Getting started with data warehousing*. [S.l.]: IBM Redbooks, 2011. Citado na página 23.

STAKE, R. E. *The art of case study research*. Thousand Oaks: Sage Publications, 1995. ISBN 0803957661. Citado na página 37.

WOHLIN, C. *Experimentation in software engineering*. New York: Springer, 2012. Citado na página 33.

YIN, R. K. *Estudo de caso: planejamento e metodos*. Porto Alegre: Bookman, 2001. Citado 4 vezes nas páginas 16, 33, 37 e 38.